Predicting voluntary employee turnover using core employee data

Sjoerd van Bekhoven, BSc

to obtain the degree of Master Of Science,
at the Delft University of Technology,
to be defended publicly on August 30, 2017 at 3:00 PM

Student number:	4014774	
Project duration:	November 14, 2017 – August 30, 2017	
Thesis committee:	Prof. dr. ir. M.J.T. Reinders,	TU Delft (chair)
	Prof. dr. M. Loog,	TU Delft (supervisor)
	Prof. dr. S. Bhulai,	VU Amsterdam
	Drs. D. Jonker,	Crunchr
	W.M. Kouw, MSc,	TU Delft

Predicting voluntary employee turnover using core employee data

Sjoerd van Bekhoven, BSc
Delft University of Technology
svanbekhoven@gmail.com
August 22, 2017

***Abstract*—Voluntary employee turnover is the process of an employee voluntarily choosing to resign from a company. High voluntary turnover has been shown to have negative effect on both organisational and financial performance of companies. Therefore, if companies were to know which individuals would leave their company in the coming months, this would open doors for many applications in human resources (HR) management. However, data directly related to voluntary employee turnover (for example: exit interviews) is scarce, while all companies have HR information systems in which they capture core employee data. This study therefore aims to research to what extent voluntary employee turnover can be predicted on an individual employee level using only core employee data. We design and extract features related to voluntary turnover using the hierarchical structure of the company such as features measuring team diversity and the hierarchical position of an employee in the company. This allows us to considerably increase performance compared to only using the original numerical features. Despite this increase, we feel performance needs another boost to make it usable in practice. We therefore investigate the underlying problems, showing that the root problem is not likely to be the inherent class imbalance or a lack of sample size, but rather a severe class overlap, meaning that the extracted feature sets are likely not informative enough to separate the voluntary leavers from the other employees.**

***Index Terms*—voluntary employee turnover, people analytics, binary classification, feature design, class imbalance, class overlap, feature importance**

I. Introduction

Employee turnover is formally described as the rotation of workers around the labor market; between companies, jobs, and occupations; and between the states of employment and unemployment [1]. In practice, human resources (HR) managers refer to employee turnover when talking about the number of employees that leave a company, either voluntary or involuntary. Voluntary employee turnover, the focus of this paper, is the process of an employee voluntarily choosing to resign from a company. High voluntary employee turnover is a widely studied phenomenon and has been shown to incur significant cost to companies [2, 3]. There is little doubt about high voluntary turnover rates having a negative effect on both organisational and financial performance of companies [4].

Numerous statistical social studies on indicators for voluntary turnover can be found. Most of these studies focus on finding single or a combination of several factors which are informative for the prediction of voluntary turnover rates. Companies use these findings to for example design employee retention strategies in order to retain their employees more effectively. However, to the best of our knowledge there are no recent studies using (combinations of) these factors to actually predict voluntary turnover on an individual employee level in large companies. Being able to exactly pinpoint which employees have a high probability of leaving a company voluntarily in the future opens doors for creating more specific retention strategies, more accurate workforce planning, designing performance metrics for managers and many more applications in HR management.

However, data directly related to voluntary turnover is often scarce, since common reasons for voluntary turnover are factors such as poor and ineffective leadership, a lack of growth, job mismatch, feeling unappreciated, low job satisfaction, internal pay inequity and many other factors [5]. To directly measure these factors, one would need information about how people behave and feel, which is often not available. The data that *is* available to nearly every company is core employee data with attributes such as the employee's age, location, position and salary. Problematic about this data is that the majority of attributes is textual with a fairly high number of unique values per attribute compared to the number of employees. This kind of data is generally hard to use for fitting statistical models, because it is hard to make proper generalizations. Also, since the number of employees voluntarily leaving a company is relatively small (several percentages per month), combining datasets of multiple companies might be preferable, but is very hard with textual values varying over companies.

On the other hand, since nearly every company possesses this kind of data, it would be of great value if predicting voluntary turnover would be feasible only using core employee data. To research the feasibility, in this paper we research the possibilities using data of a single company. We extract voluntary turnover related features from the core employee data, exploiting the hierarchical structure of companies. These features are solely numerical which makes them usable in many statistical models and for combining datasets of multiple companies. Using these features, we perform a comparison to identify the most promising groups of features. From these tests, we find that there are several challenges to overcome in order to further increase performance. We hypothesize that the underlying problems keeping us from increasing performance might be the inherent class imbalance of the data, a lack of train data or a severe class overlap. These possible causes are then discussed consecutively after which we analyse and/or attempt to overcome them. Finally, we study the informativeness of individual features using two feature importance metrics to gain insight in the most important indicators for the prediction of voluntary turnover.

This paper is structured as follows. First, in section II we consider recent work related to (predicting) voluntary employee turnover. Then, in section III we discuss the problem, the dataset and the method of evaluation. Following up on that, we discuss the extraction of several voluntary turnover related groups of features and their performance in section IV and V. In section VI we discuss the possible underlying problems that keep us from further improving performance. The importance of individual features is discussed in section VII after which we end this paper with a discussion and conclusion in section VIII.

II. Related work

Traditional research related to voluntary employee turnover is mostly focused on negative job attitudes such as low levels of job satisfaction and organisational commitment [6, 7], low performance [8] and absenteeism [9] as the main causes of leaving. However, in a model based on decision paths, it has been shown that many people who leave are often relatively satisfied, do not search for other jobs before leaving and leave because of a sudden (external) event, instead of the commonly believed effects of negative job attitudes [10, 11]. Following up on that, a measure called "job embeddedness" has been developed by Mitchell et al., quantifying the *links* of an employee with other employees within an organization, its *fit* within the organization in terms of skills, personal values and career goals and the *sacrifice* an employee would have to make for leaving a job in terms of loss of possible career advancements or colleagues [12]. They subsequently empirically show this measure to predict the variance for voluntary employee turnover rates to a larger extent than the traditional negative job attitude related measures. Around that time, Mitchell also called for the need of considering time in these types of research [13], which has been repeated by Lee et al. [14], since most of the performed studies do not look at trends over time in order to predict employee turnover. A relatively old study by Jackson et al. shows that team diversity had a strong effect on the turnover rates within the team [15]. More recently, Nishii et al. studied the the effect of the relation between manager and employee in combination with the diversity of a team on voluntary turnover and found that this relation has a moderating effect [16].

To the best of our knowledge, not much research has been published on actually predicting voluntary employee turnover on an individual level using machine learning methods. Speaking with fellow researchers from other companies, we must note that companies are in fact attempting to model and predict voluntary turnover, but that the results are often not published because of their market value. A study we found that is published, is the work by Hong et al., which compares the usage of a classification and regression model in predicting employee turnover [17]. They use both subjective data gathered by a questionnaire regarding job performance and organisational commitment as well as core employee data such as age, gender and race. They achieve reasonable performance, but only have a sample size of 130 employees.

Finally, comparable prediction research has been performed in other areas, such as the prediction of leaving customers, generally called customer churn prediction. Examples are customer churn prediction in the gambling industry [18] and the telecom industry [19]. Especially churn prediction in the telecom industry has received a lot of attention, of which a review of the features used in multiple studies can be found in a paper by Keramati et al. [20]. Most of the discussed papers at least adopt customer age, tenure, gender, satisfaction and the number of complaints. In these studies, various models are being used from complex models such as neural networks and random forests to rather simple linear models such as logistic regression [21, 22] which both show successful results.

III. Problem, Dataset & Methodology

A. Problem Definition

The problem faced in this study is identifying employees who are at risk of voluntarily leaving the company within the next k months. In order to do so, we use binary classification which means that for every month, the dataset will be split up in two classes: the employees who have left voluntarily and the others. More formally, let $O = [(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)]$ be a dataset consisting of tuples (x_i, y_i). Then, for n the number of samples and m the number of features, let $x_i \in \mathbb{R}^m$ denote a sample (employee) represented by m features, $X = [x_1, x_2, \ldots, x_n]^T \in \mathbb{R}^{n \times m}$ the feature matrix, $y_i \in \{-1, 1\}$ the label for a sample and $y = [y_1, y_2, \ldots, y_n]^T \in \{-1, 1\}^{n \times 1}$ the label vector. We use n_+ and n_- for the positive and negative class size. The formal definition of the binary classification problem is then to design and implement a model that is trained on some subset $O \subseteq O$ and is able to predict the label y_j of a sample x_j for a new tuple $(x_j, y_j) \notin O$. Such a (binary decision) model is generally called a classifier, terms we use interchangeably throughout this paper.

TABLE I
Final attributes (C = categorical, D = date, T = target)

Type	Attribute	Description
C	gender	Gender of employee.
D	date_of_birth	The date of birth of employee.
D	date_in_service	The date the employee started at the company.
C	title	The current position title of the employee.
C	manager_id	The id of the manager of the employee.
C	business_unit_level1	Level 1 of business unit: e.g. Clothing.
C	business_unit_level2	Level 2 of business unit: e.g. Shoes.
C	business_unit_level3	Level 3 of business unit: e.g. Nike.
C	functional_area_level1	Level 1 of functional area: e.g. IT.
C	functional_area_level2	Level 2 of functional area: e.g. Development.
C	functional_area_level3	Level 3 of functional area: e.g. Programmer.
C	location_level3	Level 3 of location: e.g. UK.
C	location_level4	Level 4 of location: e.g. London.
T	leave_type	Reason for leaving if employee left: e.g. voluntary.

B. Dataset

The dataset consists of monthly snapshots (time slices) of the data of all employees of a single company from January 2014 to March 2017 made available to us by Crunchr, a company which offers an online platform for people analytics serving companies worldwide. In every time slice, all available data of all employees of the company at that particular month is included. A large amount of the data in consecutive time slices is therefore equal. Changes in the data of a specific employee mean that a certain event has happened such as for example a change in position, location, etc. New hires will appear as new rows in the data and

employees that have left the company will disappear. The dataset consists of approximately 9.500 distinct employees and 5.500 employees in the last month.

The data suffers from missing values which is why we have discarded all attributes with more than 15% missing values. We have imputed the values of the remaining attributes. For example, gender has been imputed using the first name of the employee and dates have been imputed using multiple imputation. Further details can be found in Appendix A. The remaining attributes consist of basic employee information (gender, date of birth, date in service) and information about the current position of the employee (title, manager, business unit, functional area and location). The types, names and a description of all attributes can be found in Table I. The dataset can be split up in 38 separate datasets according to their time slice. We define such a dataset as D^t with t the number of the time slice.

C. Labeling

For binary classification, the employees have to be divided into two classes using a label as described in section III-A. For the composition of the label vector y at time slice t, the employees (samples) are labeled as whether they voluntarily leave the company in the next k time slices or not. In practice, this means that if the leave type for sample i is "voluntary" in time slices $\{t, t+1, \ldots, t+k-1\}$, we set $y_i = 1$, otherwise $y_i = -1$. This variable label vector enables us to set up experiments for different goals such as predicting short-term and long-term turnover. In this study, we focus on long-term turnover which is defined as an employee leaving in the next 6 months. The label vector based on dataset D^t encoding employees voluntarily leaving in the next k time slices is denoted as $y^{t,k}$ in which $y^{t,1}$ thus only encodes employees voluntarily leaving in time slice t. We distinguish between the *positive* and the *negative* class, in which the positive class consists of all samples that are positively labeled (employees who leave voluntarily) and the negative class consists of all negatively labeled samples (employees who do not leave voluntarily).

It must be noted that only for the last 12 time slices the dataset contains a distinction between voluntary and involuntary turnover, so these are the only time slices we can train and test on. However, the earlier time slices can be used to gather information about an employee in the past and are thus not useless. An overview of the size of each time slice and the ratio between the positive and negative class for labels containing information about 1 to 6 months in the future can be found in Table II. From this table one can immediately see that the classes are imbalanced, since the number of employees voluntarily leaving a company is considerably lower than the number of employees who do not.

D. Evaluation Metric

To assess the performance of a classifier, a performance metric needs to be selected. For this use case, we are specifically looking for a performance metric that represents how well the positive class (the employees who are leaving voluntarily) is predicted, since this is the class we are most interested in. To define such a performance metric, we introduce the confusion matrix in Table III which counts the number of samples that have been predicted as positive or negative per class. True positives (TP) are samples correctly labeled as positives. False positives (FP) refer to negative samples incorrectly labeled as positive. True negatives (TN) correspond to negatives correctly labeled as negative and false negatives (FN) refer to positive samples incorrectly labeled as negative.

