



MSc Thesis

Predicting track quality using structural elements and relative track geometry data measured by RILA: A Machine Learning Classification approach

Karim El Laham
FUGRO/TU DELFT



Predicting track quality using structural elements and relative track geometry data measured by RILA: A Machine Learning Classification approach

Graduation Thesis

University	Delft University of Technology
Faculty	Civil Engineering and Geosciences (CiTG)
Master's Program	MSc Structural Engineering: Road and Railway Specialization
Annotation	Railway Systems

Author

Name	Karim El Laham
Student Number	5061253
Contact Details	karimlaham1997@gmail.com

Graduation Committee

Chairman, TU Delft	Dr. V.L. Markine
1 st Member, TU Delft	Prof.dr.ir. R.P.B.J. Dollevoet
2 nd Member, PDEng	Jeroen Wegdam
3 rd Member, TU Delft	Prof.dr. R.M.P. (Rob) Goverde
4 th Member, Fugro	Dr. Neda Sepasian
5 th Member, ProRail	Dr. Haoyu Wang

Abstract

Track quality measures differ from standard to standard. Widely used indices for track quality are the standard deviations of the rail longitudinal level data and the standard deviations of alignment data. Researchers have opted for various learning methods as to relate track quality to different parameters. Some research has gone to the point of using track quality interchangeably with the notion of defect density. What is common in all researches is that a track quality index is used as label for predictions. The differences are inherent to the variables/features that were chosen to conduct such predictions.

With the plethora of data at the hands of asset managers nowadays, it has become easier to opt for these learning methods. Machine Learning classification is a method wherein a number of features are used to classify a datapoint into a label. With thorough learning, by inputting the features, the label would then be predicted.

The features of interest in this project are structural elements describing the track sections. This can go from the sleeper type and subsoil type all the way to the distance from the nearest switch or bridge. This will allow for the relationships between the structural elements and the track's quality to be bolstered all the while creating a predictor based on data that need not be measured. The research question is thus formulated as follows:

How to relate track quality in a section to its surrounding structural elements and form a predictor out of it?

1. What structural elements are to be studied?
2. How to automate the generation of the features?
3. Which classifier algorithm yields the most accurate predictions?
4. Which structural features are the most important in predicting track quality?

The research question is divided into 4 sub-research questions each pertaining to a part of the research.

1. Decide on which structural elements are the most relevant for the study. Some elements may be redundant with respect to other elements. For instance, subsoil type and ballast settlement could form a redundancy. This will be decided on from the literature. This part will contribute to a section in the thesis in which a strong relation will be drawn with respect to the courses taken and studied at the TU Delft.
2. Data Gathering is a huge part of this research. Input data will be from Fugro's relative track geometry data and KRDZ data. In addition, ProRail's informatieportaal, which is a website containing all the railway assets and their locations on the map will also be of huge aid to gather the structural elements data. The answer to this sub-research question lies in automating the structural feature generation, making them ready for learning.
3. Running the Machine Learning classification code on sections of a real-life track and comparing the accuracies of each classification algorithm as to come up with recommendations for future work.
4. Some machine learning algorithms, namely the random forest classifier have in their output an importance of features measure. This will give us insight on which structural elements affect track quality the most, which would allow for better resource allocation in the long run.

In the goal of answering all the research questions and to realize the research objective, an outline of the research is included in this report along with a workflow diagram.

The project has shown that predicting track quality using structural elements and relative track geometry can be done using classification methods, on the condition that the predicted track and the trained track have similar feature importance values, and that structural features that are displayed in the track to predict are also available in the trained model.

Preface

This master's thesis was conducted amidst the COVID-19 pandemic. Like never before, mental health became more and more important, and relevant. I realized with this project, that the key to work on anything is to be passionate about it and enjoy it, or every single hurdle and barrier will seem like the end of the road, especially during a pandemic that has obliged everyone to work from home. And that is exactly what this project represented for me: it was a musical piece, with the report being the lyrics and the results representing the chorus. Which brings me to thanking the producers of this piece, my committee members who pushed me and guided me from every single standpoint. A special thank you goes to Dr. Valeri Markine, who has been a mentor to a certain extent, guiding me in every decision that I have had to make. Dr. Haoyu Wang and Dr. Neda Sepasian were the curators of my progress, and the biggest supporters of my work, and to them I am also superbly grateful. Finally, I would like to thank Fugro for believing in me and allowing me to launch an ambitious project such as this one.

Karim El Laham,

15/6/2021

Contents

Abstract	iv
Preface	vi
1. Introduction	1
1.1 Background	1
1.2 Problem Definition	2
1.3 Research Objective	3
1.4 Research Question and Sub-Questions	3
1.5 Research Relevance	4
1.5.1 Academic Relevance	4
1.5.2 Industrial Relevance	4
1.6 Research Design	4
1.6.1 Research Scope	4
1.7 Methodology	5
1.8 Results	8
1.9 Limitations	9
2 Literature Review	10
2.1 The Railway Track	10
2.1.1 Superstructure elements	11
2.1.2 Substructure elements	13
2.2 Relative Track Geometry	14
2.2.1 Longitudinal Level	16
2.2.2 Alignment	16
2.3 Track Quality Parameters	18
2.4 Factors affecting track quality	23
2.4.1 Forces on the track	23
2.4.2 Most important track defects	25
2.4.3 Factors hindering track quality	26
2.5 Machine Learning	28
2.5.1 Regression	29
2.5.2 Dimension Reduction	29
2.5.3 Classification	29
3 Data Gathering and Automation	33
3.1 Data Used	33
3.2 ProRail Rail Branch Naming System	34
3.3 Methodology	34

4	Machine Learning Classification	43
4.1	Pre-Processing	43
4.2	Exploratory Statistics.....	45
4.3	Questions to answer with the model	47
5	Results and Discussion	48
5.1	Performance Measures	48
5.2	Results: Performance Measures	50
5.3	Results: Feature Importance	53
5.4	Hyperparameter Tuning and Model Saving.....	61
5.5	Inference.....	64
5.5.1	The second way test	64
5.5.2	The last piece of track test	66
5.5.3	Different subsoil type test	68
6	Limitations, Conclusions and Recommendations.....	76
6.1	Research Limitations	76
6.2	Conclusions.....	77
6.3	Recommendations for further research.....	80
	Appendix A: Utrecht-Den Bosch Project Sample Data	81
	Appendix B: Data Gathering Pseudo-Codes and Output	82
	Appendix B.1: Table showing the linked data using NNJoin	82
	Appendix B.2.1: Spoortak_Sectioning Pseudo-Code	83
	Appendix B.2.2: Spoortak_Sectioning Output Table (Utrecht-Den Bosch).....	84
	Appendix B.3.1: Physical Object Linking Pseudo-Codes	85
	Appendix B.3.2: Physical Object Linking Table (Utrecht-Den Bosch)	87
	Appendix B.4.1: Section Cleaning Pseudo-Code	88
	Appendix B.4.2: Section Cleaning Output Table (Utrecht-Den Bosch)	90
	Appendix B.5.1: Switches Adding Pseudo-Code.....	91
	Appendix B.5.2: Switches Adding Output Table (Utrecht-Den Bosch)	93
	Appendix B.6.1: Section Thresholding Pseudo-Code.....	94
	Appendix B.6.2: Section Thresholding Output Table (Utrecht-Den Bosch)	97
	Appendix C: Machine Learning Classification Model Performance Results	98
	Appendix C.1: Performance results of varied track quality aggregation methods	98
	Appendix C.2: Performance results of varied track section length used	100
	Appendix C.3: Performance results with the two most important features omitted	102
	Appendix C.4: Performance results with all the un-important features omitted	103
	Appendix C.5: Hyperparameter tuning pseudo-code.....	civ

Appendix C.6: Hyperparameter tuning performance measures and CV accuracies	cvi
Appendix D: Model Inference Module	cix
Appendix D.1: Second Way Inference Performance Results	cix
Appendix D.2: Last Piece of Track Inference Performance Results	cx
Appendix D.3: Section Thresholding Output Table (Den Bosch Eindhoven Track)	cxii
Appendix D.4: Den Bosch Eindhoven Track Prediction Performance Results	cxiii
References	cxiv

Figures

Figure 1: Workflow diagram for research work	6
Figure 2: Schematic of Railway Track Structure (Lopes & Papaelias, 2017)	10
Figure 3: Conventional railway cross-section	11
Figure 4: Monoblock sleeper on the left (https://ballast.ir/sleeper/) and Duoblock sleeper on the right (http://www.rail-fastener.com/concrete-sleeper.html)	11
Figure 5: Ballastless track on the left in a Taiwanese High Speed Line vs a ballasted track on the right	12
Figure 6: Fastener: "Fastclip"	13
Figure 7: RiLA train-borne system and its components (Wang et al., 2019)	15
Figure 8: Longitudinal level/top definition, EN13848-1	16
Figure 9: Alignment definition, EN13848-1	17
Figure 10: Longitudinal Level D1 track quality classification EN13848-6	20
Figure 11: Alignment in D1 track quality classification EN13848-6.....	20
Figure 12: Possible applications of choosing the TQIref value based on the method, EN13848-6 page 20.....	21
Figure 13: Correlation heatmap of LL left and right and Alignment left and right	22
Figure 14: Wear of the wheel shown, the ruler is to show the material removed.....	25
Figure 15: Squat (on the left) and head check (on the right).....	25
Figure 16: Corrugated railway.....	26
Figure 17: Hyperplanes in SVM https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47	30
Figure 18: SVM https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47	30
Figure 19: SVM https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47	31
Figure 20: Data Gathering Roadmap	35
Figure 21: Utrecht-Den Bosch data points shown on QGIS	36
Figure 22: Spoortak shapefile in QGIS	36
Figure 23: Standard deviation calculation methods compared for section 245 out of 429. The pointed lines show the track quality class decision boundaries from EN13848-6 using velocity range of 120km/h to 160km/h. The x axis represents the indices of the points from table of Appendix A.	39
Figure 24: Standard deviation calculation methods compared for section 322 out of 429. The pointed lines show the track quality class decision boundaries from EN13848-6 using velocity range of 120km/h to 160km/h. The x axis represents the indices of the points from table of Appendix A.	39

Figure 25: Standard deviation calculation methods compared for section 106 out of 429. The pointed lines show the track quality class decision boundaries from EN13848-6 using velocity range of 120km/h to 160km/h. The x axis represents the indices of the points from table of Appendix A.	39
Figure 26: Soil map of the Netherlands in QGIS showing all different soil types and the midpoints of the sections	40
Figure 27: Transition zones in the Netherlands, blue is for culverts, green for tunnels, beige for level crossings and orange for railway bridges	42
Figure 28: Buffers shown	42
Figure 29: Hot encoding of object features, https://www.kaggle.com/dansbecker/using-categorical-data-with-one-hot-encoding	44
Figure 30: Correlation matrix drawn between the pre-processed feature values	45
Figure 31: Example Confusion Matrix	48
Figure 32: Methods of class aggregation compared in terms of their accuracy values for each model	51
Figure 33: Section lengths used compared in terms of their accuracy values for each model	52
Figure 34: Small section with track quality D shown, switches are in red, rail branches in green	52
Figure 35: Wiegthed performance measures compared between all the used classification algorithms for the test set using the aggregated classes and all section lengths	53
Figure 36: Example decision tree made with features familiar to this project. Internal nodes are the blue nodes, and the conditions that come with them are the yellow nodes. The pink nodes are the leaves.....	54
Figure 37: Feature importance calculated using Entropy method for impurity	56
Figure 38: Feature importances calculated using Gini Index method for impurity	56
Figure 39: Prediction accuracies compared across all the models between the cases where the two most important features are included and when they are not. Section lengths used are 200m+remainder, and classification method used is the 3-class aggregated method.....	59
Figure 40: Prediction accuracies compared across all the models between the cases where only the most important features are included and when all features are. Section lengths used are 200m+remainder, and classification method used is the 3-class aggregated method.....	60
Figure 41: K-Fold Cross Validation shown, https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74	63
Figure 42: Cross-Validation Mean accuracies across the 6-folds for each algorithm	63
Figure 43: Performance measures of tuned algorithms in train-test split manner	64
Figure 44: Inference accuracies of the second way (Utrecht-Den Bosch)	65
Figure 45: Feature importances of the second way	65
Figure 46: Training track sections are shown in purple while the track on which the inference will be conducted is in green. Both sections are in both ways.	67
Figure 47: Inference accuracies of the last piece of track of Utrecht-Den Bosch two-way track	67
Figure 48: Feature importances of the last piece of track in the Utrecht-Den Bosch line	68
Figure 49: Track chosen for last model generalization test shown on QGIS	69
Figure 50: The BCSF values for all the sections are compared between the Utrecht-Den Bosch track and the inference track	70
Figure 51: The soil deformation values for all the sections are compared between the Utrecht-Den Bosch track and the inference track.....	70
Figure 52: Feature importance using Random Forest for the inference track	71
Figure 53: Accuracies of prediction using each of the three tuned algorithms	72
Figure 54: Confusion Matrix: Random Forest- Inference	72
Figure 55: Confusion Matrix: XGboost- Inference.....	73

Figure 56: Confusion Matrix: SVM- Inference.....	73
--	----

Tables

Table 1: Sample table for learning track quality class to structural elements.....	5
Table 2: Data used along with its source	33
Table 3: Features used, their data types, definition and examples	43
Table 4: Model performance measures summarized.....	50
Table 5: Most important features, the features used for hyperparameter tuning	61

1. Introduction

1.1 Background

The demand for passenger rail and freight has been rising steadily in Europe. Between 2005 and 2016 a rise in 90 billion passenger kilometers was exhibited in the EU region along with a doubling of the length the high-speed rail network (McKinsey, Kienzler, Lotz, & Stern, 2020). A reason for this rise in volume and demand is the intrinsic sustainability that rail transport carries in its very core ("2021: the European Year of Rail | News | European Parliament," 2021). Rail transport is responsible for only 0.4% of the greenhouse gas emissions in the EU, while the transportation sector as a whole embodies 25% of the said emissions ("2021: the European Year of Rail | News | European Parliament," 2021). Another reason that rail is in constant demand is the safe travel it offers with respect to other land modes of transport. In short, rail fatalities were measured by the European Railway Agency to average 0.156 fatalities per billion passengers/km in contrast to 0.23 for buses, 4.450 for cars and a whopping 38 for motorcycles (Borda-De-Água, Barrientos, Beja, & Pereira, 2017, pp. 30).

For the above reasons, railway is offered huge financial backing in Europe. In fact, in 2016 around €27 BN were spent in Europe on operation and maintenance of the rail networks (Schroten et al., 2019, pp. 59). Yet, only 7% of passengers and 11% of goods travel by rail ("2021: the European Year of Rail | News | European Parliament," 2021). If the 2050 European Green Deal goals are to be achieved in terms of climate neutrality, these numbers have to be increased drastically.

Railway technology dates back to the early 1800's (Smith, 2001, p. 22). In that regard, many of the most effective techniques of construction and maintenance are quite outdated. For instance, the ballasted track is one of the most widely used types of tracks in the world and is also quite an "old" technology (Smith, 2001, p. 22). On the same hand, this outdatedness translated later to a very reactive system of maintenance which could be at the core of the very high maintenance expenditures and the low percentage of passenger rail users. Indeed, railway operators are finding it quite difficult to keep their assets in good shape, which ultimately leads to huge maintenance expenditures and lower levels of comfort and satisfaction for the rail customers who are suffering various consequent delays and cancellations (McKinsey, Kienzler, Lotz, & Stern, 2020).

Various studies have shown that the solution to the very plight that runs the tracks of the railway sector is a shift towards predictive maintenance using analytics. Talking numbers, it was shown that operators could reduce their maintenance spending on specific tracks up to 30%, using big data techniques. The reasons why the extravagant maintenance spending could be reduced with such methods are uninformed spending. It turns out that infrastructure managers spend on sections a ton of money without knowing **the interdependencies that lie with other sections, this leads to a multitude of track qualities on the same connection**. In addition, studies have also shown that vulnerability of the maintained sections are not taken into consideration with the amount of money spent which leads to problems of **overinvestment and underinvestment** (McKinsey, Kienzler, Lotz, & Stern, 2020).

Many research attempts have been done in order to circumvent the above problems and thus lead to better maintenance decisions. Some have applied the usage of artificial neural networks hand in hand with track geometry data to be able to predict the number of defects on a section (Falamarzi, Moridpour, & Nazem, 2019, p. 156). Other studies have tried to use machine learning codes to be able to predict the influence of switches and crossings on track quality (Falamarzi, Moridpour, & Nazem, 2019, p. 156). All of these studies can help with the problem of overinvestment and underinvestment and lead to better resource allocation by predicting which sections of the track need the most

attention; however, they do not help highlight the interdependencies between the specific sections and which structural feature impacts quality the most. That is where the gap in the literature lies, a track quality predictor that aims to bolster the interdependencies between the sections themselves by relating them to their neighboring structural elements. Structural behavior is what creates the interdependencies between the specific sections. If a switch is close to a section, the dynamic amplification factors are increased when the train travels through that section and it will thus be more in danger of showing defect with time than another section that is right before the said switch (Grossoni et al., 2019, p. 1185) due to the variation in geometry that the switches and crossings engender. However, that same section does not have to be safe from defects as it could be home to a weaker subsoil. Therefore, in order to better localize and allocate resources as to avoid the problem of inefficient resource allocation all the while bolstering the interdependencies of sections that lie within their structural architecture, a new predictor has to be made: a track quality predictor that uses structural elements as features for the study.

Showing the track quality and structural elements interdependencies will thus help optimize maintenance expenditures, and thus lead to a subsequent rise in railway usage which ultimately would reduce the greenhouse gas emissions of the transportation sector.

1.2 Problem Definition

A lot of research has been done on the different types of degradation models that ultimately fall into different categories: Mechanistic Models, Statistical Models and Artificial Intelligence Models (Falamarzi, Moridpour, & Nazem, 2019, p. 156). The motivation towards these studies lies in the consequences that rail deterioration can have, from engendering massive casualties all the way to leading to damaged assets and thus huge financial loss (Falamarzi, Moridpour, & Nazem, 2019, p. 156). It should be noted that these degradations models do not directly contribute to maintenance decisions, yet help optimize them and lead to a better resource allocation (McKinsey, Kienzler, Lotz, & Stern, 2020).

The model that will be created in this research lies in the branch of Machine Learning models and in the sub-branch of classification models. A classification model is a supervised learning model and is used in predicting a categorical or numerical label using binary, numeric, categorical features. Alemazkoo et al. (2015) used classification codes to predict track geometry parameters using features such as tonnage, track length and elapsed days between two consecutive defects while achieving accurate predictions (Alemazkoo, Ruppert, and Meidani 2015). Moreover, Andrade and Teixeira (2014) predicted whether sections on a railway line would exhibit defects using the locations of bridges, switches and stations vis a vis the specific sections. A finding of this study was that the standard deviation of the longitudinal levels and of alignments in the short wavelength region are a reliable predictor for defects and track quality (Andrade & Teixeira, 2014).

Following the above dialectic and looking into more depth at the EN13848-6 Dutch standard for Railway wherein longitudinal levels and alignments were used to characterize track quality into 5 different labels from A to E. A machine learning classification algorithm was then used to predict said labels for sections on the Amsterdam to Eindhoven railway line as a case study using structural elements as input. The model, once found accurate will have to be inferred using another railway line. To better frame the problem statement of this research:

Machine Learning Classification algorithms are a widely used tool for degradation models in the railway sector. Many researches have delved into predicting specific track characteristics using some structural elements. A major finding in the said researches is that the standard deviation plays a big part in determining defects and track quality. The lack of a predictor of track quality class using

general structural aspects and longitudinal level data along with alignments is missing in the literature. This could lead to better understand the relationships between the elements of a track all the while helping to target and allocate resources to specific lower quality and more vulnerable tracks.

This research aims to fill this gap in the literature by using standard deviations of longitudinal level data to create the label for track section quality and then learning these labels to general structural elements as to create a track quality predictor highlighting interdependencies with other structural assets.

1.3 Research Objective

Following the above investigation, aiding the railway sector to thrive is a triumph for both sustainability and land travel safety. In order to do so, research shows that the employment of data analytics is of paramount importance as to help with the problem of equivocal resource allocation. Most researches aim to do that albeit ignoring a very important aspect that need be tackled: bolstering the interdependencies between the structural assets.

Ergo, the objective of this research is to follow in the footsteps of previous degradation model researches all the while adding this newfound intricacy that aims to emphasize the relationships that lie between the structural assets vis-à-vis track quality in a specific section. The degradation model will also serve as a track quality predictor that takes into account neighboring structural elements for track sections. In addition, the structural elements that impact track quality the most will be highlighted.

1.4 Research Question and Sub-Questions

The main research question is formulated to tackle the research objective and problem statement. It is expressed as follows:

How to relate track quality in a section to its surrounding structural elements and form a predictor out of it?

Breaking this down into sub-questions:

1. What structural elements are to be studied?

Answering this sub-question requires a literature review to identify structural elements that influence track quality the most. In addition, the ease of gathering the structural element data has to also be taken into consideration. Answering this sub-question will call upon a description of the chosen structural elements. This will be done using the course material of the following courses: Wheel-Rail Interface, Structural Design of Railway Track, Elements of Railway Engineering.

2. How to automate the generation of the features?

Once the structural elements to be gathered were decided upon, the usage of either a GIS program such as QGIS was used to automate the feature generation. Since the learning algorithm took place on real rail data, it was useful to automate the generation of the features into tabular form ready for learning.

3. Which classifier algorithm yields the most accurate predictions?

Machine learning classification can be done using many algorithms and many parameters could be changed. Answering this sub-question led to a finding the most accurate algorithm and parameters

for our case study. This was complemented by an analysis and explanation of the subsequent results.

4. Which structural features are the most important in predicting track quality?

The random forest classifier allowed for highlighting feature importance and to quantify it. This provided very valuable information, as we have a measure to know which structural elements are impacting track quality the most. Once the model was built and has been proven to be accurate enough according to pre-defined accuracy measures, it underwent inference which meant using data that the model has not seen predict track quality and to compare it to reality, this can be a different railway track or by simply holding out some data and testing the model.

1.5 Research Relevance

1.5.1 Academic Relevance

This research bridges an understanding regarding the structural elements and their interdependencies on a railway track thus providing insight on the degradation mechanisms and the vulnerability of a track relative to its neighboring structural elements. In other words, we can further understand because of what a track degrades and why. The analysis provided to choose the structural elements worthy of study is academically valuable as it called upon the coursework followed during the MSc study of the Railway Specialization

1.5.2 Industrial Relevance

Creating a track quality predictor that feeds solely into structural elements could prove useful knowledge for the industry as it is quantifying knowledge that is otherwise more qualitative. It will pinpoint tracks that are in bad condition and will be an aid to contractors or infrastructure managers to know which sections of a track qualify as hotspots and why.

Little to no research has been done using relative track geometry measured by train-borne systems such as RILA, the system developed by Fugro.

Expanding on the previous point, maintenance decisions could be better mobilized knowing which track sections are most vulnerable and why. For instance, if it turns out that track sections closest to switches are in the worst shape, contractors would know to focus their resources on said sections.

Moreover, knowing what structural element is impacting track quality the most can also help allocate resources in a better manner for maintenance decisions.

1.6 Research Design

An overview of the methodology that will be employed to answer the research questions that fit into a research scope will be given. Limitations of the research methods will then be discussed at the end of this section.

1.6.1 Research Scope

This research is being conducted with the help of Fugro's railway department whose RILA device has provided the longitudinal level data that will be worked on. RILA stands for rail infrastructure alignment acquisition system. It is a state-of-the-art measurement device made by Fugro that aligns with the core qualities of rail: Safety and Sustainability. In short, it is a device that can be mounted on any passenger train thus eliminating the need for personnel to be deployed on the track which consequently helps mitigate safety issues with respect to those workers. In addition, the need of specialized measurement trains is also removed thus freeing up the track and maintaining capacity. Furthermore, the time that an entire route's data can be collected using RILA is still less than the time

it would take to arrange the train paths for survey trains. All in all, this research and Fugro's RILA device follow a similar path to advancing the Railway industry and making its most marketable features better.

Many track quality indices have been used all around the world to better quantify the state of the track in terms of its defects. The best suited track quality indicator for this project will be determined from literature and used as a label for machine learning classification codes as shown in Table 1 where an example track quality index was used taken from the European Commission for standardization (CEN, 2011).

The features used for predicting the track quality are structural elements gathered from available maps, databases all provided by ProRail, the infrastructure manager.

Table 1: Sample table for learning track quality class to structural elements

Section (window m)	Closeness to Switch (m)	Sleeper Type	Last maintenance(days)	Section Length (m)	Track Quality Class (EN13848-6)
1	12	Concrete	20	1000	B
2	30	Wood	50	1000	E
...	...	...	...	...	...

The project from which the data will be extracted is a project pertaining the data measurement of the Amsterdam to Eindhoven line. The track pertaining to that line will be of interest to this research: Utrecht-Den Bosch. The model will later be tested in terms of its generalization capabilities as to check whether the predictions are up to par for any track. **The generalization capability is called inference. The feature importance values of each track for validated models could be used by Fugro as consulting value. However, this project is research based and will not be part of a product that is sold to clients.**

1.7 Methodology

The methodology deployed to answer the research questions can be summarized using the workflow diagram shown in Figure 1. The diagram shows the sections in the report and the expected findings and output of each.

Predicting track quality using structural

elements and relative track geometry data measured by RILA: A Machine Learning

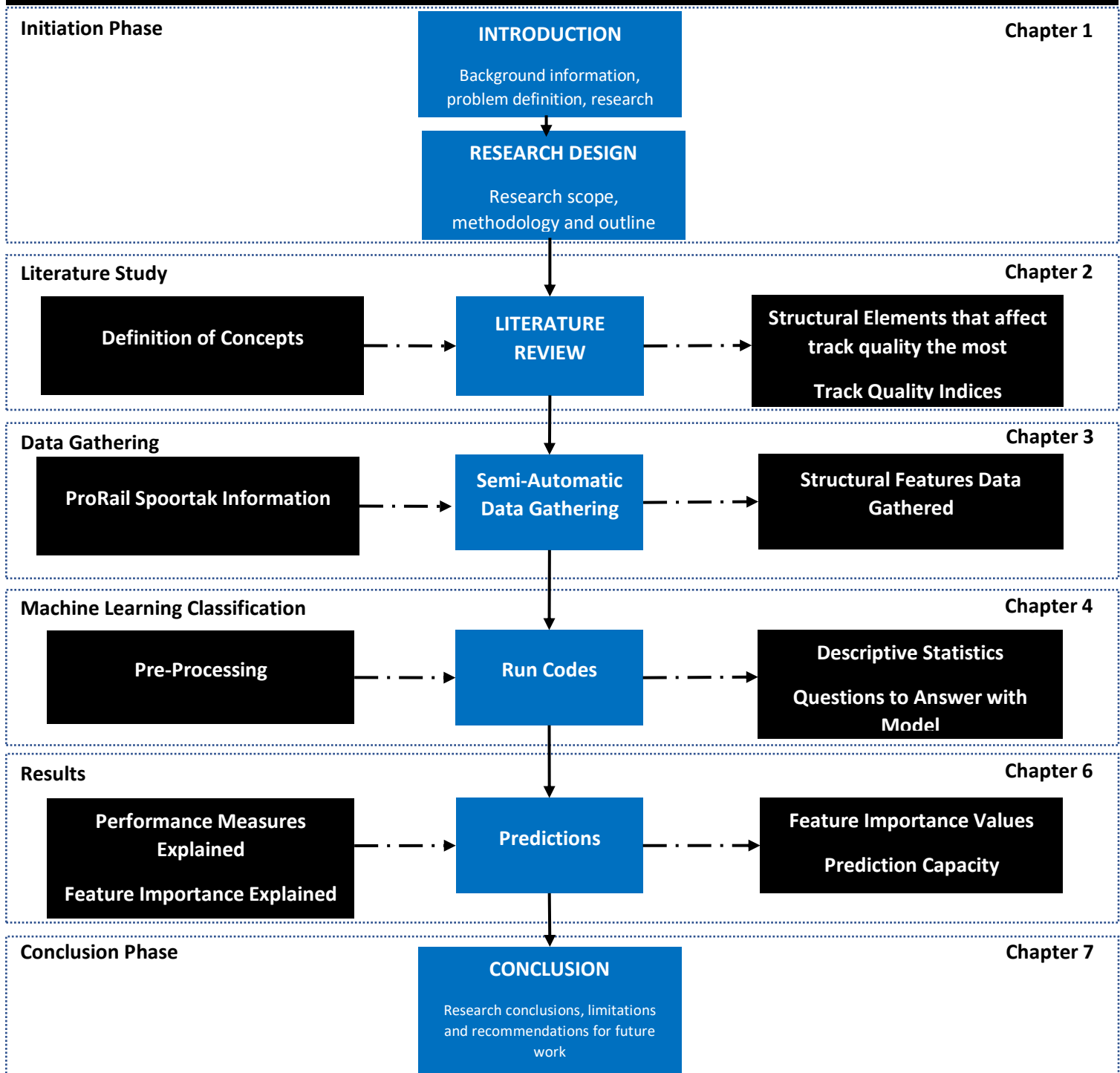


Figure 1: Workflow diagram for research work

Literature review

In order to better form the theoretical framework of the project and to, from the literature extract which structural elements have the most effect on track quality, a literature review will be conducted. The ease of collecting the data will also be taken into consideration as well, just as the ease of automating the data and making it usable for machine learning purposes.

A literature review will definitely circumscribe the structural elements that can be easily gathered and worked on all the while making sure they touch upon track quality. Literature review should also help us understand how the structural elements impact track quality and what to expect in terms of track quality when said structural element exists near the section.

The literature review will also define relative track geometry parameters and how they are used for track quality classification. The choice of which relative track geometry to use along with which track quality class to adopt will also be augmented in this section. In the same importance, previous work will also be highlighted vis-à-vis track quality prediction and AI methods to better define the choice of machine learning algorithms that will be used and defined in this section as well. The further expected result from this task is to identify the most prominent structural features to be studied in the Machine Learning process along with a structural in-depth analysis of how each one affects track quality which would lead to answering the first research sub-question.

Data Generation and Automation

All the structural features that were highlighted in the literature review will be cleaned and used as features in tabular form as shown in Table 1. This could call for the help of QGIS a geo-information system where the tracks can be visualized using their coordinates and where the structural data will be also visualized as layers is accepted and would be of great importance. This can be done using unsupervised learning methods for instance instead of QGIS.

This task will call upon writing Python codes and developing them in order to form a link between the numerical data that is used in this project and the structural elements that are surrounding a railway track.

In addition, to link the data Fugro has provided to the physical object data of ProRail, studying the nomenclature methodology of ProRail will be of paramount importance. Once both databases are linked it will be easier to create the sections out of the points provided by Fugro and link them to their physical objects that make them up.

The case study of the Utrecht to Den Bosch line will be used to visualize the output of each Python code and to understand the data linking further.

Machine Learning and Descriptive Statistics

From here on, the case study of Utrecht-Den Bosch line will be employed. The data from that line which was made ready for machine learning use in succession with the previous part will be used.

Obviously before using the data for learning, some exploratory statistics will be employed. Also, a changing the features into usable machine learning features which are supposed to be numeric is also included in the handling of the data. This is quite easy, as it can be automated. A categorical feature such as sleeper type has to be changed into binary for instance. It is therefore also very important to decide on the data type of the features that are to be used in the project.

The way this project will be conducted is similar to trial and error. The most prominent structural features will first be tested and prediction accuracies will be gathered. If the predictions give non-accurate results, then more features will be added.

The aim is to get to accurate predictions and then get the importance values of each feature to understand which feature can be dropped. The accuracy measures and evaluation techniques are also to be based on literature. A prominent accuracy measure is the confusion matrix which would identify false positives, true positives, false negatives and true negatives (Lewis & Brown, 2001). However, if all structural features from literature study were used and added to the model whilst prediction accuracy remains low, the project's goal will be to give insight and recommendations to future research.

Algorithm comparison

Machine Learning classification is part of supervised learning, which provides a predictive analysis based on an outcome that is known or can be observed (Mannila, 2016). In our case the label to be predicted which makes our enquiry a supervised enquiry is track quality.

Widely used machine learning classifiers in railway is support vector machine classifier and decision tree analysis (Falamarzi, Moridpour, & Nazem, 2019, p. 156). However, in an attempt to show the classification boundaries, which would show what values of the features need to be reached in order to travel from a certain level of track quality to another many algorithms will be used and compared in terms of accuracy of predictions.

The comparison of the classifiers' different accuracies will provide an answer to sub-question 3.

Feature importance evaluation

Some machine learning algorithms allow for a feature importance measure to be outputted. This will be done in order to understand which structural elements are impacting track quality the most, a very important insight in our project which will lead to answering sub-question 4.

Model Inference

Once the model has been trained on the Utrecht-Den Bosch line as a case study, the model will have to be tried on new data once the accuracy results are deemed acceptable. This will be done using the same structural parameters that engendered the predictions, however using a different railway track line or fabricated data whose output track quality is already known.

1.8 Results

The main aim of this part is to be able to get an accurate predictor for our case study. An analysis of the results will bring forth the following insights:

- Which structural elements affect track quality the most
- Values for the structural features could be considered as thresholds for track quality
- The most accurate classifier for such rail data

These insights will require thorough explanations and will call upon engineering knowledge to employ said explanations. These results and insights will also prove useful for future work on similar projects.

1.9 Limitations

It is very important to identify limitations in the proposed research methodology before delving into the project itself. No project goes as planned and obstacles are to be expected given any proposed methodology. The limitations that were identified are as follows:

- The limited working experience with Python will prove to be a difficulty. Both are included in the thesis planning as activities and have been given their adequate timings. If in any case learning Python proves to be a hindrance to the project's progress, it will be halted and MATLAB will be used instead. It has to be noted that Python is open source while MATLAB is not. Moreover, in terms of learning objectives, graduating with a working proficiency in Python increases my chances of finding a job. Finally, learning Python should not prove to be difficult as it resembles MATLAB and as it will be done in parallel with another activity and not hindering the start of another.
- Feature automation can prove to be difficult. This will call for innovative solutions and assumptions that need to be explained in the report. For instance, to know the distance to the closest switch for a specific section, the distance measure has to be decided on (Euclidian, longitudinal, etc.) and the start point of the distance measuring has to be also decided. This should be made in a logical manner that can be explained in an engineering manner. However, if this is done and a systematic method of definition is created, feature generation can be very well automated.

2 Literature Review

In this section, an attempt to define the theoretical framework of the entire project will be made.

“An important aspect of a part’s performance is how it interacts with other parts to affect the performance of the whole.” - Russell Ackof.

In fact, a railway structure is no exception to the above quote. Ergo, the railway components will be defined and their role in the general performance highlighted as well. In addition, track quality parameters will be bolstered all the while understanding the elements that affect it the most and how: positively or negatively.

2.1 The Railway Track

A system is a set of components all working towards the same goal as a part of a mechanism or a network. The railway track is indeed a system of smaller components with different ages, qualities and lifecycle that all work together to provide a safe and smooth passage for trains (International Union of Railways, 2010). In addition to guiding the train to safe passage, the track also acts as a load transferring medium, taking the huge train wheel loads and transferring them to the subsoil.

The track components are split into two categories:

- Superstructure
- Substructure

The superstructure as seen in Figure 2 consists of the rails, the sleepers, the ballast and the fasteners that help all components stick to each other (Lopes & Papaelias, 2017).

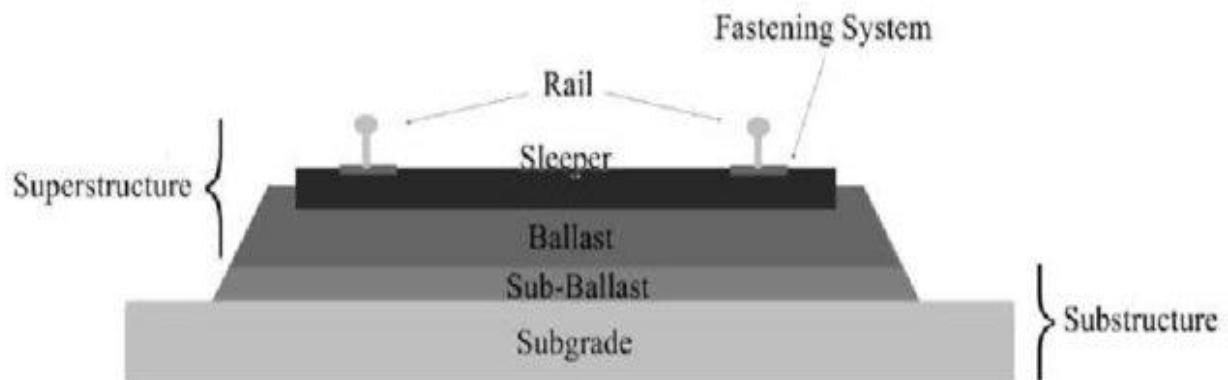


Figure 2: Schematic of Railway Track Structure (Lopes & Papaelias, 2017)

On the other hand, the substructure consists of the sub-ballast and the subgrade.

All these components have individual functions, however they all cooperate for the same end goal: a safe and guided train ride. The next sections will go more into depth into the components themselves and their functions.

Once the structural elements that make a railway track are defined along with their individual functions, track quality will be demarcated and the major structural elements that hinder it will be summarized from literature. In a later conclusion, the said structural features will be the first stepping stones to better understand and form first expectations on how track quality should change as we vary one of the defined structural elements.

2.1.1 Superstructure elements

Rails:

Rails are very important because, they are the stadium where the wheel-rail contact occurs. This contact happens at the rail head which has a surface area of approximately 3 cm^2 . This obviously leads to huge stresses which need to be transferred to the lower layers so that the axle loads could be supported without any harm. Rails can have different types. The most common one is the vignole rail which consists of a rail head, a rail web and a rail foot just as Figure 3 shows.

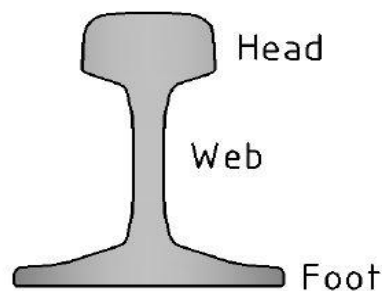


Figure 3: Conventional railway cross-section

Moreover, the rail must have enough stiffness to work analogously to a beam which transfers the concentrated wheel loads to the equidistant sleepers (Reddy, 2004).

In a structural manner, the rail acts to (Markine, 2020):

- Reduce the rolling resistance
- Distribute wheel loads to the lower layers
- Guide the wheels by reducing chance of derailment
- Distribute longitudinal forces
- Provide a running surface with adequate friction characteristics

Sleepers:

Sleepers are beams that tie both rails to each other. They come in many types as well. They can be duo-block and mono-block. As their name reveals, mono-block sleepers literally traverse under the rails perpendicularly as to tie both rails together. Duo-block sleepers on the other hand are two sleeper elements that support each rail and are connected via a steel beam-like element (Figure 4).

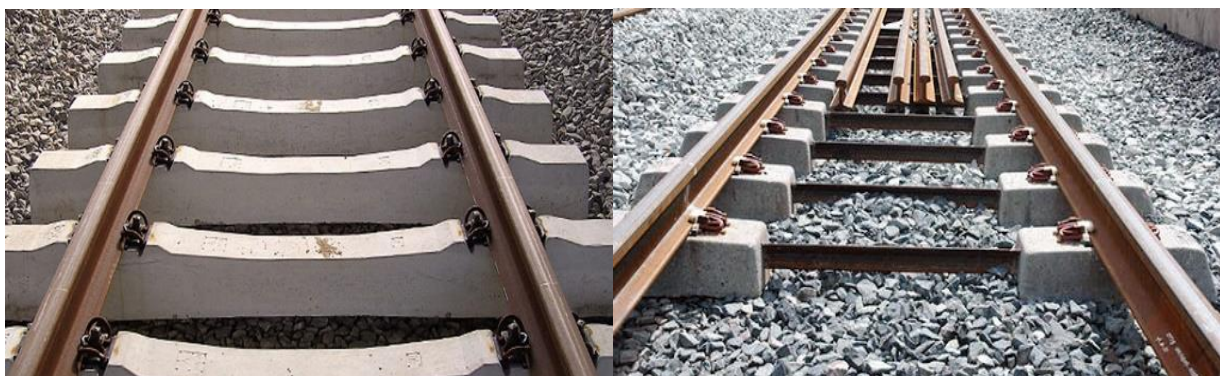


Figure 4: Monoblock sleeper on the left (<https://ballast.ir/sleeper/>) and Duoblock sleeper on the right (<http://www.rail-fastener.com/concrete-sleeper.html>)

Sleepers also come in different material types: concrete, steel, timber and even plastic. The advantages and disadvantages of each element will be tackled in the section where we look at track

quality and how the specific elements affect it. For now, the sleeper functions will be simply highlighted by their structural functions (Markine, 2020):

- they provide support and fixing possibilities for the rail foot and fastenings
- they support the vertical forces and distribute them equally to the ballast
- they provide resistance for the track in a longitudinal and lateral manner
- they preserve track geometry: gauge and inclination of rail

Ballast:

A track can either be ballasted or non-ballasted (Figure 5). Non-ballasted tracks originate from the 1960s (Alabbasi & Hussein, 2019) and are used in different operations depending on the track's location. The difference between ballasted tracks and non-ballasted tracks is multifaceted. In budget terms, a non-ballasted track is much more expensive than a ballasted track. In terms of ease of construction, a ballastless track is ahead. Functionality wise, both have their pros and cons and ultimately, the better option really depends on the track's location and its intended purpose. This project focuses on the structural performance and functions of the structural elements as they are behind track quality.