TABLE II
RELATIVE SIZE OF THE POSITIVE CLASS FOR SEVERAL TIME SLICES, LABELS DIFFERING FROM $k = 1$ TO $k = 6$

t	n	n_+/n (relative size positive class)					
		$y^{t,1}$	$y^{t,2}$	$y^{t,3}$	$y^{t,4}$	$y^{t,5}$	$y^{t,6}$
27	5239	1.8%	4.4%	5.5%	5.7%	5.7%	7.1%
28	5322	2.5%	3.7%	3.9%	4.0%	5.4%	7.0%
29	5206	1.2%	1.5%	1.6%	3.1%	4.7%	6.5%
30	5226	0.3%	0.5%	2.0%	3.8%	5.5%	6.0%
31	5246	0.2%	1.8%	3.6%	5.4%	5.9%	7.2%
32	5211	1.6%	3.4%	5.4%	5.9%	7.3%	8.6%
33	5244	1.8%	3.9%	4.5%	6.0%	7.4%	9.4%
34	5313	2.1%	2.8%	4.3%	5.8%	7.8%	
35	5218	0.7%	2.3%	3.8%	5.9%		
36	5302	1.5%	3.1%	5.2%			
37	5293	1.6%	3.8%				
38	5338	2.2%					

TABLE III
CONFUSION MATRIX FOR A TWO-CLASS PROBLEM

	Positive prediction	**Negative prediction**
Positive class	True Positives (TP)	False Negatives (FN)
Negative class	False Positives (FP)	True Negatives (TN)

An often used and intuitive metric used for binary classification problems is accuracy, which is defined as:

$$accuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{1}$$

However, since we are working with an imbalanced dataset, accuracy will not give a good reflection of the recognition of the positive class. Imagine a situation in which all employees are predicted negatively, which will achieve high accuracy, but does not recognize any samples of the positive class. If all employees are predicted positively, low accuracy will be the result, but with high recognition of the positive class. The optimal value is somewhere in between, but it is not clear what it should be, making accuracy a poor metric for this use case.

A frequently used performance visualization that denotes the balance between the positive and negative class is the receiver operating characteristic (ROC) curve. The ROC curve is a graph of the TP_{rate} (y-axis) compared to the FP_{rate} (x-axis), showing the relation between correctly classifying the positive class against misclassifying the negative class as can be seen in Figure 1. The definitions of these rates are:

- $TP_{rate} = \frac{TP}{TP+FN}$: the percentage of positive samples correctly classified
- $FP_{rate} = \frac{FP}{FP+TN}$: the percentage of negative samples misclassified

For classifiers that output a probability for a sample to be in the positive class, one can plot multiple combinations of TP_{rate} and FP_{rate} by varying the decision boundary. Then, the area under the curve can be computed, which is

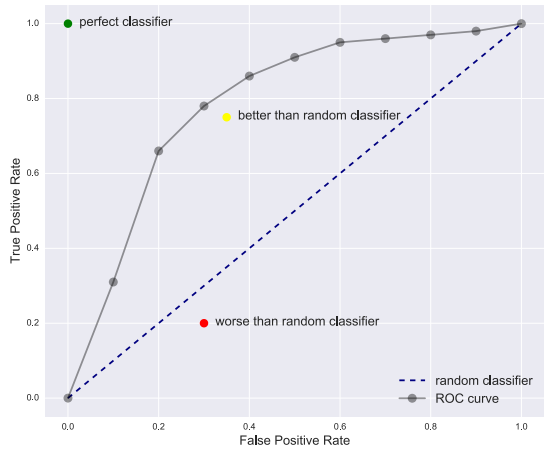

Fig. 1. Example of an ROC curve

an often used performance metric known as the area under the ROC curve ($AROC$). The $AROC$ is not affected by the class imbalance in its scoring [23], since it is based on the percentage of (in)correctly classified samples per class which makes it more suitable for usage in the imbalanced domain. However, although the overall performance of a classifier can be compared using the $AROC$ metric, the problem in the imbalanced domain is that it takes into account TN in the false positive rate. In the case of an imbalanced dataset, the true negatives (TN) will be very large compared to the false positives (FP). This causes that a considerable increase in false positives (FP) relative to the total number of positives is not reflected well in the $AROC$ metric. More formally, a high $AROC$ score will not guarantee a high performance in terms of precision and recall of the positive class [24]. The formal definitions of precision and recall are:

- precision = $\frac{TP}{TP+FP}$: the percentage of correctly predicted positive predictions
- recall = $\frac{TP}{TP+FN}$: the percentage of positive samples correctly classified

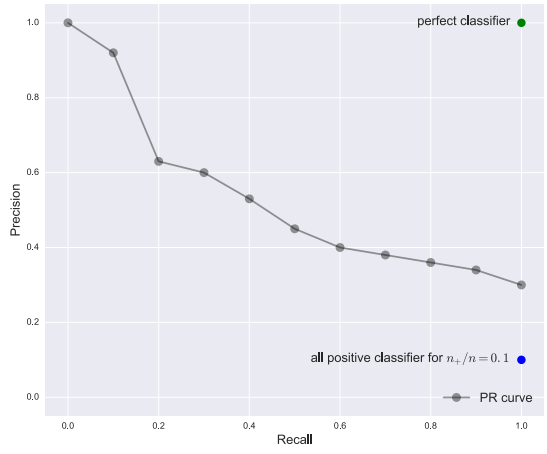

Fig. 2. Example of a PR curve (positive class of 10%)

Note that recall is the exact same thing as the true positive rate. A visual metric that does not take the large number of true negatives (TN) into account and reflects the recognition of the positive class better is the precision-recall (PR) curve as can be seen in Figure 2. This curve is a graph of the precision (y-axis) against the recall (x-axis) of the positive class. Just as for the ROC curve, the area under the PR-curve ($APRC$) can be calculated as a performance metric for precision and recall. However, although the PR-curve reflects the performance on predicting the positive class better, it must be noted that it can not be used to compare classifiers over datasets with varying class imbalances, since its final score is influenced by the distribution of the classes. Imagine comparing the performance of a classifier on two datasets which have a relative size of the positive class of 10% and 20% respectively. If this classifier scores an average precision over all recall levels ($APRC$) of 20% on both datasets, this obviously is a much better result for the first dataset than for the second, but this is not reflected since both scores are equal.

Finally, in this use case it is also interesting to investigate the $PR@k$ metric, which measures the precision at a recall level of $k\%$. This metric gives an indication of the performance of a classifier on recognizing a small part of the positive class (e.g. 10%). In this study we generally report on the $APRC$ metric, since these give us a good indication of the average performance in terms of precision on the positive class. At times, we report on the $PR@k$ metric to find if a classifier or set of features performs well on recognizing only a smaller part of the positive class. When comparing performance of classifiers on datasets with varying class balances, the $AROC$ score will be used to measure performance since it is class distribution invariant.

IV. Feature Design

There are several challenges related to the acquired dataset. First, common reasons for voluntary turnover mentioned in literature are factors such as poor and ineffective leadership, a lack of growth, job mismatch, feeling unappreciated, low job satisfaction, internal pay inequity and many other factors [5]. These factors are generally hard to measure or quantify directly only using core employee data with attributes such as age, location, position and functional area. However, it is possible to extract features that might measure these factors in an indirect way. For example:

- Internal pay inequity might relate to the average pay in an employee's team compared to its own salary.
- Having conflicts with a manager might relate to a certain similarity between the manager and the employee.
- Feeling unappreciated or perceptions of unfair treatment might relate to the diversity of the employee's team.
- A lack of growth might relate to the number of months ago it was since an employee increased a level within the company.

These are all features that are extractable from core employee data. By building a complete tree of all employees from the CEO downwards using the manager id, we are able to extract features related to the hierarchical structure, including features using information about the team and manager of each employee.

The second challenge lies within the fact that most attributes are of categorical nature, often containing a high

TABLE IV
FEATURE GROUPS

Feature Group	Feature Count
Original (OR)	3
Hierarchical (HI)	3
Team (TE)	5
Diversity (DI)	24
Minority (MI)	12
Manager Similarity (MS)	14
Temporal (TP)	28
Total	**89**

number of unique values and the need for support of combining datasets of multiple companies in the future. The problem here is that, when for example looking at business units, one company may call their IT department "IT department" while another may call it "Information Technology". These will be inferred as two different departments, which will influence performance. Therefore, there is a need to extract features that focus on a higher, meta-level of these categorical attributes. One could think of for example measuring the relative frequency of a business unit at a certain level within the team of an employee instead of adopting the actual business unit itself as a feature.

To cope with these challenges, we have extracted features potentially related to voluntary turnover of which the details can be found in Appendix B. We inspired these features on the knowledge gained by studying related work and by discussion with local experts at Crunchr. As mentioned, the categorical attributes (except for gender) are not adopted as features to allow for combining datasets of multiple companies. Therefore, all features are solely numerical, which means that for combining datasets of multiple companies, one could simply extract the features for each company separately and combine the datasets afterwards without problems of non-compliant textual values for the categorical attributes.

A. Feature groups

The features that have been extracted can be divided into 7 feature groups:

a) Original features (OR): Features such as the age of the employee, the gender of the employee and the years in service of the employee.

b) Hierarchical features (HI): The layer of the employee in the hierarchy and the number of people directly and indirectly reporting to the employee in the hierarchy to measure where employees are located in the hierarchy.

c) Team features (TE): Team averages such as the average age, average years in service and average direct span of control of the team of the employee.

d) Diversity features (DI): The number of distinct values for an attribute within the team, such as the number of unique position titles. Also, features that describe how evenly the values of categorical attributes are distributed over the team to measure diversity in the teams, such as the distribution of genders over the team. For example, a team with 5 male and 5 female employees is distributed evenly, while a team with 1 male and 9 female employees is not.

e) Minority features (MI): The relative frequency of an employee's value for an attribute within the team. For example a minority feature for gender with value 0.2 would indicate that only 20% of the team of the employee has the same gender of the employee with this value (i.e. the employee is male and 2 out of 10 employees in its team are male).

f) Manager similarity features (MS): Features to denote similarity between the employee and its manager, such as for example a feature indicating whether they have equal gender, equal functional area or features measuring the difference in age.

g) Temporal features (TP): Features for the number of changes in the last k time slices and the differences compared to k time slices ago for $k = 6$ and $k = 12$. For example, the number of changes in manager or the number of changes in position title during the last 6 months or the difference in layer in the company's structure compared to 12 months ago.

TABLE V
FEATURE SET CONSTRUCTION BY COMBINING FEATURE GROUPS

	FS																			
FG	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**
ID	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
HI			x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
TE					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
DI							x	x					x	x	x	x			x	x
MI									x	x			x	x			x	x	x	x
MS											x	x			x	x	x	x	x	x
TP		x		x		x		x		x		x		x		x		x		x
m	3	31	6	34	11	39	35	63	23	51	25	53	47	75	49	77	37	65	61	89

B. Feature sets

A total of 89 features has been extracted of which a break-down can be found in Table IV. By combining these 7 feature groups in different ways we are able to construct 128 different feature sets. For ease of reporting, we narrow this down to 20 feature sets. Since we quickly discovered that the original, team and hierarchical features form a good basis for a feature set, we first combine these three and then combine them with the minority, manager similarity and diversity features in several ways, giving us a total of 10 combinations. As we are interested in the effects of the temporal features, since these have not been used in voluntary turnover related research before, we combined all these combinations with temporal features, thus ending up with 20 combinations which can be found in Table V. To make a distinction between the actual dataset and these combinations of features, we call these feature sets (FSs).

V. FEATURE SET COMPARISON

In order to get an idea of the general performance of the extracted features, we conduct a comparison of the constructed feature sets.

A. Experimental Setup

The experiment will involve testing the performance of the feature sets using a basic linear classifier, logistic regression, for which we use the implementation of the scikit package [25]. We choose a simple linear classifier for three reasons. First, since this is a first experimental study on this subject on this kind of data, it is preferable to be able to interpret the results of the classifier to evaluate our work. Also, with an eye on the future, it is preferable for the business to be able to interpret what the actual strong indicators

of voluntary turnover are (which we will do in section VII) instead of only having the outcome of the classifier. Such an interpretation quickly becomes hard when using more complex classifiers. Secondly, complex classifiers require more samples in order not to suffer from overfitting. Given the fact that the number of positive samples is very small (a couple of hundred) and we have extracted feature sets with, relative to the number of (positive) samples, a large number of features, we argue that a complex classifier for now is out of scope. Also, preliminary experiments with support vector classifiers, nearest neighbors classifiers, decision trees and random forests (including hyper parameter tuning) showed no considerable performance improvements compared to using simple classifiers. Finally, comparable studies in customer churn prediction have used logistic regression with good results [21, 22].

In this experiment we focus on predicting long-term turnover, meaning we predict six months ahead. Since we only have 12 months of data of which we explicitly know which employees left voluntarily, this means that we train on the first six months and test on the last six months. We choose this setup because it gives as a reasonable relative and absolute number of employees in the train and test set. Therefore, we use time slice 27 with label $y^{27,6}$ for training, meaning we cover employees leaving at time slices 27 to 32. The train set consists of 5239 samples of which 372 ($\sim$ 7.1%) are positive. Testing is done using time slice 33 with label $y^{33,6}$, meaning we cover employees leaving at time slices 33 to 38. The test set consists of 5260 samples of which 501 ($\sim$ 9.5%) are positive.