Thus, in summary, a ballasted track tends to perform better in terms of elasticity, maintainability and noise absorption (Markine, 2020), but loses in the fact that after service, ballast material deforms and degrades and hence loses its functionality (Alabbasi & Hussein, 2019), while its maintenance is nevertheless an expensive activity (Indraratna et al., 1998). In addition to the above-mentioned ballasted tracks are laterally weak in curves (Markine, 2020).



Figure 5: Ballastless track on the left in a Taiwanese High Speed Line vs a ballasted track on the right

Having discussed the general differences between ballasted tracks and non-ballasted tracks, it is important to highlight the main structural functions of ballast (Alabbasi & Hussein, 2019):

- it transmits the loads from superstructure to the substructure and acts as a bearing platform
- it enhances the entire track's stability
- it acts as a damping medium and reduces vibrations and noise

Fasteners:

Fasteners are used to maintain the rails on the sleepers. Taking that into consideration they preserve track gauge all the while not allowing the rail to overturn in any direction, a phenomenon caused by rail forces and temperature effects (Reddy, 2004). The fastclip fastener is a type of fastener that can be easily clipped on the rail foot (Figure 6).

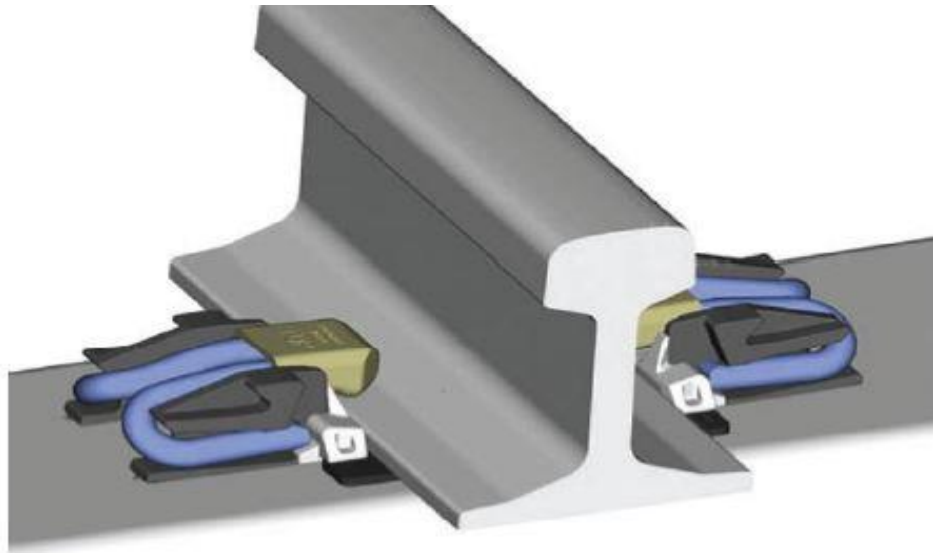


Figure 6: Fastener: "Fastclip"

Moreover, structurally the fasteners have the following functions (Giannakos, 2014):

- In case of rail rupture, fasteners keep the gap small
- Helps track gauge be preserved
- Adds longitudinal resistance to the track

The superstructure is the stadium where the wheel-rail interface occurs. Track geometry is measured on the superstructure. This does not mean that the substructure has no structural role in withstanding the wheel loads and in preserving track geometry too.

2.1.2 Substructure elements

The substructure usually consists of two layers. Sub-ballast and subgrade or formation. Most of the time, the sub-ballast layer is of 15cm. In some exceptions, the lower layer (subgrade) could have a bigger thickness and circumvent the need of any sub-ballast layer.

Sub-Ballast:

The sub-ballast layer acts as a transition between the upper layer (ballast) which tends to have larger particles and the lower layer of subgrade which tends to consist of finer particles. In summary the functions of the sub-ballast layer are as follows (Markine, 2020):

- It diminishes the value of the stress at the bottom of the ballast layer such that the subgrade can handle it. This can be called load distribution
- It prevents the mixing of particles between the ballast and the subgrade which allows for better drainage. This is why the particle sizes are different between the said layers

Subgrade:

This layer acts as a foundation to the track. It is also called formation level when an extra layer is used to level the profile of the track bed. The subgrade should be well compacted using soil improvement techniques as the track geometry's integrity relies on it.

The subgrade:

- Acts as a foundation and works as an ultimate support for the track bed
- Bears the load that results from the transmission all the way from the railways to the bottom layer
- Provides a smooth leveled platform for the track to rest on

It is important to understand the specific parts that form a track as they themselves will dictate track quality and track geometry integrity: two notions that will be defined in the next sections. The findings of this section are to give the reader a better scope and understanding of the track elements, as when explaining the degradation of these elements later on in the report, a good foundation of understanding would have already been laid down.

2.2 Relative Track Geometry

There can be made a distinction between absolute track geometry and relative track geometry. The former is nothing but a description in 3D of how the track looks like in terms of track position, gradients, curve radii and much more (H Wang et al., 2019). The latter is on the other hand, processed data that is morphed into track gauge, longitudinal level, alignment, cant and twist: measures which contribute to defining track quality (H Wang et al., 2019). For the same reasons, it is very important to define which relative track geometry data this project is used for track quality labeling, what it means and how it is processed by Fugro.

Relative track geometry and absolute track geometry are usually measured by measurement trains in a non-simultaneous fashion, despite the fact that both datasets are needed to provide a clear view on the state of the track and to make maintenance decisions. This means that in one run, either relative track geometry is measured or absolute track geometry, using the old-school system that is in place right now. This is due to the fact that both measurements require different devices which operate at different frequencies (H Wang et al., 2019). This results in having to measure the same track twice which leads to higher measurement costs.

Fugro, the company involved in this project have created an innovative measurement device dubbed train-borne system measurement wherein, as its name states a system of many small measurement components: RILA, is mounted on any passenger train using GNSS (Global Navigation Satellite System), IMU (Inertial Measurement Unit), LiDAR scanner, video cameras, and laser vision technologies (as seen in Figure 7) to measure absolute track data and relative track geometry data (Bronsvort & Hurk, 2013). This leads to the lack of overlap between both measurements and thus to lower measurement costs. In an indirect manner, the train-borne system also eliminates the need of using a measurement train which occupies the track due to its slow operating speed which contrasts the speeds that the train-borne system operates at: up to 200km/h. Another benefit of using the said system is the needlessness of personnel on the track which decreases safety hazards (Hurk, 2016).

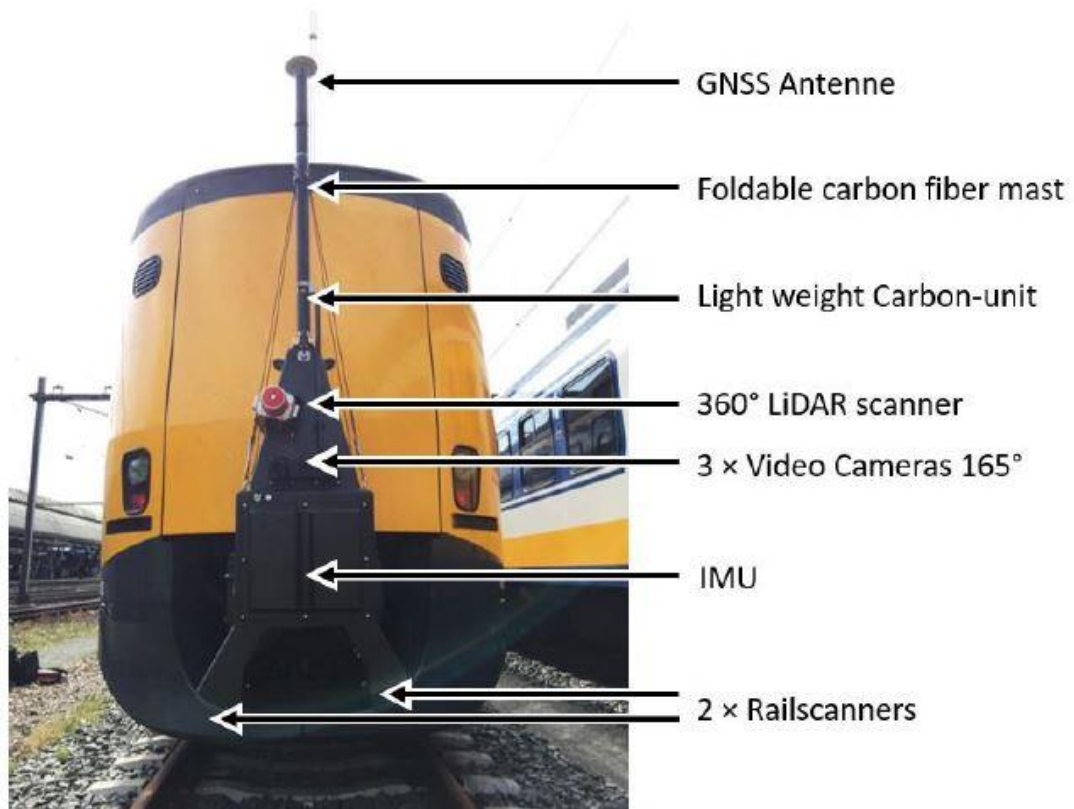


Figure 7: RiLA train-borne system and its components (Wang et al., 2019)

The research using the train-borne relative track geometry data which complies with the uncertainty parameters of the European Committee for Standardization's (CEN) standard depicted in EN13848-1 (CEN, 2011) is state of the art which brings us back to the research relevance of this project.

This project will not take into account all forms of relative track geometry. Namely, the lens will be shifted onto two specific relative track geometry parameters. Alignments and longitudinal levels in the short wavelength domain. The reason behind this focus will be verified later in the report.

For now, it is important to define these relative track geometry parameters, which segues the report into the next subsection.

After these two parameters are explained, the report shall go into depth on why exactly they have been chosen to represent track quality, based on the literature and the work done on track quality classification.

2.2.1 Longitudinal Level

Longitudinal level is relative track geometry measured in mm, which represents vertical deformations in the longitudinal railway plane vis-à-vis the baseline which is conveyed by the pointy line in Figure 8. The longitudinal level expresses the levelness of the track: a value of 0 is nothing but the baseline (H Wang et al., 2019). Larger values (negative or positive) show worse track quality.

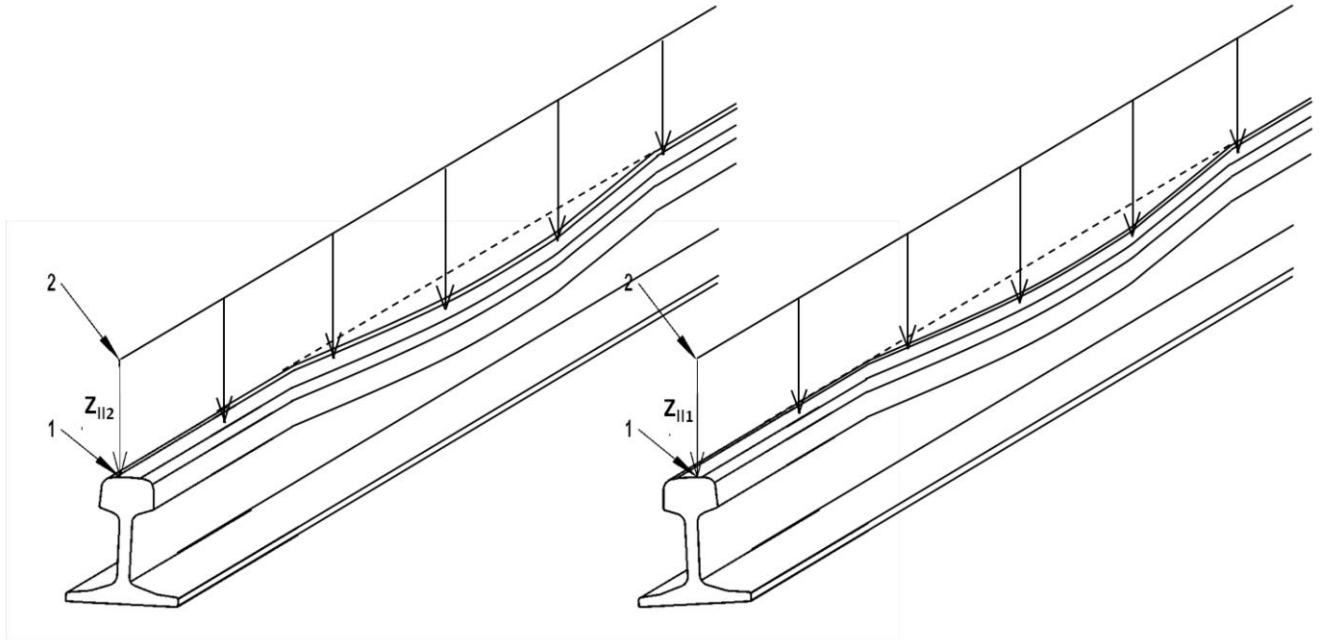


Figure 8: Longitudinal level/top definition, EN13848-1

Using the rail-head scanners that feed into both rails (Figure 8) using laser technology, the measurement done by Fugro returns (X,Y,Z) coordinates of the rail head itself. Tagging this with the video-camera's images, the profile of the rail-top is deduced. A baseline is then drawn for the railways themselves. The difference in height between the actual rail head and the nominal rail head results in the longitudinal level data. The data is then segregated into two wavelength domains using a filtering method according to (CEN, 2011). This filtering method is called the decoloring method and it should comply with the rules of annex C of EN-13848-1 (CEN, 2011), which gives bounds for the transfer function that should be used. The domains are called:

- D1: $3 \text{ m} < \lambda \leq 25 \text{ m}$
- D2: $25 \text{ m} < \lambda \leq 70 \text{ m}$

The data consists of points that are spaced by 0.25m associated with a longitudinal level for the left rail and the right rail in the D1 wavelength domain. Each point is also given the global coordinates that are inferred by the GNSS system.

What this piece of data gives insight towards is the vertical forces that are on the wheel rail interface. The D1 region data would give insight on the general track condition (OBrien et al., 2018).

2.2.2 Alignment

Similar to the longitudinal level data, alignments are a relative track geometry data that give similar insight however in the lateral direction. This means that a baseline is also set in the horizontal plane for how the nominal track looks like, and the lateral deviations from that baseline are recorded as the alignment relative track geometry (CEN, 2011).

This can be better visualized using Figure 9, taken from the definition of the alignment relative track geometry in the EN13848-1 standard (CEN, 2011).

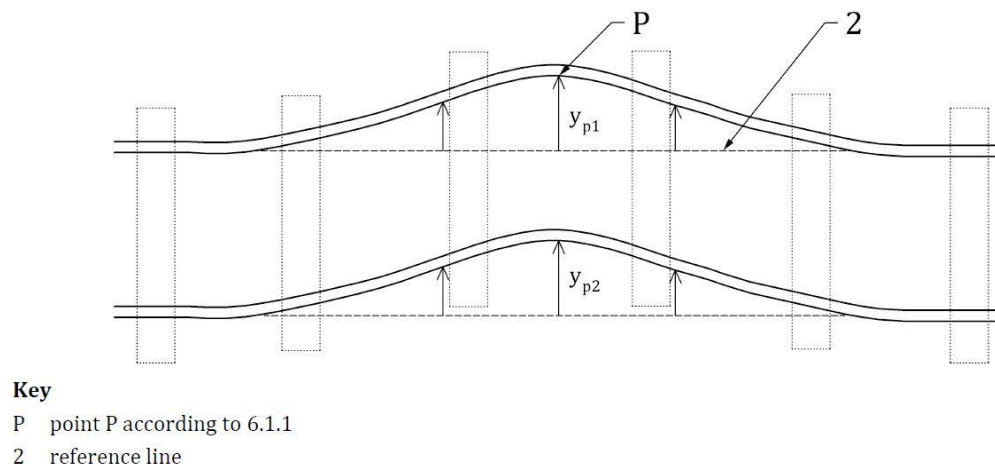


Figure 9: Alignment definition, EN13848-1

This type of data is also gathered in a similar manner using the rail-head scanners of the train-borne system RiLA that collects (X,Y,Z) coordinates of the rail heads and then using post processing tools that in fact create the baseline that is shown in Figure 9, calculate the deviations from that line for every 0.25m. The data is then filtered into the same two wavelength domains shown above according to the same rules of annex C of EN-13848-1 (CEN, 2011).

Each data point includes values of longitudinal level data in the D1 region for the left rail and the right rail along with the alignment data in the D1 region for the left rail and the right rail (see Appendix A: Utrecht-Den Bosch Project Sample Data).

The wavelength region of the irregularities is of paramount importance for track-vehicle interaction (Tacıroğlu et al., 2020). The reason that the shorter wavelength data is focused on for both relative track geometry parameters is that short wavelength irregularities are a bigger safety concern than longwave irregularities which start at 25m of wavelength (Grassie, 2012). The latter type of data is only of a nuisance on the comfort level rather than safety level and thus is less used for maintenance decisions (Ben-Daya, Kumar, & Murthy, 2016). In another light, most maintenance decisions are based on the D1 wavelength alignments and longitudinal levels (López-Pita et al., 2008) as the conventional measurement trains use a chord length of 25m for track inspection. Using a longer chord length would result in a much longer measurement train which is very rare (Markine, 2020). **However, it should be noted as well, that the D1 region irregularities belong to the comfort control range and this will be shown in the next section but are also important as they play a part in resonance phenomena in sprung masses, as well as semi-sprung masses of the vehicles and play a huge part in track degradation** (Paixão et al., 2016).

Any irregularity that is in the D1 region is due to ballast settlements, fabrication errors in the rails or a problem in the alignment of the sleepers. (Yu et al., 2016). These irregularities show up in the beginning of the lifecycle of the track. However, in theory a track's life starts in a "defect-less" way and the defects build up with usage of the track and the engenderment of the wheel-rail interface forces. The methods to uncover the track's state or quality are all embedded in track quality classes that shall be investigated in the next section.

The findings of this section are the definition of relative track geometry and the two most prominent ones that are used for track quality classification. The next section will show exactly why that is all the while discussing ways to turn the alignment and longitudinal level data into track qualities, using work previously done in track quality indexing. The focus on the D1 wavelength region was highlighted in this section and will also be reiterated in the next section.

2.3 Track Quality Parameters

Track quality indices (TQI) are a way to quantify track quality and are very important for infrastructure managers to be able to guide their maintenance decisions.

In their research (Offenbacher et al., 2020), an overview of existing track quality parameters is made and a distinction between two types of TQI's is made: individual TQI and holistic TQI. The former represents an index specific to one relative track geometry data, basically the deviation of rail position in a specific direction (Liu et al., 2015). Holistic TQI's or combined TQI's on the other hand as their name states combine many relative track geometry parameters. In retrospect, (Andrade & Teixeira, 2014) point out in their paper that for planned-maintenance decisions, the individual TQI using the standard deviation for the short wavelength (3–25m) of longitudinal-level is seen as the main indicator for planned-maintenance decisions. It is argued in the same paper however that for a better overview of track quality and the forces that define it, a horizontal component should be added to the decision-making: alignment data in D1 region, as the small wave longitudinal level and the small wave alignments are highly related to the vertical forces and horizontal forces on the track, respectively as discussed and shown in (Giannakos, 2014; Lichtberger, 2005). It is also pointed out in (Giannakos, 2014) that the most commonly used track geometry parameters for standard deviation are indeed the short-wave alignment and the short-wave longitudinal level. Although, these two parameters are as above stated widely used for planned maintenance, (Andrade & Teixeira, 2014) conclude their paper by stating that the said parameters are also good predictors of unplanned-maintenance which all the more motivates their usage for track quality indexing in this project.

According to (Offenbacher et al., 2020), some indices count the isolated defects according to threshold limits defined by standards. Other indices use standard deviation values of the relative track geometry signals. In short, isolated defect counts don't show the overall state of the track as they neglect the amplitude of these specific defects as they are based on counting values above a certain threshold.(Offenbacher et al., 2020). Standard deviation values on the other hand can better depict the general track quality (Liu et al., 2015; Offenbacher et al., 2020; Sadeghi, 2010) since the standard deviation values would show the scattering of the data and how far the RTG data is from 0 taking into consideration all the values in the section, while isolated defects are more of a local indicator of track quality.

In another light, combining relative track geometry parameters can be detrimental to the TQI values as sometimes underestimation and overestimation occurs due to the fact that not all relative track geometry signals are in the same amplitude ranges (Offenbacher et al., 2020).

The first finding is therefore that the longitudinal level data and the alignment data in the D1 region as defined in the previous sections, will be used for track quality indexing. Namely, the standard deviations of the said data are going to be used according to the EN13848-6 tables for track quality classification using a speed between 120 and 160 km/h (CEN, 2011). This speed was taken from the records of the railway line in study: Utrecht-Den Bosch.

It has been shown that the preferred method of maintenance decisions is to use the alignments and the longitudinal level data in the D1 region. This project being a complement to maintenance decisions

in the Netherlands where this parameter is widely used takes this as enough of a reason to work on the said data. However, this should be made clear in addition to what was stated in the last section as to why it is so preferred. The human body interacts with different frequencies in different ways. Many standards have been developed to evaluate human passenger comfort. The most widely used standard in Europe is the EN 12299:2009 Ride comfort for passengers, developed by the CEN wherein the method of evaluating ride comfort is known as the mean comfort method (Jiang et al., 2019; Paixão et al., 2016; Pradhan & Samantaray, 2018). Other widely used standards for similar applications are the ISO 2631, and UIC 513R as well as the Sperling Index (Pradhan & Samantaray, 2018).

In their research, (Jiang et al., 2019) highlighted the sensitive ranges of each standard using each method of comfort evaluation. This was also confirmed by (Madalina Dumitriu & Dragos, Ionut, 2021) in their research that included using every single standard and comfort thresholds of each method to highlight the most sensitive frequencies for lateral and vertical directions. Both papers concluded that the most sensitive frequency domains per standard direction were (Jiang et al., 2019; Madalina Dumitriu & Dragos, Ionut, 2021):

- Using the mean comfort method that is included in the standards of EN12299 and UIC 513, the sensitive frequency ranges are 4–16 Hz and 0.6–2 Hz for vertical and lateral directions respectively.
- Using the ISO 2631 certification, the sensitive frequency range was found to be 4–12 Hz and 0.6–2 Hz for vertical and lateral directions respectively.
- The Sperling method showed that the sensitive frequency domains for both directions were the same: 3–7 Hz.

The frequency of vibration caused by an irregularity is calculated by the equation:

$$f = V/L$$

Where f is the forcing frequency in Hz, v is the velocity of the vehicle in m/s and L is the irregularity's wavelength in m.

Using a speed of 140 km/h or 38.9 m/s, which is the common speed for passenger trains in the Netherlands, and wavelengths of 3m and 25m, we get a frequency range of 1.56 Hz to 12.96 Hz. This frequency range fits into the frequency range for which humans are the most sensitive and covers it the most. The longer wavelength domain D2, of 25m to 75m is related to the frequency domain of 0.52 Hz to 1.56 Hz. This also fits into the sensitive comfort domain, and can be used for track quality evaluations, however only for comfort purposes. While the D1 domain also belongs to the resonance frequency range of the sprung masses and the semi-sprung masses as already discussed above and according to (Paixão et al., 2016). **Another point to be taken away here is that due to the comfort criteria in both lateral and vertical directions, it is very important to look at both the alignment data and the longitudinal level data in addition to the reasons already mentioned above.**

Shifting the lens back to the classification of track quality, the CEN has created the below tables to classify track quality using standard deviation thresholds into 5 classes: A-E. These numbers were based and formed during a European track survey that took place in 2010. The cumulative distribution of the standard deviations was drawn up for 5 speed ranges for both the alignment data and the longitudinal data gathered (D1). Using the cumulative distributions of the standard deviation values, the thresholds in the tables for track classification were decided following the below reasoning (CEN, 2011):

- Class A – best 10 % of the distribution of European Track Quality

- Class B – between 10 % and 30 % of the distribution of European Track Quality
- Class C – between 30 % and 70 % of the distribution of European Track Quality
- Class D – between 70 % and 90 % of the distribution of European Track Quality
- Class E – above 90 % of the distribution of European Track Quality which represents the worst 10 % of the distribution

Speed (in km/h)	Limit value of standard deviation (in mm)				
	Track quality class				
	A	B	C	D	E
$V \leq 80$	$< 1,25$	1,75	2,75	3,75	$> 3,75$
$80 < V \leq 120$	$< 0,75$	1,10	1,80	2,50	$> 2,50$
$120 < V \leq 160$	$< 0,65$	0,85	1,40	1,85	$> 1,85$
$160 < V \leq 230$	$< 0,60$	0,75	1,15	1,60	$> 1,60$
$230 < V \leq 300$	$< 0,40$	0,55	0,85	1,15	$> 1,15$
$V > 300$	Not available	Not available	Not available	Not available	Not available

Figure 10: Longitudinal Level D1 track quality classification EN13848-6

Speed (in km/h)	Limit value of standard deviation (in mm)				
	Track quality class				
	A	B	C	D	E
$V \leq 80$	$< 0,90$	1,25	1,95	2,70	$> 2,70$
$80 < V \leq 120$	$< 0,50$	0,70	1,05	1,45	$> 1,45$
$120 < V \leq 160$	$< 0,45$	0,55	0,75	1,00	$> 1,00$
$160 < V \leq 230$	$< 0,40$	0,50	0,70	0,90	$> 0,90$
$230 < V \leq 300$	$< 0,35$	0,40	0,50	0,65	$> 0,65$
$V > 300$	Not available	Not available	Not available	Not available	Not available

Figure 11: Alignment in D1 track quality classification EN13848-6

For each section, a standard deviation for both the longitudinal level and the alignment will be calculated for the left rail and the right rail. **The EN13848-6 standard pushes to average the values of the left rail and the right rail, according to the Annex C method of calculation, however it must be noted that some information is lost using that method, as sometimes in curves, the outer rail and the inner rail exhibit different problems and averaging both standard deviation values can underestimate the severity of the defects in those areas. However using standard deviation values in general are to look on a more general view of the track quality, and averaging the left rail SD value with the right rail's aligns with that. Using maxima in any aspect localizes the track quality class and provides a track quality for a section of 200m which is defined by only one point of said section.**

The combined track quality for a section will be the worse track quality of both alignment and longitudinal level. This means if a section is qualified as A for longitudinal level but C for alignment, it

undertakes the track quality class of C. The reason behind this is that the standard deviation values cannot be averaged between the longitudinal level and the alignment as they both are in different ranges and their respective tables that allow for classifications also contain different ranges (Figure 10, Figure 11).

Before moving on to the next section, the calculation method of the standard deviation should be discussed in terms of two points that are deemed important for qualifying track quality (Offenbacher et al., 2020):

- Influence length: the length of the section for which the track quality will be carried on.

In almost every literature, a section of 200m is used for track quality classification (Andrade & Teixeira, 2014; Offenbacher et al., 2020; Sadeghi, 2010; Soleimanmeigouni et al., 2020; Yu et al., 2016). It is also argued in the EN13848-6 standard that typically 200m or 100m sections are used (CEN, 2011). In our project 200m sections will be used for the sole reason that it fits our data type more and this shall become clearer in the data gathering section where the section thresholding python code is highlighted. In short, to avoid repeatability of the data which kills the variance and in turn the machine learning algorithm, 200m sections will be used.

- Calculation method of the standard deviation: this can be either in a moving or continuous manner or in a discrete manner.

It is argued in (Offenbacher et al., 2020) that using a moving method is better as it follows the data more closely. The EN13848-6 standard from which the tables in Figure 10 and Figure 11 advises the usage of a continuous moving standard deviation calculation at steps of 0.25m while averaging the values of the left rail and the right rail. Then, each section would contain 800 moving standard deviation values for the alignment and the longitudinal level. The standard offers 4 methods to choose the standard deviation value or TQI_{ref} that will be used to assign track quality class to the section. Using the maximum value, the average, percentiles of cumulative distributions, and threshold percentages for each track quality class. All those methods were compared with each other by the standard in terms possible applications (Figure 12).

Application	Maximum value	Mean value	Percentile of a distribution of TQI _{ref}	Percentage of a required TQC
Key performance indicator used in a high level maintenance strategy	--	+	++	++
Detailed working plan for track maintenance	++	++	+	+
Acceptance of track works influencing the track geometry quality	++	+	+	--
Monitoring of the global quality of the track for contractual purposes between infrastructure manager and the infrastructure owner	--	-	++	++
Contractual purposes between train operator and infrastructure owner	--	+	+	++
Design of a vehicle according to ride quality requirements and track quality of the lines where the vehicle will run	+	+	++	+
Selection of track sections for vehicle acceptance	+	+	++	+

Figure 12: Possible applications of choosing the TQI_{ref} value based on the method, EN13848-6 page 20

The ranges ++/+/-/-- are from most applicable to least applicable. As we can see, the percentile of a distribution of a TQIsref, which implies choosing a value of a global percentage X% that applies to all the sections, and to look at the cumulative distribution of the standard deviations on that section. The standard deviation value for which X% of the data lies under it, will be chosen as TQIref to assign track quality class. This method gives good global quality insight of the track for contractual purposes between the infrastructure manager and the owner while most other methods don't. A similar method is to also choose a percentage Y% for which the data has to adhere to a certain quality as to be classified by it. These global percentage methods give a good global comparison between the sections. However, that is not a requirement to this project that is looking to form detailed working plans for track maintenance and an acceptance of that maintenance vis-à-vis the track geometry quality. Instead, local means and maxima are of interest to the predictor that will be formed. The maximum standard deviation values would give very local indicators while the average method would actually still have one foot in a global comparison between the sections. **These two methods will be shown and compared in the data gathering section on specific track sections. Graphs for sections will show the discrete standard deviations and the continuous standard deviation as well as the maximum standard deviation values and the average standard deviation values of the continuous calculations. The moving standard deviation will be calculated for each and every point for the left rail and right rail for both the longitudinal values. The left rail standard deviation values will be averaged with that of the right rail as these values are already somewhat correlated (Figure 13).**

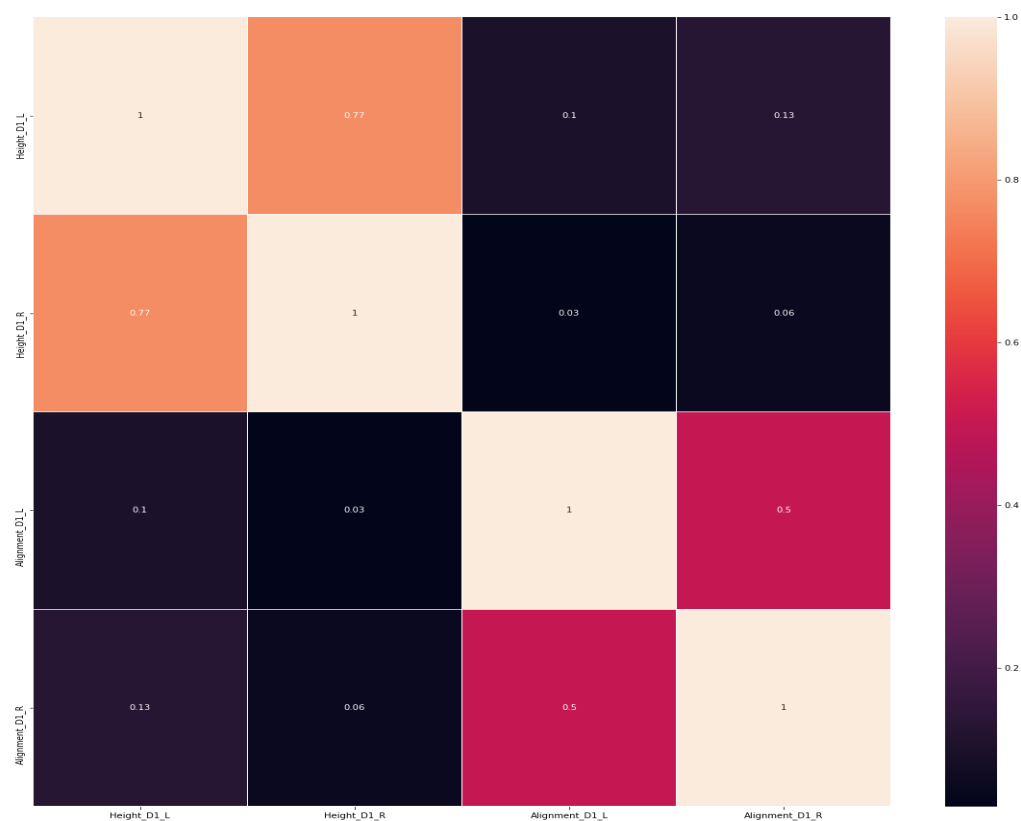


Figure 13: Correlation heatmap of LL left and right and Alignment left and right

What can be seen is that the longitudinal levels or heights have 0.77 of correlation between the left and the right rail, while the alignments left and right have a 0.5 correlation between them. Longitudinal levels and alignments have no correlation in between them. The high correlation values

are due to the fact that irregularities in the D1 region are usually settlements for heights and lateral track shift for alignments. These phenomena occur similarly on both rails.

The next section will take a look at the structural elements that affect track quality the most, according to the literature. Those features will be explained and the expected behavior of the track will be also highlighted vis-à-vis any change in those prominent structural elements that impact track quality.

2.4 Factors affecting track quality

In this section, an overview of the structural parameters affecting track quality will be made. To do so, it is important to sift through the forces that act on the railway track. Once these are well understood, the degradation of the track will be made clear with respect to the most common defects that can be found on it. In the final part of this section, the factors that affect the track quality will be highlighted and the expected behavior of the track with respect to the specific factors will be shown. The ultimate goal of this section is to set the stepping stones for the data gathering part of the project wherein the factors highlighted here will have to be gathered and turned into a usable format for machine learning classification codes. However, it is important that by doing so the reader understands what it is to expect if one of the factors here is varied in terms of track quality.

2.4.1 Forces on the track

There are three type of forces that act on the railway track (Giannakos, 2014):

- Vertical forces

Forces that are caused by changes in track stiffness and track geometry. These act vertically on the railway track and the train wheels.

- Lateral forces

Forces that are acting laterally on the rail, and these are mostly due to curves in the railway track, where to be able to negotiate the curve, the train would have to create higher rolling radius differences and thus leads to flange contact and lateral forces. To counteract the curve centrifugal forces, superelevation or cants are used in the railway track to evenly distribute the vehicle loading on both tracks (Markine, 2020).

- Longitudinal forces

The longitudinal forces are mostly acceleration and deceleration forces. They are also caused by temperature effects that would lead the railway to expansion or contraction (Markine, 2020).

Each direction of the forcing impacts the railway track in a different plane. The vertical forces will engender a vertical deformation, which will show up in the longitudinal level data, for instance, while the lateral forces which act horizontally on the track will show up in the alignments data that is gathered.

It is important to understand the nature of these forces, and how they are caused. Another distinction that can be made with respect to the forces, other than the direction in which they act in, is their nature, which can also be split into three categories (Giannakos, 2014):

- Static Loads

These loads engendered from the train's weight and are considered a minimum value of the force itself (Giannakos, 2014).

- Quasi-Static Loads

Those are loads caused by the vehicle's body's interaction with the track itself. Meaning, that the track would apply certain forces depending on the vehicle's properties. The loads that belong to this category are typically caused by an abnormal contact between wheel and rail or by unevenly distributed loads on specific parts of the track (Giannakos, 2014). However, it must be noted if the conditions of the rails and the wheels are perfect, then this type of loading would be nothing but the static load itself. If not, it can reach up to a multiplier of 1.4 to 1.6 of the static loads (Giannakos, 2014).

- **Dynamic Loads**

These are the loads that interest our project the most as they are highly dependent on the wheel-rail interface's condition. Not only that, but the dynamic loads are also dependent on the structural elements that neighbor the track. Namely, any sudden change in stiffness or geometry in the railway system, a dynamic load is engendered and can reach up to 3.5 times the static load (Giannakos, 2014). This comes to show that the track's condition is a double-edged sword, as any irregularity which is caused by the load in the first place can increase that load and thus lead to worse track quality (Sun et al., 2013). For the above reasons, from hereon this section will focus on dynamic loads as they have the most potential for railway degradation and are dependent on the condition of the track itself and on the neighboring structural elements, which is exactly what this project focuses on.

In fact, dynamic loads can be split into two: the P1 load and the P2 load (Giannakos, 2014) depending on their frequency. It should be noted that a high frequency load would be characterized as an impact load and thus lead to a higher dynamic amplification factor. To explain this in simpler terms, a high frequency force would lead to a higher multiplier of the static load (up to 3.5).

P1 load:

This type of loading is the impact load discussed above; its range is from 100 Hz up to 2000 Hz. This type of loading is caused by any small wavelength change in geometry. These are typically surface irregularities such as welds and joints, but can also be rail defects or wheel defects. The cause of such high impact forces is cracking of concrete sleepers, and rail irregularities (Giannakos, 2014).

To put this into more perspective, the dynamic amplification factor, which is the number that multiplies the static force when subjected to such an irregularity is given by the formula below:

$$DAF = \frac{1}{1 - \frac{\omega^2}{\omega_n^2}}$$

Where ω is the forcing frequency shown above as f and ω_n is the natural frequency of the dynamic system. What is important to note here is that the forcing frequency is immediately related to the dynamic amplification factor which leads to a higher dynamic load on the rail and subsequently worse track quality because the shorter the wavelength the higher the frequency.

P2 load:

The P2 load is the load that comes right after the P1 load. It belongs to lower frequency forcing, thus to larger geometric irregularities such as ballast settlement and sleeper positions (Markine, 2020).

The P2 loads have a forcing frequency that ranges from 30 to 100 Hz (Giannakos, 2014). Their maximum dynamic amplification factor can reach up to 2.5, which leads to a force that is 2.5 times the static force, and this can lead to serious degradations in the railway track, and thus to a lower quality of the track which will show up in the model that is going to be developed in this project, since the wavelength domain D1 is called in the literature "short" wavelength domain but only in

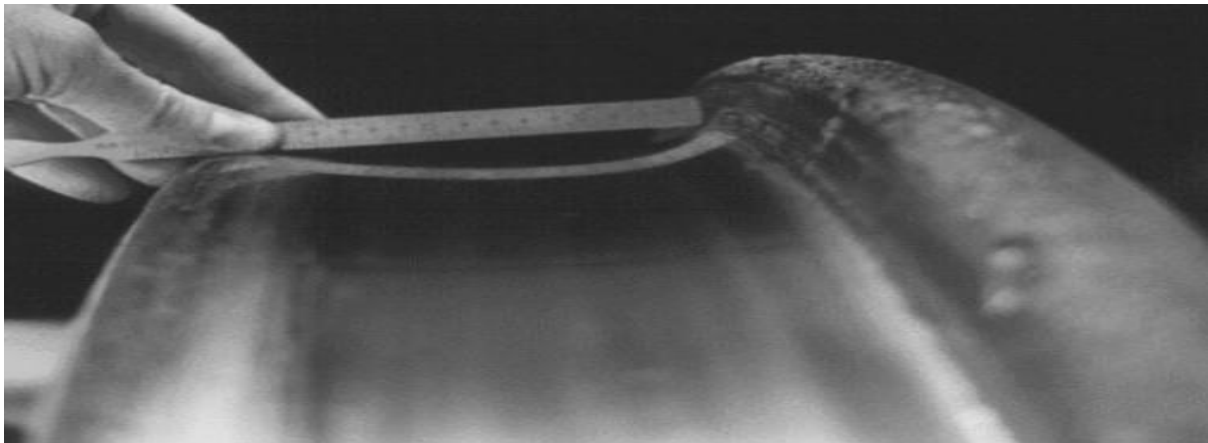
comparison to the D2 domain. It is still however part of the long wavelength region, wherein mostly ballast settlement and sleeper positions impact the data and a P2 load is thus engendered.

It has to be noted that irregularities whose wavelengths are less than 0.5m are the main cause of the P1 dynamic loading. Examples of these track irregularities are track defects (section 2.4.2), welds, insulated joints and any very small and sharp change in geometry. What needs to be highlighted is, that although these elements cannot be included in the model as structural features impacting track quality due to their difficulty of tracking, they also play a big part in inducing longer wave irregularities that this project is focusing on (3-25m). This occurs due to the high impact load that travels through the track as to affect the ballast and the sub-ballast and thus cause settlement (Markine, 2020).

2.4.2 Most important track defects

The consequences of the above discussed forces are irregularities on the track which are ultimately what show up on the longitudinal level data and the alignment data. These are thus the reason behind a poor or good track quality. These defects will be listed below along with the major causes behind them.

1. Wear: material removal due to contact stresses. This occurs on rails and wheels (Figure 14).



2. Rolling contact fatigue defects (RCF): squats and head checks which respectively occur on straight tracks and curved tracks. These occur with cyclic loading and age (Figure 15).

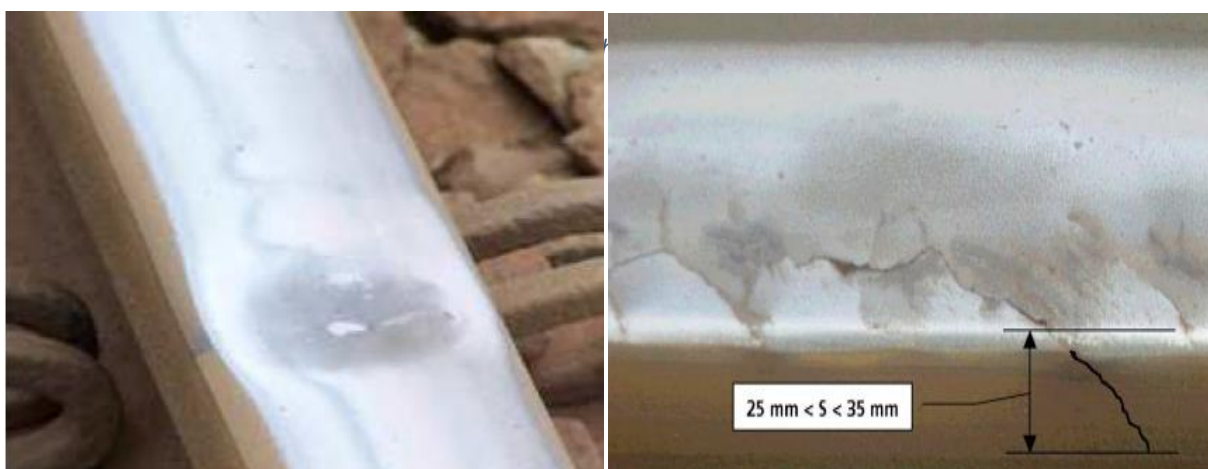


Figure 15: Squat (on the left) and head check (on the right)

3. Corrugation: this is similar to wear but occurs in patterns, which is because it is due to dynamic loading (Figure 16).



Figure 16: Corrugated railway

There are more defects and irregularities to be discussed, yet these are the most prominent ones. It's important to note that all these defects occur more and more as the rail reaches its lifecycle end (Giannakos, 2014). Those defects also add to the high impact load occurrence and in the long run cause more settlement that shows in the D1 wavelength region (Markine, 2020).

To conclude this subsection, it is important to have understood the types of railway forces and the way they act on the railway track as to better understand the factors that hinder track quality. We can already name a few after this section: age of the track elements, speed of the train which plays into the dynamic amplification factor, and any small wavelength geometry change that engenders a P1 dynamic force. Also, obviously, the static load is quite important, as it shows up in every single type of loading: dynamic and quasi-static. The final part of this subsection included highlighting the most common defects that occur on the railway track and how they are caused as to relate them to the project at hand and how they end up causing settlement that shows up in the D1 wavelength region. **However, these defects along with small geometry transitions (insulated joints, welds, expansion joints) cannot be tracked and thus will not be included in the model which will play a part in reducing its accuracy.**

2.4.3 Factors hindering track quality

Following the argumentation that was built above, forces on the track create defects on the track which in turn will show up in a lower track quality class (Grassie, 2012; Sun et al., 2013). It is therefore this section's goal to highlight the factors on a railway track that allow for the track forces to become more accentuated and thus hinder track quality.

In the Netherlands, a huge amount of the railway tracks is situated on subsoils. Some research points towards the fact that most tracks on soft subsoils require double the maintenance of tracks that are on stiff subsoils (López-Pita et al., 2008; Zhai et al., 2004). This goes back to the definition of the subsoil and its core function: carry the loads transmitted from the wheel-rail interface in a load bearing manner. **It is thus important to include the subsoil type in the model of the project as it definitely impacts track quality and the settlements engendered from the weakness of the subsoils show up in the D1 wavelength region.**

Transitions zones and switches are found to require 4-8 times more maintenance than regular free track (Coelho, 2013).

Transition zones are locations on the track where a sudden change in stiffness occurs in the sub-structure. These locations are tunnels, bridges, culverts, level crossings or even a location where the track changes from a ballasted to a non-ballasted. According to Alves Ribeiro et al. 2007, the higher the stiffness variation, the higher the interaction force. Some research was made using numerical models to check the influence of this variation in stiffness, and it was found that were the stiffness change gradual and not sudden as in most cases in real applications, the interaction force would be smaller (Lundqvist et al., 2006). **Therefore, it is important to include in the model transition zones as they increase the forces on the track and ergo impact track quality.**

In the Swedish railway network, switches and crossing constitute 5.5% of the entire railway network. Albeit, the maintenance cost for these elements constitutes 13% of the total allocated maintenance expenditures for the entire railway network, which showcases their importance vis-à-vis track quality (Nissen, 2009). The reason behind these high maintenance expenditures lies in the very spatial and geometrical definition of these elements. A turnout, which includes a switching zone and a crossing zone is created to help vehicles merge into a different railway line. What is important to note is that in the crossing zone, a geometric gap is needed for the vehicle to change lines, which results in a large P1 impact force. This leads to wear and degradation of the track and reduction of track quality. This also leads to more settlements in the long run (Markine, 2020). **Switches are thus important to add to the model as they impact track quality and having them before a railway branch would lead it to more impact forces and defects.**

To re-iterate what was concluded in the sections above, the dynamic forces are increased if the static weight of the train is increased, meaning the tonnage value. **This can be added to the model, with an expectation that the higher the tonnage value, the higher the dynamic forces and the worse the track quality.** In addition to that, speed of the traveling train is important as it is directly related to dynamic amplification factors. We can thus conclude that when train speed is higher, the forcing frequency is higher and thus the dynamic amplification factors (DAF) are also higher. **Thus, passenger train speed is important to also add in the model because dynamic forcing is increased with it and thus track quality is worse.** Moreover, the quality and the age of the track elements are very important to be added to the model, because once the sleepers, rails and ballast reach their lifecycle end, they impact track quality in a worse way. **Hence, the age of the sleepers, the rails and the ballast will be added, keeping in mind that the higher the age the worse the track quality of the sections that exhibit them.**

Sleepers come in many types as discussed in section 2.1.1. In fact, timber sleepers are the most cost effective, resilient, corrosion resistant, and workable sleeper. They also are easy to handle and have a high potential of re-use and insulation (McAlpine, 1991). However, they provide less lateral and longitudinal resistance and are a source of ballast settlement because of that (Markine, 2020). Concrete sleepers on the other hand, which take the form of the NS90 sleeper in the Netherlands are more expensive and have higher resistances in both the lateral and longitudinal directions. They have a better impact on track quality of the section exhibiting them. **The sleeper type will thus be included in the model as to better highlight the expectation: timber sleepers perform worse than concrete sleepers in terms of track quality.**

The ballast thickness is an important parameter for evaluating the ballast's performance (Alabbasi & Hussein, 2019; Indraratna et al., 1998; Zhai et al., 2004). The thicker the ballast the more the load is well distributed to the lower layers. This can be very good for vibration properties and thus noise

emissions. **It is also important for track quality, as the thicker the ballast the better the track, so to expect.**

Having named the factors that affect track quality the most and in which manner they do, these will be implemented in the model and a conclusion will be made on which parameter is the most important for deciding track quality following the assumptions of the model. However, before getting to the data gathering, machine learning will be explained and how it applies to this project. The algorithms that will be used will be explained in this next section along with some small examples on how they were used in the literature for better understanding. However, following the previous sections, it has to be reiterated that the small wave length irregularities that also play a huge part in causing long wavelength irregularities that fit in this project could not be included in the data as they are very difficult to keep track of. The project does indeed take a look into the most important factors impacting track quality, seeing their ease of data-gathering.

2.5 Machine Learning

Relative track geometry (RTG) data exhibits the 5 V's of big data: volume, value, veracity, velocity and variety (Lasisi & Attoh-Okine, 2018). The volume stems from the huge amount of data that is available, which applies to this project that consists 351,852 data points. The value stems from the correlation between track quality indices which are derived from the RTG data and the real track quality. The veracity of the data has been proven by (H Wang et al., 2019), where the measured data of the train-borne system was proven to comply with the CEN requirements. The velocity lies in the processing time of the RTG data generation which is Fugro's business, however using automated processing methods, this is also abided by. The variety, being one of the most important elements lies in the usage of many datasets, which is in core what the data gathering phase will look like, using Fugro's relative track geometry data, and ProRail's physical objects and attributes as well as open-source maps and map services that will give us special structures.

Machine learning is a consequence of big data. It is defined as a subset of computer science that allows the computer, much like the brain to make associations using data and to learn without being explicitly programmed to do so (Munoz, 2015). Using such algorithms for insight and for predictions is becoming more and more abundant in the civil engineering fields and namely the railway engineering field (Falamarzi et al., 2019; Lasisi & Attoh-Okine, 2018; Practice, 2020; Soleimanmeigouni et al., 2020). Namely, there are two sub-branches of machine learning, wherein one of them includes statistical learning which is also called data mining; a type of learning that attempts to give exploratory data analysis, by clustering the data, or finding anomalies or much more. This is called unsupervised learning. However, the second type of learning is more of relevance to this project: supervised learning allows for predictive analysis using labeled data which in this project's case is the TQI. Supervised learning contains regression, classification, dimension reduction and many more terms that will be defined in this section. After having defined these terms, an explanation will be given on why classification problems fit this project the most and an overview of the available algorithms that allow classification will be made.

Some terms need to be explained to the neutral reader. Features are the elements used to predict the label or the class. In this project for instance, **the features** are the structural elements that were discussed in the previous sections. They will be used to predict **the label** which is the combined track quality class of the alignment and the longitudinal level in the D1 domain. The machine learning model is trained by splitting the data into two sets: training and testing. These are usually split in a 70-30 manner (Falamarzi et al., 2018, 2019; Lasisi & Attoh-Okine, 2018; Soleimanmeigouni et al., 2020). **The training set** is the data that the model will use to learn the associations and the intricacies and

variations of the data by looking at the features and the labels. The learning method obviously depends on the algorithm used and the mathematics behind it. **The test set** contains the features that will be used to predict either a number (regression models) or a class (classification models) using the associations and the learned patterns formed in learning. There are many methods to evaluate the prediction accuracy of the test set, which ultimately depends on the model type.

2.5.1 Regression

Regression models are models that feed into the features provided for predictions and predict numerical values (Falamarzi et al., 2019). In fact, in one of their papers, (Falamarzi et al., 2018) worked on a regression model to predict track gauge deviation values in the Melbourne tram system using features such as rail type, rail support type and rail profile. The predicted values of the testing set were compared to the real gauge deviation values and the mean squared error (MSE) parameter was used to evaluate the accuracy of the predictions. The formula for MSE is as follows:

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_{i, \text{actual}} - y_{i, \text{predicted}})^2,$$

With N the number of data that is compared, $y_{i, \text{actual}}$ the actual value and $y_{i, \text{predicted}}$ the predicted value.

Regression models could apply to this project, however due to the combining of the alignment TQC with the longitudinal level TQC, it becomes complicated to predict the standard deviation values separately and combine them again. For this reason, regression models will not be used for this project.

2.5.2 Dimension Reduction

Dimensionality reduction is used for feature engineering. In a case where many features are included in a model, dimensionality reduction can be applied to create features that belong in a different coordinate system that explain the variance of the N input features. In their work on creating an artificial TQI (Lasisi & Attoh-Okine, 2018) used several RTG data such as alignments for left and right rail in both D1 and D2 wavelength domain, longitudinal level data with similar parameters and gauges along with cross-level data. The goal of this research was to transform the RTG data into **principal components** which are planes created by the machine learning algorithm to create new features that explain the variance of the multi-feature data. Once the principal components were formed and the new features with them, those were used to predict whether a section contains a defect or not using a classification model wherein the label would just be “YES” or “NO” based on real observations. This research really shows the distinction between the function of the dimensionality reduction algorithms such as principal component analysis (PCA) and classification models which are to be discussed shortly.

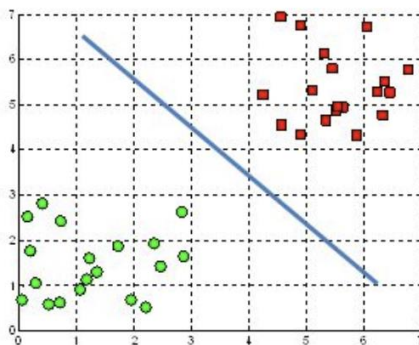
2.5.3 Classification

A lot of work has been done in the railway engineering industry with classification algorithms (Falamarzi et al., 2019). An interesting research that resembles the project at hand is the classification of soil using random forest by (Gambill et al., 2016) The research looks into using elements of soil classification of the United States Department of Agriculture (USDA) to get to a different classification system, namely that of the Unified Soil Classification System (USCS). The features that are usually used for the USDA soil classification come in all types of data forms and units: strings, percentages, tons, cm’s and much more. This is in fact similar to the features that the model at hand will use: sleeper type which will be either concrete or timber is a string feature, while ages of the elements come in integer values. Due to that, the project qualifies for the usage of the random forest classifier as well.

Other research namely in the railway sector has used the random forest classifier in parallel with support vector machine (SVM) and linear discriminant analysis (LDA) to classify whether a track will have defects using an artificial TQI (Lasisi & Attoh-Okine, 2018). The methods of evaluating prediction accuracy all depend on the confusion matrix which will be explained along with the measures that are by-products of this matrix such as recall, precision and f1-score which will be explained in the results section.

2.5.3.1 Support Vector Machine

A hyperplane in $\mathbb{R}^2$ is a line



A hyperplane in $\mathbb{R}^3$ is a plane

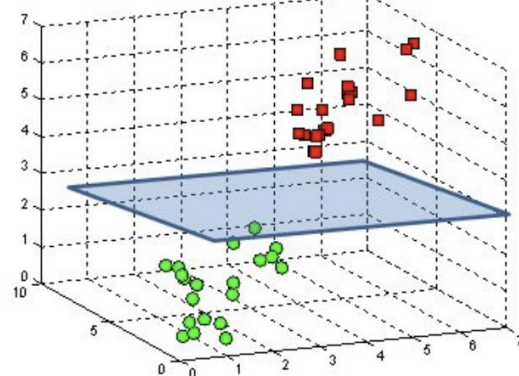


Figure 17: Hyperplanes in SVM <https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47>

The SVM algorithm is based on the concept of hyperplanes that define decision boundaries. A decision boundary is a boundary that separates the set of features that pertain to different classes, in this project's case: class A,B,C,D, or E. On either side of the hyperplane, a different class is given as shown in Figure 17. If the number of features is X then the dimension of the hyperplane is $X-1$. As seen in Figure 17, the line corresponds to two features while the plane corresponds to three features. In this project's case it is difficult to imagine the hyperplanes as more than 5 features are being implemented to be able to classify the data.

The support vectors are the datapoints that are closest to the hyperplane. These points form the start of what is called the margin which is the pointy line seen in Figure 17.

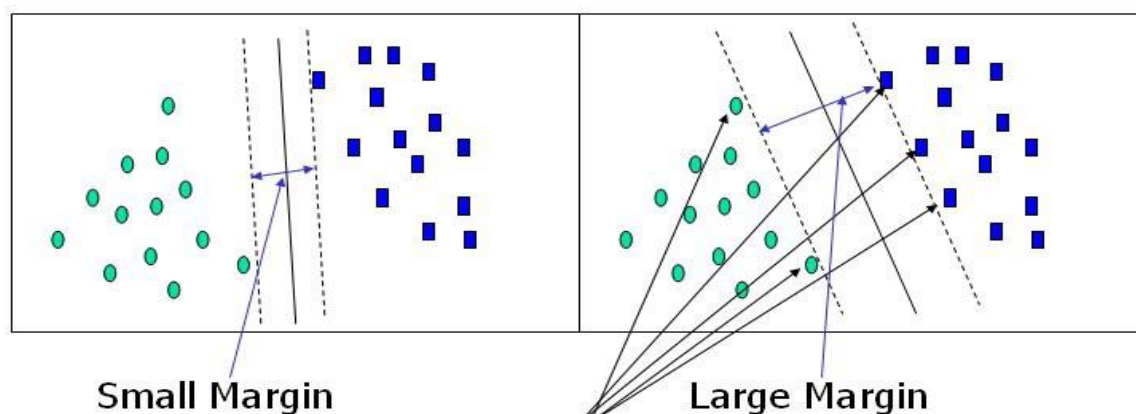


Figure 18: SVM <https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47>

The way the SVM algorithm optimizes the choice of the margins is by minimizing the cost function which would lead to maximizing the margin while decreasing the misclassification error which is the data that lies on the wrong side of the hyperplane. This is done in an iterative manner until the optimization problem is done .

The advantages of SVM are its versatility, and its effectivity in high dimensional spaces all the while being efficient in memory usage (“1.4. Support Vector Machines — scikit-learn 0.24.2 documentation,” 2004).

2.5.3.2 Random Forest

Random Forest algorithm is an ensemble machine learning algorithm, in that it takes the properties of the decision tree algorithm and takes samples at random and creates a decision tree analysis on it and reaches a class (in our case A,B,C,D,E). The forest creates a number of decision trees based on the user’s input and the more trees it has the more robust it is (Gambill et al., 2016; Lasisi & Attoh-Okine, 2018). Each Decision tree that is formed from the training set will have a decision on the track quality class for the data point in the test set and then using a majority voting, the final predicted class will be chosen (Donges, 2020), (Figure 19).

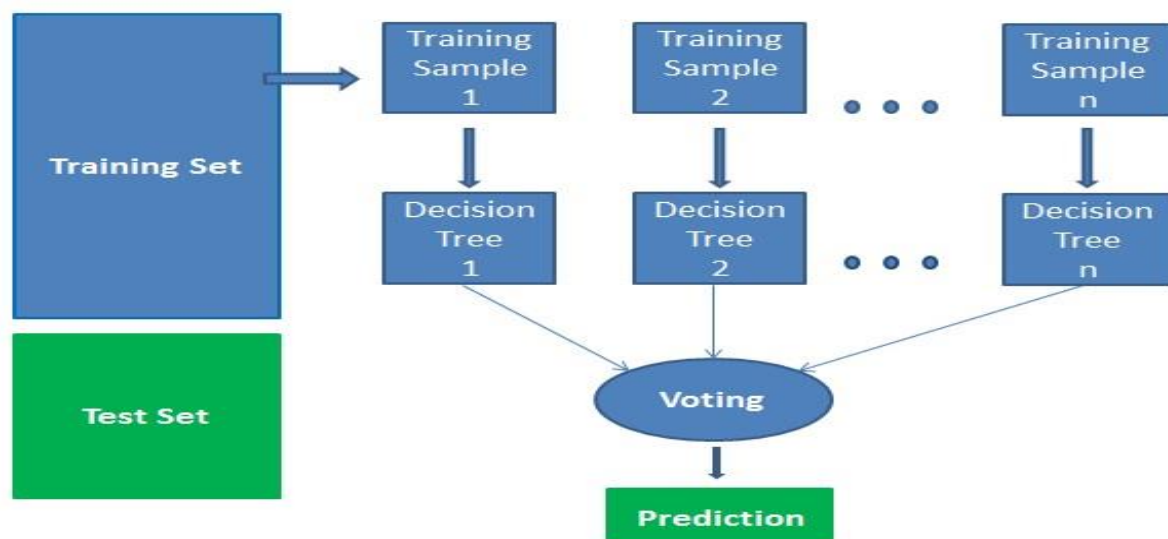


Figure 19: Random Forest algorithm: <https://www.datacamp.com/community/tutorials/random-forests-classifier-python>

The advantages of this classifier are that it does not require any pre-processing of the data, in terms of normalization or standardization. It also feeds into categorical and string values very easily without any prior handling. Another important aspect is that it avoids overfitting and thresholding which are problems that any other machine learning algorithm faces. Overfitting, creating decision boundaries specific to one data point to classify it well without looking at the rest of the data. Thresholding is the same idea as overfitting but in a threshold manner (if a feature is bigger than a certain threshold than it is a certain class). The most important advantage is that it gives us the importance of the features vis-à-vis the predictions. This would give us insight on which structural features affect the track quality the most. Other less important features could then be disregarded (Donges, 2020).

2.5.3.3 XGboost

This algorithm has many names, gradient boosting, multiple additive regression trees or just XGboost. This algorithm is branded by a performance superiority in terms of processing time and memory usage. It has been applied in many projects and fields namely in cancer diagnosis(C. W. Wang et al., 2018), network intrusion algorithms (Devan & Khare, 2020) and much more. Its fame comes from its

usage in almost every Kaggle competition winning almost every time (C. Wang et al., 2020). This ensemble algorithm works in a similar manner as the random forest, however it works in a “boosted” manner. This boosting is the phenomenon where new models are added to correct the errors made by existing models. Models keep being added until no improvement can be reached anymore (Brownlee, 2021). XGboost however does what we call gradient boosting wherein the new models that are added predict the residuals of prior models. The term gradient boosting stems from the fact that this algorithm uses a gradient descent algorithm to reduce the error whilst adding the new models. The usage of this model in railway applications is quite state of the art and RTG data being voluminous could benefit from the low memory usage this algorithm requires. Also, no prior pre-processing is needed for the data and categorical data is also accepted, similar to the random forest algorithm.

3 Data Gathering and Automation

In this section, it will be made clear how the data went from being data points with X, Y coordinates and RTG data of longitudinal levels and alignments for D1 region in the left and right rail all the way to becoming sectioned data with structural features related to it.

A small overview of the data used will be given and then an explanation of ProRail's rail branch naming mechanism will be given as it is of paramount importance to form the link between the RTG data that Fugro have and the physical object data that ProRail have. Later the methodology will be explained by going roughly through each and every Python script that was developed and QGIS's role in gathering some data.

3.1 Data Used

In Table 2 the structural features that were generated will be shown along with the source that was used for said gathering. The multiple datasets that were combined to achieve the gathering of the features confirms one of the 5 v's of big data: variety.

Table 2: Data used along with its source

Data	Data Source
X,Y,Z data	Fugro Raildata project
RTG data	
Ballast Age	Informatieportaal: Ballast, Fysiek_Object.csv
Ballast Type	
Ballast Thickness	
Sleeper Age	Informatieportaal: Dwarsliggers, Fysiek_Object.csv
Sleeper Type	
Tonnage Values	Informatieportaal: wisselberijding.csv
Rail Age	Informatieportaal: Rail, Fysiek_Object.csv
Rail Type	
Rail Material	
Switches	https://mapservices.prorail.nl/arcgis/rest/services/Geleidingssysteem_009/MapServer
Railway Branches	https://mapservices.prorail.nl/arcgis/rest/services/Geleidingssysteem_009/MapServer
Subsoil Type	https://www.tudelft.nl/en/library/collections/map-room/map-collection/thematic-maps/soil-map-of-the-netherlands-150000
Tunnels	https://mapservices.prorail.nl/arcgis/rest/services/ProRail_basiskaart_001/FeatureServer
Bridges	
Level Crossings	
Culverts	

3.2 ProRail Rail Branch Naming System

It is important to state that the structural features data is taken from ProRail's informatieportaal wherein physical object data is included in a csv file format for every single element that is included on the track: from sleepers to ballast. The thing is, each physical object data is related to a railway branch name and two numbers that represent the start and the end of that railway branch in a linear referencing manner. Knowing this, it was important to understand ProRail's rail branch naming methodology. In short, there is made a distinction between switches and rail branches (Berghuis, 2017). The naming of the rail branches is thus made from switch to switch. This means that whenever a switch shows up, the railway branch after it has a new name. The name is related to the switch itself but the details of the naming is not important. What is important is knowing that each rail branch has a unique name following the format:

555_123V_83.5

- The red numbers in the beginning belong to what is called a geo-code and it is nothing but a number that relates to a city or region that ProRail has coded in a manner to identify immediately where the track's whereabouts should be. An example is that a geocode of 539 means the track lies in Schiedam Centrum.
- The green code in the middle represents the switch that gave the name to the rail branch. 123V is a switch that can be searched for in the switches data. The available information that is of use in that file is the slope of the switch and the type of the switch.
- The yellow number is the starting kilometer value. This is used in a linear georeferencing manner. ProRail has for each railway branch a start and an end kilometer value to better identify the rail branches.

It has to be noted that while the rail branch or 'spoortak' name in the format above is unique, some of those branches are broken down into more pieces with a different KM start and KM end value, albeit under the same name of the railway branch, and with no other elements in between them. This has become clear once looking at the physical object data wherein one railway branch name has many different sleeper or ballast types associated with it, depending on the km end and start. Thus, to conclude, the 'spoortak' identifier above is unique, however it is broken down into many more of the same branch as to distinguish the different physical objects in between. This will become clearer as the methodology of feature gathering is explained.

3.3 Methodology

The methodology of gathering the structural features and sectioning the data is summarized in Figure 20. An overview of each step will be given in this section along with the assumptions made for that step. It has to be known that QGIS, an open-source geographic information system program was used to visualize the geographic data and to extract some geographic properties of the data, namely distance measures and closeness.

In Figure 20, the yellow boxes are the python scripts or the QGIS functions that were used to generate the piece of data that is included in the blue diamond. It can be noted that 8 actions were taken to achieve the final csv file that includes all features.

The outputs that will be shown belong to the case study of Utrecht-Den Bosch, namely the sampling done in January 2019. The reason behind that is to better understand the methodology and to be able to see the output after every step.

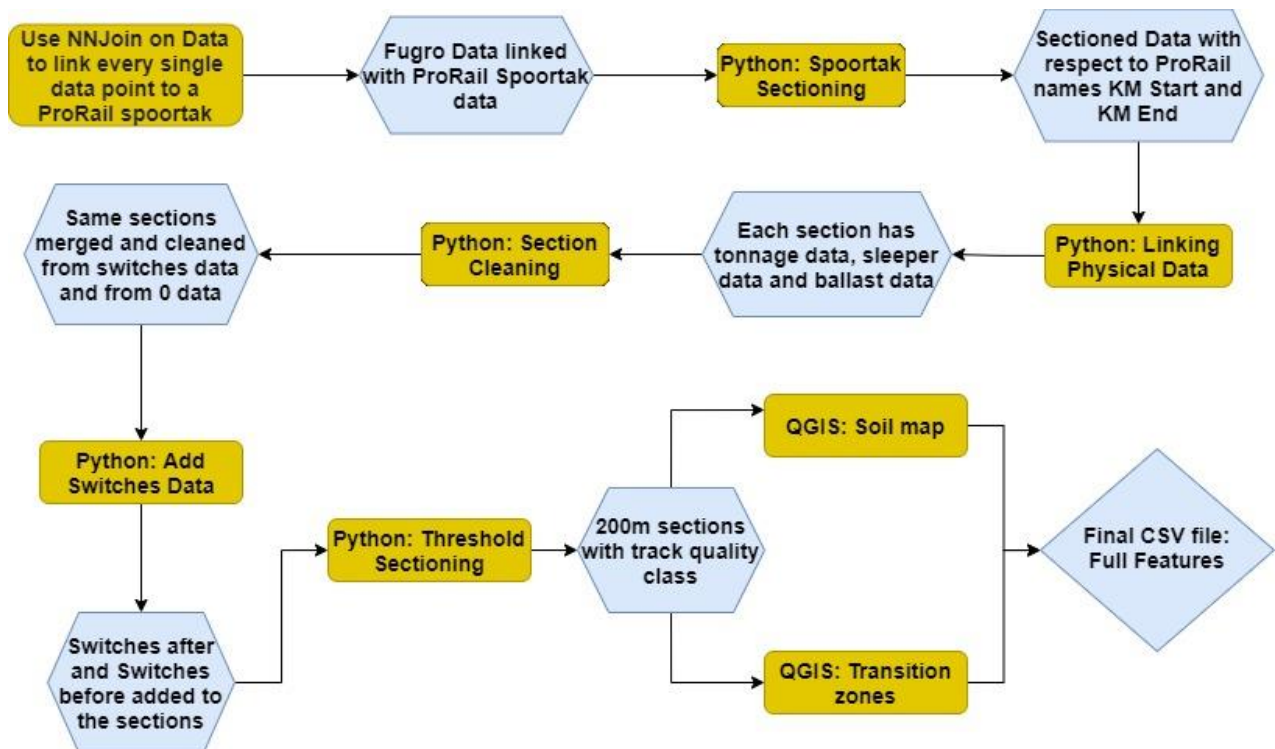


Figure 20: Data Gathering Roadmap where yellow boxes represent the method/tooling used and the blue boxes represent the output from said tooling

The raw data is first and foremost extracted by the train-borne system RiLA as discussed in the previous sections. This raw data is then processed to give the CEN approved RTG data that is a set of data spaced by 0.25m intervals and includes longitudinal level data and alignments in the D1 region for the left rail and the right rail each, which is 4 features. Each data point includes an X and a Y coordinate that can be uploaded on QGIS to visualize the track that is in study. Moreover, a value of a RiLA Chainage is attributed to each point which is nothing but a linear reference that is along the track. Next, the centered moving standard deviation values for alignments and longitudinal levels were calculated at every single data point in a manner that said data point is the center of the window for which the standard deviation will be calculated. This window is of 800 points, or 200m. To better understand this, consider the following set of data: [1, 2, 3, 4, 5, 6, 7, 8]. The moving mean using a window of 3 returns: [1.5, 2, 3, 4, 5, 6, 7, 7.5]. The first and last points are not able to be the centers of a window of 3 points thus the first two points were used instead for the mean. In the same manner, the moving standard deviations were calculated for every single point with a window of 800. This allows for every point to have an overview of its values and that of its sections and neighboring sections which are all impacting its quality. This will help push the very reason this project was made in the beginning, and to better relate sections to their neighboring sections using data and structural elements. Nevertheless, the averages of the left rail and the right rail were added to the input table. This input table is included in Appendix A: Utrecht-Den Bosch Project Sample Data.

Figure 21 shows the Utrecht-Den Bosch track on this software wherein the points can be seen well enough. The points belong to the track centerline and the reason that two tracks are shown is that the survey that was carried out in 2018 by Fugro belonged to Utrecht-Den Bosch and Den Bosch-Utrecht in that order of direction.

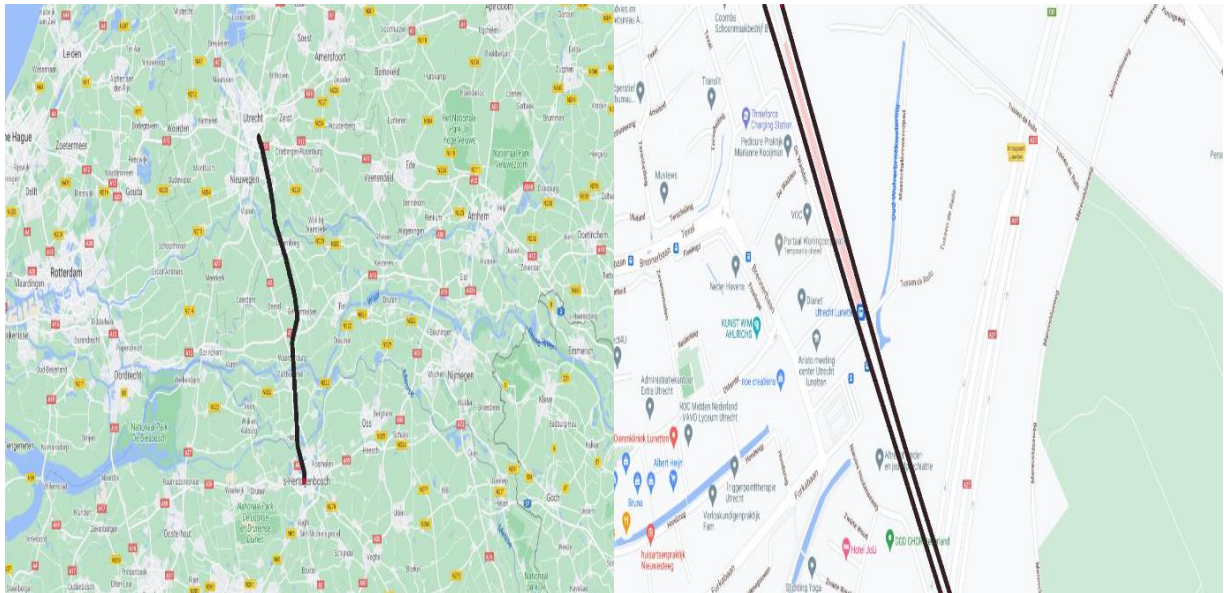


Figure 21: Utrecht-Den Bosch data points shown on QGIS

The rail branch data that was uploaded from the ProRail map services online, included the entire railway branches in the Netherlands. This was linked to the QGIS program and the “spoortak” shapefile was loaded. Each rail branch element included attributes such as the name of the branch as discussed above, the KM start and the KM end of the branch as well. In fact, this shapefile also includes the switches. Thus, the type of branch is also part of the attributes. Figure 22 helps better visualize the shapefile in QGIS.

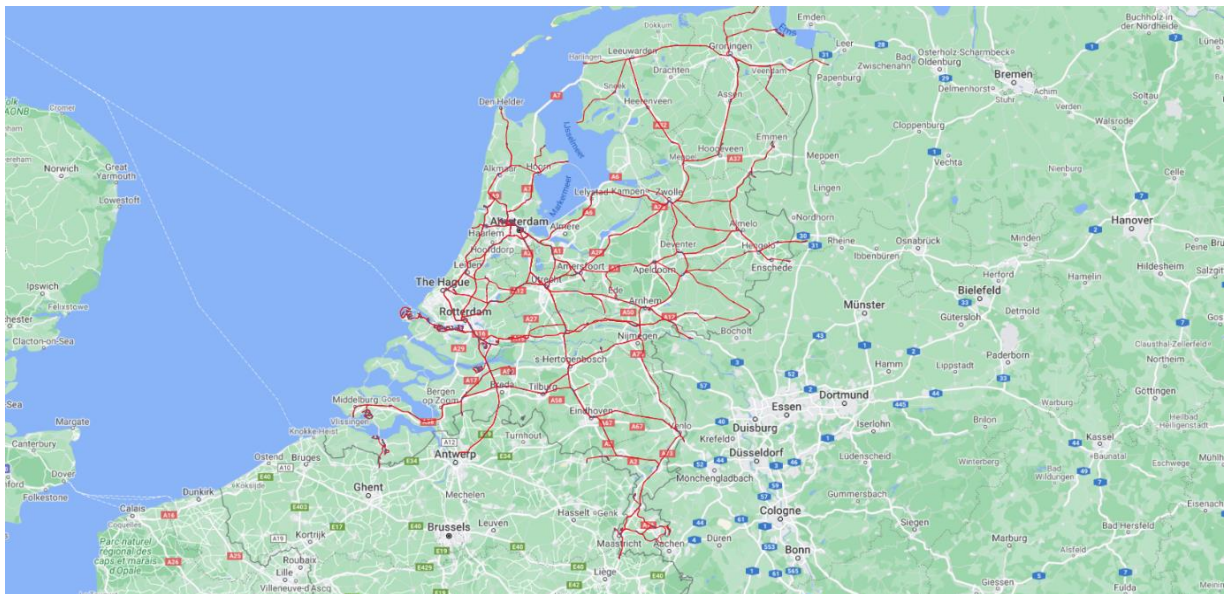


Figure 22: Spoortak shapefile in QGIS

1. The first step was to use the NNJoin plugin which can be downloaded freely online on <https://plugins.qgis.org/plugins/NNJoin/>. This plugin basically takes two input shapefiles, which in the case of this project are the points as shown in Figure 21 and the rail branches as shown in Figure 22. The algorithm looks at each point and calculates the nearest distance to a rail branch and adds its attributes to that point. Thus, the end product is the same table shown in Appendix A: Utrecht-Den Bosch Project Sample Data, but added to each data point, is the rail branch properties that it belongs to such as the type: rail branch/switch, the unique name of the branch

as discussed above and the KM start and KM end values as shown in Appendix B.1: Table showing the linked data using NNJoin. This is how the Fugro data is now linked with the ProRail data.

2. Next, the Python script entitled: Spoortak_Sectioning was applied. This script feeds into the newly linked data and looks at all the successive points that belong to the same rail branch in terms of type, name, KM start and KM end values and adds aggregates them together while saving the data numbering of the first point and the data numbering of the last point of that section for future use. The start RILA Chainage and end RILA chainage of the section are included for reference as well. Also, the section length is calculated and included. It has to be noted, that from hereon, the points that belonged to switches were removed from the data after the sectioning has occurred. The Python pseudo-code is included in Appendix B.2.1: Spoortak_Sectioning Pseudo-Code, and the sectioned data is shown in Appendix B.2.2: Spoortak_Sectioning Output Table (Utrecht-Den Bosch). It is important to note that some section names are repeated, but what differs is the KMstart and KMend values (refer to section 3.2). 351,852 data points are turned into 112 sections.
3. Every single section was now eligible to have the physical object data linked to it using the name of the section, the KMstart and KMend value of the section. The ballast, sleeper, rail and tonnage data each are in different csv files. What is common between all those files is that each rail branch is given information about those objects using the unique name and the KMstart and KMend as identifiers. Thus, for each rail branch in the project at hand, attributes were given using just those identifiers. The ages of the physical objects were calculated using the placing date of the object as a reference and the date of the sampling as the present date: 1st January 2019. Some ages are negative, which means that the element has been placed after the survey, this is not a problem as it gives us insight that it had to be changed probably due to a long age. It has to also be noted that each element to be added had a different Python script that fed into a different csv file, however all the scripts worked in the same way. After linking all of the relevant data from the ballast, sleeper, rail and tonnage data, the table in Appendix B.3.2: Physical Object Linking Table (Utrecht-Den Bosch) is generated. Two things can be inferred from this table, some sections that have the same name and different KM start and end values have the same exact features, and some sections have no features at all. Those two problems will be solved by the section cleaning Python code. Yet, it is Important to explain why some data has no attributes. This is due to the fact that the physical objects data was updated before the station of Geldermalsen was renovated. Thus, the new rail branches that were created in this renovation cannot be found in the physical objects data. The survey that Fugro carried out in January 2019 was on the renovated tracks. Thus, those specific tracks have no attributes and will be dealt with in the section cleaning script.
4. The sections that have the same name and same features are to be combined, despite the different KM start and KM end values. Also, the values with no features have been simply removed. From 112 sections, we now have 43 sections. The start points and end points indices made combining the sections simpler. This can be better understood with the pseudo-code included in Appendix B.4.1: Section Cleaning Pseudo-Code. The output table is also shown in Appendix B.4.2: Section Cleaning Output Table (Utrecht-Den Bosch). The interesting thing to note is that the start points and endpoints are also combined, also the RILA chainages. The start point of the combined section is the start point of the first section that was combined and the last point of the combined section is the end point of the last section that was combined. This works the same for the chainages. The section lengths are added to get the combined section length.

5. Each section name contains before and after information in the 'spoortak' shapefile. In the attribute table, a column of "begrenzert" and a column of "begrenze_1" describes what is above (north) the rail branch and below it (south). It can be a switch or it can be just nothing. To know whether this above or below corresponds to before or after in terms of direction, it is important to understand the directions of the railway tracks. In fact, from 0 to 44 km of railway chainage, the railway corresponds to the upward bound (Den Bosch to Utrecht) rail which as seen in figure 21 is the left railway line. From 44km of chainage to 88km the railway goes from Utrecht to Den Bosch and is southward bound. Using this information, it was straightforward to write the add_switches script which is highlighted in the pseudo-code included in Appendix B.5.1: Switches Adding Pseudo-Code. The switch data that was added is the slope of the switch, and the type of the switch and the name of that switch for reference. The reason that this has been done in a direction sensitive manner is that having a switch before a rail branch affects the rail branch more than the opposite. This is due to the fact that the crossing nose of the switch will hurt the rail and wheel and impact the track branch after it.

If a section contains no switches before or after, the values will be 0. To better understand this, see Appendix B.5.2: Switches Adding Output Table (Utrecht-Den Bosch), where the cleaned sections have the switches, data added.

The table so far contains the physical object data and the tonnage data for each section along with the data pertaining to the switches before each rail branch and after each rail branch. The sections that we have so far are 43 with the maximum section length being of 6539.75 m. It has already been established from literature that the influence length or the length of the sections will be of 200m for track quality evaluation. The section thresholding python script will thus feed into the table that is shown in Appendix B.5.2: Switches Adding Output Table (Utrecht-Den Bosch) and the table that is included in Appendix A: Utrecht-Den Bosch Project Sample Data and using the start and end point indices will break down the sections that are above 200m into 200m sections and a remainder section. Each section will have a standard deviation value are calculated for the longitudinal levels and alignments and using the tables shown in figures 10 and 11 will qualify the track into an A,B,C,D,E track quality pertaining to the worst of the two. However, the method of calculating the standard deviation has to be chosen and justified. This script uses the endpoints and the startpoints to track the RTG data and do the discrete standard deviation.

This script also looks at the averaged left rail and right moving standard deviation values and takes the average of those values per section also using the endpoint and startpoint indices. The track quality class characterization was made using the average of the moving standard deviation values and not other methods. The figures 23, 24, 25 will highlight the difference between the methods.