Both the train set and the test set are standardized (zero-mean and unit-variance) using only information of the train set. For validation, the test set is split into 20 folds after shuffling with a random seed in order to make sure the split of employees is the same for every test. As readily explained in section III-D, we report on the mean $APRC$ and $PR@10$ and the standard error over the folds. In order to get an intuition of how the feature sets differ in performance, we perform a statistical comparison using a paired two-sided student t-test at a 95% confidence interval. It must be noted that one should be cautious with the interpretation of the results of this comparison, since we violate the independence assumption of the student t-test by the overlap of training data. Studies have shown that tests of this form are likely biased [26], but since we are merely interested in how these feature sets compare to each other and not necessarily in their statistical significance, we find it appropriate to adopt this test for this use case. However, the tests deployed in this paper should therefore be viewed as approximate, heuristic tests instead of rigorously correct statistical tests.

B. Results

The classifier is trained on the train set and tested on the 20 folds. We report on the mean and standard error of the $APRC$ and $PR@10$ over 20 folds in Figure 3 and 4 respectively. All feature sets are compared to a baseline, the feature set consisting only of original features (FS0) which is indicated with a blue error bar. We choose this as a baseline since this set consists of features that all companies have at their disposal already. The green error bars indicate a statistically significant increase compared to the baseline, whereas a red error bar indicates a statistically significant decrease.

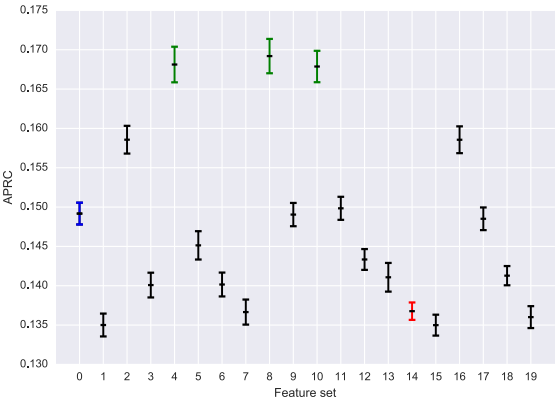

Fig. 3. Mean and standard error of $APRC$ for all feature sets

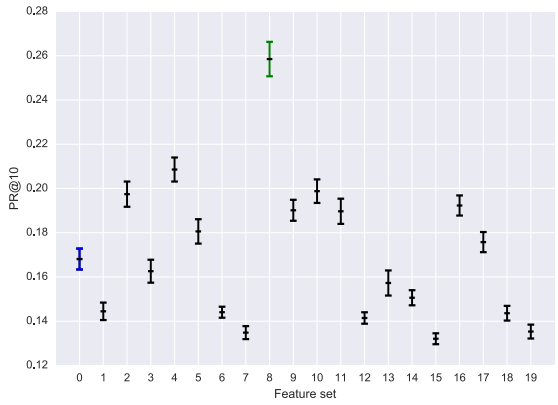

Fig. 4. Mean and standard error of $PR@10$ for all feature sets

As can be seen in the figures, in terms of the area under the PR-curve ($APRC$), only three feature sets score significantly higher than the baseline. Using original, hierarchical, team features (FS4) and combining these with minority features (FS8) or manager similarity features (FS10) results in the highest performance in terms of $APRC$ and these all have reasonably low p-values as can be seen in Table VI. It is also clear that adding only diversity to the original, hierarchical and team features (FS6), decreases performance albeit not statistically significantly. Furthermore, all feature sets containing temporal and/or diversity features generally perform worse than feature sets without these features. It must be noted that this might be caused by the large increase in dimensionality of these datasets when adding temporal or diversity features.

Since the $APRC$ metric can be interpreted as the mean precision over all recall levels, we find that with FS8 and a standard linear classifier tuning, we are able to achieve a mean precision of $\sim$ 17%. This is an increase of $\sim$ 7.5 percent point compared to predicting all samples as being positive, which would result in a precision of $\sim$ 9.5%.

In terms of the precision at 10% recall, we find the most significant increase for FS8, which contains the original, hierarchical, team and minority features and results in a

precision of $\sim$ 26%. This is an improvement of over $\sim$ 9 percent point compared to the baseline. Comparing to predicting all samples as being positive, resulting in a precision of $\sim$ 9.5%, an increase of $\sim$ 16.5 percent point is observed.

As already mentioned a pairwise comparison of the *APRC* metrics for all feature sets can be found in Table VI, reporting on the p-values of the statistical tests. The green dots denote a statistically significant increase in *APRC* of FSI compared to FSII, whereas a red dot denotes a statistically significant decrease. The feature sets that are never statistically significantly outperformed by other feature sets are feature FS2, FS4, FS8, FS10, FS11 and FS16.

To interpret these results, imagine a case in which an HR manager would like to set up a retention policy. Using the best performing model (FS8 with PR@10, precision $\approx$ 25%, recall $\approx$ 10%), the manager could identify a subgroup of employees which includes 10% of all leaving employees. Given that there are 5200 in total, 500 employees leaving in the next six months (as is the case in the test set) and that on average 25% will actually leave the company within 6 months, such a subgroup contains $500 \times 10\% = 50$ leaving employees and a total of $\frac{50}{25\%} = 200$ employees. The manager could now pinpoint its retention strategy on this subgroup of 200 employees, targeting 10% of the total number of leavers. If a manager would want to target 50 employees without any model, he would do no better than $\frac{500}{5200} \approx 9.5\%$ precision and would therefore need to target $\frac{50}{9.5\%} \approx 526$ employees instead of only 200.

Of course, such a specific use case is useful, but also limited. In general, being able to predict whether an employee is going to leave a company with an average precision of about $\sim$ 17% (FS8 on *APRC*) instead of 9.5% is hardly usable in practice. Therefore, we seek to find the underlying problems that hinder us in further improving performance in the next section.

VI. Exploring possible problems

We discuss three possible causes which stop us from increasing performance further. For these problems: class imbalance, class overlap and lack of sample size, we analyse their origin and if possible aim to overcome them with techniques from related literature.

A. Class imbalance

Class imbalance is a common problem in binary classification which can be defined as a significant difference between the prior probabilities of the positive and the negative class. The smaller of the two, in our case the positive class, is often called the minority class, while the larger class is called the majority class. The minority class is in many cases the class that is most important to classify correctly (or most costly to classify incorrectly) while standard classifiers are often biased towards the majority class.

It must be noted however that the class imbalance on itself it not necessarily the main or only problem. If the majority and minority class are perfectly linearly separable, most standard classifiers will be able to achieve high accuracy on both classes [27, 28]. There may be other problems which are either caused or amplified by the class imbalance, causing classifiers to perform poorly.

In order to check whether the class imbalance is one of the root causes of the lack of performance, we discuss several techniques representative for the state-of-the-art techniques and apply these to our problem.

1) Techniques: There generally are three categories of techniques: pre-processing the data in order to balance the class distribution before classification (sampling), tuning classifiers to be less affected by the class imbalance (algorithmic modification) and combining both strategies resulting in bagging and boosting techniques. Recently, sampling and combining both techniques have gotten the most attention which is why we focus on these.

Sampling is a preprocessing step which aims to balance the classes in order to achieve higher performance either by oversampling the minority class, undersampling the majority class or both. The most simple sampling techniques are random oversampling and random undersampling, which randomly duplicate samples from the minority class and randomly remove samples from the majority class respectively. A popular technique used for oversampling is called Synthetic Minority Oversampling Technique (SMOTE) [29] which oversamples the minority class by creating synthetic samples by randomized interpolation over a number of neighboring samples of samples in the minority class. One of its limitations is that it samples uniformly over all minority class samples and thus might increase overlap between classes causing overgeneralization [30]. To overcome this problem, techniques such as Adaptive Synthetic Sampling (ADASYN) [31] have been introduced which use heuristics to select specific minority class samples for oversampling. Undersampling techniques are often based on data cleaning techniques, which tend to remove overlap between the two classes in order to end up with better defined clusters of classes. Such techniques are for example based on removing the majority class sample of a pair connected by a Tomek link [32], which connects a pair of minimally distanced nearest neighbors of opposite classes. Another technique is the edited nearest neighbors (ENN) rule [33] which iteratively removes samples of the majority class that differ in class label to the majority of a fixed number of nearest neighbors. Oversampling and undersampling techniques are also combined, resulting in techniques such as SMOTE+Tomek and SMOTE+ENN [34]. These techniques first apply SMOTE for oversampling of the minority class and then remove samples from the majority class with ENN or Tomek links in order to create better-defined areas in the data space and prevent overgeneralization.

Finally, we look into ensembles using undersampling techniques such as EasyEnsemble and BalanceCascade [35]. EasyEnsemble creates an ensemble of classifiers trained on k randomly undersampled datasets. This means that the minority class is equal in all subsets, but that the majority class is randomly undersampled to the size of the minority class. For BalanceCascade, again an ensemble of k classifiers is made, but now the models are trained consecutively. First, the base dataset gets undersampled and the first classifier is trained. All majority class samples that are predicted correctly by the trained classifier are removed from the base dataset and again, the base dataset is undersampled and the procedure is repeated k times or until the majority class in the base dataset is smaller than the minority class. We study one boosting technique: Ad-

TABLE VI
P-VALUES OF PAIRWISE COMPARISON OF THE *APRC* METRIC FOR ALL FSS

	FSII																			
FSI	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
0	1.00	0.09	0.07	0.29	0.03	0.66	0.17	0.13	0.03	0.99	0.04	0.94	0.36	0.32	0.03	0.06	0.24	0.93	0.18	0.06
1	0.09	1.00	0.02	0.00	0.00	0.00	0.58	0.75	0.01	0.01	0.01	0.00	0.33	0.39	0.82	1.00	0.03	0.02	0.44	0.88
2	0.07	0.02	1.00	0.06	0.18	0.19	0.03	0.01	0.18	0.24	0.22	0.32	0.05	0.04	0.00	0.00	1.00	0.16	0.02	0.01
3	0.29	0.00	0.06	1.00	0.01	0.04	0.99	0.52	0.03	0.12	0.02	0.03	0.71	0.90	0.68	0.39	0.08	0.15	0.89	0.55
4	0.03	0.00	0.18	0.01	1.00	0.03	0.01	0.00	0.88	0.03	0.98	0.07	0.01	0.01	0.01	0.00	0.24	0.02	0.01	0.00
5	0.66	0.00	0.19	0.04	0.03	1.00	0.62	0.12	0.07	0.48	0.07	0.36	0.85	0.59	0.37	0.13	0.22	0.59	0.68	0.21
6	0.17	0.58	0.03	0.99	0.01	0.62	1.00	0.62	0.01	0.31	0.01	0.30	0.49	0.91	0.53	0.46	0.05	0.37	0.83	0.53
7	0.13	0.75	0.01	0.52	0.00	0.12	0.62	1.00	0.01	0.02	0.01	0.05	0.31	0.34	0.99	0.71	0.02	0.07	0.50	0.89
8	0.03	0.01	0.18	0.03	0.88	0.07	0.01	0.01	1.00	0.03	0.88	0.10	0.00	0.01	0.00	0.01	0.10	0.03	0.00	0.00
9	0.99	0.01	0.24	0.12	0.03	0.48	0.31	0.02	0.03	1.00	0.05	0.88	0.36	0.10	0.13	0.02	0.18	0.87	0.27	0.01
10	0.04	0.01	0.22	0.02	0.98	0.07	0.01	0.01	0.88	0.05	1.00	0.06	0.02	0.01	0.00	0.00	0.11	0.02	0.01	0.00
11	0.94	0.00	0.32	0.03	0.07	0.36	0.30	0.05	0.10	0.88	0.06	1.00	0.44	0.26	0.07	0.00	0.30	0.74	0.26	0.03
12	0.36	0.33	0.05	0.71	0.01	0.85	0.49	0.31	0.00	0.36	0.02	0.44	1.00	0.71	0.24	0.22	0.05	0.48	0.52	0.14
13	0.32	0.39	0.04	0.90	0.01	0.59	0.91	0.34	0.01	0.10	0.01	0.26	0.71	1.00	0.59	0.33	0.03	0.24	0.98	0.20
14	0.03	0.82	0.00	0.68	0.01	0.37	0.53	0.99	0.00	0.13	0.00	0.07	0.24	0.59	1.00	0.72	0.01	0.11	0.24	0.88
15	0.06	1.00	0.00	0.39	0.00	0.13	0.46	0.71	0.01	0.02	0.00	0.00	0.22	0.33	0.72	1.00	0.01	0.01	0.29	0.77
16	0.24	0.03	1.00	0.08	0.24	0.22	0.05	0.02	0.10	0.18	0.11	0.30	0.05	0.03	0.01	0.01	1.00	0.11	0.02	0.00
17	0.93	0.02	0.16	0.15	0.02	0.59	0.37	0.07	0.03	0.87	0.02	0.74	0.48	0.24	0.11	0.01	0.11	1.00	0.30	0.01
18	0.18	0.44	0.02	0.89	0.01	0.68	0.83	0.50	0.00	0.27	0.01	0.26	0.52	0.98	0.24	0.29	0.02	0.30	1.00	0.26
19	0.06	0.88	0.01	0.55	0.00	0.21	0.53	0.89	0.00	0.01	0.00	0.03	0.14	0.20	0.88	0.77	0.00	0.01	0.26	1.00

aBoost [36]. AdaBoost builds an ensemble of k classifiers by training them consecutively on all samples. For each iteration, it increases the sample weight of the samples that are incorrectly classified and decreases the sample weight of the samples that are correctly classified in the previous iteration in order to increase performance. Furthermore, a weight is given to each classifier in the ensemble according to its performance on the train set. All ensembles decide upon a label for a new sample by a (weighted) majority vote.