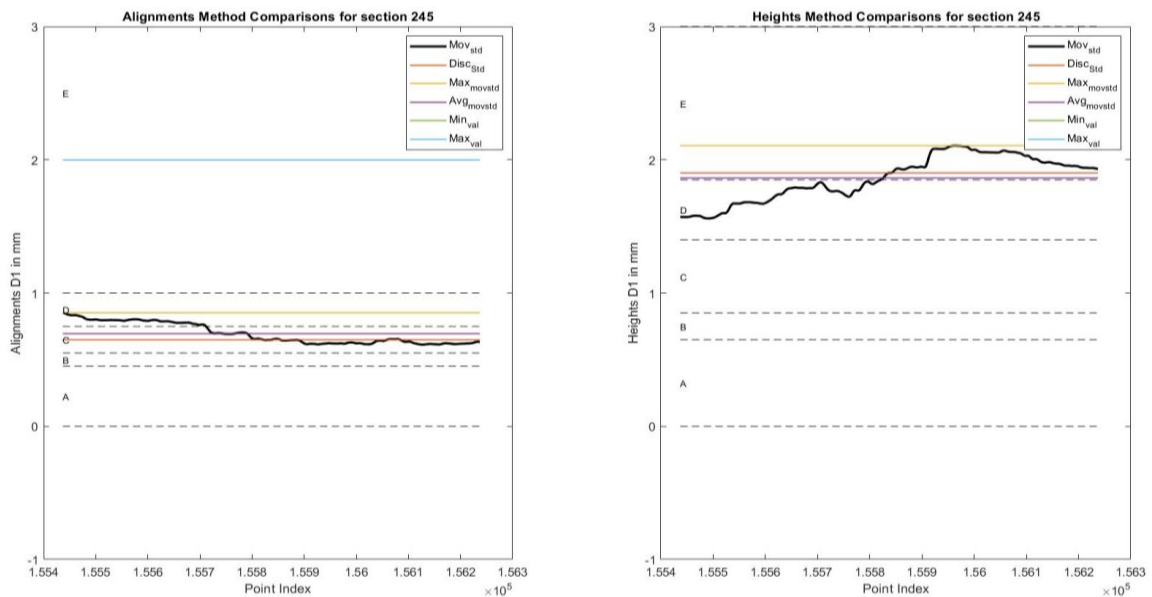


Figure 23: Standard deviation calculation methods compared for section 245 out of 429. The pointed lines show the track quality class decision boundaries from EN13848-6 using velocity range of 120km/h to 160km/h. The x axis represents the indices of the points from table of Appendix A.

Figure 23 shows first and foremost that the average of the moving mean is quite close to the discrete standard deviation. Another thing to note is the longitudinal level or height data is higher than that of the alignment, which also shows up in the different ranges of track quality classification of both of them and confirms the fact that the worse track quality case is used for classification of a section. Another thing to note is that the maximum of the moving standard deviation is quite a conservative method, and in some cases can give different results such as in Figure 23, the alignment track quality class would be D using the maximum value, and C using

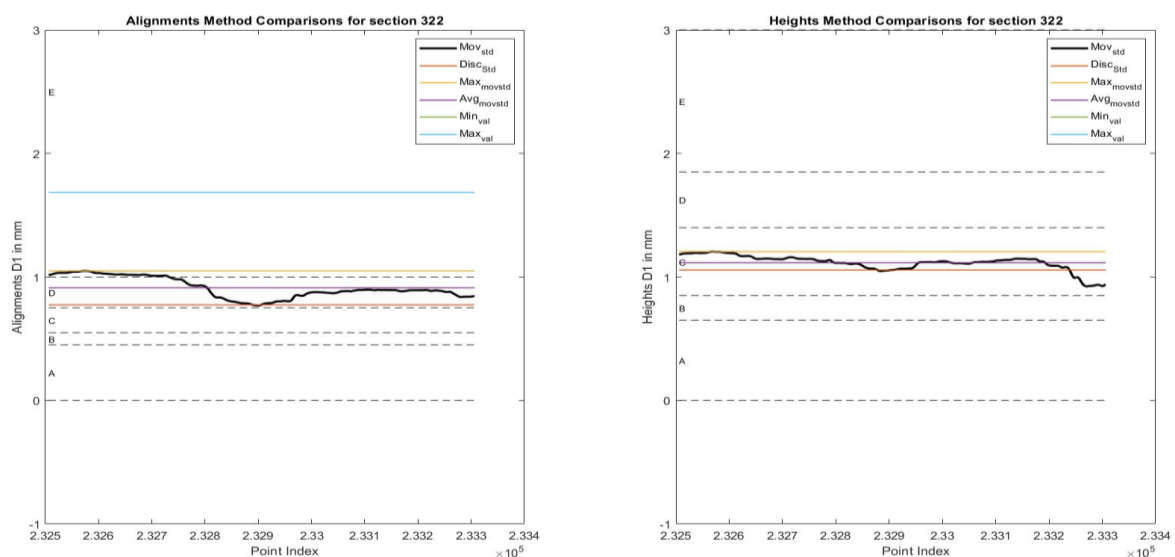


Figure 24: Standard deviation calculation methods compared for section 322 out of 429. The pointed lines show the track quality class decision boundaries from EN13848-6 using velocity range of 120km/h to 160km/h. The x axis represents the indices of the points from table of Appendix A.

either the discrete or the average method. In the case of the heights for section 245, all methods give E track quality. This section will be classified as “E”, as the worst case of both.

It has to be noted that in some cases the discrete standard deviation values are quite different from the average, however they would still give the same decision (Figure 24). This all depends on the maximum values and the minimum values in a section. The closer these are to each other, the closer the results of each calculation method (Figure 25).

The maximum method of calculation is however much more conservative and would always exaggerate the track state, due to some local anomalies in either this section or the neighboring ones. Thus, for a more general and complete overview of the track quality, the average of the moving standard deviation values in a section was used.

Now back to the sectioning, if a section is of 5428m for example, it will be broken down into 27 sections of 200m and a section of 28m which is the “remainder” section or the last section. The remainder sections are important for the analysis despite their length not being the same as the chosen influence length. They provide us with some variability in the model.

The broken-down sections will take all the attributes from the “mother” section except for the switches data. The before data and after data of the switches will be transferred respectively to the first 200m section and the last section. The reason why 200m sections have been chosen and not 100m sections is clearer now: the sections that are being broken down all have similar attributes and thus low variation between them. If 100m sections are used then more sections with similar attributes will be there and will thus affect the machine learning algorithm in a bad way decreasing the variability of the data. Check Appendix B.6.1: Section Thresholding Pseudo-Code for the pseudo-code and Appendix B.6.2: Section Thresholding Output Table (Utrecht-Den

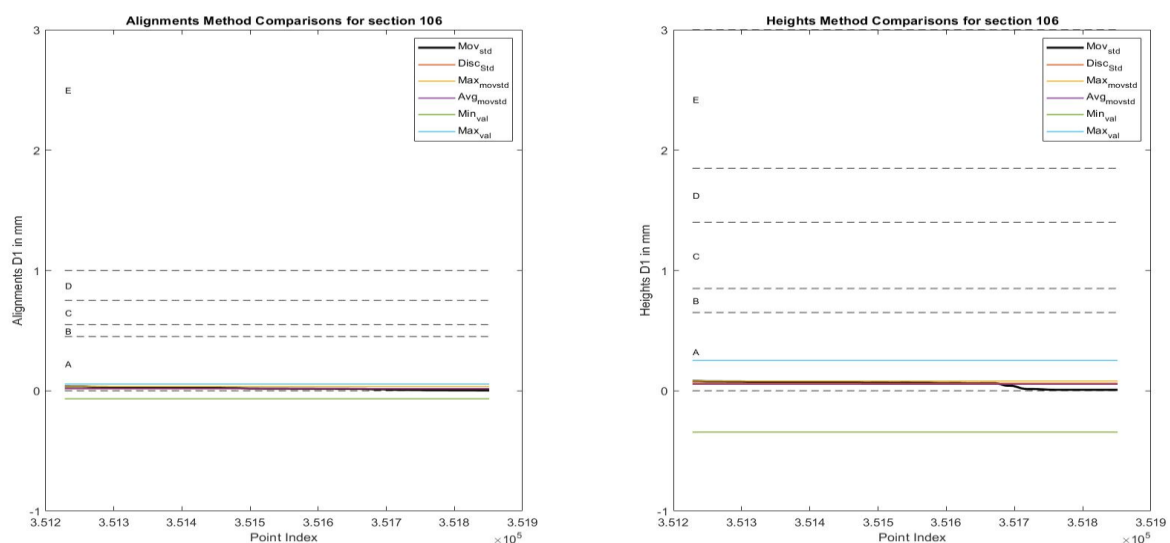


Figure 25: Standard deviation calculation methods compared for section 106 out of 429. The pointed lines show the track quality class decision boundaries from EN 13848-6 using velocity range of 120km/h to 160km/h. The x axis were included to later upload and visualize the sections on QGIS. The midpoint's coordinates were not the average of the first and last point in the section, instead the indices of the start and endpoints were averaged to find the actual midpoint that exists and those coordinates were used. This is to avoid having any coordinates that are outside the track due to a curve for instance.

6. The soil map of the Netherlands was uploaded to QGIS and the colors in Figure 26 belong to a different soil type each. Using QGIS the midpoints of each section were uploaded. A function included in QGIS is called “add polygon attributes to points”. This function, using the location of

the points adds a certain attribute (soil type) from the polygon (soil map) it is located on. Thus, using this function every section now has a soil map. Obviously, this is not a very accurate way of adding the soil type to the sections due to the midpoint method, however it's an assumption that made the data automation easier. The result is extracted as a csv file.

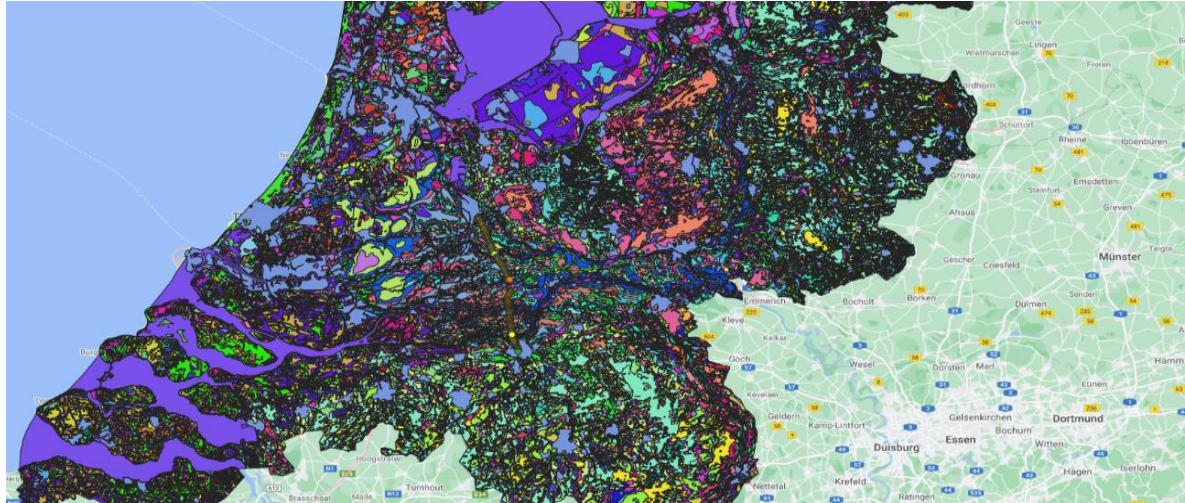


Figure 26: Soil map of the Netherlands in QGIS showing all different soil types and the midpoints of the sections

Once each section has been qualified with a soil type, the next step is to quantify the soil type into two parameters: deformation and bearing capacity as per Schoorl, (2021). The general method used to quantify each soil is based on looking at the soil types and using their respective cohesion factors and angles of friction. However, to get the soil-specific parameters, some CPT test results were used to get the subsequent cone resistances, sleeve friction values, and penetration pore water pressure values. Once the angle of friction was evaluated, the bearing capacity factors can be evaluated. These are used to get the ultimate bearing capacity values based on the failure mode of a heavily stressed part of the track. Assuming the failure mode is important is it leads to a different assumption of the footing width to be used, which is a third of the sleeper length, thus 87 cm. Then, once the bearing stress has been calculated, the stress at the subsoil surface needs to be determined, and this was done using the pyramid model that assumes the stress distribution from a surface to its depths is trapezoidal in 2D and pyramidal in 3D. Schoorl (2021) argues that the angle of dispersion is of 35°, and goes on to estimate the areas through which the load disperses. In short, using the stress values at the surface of the subsoil along with the ultimate bearing load that was derived from the soil parameters, a value called bearing capacity safety factor (BCSF) is derived by dividing the two values by each other. If the ultimate bearing capacity is twice or above the actual stress, then the quality is desirable. Anything less, makes the quality worse on a specific section (Schoorl, 2021).

As for the deformations that were included, an approximation of the subsoil's flexibility coefficient was used to include in the Winkler beam model, which is an approximation of how the railway system acts on the soil. In short, the Winkler model is a beam on an infinite elastic support with the ordinary differential equation:

$$EI * w'''' + k * w = 0$$

Where E is the young's modulus of the beam, I, is its moment of inertia and k is the stiffness of the elastic support under the beam which encompasses all the layers including the subsoil layer. This value of k is calculated using all other layers' stiffness values by assuming a system in series.

The values of E and I were taken as that of the rail and the value of k was estimated using the cone resistance values and the sleeve friction values. The model will be solved using passenger train axle values as the point load.

The deformations and the bearing capacity of the soil should provide a good insight for the model about the strength/weakness of the subsoil.

It has to be noted that the case study used is mostly composed of clayey subsoils, which led to high deformation values and low BCSF values in certain areas.

Several assumptions were made to quantify the subsoil strength (Schoorl, 2021):

- The bearing mode of failure
 - The stress transfer method and geometry
 - The approximation of the stressed areas
 - The values of EI for all rails to be the same
 - The approximation of the value of k as a system in series
 - The usage of the Winkler model with a point load
 - Interpolation of the missing values where CPT results were not found
 - Assuming saturated soil conditions and thus using the saturated volumetric weight for calculations
 - Assuming the DAF values to calculate the axle load used for the Winkler Model
7. Culverts, bridges, level-crossings and tunnels were all added into QGIS and merged into one shapefile called: transition zones, see Figure 27. The midpoints of the sections were also uploaded in QGIS.



Figure 27: Transition zones in the Netherlands, blue is for culverts, green for tunnels, beige for level crossings and orange for railway bridges

For each section midpoint, a buffer with half its length was drawn. This means that for each midpoint a circle is drawn with the midpoint being its center and the section length being its diameter, see Figure 28. Each buffer represents the entire section at hand. Afterwards, using the “count points in polygon”, for each buffer representing a section the number of transition zones contained is counted. This is extracted later as a csv file.

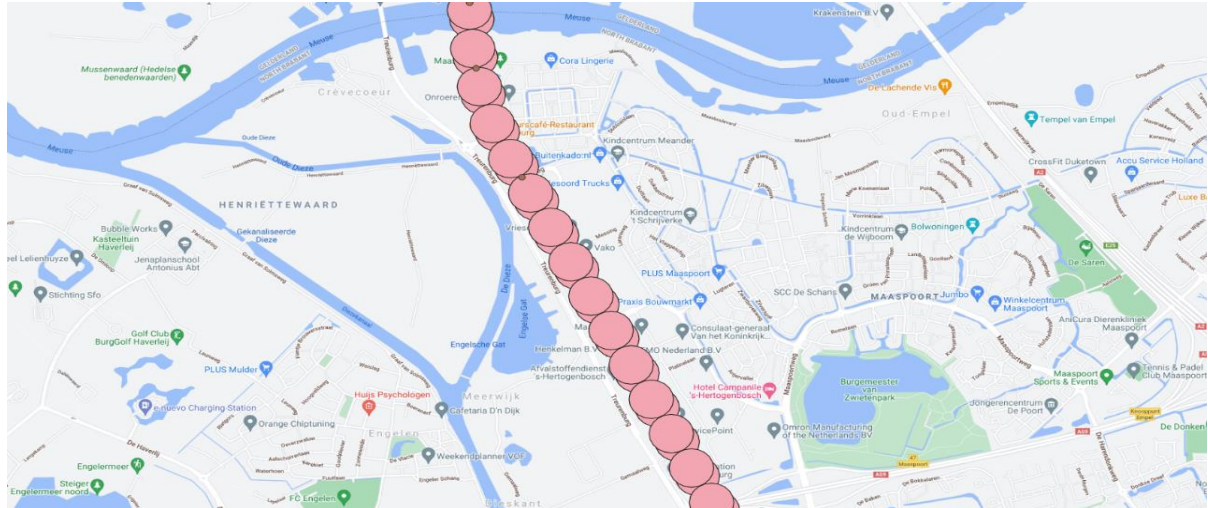


Figure 28: Buffers shown

The features are all thus gathered in a semi-automatic manner using Python and QGIS built-in functions. The next section will take a look at the data that has been gathered and analyze this data in terms of descriptive statistics and pre-processing for the Machine Learning phase where the combined track quality will be predicted using all the features that were gathered in this section.

4 Machine Learning Classification

In this section, the data gathered for the Utrecht-Den Bosch line will be used to train a machine learning classification model using the three previously defined algorithms. The features used to predict track quality will be looked at in an exploratory manner in first place and then the pre-processing methods that were used to prepare for the model running will be highlighted. Once these preliminary steps have been made, the questions that the machine learning algorithm needs to answer will be discussed and related to the research questions.

4.1 Pre-Processing

First and foremost, it is important to start by defining the variables that will be in use in the classification algorithm. Table 3 shows the features that were gathered and their following data type.

Table 3: Features used, their data types, definition and examples

Feature Name	Data Type	Definition	Examples
Speed_Pass	int64	Passenger Speed in km/h	140 km/h
Tonnage_To	float64	Daily Tonnage Value	4910 tons/day
Ballast_Ty	object	Type of the Ballast	Steenslag 31,5/50 Geen ballast
Ballast_Ag	int64	Age of the Ballast in days	1766 days

Ballast_Th	int64	Thickness of Ballast	5cm ,17.5cm, 35cm,
Sleeper_Ty	object	Sleeper Type	NS 90 SKL-14 Duoblok rechtstand klemhouder Hardhout rug(helling)plaat
Sleeper_Ag	int64	Age of the Sleeper in days	1766 days
Rail_mat	object	The material that the rail is composed of	R260Mn (oud 900B)
Rail_profile	int64	Rail Profile	540
Rail_age	int64	Age of the Rail in days	1766 days
Switch_b_1	float64	The slope of the switch before the section	1/9=0.1111, 1/15=0.0667
Switch_a_1	float64	The slope of the switch after the section	1/9=0.1111, 1/15=0.0667
Transition_Count	int64	The number of transition zones within the section length	4
Deform_Soil	float64	The deformation value in mm calculated for the specific soil type in the section	9.76555
BCSF_Soil	float64	The bearing capacity safety factor (unitless)	3.456

The data types are quite important for the pre-processing part. The object data types are lists of strings. They need to be changed into a binary format. For instance, for the type of ballast, instead of just having each type in one Feature Name called “Ballast Type”, this will be changed into X columns where X is the type of ballast that is included in the original column. We would then have Ballast_Ty_GeenBallast and Ballast_Ty_ Steenslag columns, with binary values to indicate what the ballast is composed of. Values of 1 and 0 respectively would mean we have no ballast or “geen ballast”. This method is called one hot encoding, and Python allows for its usage very easily by calling the pandas function of “get_dummies.” To better understand this process in action, Figure 29 helps us visualize the object data type of color that includes colors of red, yellow and green being handled in the same manner.

Color		Red	Yellow	Green
Red		1	0	0
Red		1	0	0
Yellow		0	1	0
Green		0	0	1
Yellow		0	0	1

Figure 29: Hot encoding of object features, <https://www.kaggle.com/dansbecker/using-categorical-data-with-one-hot-encoding>

It is obvious that each data point has only one “1” value as its color cannot be both red and yellow and green at the same time. The same applies for the object data types that are included in this project.

In a second manipulation, it has to be noted that the Ballast Thickness feature and the slopes of the switches were previously objects as well. The ballast thicknesses were of “15-25cm”, “<15cm”, and “>25cm”. These were handled and changed respectively into 20cm, 5cm, and 35cm as numerical types for ease of processing. As for the slopes, these were also object data in the form of “1:9” or “1:15”. Instead of having them in the said string formats, they have been turned into their float values. Also, some ages values were negative, signifying that the respective elements were replaced just after the RiLA data measurement. To handle this, those values were replaced by 5 times the maximum age that is exhibited in the table for that specific element. This is obviously an assumption, that is based on the fact that replacing an element just after the measurement could mean that that element has reached its lifecycle end, or was defective. Thus, giving it a bigger age than other elements in the table would tell the machine learning algorithm that this should be considered negatively and account for a worse track quality.

The reason this is important is that the Xgboost and Random Forest algorithms allow the usage of object data along with numerical data (integers or floats) as features, however the object data types have to be handled as shown above. The SVM algorithm on the other hand only takes numerical data as features, thus the object data types were not used for the learning. That numeric data for the SVM algorithm also need to be scaled as not to create any bias towards values that are naturally higher such as tonnage vs ages. Without scaling tonnage values would decide on their own the track quality class as they are much larger than other numbers used. The method used was the standardization method, which takes all the values in one column and transforms them into a different scale using the below formula:

$$Z = (X - \mu) / \sigma$$

Where Z is the new value of the data point, X is the old value, μ , is the mean of the values and σ is the standard deviation. Doing this for all the features results into scaled values and no bias for the machine learning process.

One last thing that was done before delving into the model training, was that for the Xgboost and SVM algorithms, a label encoder was used for the track quality classes, turning A into 0, B into 1, C into 2, D into 3 and E into 4. The reason behind that is that these algorithms unlike the random forest which in this section has already shown its flexibility, cannot label string values in the form of “A”, ..., “E”.

4.2 Exploratory Statistics

Once those object data types were handled, the correlation matrix was plotted (Figure 30) to show the correlation between existing features and whether these correlations make sense. This is what is meant by exploratory statistics. An exploration on the data linkages is made and explained.



To remind the reader of what correlations are, imagine a line of $x=y$. This is the most perfect example of perfectly correlated data. The correlation value here would be of 1, it signifies that both variables move in the same direction, i.e., when one increases or decreases so does the other. The correlation coefficient is thus a value that shows whether the change of values of a variable is associated with that of another. A correlation can be negative or positive also. If we have a correlation of -1, it means that when one variable is high, the other is low.

- Passenger speed is correlated with none of the other structural features. We would expect that this value is correlated to the switch slope (before or after) as, the steeper the slope (1:9), the lower the allowable speed, as it is harder to negotiate with higher speeds. This leads to doubting the passenger speed data.
- The daily tonnage was only found to be mildly correlated to the rail age (0.62). This could be logical and understood in the fact that the older railways are the more dependent the timetable is on them, and the more they are used. An old railway could have more traffic on it than newer railways. However, this provides a rough explanation to this correlation and leaves us with small doubt.
- The most logical finding was the fact that the ballast age was highly correlated (0.77) with the sleeper age. This is due to the fact that when the ballast was placed and added to the track,

the sleeper was too, at the same time. Questions could be asked on why the rail age is not correlated to both those elements, which also could lead to doubting the data.

- The ballast thickness was found to be highly negatively correlated with the double block sleeper type (-0.7). This means that the lower the ballast thickness, the more those types of sleepers were used. This is a very logical finding and can be explained by the fact that duo-block sleepers provide a higher lateral and bearing resistance in the ballast bed due to the doubling of the surface area under the rail (Markine, 2020). This leads to needing less ballast.
- Transition zones were not correlated with any of the other features, which makes sense. Although soil deformation should be correlated to that feature, due to the differential settlement that the transition zones engender. However, the deformations that were calculated only looked at the soil properties and didn't touch upon any special structures in the environs.
- As expected, the sleeper types are negatively correlated with each other as well as ballast type and any other object data type turned into binary, because if one is present the other is absent for sure. You cannot have a duo-block sleeper and an NS90 sleeper at the same time for instance.
- The rail materials R260Mn and R260 are correlated with the concrete sloped sleeper which is not the new NS90 sleeper. The difference between the two materials is the chemical composition in that the R260Mn contains much more manganese than R260 ("Mechanical properties of Rails - ArcelorMittal Rails," 2018). Manganese in steel increases the strength and decreases the brittleness of the material (Cannon, 2014). The concrete sleeper concerned is applied in slopes ("Betondwarsligger, m rughellingplaat 54E1," 2021). In a slope, there is a redistribution of forces that occur, and the two rails are not sharing the load in an equal manner anymore. This extra load on any rail could lead it to brittle failure, which is why it makes sense to have the combination of the R260Mn rail material along with the concrete sleepers applied in slopes.
- The deformation of the soil and the bearing capacity safety factor are negatively correlated (-0.61). This is logical because the higher the deformation, the weaker the soil and thus the less bearing strength it exhibits. Any other correlation with those two features cannot be explained as none of the factors were used to calculate the values.

Not all the correlations were as expected, and this could be due to a slight fault in the data gathered. In the next sections it will become apparent that the over-reliance on the ProRail data is an Achilles' heel to this project.

4.3 Questions to answer with the model

When constructing a model, it is always important to delineate the questions that the model aims to answer. In this case, the research questions are quite important to have answered. However, in order to tune the algorithms to the maximum an extra set of questions are added:

1. Which algorithm is the best in terms of prediction accuracy?

The accuracy measures for classification will be defined in the next section, along with a comparison of each. Once the accuracies of each tuned algorithm have been calculated, an aggregation of the labels from A,B,C,D,E to Good, Medium and Bad will be made to better the prediction scores. The usage of only 200m sections and 200m sections and remainders will be also evaluated. Thus, 4 comparisons will be done.

2. What hyperparameters will be used for each algorithm?

Once the most accurate methods have been chosen, the algorithms will be tuned to try and increase the accuracies of predictions. A hyperparameter is a parameter that defines the algorithm. For instance, in the random forest, a set of hyperparameters can be chosen and tuned accordingly. One of them is the number of trees in the forest, “n_estimators”. These parameters can be tuned in a random fashion and chosen iteratively once the best accuracy is revealed.

3. What features are the most important for predicting track quality?

The Random Forest algorithm allows for showcasing the most important features that defined the track quality. This insight will be used to better understand the structural features’ importance towards track quality and to critique the model if it opposes what has been read in the literature.

4. How to validate the models?

The models will all use the cross-validation method for accuracies once the best hyperparameters have been chosen. This method will be explained in the results section

5. How to infer the model?

This question relates to testing the model on new-found data and whether it can be generalized. It can very well be that the model performs very well on the test set, however once exposed to a new track, it does not perform well due to bias and overfitting. The models will be tested on already seen data by re-training the model on the 44km track going from Utrecht-Den Bosch and try to predict the track qualities of the other way. This will give insight on the generalization capabilities of the built models and the research method. In a second test, the data will cut off two 5.5 km tracks in both ways to check whether the model can generalize to two-way tracks that are similar in features.

In a third manner, the model will also be tested on a 7km track from Den-Bosch to Eindhoven whose subsoil is mostly sandy to check whether the model can generalize its predictions to features it has never seen. This is an important check as this 7km track will undergo the same data gathering procedure which would also lead to validating the algorithms that were harnessed for the Utrecht-Den Bosch case study.

5 Results and Discussion

To recall, after the section thresholding algorithm, some remainder sections were generated. These remainder sections have also been qualified with their respective track qualities. Their section length is less than 200m. These sections will be used for the classification algorithm to be compared against using the 200m sections alone. In a different subject, the track qualities A,B,C,D,E will be aggregated into three qualities: Good, Medium and Bad in a manner that A,B tracks are the good tracks, C the medium tracks and D and E the bad tracks. This is also done to compare the aggregated manner of track qualities with the original manner of doing so. When comparison is mentioned here, it is in terms of accuracy.

5.1 Performance Measures

Speaking of accuracy, the accuracy measures that are widely used in the Machine Learning classification world should be defined. It should be reminded that the model training takes 70% of the data and trains it in a manner to find the variations in features and the patterns and how they relate to the labels. Once the training has been done, the 30% of the features that are left are used to predict the labels (track quality). These predicted labels are compared with the real or true labels and a confusion matrix is drawn. A few parameters derived from the said confusion matrix such as the accuracy, the precision, the recall and the f1score are all quantifications of the efficacy of the predictions ("Classification: Accuracy | Machine Learning Crash Course," 2020). To better understand these terms an example will be given on fabricated results.

		PREDICTED LABELS				
		A	B	C	D	E
TRUE LABELS	A	2	0	0	3	0
	B	0	0	1	0	0
	C	0	0	20	18	0
	D	0	0	9	21	4
	E	1	2	5	14	6

Figure 31: Example Confusion Matrix

In Figure 31, the numbers shown on each matrix entry represent the number of predicted values. The first entry (1,1) is the number of predicted values as A when A was the real label. The second value on the first row: 0, represent the number of predicted values as A when the real label was B etc...

The diagonals represent the numbers of the accurate predictions. And thus stems the first measure of prediction efficacy: **accuracy. It is the number of labels correctly predicted divided by the total number of observations.** The total observations in this example are: 106. The accuracy of the model is thus: $(2 + 0 + 20 + 21 + 6)/106 = 49/106 = 46.22\%$.

To better understand the notions of precision and recall and f1score which is just a derivation from the previous two measures, the notions of false positive/negative and true positive/negative need to be defined. The term negative and positive refer to the prediction itself. For A: the positive values refer to the number of values predicted as A, correctly or incorrectly. While the negative values are all the values that have been predicted as not A, also correctly or incorrectly. The correctness or incorrectness of the predictions are where the terms: true and false come from, respectively. To better understand this, the calculations for the value of A will be made below:

True Positive (TP) = 2. These are the number of labels predicted as A and are truly A.

True Negative (TN) = 1 + 20 + 18 + 9 + 21 + 4 + 5 + 14 + 6 + 2 = 100. These are the number of labels predicted as non-A and are truly non-A.

False Positive (FP) = 1. These are the number of labels predicted as A but were not truly A.

False Negative (FN) = 3. These are the number of labels predicted as non-A but were truly A.

The sum of TP+TN+FP+FN is the total number of labels in the test set.

$$Precision (\%) = TP / (TP + FP);$$

$$Recall (\%) = TP / (TP + FN);$$

Precision represents the ratio of a correctly predicted label divided by everything that was predicted as that label, it thus shows the proportion of positive identifications that were correct. While recall represents the ratio of all the values correctly predicted as a label to all the true values of that label. It thus aims to show the model's ability to correctly predict true positives. Each label has thus a precision and a recall value, and the closer they are to 1, the better the model. It has to be noted that these two measures come at a tradeoff between each other ("Classification: Accuracy | Machine Learning Crash Course," 2020).

The f1-score, is nothing but the harmonic mean of the precision and the recall. The closer to 1 it is, the better the model's performance.

$$F1score (\%) = 2 * (precision * recall) / (precision + recall)$$

To make everything clear, these metrics will be calculated for the label A.

$$Precision_A (\%) = 2 / (2 + 1) = 66.67\%$$

$$Recall_A (\%) = 2 / (2 + 3) = 40\%$$

$$F1score_A (\%) = 2 * (precision * recall) / (precision + recall) = 50\%$$

In terms of macro-performance, thus to evaluate the entire model and not just do this exercise for each label, accuracy is used, as well as averages of the precision, recall and f1score, either in a weighted fashion or in an arithmetic fashion as shown in Table 4.

Table 4: Model performance measures summarized

	precision	recall	f1-score	support
A	0.67	0.4	0.5	5
B	0	0	0	1
C	0.57	0.53	0.55	38
D	0.38	0.62	0.47	34
E	0.6	0.21	0.32	28
accuracy			0.46	106
macro avg	0.44	0.35	0.37	106
weighted avg	0.52	0.46	0.45	106

The support values represent how many sections are really a certain track quality: C was the most common track quality in the test set (38). From Table 4, we can conclude that the model performs poorly with an accuracy of 0.46. The best predicted labels were A and C, with f1-scores of 0.5 and 0.55 respectively. This insight tells us that the features data is not varied enough to be able to tell between the labels. It would be a good idea to aggregate the classes (A,B into good, C into medium and D,E into bad) and see what happens to the overall quality. If the performance increases, it would tell us that the low predictions come from not being able to tell between the track qualities that are close to each other such as A and B. This comes again from the lack of variability between the features.

5.2 Results: Performance Measures

In this section, the results of each of the three models will be analyzed. Four different methods will be baselined against each other and the most accurate results will decide the method used for each algorithm. The methods include the aggregation of the track quality classes into 3 classes or 5, and the usage of 200m sections or 200m sections along with the remainder sections. Moreover, a comparison of each model accuracy will be made, as to answer the third research question. The results will also be analyzed and an explanation will be made about them.

It has to be noted that after outlier removal, 424 sections were available. Using the 70-30 split for training and testing sets, the training sections are of 318 and the testing sections are a total of 106.

The first comparison will look at the method of aggregating the results. The used sections were of 200m. Only the method of aggregating the results was varied to provide a comparison. Appendix C.1: Performance results of varied track quality aggregation methods will show the table of performance measures for both methods and each algorithm. Thus 6 tables are shown there. The analysis of the result for now will just focus on each algorithm comparing the performances between the methods and not look at an inter-algorithm comparison, just yet.

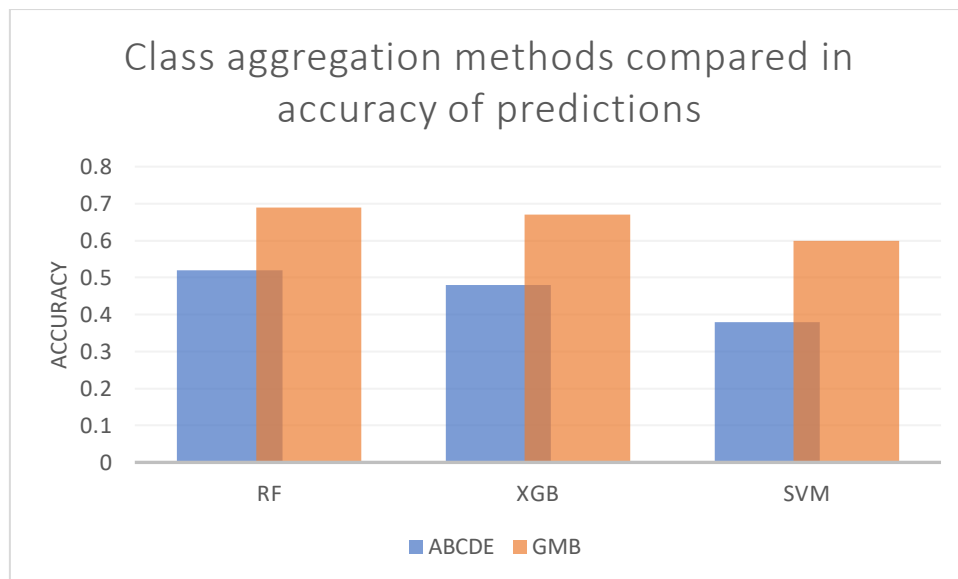


Figure 32: Methods of class aggregation compared in terms of their accuracy values for each model

It's quite obvious from Figure 32 that **the aggregated track qualities have better accuracy of test set predictions across all the classification algorithms**. This can be explained when referring to the tables in Appendix C.1: Performance results of varied track quality aggregation methods. The track qualities that were aggregated are A,B (Good) and D,E (Bad). The increase in accuracy is seen mostly with the aggregation of D and E track qualities. This can be explained that a lot of misclassifications have occurred between these two labels. In other words, many tracks predicted as D were E and vice versa. This gives the first insight about the data, and the models that were built so far: **the data is not varied enough to be able to differentiate between 5 classes**. Another important insight comes from the fact that once the track qualities were aggregated, the bad values had the highest performance measures, and were thus predicted the best. This tells us that the model is predicting extremes better than any other class due to the values of the data. **The model is unable to differentiate Good tracks from Medium tracks, but it can predict Bad tracks, and from the bad tracks it is not differentiating between D and E track qualities**. Again, this comes from the low variability of the features, especially the numerical ones. If the data were more varied, decision boundaries would form much easier. Also, the nature of the numerical features that were used such as daily tonnage, deformations, transition zone count and ages help to identify the worse track qualities. However, a medium track can still have high asset ages, transition zone count, weak soil and high daily tonnage and this would not show in the data due to a recent maintenance. It's thus a good idea to add the maintenance records which are not easily available.

The second analysis will come from comparing using the 200m and the 200m tracks along with the remainder sections. The sections after cleaning went through thresholding with a 200m section and were cut into smaller sections for which track quality was evaluated. It can be argued that comparing 200m sections with other sections is not feasible. However, if the section length is included as a feature, which it is for the model that uses sections of 200m and remainders.

When using only 200m section lengths, section length does not contribute as a feature because no variability is contained in the feature. This helps to better understand the way the machine learning classification functions and perceives the values of each feature. Variability is what defines the model's learning process. The differences between the values of each data point/s features with respect to its respective label is important.

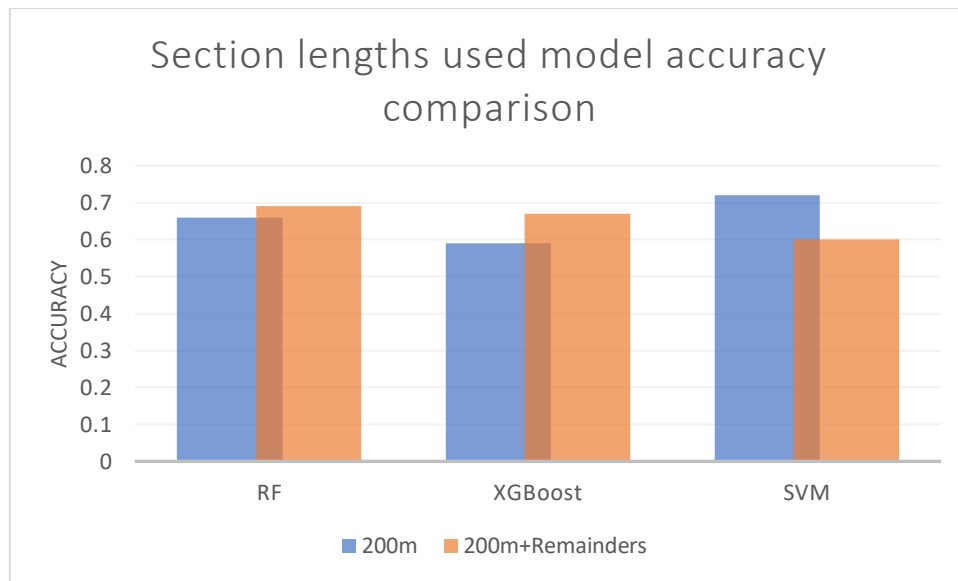
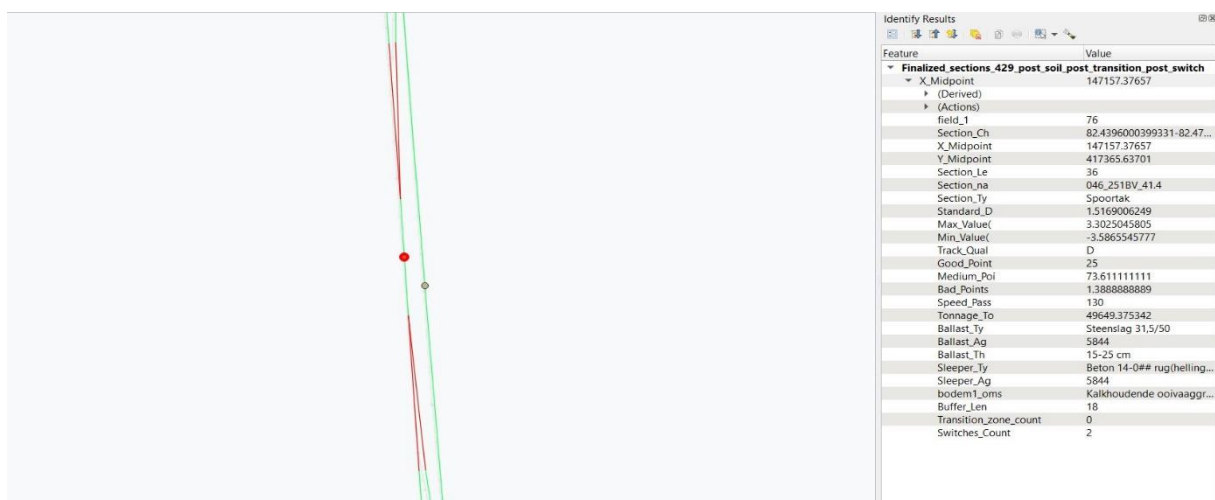


Figure 33: Section lengths used compared in terms of their accuracy values for each model

Once the sections that were shorter than 200m were removed, the training set was reduced from 318 to 284 and the test set was reduced from 106 to 95. The smaller training set is one of the reasons why the 200m section models perform worse everywhere except for the SVM. In the SVM case, the lesser the data, the better as the model's hyperplane (Figure 17) would have to fit lesser data and would thus be more accurate. However, in the other models, the alternative section lengths helped the model perform better, and this is due to the nature of the remainder sections (Figure 33). **Most of the remainder sections carry the after-switch data, since they are last part of a section.** For instance, a 5420m section that has a switch after it, will be broken down into 27x200m sections and one 20m section that is on the edge of the switch. That section contains important information and is thus valuable to keep in the model as long as section length is included as a structural feature. The other sections that are smaller than 200m, are not remainders of section thresholding. **They are small rail branches in between two switches, and they tend to have the worst track quality because of the switches' crossing part that increases the dynamic amplification factors and thus impact track quality as shown in Figure 34.**



It is time to compare the model accuracies between each other. The random forest algorithm performed the best everywhere, in terms of the weighted performance measures as shown in Figure 35. This can be explained by the fact that the Random Forest model takes in the categorical data better and doesn't require any pre-processing and that fits the dataset used in this project the most. The SVM model performed the poorest here, because its parameters were not tuned and that makes it behave sub-optimally. Section 5.4 will look into the performance of the models again after tuning their parameters.

Figure 34: Small section with track quality D shown, switches are in red, rail branches in green

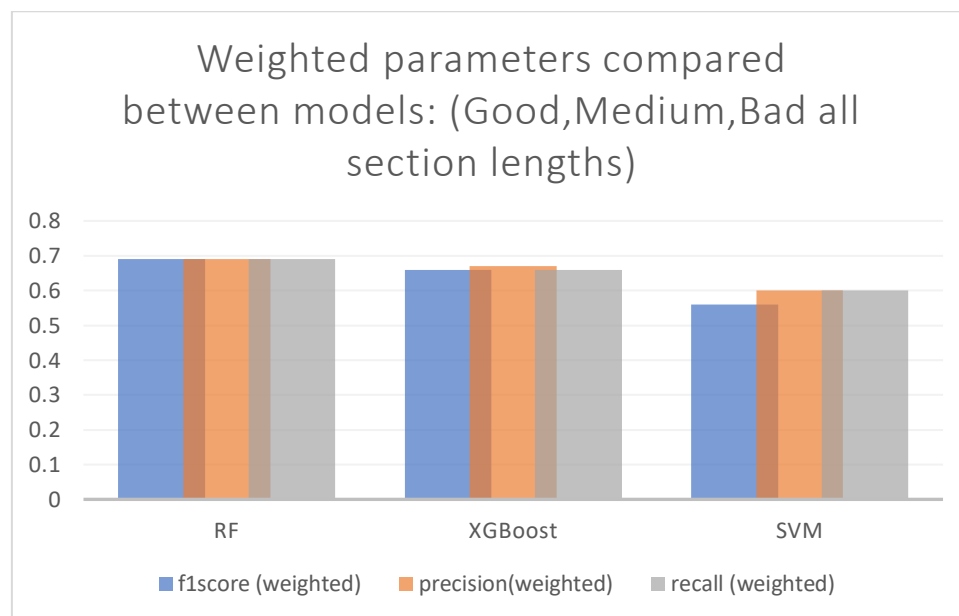


Figure 35: Wieghted performance measures compared between all the used classification algorithms for the test set using the aggregated classes and all section lengths

This sub-section leads to conclude that the best methods of predicting the results is by aggregating the track quality classes into 3: Good, Medium and Bad, and by including all section lengths. From now on, these methods will be used to provide all the other results: feature importance, and hyperparameter tuning validations. As for the third research question that asks which algorithm is the

most accurate for predicting track quality, so far, we have an answer that conforms with our expectations: Random Forest. This will be validated in section 5.4.

5.3 Results: Feature Importance

In this sub-section the importance of the features will be highlighted in an attempt to distinguish between the most important structural features with respect to track quality. However, before that it is vital to define how the random forest algorithm calculates the feature importance.

According to Płoński, (2020), it is of paramount importance to understand the decision tree before understanding the notion of feature importance in the Random Forest algorithm. A Random Forest is a set of decision trees as discussed in section 2.5.3.2. The training data is broken down into subsets and decision trees are formed based on the data of each subset. Then, when the testing set is used, each decision tree is used to make a decision for a datapoint's label, and a majority voting takes place to decide how the prediction will look like.

Each decision tree is made of leaves, and internal nodes. The leaf, is at the end of a branch and represents the label that has to be chosen. The internal node however, is the attribute or the criteria that is set on a feature in the decision tree. Figure 36 shows exactly what is meant by those two terms in a scope related to this project.

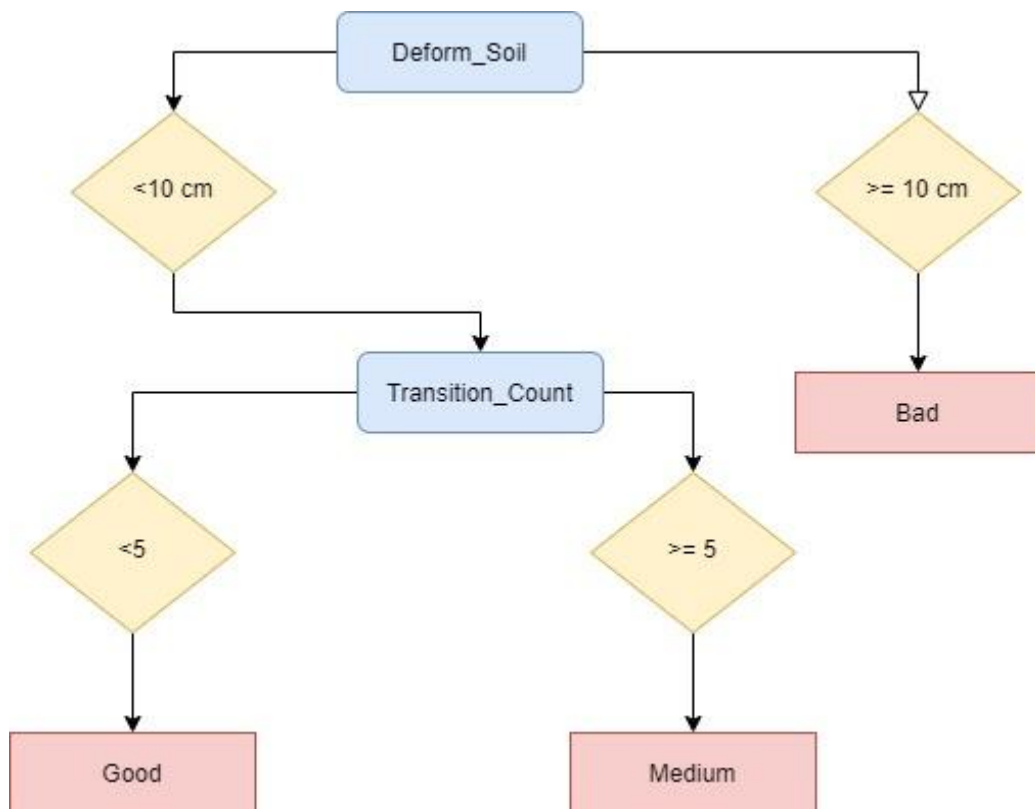


Figure 36: Example decision tree made with features familiar to this project. Internal nodes are the blue nodes, and the conditions that come with them are the yellow nodes. The pink nodes are the leaves

The way the internal nodes are decided is based on two different methods when using the classification functionalities of the random forest algorithm: Information Gain or Gini Index (Saxena, 2017). The method of choice can be specified in the hyperparameters of the random forest algorithm.

The information gain method looks at the entropy of each feature and the entropy of the target. Entropy is defined as:

$$Entropy = \sum_{i=1}^c -f_i * \log_2(f_i)$$

Where f_i represents the frequency of label i for each feature. C represents the total number of classes.

The entropy of the labels is first calculated, by looking at the occurrence of each label in general. If we have 40 Good labels, 60 Medium and 60 Bad, the entropy of the targets would then be:

$$Entropy_{targets} = -\left(\frac{40}{160}\right) * \log_2\left(\frac{40}{160}\right) - \left(\frac{60}{160}\right) * \log_2\left(\frac{60}{160}\right) - \left(\frac{60}{160}\right) * \log_2\left(\frac{60}{160}\right) = 1.561$$

Then the entropy of each feature related to the target will have to be calculated. Relating this to the fictitious example in figure 36, let's suppose that 60 datapoints have values above 10cm for soil deformation and 100 data points have values less than 10cm.

For $deform_soil \geq 10$ cm: 60 occurrences in total

60 values are bad, 0 values are good and 0 labels are medium.

$$Entropy_{deform_{soil} \geq 10} = 0 + 0 - \left(\frac{60}{60}\right) * \log_2\left(\frac{60}{60}\right) = 0$$

Then for $deform_soil < 10$ cm: 100 occurrences in total

0 values are bad, 50 values are good and 50 labels are medium:

$$Entropy_{deform_{soil} < 10} = 0 - (50/100) * \log_2(50/100) - (50/100) * \log_2(50/100) = 1$$

Then finally:

$$Entropy(Target, Dsoil) = p(\geq 10) * Entropy_{deform_{soil} \geq 10} + p(< 10) * Entropy_{deform_{soil} < 10}$$

$$Entropy(Target, Dsoil) = 0 * \frac{60}{160} + 1 * \frac{100}{160} = 0.625$$

The information gain from that specific feature is then calculated as:

$$Information\ Gain(feature) = E(Target) - E(Target, feature)$$

Thus, the information gain of the $deform_soil$ feature in the example is of: $1.561 - 0.625 = 0.936$.

The same is done for all the other features, and the features with the highest information gain are prioritized in the decision tree to be internal nodes as they split the data into sets that are the most homogeneous. In the random forest algorithm, this is done for each decision tree.

Moving on, the gini index works in the same way, however the equation changes and is calculated as such:

$$Gini\ Index = \sum_{i=1}^c f_i * (1 - f_i)$$

Where f_i represents the frequency of label i for each feature. C represents the total number of classes.

The difference here, is that the target doesn't have a gini index, and only the features do. The features with the lowest gini index are prioritized for the choice of internal nodes. The two methods of calculation are called impurity measures. In the random forest algorithm, these values are averaged

along all the decision trees and that is how the importance measure is formed. The gini index and the information gain measures show how each feature decreases the inhomogeneity of the splits in terms of labels, and thus their importance in deciding the labels (Płoński, 2020).

The random forest importance measures are not perfect as they tend to (Płoński, 2020):

1. Give higher importance to categorical data that is unique or highly cardinal which is not necessarily correct.
2. Give less importance to highly correlated features which can be wrong.

Now that the definition of the feature importance has been highlighted along with the drawbacks that the measure contains, it is time to present the results engendered from the two methods of calculation.

Figure 37 and Figure 38 show bar plots of the structural features for the random forest classifier algorithm. These will be analyzed in parallel with engineering knowledge.

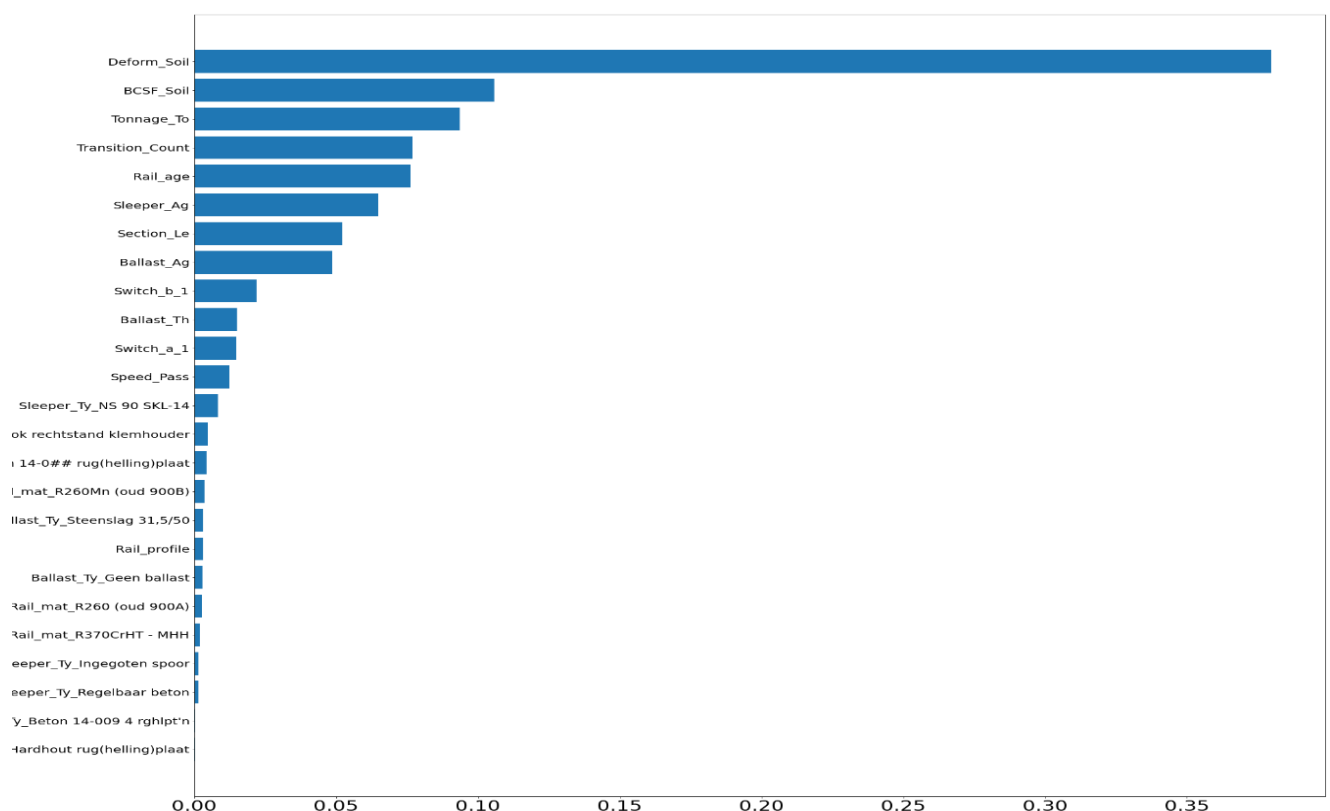


Figure 37: Feature importance calculated using Entropy method for impurity

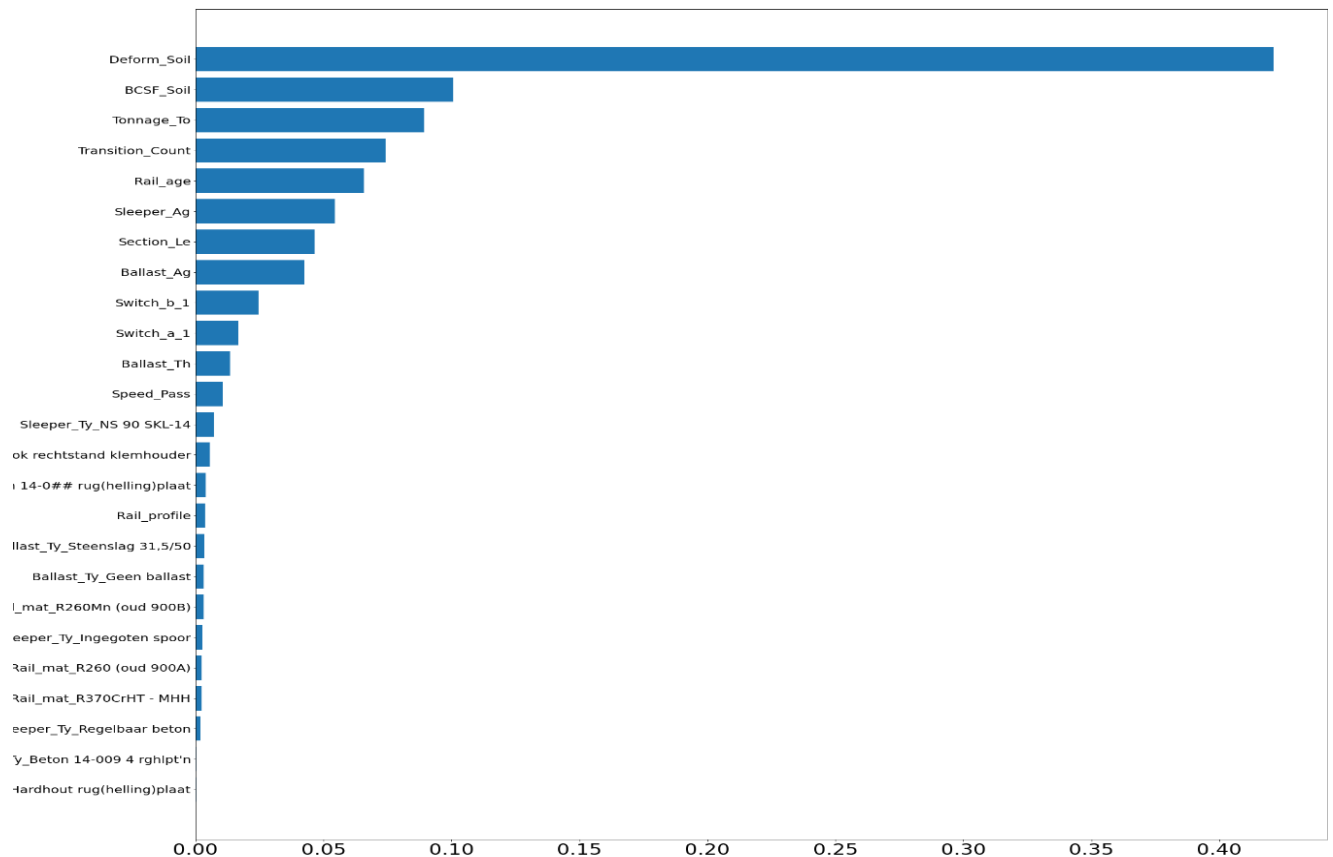


Figure 38: Feature importances calculated using Gini Index method for impurity

Figure 37 and Figure 38 show almost the same results. Using the feature importance values, the model can be reduced to the most important features, and that is what has been done in the hyperparameter tuning section. All the algorithms have been tuned and saved using the most important features only.

The takeaways from the results are that:

- The deformation parameters: soil deformation and bearing capacity safety factor are the most important features for predicting track quality. They have importance values of (0.42, 0.38) and (0.1, 0.11) respectively with the first value corresponding to the Gini Index and the second value to the entropy method. This is a logical result, as the RTG data that is being used is in the D1 wavelength region which relates the most to soil settlements. It was already mentioned that most of the maintenance decisions are made based on the D1 wavelength RTG data, and the majority of these works are tamping works which help reinstate alignment of the track. **The result shown here confirms this: the most important track quality deciders are the soil parameters, namely the settlement. Thus, when using the D1 RTG data, a bad track can mean most of the time that the track alignment needs to be fixed due to the settlement of the track and the ballast bed especially.** Also, to recall, (López-Pita et al., 2008; Zhai et al., 2004) both showed that maintenance activities are doubled on tracks with weaker subsoils. **In fact, the Utrecht-Den Bosch track in both ways mostly consists of clayey subsoil which is much weaker in terms of deformations.**
- The daily tonnage values are the third most important feature to decide track quality (0.1). This is also a very logical finding, as the tonnage values and the train weights are what decide the train's dynamic force, which is at the end of the day the reason behind any type of irregularity from D0 to D3 region. Daily tonnage not only represents the dynamic loading, it

also represents the cyclic loading, both of which contribute to the soil settlement. This can be confirmed by the quantification of the soil parameters that was conducted. **The Winkler model which is a semi-analytical model to calculate deformations takes the axle load P as an input. The higher the load P, the worse the deformations. Thus, daily tonnage is indeed a very important parameter when it comes to track quality in the D1 region.**

- The transition count, which represented the amount of transition zones in the vicinity of the track, from culverts, to bridges to tunnels to level crossings, turned out to be the fourth most important feature for predicting track quality (0.09). All these special structures, due to their differential stiffness vis-à-vis the railway track and the fact that the railway track settles while the engineering structure does so in a lesser manner create differential settlements in the approach zone of the transition zone (Markine, 2020). These differential settlements belong to the D1 region: 3 to 25m. This leads to all types of problems related to passenger comfort and breakages of track components due to the engendered geometrical irregularity. In fact, it had been stated above that transitions zones and switches are found to require 4-8 times more maintenance than a regular free track (Coelho, 2013) which confirms the result shown above. **Following that idea, we could've expected a larger importance factor for the transition zone count, however since the values are normalized, and the soil deformation has an importance of around 0.4, what can be concluded from the results is that in the D1 region, soil deformation has a bigger importance for track quality than transition zones.** In their research tackling structural health monitoring with satellite radar data, (Haoyu Wang et al., 2018) showed that the dynamic displacements of the railway measured by a DIC device in the locations closest to the bridges had the most vertical displacement. Those displacements were mostly located in the frequency ranges of 0-2Hz. The D1 region corresponds to the frequency range of 1.56-12.96Hz using a vehicle speed of 140km/h. The D2 region corresponds to the frequency range of 0.56-1.56Hz. **Thus, the transition zones according to this research shows up more in the RTG data of the D2 region. However, the D1 region's focus of this project was chosen as it's the preferred method of maintenance in Europe, due to the generic measurement train chord length being of 25m. It can be mentioned here that an extension to the project can be made and the D2 region to be included.**

To extend this analysis further, some facts on the data can provide further insight. Out of 429 sections, 58 sections that had transition zones were qualified as "Bad" tracks, meaning D/E quality of tracks according to the EN13848-6 standard. However, 81 sections from 429 in total had transition zones. This can explain the place of the transition zone count as being in the top three most important features but not the most important.

- The most important features from here on are the ages of the sleeper, the ballast, the rail and the section length. The section length got its importance due to the fact that most small railways have bad track quality due to their being connected to one or two switches. **The section length feature is not very indicative of track quality in general, however in this project it was.** As for the element ages, this can be quite indicative of the cyclic loading on the track and its subsequent settlement. The assumption here is that the age represents the time for which cyclic loading has occurred. In fact, (Goodall, 2003), makes a comparison between the effect of impact loads on settlement and the effect of cyclic loading on settlements. The comparison entailed creating a test rig wherein impact loads of 240 kN will be made and the settlement of the soil is recorded. This was compared with 500,000 cycles of 10-240kN cyclic forcing, and it was concluded that cyclic loading causes more settlements than the high impact loads. **Thus, a high age for ballast, rail and sleepers indicates that the track at that section has undergone extensive cyclic loading and thus high settlements.** Let it be noted however, that the age also represents in a way the condition of the elements. The older the sleeper for

instance, the lower its condition. This also applies for rails and ballast. This can also explain the reason behind a low track quality. A counter-argument however is when tamping takes place or any corrective or regenerative type of maintenance that renders the usage of the age of these elements moot.

- Finally, the switches before and after occupy the last of the important features along with the ballast thickness and the passenger speeds. Other features have importances of 0.01 or 0.02 which are negligible. **The low importance values for switches are not expected, it could be due to the way they were implemented; however, it has to be noted that the feature importance values do not in any way represent causality, they are based on information gain parameters, and having the switches being in lower positions just means that in this particular dataset the switches data did not help the model with classification as much as other parameters.** The feature importance values are not correlation values for the bad track qualities, they are values of a specific feature and the higher it is, the more homogeneous the split it creates in terms of classes. **Furthermore, out of 429 sections, only 15 sections had switches after and 20 sections had switches before. This explains the position of the switches as a middle place holder in terms of feature importance values.** With such little switches, being in the said position of feature importance shows that the switches do impact track quality quite much, but to be able to better visualize it, a track with more switches should be analyzed.

Passenger speed should be more important also, the track that was studied for this project didn't include high variability in train speeds. Most tracks had a 130km/h or 140km/h speed limits. This explains the undervaluation of the importance of the passenger speed which is at the heart of the DAF. On another hand ballast thickness had an importance of 0.01, this can mean that ballast thickness was slightly decisive in predicting track quality. **Ballast thickness provides more strength and damping of the vibrations from the wheel-rail interface. The thicker the ballast the less the value of the stress at the subsoil surface.** This was shown in the quantification of the subsoil parameters, where stresses had to be evaluated using the pyramid shape of stress transfer.

- We expected the ballast type and the sleeper type to play a part in predicting track quality however they were the least influential. This can be explained by the fact that due to their categorical data type, along with the rail type and the rail material, they were trumped by the numerical data types and their importance wasn't highlighted enough. The ballast type, in terms of ballasted or non-ballasted plays a big part in whether specific tracks encounter settlements or not. The non-ballasted track obviously settles less than the ballasted track. However, the ballasted track poses a certain transition zone when going from a conventional track type. This could've also been added as a structural feature. The sleeper type was also expected to be important, as timber sleepers tend to engender more settlements due to their weaker stiffness. All these inconsistencies contribute to the model's limitations.

To better understand these results and the reflection behind them, the two most important features have been omitted from the model and the predictions were evaluated. Without the soil deformation values and the BCSF values, the model accuracies were decreased in the regards of every single accuracy parameter. The values can be seen in Appendix C.3: Performance results with the two most important features omitted. A summary of the accuracies with these two features and without is displayed in Figure 39.

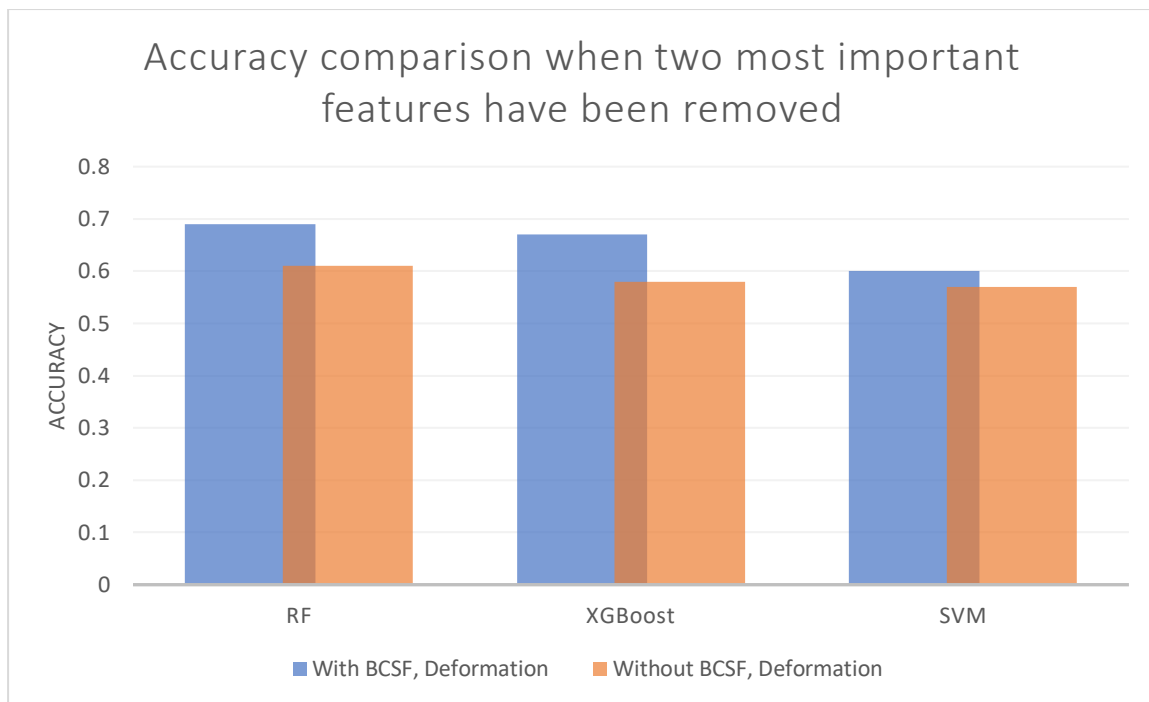


Figure 39: Prediction accuracies compared across all the models between the cases where the two most important features are included and when they are not. Section lengths used are 200m+remainder, and classification method used is the 3-class aggregated method

Figure 39 and Appendix C.3 show that without the two most important features, the model performs worse in predicting the test set across all classes and all models. Thus, feature importance, in the structural sense, can be interpreted as the factors that most define track quality and impact track quality. Soil properties don't necessarily have to worsen track quality more than transition zones, however from the generated results, they are the most decisive in terms of track quality, whether, that quality be good, medium or bad.

The other side of the coin to better understand the feature importances, is to omit all the "un-important" features and check the model performance. The features used are the passenger speed, the soil deformation values, the BCSF values, the ages of the elements, the tonnage values, the transition count and the switch slopes (before and after) along with the ballast thickness values. These amount to 11 features, instead of the 23 features that were used and displayed in figure 30.

The tables in Appendix C.4: Performance results with all the un-important features omitted and Figure 40, show that the performance measures are the same when omitting all the "un-important" features. They vary slightly for the random forest model, however not enough to create the need of including them in the final model. Only the important features will be used for the hyperparameter tuning.

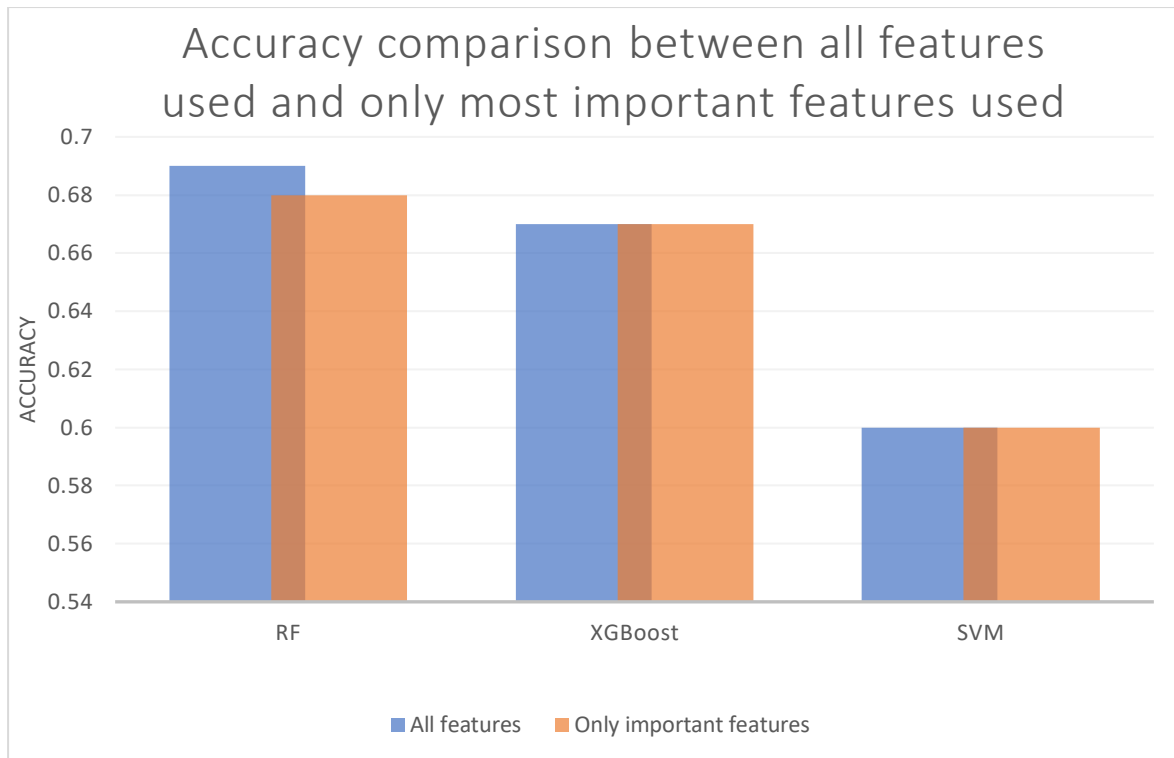


Figure 40: Prediction accuracies compared across all the models between the cases where only the most important features are included and when all features are. Section lengths used are 200m+remainder, and classification method used is the 3-class aggregated method

In a nutshell, feature importance values extracted from the random forest model define the need of using each feature in order to define a track quality for the sections. It was shown that the soil properties were the most influential in determining track quality, relatively, and for those specific sections. Using the data gathering algorithm that has been built and the random forest algorithm, each track can be scored in terms of which structural element is impacting track quality the most, and resources could be allocated in monitoring the weakest elements with respect to those structural parameters. It can be concluded for now that for the Utrecht-Den Bosch two-way line, in January 2019, soil parameters affected track quality the most.

5.4 Hyperparameter Tuning and Model Saving

Each algorithm has its parameters that can be changed, and in turn change the entire way the algorithm functions. This sub-section will look into each model and its hyperparameters while highlighting the pseudo-codes used to tune those parameters in Appendix C.5: Hyperparameter tuning pseudo-code. Each model will use as input the most important features, namely what is shown in Table 5. Those features were deemed as the most important by the random forest classifier. It should be also noted that they are all numerical types.

Table 5: Most important features, the features used for hyperparameter tuning

Feature Name	Data Type	Definition	Examples
Speed_Pass	int64	Passenger Speed in km/h	140 km/h

Tonnage_To	float64	Daily Tonnage Value	4910 tons/day
Ballast_Ag	int64	Age of the Ballast in days	1766 days
Ballast_Th	int64	Thickness of Ballast	5cm ,17.5cm, 35cm,
Sleeper_Ag	int64	Age of the Sleeper in days	1766 days
Rail_age	int64	Age of the Rail in days	1766 days
Switch_b_1	float64	The slope of the switch before the section	1/9=0.1111, 1/15=0.0667
Switch_a_1	float64	The slope of the switch after the section	1/9=0.1111, 1/15=0.0667
Transition_Count	int64	The number of transition zones within the section length	4
Deform_Soil	float64	The deformation value in mm calculated for the specific soil type in the section	9.76555
BCSF_Soil	float64	The bearing capacity safety factor (unitless)	3.456

Hyperparameter tuning refers to the action of choosing random values for randomly chosen hyperparameters and running the model with random combinations and comparing the accuracies. The best random model is then highlighted with its best parameters. Afterwards, for even more detailed tuning, an iterative process can be made to start tweaking the parameters around the best random parameters that were found. For example, let's say that the model at random found that the best number of decision trees to use in the random forest is of 1000. Then, using a grid, we can try to check 990 and 1010. This process can be maintained until convergence of parameters is reached.

Random Forest:

A wide range of hyperparameters will be tuned for this algorithm. The pseudo-code that explains and shows how this was done is included in Appendix C.5: Hyperparameter tuning pseudo-code.

The hyperparameters that were tuned for the Random Forest algorithm are the following:

- `n_estimator` = this corresponds to the number of trees in the random forest
- `max_features` = max number of features considered for splitting a node
- `max_depth` = how many levels in a decision tree
- `min_samples_split` = the minimum number of datapoints in a sample before splitting
- `min_samples_leaf` = the minimum number of datapoints a leaf node should have
- `bootstrap` = method for sampling data points (with or without replacement)
- `criterion` = the index used to split the data entropy/gini index

XGboost:

The XGboost algorithm requires more rigorous pre-processing as per section 4.1. The hyperparameters that were tuned for this algorithm are the following:

- `n_estimators` = this corresponds to the number of trees in the random forest
- `max_depth` = how many levels in a decision tree
- `learning_rate` =
- `objective` = this is to tell the algorithm what it will be doing: multi:softmax relates to multi-class prediction, while multi:softprob returns a probability of prediction for each data point vis-à-vis each class

- `eval_metric` = this is the evaluation metric used for the validation data, to compare the predicted values with the test labels: two options are available for multi-class classification problems: `merror` and `mlogloss`. They represent respectively multiclass classification error rate, and multiclass logarithmic loss values
- `gamma` = nodes are split if and only if the loss function returns a positive value, of a minimum of `gamma`
- `subsample` = the fraction of data to be sampled at random for every tree
- `colsample_bytree` = the fraction of columns to be sampled at random for every tree

SVM:

For the SVM algorithm a grid search will be made instead of a random search because of how little hyperparameters the SVM has relative to the other two algorithms. In a grid search, as opposed to the random search, each combination is computed and not only random combinations that adhere to the number of iterations chosen.

The hyperparameters that were tuned are the following for the SVM algorithm:

- Kernel function: the kernel function defines how the support vectors look like (figure 17). These support vectors could be linear or could be polynomials or any other function.
- C values: this is called the soft margin. The smaller the c values the bigger the margin of the hyperplane. Also, the smaller the c value, the more points on the boundaries that are ignored (Ben-Hur & Weston, 2010).
- Gamma values: for a fixed c value, the higher the gamma value, the more flexible the kernel is. This means if the gamma value were very high, almost each point would be trained to be classified correctly. This leads to a problem of overfitting. Overfitting is the phenomenon of having all the points in the training set well fit but to a very specific manner such that validation would not work (Ben-Hur & Weston, 2010).

Once each hyperparameter has been tuned, K-fold cross-validation, a method of validation that takes the entire dataset and breaks it into a different test set and a different training set K times will take place. This method of validation (Figure 41) is to make sure that the model is not biased and can be validated on any testing set. The cross-validation score is a computation of the accuracy for each experiment. The mean can be used to show the general accuracy of the model.

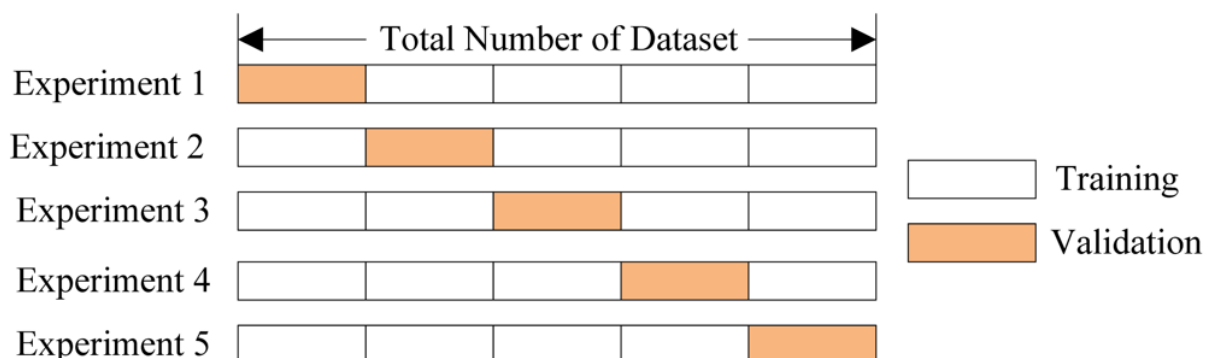


Figure 41: K-Fold Cross Validation shown, <https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74>

Once the models have been validated, they are saved using “pickle” in Python, to be called later again to infer the model (Appendix C.5: Hyperparameter tuning pseudo-code). The performance measures of each tuned algorithm are shown in Appendix C.6: Hyperparameter tuning performance measures

and CV accuracies, along with the cross-validation accuracies and the optimized hyperparameters that were used to save the models. **Figure 42 shows that the SVM performed the best in terms of cross-validation, meaning it's suffering the least from overfitting.** Looking at the values of accuracies across all folds in Appendix C.6: Hyperparameter tuning performance measures and CV accuracies, it can be deduced that the XGboost and the Random Forest algorithms do indeed suffer from overfitting and bias, as some of the cross-validation folds performed much worse than others with accuracies of 0.2. This will show in the inference of the model on new data. It is understandable, seeing the nature of the data used for this project to have some overfitting issues, as the threshold sectioning creates sections with similar structural features. Thus, when learning the features and labels belonging to the same mother section and then predicting a test set of similar sections, the models will perform much better compared to other sections. This already highlights a problem with the nature of the data and the similarity that lies within it due to the section thresholding algorithm.

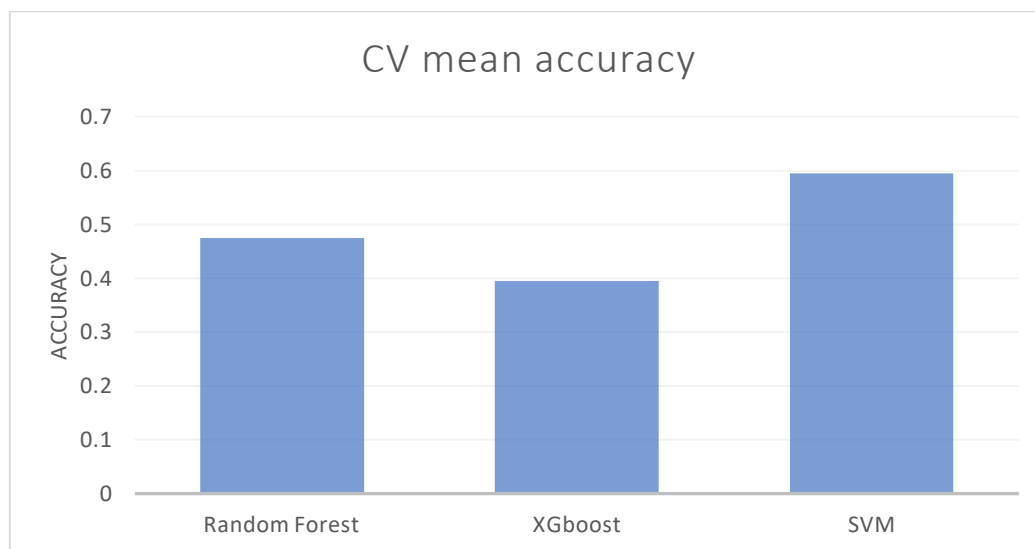


Figure 42: Cross-Validation Mean accuracies across the 6-folds for each algorithm

The best performances capture of each model however is shown in Figure 43. **The random forest algorithm performs the best in terms of predicting the test set, and undergoing validation. The SVM algorithm does perform the poorest there, however comparing with Figure 42, this tradeoff comes at the price of circumventing overfitting. The Xgboost algorithm lies in the middle looking from both lenses.** Although, it is important to note that the Xgboost algorithm showed its value in the hyperparameter tuning process as it was the quickest to compute all the combinations. That is where the Xgboost algorithm shines, in its low memory usage and its quick computation. At the end of the day Machine Learning is a tradeoff between computation effort and results and the results here show exactly that: **the xgboost although lying in the middle of the accuracy comparisons was still the least computationally demanding.**

We can expect that in the inference of the model, that the SVM performs the best out of all the other algorithms, however the prediction capacity of this model is limited due to the low variation and the overfitting that the models have. A way to circumvent this is to add more distinct data and sections from different tracks. The fact that the Utrecht-Den Bosch track, on which these models were built on are a two-way track of 44km each way. Both ways in a two way-track are usually very similar in terms of structural elements and this can be seen in Appendix B.4.2: Section Cleaning Output Table (Utrecht-Den Bosch).