2) Experiments: For the experiments, we adopt the 12 described techniques all using logistic regression as their base classifier. We use a simple adjustment of the weights of the samples according to the relative class sizes during optimization, a technique we call "Weighting". Furthermore, we use all techniques described above, using implementations of both imblearn [37] and the sci-kit package [38].

We argue that the techniques selected are representative for the set of algorithms that are used in order to cope with class imbalances in recent literature. Recent papers that have compared basic (basic sampling) and more advanced (ensembles and boosting) techniques to using just a basic classifier show that at least the EasyEnsemble and BalanceCascade used in this paper perform comparably to most of the promising techniques [39]. For number of internal classifiers used in the ensembles we used $k = 10$ since this is the number that is used in most papers comparing ensemble techniques such as [28] and [39].

The same experimental setup as was used for the base experiments is used except for the fact that we now only used the four most promising feature sets according to the comparison in the previous section, namely FS4, FS8, FS10 and FS16. The mean *APRC* scores over the 20 folds of these experiments can be found in Table VII. None of the tested techniques performs statistically significantly better than the baseline, denoted by the technique "None" in the table. A small improvement can be found for some techniques, especially for AdaBoost, but overall we may conclude that using class imbalance techniques does not result in a considerable performance improvement.

From these results we conclude that the inherent class imbalance of this dataset is not the root cause stopping us from increasing performance. However, if one manages to find and solve the root problem it is likely to be valuable to repeat these experiments in order to increase performance further since a severe class imbalance could be amplifying other, more minor problems. In order to identify the root cause, we explore two other possible causes, namely a possible class overlap and a possible lack of train samples.

TABLE VII
MEAN *APRC* FOR SEVERAL IMBALANCE RELATED TECHNIQUES

	FS			
Technique	**4**	**8**	**10**	**16**
None	0.1681	0.1692	0.1679	0.1586
Weighting	0.1668	0.1633	0.1617	0.1550
Sampling				
Random oversampling	0.1692	0.1580	0.1578	0.1582
SMOTE oversampling	0.1694	0.1632	0.1678	0.1552
ADASYN oversampling	0.1692	0.1618	0.1590	0.1537
Random undersampling	0.1647	0.1524	0.1609	0.1462
Tomek links undersampling	0.1676	0.1690	0.1684	0.1581
ENN undersampling	0.1690	0.1683	0.1635	0.1557
SMOTE+ENN	0.1694	0.1646	0.1688	0.1560
SMOTE+Tomek	0.1670	0.1623	0.1630	0.1610
Bagging				
BalanceCascade	0.1674	0.1541	0.1547	0.1566
EasyEnsemble	0.1667	0.1678	0.1651	0.1606
Boosting				
AdaBoost	0.1707	0.1771	0.1631	0.1663

B. Class overlap

The problem of class overlap appears when there are areas in the data space in which the boundaries of classes overlap. When trying to fit any classifier on these areas, it is hard not to say impossible to decide upon a suitable decision boundary, since one will either misclassify the one class or the other. The effects of a class imbalance and class overlap have been studied separately and combined [40, 41]. These studies show that for artificial datasets in which both the imbalance and the overlap is controlled, class overlap generally has a greater effect on the deterioration of the performance.

Class overlap is often measured in terms of separability. Separability measures have been investigated over the years, but to the best of our knowledge there is no generally accepted measure that is widely used. An intuitive measure is Thornton's separability index (SI) [42]. Although this index is known to have some disadvantages [43], it gives a

TABLE VIII
SI FOR VARYING CLASS OVERLAP (FIGURE 5)

Class overlap	SI	$SI(+1)$	$SI(-1)$
~ 0%	0.999	0.987	1.000
~ 20%	0.975	0.806	0.987
~ 40%	0.950	0.637	0.972
~ 60%	0.931	0.444	0.966
~ 80%	0.902	0.272	0.947
~ 100%	0.882	0.105	0.937

good intuition of the separability of two classes. The SI is defined as follows:

$$SI(f) = \frac{\sum_{i=1}^{n} f(x_i) + f(x_i^{'}) + 1 \mod 2}{n} \quad (2)$$

in which f is a binary target function, x_i a sample and $x_i^{'}$ the nearest neighbor of x_i using the euclidean distance measure. Basically, this means that if every sample is closest to a neighbor which has the same label, $SI(f) = 1$ and if every sample is closest to a neighbor with another label, $SI(f) = 0$. Since our datasets suffer from a class imbalance, we have attempted to adapt this measure to see how recognizable the positive class is by deriving a label-dependent version. Therefore, for some label y we define:

$$SI(f, y) = \frac{\sum_{i \in S(y)} f(x_i) + f(x_i^{'}) + 1 \mod 2}{|S(y)|} \quad (3)$$

in which $S(y)$ denotes the indices of the samples with label y and $|S(y)|$ the size of class y. Again a $SI(f, +1)$ close to 1 would mean that all positively labeled samples have closest neighbor with the same label. We use $SI(f) = SI$, $SI(f, +1) = SI(+1)$ and $SI(f, -1) = SI(-1)$ for ease of notation.

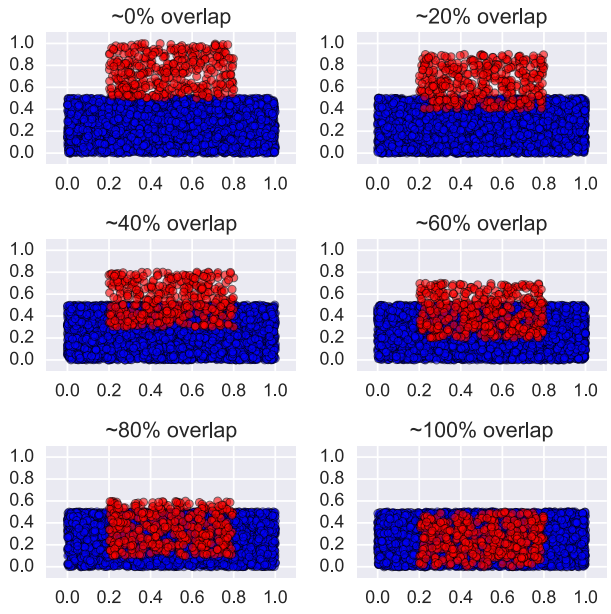

Fig. 5. Examples of datasets with varying class overlap

In Figure 5 one can find 6 artificial datasets with 5239 samples of which are 372 (~ 7.1%) are positive (the red class) with varying class overlap. The samples are drawn from a 2-dimensional uniform distribution. This is the same class imbalance as can be found in the train set used throughout this paper. The separability indexes of these datasets can be found in Table VIII. In the top-left graph, the classes are nearly fully separable which results in an SI close to 1 for both the positive and negative class. From top-left to bottom-right we increase the overlap with 20% each time and see a moderate decrease of the SI and $SI(-1)$ and a steep decrease for the $SI(+1)$.

In Figure 6 the separability indexes on our dataset for all feature sets can be found. Interestingly, we generally find increased scores (especially for the positive class) compared to the baseline for most of the feature sets that scored well in our previous tests (e.g. FS4, FS8 and FS10). This indicates that indeed the introduction of new features seems to increase the separability of the classes. Since the $SI(+1)$ lies between 6% and 14% for all feature sets, many of the samples in the positive class seem to lie very close to samples of the negative class, which makes separating these rather hard. On the other hand, if 14% of the positive samples does have a closest neighbor of the positive class (as is the case for FS17), this indicates that at least not all samples of the positive class are spread throughout the space, but that there seems to be some structure.

Overall, we find that all feature sets have separability indexes that point towards a severe class overlap. The newly extracted features indeed do seem to increase the separability of the classes, but only to some extent. It is likely that this class overlap is one of the root causes for our inability to increase performance further.

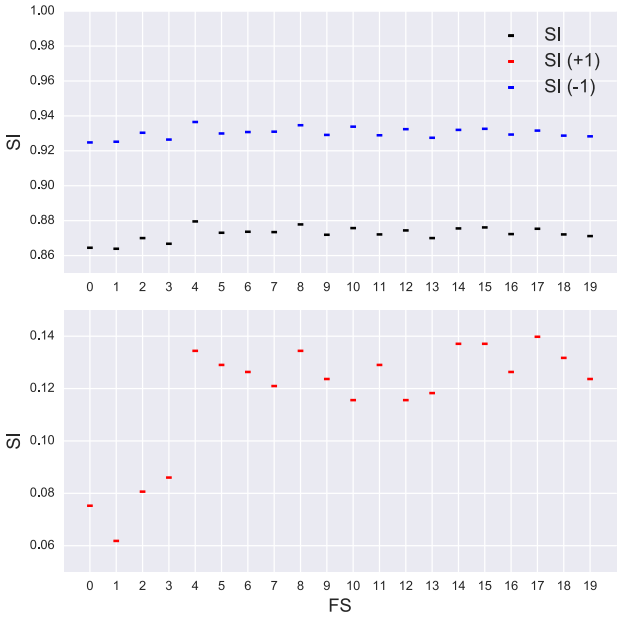

Fig. 6. Separability index (SI) for all feature sets (FSs)

C. Lack of sample size

An often encountered issue in learning problems is a lack of sufficient data. Especially when combined with a severe class imbalance, the positive class can become very small in absolute sense. Therefore, making a proper generalization of the positive class becomes even harder since it is more difficult to distinguish between outliers and regular samples. This may cause these classifiers to treat the samples in

the minority class as noise or may lead to overfitting on the low number of available samples, both of which are undesirable. In order to see if a lack of sample size is one of the problems, it is useful to see how the performance of the classifier progresses for validating on the train and test set when training on differently sized subsets of the train data. If the performance on the test set steeply increases when reaching the full train set while it has not yet reached the level of performance of the training set, this would indicate that there is room for improvement upon gathering more data for training.

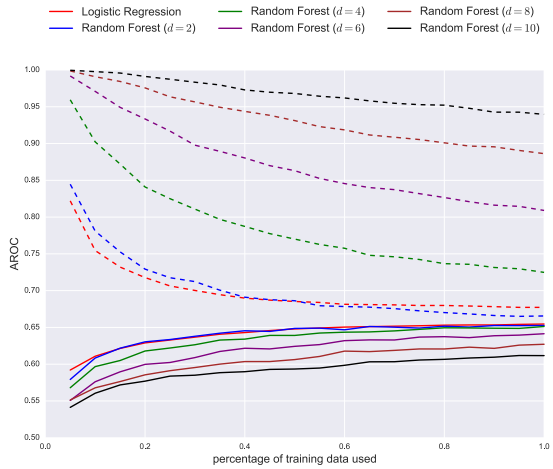

Fig. 7. Learning curve for several classifiers on FS8 (d = maximum depth of trees in random forest, solid line = test score, dotted line = train score)

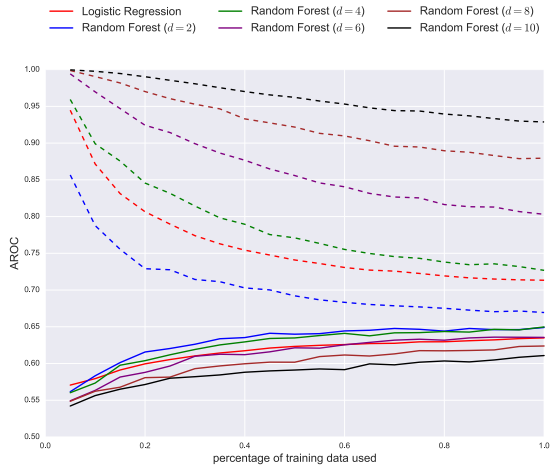

Fig. 8. Learning curve for several classifiers on FS17 (d = maximum depth of trees in random forest, solid line = test score, dotted line = train score)

1) Experiments: To obtain the learning curves we again train on time slice 27, test on time slice 33 and use labels for employees leaving within 6 months. As for features, we use the most promising feature set from the comparison, FS8, which has a relatively low dimensionality (m = 23) and the feature set which showed promising results in the class overlap experiments, FS17, which has a relatively high dimensionality (m = 65). The training dataset is randomly split in 20 subsets while preserving the class balance in each subset. First, training is done on one subset after which another subset is added for each iteration until training on the full dataset. Since we compare the results on validating on the train and test set, which have different class balances, the *APRC* metric is not suitable, because it is not class distribution invariant, as is mentioned in section III. Therefore, we use the class distribution invariant metric *AROC*. This experiment is repeated 50 times after which the results are averaged.

The results of this analysis can be found in Figure 7 and Figure 8 for FS8 and FS17 respectively. In these graphs one can find the results for classifiers of varying complexity, but we first focus on the results of the logistic regression classifier (the red line) which we have used for previous experiments.

a) Logistic regression: In Figure 7 we see a typical learning curve. With very few samples, the classifier overfits on the data and thus the train score starts high and decreases once more samples are added. The opposite happens for the test score and towards 100% of the training data, the train and test score flatten out and near each other with the test score staying somewhat under the train score. This means there is some bias towards the train set, but the fit seems rather good and therefore adding more data of the same distribution is not expected to increase performance.

When looking at Figure 8, the results on the higher dimensional feature set, we see similar behavior, although now the train and test score decrease and increase more gradually. Also, the gap between the test and train score when training on the full dataset is larger and the test curve still has a slight upward trend when nearing 100% of the train set. However, the curves already start to flatten out and the gap likely will not be closed when adding more data, meaning that overfitting occurs to a larger extent than when using FS8. Even if we would for example tune the regularization parameter of the logistic regression classifier to overcome overfitting, adding more data is only expected to improve performance moderately.

The classifier used throughout this paper, logistic regression, is a relatively simple linear classifier which might explain why adding more data will likely not increase performance. Therefore, we attempt to use a more complex classifier, the random forest classifier, to see if this results in a more promising learning curve.

b) Random forest: We use a random forest classifier [44] which builds an ensemble of decision trees and decides on the label of a sample by averaging over the outputted probabilities of the decision trees in the ensemble. By varying the maximum depth of the trees in the ensemble, we can control the complexity of the decision boundary of the trees and therewith the complexity of the decision boundary of the random forest. As can be seen in the figures, the scores on the test and train set change accordingly. For complex models, the model overfits on the training data which results in a high train score and a relatively low test score. When simplifying the model by reducing the maximum depth, the model generalizes more and thus the train score decreases and the test score increases.

In Figure 7, the results on FS8, the random forests with maximum depth d = 10, d = 8 and d = 6 are greatly overfitted when nearing 100% of the train data, resulting in a lower test score then the logistic regression classifier. The classifiers with maximum depth d = 2 shows similar behav-

ior as the logistic regression classifier. The other classifiers show a steeper curve for the test score and a train score that is not yet flatted out, meaning that acquiring more samples might be beneficial to some extent. In Figure 8, similar conclusions can be drawn, but interestingly the random forests with maximum depth $d = 4$ and $d = 2$ outperform the logistic regression classifier in terms of $AROC$.

Overall, when using a simple linear classifier and the features designed in this paper, the fit of the model seems rather good and adding more data will not likely result in performance improvements. When using a more complex model, test curves are a little steeper when nearing training on the full train set while the train and test scores are not yet nearing each other and thus adding more samples can be expected to improve performance moderately. However, one must keep in mind that acquiring more samples often simply is not possible, since a company only has a certain number of employees. A possibility could be to add samples from multiple companies together, but since these will be distributed differently, the effects are uncertain. Finally, it must be noted that although the more complex random forest classifiers outperform the logistic regression classifier in some cases in terms of $AROC$, the average precision ($APRC$) falls short in all cases.

Having researched some of the possible underlying problems keeping us from improving performance further, we may conclude that the root causes are not likely the inherent class imbalance or a lack of sample size, but rather a severe class overlap. Therefore, since it seems like a logical step for future research to look into feature selection or collecting more informative features and because it is interesting from a business perspective, we study the informativeness of the individual features used in this study in the next section.

VII. Feature Importance

From the comparison in section V we have learned that the minority and manager similarity features combined with original, team and hierarchical features perform best in terms of $APRC$. Finally, we have seen from section VI-B that FS17, the feature set combining all these well-performing feature groups combined, resulted in the best performance in terms of the separability index for the positive class. This section aims to give an intuition about the informativeness of the extracted features as a handle for future research to work with. We do so by excluding the diversity features since they generally have shown to only decrease classification performance. Therefore, we use FS17 in the experiments in this chapter.

To study the informativeness of the features, we take a look at the importance of the features with two metrics using the logistic regression classifier. First, the weights corresponding to the features which are computed during optimization of the logistic regression classifier can be interpreted as importance metrics for the features. Note that for this to work, the features have to be standardized. The larger the absolute value of the weight, the more it contributes to the output. As a metric for importance, we therefore take the squared weight (since weights can be negative) and call this metric the weight importance (WI):

$$WI(f) = (w_f)^2$$

in which w_f is the weight of feature f computed while optimizing the logistic regression classifier.

Secondly, we study the permutation feature importance, which is derived from an importance measure used for Random Forests [45]. The measure is computed by first training the model on the train data and validating it on the test data for a baseline score. Then, the values of feature f in the test set are permuted, after which the trained model is run on the permuted test set to get a performance score. Consecutively, this permutation and testing routine is repeated k times after which the scores are averaged in order to get an overall score for this feature. If the feature is informative, the average performance of the model is expected to drop after permutation while if the feature is not informative, the effect should be negligible or the performance might even increase. We define the permutation feature importance (PFI) as follows:

$$PFI(f) = s_{\text{base}} - \frac{1}{k}\sum_{i=1}^{k} s_{f,i}$$

in which s_{base} is the performance on the non-permuted test set and $s_{f,i}$ the performance on the test set with f permuted in iteration i. For our experiments, we repeated the permutation $k = 500$ times and used the $AROC$ metric (average precision) to measure performance.

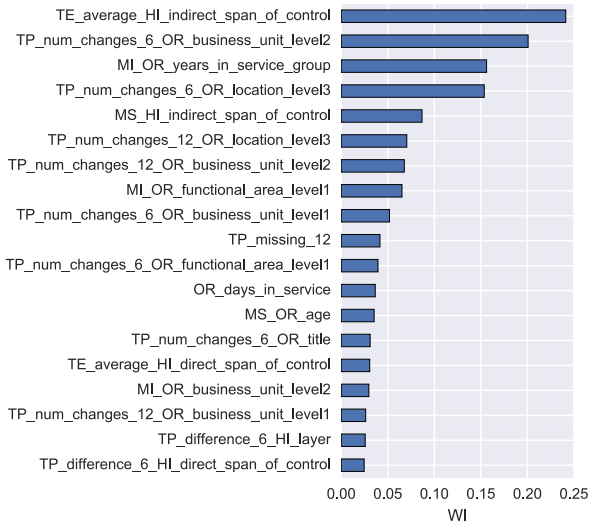

Fig. 9. Top 20 features by weight importance (WI)

Interesting about comparing these two feature importance metrics is that WI uses only the fit on train set, while PFI is the result of a validation on the test set. Therefore, WI gives an intuition about the importance of the features as a result from optimizing the classifier, while PFI gives an intuition of the actual effect on the classification performance. The top 20 (of a total of 65) of the features ranked according to these two importance measures can be found in Figure 9 and Figure 10.

An interesting find is that while the team (TE) average based on the indirect span of control has the highest score for WI, it does not have the largest impact on actual performance as we can see at PFI. In fact, its importance is considerably lower than the highest ranking feature for PFI. The highest ranking feature for PFI is the minority (MI) feature based on the years in service, for which the

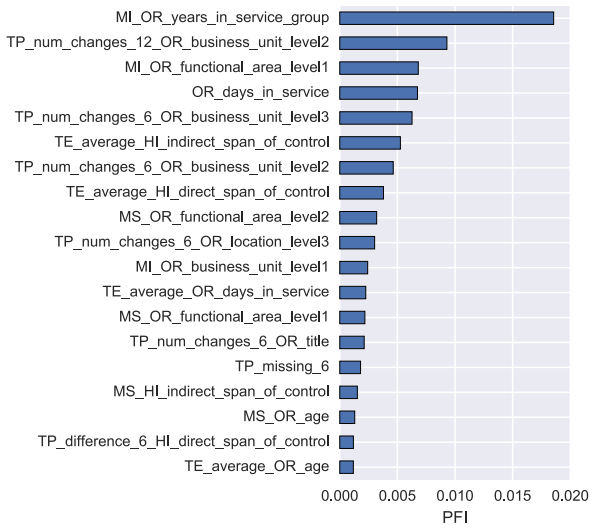

Fig. 10. Top 20 features by permutation feature importance (PFI)

average precision of the model drops by almost 2 percent point compared to the baseline. All other features show considerably lower results. We also find that the original (OR) feature for the days in service, which is the basis for the minority feature derived from this feature, also ranks considerably higher in PFI than for WI, indicating that these features are both important for the actual prediction.

The minority features based on functional area and the time an employee has been in service of a company are high in the top 20 for both metrics. Also, the features containing information about the team averages over the spans of control within the teams of the employees score well for both metrics. High ranks can be found on both metrics for temporal (TE) features describing the number of changes in business unit and location. Finally, for the manager similarity (MS) features we find that the features extracted from the age of the employee and the indirect span of control are in the top 20 of both rankings, but their impact seems negligible. The only high ranked manager similarity feature in terms of WI is the one based on the indirect span of control, ranked fifth, but this is not present in the top 20 for PFI. Also, the highest ranked manager similarity feature in terms of PFI describes the functional area at the second level, which is not present in the top 20 of WI.

Overall, we think it is fair to conclude that the team averages over the spans of control, minority features related to the functional area and the time in service, and temporal features related to the changes in business unit and location seem to be among the strongest predictors for voluntary employee turnover. In general, the time in service seems to be the attribute of the employee that contains most information. The manager similarity features are generally ranked low and show inconsistent results in terms of PFI and WI, making them less reliable.

VIII. Discussion, limitations & conclusion

In this section we go over the results from section V to VII and expose the limitations of our work while giving recommendations for future work. Finally, we end with a summary of the main conclusions.

A. Discussion

In this study we have shown that the hierarchical, team, minority and manager similarity features we extracted based on core employee data increase performance compared to only using the original features. The feature sets including minority features also showed a significant increase in performance on only recognizing a small part of the employees that are leaving within six months. The usage of diversity related features only decreased performance compared to using the original features. Furthermore, feature importance experiments have led us to the conclusion that the team averages based on spans of control, minority features based on functional area and the time in service and the number of changes in business unit and location are among the most promising features. Especially the attribute describing the time employees have been in service seems to be a strong basis for informative features.

Interestingly, we found that team averages related to the spans of control of the team members seem to have a large impact on performance while we have not seen features alike in work researching associations between variables and voluntary employee turnover. Furthermore, while diversity metrics measuring the diversity within a team as a whole using diversity indexes have been shown to relate to voluntary turnover rates [15], we found diversity features to be among the worst performing features, even decreasing classification performance. On the other hand, the minority features, measured in terms of a relative frequency compared to the rest of the team, that are also closely related to team diversity have proven to be among the best predictors. A possible explanation might be that the minority features actually describe how an employee is affected by the diversity in the team, instead of describing the diversity of a team as a whole as is done in the diversity features. Again, we have not found any studies using features like the minority features to describe diversity, while these seem very informative. Finally, we have attempted to incorporate a time element in the prediction of voluntary turnover as was encouraged by Mitchell et al. [13] and Lee et al. [14] and found that the number of changes in business unit and location over time indeed resulted in promising features.

Although we have shown that there are use cases for which the achieved performance can come in useful, we concluded that the performance achieved by our experiments would generally not satisfy practical requirements. In this study we have shown that the underlying problem stopping us from increasing performance is not likely to be the inherent class imbalance of the dataset and that adding more data of the same distribution is only expected to improve performance moderately. At the same time, separability index experiments point towards the presence of a severe class overlap which is likely to be the root problem.

Assuming that the classes indeed severely overlap, as we suggest, this means that the combinations of extracted features we have tested are not informative enough to predict voluntary employee turnover. Of course, it is possible to perform other experiments with more complex classifiers, but we do not feel this is the direction to go to. First, we think that looking further into capturing temporal patterns is an important direction when working with core employee data. Mainly because the environment, position and location

of employees are quite static and do not change every day, week or month, it is difficult to understand why the same employee stays in one time slice and leaves in another one while represented by (nearly) the same values. By looking at long-term trends, one could capture patterns about their progress through the company. We have attempted to do so by extracting temporal features, but have only touched the surface. It is therefore that we would recommend to focus on for example survival models to compute the expected time until an event happens as has been done in customer churn prediction [46] or hidden markov models which are often used for temporal pattern recognition. Secondly, if one would want to stick to objective data sources such as core employee data, we think a good direction would be to experiment with other voluntary turnover related features based on objective data. For example, the job embeddedness model by Mitchell et al. [47] pointed towards measuring the links of employees within the organisation, which could for example be done by building a network based on e-mail or calendar data.