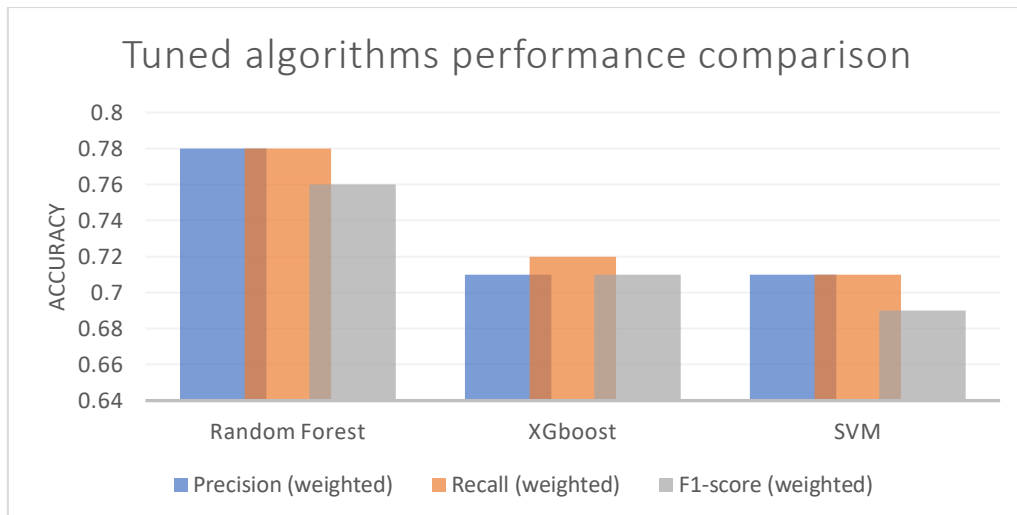


Figure 43: Performance measures of tuned algorithms in train-test split manner

The next step, is to look at a different piece of track with distinct data and check whether the above tuned algorithms can predict their track quality classes well enough. If not, then the models suffer from severe over-fitting and the remedies will be discussed. If they can, then the models formed can work as predictors of track quality class.

5.5 Inference

It is important to note that to use each model, the same pre-processing that model has undergone needs to be made for the inference data. For instance, to use the XGboost and the SVM algorithms, the features have to be scaled with the standardization method. Moreover, the features that were used to conduct the predictions were the 11 most important features decided on in section 5.3. This is valid for each of the inference tests.

5.5.1 The second way test

The first step is to train the models and tune them on one way of the track just as was done in the previous sections. The Den Bosch–Utrecht track (one-way) represents the RILA chainages ranging from 0 to 44.07 km. This can be found in the data looking at 0 to 216 rows. The rest represents the opposite way going from Utrecht to Den Bosch. The models are trained and tuned using the Den Bosch–Utrecht track to predict the track qualities of the second way. It has to be noted that the rest of the data is not seen by the model and thus is treated as new data. The results can be seen in Appendix D.1: Second Way Inference Performance Results. The trained models were validated in the same way as shown as in the previous sections, using cross-validation and using the same performance measures.

The training set consists of 216 sections and the second way used for inference consists of 213 sections.

Figure 44 summarizes the accuracy results of prediction of the second way.

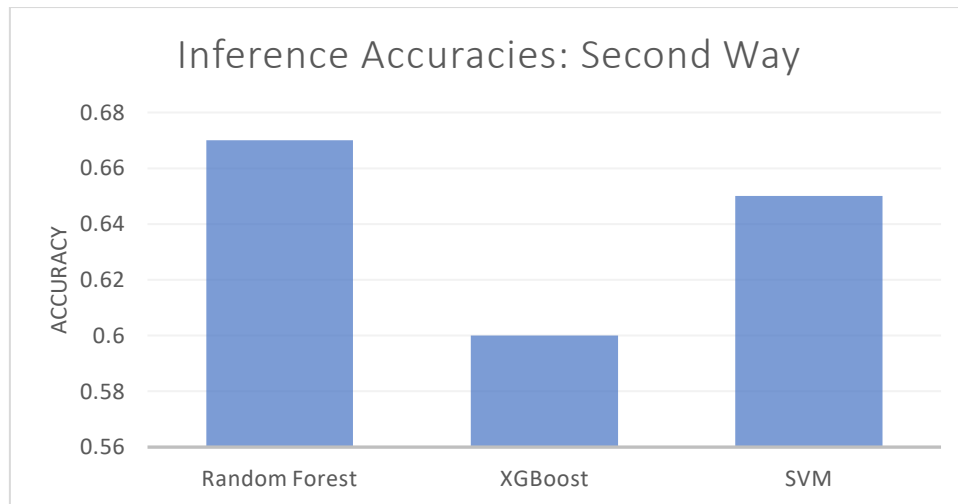


Figure 44: Inference accuracies of the second way (Utrecht-Den Bosch)

The accuracies are quite high for each of the algorithms, which is expected. However, it has to be noted that the prediction of the Good/Medium points was low for all three algorithms (Appendix D.1: Second Way Inference Performance Results). The second way predictions are similar to the test set of the entire track (sections 5.3 and 5.4). This makes sense, as, in terms of performance, the new data cannot be predicted better than the test set. In fact, recalling from previous sections, the test set also showed the same weaknesses in predicting the Good/Medium sections while predicting the Bad sections relatively well.

To understand this a bit more the feature importances of the second way are displayed in Figure 45.

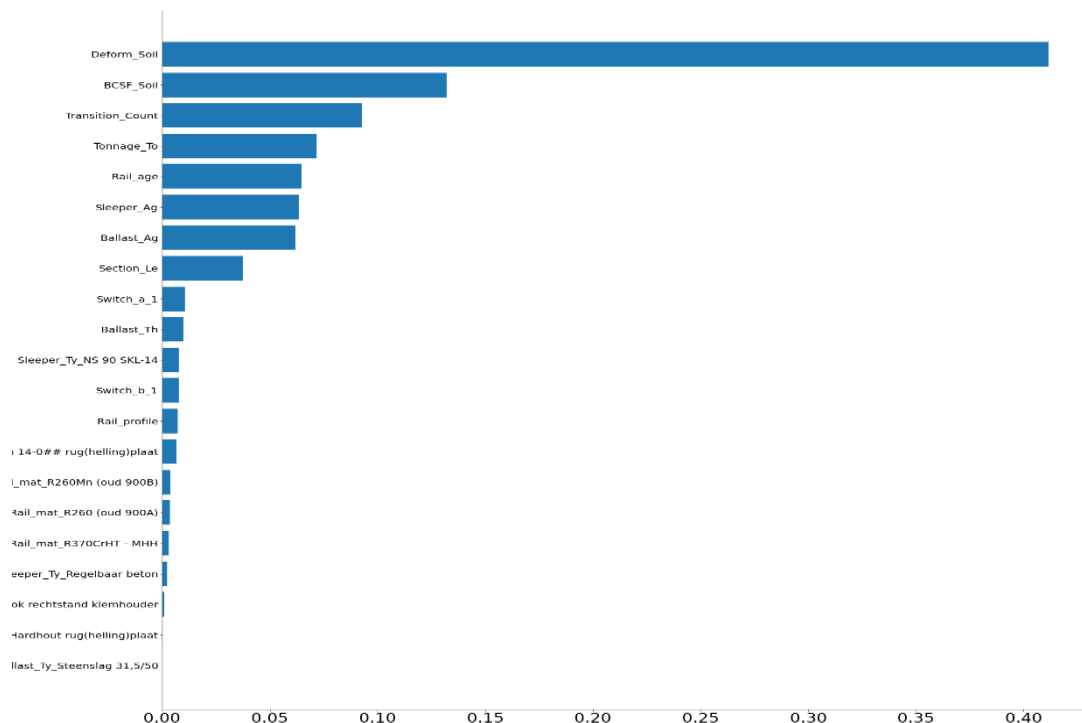


Figure 45: Feature importances of the second way

What can be understood is that the second way has the same feature importance values of the entire two-way track which explains why the prediction performances of the second way were so

close to the test set of the entire track. The entire track of 88km thus behaves as two similar 44km tracks. This will definitely create some bias and lower accuracies when predicting other sections which is what the two remaining tests will show.

This result provides with powerful insight:

- The second way can be predicted up to an accuracy that is similar to the testing set accuracy of the entire model. This means that the first way and the second way are very similar in track qualities, because they are similar in terms of structural elements. If the model's entire performance were increased with more data, the second way can then be inferred following this case study. Thus, if an infrastructure manager wants to monitor a two-way track in terms of track qualities, they can only measure one track and predict the qualities of the second way with an accuracy close to 60%, in terms of Good/Medium/Bad. The locations with bad qualities can then be focused on, and resources could be allocated accordingly. **This saves money in the long run, cutting the need of measuring both ways, which, in the case of normal measurement trains would save on the track's capacity, the cost of personnel, and the cost of using the measurement train. However, in the train-borne system, it would just save on the cost of using the RiLA device.**
- This result can let us expect that the model is slightly biased, as the 88km track that was used for training, turned out to be two similar 44km tracks which decreases in the variability of the data. The generalization capacity of the built models is thus expected to not be very high. However, those models were validated using the test sets, and their feature importance values can be used with confidence.
- The Random Forest algorithm is again the best at predicting track quality for the sections in study.
- **When two tracks have similar feature importance values (values and hierarchy of structural features), then the prediction accuracies of the second track using the model trained on the first will give accuracies close to the test set of the model.**

5.5.2 The last piece of track test

The second test to check the generalization capacity of the model, is to cut out a small piece of track in both ways from the Utrecht-Den Bosch track. This will be done in a way such that the section thresholding doesn't play any part in helping increase the accuracies of the results. To better explain this, the section thresholding algorithm which broke down sections into a number of 200m sections, created repeatability in the structural features (all the features but the transition count and the soil parameters). Thus, with this test, the model will be checked for its ability to generalize the deduced patterns and variations from the data it was provided with to a track that has similar structural features but hasn't been trained on. To better understand this, Figure 46 portrays the trained track that will be used vs the track on which the inference will take place on.



Figure 46: Training track sections are shown in purple while the track on which the inference will be conducted is in green. Both sections are in both ways.

The performance results of this second test are shown in Appendix D.2: Last Piece of Track Inference Performance Results. The summary of the accuracies is shown in Figure 47.

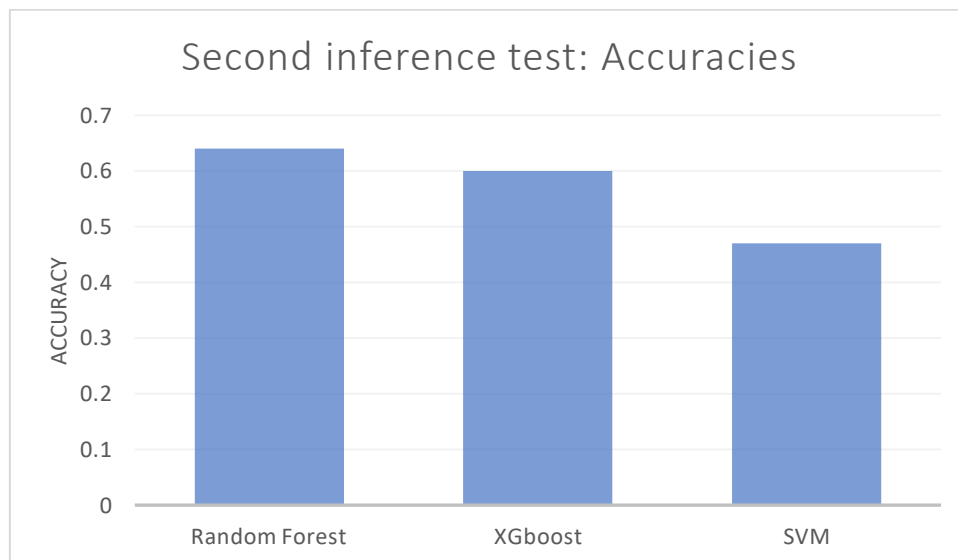


Figure 47: Inference accuracies of the last piece of track of Utrecht-Den Bosch two-way track

Referring to Figure 47 and the tables shown in Appendix D.2: Last Piece of Track Inference Performance Results, it can be seen again that only the bad sections have been predicted well. This allowed the accuracies to be around the 60% mark for the classifier algorithms. What this second test has allowed to identify is that the data and the models are able to predict with 60% accuracy the track qualities of sections in both ways whose structural elements are similar to the trained model's. It has to be noted that the bad sections are being predicted the best in every case. The accuracies are less than the accuracies of prediction of the second way and looking at Figure 48, it can be understood that the feature importance values are now different than those of the entire model.

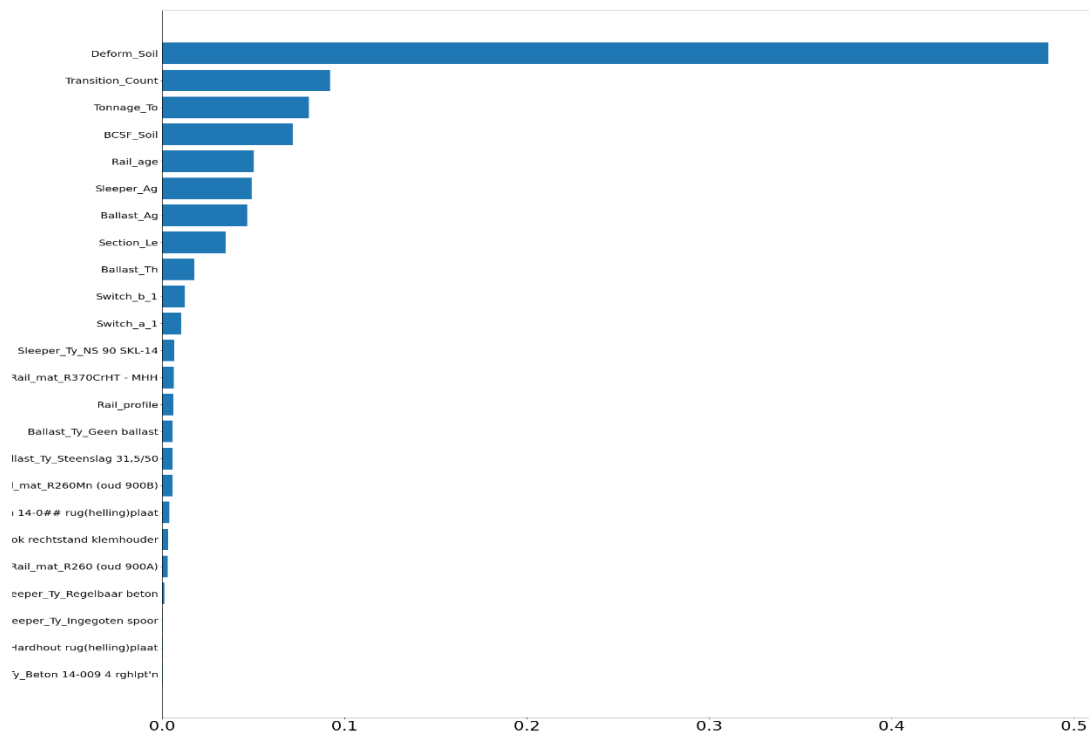


Figure 48: Feature importances of the last piece of track in the Utrecht-Den Bosch line

- Thus, as a first conclusion to this second test, it can be said based on the results and the data that backs it, that **predicting track quality with similar features to that of the trained model can be done relatively well using all three algorithms, however the best with the random forest algorithm, especially for the bad sections.**
- A second conclusion is that when the predicted track has different feature importance values, then the predictions are less accurate, which makes complete sense. This can be shown by comparing Figure 45 with Figure 47. **The second way which had the same feature importance values as the first, was predicted with better accuracy, while the last piece of track which had different feature importance values than the trained model, was predicted with less accuracy.**

One question remains, what if one of the structural elements has never been seen? For instance, the Utrecht-Den Bosch track is of mostly sandy subsoil. Thus, can the trained models based on the Utrecht-Den Bosch railway line predict a mostly clayey railway track's section quality? This is where the third test comes in.

5.5.3 Different subsoil type test

It is now time to use a new set of data and a new track to test the generalization capabilities of the trained models. The track chosen is a 7km one way track that goes from Den-Bosch in the direction of Eindhoven as seen in Figure 49, for ease of writing this track will be referred to as "new track" from now on. This track, using the alignment and height data has been qualified with a track quality for each of its sections. The models will predict its track qualities using its structural features, and they will be then compared with the real evaluated track qualities. This is to understand whether the model created performs only well on predicting track qualities of already seen structural features, since the track shown in Figure 49 is mostly made of sandy subsoil while the Utrecht-Den Bosch track mostly consists of clayey subsoil.

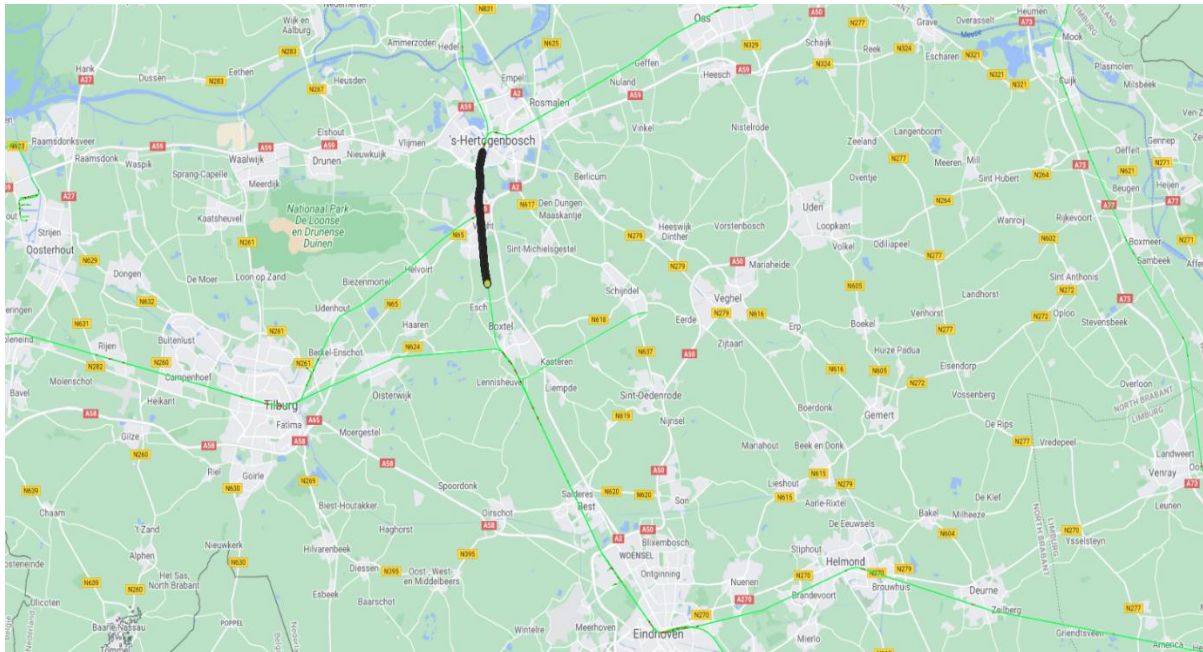


Figure 49: Track chosen for last model generalization test shown on QGIS

This track underwent the same process of data gathering as the Utrecht-Den Bosch did with no errors and no difficulty. The inference check is made not only on the prediction performance but also on the data gathering model itself which was successful. The features that were gathered are shown in Appendix D.3: Section Thresholding Output Table (Den Bosch Eindhoven Track). It has to be noted that the data gathering was semi-automatic, meaning that the data gathering algorithms in Python that are automatic are complemented by the usage of QGIS which required manual work.

Since the method of aggregating the track qualities and using the 200m+remainder sections was found to be the most accurate in the training results, it was also used on this data to get the predictions.

Before going into the results, Figure 50 and Figure 51 show the soil parameter values of both the Utrecht-Den Bosch track on which the models were trained and the new track. What can be seen immediately is that the new track (blue) has lower deformation values and higher BCSF values than the Utrecht-Den Bosch track (orange). This is logical since the new track consists of sandy subsoil while the majority of the Utrecht-Den Bosch track consists of clayey subsoil. Clay deforms more and has lower bearing strength, and this is shown in those figures. **What can be expected by the models, is an over-estimation of the track quality, by predicting more good sections than bad.** This would follow the conclusions made from the first two tests in that the models that were built are able to predict sections whose structural features have been already seen.

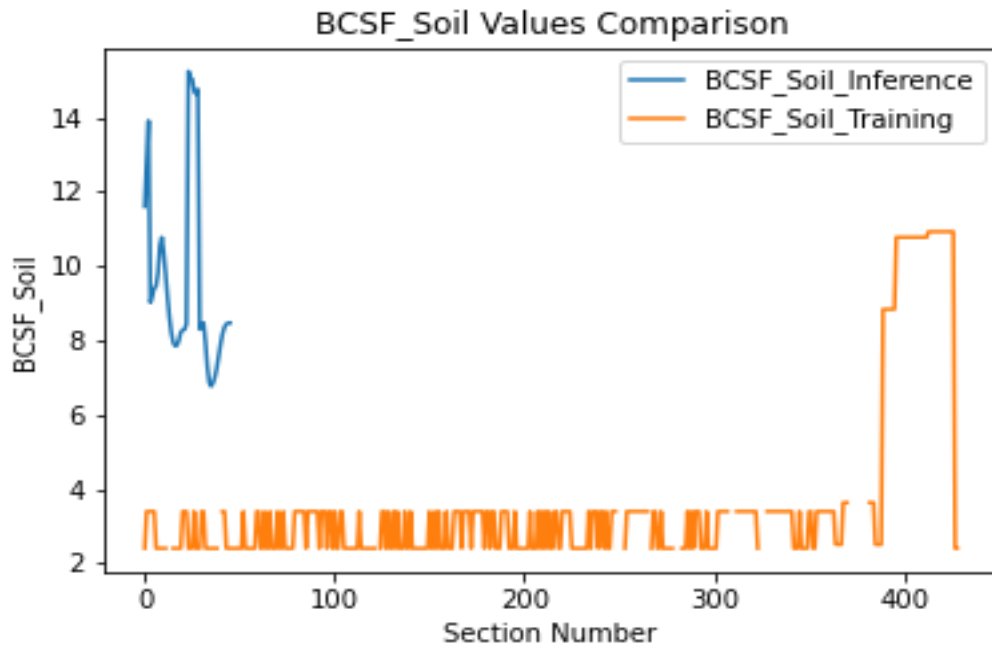


Figure 50: The BCSF values for all the sections are compared between the Utrecht-Den Bosch track and the inference track

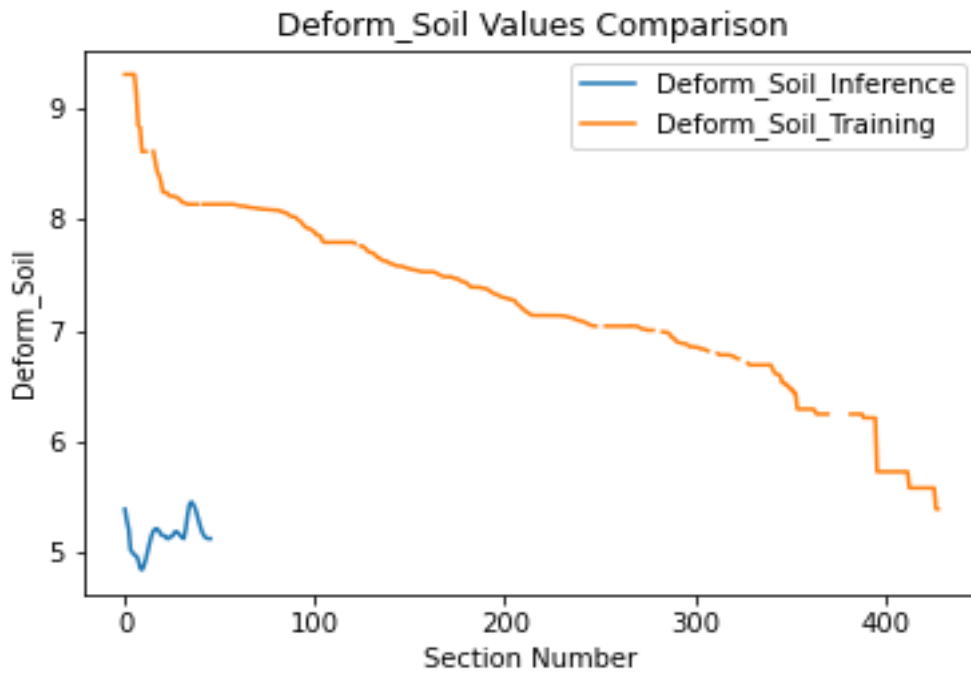


Figure 51: The soil deformation values for all the sections are compared between the Utrecht-Den Bosch track and the inference track

To form more expectations and check whether the conclusions of the first two tests ring true, the feature importance values of this track are shown in Figure 52.

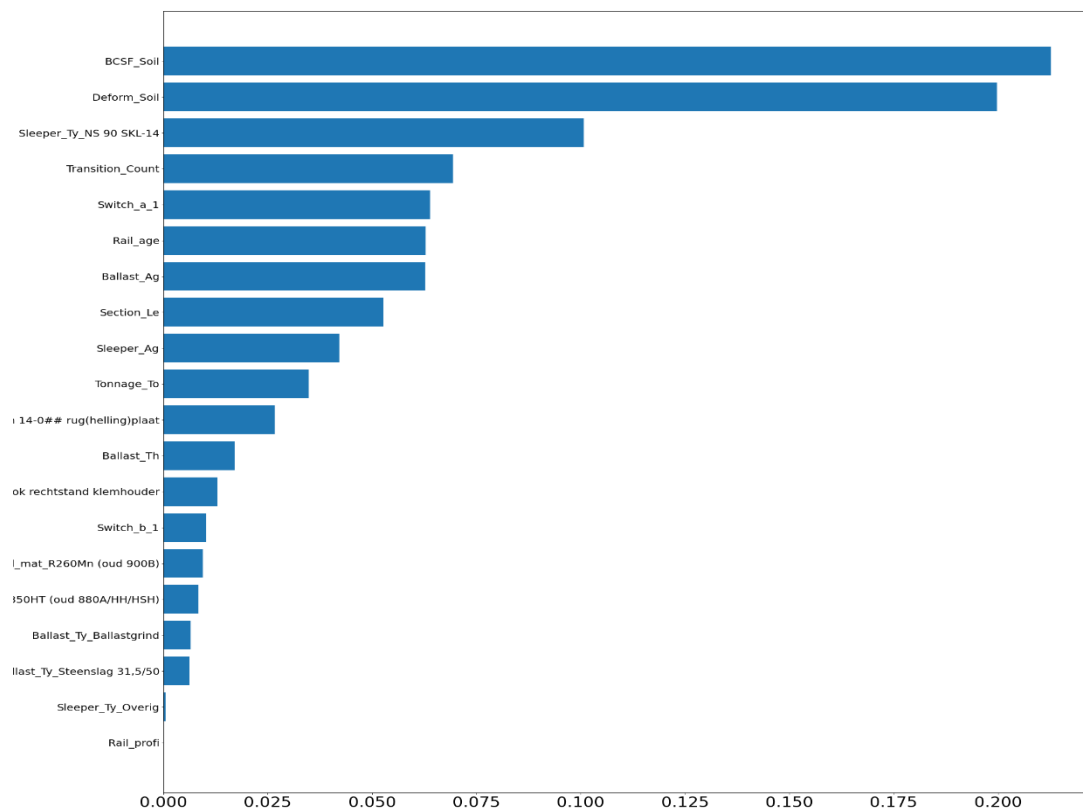


Figure 52: Feature importance using Random Forest for the inference track

What can be concluded by looking at Figure 52, is that the importances of the features are not the same on all the tracks. For instance, the usage of the standard NS90 sleeper was deemed to be the third most important feature for predicting track quality on this railway line. Looking at the inference tracks' sections and the structural features displayed there, it can be seen that none of the bad tracks (D/E track quality class) had the NS90 sleeper, which is why for this specific case, the sleeper type was an important feature, **which brings us back to the fact that feature importance values is not always in direct causality of track quality, since the NS90 sleeper is the newest sleeper type and its usage could mean that the tracks are less impacted because of that and not because of the superiority of said sleeper.** The soil parameters were however the first and second most important features just like the trained models, but otherwise, there are some differences. **Thus, following previous conclusions, it can be expected that the models will not perform very well in predicting this track due to the difference in feature importance values and due to structural features, that were not seen in the trained model.**

The performance measures of the predictions are included in Appendix D.4: Den Bosch Eindhoven Track Prediction Performance Results. As a summary, Figure 53 shows that the SVM is the most accurate out of all three algorithms with all algorithms being less accurate than 40%. This is misguided as the SVM algorithm only predicted Bad values as can be seen in Appendix D.4: Den Bosch Eindhoven Track Prediction Performance Results which isn't what was expected, this can be attributed to features other than the soil parameters. Therefore, the SVM model cannot be used as a reference for the generalization of the data. The second best, is the Random Forest algorithm with a 30% accuracy of prediction. The XGBoost algorithm comes in third place with an accuracy of 27%. **The low accuracies of prediction were expected due to the fact that the feature importance values did not match the trained models'.**

It has to be noted that both those algorithms predicted the Bad values the best with respective recall values of 75% and 62%, which wasn't expected. Also, the fact that the SVM model predicted only bad values wasn't expected seeing that the soil parameters should push the model to over-estimate the quality and not the opposite. **However, it can be surely said that the models cannot be generalized to predict sections whose feature importance values do not match the trained models', and sections with different structural features than the trained models'.**

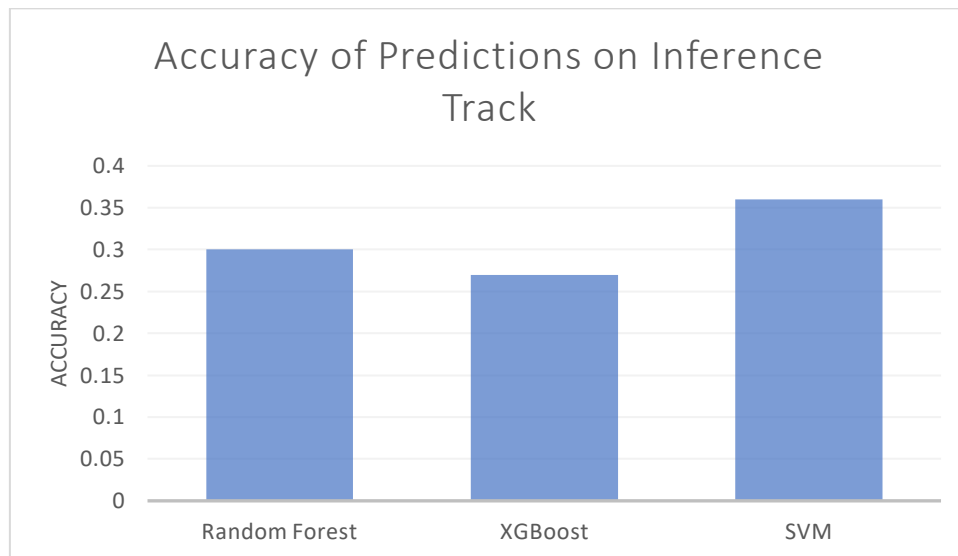


Figure 53: Accuracies of prediction using each of the three tuned algorithms

Looking at Appendix D.4: Den Bosch Eindhoven Track Prediction Performance Results and the confusion matrices (Figure 54, Figure 55, Figure 56), the definitions of precision and recall become important. For all three algorithms the bad values were predicted the best. However, in the SVM case, all the datapoints were predicted as Bad (Figure 56). The random forest algorithm predicted 75% of the bad tracks correctly (Appendix D.2), with 12 out of 16 (Figure 54). However, out of all the datapoints that were predicted as bad (38), only 12 were correct, giving a precision value of 32%. As for the XGboost, algorithm, it predicted 63% of the bad datapoints correctly (Appendix D.2) with 10 out of 16 (Figure 55). However, out of all the 29 predicted data points as bad only 10 were correct which gave a precision value of 32%.

		PREDICTED LABELS		
		BAD	GOOD	MEDIUM
TRUE LABELS	GOOD	12	4	0
		11	1	0
	MEDIUM	15	1	0

Figure 54: Confusion Matrix: Random Forest- Inference

		PREDICTED LABELS		
		BAD	GOOD	MEDIUM
TRUE LABELS	BAD	10	5	1
	GOOD	7	0	5
	MEDIUM	12	2	2

Figure 55: Confusion Matrix: XGboost- Inference

		PREDICTED LABELS		
		BAD	GOOD	MEDIUM
TRUE LABELS	BAD	16	0	0
	GOOD	12	0	0
	MEDIUM	16	0	0

Figure 56: Confusion Matrix: SVM- Inference

Having gone through all the tests and their prediction results, some conclusions and discussions can be made about them:

- The conclusions that arise here are firstly that feature importance is track-specific and not something that can be generalized, and that secondly, the track-specific feature importance can be used as a complement for maintenance decision for tracks. This means that since each track has its specific features that are the most influential for track quality determined by RTG data, resources can be better allocated to solve this specific problem or to sections that exhibit low performances in terms of these structural features. **Track qualities can be predicted if the trained model and the track to be predicted have similar feature importance values.**
- A second conclusion is that by sampling only one way and computing its track qualities, the second way's sections' track qualities can be predicted with an acceptable accuracy.
- A third conclusion is that the models that were built can also predict nearby sections with similar feature importances and structural elements, therefore if the trained model were built using more varied data, then following the conclusions shown above, the trained model would be able to predict more and more sections in the Netherlands. To achieve this, much more data is needed, and this comes at a cost obviously, memory and time of processing and data gathering.
- A fourth conclusion is that the models that were built suffer from bias and overfitting and this can be attributed to the fact that the training track that was used had a lot of repeatability due to it being a two-way track, with each way being similar in terms of structural features.
- A fourth conclusion is related to the data gathering algorithms and how the inference worked on them. The data-gathering was semi-automatic, however linked the structural elements to

each section very well by linking the RTG data Fugro has to the ProRail database using their nomenclature system.

- **The random forest proved to be the most accurate predictor across all model tests and across all model preparation modules.** This can be once again attributed to the ease of usage of the random forest algorithm and its wide variation of hyperparameters than can be tuned. Moreover, the random forest requires no pre-processing and accepts all types of data types, which fit this project perfectly.

To conclude this module, it can be said that to form a more generalized track quality predictor out of structural elements, more tracks and more sections need to be included in the model training as to provide more variability and to avoid bias. This is demanding in terms of time and resources, because the Utrecht-Den Bosch line an 88km two-way line consist of 351,852 datapoints that ended up being sectioned into 41 sections with different structural elements, which does not offer enough variability to the model to be generalized. Even those 41 sections contain similarities as a two-way line provides similar features on each way, in terms of structural features. The 41 sections were then broken down into 429 sections using a 200m threshold for sectioning. Each of the 429 sections had different soil parameter values (deformation and BCSF), however the ranges were limited to sandy ranges and this led to an incapability of generalization of the model to clayey tracks; despite the relatively high validation accuracies for each tuned model. In addition to the variability problems, each track has its own values of feature importance which define the influence of each structural element in defining track quality. **Feature importance values do not show the worst performing structural elements, but the structural elements that have the biggest aid in predicting track quality: good, medium or bad.** This leads to the same point again: the trained model needs to include all the combinations of a categorical feature, so that when the model is being tested on new data it has already seen all sleeper types for instance. However, the results also include a silver lining in the fact that feature importance values are track-specific and can be used as an extra advice for the infrastructure manager to better allocate their resources. These are verified by doing the simple exercise of removing the most important features and adding them again to check whether prediction accuracy changes. Another silver lining is the fact that building a track quality predictor based on one way can predict the other way with an acceptable accuracy especially for the bad sections, which are of more importance. This can save money for the contractors and free up some tracks from being occupied by measurement trains. Finally, tracks with similar structural features to the trained models can have their track qualities also predicted with an acceptable accuracy, with the bad sections being predicted the best. Thus, from an infrastructure manager's standpoint, this can be very useful.

Looking at other stakeholders, Fugro will highly benefit from the data gathering module that was built and from the data enrichment that it engendered. The conclusions provided by the track quality predictor are also of importance due to the fact that this project can be extended to larger parts of the Netherlands as to create a more accurate predictor.

6 Limitations, Conclusions and Recommendations

This section will entail going through the research limitations of this project, the conclusions that this project engendered along with the answers to the sub-research questions and finally end with recommendations for future research.

6.1 Research Limitations

The research methodology that was adopted provided a track quality predictor that feeds into structural elements, along with insights about the structural features that impact track quality the most. Yet, some limitations to the research method are laid down below:

- The Utrecht-Den Bosch is a two-way track of 88km (44km each way). Training a predictor on such a track proved to be counter-productive as the models developed bias to the structural features of this track that showed similarity in both ways. This led to high processing times for the RTG data as well. Using an 88 km one way track would have had the same processing time and data gathering time while providing more variability in the data. This would have led to a more generalizable model. However, every cup half empty is also half full. Using the two-way track allowed to understand more the similarities of each way to the other. It was found that training the model on one way could lead to accurate predictions for the second way.
- The data gathering phase entailed gathering structural features for each section that was generated. The way the data was gathered relied heavily on ProRail's database. Some mistakes were found in said database with placing dates of elements being in 1950, and with some sections not being available in the database due to renovations such as the Geldermalsen station. It is always important to not rely on one source of data as it leaves the machine learning model vulnerable, however ProRail are the sole infrastructure manager in the Netherlands and they are the proprietors of the structural elements' data. This could've been circumvented by using imagery detection techniques to tap into the point cloud data and the videos that RiLa generates as to gather structural features for each section, however that is a project of its own. It is nice to mention though, that it would be a nice extension of this project.
- The data gathering process was semi-automatic and not fully automated which led to more time required to gather the structural features. This was due to the manual work that was required in the QGIS software. However, some of the software's capabilities could've been replicated in Python, albeit with difficulty.
- Many assumptions were taken and the reasoning behind them was explained, such as the manipulation of the negative ballast/sleeper/rail ages, and the parametrization of the soil types. Those assumptions were quick and logical fixes, however assumptions always come with limitations.
- The shorter wave irregularities such as insulated joints, welds, expansion joints, defects (squats, head checks, corrugation etc...) are quite important in forming longer wave irregularities that would've been detected by the built model. The difficulty in adding them definitely played a part in the lower accuracy of prediction of the said model. As mentioned in the previous sections, short wave irregularities cause high-impact loads that in turn affect the ballast and the sub-ballast layers that in turn end up creating longer wave irregularities in time, which represent the D1 wavelength region tackled by this project.
- Using a fixed section length of 200m could have decreased the importance of the transition zones with respect to track quality. Dynamically speaking, each induced vibration has an influence length for which the vibration later disappears, and 200m could be larger or smaller than that, which would downplay the dynamic effect of transition zones.

6.2 Conclusions

The project at hand tackled the creation of a track quality predictor that fed into general structural elements. The goal behind said project was to bridge the gap in the literature and provide more understanding on the interdependencies between track quality and its structural components. To reach this goal, a general research question was framed:

How to relate track quality in a section to its surrounding structural elements and form a predictor out of it?

This research question was broken down into four sub-research questions which led to breaking down the project into four main parts. 1) A literature review was conducted to understand a railway track and how its components behave. Each of the railway track's components was then analyzed and its main functions highlighted. This led to a better understanding of how a track also interacts with its surrounding structural elements and how it degrades. In the same literature review, an overview of widely used track quality indicators was made along with an explanation of the notion of relative track geometry and how it relates to real-time quality indicators. This section led to a choice of track quality indicator and a bolstering of the structural elements that impact track quality the most. 2) Then, it was time to gather the chosen structural elements for the Utrecht-Den Bosch track, which consists of 351,852 datapoints. This could have not been done manually, thus an attempt to automate the workflow was made, and the track was turned into sections with track qualities and structural elements that describe said sections. 3) The literature had also defined machine learning and had gone through similar classification work and research, and three algorithms had been chosen to test the data on. The third part of the project entailed training machine learning classification algorithms and checking which algorithm yielded the most accurate predictions. 4) The models generated also gave an insight on which structural elements are the most prominent and powerful in predicting track quality. This output was analyzed along with the generalization capacity of the created models.

As a result, a semi-automatic data gathering algorithm that feeds into RiLA generated RTG data and creates sections with track qualities and structural features was created, along with a method to form a track quality predictor for sections that do not have any RTG data that can be turned into track qualities.

To conclude this project, the 4 sub-research questions will be answered below:

1. What structural elements are to be studied?

All the structural elements and components that through literature were deemed to impact track quality were added to the model. Going through the literature, an understanding of the state of the forcing in the railway track was built. In short, the most impactful forcing is dynamic forcing, which can be segregated into two different types of loads: P1 load and P2 load. Those loads represent an increase in the static load of up to 3.5. This increase in static loading is called dynamic amplification factor (DAF). This value relates to the **train speed**, the wavelength of the irregularity that caused the dynamic load. The dynamic load is also proportional to the **axle load** as well which had to be included to better understand the degradation of the track. Research also showed **that transition zones and switches** were areas where much more maintenance was required than other sections (Coelho, 2013). In the same regard, sections with softer soil required more maintenance than other sections in stronger soils (López-Pita et al., 2008; Zhai et al., 2004), thus **subsoil parameters** were important to add to the model. Literature also showed that when the elements age, they degrade and their performance can degrade as well which would lead to less track quality in general on the affected sections, thus, **ballast, sleeper, and rail ages** were all added to the model as well. **Sleeper types** such

as timber sleepers were found to cause more settlement than other sleeper types, and **ballasted track types** were also found to settle more than non-ballasted tracks, and this is important because the data that is being studied lies in the D1 wavelength region where settlements are important to track.

Moreover, general characteristics of the track were also added to check whether they contributed to the track quality such as, **ballast thickness, rail profile and rail material**.

Smaller wavelength elements that cause a sharp change in geometry such as insulated joints, expansion joints, welds and rail defects are also important to include in the model as they end up causing settlements with time, however they are difficult to detect and were not included in the model.

Understanding the type of data that is being used, which is RTG data in the D1 region was of paramount importance of choosing the structural elements to be studied and the way the track quality indicators would be incorporated into the project for later classification. Through an argumentation, it was found that using the alignment data along with the longitudinal level data would give more insight on the general state of the track. The classification was done by averaging the moving standard deviation values for the left and right rails, and then taking the average of the moving standard deviation values for each section to choose the classes of that section.