It is noteworthy that simple models have been successful in customer churn prediction [21, 22], especially in the telecom industry, while for this use case it seems more difficult to achieve such results. One of the reasons could be that while telecom customers simply cancel their contract and switch companies if they are dissatisfied, employees *do* have something to lose: their jobs. This makes their reasons for leaving more complicated and diverse, such that even after more than hundred years of research towards reasons for voluntary employee turnover, social researchers still do not have a unambiguous answer to this problem, let alone being able to actually measure it. Also, we find that in contrary to customer churn prediction, companies do not often publish their work about voluntary employee turnover prediction, while we know that a lot of research is performed in this area.

We think that the prediction of voluntary employee turnover on an individual level using only objective data sources such as core employee data is a promising area. One of the great advantages of using objective data over subjective data, is that it can be collected easily and regularly. Subjective data sources such as for example a survey are generally hard to collect for every employee and can not be collected frequently. Furthermore, objective data is not influenced by company politics or human discretion, making it more reliable. Also, there are a lot of different objective data sources to be explored and many other approaches which can be tried. However, predicting irrational human behavior using only objective measurements is and will continue to be a hard task. Therefore, one should not expect to achieve near perfect performance and should not be withheld when only increasing performance with small steps at a time.

B. Research limitations

A possible limitation of our work is that some of the extracted features are correlated since they are based on hierarchical attributes such as the functional area, business unit and location. These attributes consist of several levels, which means that a change in a value on a higher level likely results in a change on a lower level as well. Therefore, all meta-features in this study based on the same hierarchical attribute can be correlated to some extent. This might have influenced the results since we have used a logistic regression classifier, which assumes no correlation between the features. However, as can be seen in the section about the lack of training samples, using random forests, which are known to be able to handle correlated features rather well, does not result in considerable performance improvements.

As for the quality of the statistical comparison of the feature sets, it has already been mentioned that, since the training sets of the performance scores that have been tested by the two-sided t-test overlap, it is probable that the test scores give an overly positive view. Furthermore, the performance scores used in the statistical tests are acquired by using a default logistic regression classifier with no tuning of for example the regularization parameter. This might explain why the performance of the feature sets with high dimensionality is generally lower than the performance of low dimensional feature sets. Especially since it has also been found in the separability experiments that some of these high dimensional feature sets in fact do increase the separability index this might have given a distorted view of reality.

A limitation of the experimental setup used throughout this paper is that it consists of only two distinct sets of for training and testing of only one company. Therefore, the results might be biased towards these specific time slices of this specific company and thus one must be careful to generalize the presented results. Once more time slices become available, all experiments could be repeated by adopting a sliding window over the time slices (train on 28, test on 33, train on 29, test on 34, etc.) and averaging over the results. This would allow for better generalization possibilities of the results. Also, in this study we have focused on the prediction of long-term voluntary turnover. It would be interesting to research short-term turnover, meaning one would predict only three months ahead instead of six. Although this decreases the size of the positive class, this might also have a positive effect on the homogeneity of the positive class.

There is one issue that has not been taken into consideration at all during this study, which is choosing a threshold (the trade-off between the true positive rate and false positive rate) for the classifiers used. Although *AROC* and *APRC* scores give a good overall view of the performance of a classifier, using these instead of actual rates basically avoids the choice of a threshold. Besides making the actual trade-off, another difficulty that might come up when having to make this decision is the fact that the priors of the train and test set (often) differ in this use case. At least, the priors of the future are not known up front since one does not know how many employees are going to leave a company in the future. This prior shift is a well-known problem in literature and will offer yet another challenge.

Finally, besides studying a possible lack of training samples, class overlap and class imbalance, there are multiple other possible causes that would be worthwhile investigating. Lopez et al. for example mention, besides the three possible causes that have been studied in this paper, small disjuncts within the positive class, a dataset shift and noisy data [28]. Several experiments have been performed on especially showing small disjuncts in the positive class, but

these have not been fruitful.

C. Conclusion

In this study, we have designed and extracted several groups of features based on reasons for voluntary turnover. These groups of features have been compared and we have shown that features containing information about team averages (team features), features describing the relative diversity of an employee compared to its team (minority features) and features describing the similarity of an employee with its manager (manager similarity features) resulted in the largest performance increase. The addition of these features allows for a considerable increase in performance compared to using only the original numerical attributes. Especially when only focusing on recognizing a small part of the voluntarily leaving employees, we are able to achieve a substantial increase when using the minority features. However, although we have shown that the achieved performance can be used for specific use cases, performance of this magnitude is hardly usable in practice. Therefore, we have investigated the underlying problems keeping us from improving performance. We have shown that this is not likely to be caused by the inherent class imbalance of the data or a lack of sample size, but rather due to a severe class overlap, meaning that the extracted feature sets are likely not informative enough to separate the voluntary leavers from the other employees. Finally, to give an idea of the features that are of importance, we performed a short study on the informativeness of the extracted features. We found that the minority features based on functional area and the time in service, the team averages over the spans of control and the features describing the number of changes in business units and location over time are among the strongest indicators of voluntary turnover.

References

[1] Simon Burgess. Analyzing firms, jobs, and turnover. *Monthly Labor Review*, 121(7):55–57, 1998.

[2] Kevin M Morrell, John Loan-Clarke, and Adrian J Wilkinson. Organisational change and employee turnover. *Personnel Review*, 33(2):161–173, 2004.

[3] Brooks C Holtom, Terence R Mitchell, Thomas W Lee, and Marion B Eberly. 5 turnover and retention research: a glance at the past, a closer review of the present, and a venture into the future. *The Academy of Management Annals*, 2(1):231–274, 2008.

[4] Arie C Glebbeek and Erik H Bax. Is high employee turnover really harmful? an empirical test using company records. *Academy of Management Journal*, 47(2):277–286, 2004.

[5] Human Resource Management. *Open Textbooks for Hong Kong*, pages 192–193, 2016.

[6] Michael Jenkins and R Paul Thomlinson. Organisational commitment and job satisfaction as predictors of employee turnover intentions. *Management Research News*, 15(10):18–22, 1992.

[7] Scott D Camp. Assessing the effects of organizational commitment and job satisfaction on turnover: An event history approach. *The Prison Journal*, 74(3):279–305, 1994.

[8] Charles R Williams and Linda Parrack Livingstone. Another look at the relationship between performance and voluntary turnover. *Academy of Management Journal*, 37(2):269–298, 1994.

[9] Paula C Morrow, James C Mcelroy, Kathleen S Laczniak, and James B Fenton. Using Absenteeism and Performance to Predict Employee Turnover: Early Detection through Company Records. *Journal of Vocational Behavior*, 55:358–374, 1999.

[10] Thomas W Lee and Terence R Mitchell. An Alternative Approach: The Unfolding Model of Voluntary Employee Turnover. *Source: The Academy of Management Review*, 19(1):51–89, 1994.

[11] Thomas W Lee, Terence R Mitchell, Brooks C Holtom, Linda S Mcdaniel, and John W Hill. The Unfolding Model of Voluntary Turnover: A Replication and Extension. *Source: The Academy of Management Journal*, 42(4):450–462, 1999.

[12] Terence R Mitchell, Brooks C Holtom, Thomas W Lee, Chris J Sablynski, and Miriam Erez. Why people stay: Using job embeddedness to predict voluntary turnover. *Academy of management journal*, 44(6):1102–1121, 2001.

[13] Terence R Mitchell and Lawrence R James. Building better theory: Time and the specification of when things happen. *Academy of Management Review*, 26(4):530–547, 2001.

[14] Terence R Mitchell, Tyler C Burch, and Thomas W Lee. The need to consider time, level, and trends: A turnover perspective. *Journal of Organizational Behavior*, 35(2):296–300, 2014.

[15] Susan E Jackson, Joan F Brett, Valerie I Sessa, Dawn M Cooper, Johan A Julin, and Karl Peyronnin. Some differences make a difference: Individual dissimilarity and group heterogeneity as correlates of recruitment, promotions, and turnover. *Journal of applied psychology*, 76(5):675, 1991.

[16] Lisa H Nishii and David M Mayer. Do Inclusive Leaders Help to Reduce Turnover in Diverse Groups? The Moderating Role of LeaderMember Exchange in the Diversity to Turnover Relationship. *Journal of Applied Psychology*, 94(6), 2009.

[17] Wei-Chiang Hong and Ruey-Ming Chao. A comparative test of two employee turnover prediction models. *International Journalnt of Manageme*, 24(2), 2007.

[18] Kristof Coussement and Koen W De Bock. Customer churn prediction in the online gambling industry: The beneficial effect of ensemble learning. 2013.

[19] Ning Lu, Hua Lin, Jie Lu, and Guangquan Zhang. A customer churn prediction model in telecom industry using boosting. *IEEE Transactions on Industrial Informatics*, 10(2):1659–1665, 2014.

[20] A Keramati, R Jafari-Marandi, M Aliannejadi, I Ahmadian, M Mozaffari, and U Abbasi. Improved churn prediction in telecommunication industry using data mining techniques. *Applied Soft Computing*, 24:994–1012, 2014.

[21] Jonathan Burez and Dirk Van den Poel. Handling class imbalance in customer churn prediction. *Expert Systems with Applications*, 36(3):4626–4636, 2009.

[22] Marcin Owczarczuk. Churn models for prepaid cus-

tomers in the cellular telecommunication industry using large data marts. *Expert Systems with Applications*, 37(6):4710–4712, 2010.

[23] Laszlo a. Jeni, Jeffrey F. Cohn, and Fernando De La Torre. Facing Imbalanced Data–Recommendations for the Use of Performance Metrics. *2013 Humaine Association Conference on Affective Computing and Intelligent Interaction*, pages 245–251, 2013.

[24] Jesse Davis and Mark Goadrich. The Relationship Between Precision-Recall and ROC Curves. *Proceedings of the 23rd International Conference on Machine learning – ICML'06*, pages 233–240, 2006.

[25] Logistic Regression classifier by scikit-learn. `goo.gl/RaV9ns`.

[26] Thomas G Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural computation*, 10(7):1895–1923, 1998.

[27] Yanmin Sun, Andrew K C Wong, and Mohamed S Kamel. Classification of imbalanced data: a review. *International Journal of Pattern Recognition and Artificial Intelligence*, 23(4):687–719, 2009.

[28] Victoria López, Alberto Fernández, Salvador García, Vasile Palade, and Francisco Herrera. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information Sciences*, 250(January):113–141, 2013.

[29] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[30] BX Wang and Nathalie Japkowicz. Imbalanced data set learning with synthetic samples. In *Proc. IRIS Machine Learning Workshop*, volume 19, 2004.

[31] Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, pages 1322–1328. IEEE, 2008.

[32] Ivan Tomek. Two modifications of cnn. *IEEE Trans. Systems, Man and Cybernetics*, 6:769–772, 1976.

[33] Dennis L Wilson. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man, and Cybernetics*, 2(3):408–421, 1972.

[34] Gustavo EAPA Batista, Ronaldo C Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *ACM Sigkdd Explorations Newsletter*, 6(1):20–29, 2004.

[35] Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou. Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(2):539–550, 2009.

[36] Robert E Schapire. A brief introduction to boosting. In *Ijcai*, volume 99, pages 1401–1406, 1999.

[37] Imblearn. `https://goo.gl/UCe3jh`.

[38] Scikit-learn. `https://goo.gl/bHFKg5`.

[39] Mikel Galar, Alberto Fernández, Edurne Barrenechea, Humberto Bustince, and Francisco Herrera. A Review on Ensembles for the Class Imbalance Problem: Bagging-, Boosting-, and Hybrid-Based Approaches. *APPLICATIONS AND REVIEWS*, 42(4), 2012.

[40] Ronaldo C Prati, Gustavo E A P A Batista, and Maria C Monard. Class imbalances versus class overlapping : an analysis of a learning system behavior.

[41] Vicente García, Roberto Alejo, José Salvador Sánchez, José Martínez Sotoca, and Ramón Alberto Mollineda. Combined effects of class imbalance and class overlap on instance-based classification. In *International Conference on Intelligent Data Engineering and Automated Learning*, pages 371–378. Springer, 2006.

[42] Chris Thornton. Separability is a learners best friend. In *4th Neural Computation and Psychology Workshop, London, 9–11 April 1997*, pages 40–46. Springer, 1998.

[43] Linda Mthembu and Tshilidzi Marwala. A note on the separability index. 2008.

[44] Random Forest classifier by scikit-learn. `https://goo.gl/V2PKAT`.

[45] Silke Janitza, Carolin Strobl, and Anne-Laure Boulesteix. An auc-based permutation variable importance measure for random forests. *BMC bioinformatics*, 14(1):119, 2013.

[46] Zainab Jamal and Randolph E Bucklin. Improving the diagnosis and prediction of customer churn: A heterogeneous hazard modeling approach. *Journal of Interactive Marketing*, 20(3-4):16–29, 2006.

[47] Terence Mitchell, Thomas Lee, Chris Sablynski, James Burton, and Brooks Holtom. The effects of job embeddedness on organizational citizenship, job performance, volitional absences and voluntary turnover. *Academy of Management Journal*, 47(5), 2004.

[48] Joseph L Schafer and John W Graham. Missing data: our view of the state of the art. *Psychological methods*, 7(2):147, 2002.

[49] Maytal Saar-Tsechansky and Foster Provost. Handling Missing Values when Applying Classification Models. *Journal of Machine Learning Research*, 8:1625–1657, 2007.

[50] A Rogier T Donders, Geert JMG van der Heijden, Theo Stijnen, and Karel GM Moons. Review: a gentle introduction to imputation of missing values. *Journal of clinical epidemiology*, 59(10):1087–1091, 2006.

[51] Melissa J Azur, Elizabeth A Stuart, Constantine Frangakis, and Philip J Leaf. Multiple imputation by chained equations: what is it and how does it work? *International journal of methods in psychiatric research*, 20(1):40–49, 2011.

[52] Genderize.io. `https://goo.gl/U4ySvx`.

[53] Fancyimpute. `https://goo.gl/cW3Nsj`.

[54] Mark O Hill. Diversity and evenness: a unifying notation and its consequences. *Ecology*, 54(2):427–432, 1973.

APPENDIX A
MISSING DATA

As can be seen in the Table IX, several attributes in the employee dataset suffer from missing values. The missing data is mainly caused by the fact that the dataset consists of multiple smaller datasets which do not all contain all attributes. For proper prediction of employees at risk of voluntary leaving it is in most cases a necessity to have complete data in which no important attributes are missing. Therefore in this section we discuss the types of missing data recognized in literature, the most common ways to handle these types of missing data and how these apply to this dataset.

TABLE IX
DESCRIPTION AND TYPE OF ATTRIBUTES OF THE EMPLOYEE DATASET

Feature	Description	Type
gender	Gender: male / female	categorical
date_of_birth	Date of birth	date
date_in_service	Date employee was first hired	date
date_in_position	Date the employee started at current position	date
fte	Number of full-time equivalents	numerical
employee_grade	The international Hay grade of the employee	numerical
position_grade	The international Hay grade of current position	numerical
title	Title of current position, e.g. Assistant Recruiter	categorical
solid_line	The id of the direct manager	categorical
dotted_line	The id of a manager from another business unit	categorical
business_unit	5-level hierarchy of the type of business within company: Pharma / Life Sciences / Outsourcing etc.	categorical
functional_area	4-level hierarchy of the type of work within company: Finance / HR / Legal etc.	categorical
location	6-level hierarchy of the location the employee works: Europe / Netherlands / Amsterdam etc.	categorical
legal_entity	4-level hierarchy of the legal entity within the company: Company Holding / Company Nederland etc.	categorical
cost_center	7-level hierarchy of the cost center within company: Corporate / Sales / Healthcare etc.	categorical
contract_type	The type of contract an employee has: permanent / part-time / full-time etc.	categorical
talent_status	Status which indicates the potential of an employee: potential / well-placed / high-potential etc.	categorical
potential	The level of potential according to direct manager: high / low etc.	categorical
mobility	The area an employee is willing to move within for work: regional / global / not mobile etc.	categorical
performance_status	The level of performance according to direct manager: high / low etc.	categorical
retention_risk	The level of risk of retention according to direct manager: high / low etc.	categorical
retention_risk_reason	The reason of risk of retention according to direct manager: compensation issiues / personal etc.	categorical
solid_line_layer	The layer in the company of the direct manager	numerical
employee_status	The status of an employee: active / inactive (maturnity) / sick leave etc.	categorical
base_salary	The base salary (without bonus etc.) of an employee	numerical
currency	The currency of the base salary	categorical
crunchr_reward_id	Reward id	categorical
hire_type	If the employee was hired this month, the type of hire: hire / rehire etc.	categorical
leave_type	If the employee left the company this month, the reason of leaving: left / voluntarily / terminated etc.	categorical
target_bonus	The bonus if a certain target is reached	numerical

A. Types of missing data

Missing data is a problem that is recognized in multiple domains. Generally, three kinds of missing data are recognized in literature [48]:

Data missing completely at random (MCAR) If entries with missing data are a random subset of the complete sample of entries, the data is considered to be MCAR. For example if the age of an employee is missing because an HR manager forgot to ask for the employee's birth date.

Data missing not at random (MNAR) If the probability of missing data is not dependent on an observed attribute, the data is considered to be MNAR. For example if an HR manager asks all employees for their salary at their previous organization it might be the case that people who had very high salaries in the past tend to not answer this particular question. This means the fact that this question is left empty is dependent on the value of this particular question.

Data missing at random (MAR) If the fact that data is missing is dependent on an observed attribute, the data is considered to be MAR. For example if one knows when data was collected and an HR manager only started collecting a certain attribute from a certain date on, it is known that the fact that the data is missing is dependent on the time the value was observed.

B. Handling missing data

Since most classification methods require a dataset without missing values, it is necessary to choose from (a combination of) the following strategies: discarding objects, discarding attributes, acquiring missing values or imputation of missing values [49]. In this section we shortly describe both the advantages and disadvantages of each method:

Acquiring missing values In many cases it is possible to obtain missing data in some way. Especially if attributes have a lot of missing data, but are of significant importance, it could be worth acquiring the data by requesting it or sending out a questionnaire. Of course, acquiring missing data comes with a certain cost and thus a cost-benefit analysis would need to be made.

Discarding objects A possibility is to simply discard the objects with missing values if the amount of objects with missing data is rather small. This is a reasonable approach when it is required to for example assess the performance of a certain model. However, when it is required to classify all objects, discarding objects is not an option. This method can be safely adopted when the data is MCAR, but will likely result in a bias when data is either MNAR or MAR.

Imputation of missing values Instead of discarding the whole object, it is tempting to predict the missing values by some kind of heuristic when only a certain amount of values for a attribute are missing. Of course, when dealing with data that is not MAR or MCAR, this could introduce a bias in the final results.

Discarding incomplete attributes If attributes are not known for a multitude of objects, one can simply discard the attributes that have too many missing values. This reduces the dimensionality of the data and may thus affect prediction quality, but can be safely adopted for any type of missing data.

Crunchr has already put a considerable amount of effort in acquiring data of high quality and data validation. Acquiring the missing values by requesting further research by the organization that has released this dataset is therefore not feasible. The other options are all reasonable and are therefore discussed consecutively in the following sections.

TABLE X
ATTRIBUTE SUMMARY FOR THE DATASET

Feature	Distinct	Completeness	Included
gender	2	85.83%	✓
date_of_birth	5314	85.79%	✓
date_in_service	2399	99.45%	✓
date_in_position	1143	99.74%	✓
fte	1	100.00%	
employee_grade	0	0.00%	
position_grade	0	0.00%	
title	1980	100.00%	✓
solid_line	2059	99.99%	✓
dotted_line	0	0.00%	
business_unit_level0	1	100.00%	
business_unit_level1	5	97.13%	✓
business_unit_level2	20	95.44%	✓
business_unit_level3	20	75.92%	
business_unit_level4	0	0.00%	
functional_area_level0	1	100.00%	
functional_area_level1	3	89.23%	✓
functional_area_level2	17	89.23%	✓
functional_area_level3	92	89.23%	✓
location_level0	1	100.00%	
location_level1	1	100.00%	
location_level2	1	100.00%	
location_level3	41	94.77%	✓
location_level4	424	94.72%	✓
location_level5	0	0.00%	
contract_type	0	0.00%	
talent_status	7	2.81%	
potential	4	2.76%	
mobility	0	0.00%	
performance_status	4	2.76%	
retention_risk	3	0.12%	
retention_risk_reason	5	0.14%	
solid_line_layer	1	0.02%	
employee_status	1	100.00%	
base_salary	0	0.00%	
currency	0	0.00%	
crunchr_reward_id	0	0.00%	
hire_type	3	5.22%	
leave_type	3	2.50%	✓
target_bonus	0	0.00%	
relative_salary_position	0	0.00%	

1) Discarding incomplete attributes: A summary of the attributes in the dataset can be found in Table X. In this table, one can also find which attributes have been discarded. First, all attributes that have less than 85% completeness or have 0 or 1 unique values have been discarded. Accepting about 15% of missing values per attribute seems like a reasonable heuristic given the fact that we can then hold on to the attributes *gender* and *date of birth*, which we expect to be important for later use. One exception has been made for the leave type, which is only populated when an employee has left in that particular month and will be necessary to form a target variable. After discarding these attributes, the final set of attributes can be found in Table XI.

2) Discarding objects: Discarding objects is definitely not preferable because of several reasons. First of all, the amount of data is rather limited so discarding incomplete objects ($\sim$ 25% of all objects) would most probably result in a decrease of performance for the final model since fewer objects are available for training or proper feature extraction. Furthermore,

TABLE XI
FINAL ATTRIBUTE SELECTION AND CORRESPONDING NUMBER OF DISTINCT VALUES, COMPLETENESS AND USED IMPUTATION METHOD (F = BY FIRST NAME, M = MULTIPLE IMPUTATION, N = NO IMPUTATION, D = DISCARDING INCOMPLETE INSTANCES, I = INDICATOR METHOD)

Feature	Distinct	Completeness	Imputation
gender	2	85.83%	F
date_of_birth	5314	85.79%	M
date_in_service	2399	99.45%	M
date_in_position	1143	99.74%	M
title	1980	100.00%	N
solid_line	2059	99.99%	D
business_unit_level1	5	97.13%	I
business_unit_level2	20	95.44%	I
business_unit_level3	20	75.92%	I
functional_area_level1	3	89.23%	I
functional_area_level2	17	89.23%	I
functional_area_level3	92	89.23%	I
location_level3	41	94.77%	I
location_level4	424	94.72%	I
leave_type	3	2.50%	N

the model would ideally generate a prediction for every employee in the dataset, which is not possible for objects which are removed from the dataset.

In the case of attributes such as solid line (the employee's manager), it is impossible to do an imputation. The main reason for this is that an imputation of an employee's manager would mean that when extracting features based on the hierarchy (e.g. team features), would not comply at all with the employee. Therefore, employees with a missing value for the solid line attribute are discarded. The number of objects discarded, 35, is surmountable.

3) Imputing missing values: Two rather simple ways of working with missing data are the missing indicator method and the mean imputation method. The missing indicator method adds an extra attribute for every attribute with missing values which is set to 1 only when the corresponding attribute is missing and sets the corresponding attribute to 0. The mean imputation method just imputes the overall sample mean at every missing value. The problem with these rather simple methods is that they, even when data is MCAR, have a tendency to cause a bias in final results.

A more sophisticated way of imputing missing data would be to use *single imputation*, which uses for example a multivariate regression model to estimate the distribution of missing values [50]. Finally, a popular and currently state-of-the art technique is a form of *multiple imputation* called MICE. Basically, this technique first imputes all missing values with the mean over the whole population. Then, it performs multiple cycles of multivariate regression over the attributes with missing values using the other attributes in a certain order until convergence. For the final estimation, the mean over the cycles is taken after discarding several burn-in cycles. MICE generally performs very well with MAR and MCAR data, but has the disadvantage that there is no theoretical justification as there is for other imputation methods. [51]

There are three types of attributes that we need to consider: hierarchical attributes, date attributes and the gender attribute. We shortly discuss these attribute types and the way the missing values have been handled.

a) Hierarchical Attributes: A large portion of the categorical attributes is hierarchical, which means it is not necessary for them all to be non-empty. For example, the CEO only works on a global level in terms of functional area and thus, except for the first level, the functional area levels will be empty. This means that it is simply not possible to estimate a correct value. Therefore, we use the indicator method, meaning in this case that we will just replace all missing values with an empty string.

b) Gender: For the missing genders, we use Genderize.io [52] which is an API that determines the gender by a first name and optionally location. Its database consists of more than 200.000 names of which the gender is extracted from social media. In total $\sim$ 15.000 objects have a missing gender with $\sim$ 1.000 distinct names. About $\sim$ 900 names are recognized by Genderize.io, which leaves $\sim$ 1.000 genders missing. These are imputed randomly according to the distribution of genders over the dataset.

c) Dates: To impute missing dates, we use a Python implementation of MICE [51] in the fancyimpute package [53]. We use multiple imputation based on six numerical attributes, namely date of birth, date in position, date in service, direct span of control, indirect span of control and layer. The dates are converted to the number of days since that particular date to acquire an integer.

For the configuration of MICE we use ridge regression for prediction, 1000 cycles, 100 burn-in samples and mean imputation as initial imputation in the first cycle. We created a test set by removing missing values in the complete dataset according to the distribution of the missing data in the incomplete dataset. When imputing with the above method, the method acquires an R^2 score of about ~ 0.92, which we consider well enough.