2. How to automate the generation of the features?

Once the decisions were made based on the literature, it was time to develop a workflow for the data that would take the RiLA generated excel files, and turn the datapoints that included X, Y coordinates along with height and alignment values into sections with track qualities and structural features that describe them. This was done in a semi-automatic manner that involved writing algorithms that would link the Fugro data with the ProRail database of physical elements along with some QGIS manipulations. The key to achieving this link was through understanding the ProRail nomenclature system of the rail branches. Once the understanding was built, QGIS was used to link every single datapoint to its rail branch. The rail branch name was thus added as an attribute to each datapoint. The datapoints were then sectioned using their name, and sections were created. This process later included cleaning of the sections and adding the structural attributes to them. Once everything was ready, QGIS was used again to add to each section its soil type and its transition zone count. The soil type was parametrized using the work of Schoorl (2021).

The end result was thus sections that had track qualities associated to them along with their structural features. What was left, was to train the models using those elements.

3. Which classifier algorithm yields the most accurate predictions?

Three algorithms were used for training the data: the random forest, the xgboost and the support vector machine (SVM). The methods of using the data in terms of whether to use all sections lengths such as 200m and different values were studied, and the need of aggregating the track quality classes from A,B,C,D,E to Good, Medium, Bad was also studied. This was done using performance measures in evaluating the prediction accuracy of the test set. The results yielded that the models all suffered from bias and a lack of generalization capacity when it comes to looking at sections with different structural feature importance values, and sections with structural components that the trained model hadn't seen. Despite that fact, the random forest model was still the most accurate amongst the three classifier algorithms due to its versatility and its acceptance of all forms of data types, which is exactly what this project needed.

Let it be noted that the algorithms could predict with an acceptable accuracy of approximately 65% and an almost perfect prediction of the bad quality sections, the track qualities of sections that are neighboring the trained track that had similar structural features and feature importance values. This was also the case when training the models on one way of a two-way track and predicting the second way. The very good prediction of the bad quality sections is of interest, as they are the sections that need to be monitored. This result could lead to having to do less data sampling which would save contractors time and money all the while freeing up some tracks from measurement trains.

4. Which structural features are the most important in predicting track quality?

It was discovered that feature importance values are track-specific by performing the last generalization test. However, for the Utrecht-Den Bosch track, the most important structural features that were found in terms of predicting track quality were the soil parameters: the soil deformation and the soil BCSF. This was attributed to the fact that this track contained majorly clayey subsoil which is considered soft. Moreover, the third most important structural feature on this specific track was found to be the transition count. It was also pointed out that this was due to the fact that transition zones' influence shows up more in the D1D2 wavelength region: 3-70m, as it induces a longer wavelength settlement.

The structural feature importance values were found to be logical and conformant with expectations, and could be used in parallel with the developed algorithm of data gathering to understand in a specific track which features are the most impactful, and allocate resources accordingly. However, keeping in mind that those values do not represent direct causality in terms of track quality degradation. The feature importance values rely on information gain parameters and tell the good tracks from the bad from the medium. A high feature importance value does not mean necessarily that it causes the worst track quality, which could be seen with the switches having low importance values.

It is also important to mention these results show the importance of features and variables measured within the data sets. Forming conclusions about which feature is indeed important could be only done with a more complete dataset that include not only various soil parametrizations but also other important features that could not be included such as any short wavelength irregularities (welds, insulated joints, expansion joints etc...)

Finally, through the four sub-research questions, an answer to the major research question can be ascertained:

How to relate track quality in a section to its surrounding structural elements and form a predictor out of it?

Data gathering is the most important answer to this question. Once a data-gathering algorithm has been built, it is important to know which data to gather. This thesis has shown that gathering data for tracks with high structural variability would lead to a more generalizable model. This comes at a cost obviously, which is time and resources. The data used for this project consisted of 351,852 data points, and was reduced to 41 sections. The nature of the railway track is that sections of 5km or more sometimes have the same structural features, which would mean that much more data is needed to be able to achieve structural feature variability.

On another note, it should be mentioned that once the data has been gathered and the sections have been prepared, the random forest algorithm for classification is a very good algorithm to run the structural elements data on. This algorithm was made for such projects and such applications. Tuning

the algorithm until high accuracy levels have been reached is of paramount importance. Then, all that remains is to test the model on unseen data in a similar manner to this research to check the generalization and inference capabilities of the model.

An interesting output that comes with the random forest classifier is the structural features variable importance. The model, once accurate enough, can give this output which could be used as insights for infrastructure managers to better allocate their resources and know what the biggest problem is in a specific railway track in terms of track quality.

6.3 Recommendations for further research

The research could be extended/improved in many ways, the recommendations for future work are as follows:

- Creating an application programming interface (API) that would feed into a constructed backend structure of database such as SQL and would have a smooth dataflow with no interruptions needed and any manual work. This is recommended as to be able to create many sections on which the model could be better developed and trained with more variability all the while being able to generate data as soon as any sampling run is made by RiLA. Thus, the model would take the date of the run and the excel sheet of the pre-processed data and have it undergo the workflow and enrich the model in real-time.
- This research focused on rail branches and used switches as a structural feature to those branches. It is recommended that the same research be extended to switches and try to understand the degradation and the reasons behind the track qualities with respect to structural features.
- The project focused on RTG data in the D1 wavelength region as it is the most widely used by contractors in the railway business due to the most commonly available measurement trains and the chord lengths that they have. However, it is interesting to extend this research as to include the alignment and height data available in the D1D2 region.
- Transfer learning is a method that could take in the information and the patterns learned from a different structural feature class that hadn't been learned in the trained model and adds this new-found information to the model. This method could be investigated in terms of its efficacy in rendering the model more generalizable and inferable.
- Looking at acceleration data or data in the D0 region would have helped determine spots with sharp changes in geometry or short wavelength irregularities. Categorizing the data into their respective classes (welds, insulated joints etc...) would've not been possible however, including a feature that has the count for each section on those sharp geometry changes is also a recommendation to add to the prediction accuracy of the model.
- Calculating influence length for transition zones and deciding the section length based on that is a recommended way to extend the research as to increase the accuracy of the transition zones' influence on the data.

Appendix A: Utrecht-Den Bosch Project Sample Data

This appendix shows a sample dataset of the Utrecht-Den Bosch project for the January 2019 dataset.

Data Numbering	Rila_Chainage	X	Y	Height_D1_L	Height_D1_R	Alignment_D1_L	Alignment_D1_R	MovSTD_AL	MovSTD_LL
0	0.00000	148760.12544	412139.89386	-0.01523	-0.01830	-0.00507	-0.00012	0.012254	0.020986
1	0.00025	148760.20087	412140.06559	-0.01648	-0.01941	-0.00500	0.00264	0.012255	0.020973
2	0.00050	148760.30138	412140.29455	-0.01737	-0.01997	-0.00496	0.00531	0.012252	0.020959
3	0.00075	148760.40196	412140.52342	-0.01781	-0.01988	-0.00481	0.00736	0.012245	0.020945
4	0.00100	148760.50261	412140.75228	-0.01775	-0.01911	-0.00445	0.00840	0.012232	0.020930
5	0.00125	148760.60329	412140.98113	-0.01718	-0.01767	-0.00383	0.00833	0.012217	0.020917
6	0.00150	148760.70391	412141.21001	-0.01613	-0.01568	-0.00307	0.00733	0.012203	0.020905
7	0.00175	148760.80441	412141.43887	-0.01464	-0.01331	-0.00230	0.00576	0.012190	0.020894
8	0.00200	148760.90510	412141.66769	-0.01281	-0.01075	-0.00149	0.00386	0.012177	0.020886
9	0.00225	148761.00561	412141.89656	-0.01076	-0.00822	-0.00053	0.00183	0.012164	0.020878
10	0.00250	148761.10600	412142.12544	-0.00867	-0.00590	0.00063	-0.00016	0.012152	0.020871
11	0.00275	148761.20656	412142.35426	-0.00668	-0.00390	0.00190	-0.00196	0.012142	0.020862
12	0.00300	148761.30699	412142.58316	-0.00491	-0.00225	0.00311	-0.00353	0.012139	0.020853
13	0.00325	148761.40759	412142.81199	-0.00343	-0.00090	0.00410	-0.00499	0.012142	0.020845
14	0.00350	148761.50818	412143.04079	-0.00227	0.00019	0.00479	-0.00641	0.012149	0.020840
15	0.00375	148761.60877	412143.26964	-0.00141	0.00111	0.00511	-0.00777	0.012153	0.020839
16	0.00400	148761.70950	412143.49842	-0.00079	0.00192	0.00501	-0.00896	0.012151	0.020842
17	0.00425	148761.81008	412143.72726	-0.00034	0.00265	0.00453	-0.00984	0.012143	0.020847
18	0.00450	148761.91098	412143.95595	0.00000	0.00330	0.00380	-0.01033	0.012131	0.020850
19	0.00475	148762.01146	412144.18485	0.00027	0.00381	0.00302	-0.01027	0.012118	0.020850
20	0.00500	148762.11190	412144.41376	0.00048	0.00411	0.00235	-0.00955	0.012104	0.020844
21	0.00525	148762.21246	412144.64259	0.00062	0.00412	0.00196	-0.00814	0.012091	0.020834
22	0.00550	148762.31293	412144.87148	0.00066	0.00381	0.00202	-0.00610	0.012081	0.020826
23	0.00575	148762.41341	412145.10036	0.00060	0.00322	0.00252	-0.00364	0.012254	0.020825
...	...	...	...	...	...	...	...	...	...
351851	88.0869457	148753.997	412140.8408	-0.003845	-0.003941	0.0001366	0.0001416	0.003027	

Appendix B: Data Gathering Pseudo-Codes and Output

In this appendix, the output of each data gathering step will be shown along with the pseudo-code of each Python script.

Appendix B.1: Table showing the linked data using NNJoin

Data Numbering	Rila_Chainage	X	Y	join_NAAM	join_layer	join_KM_Start	join_KM_End	MovSTD_AL	MovSTD_LL
0	0.00000	148760.12544	412139.89386	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012254	0.020986
1	0.00025	148760.20087	412140.06559	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012255	0.020973
2	0.00050	148760.30138	412140.29455	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012252	0.020959
3	0.00075	148760.40196	412140.52342	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012245	0.020945
4	0.00100	148760.50261	412140.75228	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012232	0.020930
5	0.00125	148760.60329	412140.98113	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012217	0.020917
6	0.00150	148760.70391	412141.21001	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012203	0.020905
7	0.00175	148760.80441	412141.43887	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012190	0.020894
8	0.00200	148760.90510	412141.66769	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012177	0.020886
9	0.00225	148761.00561	412141.89656	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012164	0.020878
10	0.00250	148761.10600	412142.12544	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012152	0.020871
11	0.00275	148761.20656	412142.35426	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012142	0.020862
12	0.00300	148761.30699	412142.58316	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012139	0.020853
13	0.00325	148761.40759	412142.81199	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012142	0.020845
14	0.00350	148761.50818	412143.04079	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012149	0.020840
15	0.00375	148761.60877	412143.26964	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012153	0.020839
16	0.00400	148761.70950	412143.49842	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012151	0.020842
17	0.00425	148761.81008	412143.72726	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012143	0.020847
18	0.00450	148761.91098	412143.95595	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012131	0.020850
19	0.00475	148762.01146	412144.18485	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012118	0.020850
20	0.00500	148762.11190	412144.41376	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012104	0.020844
21	0.00525	148762.21246	412144.64259	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012091	0.020834
22	0.00550	148762.31293	412144.87148	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012081	0.020826
23	0.00575	148762.41341	412145.10036	046_255BV_41.6	Spoortak	46.70000	47.22928	0.012254	0.020825
...	...	...	...	...	...	...	...	...	...
351851	88.08695	148753.99697	412140.84083	046_255AR_41.4	Spoortak	46.70025	47.22910	0.003027	0.007339

Appendix B.2.1: Spoortak_Sectioning Pseudo-Code

This pseudo-code belongs to the Spoortak_Sectioning Python code that turns the linked Fugro and ProRail data into sectioned data with respect to the rail branch names and KM start and KM end values.

- **Read_file= read the excel sheet containing Fugro data linked with ProRail data**
 - Example data is in Appendix B.1
- **Initialize arrays: startpoints, endpoints, section_length, features**
 - startpoints is the array that will contain the indices of the start points of the sections in relation to the linked data
 - endpoints is the array that will contain the indices of the end points of the sections in relation to the linked data
 - section_length is the array that will contain the lengths of the sections
 - features is the array that will contain the section name, the section type: Wissel/Spoortak, Kmend and Kmstart values
- **size = the length of the input table or read_file**
- **Set startpoints first value equal to 0**
- **i starts at 0, window starts at 0, sampling_frequency=0.25**
 - the sampling frequency is the distance value at which the data is spaced by, Fugro gathers data with a 0.25m spacing
 - window will be the counter of the values included in each section
 - i is nothing but index for the while loop to go through every single row of the input table
- **While i < size-1**
- **if the name, type and Kmstart of row i are the same of row i+1**
- **add 1 to window, end if**
- **else**
- **add the value of i to endpoints array**
- **add the value of i+1 to the startpoints array**
- **add the features of the row i to features array**
- **section_length= window*0.25, add this value to section_length array**
- **add 1 to i, end else and while**
- **add to endpoints the last index of the table**
- **window= last endpoint – last startpoint**
- **add to section_length the last length using window*0.25**
- **add to features the features of the last row**
- **Turn all the arrays into pandas dataframe**
- **Df=Concatenate the arrays into columns next to each other and name the columns**
- **Filter df to give only the 'Spoortak' type part of the table and export it to get the sectioned spoortak data**

Used Libraries:

- Pandas: <https://pandas.pydata.org/>
- Numpy: <https://numpy.org/>

Appendix B.2.2: Spoortak_Sectioning Output Table (Utrecht-Den Bosch)

Section_Chainages	Section_Length(m)	Section_name	Section_Type	KM_Start	KM_End	startpoints	endpoints
0.0-0.355435840344926 km	355.5	046_255BV_41.6	Spoortak	46.7	47.229276	0	1422
0.355685737307303-5.4034045075298 km	5047.75	046_255BV_41.6	Spoortak	41.651196	46.7	1423	21614
5.40365431431641-5.48065 km	77	046_255BV_41.6	Spoortak	41.573985	41.651196	21615	21923
5.528898765826059-5.710651386786551 km	181.75	046_251AL_41.3	Spoortak	41.343483	41.525866	22116	22843
5.759400271787981-5.90240707383783 km	143	046_227AL_36.0	Spoortak	41.152027	41.295364	23037	23609
5.9026570944022-10.8931360888631 km	4990.5	046_227AL_36.0	Spoortak	36.160725	41.152027	23610	43572
10.893386223658-11.0083857036982 km	115	046_227AL_36.0	Spoortak	36.045494	36.160725	43573	44033
11.03739-11.08488 km	47.25	046_225BV_36.0	Spoortak	35.969532	36.016902	44148	44337
11.1136384597265-11.2291358514393 km	115.5	046_203R_34.7	Spoortak	35.825168	35.940949	44452	44914
11.2293864948856-12.3148786215897 km	1085.5	046_203R_34.7	Spoortak	34.739652	35.825168	44915	49257
12.3151288248992-12.3611254396902 km	46	046_203R_34.7	Spoortak	34.693592	34.739652	49258	49442
12.39014-12.4018866429958 km	11.75	046_201BV_34.7	Spoortak	34.652954	34.665009	49557	49604
12.4311296455112-12.5876277584155 km	156.5	046_201BR_34.7	Spoortak	34.467377	34.624358	49720	50346
12.5878777141552-16.1543118355606 km	3566.5	046_201BR_34.7	Spoortak	30.900021	34.467377	50347	64613
16.1545616171472-17.7256487707359 km	1571	046_201BR_34.7	Spoortak	29.322316	30.9	64614	70898
17.7258989847469-17.9008986430981 km	175	046_201BR_34.7	Spoortak	29.146235	29.322316	70899	71599
17.9494000014181-17.9636447623729 km	14.25	666_441L_29.0	Spoortak	346.117711	346.132038	71792	71849
18.0121499721478-18.0298993497283 km	17.75	666_439V_29.0	Spoortak	29.018335	29.03627	72042	72113
18.078404847795-18.3439034590677 km	265.5	666_411R_27.5	Spoortak	28.704165	28.97018	72306	73368
18.344153147849-19.1826563954786 km	838.5	666_411R_27.5	Spoortak	27.863985	28.704165	73369	76723
19.1829065160566-19.4689028565546 km	286	666_411R_27.5	Spoortak	27.577401	27.863985	76724	77868
19.51741-19.9526538820107 km	435.25	513_143BV_26.3	Spoortak	27.094266	27.529267	78061	79802
19.9529039268318-20.3326153933147 km	379.75	513_143BV_26.3	Spoortak	26.714651	27.094266	79803	81322
...	...	...	...	...	...	...	...
87.73116925815079-88.08694569917009 km	355.75	046_255AR_41.4	Spoortak	46.700251	47.229095	350428	351851

Appendix B.3.1: Physical Object Linking Pseudo-Codes

The items that were linked to the 'spoortak' sectioned data are the sleeper data, the rail data, the ballast data and the tonnage values per spoortak. Reading into the excel files that contain that information, the linking occurs based on the 'spoortak' name, and the KMstart and KMend values. Once those are found, the data of interest is easily added. Each of these elements has its own Python Script, however they all work in the same exact way as the files containing the information all have the same format. For the exact features that were added in each code refer to the main text. The example pseudo-code will be given for the ballast data.

- **sections= read the sectioned spoortak excel sheet**
 - Example data is in Appendix B.2.2
- **Ballast_info= read the physical object excel sheet taken from infomatieportaal**
- **Relevant_ballast_info is the ballast_info table where only the columns we want are included**
 - In this case the columns of importance are the placing date of the ballast, the thickness of the ballast, the type of the ballast: crushed stone, ballastless, etc...
- **Num_sections is the length of the sections that we have in the sections table**
- **Initialize arrays: ballast_type, placing_date, thickness_ballast, arrays, date_since**
 - The date_since array is basically the age
 - The placing date is included in the excel sheet in a string format such as: "2018-01-01 00:00".
- **Save Date_baseline as the date of January 1st 2019**
 - This is the date of the Fugro sampling of the Utrecht-Den Bosch line, it will act as the baseline for calculating the age of the ballast
- **For i ranging from 0 to the number of sections**
- **X = the section name of row i**
- **Y = X == all the names in the ballast document**
 - Y has the indices of the ballast document for which we have the section name the same as the ones in the ballast document. One section can appear many times with different KMstart and KMend values though.
- **If the number of times that the section name appears in the ballast file is 1**
- **Split the name of the placing date into numerical and use the date function to save it**
- **Date_since= date_baseline – saved_date, extract this subtraction as a number of days**
- **Add the ballast type and the thickness to the respective arrays, end if**
- **Elseif the number of times that the section name appears in the ballast file is 0**
- **Add to all the arrays the value of 0, end elseif**
- **Elseif the number of times that the section name appears in the ballast file is larger than 1**
- **Initialize a comparison temporary array**
- **Compare the KMstart and KMend values for the rows that have the same name**
- **Take the information of the row that has the most similarity and extract the placing date and add the thickness and type of ballast to the respective arrays**
- **Do the same thing for date_since as before to get the day value, end elseif, end while**
- **Initialize array: sectioned_chainages**
 - This array will contain the RILA chainages of the first point of the section and the endpoint for reference
- **For i ranging from 0 to number of sections**
- **Add to sectioned_chainages in string format the value of the chainage of the first point in the section and the last one in a format: "Chain1 – Chain2 km"**

- **Turn every array into a pandas dataframe and concatenate them together into one table naming the columns adequately and export this matrix to an excel file**
 - We have a table with sectioned chainages, daily tonnage values per spoortak, ballast info, rail info, sleeper info and ballast info. These are all merged into one file (appendix B3.2.2)

The libraries used for this Python script are:

- Pandas: <https://pandas.pydata.org/>
- Numpy: <https://numpy.org/>
- <https://docs.python.org/3/library/datetime.html#module-datetime>

Appendix B.3.2: Physical Object Linking Table (Utrecht-Den Bosch)

Section_Chainages	Length (m)	Name	Section_Type	KM_Start	KM_End	startpoints	endpoints	Speed (km/h)	Tonnage (ton/day)	Type_Ballast	Age_B (days)	Thickness_Ballast	sleeper_type	Age_S (days)	rail_materiaal	rail_profile	Age_R (days)
0.0-0.355435840344926 km	355.5	046_255BV_41.6	Spoortak	46.7	47.22928	0	1422	130	50811.09589	Steenslag 31,5/50	1999	15-25 cm	Beton 14-0## rug(helling)plaat	1999	R260Mn (oud 900B)	54E1	1999
0.355685737307303-5.4034045075298 km	5047.75	046_255BV_41.6	Spoortak	41.6512	46.7	1423	21614	130	50811.09589	Steenslag 31,5/50	-573	> 25 cm	NS 90 SKL-14	-573	R260Mn (oud 900B)	54E1	-573
5.40365431431641-5.48065 km	77	046_255BV_41.6	Spoortak	41.57399	41.6512	21615	21923	130	50811.09589	Steenslag 31,5/50	5844	> 25 cm	NS 90 SKL-14	5844	R260Mn (oud 900B)	54E1	5844
5.528898765826059-5.710651386786551 km	181.75	046_251AL_41.3	Spoortak	41.34348	41.52587	22116	22843	130	50300.03562	Steenslag 31,5/50	5844	> 25 cm	NS 90 SKL-14	5844	R260Mn (oud 900B)	540	5844
5.759400271787981-5.90240707383783 km	143	046_227AL_36.0	Spoortak	41.15203	41.29536	23037	23609	130	50762.97534	Steenslag 31,5/50	11688	> 25 cm	NS 90 SKL-14	11688	R260Mn (oud 900B)	54E1	11688
5.9026570944022-10.8931360888631 km	4990.5	046_227AL_36.0	Spoortak	36.16073	41.15203	23610	43572	130	50762.97534	Steenslag 31,5/50	11688	> 25 cm	NS 90 SKL-14	11688	R260Mn (oud 900B)	54E1	11688
10.893386223658-11.0083857036982 km	115	046_227AL_36.0	Spoortak	36.04549	36.16073	43573	44033	130	50762.97534	Steenslag 31,5/50	11688	> 25 cm	NS 90 SKL-14	11688	R260Mn (oud 900B)	54E1	11688
11.03739-11.08488 km	47.25	046_225BV_36.0	Spoortak	35.96953	36.0169	44148	44337	130	51224.49863	Steenslag 31,5/50	14610	> 25 cm	NS 90 SKL-14	7305	R260Mn (oud 900B)	54E1	14610
11.1136384597265-11.2291358514393 km	115.5	046_203R_34.7	Spoortak	35.82517	35.94095	44452	44914	130	50180.06301	Steenslag 31,5/50	11688	> 25 cm	NS 90 SKL-14	11688	R260Mn (oud 900B)	54E1	11688
11.2293864948856-12.3148786215897 km	1085.5	046_203R_34.7	Spoortak	34.73965	35.82517	44915	49257	130	50180.06301	Steenslag 31,5/50	11688	> 25 cm	NS 90 SKL-14	11688	R260Mn (oud 900B)	54E1	11688
12.3151288248992-12.3611254396902 km	46	046_203R_34.7	Spoortak	34.69359	34.73965	49258	49442	130	50180.06301	Steenslag 31,5/50	11688	> 25 cm	NS 90 SKL-14	11688	R260Mn (oud 900B)	54E1	11688
12.39014-12.4018866429958 km	11.75	046_201BV_34.7	Spoortak	34.65295	34.66501	49557	49604	130	51082.16712	Steenslag 31,5/50	14610	> 25 cm	Beton 14-0## rug(helling)plaat	6909	R260 (oud 900A)	540	14610
12.4311296455112-12.5876277584155 km	156.5	046_201BR_34.7	Spoortak	34.46738	34.62436	49720	50346	130	50547.36712	Steenslag 31,5/50	15340	> 25 cm	Beton 14-0## rug(helling)plaat	15340	R260 (oud 900A)	54E1	15340
12.5878777141552-16.1543118355606 km	3566.5	046_201BR_34.7	Spoortak	30.90002	34.46738	50347	64613	130	50547.36712	Steenslag 31,5/50	15340	> 25 cm	Beton 14-0## rug(helling)plaat	15340	R260 (oud 900A)	54E1	15340
16.1545616171472-17.7256487707359 km	1571	046_201BR_34.7	Spoortak	29.32232	30.9	64614	70898	130	50547.36712	Steenslag 31,5/50	10227	> 25 cm	Beton 14-0## rug(helling)plaat	-244	R260Mn (oud 900B)	54E5	-244
17.7258989847469-17.9008986430981 km	175	046_201BR_34.7	Spoortak	29.14624	29.32232	70899	71599	130	50547.36712	Steenslag 31,5/50	15340	> 25 cm	Beton 14-0## rug(helling)plaat	15340	R260 (oud 900A)	54E1	15340
17.9494000014181-17.9636447623729 km	14.25	666_441L_29.0	Spoortak	346.1177	346.132	71792	71849	130	49445.54521	Steenslag 31,5/50	5113	> 25 cm	Hardhout rug(helling)plaat	5113	R260Mn (oud 900B)	540	5113
18.0121499721478-18.0298993497283 km	17.75	666_439V_29.0	Spoortak	29.01834	29.03627	72042	72113	130	51449.77534	Steenslag 31,5/50	5113	> 25 cm	Hardhout rug(helling)plaat	5113	R260Mn (oud 900B)	540	5113
18.078404847795-18.3439034590677 km	265.5	666_411R_27.5	Spoortak	28.70417	28.97018	72306	73368	130	49843.84384	Steenslag 31,5/50	330	> 25 cm	NS 90 SKL-14	5844	R260Mn (oud 900B)	54E1	10227
18.344153147849-19.1826563954786 km	838.5	666_411R_27.5	Spoortak	27.86399	28.70417	73369	76723	130	49843.84384	Steenslag 31,5/50	330	> 25 cm	NS 90 SKL-14	5844	R260Mn (oud 900B)	54E1	10227
19.1829065160566-19.4689028565546 km	286	666_411R_27.5	Spoortak	27.5774	27.86399	76724	77868	130	49843.84384	Steenslag 31,5/50	330	> 25 cm	NS 90 SKL-14	5844	R260Mn (oud 900B)	54E1	10227
19.51741-19.9526538820107 km	435.25	513_143BV_26.3	Spoortak	27.09427	27.52927	78061	79802	130	53576.10137	0	0	0	0	0	0	0	0
19.9529039268318-20.3326153933147 km	379.75	513_143BV_26.3	Spoortak	26.71465	27.09427	79803	81322	130	53576.10137	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
87.73116925815079-88.08694569917009 km	355.75	046_255AR_41.4	Spoortak	46.70025	47.2291	350428	351851	130	49138.00274	Steenslag 31,5/50	1999	> 25 cm	Beton 14-0## rug(helling)plaat	1999	R370CrHT-MHH	54E1	1999

Appendix B.4.1: Section Cleaning Pseudo-Code

As discussed in the main text, some sections with the same name have different Kmstart and Kmend values which are references that ProRail uses. The reason behind that is that some of the sections with the same name on different parts of the rail branch have different physical objects and characteristics. Once those characteristics were assigned as shown in Appendix B.3.2, it was clear that some of the branches with the same name and different reference values still had the same physical characteristics. To avoid this, the section cleaning code was created to merge those sections into one such that no repetition exists.

Task 1:

- **Read_file= read the sectioned spoortak excel sheet including all the features**
 - Example data is in Appendix B.3.2
- **X= Filter rows to remove the sections that have 0 values for physical objects**
 - Some sections have missing data due to reasons explained in the main text, those were removed from the analysis
- **Initialize arrays: sections, full_sections**
- **For i ranging from 0 to the length of x-1**
- **Add the value of i to the array sections**
- **Compare the row i of x with the row i+1 of x in terms of the columns that include the features**
- **if that comparison shows that all values are the same**
- **add i+1 to the array sections, end if**
 - Every row index is saved at the beginning. It is compared to the next row, if it is the exact same, add the value of that row to sections array. Then we start again with that next row. It is obvious that there will be repetition of row indices but that will be handled by the unique function in python later
- **else**
- **add to full_sections the unique values of the array sections**
 - As soon as we have a row different than the next one, we stop and save the indices that have the same values to full_sections, however their unique values to avoid repetition. Full_sections has thus in each row the number of the rows that have the same features, the sections that are distinct will have only one number in their row
- **reset the array sections, end else, end for**
- **Add last row index to the full sections array**
 - Since we cannot compare i to i+1 for the last row, it was handled outside of the for loop

Task 2:

The full_sections array includes in every single row the indices of the rows that are the same and should be merged. This second task does exactly that.

- **Initialize arrays: to_add and features**
- **For i ranging from 0 up to the length of full_sections**
 - The length of the full_sections array is the amount of sections that we should be left with at the end of this manipulation
- **First point = the first value of the full_sections row i**
 - This represents the first row index of the combined section
- **Last point = the last value of the full_sections row i**
 - This represents the last row index of the combined section

- **If the last point == the same point**
 - This is the case where only one value exists in the full_sections arrow
- **Add the features that belong to the only row included in full sections to the array features, end if**
- **Else**
 - This is the case where more than one value is in full_sections, thus the sections have to be combined into 1
- **Section_length = sum of the all the section lengths whose row index is included in full_section**
- **First_chainage = first RiLA chainage value of the first section to be combined**
 - Splitting the "Chain1-Chain2" format into two numbers and saving Chain1
- **Second_chainage = last RiLa chainage of the last section to be combined**
 - Splitting the "Chain1-Chain2" format into two numbers and saving Chain2
- **Sectioned_chainage = First_chainage and Secon_chainage in a format "Chain1-Chain2 km"**
- **Startpoint = the startpoint of the first section to be combined**
- **Endpoint = the endpoint of the last section to be combined**
- **Features0= sectioned chainages and section length concatenated into one array**
- **Features1=take all the features that belong to the combined section**
 - These are the physical features that are the same for all the sections in the combined section
- **Features2= startpoint and endpoint**
- **To_add = Concatenate features0, features1, and features2 into one array as columns, end else, end for**
 - The reason all those features have been concatenated in that order is to match the same columns of the non-combined matrix in the if loop
- **Concatenate the features and to_add matrices above each other**
 - The features matix is the matrix of the sections that didn't need combining
 - The to_add matrix is the one that has been combined
 - They are added above each other in a manner that the columns match and are in the same place
- **Export this new matrix to excel as cleaned_matrix**

The libraries used for this Python script are:

- Pandas: <https://pandas.pydata.org/>
- Numpy: <https://numpy.org/>

Appendix B.4.2: Section Cleaning Output Table (Utrecht-Den Bosch)

Section_Chainages	Length (m)	Name	Section_Type	KM_Start	KM_End	startpoints	endpoints	Speed (km/h)	Tonnage (ton/day)	Type_Ballast	Age_B (days)	Thickness_Ballast	sleeper_type	Age_S (days)	rail_materiaal	rail_profile	Age_R (days)
0.0-0.355435840344926 km	355.5	046_255BV_41.6	Spoortak	46.7	47.22928	0	1422	130	50811.1	Steenslag 31,5/50	1999	15-25 cm	Beton 14-0## rug(helling)plaat	1999	R260Mn (oud 900B)	540	1999
0.355685737307303-5.4034045075298 km	5047.75	046_255BV_41.6	Spoortak	41.6512	46.7	1423	21614	130	50811.1	Steenslag 31,5/50	-573	> 25 cm	NS 90 SKL-14	-573	R260Mn (oud 900B)	540	-573
5.40365431431641-5.48065 km	77	046_255BV_41.6	Spoortak	41.57399	41.6512	21615	21923	130	50811.1	Steenslag 31,5/50	5844	> 25 cm	NS 90 SKL-14	5844	R260Mn (oud 900B)	540	5844
5.528898765826059-5.710651386786551 km	181.75	046_251AL_41.3	Spoortak	41.34348	41.52587	22116	22843	130	50300.04	Steenslag 31,5/50	5844	> 25 cm	NS 90 SKL-14	5844	R260Mn (oud 900B)	540	5844
5.759400271787981-11.0083857036982 km	5248.5	046_227AL_36.0	Spoortak	41.15203	41.29536	23037	44033	130	50762.98	Steenslag 31,5/50	11688	> 25 cm	NS 90 SKL-14	11688	R260Mn (oud 900B)	540	11688
11.03739-11.08488 km	47.25	046_225BV_36.0	Spoortak	35.96953	36.0169	44148	44337	130	51224.5	Steenslag 31,5/50	14610	> 25 cm	NS 90 SKL-14	7305	R260Mn (oud 900B)	540	14610
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
87.73116925815079-88.08694569917009 km	355.75	046_255AR_41.4	Spoortak	46.70025	47.2291	350428	351851	130	49138	Steenslag 31,5/50	1999	> 25 cm	Beton 14-0## rug(helling)plaat	1999	R370CrHT - MHH	540	1999

Appendix B.5.1: Switches Adding Pseudo-Code

This Python script is designed to add the switches data to the table of features that has been formed so far. The columns that will be added are split into two categories: switch before that section and switch after that section. The before and after data will contain features such as the name of the switch, the type of the switch and the slope of that switch. To understand the dialectic behind the script refer to the main text.

- **Read_file = read the cleaned sections file**
 - The table shown in appendix B4.2
- **Spoortak = read the spoortak table**
 - The spoortak data taken from `mapservices.prorail.nl`
- **Switches = read the switches table**
 - The switches data taken from `mapservices.prorail.nl`
- **Initialize arrays: before, after**
- **Midpoint threshold = 44.3**
 - The sampling of the data has been done in both directions. The Utrecht-Den Bosch track is of 88 km. From 0 to 44.3km of Rila chainage, the line that is in study is the Den Bosch to Utrecht line going northward. From 44.3 to 88 km the opposite is true. Since the after and before definitions are direction dependent, defining this threshold is of paramount importance.
- **For i ranging from 0 to the length of read_file**
 - This would represent the
- **Direction_ID: take the sectioned chainages and turn them to floats for the section in row i**
- **Identifier_comp = compare the section identifiers to find it in the spoortak data**
 - Basically, the spoortak name, the KMstart and KMend values are used to find each rail branch in the spoortak dataset. This comparison creates an array of indices that allow to find that specific section
- **Spoortak_row = turn the identifier comp array into a row number that refers to the section that has been found**
- **Boundary_1 = the value of the BEGRENZERT column of the spoortak row**
 - This column's values are strings that tell us what is north of the rail branch
- **Boundary_2 = the value of the BEGRENZE_1 column of the spoortak row**
 - This column's values are strings that tell us what is south of the rail branch
- **x = comparison between [boundary_1, boundary_2] and [WISSEL, WISSEL]**
 - This is a binary array, a value of 0 means that there is no switch and a value of 1 means there is a switch. Obviously it matters where the 0 and the 1 is.
- **if direction_ID > midpoint_threshold:**
 - if `direction_ID > midpoint`, we are on the left track and then `begrenzert` is before and `begrenze_1` is after
- **if the first value of x is 0**
- **add 0 to the before array, end if**
- **if the second value of x is 0**
- **add 0 to the after array, end if**
- **if the first value of x is 1**
- **boundary_1_name = the value of the BEGRENZERN column in the spoortak row**
- **use the GEOCODE values and the boundary_1_name to find the switch details in the switches table**

- *add the switches features to the before array, end if*
- *if the second value of x = 1*
- *do everything that was done for the case where the first value of x is 1 but this time add the switches features to the after array and use the column BEGRENZE_2 for the switch name, end if, end if (direction_ID > midpoint_threshhold)*
- *else direction_ID < midpoint_threshhold*
 - *if direction_ID < midpoint, we are on the right track and then begrenzt is after and begrenze_1 is before*
- *repeat everything that was done for the case of direction_ID > midpoint_threshhold but in the conditions where the features were added to before array, they are now added to the after array, end else, end for*
 - *This means that the first value of x now represents the after data and the second value represents the before data*
- *Turn the before and after arrays into pandas dataframe and concatenate them*
- *Add the concatenated dataframe to the cleaned switches data*

The libraries used for this Python script are:

- Pandas: <https://pandas.pydata.org/>
- Numpy: <https://numpy.org/>

Appendix B.5.2: Switches Adding Output Table (Utrecht-Den Bosch)

The table here was cut to be able to fit the page. The attributes in the table are the same as the section cleaning output table that is shown in Appendix B.4.2 but with the switches columns included. The other attributes were hidden to fit the page. Switch_b means the switch before and switch_a means the switch after. This is direction dependent.

Section_Chainages	Section_Length	Section_name	Section_Type	KM_Start	KM_End	startpoints	endpoints	Switch_b_naam	Switch_b_slope	Switch_b_type	Switch_a_naam	Switch_a_slope	Switch_a_type
0.0-0.35543584km	355.5	046_255BV_41.6	Spoortak	46.7	47.22928	0	1422	0	0	0	0	0	0
0.355685-5.403 km	5047.75	046_255BV_41.6	Spoortak	41.6512	46.7	1423	21614	0	0	0	0	0	0
5.40365-5.485 km	77	046_255BV_41.6	Spoortak	41.57399	41.6512	21615	21923	0	0	0	255B	1:15	GW
5.528-5.71065 km	181.75	046_251AL_41.3	Spoortak	41.34348	41.52587	22116	22843	255B	1:15	GW	251A	1:15	GW
5.759981-11.002 km	5248.5	046_227AL_36.0	Spoortak	41.15203	41.29536	23037	44033	251A	1:15	GW	0	0	0
11.037-11.085 km	47.25	046_225BV_36.0	Spoortak	35.96953	36.0169	44148	44337	227A	1:9	GW	225B	1:9	GW
11.1136-12.3611km	1247	046_203R_34.7	Spoortak	35.82517	35.94095	44452	49442	225B	1:9	GW	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
16.1545-17.7256 km	1571	046_201BR_34.7	Spoortak	29.32232	30.9	64614	70898	0	0	0	0	0	0

Appendix B.6.1: Section Thresholding Pseudo-Code

This pseudo-code belongs to the final Python script in the automation of the data gathering. The sections that have already been cleaned now have all the features that are needed. However, they need to be broken down into pieces of 200m tracks, and the track quality class needs to be added.

Task 1:

- **Reaad_file_two = read the cleaned sections file that includes the switches data**
 - The table shown in appendix B5.2
- **Read_file = read the linked data between Fugro and ProRail**
 - The table shown in appendix B1
- **endpoints = the column in read_file called endpoints**
- **startpoints = the column in read_file called startpoints**
- **section_lengrth = the column in read_file pertaining to section lengths**
- **type_section = the column in read_file pertaining to section type**
- **naam = the column in read_file that includes the spoortak names**
 - These columns are being saved in arrays as to not have to tap into the read_file table every time
- **Sampling_Frequency = 0.25**
- **Initialize arrays: midpoint_x, midpoint_y, features, stdav_list, col3, stdav_list_al, col3_al, enpoints_SS_vector, startpoints_SS_vector, type_subsection, name_subsection, sectioned_chainages, before_switch and after_switch**
 - Respectively, these arrays contain the x coordinate of the midpoint of the section, the y coordinate of the midpoint of the section, the features to copy, the list of standard deviation values for the longitudinal levels, the track quality class for the longitudinal levels, the list of standard deviation values for the alignment,, the track quality class for the alignments, the endpoints of the subsections, the startpoints of the subsections, the type of the subsection (all spoortak), the name of the subsection, the sectioned chainages of the subsections, the before data of the switches and finally the after data
- **Susbsection_length = 200**
- **Subsection_window_min = subsection_length/sampling frequency**
 - This is the number of points that the susbsection will continue as a threshold
- **Num_sections = length of read_file_two**
- **For i ranging from 0 to num_sections**
- **Get the midpoint index by averaging the startpoint and endpoint**
- **Add to X_midpoint and Y_midpoint the coordinates of the midpoint**
- **if the section length of spoortak of row i is less than the threshold (200m)**
 - In this case no breaking down of the sections is needed
- **Copy all the relevant features to the respective rows**
- **Using the endpoints and the startpoints calculate the average moving standard deviation values for alignments and longitudinal levels**
 - The startpoints and endpoints are the indices that go from 0 to 351,851 and refer to the index of the points in the linked data file containing the RTG data (Appendix A)
- **Classify the track quality using the tables from EN13848-6 (figures 10 and 11) and add the classifications to col3 and col3_al, end if**
 - The case where section length is already less than the threshold is straightforward and easy. No changes have to made. The only thing that was done is the std calculation and the track classification

- **Else**
- **Startpoints_SS = startpoint of the section to be broken down**
- **Total_num_widows = endpoint of the section – startpoint of the section**
- **Remainder= Total_num_widows modulo subsection_window_min**
 - this gives us the remainder of the points once the threshold (200m) sectioning has happened
- **num_sub_sections=1+((total_num_windows-remainder)/sub_section_window_min)**
 - this gives us the total number of subsections that the section will be broken into
- **windows_sizes = array of the size of the number of subsections where all the entries are the subsection_window_min values and the last entry is the remainder**
 - so for the case of a 5420 m section, this will be broken down into 27 sections of 200 m and 1 section of 20 m. The windows_sizes array will be 27 entries of 800 and 1 entry of 80. These represent the number of points that each section will contain
- **for j ranging from 0 to number of subsections**
- **if j = 0**
- **give that section the before switch information**
 - obviously the first section to be broken down gets the before switch information and the last section gets the after switch information
- **Add 0's to the after array, end if**
- **Else**
 - This is the case of the middle sections
- **Add 0's to the before and after arrays, end else**
 - Middle sections have no switches around them
- **Endpoints_SS = startpoints_SS + window_sizes [j]**
 - The new endpoint is the startpoint of the subsection + the jth row value of the window sizes
- **Get the midpoint index by averaging the startpoint and endpoint**
- **Add to X_midpoint and Y_midpoint the coordinates of the midpoint**
- **Copy all the relevant features to the respective rows**
- **Using the endpoints and the startpoints calculate the standard deviation values for alignments and longitudinal levels**
- **Classify the track quality using the tables from EN13848-6 (figures 10 and 11) and add the classifications to col3 and col3_al, end for**
- **Endpoints_SS = startpoints_SS + the last value of the window_sizes array**
 - In every step the startpoints were calculated. The for loop stops at the penultimate section and the last section has to be catered for manually
- **Get the midpoint index by averaging the startpoint and endpoint**
- **Add to X_midpoint and Y_midpoint the coordinates of the midpoint**
- **Add all relevant features to their respective arrays**
- **Add the after data for the switches**
 - The last subsection is faced with a switch or not after it!
- **Add 0's to the before array**
- **Using the endpoints and the startpoints calculate the standard deviation values for alignments and longitudinal levels**
- **Classify the track quality using the tables from EN13848-6 (figures 10 and 11) and add the classifications to col3 and col3_al, end for**

Task 2:

- *Turn the track qualities of the alignment into numbers: A is 5, B is 4 etc... all the way to E is 1*
- *Quality_combined= array of two columns where one column is for alignment and the other is for longitudinal level*
- *Initialize array: mutual_quality*
- *For i ranging from 0 to length of combined quality*
- *Get the minimum value of each row*
- *Turn that value into a letter again as shown above, end for*
- *Turn everything into pandas dataframe, combine them and extract the table*

The libraries used for this Python script are:

- Pandas: <https://pandas.pydata.org/>
- Numpy: <https://numpy.org/>
- Statistics: <https://docs.python.org/3/library/statistics.html>

Appendix B.6.2: Section Thresholding Output Table (Utrecht-Den Bosch)

Some of the features that were already shown above were hidden to fit in this page. What is important here is to notice the section lengths and the combined track quality class column.