APPENDIX B
FEATURE EXTRACTION

For definition of the features, let $\mathcal{E}$ be the set of all employees in some dataset and in general let $E \in \mathcal{E}$ be the employee for which a particular feature will be extracted. Now, let T be the team of employee E consisting of employee objects T_i such that $E \in T$ (an employee is a member of its own team). In general, let a be an attribute (or feature) of an employee and let $E.a$ return the attribute a of employee E. Also, for any set of employee objects S, let $S.a = \{E.a : E \in S\}$, a list of the attributes of all employees in set S. Finally, for any list of attributes L, let distinct(L) return the distinct values in the list, and count(L) return a list of (v_i, n_i)-pairs in which is v_i is a distinct value and n_i is the number of occurrences of this value in the list L.

The attributes can be of categorical, numerical or date nature. The categorical attributes are: title, manager id, business unit level 1 to 3, functional area level 1 to 3 and location level 3 and 4. The date attributes are the date of birth and the date in service.

A. Original features

As a basis for the feature set, the basic attributes of the employees as have been discussed in Section III-B are used. To use these attributes as features, every type of data needs its own kind of transformation, which are discussed in this section.

Age The number of years since the date of birth:

$$\text{age}(E) = (\text{now} - E.\text{date\_of\_birth}).years$$

Years in service The number of years since the date in service:

$$\text{years\_in\_service}(E) = (\text{now} - E.\text{date\_in\_service}).years$$

Gender An indicator feature for the gender of the employee:

$$\text{gender}(E) = \begin{cases} 1 & \text{if } E.\text{gender} = \text{'F'} \\ 0 & \text{if } E.\text{gender} = \text{'M'} \end{cases}$$

1) Individual helper features: In order to also collect minority and diversity features from numerical original features, we convert the age and years in service features to an ordinal, categorical feature. Therefore, the following two features are only extracted to help in the extraction of features in the next section, but are not actually adopted as features:

Age group Ordinal feature for the age of the employee:

$$\text{age\_group}(E) = \begin{cases} 0 & \text{if } 0 \leq E.\text{age} \leq 10 \\ 1 & \text{if } 10 < E.\text{age} \leq 20 \\ 2 & \text{if } 20 < E.\text{age} \leq 30 \\ 3 & \text{if } 30 < E.\text{age} \leq 40 \\ 4 & \text{if } 40 < E.\text{age} \leq 50 \\ 5 & \text{if } 50 < E.\text{age} \leq 60 \\ 6 & \text{if } 60 < E.\text{age} \leq 70 \\ 7 & \text{otherwise} \end{cases}$$

Years in service group Ordinal feature for the number of years since the employee has been in service:

$$\text{years\_in\_service\_group}(E) = \begin{cases} 0 & \text{if } 0 \leq E.\text{years\_in\_service} \leq 5 \\ 1 & \text{if } 5 < E.\text{years\_in\_service} \leq 10 \\ 2 & \text{if } 10 < E.\text{years\_in\_service} \leq 15 \\ 3 & \text{if } 15 < E.\text{years\_in\_service} \leq 20 \\ 4 & \text{if } 20 < E.\text{years\_in\_service} \leq 25 \\ 5 & \text{if } 25 < E.\text{years\_in\_service} \leq 30 \\ 6 & \text{if } 30 < E.\text{years\_in\_service} \leq 35 \\ 6 & \text{if } 35 < E.\text{years\_in\_service} \leq 40 \\ 7 & \text{otherwise} \end{cases}$$

B. Hierarchical features

Several features are extracted from the hierarchical structure of the data. To do so, a tree is built from the CEO downwards using the manager id that is present for each employee.

Layer The layer in the organisation in which 0 is the first layer and the maximum layer the lowest layer. In this specific dataset, most time-slices have 11 layers. The CEO of the organization is always in layer 0 of the hierarchy. Let employee E.title = 'CEO' be the CEO of the organization, so the first layer.

$$\text{layer}(E) = \begin{cases} 0 & \text{if } E.\text{title} = \text{'CEO'} \\ \text{layer}(E.\text{parent}) + 1 & \text{otherwise} \end{cases}$$

Direct span of control The direct span of control is the number of employees that report directly to E:

$$\text{direct\_span\_of\_control}(E) = |E.children|$$

Indirect span of control The indirect span of control is the number of employees that report indirectly to E:

$$\text{indirect\_span\_of\_control}(E) = \begin{cases} 1 & \text{if } |E.\text{children}| = 0 \\ \sum_{F \in E.\text{children}} \text{indirect\_span\_of\_control}(F) & \text{otherwise} \end{cases}$$

C. Team features

Average The average over all employees in the team of the employee for an attribute a:

$$\text{average\_}a(T) = \frac{\sum_{i=0}^{|T|} T_i.a}{|T|}$$

This feature is extracted for the the direct span of control, indirect span of control, age and years in service.

Team size The number of team members:

$$\text{team\_size}(T) = |T|$$

D. Diversity features

In ecology and biology related literature, several indexes have been described to describe diversity within a population of species such as Simpon's index, Shannon's entropy and the total number of species [54]. The most commonly used measure is Shannon's entropy:

$$H(p_1, \ldots, p_k) = \exp\left(-\sum_{i=1}^{k} p_i \ln(p_i)\right)$$

in which k is the total number of species, p_i is the proportion of species in the population that corresponds to species i, such that $\sum_{i=1}^{k} p_i = 1$. Shannon's entropy satisfies $k - 1 \leq H(p_1, \ldots, p_k) \leq k$ and the closer the value is to k, the more evenly the values are distributed over the team. Basically, this means that Shannon's entropy combines two measures of diversity: the number of distinct values and the evenness in which the distinct values are distributed over the population. Since for this research a comparison between several teams of different sizes must be made, these measures are split by dividing Shannon's entropy by the maximum value, the number of distinct values (k), which is commonly called Shannon's equitability measure:

$$E(p_1, \ldots, p_k) = \frac{\exp\left(-\sum_{i=1}^{k} p_i \ln(p_i)\right)}{k}$$

Shannon's equitability measure scales between 0 and 1 and is thus comparable over multiple teams. Just adding the number of distinct values as a feature should be comparable among teams since later on the team size will also be added to the feature space. The model should be able to fit on these 2 values in a proper way. These features are defined as:

$$\text{evenness\_}a(T) = \frac{\exp\left(-\sum_{(v_i, n_i) \in \text{count}(T.a)} \frac{n_i}{|T|} \ln(\frac{n_i}{|T|})\right)}{|\text{distinct}(T.a)|}$$

$$\text{distinct\_}a(T) = |\text{distinct}(T.a)|$$

These features are extracted for the age group, business unit level 1 to 3, functional area level 1 to 3, gender, location level 3 and 4, title and the years in service group.

E. Minority features

For minority features, we encode the relative frequency of attribute values in the teams.

$$\text{minority\_}a(E, T) = \frac{|\{v : T.a | v = E.a\}|}{|T|}$$

This minority features is extracted for the age group, business unit level 1 to 3, functional area level 1 to 3, gender, location level 3 and 4, title and years in service group.

F. Manager similarity features

To describe the similarity of an employee compared to its manager, we encode this with the following features. Let a be the attribute for which we want to extract a similarity feature and let E be the employee and M be the employee's manager. For categorical attributes a the feature is encoded as:

$$\text{manager\_similarity\_}a(E, M) = \begin{cases} 1 & \text{if } E.a = M.a \\ 0 & \text{otherwise} \end{cases}$$

And for numerical attributes a the feature is encoded as:

$$\text{manager\_similarity\_}a(E, M) = E.a - M.a$$

This feature is extracted for the age, business unit level 1 to 3, the days in service, functional area level 1 to 3, gender, location level 3 and 4 and title.

G. Temporal features

Finally, features that capture the temporal aspect of the dataset by comparing attributes and features of employees over time. For specification of the features, define $E[t].a$ as the attribute a at time t for employee E. Let time-step t be the current time.

Missing historical data k months ago A problem is that for newly hired employees, no historical data is available. Therefore, a missing historical data feature is implemented and the features that would normally have used historical data are set to 0. This feature is extracted for both $k = 6$ and $k = 12$:

$$\text{missing\_history\_}k(E,t) = \begin{cases} 1 & \text{if } E[t-k] \text{ is missing} \\ 0 & \text{otherwise} \end{cases}$$

Difference since k months ago The difference for an attribute compared to k months ago. For numerical attributes, define:

$$\text{time\_difference\_}a\text{\_}k(E,t) = \begin{cases} 0 & \text{if } E[t] \text{ or } E[t-k] \text{ is missing} \\ E[t].a - E[t-k].a & \text{otherwise} \end{cases}$$

This feature is extracted for the direct span of control, indirect span of control and layer for both $k = 6$ and $k = 12$.

Number of changes since k months ago The number of changes of an attribute since k months ago.

This feature is extracted for business unit level 1 to 3, functional area level 1 to 3, location level 3 and 4, solid line and title for both $k = 6$ and $k = 12$.

H. Final Features

The final features can be found in Table XII.

TABLE XII
THE FINAL FEATURES AND THEIR CORRESPONDING FEATURE GROUPS (FG).

Feature	FG
DI_distinct_OR_age_group	DI
DI_distinct_OR_business_unit_level1	DI
DI_distinct_OR_business_unit_level2	DI
DI_distinct_OR_business_unit_level3	DI
DI_distinct_OR_functional_area_level1	DI
DI_distinct_OR_functional_area_level2	DI
DI_distinct_OR_functional_area_level3	DI
DI_distinct_OR_gender	DI
DI_distinct_OR_location_level3	DI
DI_distinct_OR_location_level4	DI
DI_distinct_OR_title	DI
DI_distinct_OR_years_in_service_group	DI
DI_evenness_OR_age_group	DI
DI_evenness_OR_business_unit_level1	DI
DI_evenness_OR_business_unit_level2	DI
DI_evenness_OR_business_unit_level3	DI
DI_evenness_OR_functional_area_level1	DI
DI_evenness_OR_functional_area_level2	DI
DI_evenness_OR_functional_area_level3	DI
DI_evenness_OR_gender	DI
DI_evenness_OR_location_level3	DI
DI_evenness_OR_location_level4	DI
DI_evenness_OR_title	DI
DI_evenness_OR_years_in_service_group	DI
HI_direct_span_of_control	HI
HI_indirect_span_of_control	HI
HI_layer	HI
OR_age	ID
OR_days_in_service	ID
OR_gender	ID
MI_OR_age_group	MI
MI_OR_business_unit_level1	MI
MI_OR_business_unit_level2	MI
MI_OR_business_unit_level3	MI
MI_OR_functional_area_level1	MI
MI_OR_functional_area_level2	MI
MI_OR_functional_area_level3	MI
MI_OR_gender	MI
MI_OR_location_level3	MI
MI_OR_location_level4	MI
MI_OR_title	MI
MI_OR_years_in_service_group	MI
MS_HI_direct_span_of_control	MS
MS_HI_indirect_span_of_control	MS
MS_OR_age	MS
MS_OR_business_unit_level1	MS
MS_OR_business_unit_level2	MS
MS_OR_business_unit_level3	MS
MS_OR_days_in_service	MS
MS_OR_functional_area_level1	MS
MS_OR_functional_area_level2	MS
MS_OR_functional_area_level3	MS
MS_OR_gender	MS
MS_OR_location_level3	MS
MS_OR_location_level4	MS
MS_OR_title	MS
TE_average_HI_direct_span_of_control	TE
TE_average_HI_indirect_span_of_control	TE
TE_average_OR_age	TE
TE_average_OR_days_in_service	TE
TE_size	TE
TP_difference_12_HI_direct_span_of_control	TE
TP_difference_12_HI_indirect_span_of_control	TE
TP_difference_12_HI_layer	TE
TP_difference_6_HI_direct_span_of_control	TE
TP_difference_6_HI_indirect_span_of_control	TE
TP_difference_6_HI_layer	TE
TP_missing_12	TE
TP_missing_6	TE
TP_num_changes_12_OR_business_unit_level1	TE
TP_num_changes_12_OR_business_unit_level2	TE
TP_num_changes_12_OR_business_unit_level3	TE
TP_num_changes_12_OR_functional_area_level1	TE
TP_num_changes_12_OR_functional_area_level2	TE
TP_num_changes_12_OR_functional_area_level3	TE
TP_num_changes_12_OR_location_level3	TE
TP_num_changes_12_OR_location_level4	TE
TP_num_changes_12_OR_solid_line	TE
TP_num_changes_12_OR_title	TE
TP_num_changes_6_OR_business_unit_level1	TE
TP_num_changes_6_OR_business_unit_level2	TE
TP_num_changes_6_OR_business_unit_level3	TE
TP_num_changes_6_OR_functional_area_level1	TE
TP_num_changes_6_OR_functional_area_level2	TE
TP_num_changes_6_OR_functional_area_level3	TE
TP_num_changes_6_OR_location_level3	TE
TP_num_changes_6_OR_location_level4	TE
TP_num_changes_6_OR_solid_line	TE
TP_num_changes_6_OR_title	TE