Section_Ch	X_Midpoint	Y_Midpoint	Length(m)	Combined_Quality	Speed(km/h)	Tonnage (ton/day)	Ballast_Type	...	...	Age_R (days)
0.0-0.199967541752326 km	148800.3	412231.4	200	A	130	50811.1	Steenslag 31,5/50	...	...	1999
0.199967541752326-0.355435840344926 km	148865.7	412396.5	155.5	A	130	50811.1	Steenslag 31,5/50	...	...	1999
0.355685737307303-0.5556667277511012 km	148895.2	412571.6	200	A	130	50811.1	Steenslag 31,5/50	...	...	-573
0.5556667277511012-0.7556514818759359 km	148899.5	412771.4	200	A	130	50811.1	Steenslag 31,5/50	...	...	-573
0.7556514818759359-0.9556318711310479 km	148876.2	412969.9	200	B	130	50811.1	Steenslag 31,5/50	...	...	-573
0.9556318711310479-1.15561245152715 km	148827.5	413163.7	200	D	130	50811.1	Steenslag 31,5/50	...	...	-573
1.15561245152715-1.35560365149605 km	148752.5	413348.9	200	E	130	50811.1	Steenslag 31,5/50	...	...	-573
1.35560365149605-1.55560400667382 km	148662.8	413527.7	200	D	130	50811.1	Steenslag 31,5/50	...	...	-573
1.55560400667382-1.75559690777646 km	148572.8	413706.3	200	D	130	50811.1	Steenslag 31,5/50	...	...	-573
1.75559690777646-1.95559898308907 km	148482.8	413884.9	200	D	130	50811.1	Steenslag 31,5/50	...	...	-573
1.95559898308907-2.1555976312093 km	148392.7	414063.5	200	D	130	50811.1	Steenslag 31,5/50	...	...	-573
2.1555976312093-2.35559892487909 km	148302.7	414242	200	E	130	50811.1	Steenslag 31,5/50	...	...	-573
2.35559892487909-2.55559698506136 km	148212.5	414420.6	200	C	130	50811.1	Steenslag 31,5/50	...	...	-573
2.55559698506136-2.75559758075765 km	148122.4	414599.1	200	C	130	50811.1	Steenslag 31,5/50	...	...	-573
2.75559758075765-2.95559639155157 km	148032.2	414777.6	200	C	130	50811.1	Steenslag 31,5/50	...	...	-573
2.95559639155157-3.15559716612362 km	147942.1	414956.2	200	C	130	50811.1	Steenslag 31,5/50	...	...	-573
3.15559716612362-3.35559808012517 km	147852	415134.7	200	C	130	50811.1	Steenslag 31,5/50	...	...	-573
3.35559808012517-3.55559512953129 km	147761.9	415313.3	200	C	130	50811.1	Steenslag 31,5/50	...	...	-573
3.55559512953129-3.75559533237455 km	147671.8	415491.8	200	C	130	50811.1	Steenslag 31,5/50	...	...	-573
3.75559533237455-3.95560067366175 km	147581.8	415670.4	200	C	130	50811.1	Steenslag 31,5/50	...	...	-573
3.95560067366175-4.15559713723001 km	147491.7	415849	200	C	130	50811.1	Steenslag 31,5/50	...	...	-573
4.15559713723001-4.355602995859201 km	147401.7	416027.6	200	D	130	50811.1	Steenslag 31,5/50	...	...	-573
...	...	...	...	...	...	...	...	...	...	...
87.9311995226264-88.08694569917009 km	148785.4	412212.2	155.75	A	130	49138	Steenslag 31,5/50	...	...	540

Appendix C: Machine Learning Classification Model Performance Results

In this section, the comparison of results will be made between the algorithms themselves, the section length used (200m, 200m+remainder), and the classes used (ABCDE vs good, medium, bad). Also, the pseudo-codes used for the hyperparameter tuning will be displayed. Finally, this appendix will also display some tables generated for the inference track.

Appendix C.1: Performance results of varied track quality aggregation methods

The default method is to use the track qualities ranging from A all the way to E. The second method takes the track qualities of A and B and aggregates them into “Good”, C into “Medium”, D and E into “Bad”.

For the Random Forest model:

Random Forest: ABCDE				
	precision	recall	f1-score	support
A	0.75	0.6	0.67	5
B	0	0	0	1
C	0.61	0.58	0.59	38
D	0.42	0.65	0.51	34
E	0.57	0.29	0.38	28
accuracy			0.52	106
macro avg	0.47	0.42	0.43	106
weighted avg	0.54	0.52	0.51	106

Random Forest: Good/Medium/Bad				
	precision	recall	f1-score	support
Bad	0.73	0.77	0.75	62
Good	0.75	0.5	0.6	6
Medium	0.61	0.58	0.59	38
accuracy			0.69	106
macro avg	0.7	0.62	0.65	106
weighted avg	0.69	0.69	0.69	106

For the XGBoost model:

XGboost: ABCDE				
	precision	recall	f1-score	support
A	0.67	0.4	0.5	5
B	0	0	0	1
C	0.59	0.53	0.56	38
D	0.4	0.65	0.49	34
E	0.58	0.25	0.35	28
accuracy			0.48	106
macro avg	0.45	0.36	0.38	106
weighted avg	0.52	0.48	0.47	106

XGboost: Good/Medium/Bad				
	precision	recall	f1-score	support
Bad	0.73	0.79	0.76	62
Good	0.4	0.33	0.36	6
Medium	0.59	0.53	0.56	38
accuracy			0.67	106
macro avg	0.57	0.55	0.56	106
weighted avg	0.66	0.67	0.66	106

For the SVM model:

SVM: ABCDE				
	precision	recall	f1-score	support
0	0.25	0.2	0.22	5
1	0	0	0	1
2	0.64	0.24	0.35	38
3	0.34	0.82	0.48	34
4	0.33	0.07	0.12	28
accuracy			0.38	106
macro avg	0.31	0.27	0.23	106
weighted avg	0.44	0.38	0.32	106

SVM: Good/Medium/Bad				
	precision	recall	f1-score	support
Bad	0.61	0.87	0.72	62
Good	0.25	0.17	0.2	6
Medium	0.64	0.24	0.35	38
accuracy			0.6	106
macro avg	0.5	0.42	0.42	106
weighted avg	0.6	0.6	0.56	106

Appendix C.2: Performance results of varied track section length used

The tables of the model performances will be shown for each section length used: 200m and 200m+remainder.

For the Random Forest model:

Random Forest: 200m sections				
	precision	recall	f1-score	support
Bad	0.79	0.75	0.77	64
Good	0	0	0	3
Medium	0.47	0.54	0.5	28
accuracy			0.66	95
macro avg	0.42	0.43	0.42	95
weighted avg	0.67	0.66	0.66	95

Random Forest: 200m + remainder sections				
	precision	recall	f1-score	support
Bad	0.73	0.77	0.75	62
Good	0.75	0.5	0.6	6
Medium	0.61	0.58	0.59	38
accuracy			0.69	106
macro avg	0.7	0.62	0.65	106
weighted avg	0.69	0.69	0.69	106

For the XGBoost model:

XGBoost: 200m sections				
	precision	recall	f1-score	support
Bad	0.74	0.67	0.7	64
Good	0.25	0.33	0.29	3
Medium	0.36	0.43	0.39	28
accuracy			0.59	95
macro avg	0.45	0.48	0.46	95
weighted avg	0.61	0.59	0.6	95

XGboost: 200m + remainder sections				
	precision	recall	f1-score	support
Bad	0.73	0.79	0.76	62
Good	0.4	0.33	0.36	6
Medium	0.59	0.53	0.56	38
accuracy			0.67	106
macro avg	0.57	0.55	0.56	106
weighted avg	0.66	0.67	0.66	106

For the SVM model:

SVM: 200m sections				
	precision	recall	f1-score	support
Bad	0.73	0.95	0.82	64
Good	0	0	0	3
Medium	0.64	0.25	0.36	28
accuracy			0.72	95
macro avg	0.45	0.4	0.39	95
weighted avg	0.68	0.72	0.66	95

SVM: 200m + remainder sections				
	precision	recall	f1-score	support
Bad	0.61	0.87	0.72	62
Good	0.25	0.17	0.2	6
Medium	0.64	0.24	0.35	38
accuracy			0.6	106
macro avg	0.5	0.42	0.42	106
weighted avg	0.6	0.6	0.56	106

Appendix C.3: Performance results with the two most important features omitted

The 200m + remainder sections were used along with the aggregated track qualities.

Random Forest: deform_soil, BCSF_soil omitted, 200m + remainder sections				
	precision	recall	f1-score	support
Bad	0.67	0.74	0.7	62
Good	0.6	0.5	0.55	6
Medium	0.5	0.42	0.46	38
accuracy			0.61	106
macro avg	0.59	0.55	0.57	106
weighted avg	0.6	0.61	0.61	106

XGboost: deform_soil, BCSF_soil omitted, 200m + remainder sections				
	precision	recall	f1-score	support
Bad	0.66	0.71	0.68	62
Good	0.33	0.33	0.33	6
Medium	0.48	0.42	0.45	38
accuracy			0.58	106
macro avg	0.49	0.49	0.49	106
weighted avg	0.58	0.58	0.58	106

SVM: deform_soil, BCSF_soil omitted, 200m + remainder sections				
	precision	recall	f1-score	support
Bad	0.6	0.85	0.7	62
Good	0	0	0	6
Medium	0.5	0.18	0.27	38
accuracy			0.57	106
macro avg	0.37	0.35	0.32	106
weighted avg	0.53	0.57	0.51	106

Appendix C.4: Performance results with all the un-important features omitted

The 200m + remainder sections were used along with the aggregated track qualities.

Random Forest: deform_soil, BCSF_soil omitted, 200m + remainder sections				
	precision	recall	f1-score	support
Bad	0.67	0.74	0.7	62
Good	0.6	0.5	0.55	6
Medium	0.5	0.42	0.46	38
accuracy			0.61	106
macro avg	0.59	0.55	0.57	106
weighted avg	0.6	0.61	0.61	106

XGboost: deform_soil, BCSF_soil omitted, 200m + remainder sections				
	precision	recall	f1-score	support
Bad	0.66	0.71	0.68	62
Good	0.33	0.33	0.33	6
Medium	0.48	0.42	0.45	38
accuracy			0.58	106
macro avg	0.49	0.49	0.49	106
weighted avg	0.58	0.58	0.58	106

SVM: deform_soil, BCSF_soil omitted, 200m + remainder sections				
	precision	recall	f1-score	support
Bad	0.6	0.85	0.7	62
Good	0	0	0	6
Medium	0.5	0.18	0.27	38
accuracy			0.57	106
macro avg	0.37	0.35	0.32	106
weighted avg	0.53	0.57	0.51	106

Appendix C.5: Hyperparameter tuning pseudo-code

This appendix highlights the hyperparameter tuning pseudo-code that was used for the Random Forest algorithm. The same template was used for the XGboost and the SVM models. The only differences are the pre-processing that the latter two algorithms require and this was already discussed in section 4.1, and the hyperparameters that were tuned which have been discussed in section 5.

- **features = finalized features excel file**
 - The table shown in appendix B.6.2
- **Features_Str=**
['Tonnage_To','Ballast_Ag','Transition_Count','Sleeper_Ag','Rail_age','Switch_a_1','Switch_b_1','Ballast_Th','Section_Le','Deform_Soil','BCSF_Soil']
 - This is to specify which features will be used for the finalized tuned model. These are the most important features.
- **Features=features[Features_Str]**
 - This is to select the features that will be used for the model
- **Pre-process the switches data to change them into numerical data instead of categorical data**
 - Slopes such as “1:9” were calculated and input as float numbers
- **Pre-process the ballast thickness data and change them into numerical data instead of string data**
 - Thicknesses such as “15-20cm” were changed into 17.5
- **Pre-processing the negative ages and changing them into positive values that are 10 times the maximum value**
- **Turning the labels A/B/C/D/E into Good/Medium/Bad**
 - A-B: Good
 - C: Medium
 - D-E: Bad
- **Labels=features[‘Combined Quality’]**
 - This is to define the labels used for the model. Those were changed already to Good/Medium/Bad
- **Feature_list=list(features.columns)**
 - This is to extract the names of the features that have been used. It is the same as Features_Str, however in a list format
- **Define: train_features, test_features, train_labels, test_labels using train_test_split function**
 - Test size is chosen to be 25%, and training 75%
 - This function also takes the labels and the features as input
- **rf= RandomForestClassifier()**
 - This is to define the default model
- **rf.fit(train_features, train_labels)**
 - Fitting the default model
- **Define the hyperparameters and their ranges:**
- **n_estimators=[200...2000 with 10 numbers in between]**
 - This creates an array of 10 integers between 200 and 2000
- **max_features=[‘auto’,‘sqrt’]**
 - This defines two methods of choosing the maximum features
- **Max_depth=[10...110 with 11 numbers in between], add [None] to Max_depth**
 - This defines the maximum depths that can be used with an option of no maximum

- **Define `random_grid` as the list containing all the hyperparameters and their names**
- **`rf_random=RandomizedSearchCV(estimator=rf, param_distribution=random_grid..., cv=3, n_iter=100)`**
 - *This is defining the randomized search and the number of cross-validations for each parameter choice: 3. The iterations are 100, thus 300 fits are being made.*
- **`rf_random.fit(train_features,train_labels)`**
 - *Fit the model with the training dataset, with 300 different fits.*
- **`best_random = rf_random.best_estimator_`**
 - *Taking the best parameters from the random estimator and saving them to "best_random"*
- **`Predictions_random=best_random.predict(test_features)`**
- **`print(classification_report(test_labels, predictions))`**
 - *Get the accuracy parameters of the predicted test labels from the best_random model*
- **Create a grid with the best parameters +- some values: `param_grid`**
 - *'n_estimators': [190,200,210] if the best random gave 200 as the best value*
- **`grid_search = GridSearchCV(estimator = rf, param_grid = param_grid, cv = 3, n_jobs = -1, verbose = 2)`**
 - *A 3-fold cross validation with X combinations in the grid. This gives 3X fits being made.*
- **`best_grid = grid_search.best_estimator_`**
 - *Save the best parameters in best_grid*
- **`predictions=best_grid.predict(test_features)`**
- **`print(classification_report(test_labels, predictions))`**
 - *Get the accuracy parameters of the predictions done from the best_grid*
- **`scores=cross_val_scores(best_grid, features, labels_temp, cv=6,scoring='accuracy')`**
 - *This is to evaluate the model and validate it using cross validation with a 6-fold cross validation method. The scoring method is based on accuracy. The validation is being done on the best_grid model*
- **`filename = 'finalized_model_randomforest.sav'`**
- **`pickle.dump(best_random, open(filename, 'wb'))`**
 - *This is to save the model with its hyperparameters to use on a different set of data: inference.*

The libraries used for this Python script are:

- Pandas: <https://pandas.pydata.org/>
- Numpy: <https://numpy.org/>
- Statistics: <https://docs.python.org/3/library/statistics.html>
- Sklearn: <https://scikit-learn.org/stable/>

Appendix C.6: Hyperparameter tuning performance measures and CV accuracies

In this section, the tuned models' parameters will be shown for each algorithm along with the performance measures. Moreover, the accuracies of the 6-fold cross validation will be shown.

Random Forest:

Random Forest: Hyperparameters tuned				
	precision	recall	f1-score	support
Bad	0.77	0.94	0.85	69
Good	0.67	0.33	0.44	6
Medium	0.8	0.5	0.62	32
accuracy			0.78	107
macro avg	0.75	0.59	0.64	107
weighted avg	0.78	0.78	0.76	107

Hyperparameters: Random Forest
{'bootstrap': True,
'criterion': 'entropy',
'max_depth': 30,
'max_features': 'sqrt',
'min_samples_leaf': 4,
'min_samples_split': 6,
'n_estimators': 800}

Cross Validation: Accuracies					
0.61971831	0.45070423	0.450704	0.53521127	0.5915493	0.2

XGboost:

XGboost: Hyperparameters tuned				
	precision	recall	f1-score	support
Bad	0.75	0.84	0.79	69
Good	0.5	0.5	0.5	6
Medium	0.67	0.5	0.57	32
accuracy			0.72	107
macro avg	0.64	0.61	0.62	107
weighted avg	0.71	0.72	0.71	10

Hyperparameters: XGboost
{'objective': 'multi:softmax',
'n_estimators': 790,
'min_child_weight': 4,
'max_depth': 4,
'learning_rate': 0.01,
'label_encoder': False,
'eval_metric': 'merror'}

Cross Validation: Accuracies					
0.61971831	0.30985915	0.338028	0.32394366	0.5492958	0.228571

SVM:

SVM: Hyperparameters tuned				
	precision	recall	f1-score	support
Bad	0.73	0.88	0.8	69
Good	0.4	0.33	0.36	6
Medium	0.72	0.41	0.52	32
accuracy			0.71	107
macro avg	0.62	0.54	0.56	107
weighted avg	0.71	0.71	0.69	107

Hyperparameters: SVM
{'C': 1.0,
'break_ties': False,
'cache_size': 200,
'class_weight': None,
'coef0': 0.0,
'decision_function_shape': 'ovr',
'degree': 3,
'gamma': 'scale',
'kernel': 'poly',
'max_iter': -1,
'probability': False,
'random_state': None,
'shrinking': True,
'tol': 0.001,
'verbose': False}

Cross Validation: Accuracies					
0.61971831	0.61971831	0.605634	0.5915493	0.6197183	0.514286

Appendix D: Model Inference Module

This appendix is used to show the results of the inference module. The results include the second way generalization, where only one way will be used to train the model and the second way will be predicted. The results will also show the prediction results of a small piece of a two-way track on the Utrecht-Den Bosch line. Finally, the output of the data gathering module will be shown for the 7km one way track of Den Bosch-Eindhoven and the prediction results for that track.

Appendix D.1: Second Way Inference Performance Results

Second-way inference: Random Forest				
	precision	recall	f1-score	support
Bad	0.77	0.73	0.75	138
Good	1	0.23	0.38	13
Medium	0.48	0.61	0.54	61
accuracy			0.67	212
macro avg	0.75	0.52	0.55	212
weighted avg	0.7	0.67	0.66	212

Second-way inference: XGboost				
	precision	recall	f1-score	support
Bad	0.7	0.74	0.72	138
Good	0.36	0.38	0.37	13
Medium	0.38	0.33	0.35	61
accuracy			0.6	212
macro avg	0.48	0.48	0.48	212
weighted avg	0.59	0.6	0.59	212

Second-way inference: SVM				
	precision	recall	f1-score	support
Bad	0.65	1	0.79	138
Good	0	0	0	13
Medium	0	0	0	61
accuracy			0.65	212
macro avg	0.22	0.33	0.26	212
weighted avg	0.42	0.65	0.51	212

Appendix D.2: Last Piece of Track Inference Performance Results

Random Forest: Second Inference Test				
	precision	recall	f1-score	support
Bad	0.61	1	0.76	33
Good	1	0.4	0.57	10
Medium	0	0	0	15
accuracy			0.64	58
macro avg	0.54	0.47	0.44	58
weighted avg	0.52	0.64	0.53	58

XGBoost: Second Inference Test				
	precision	recall	f1-score	support
Bad	0.59	1	0.74	33
Good	1	0.2	0.33	10
Medium	0	0	0	15
accuracy			0.6	58
macro avg	0.53	0.4	0.36	58
weighted avg	0.51	0.6	0.48	58

SVM: Second Inference Test				
	precision	recall	f1-score	support
Bad	0.52	0.79	0.63	33
Good	0	0	0	10
Medium	0.12	0.07	0.09	15
accuracy			0.47	58
macro avg	0.21	0.28	0.24	58
weighted avg	0.33	0.47	0.38	58

Appendix D.3: Section Thresholding Output Table (Den Bosch Eindhoven Track)

Some of the features that were already shown above were hidden to fit in this page. This table represents the finalized sections of the inference data. This section has undergone the same process as the Utrecht-Den Bosch track.

Section_Ch	X_Midpoint	Y_Midpoint	Length(m)	Combined_Quality	Speed(km/h)	Tonnage (ton/day)	Ballast_Type	...	...	Age_R (days)
0.0925073825300316-0.1132573520437 km	148507.2	411280.7	20.75	A	80	23094.61	Steenslag 31,5/50	...	...	79958
0.441248269014863-0.459998079260267 km	148423.9	410943.7	18.75	E	40	23657.2	Steenslag 31,5/50	...	...	79958
0.489249997915755-0.506999891634475 km	148415.1	410897.4	17.75	E	80	44953.59	Steenslag 31,5/50	...	...	1720
0.555499998134636-0.5619999723482951 km	148400.1	410838.8	6.5	E	80	48437.52	Steenslag 31,5/50	...	...	1720
0.5912488131107729-0.6262487435776221 km	148387.7	410790.6	35	E	80	40836.71	Steenslag 31,5/50	...	...	1720
0.7462399990799959-0.854989624245961 km	148340.7	410605.2	108.75	D	130	39003.42	Steenslag 31,5/50	...	...	1720
0.85523962413739-0.9819866342582908 km	148327.4	410488.2	126.75	D	130	39003.42	Steenslag 31,5/50	...	...	5844
0.9822366201401469-1.1822339113613 km	148340.1	410325	200	C	130	39003.42	Steenslag 31,5/50	...	...	1720
1.1822339113613-1.38223352397914 km	148358.1	410125.8	200	A	130	39003.42	Steenslag 31,5/50	...	...	1720
1.38223352397914-1.58223340611479 km	148376.2	409926.7	200	B	130	39003.42	Steenslag 31,5/50	...	...	1720
1.58223340611479-1.78223315030297 km	148394.2	409727.5	200	C	130	39003.42	Steenslag 31,5/50	...	...	1720
1.78223315030297-1.98223299162323 km	148400.3	409527.8	200	D	130	39003.42	Steenslag 31,5/50	...	...	1720
1.98223299162323-2.18223285090054 km	148368.8	409330.5	200	C	130	39003.42	Steenslag 31,5/50	...	...	1720
2.18223285090054-2.3822327118948 km	148325	409135.4	200	B	130	39003.42	Steenslag 31,5/50	...	...	1720
...	...	...	...	...	...	...	...	...	...	...
7.585973506548821-7.625723316826191 km	148860.3	403856.4	39.75	B	130	47769.61	Steenslag 31,5/50	...	...	8035

Appendix D.4: Den Bosch Eindhoven Track Prediction Performance Results

Random Forest: Inference prediction parameters				
	precision	recall	f1-score	support
Bad	0.32	0.75	0.44	16
Good	0.17	0.08	0.11	12
Medium	0	0	0	16
accuracy			0.3	44
macro avg	0.16	0.28	0.19	44
weighted avg	0.16	0.3	0.19	44

XGBoost: Inference prediction parameters				
	precision	recall	f1-score	support
Bad	0.34	0.62	0.44	16
Good	0	0	0	12
Medium	0.25	0.12	0.17	16
accuracy			0.27	44
macro avg	0.2	0.25	0.2	44
weighted avg	0.22	0.27	0.22	44

SVM: Inference prediction parameters				
	precision	recall	f1-score	support
Bad	0.36	1	0.53	16
Good	0	0	0	12
Medium	0	0	0	16
accuracy			0.36	44
macro avg	0.12	0.33	0.18	44
weighted avg	0.13	0.36	0.19	44

References

- 2021: the European Year of Rail | News | European Parliament. (2021, January 12). Retrieved from <https://www.europarl.europa.eu/news/en/headlines/eu-affairs/20210107STO95106/2021-the-european-year-of-rail#:~:text=The%20EU%20has%20designated%202021,the%20European%20Year%20of%20Rail>
- Andrade, A. R., & Teixeira, P. F. (2014). Unplanned-maintenance needs related to rail track geometry. *Proceedings of the Institution of Civil Engineers - Transport*, 167(6), 400–410. <https://doi.org/10.1680/tran.11.00060>
- Borda-De-Água, L., Barrientos, R., Beja, P., & Pereira, H. M. (2017). *Railway Ecology* (1st ed. 2017 ed.). Vairao, Portugal: Springer.
- Falamarzi, A., Moridpour, S., & Nazem, M. (2019). A review of rail track degradation prediction models. *Australian Journal of Civil Engineering*, 17(2), 152–166. <https://doi.org/10.1080/14488353.2019.1667710>
- Grossoni, I., Le Pen, L. M., Jorge, P., Bezin, Y., Watson, G. V. R., Kostovasilis, D., & Powrie, W. (2019). The role of stiffness variation in switches and crossings: Comparison of vehicle–track interaction models with field measurements. *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit*, 234(10), 1184–1197. <https://doi.org/10.1177/0954409719892146>
- McKinsey, Kienzler, C., Lotz, C., & Stern, S. (2020, June). Using analytics to get European rail maintenance on track. Retrieved from <https://www.mckinsey.com/industries/public-and-social-sector/our-insights/using-analytics-to-get-european-rail-maintenance-on-track>
- Schroten, A., van Wijngaarden, L., Brambilla, M., Maffii, S., Trosky, F., Kramer, H., ... Amaral, S. (2019). Overview of transport infrastructure expenditures and costs. Brussels, Belgium: European Commission.
- Smith, R. (2001). *Railway Technology—The Last 50 Years and Future Prospects*. Japan Railway & Transport Review, 27, 16–24. Retrieved from https://www.ejrcf.or.jp/jrtr/jrtr27/pdf/f16_smi.pdf
- Alemazkoo, N., C. J. Ruppert, and H. Meidani. 2015. “Track Geometry Analytics.” In *RAS Problem Solving Competition Report*. Mumbai.
- Mannila, H., 2016. Data mining: machine learning, statistics, and databases. In: *Int’l Conf. Scientific and Statistical Database Management*. IEEE Computer Society.
- Lewis, H. G., & Brown, M. (2001). A generalized confusion matrix for assessing area estimates from remotely sensed data. *International journal of remote sensing*, 22(16), 3223-3235.
- CEN (European Committee for Standardization). (2011). EN 13848-1 Railway applications-Track-Track geometry quality - Part 1: Characterization of track geometry
- CEN (European Committee for Standardization). (2011). EN 13848-6 Railway applications - Track - Track geometry quality - Part 6: Characterisation of track geometry quality
- Bronsvoort, E., & Hurk, N. Van Den. (2013). The RILA system - now in the UK. *Rail Infrastructure*, 57-59.
- Hurk, N. Van Den. (2016, May). Track surveying at 100mph. *Rail Infrastructure*.

Markine V, (2020). Conventional Track [PowerPoint slides]. Elements of Railway Engineering, TU Delft. <https://brightspace.tudelft.nl/d2l/le/content/193510/Home>

Markine V, (2020). Non-Conventional Track [PowerPoint slides]. Elements of Railway Engineering, TU Delft. <https://brightspace.tudelft.nl/d2l/le/content/193510/Home>

Markine V, (2020). Track Forces [PowerPoint slides]. Elements of Railway Engineering, TU Delft. <https://brightspace.tudelft.nl/d2l/le/content/193510/Home>

Markine V, (2020). [Inspection and Monitoring](#) [PowerPoint slides]. Elements of Railway Engineering, TU Delft. <https://brightspace.tudelft.nl/d2l/le/content/193510/Home>

Markine V, (2020). Performance in time [PowerPoint slides]. Elements of Railway Engineering, TU Delft. <https://brightspace.tudelft.nl/d2l/le/content/193510/Home>

Yang Z, (2020). Introduction of wheel-rail interface and contact mechanics [PowerPoint slides]. Wheel-Rail Interface, TU Delft. <https://brightspace.tudelft.nl/d2l/le/content/193508/Home>

B. Zuada Coelho, (2013). Dynamics of railway transition zones in soft soils, Delft University of Technology

Ben-Daya, M., Kumar, U., & Murthy, P. D. (2016). Introduction to Maintenance Engineering: Modelling, Optimization and Management. London: John Wiley & Sons

Alves Ribeiro, A., R. Calçada, and R. Delgado (2007). Dynamic effects induced by abrupt changes in track stiffness in high-speed railway lines. In Sousa, Fernandes, Vargas, and Azevedo (Eds.), Applications of Computational Mechanics in Geotechnical Engineering, pp. 473–483. Taylor & Francis.

Lundqvist, A., R. Larsson, and T. Dahlberg (2006). Influence of railway track stiffness variations on wheel/rail contact force. In Track for High-Speed Railways. FEUP.

McAlpine, R., (1991) Sleepers – Design, Manufacture, Supply and Performance, Railway Civil Engineering Course, Vol. 3 Paper B5

Support Vector Machines — scikit-learn 0.24.2 documentation. (2004). Retrieved from [https://scikitlearn.org/stable/modules/svm.html#:~:text=Support%20vector%20machines%20\(SVMs\)%20are,than%20the%20number%20of%20samples.](https://scikitlearn.org/stable/modules/svm.html#:~:text=Support%20vector%20machines%20(SVMs)%20are,than%20the%20number%20of%20samples.)

Donges, N. (2020, September 3). A complete guide to the random forest algorithm. Retrieved from <https://builtin.com/data-science/random-forest-algorithm>

Brownlee, J. (2021, February 16). A Gentle Introduction to XGBoost for Applied Machine Learning. Retrieved from <https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/>

CEN, (2009). EN 12299:2009 Ride comfort for passengers - Measurement and evaluation

Mechanical properties of Rails - ArcelorMittal Rails. (2018). Retrieved May 31, 2021, from <https://rails.arcelormittal.com/products/transport-rails/mechanical-properties>

Betondwarsligger, m rughellingplaat 54E1. (2021). Retrieved June 1, 2021, from <https://www.railpro.online/p/betondwarsligger-m-rughellingplaat-54e1/5201/>

- Classification: Accuracy | Machine Learning Crash Course. (2020, February). Retrieved June 8, 2021, from <https://developers.google.com/machine-learning/crash-course/classification/accuracy>
- Plański, P. (2020, June 29). Random Forest Feature Importance Computed in 3 Ways with Python. Retrieved June 10, 2021, from <https://mljar.com/blog/feature-importance-in-random-forest/>
- Saxena, R. (2017, April 21). How Decision Tree Algorithm works. Retrieved June 10, 2021, from <https://dataaspirant.com/how-decision-tree-algorithm-works/>
- Alabbasi, Y., & Hussein, M. (2019). Geomechanical Modelling of Railroad Ballast: A Review. *Archives of Computational Methods in Engineering*, 28(3), 815–839. <https://doi.org/10.1007/s11831-019-09390-4>
- Andrade, A. R., & Teixeira, P. F. (2014). Unplanned-Maintenance needs related to Rail Track Geometry. *Proceedings of the Institution of Civil Engineers: Transport*, 167(6), 400–410. <https://doi.org/10.1680/tran.11.00060>
- Ben-Hur, A., & Weston, J. (2010). A user's guide to support vector machines. *Methods in Molecular Biology (Clifton, N.J.)*, 609, 223–239. https://doi.org/10.1007/978-1-60327-241-4_13
- Berghuis, S. (2017). *Spoortak en Wissel*. 1–4.
- Cannon, W. F. (2014). Worldwide Supply of and Demand for Manganese How Do We Ensure Adequate Supplies of Manganese for the Future? *USGS Mineral Resource Program, August*, 3087. <http://minerals.usgs.gov/minerals/pubs/>
- Devan, P., & Khare, N. (2020). An efficient XGBoost–DNN-based classification model for network intrusion detection system. *Neural Computing and Applications*, 32(16), 12499–12514. <https://doi.org/10.1007/s00521-020-04708-x>
- Falamarzi, A., Moridpour, S., & Nazem, M. (2019). A review of rail track degradation prediction models. *Australian Journal of Civil Engineering*, 17(2), 152–166. <https://doi.org/10.1080/14488353.2019.1667710>
- Falamarzi, A., Moridpour, S., Nazem, M., & Hesami, R. (2018). Rail degradation prediction models for tram system: Melbourne case study. *Journal of Advanced Transportation*, 2018(May). <https://doi.org/10.1155/2018/6340504>
- Gambill, D. R., Wall, W. A., Fulton, A. J., & Howard, H. R. (2016). Predicting USCS soil classification from soil property variables using Random Forest. *Journal of Terramechanics*, 65, 85–92. <https://doi.org/10.1016/j.jterra.2016.03.006>
- Giannakos, K. (2014). Actions on a Railway Track, due to an Isolated Defect. *International Journal of Control and Automation*, 7(3), 195–212. <https://doi.org/10.14257/ijca.2014.7.3.19>
- Goodall, R. (2003). *Active suspension technology and its effect upon vehicle-track interaction. System dynamics and long-term behaviour of railway vehicles, track and subgrade*.
- Grassie, S. L. (2012). Rail irregularities, corrugation and acoustic roughness: Characteristics, significance and effects of reprofiling. *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit*, 226(5), 542–557. <https://doi.org/10.1177/0954409712443492>

- Indraratna, B., Ionescu, D., & Christie, H. D. (1998). Shear Behavior of Railway Ballast Based on Large-Scale Triaxial Tests. *Journal of Geotechnical and Geoenvironmental Engineering*, 124(5), 439–449. [https://doi.org/10.1061/\(asce\)1090-0241\(1998\)124:5\(439\)](https://doi.org/10.1061/(asce)1090-0241(1998)124:5(439))
- International Union of Railways. (2010). *INNOTRACK concluding technical report*.
- Jiang, Y., Chen, B. K., & Thompson, C. (2019). A comparison study of ride comfort indices between Sperling's method and EN 12299. *International Journal of Rail Transportation*, 7(4), 279–296. <https://doi.org/10.1080/23248378.2019.1616329>
- Lasisi, A., & Atttoh-Okine, N. (2018). Principal components analysis and track quality index: A machine learning approach. *Transportation Research Part C: Emerging Technologies*, 91(March 2018), 230–248. <https://doi.org/10.1016/j.trc.2018.04.001>
- Lichtberger, B. (2005). Track Compendium. *Hamburg, Eurailpress Tetzlaff-Hestra GmbH & Co. Publ*, 1–192.
- Liu, R. K., Xu, P., Sun, Z. Z., Zou, C., & Sun, Q. X. (2015). Establishment of track quality index standard recommendations for beijing metro. *Discrete Dynamics in Nature and Society*, 2015. <https://doi.org/10.1155/2015/473830>
- Lopes, L., & Papaalias, M. (2017). *INFLUENCES OF SLEEPER / CROSSTIE MATERIAL CHOICES ON GEORISKS IN. May*.
- López-Pita, A., Teixeira, P. F., Casas, C., Bachiller, A., & Ferreira, P. A. (2008). Maintenance costs of high-speed lines in Europe state of the art. *Transportation Research Record*, 2043, 13–19. <https://doi.org/10.3141/2043-02>
- Madalina Dumitriu, & Dragos, Ionut, S. (2021). *Study on the Evaluation Methods of the Vertical Ride Comfort of Railway Vehicle—Mean Comfort Method and Sperling's Method*.
- Munoz, A. (2015). Online Learning and Optimization. *Encyclopedia of Algorithms*, 1–8. https://doi.org/10.1007/978-3-642-27848-8_265-2
- Nissen, A. (2009). Classification and cost analysis of switches and crossings for the Swedish railway: A case study. *Journal of Quality in Maintenance Engineering*, 15(2), 202–220. <https://doi.org/10.1108/13552510910961147>
- O'Brien, E. J., Quirke, P., Bowe, C., & Cantero, D. (2018). Determination of railway track longitudinal profile using measured inertial response of an in-service railway vehicle. *Structural Health Monitoring*, 17(6), 1425–1440. <https://doi.org/10.1177/1475921717744479>
- Offenbacher, S., Neuhold, J., Veit, P., & Landgraf, M. (2020). Analyzing major track quality indices and introducing a universally applicable tqi. *Applied Sciences (Switzerland)*, 10(23), 1–17. <https://doi.org/10.3390/app10238490>
- Paixão, A., Fortunato, E., & Calçada, R. (2016). A contribution for integrated analysis of railway track performance at transition zones and other discontinuities. *Construction and Building Materials*, 111, 699–709. <https://doi.org/10.1016/j.conbuildmat.2016.02.126>
- Practice, P. S. (2020). *Using analytics to get European rail maintenance on track. June*.

- Pradhan, S., & Samantaray, A. K. (2018). Integrated modeling and simulation of vehicle and human multi-body dynamics for comfort assessment in railway vehicles. *Journal of Mechanical Science and Technology*, 32(1), 109–119. <https://doi.org/10.1007/s12206-017-1212-z>
- Reddy, V. (2004). *Modelling and Analysis of Rail Grinding & Lubrication Strategies for Controlling Rolling Contact Fatigue (RCF) and Rail* (Issue May).
- Sadeghi, J. (2010). Development of railway track geometry indexes based on statistical distribution of geometry data. *Journal of Transportation Engineering*, 136(8), 693–700. [https://doi.org/10.1061/\(ASCE\)0733-947X\(2010\)136:8\(693\)](https://doi.org/10.1061/(ASCE)0733-947X(2010)136:8(693))
- Soleimanmeigouni, I., Ahmadi, A., Nissen, A., & Xiao, X. (2020). Prediction of railway track geometry defects: a case study. *Structure and Infrastructure Engineering*, 16(7), 987–1001. <https://doi.org/10.1080/15732479.2019.1679193>
- Sun, Y. Q., Cole, C., & Spiriyagin, M. (2013). Study on track dynamic forces due to rail short-wavelength dip defects using rail vehicle-track dynamics simulations. *Journal of Mechanical Science and Technology*, 27(3), 629–640. <https://doi.org/10.1007/s12206-013-0117-8>
- Tacıroğlu, M. V., Karaşahin, M., Tığdemir, M., & Işiker, H. (2020). Fractal analysis of high speed rail geometry data: A case study of Ankara-Eskişehir high speed rail. *Measurement: Journal of the International Measurement Confederation*, 165. <https://doi.org/10.1016/j.measurement.2020.108120>
- Wang, C., Deng, C., & Wang, S. (2020). Imbalance-XGBoost: leveraging weighted and focal losses for binary label-imbalanced classification with XGBoost. *Pattern Recognition Letters*, 136, 190–197. <https://doi.org/10.1016/j.patrec.2020.05.035>
- Wang, C. W., Lee, Y. C., Calista, E., Zhou, F., Zhu, H., Suzuki, R., Komura, D., Ishikawa, S., & Cheng, S. P. (2018). A benchmark for comparing precision medicine methods in thyroid cancer diagnosis using tissue microarrays. *Bioinformatics*, 34(10), 1767–1773. <https://doi.org/10.1093/bioinformatics/btx838>
- Wang, H, Berkers, J., & V, F. N. L. L. B. (2019). *Absolute and Relative Track Geometry : Closing the Gap*. December, 1–13. <https://doi.org/10.25084/raileng.2019.0053>
- Wang, Haoyu, Chang, L., & Markine, V. (2018). Structural health monitoring of railway transition zones using satellite radar data. *Sensors (Switzerland)*, 18(2), 8–11. <https://doi.org/10.3390/s18020413>
- Yu, P. C., Li, J., & Wang, L. C. (2016). Research on the influence of track periodic short-wave irregularity on low-speed maglev train. *Chinese Control Conference, CCC, 2016-Augus*, 9305–9310. <https://doi.org/10.1109/ChiCC.2016.7554836>
- Zhai, W. M., Wang, K. Y., & Lin, J. H. (2004). Modelling and experiment of railway ballast vibrations. *Journal of Sound and Vibration*, 270(4–5), 673–683. [https://doi.org/10.1016/S0022-460X\(03\)00186-X](https://doi.org/10.1016/S0022-460X(03)00186-X)