Capturing the Spatiotemporal Dynamics of LEO ISP Performance

Forecasting Starlink Connectivity: A Data-Driven, Spatiotemporal Analysis Integrating Weather and Satellite Density

Cristian Benghe[1]

Supervisor(s): Nitinder Mohan[1], Tanya Shreedhar[1]

[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 22, 2025

Name of the student: Cristian Benghe
Final project course: CSE3000 Research Project
Thesis committee: Nitinder Mohan, Tanya Shreedhar, Qing Wang

Abstract

We present a machine learning framework aimed at forecasting Starlink (LEO satellite) network performance at fine spatiotemporal resolution. Our approach combines MLab crowdsourced measurements, weather and forecast features, and dynamic satellite density to predict packet loss, jitter, latency, and throughput. We introduce a composite Weather Index and real-time satellite density per location, and train robust ensemble models with anomaly filtering and median aggregation. Our best models achieve good predictive results with less than 17 ms for latency, and 35 Mbps for throughput. Latency is reliably predictable with meteorological and satellite context, while packet loss and jitter remain challenging. Predictions are limited to periods close to the training data and our results establish a reproducible baseline for short-term, weather-aware Starlink network forecasting.

1 Introduction

Satellite internet constellations such as Starlink [39] are rapidly transforming global connectivity, providing broadband coverage in regions previously underserved by terrestrial infrastructure. However, the performance of these networks—measured by metrics such as packet loss, latency, jitter and throughput—is highly variable and can be significantly affected by weather, geography, and satellite density. Understanding and forecasting this variability is crucial for both end-users and network planners.

Despite the promise of satellite internet constellations, users and planners still face unpredictable performance—especially under adverse weather or varying satellite availability. While recent work demonstrates weather's impact, there is limited research on integrating dense weather, satellite, and crowdsourced measurement data for accurate, location-specific forecasts. We aim to answer:

(1) **How accurately can Starlink packet loss, latency, jitter, and throughput be predicted at fine spatiotemporal resolution using weather and satellite context?**
(2) **How do machine learning models compare to simple statistical baselines and classical methods for forecasting Starlink network performance?**
(3) **What is the quantitative benefit of incorporating weather features, satellite density, and robust preprocessing strategies into predictive modeling?**
(4) **How well do these models generalize across unseen locations and future time periods, and what are the limitations of spatial and temporal extrapolation?**

Prior studies have established that environmental factors such as rain, cloudiness, and storms can degrade satellite internet quality [22, 27, 29]. Yet, existing approaches often rely on coarse spatial aggregation or lack real-time forecast integration, limiting their practical utility for location-specific prediction.

In this work, we present a data-driven, weather-aware framework for spatiotemporal forecasting of Starlink network performance. We combine crowdsourced measurements from Measurement Lab (MLab) with weather data and satellite density indices to build machine learning models capable of predicting key performance metrics at arbitrary locations and times. Our methodology incorporates both historical and forecasted weather features, leveraging APIs such as Meteostat and OpenWeatherMap for fine-grained environmental enrichment.

Our main contributions are:

(1) We provide a reproducible, data-driven pipeline to forecast Starlink quality metrics using MLab, Meteostat, OpenWeather, and satellite density features, CelesTrak being the source for satellite orbits.
(2) We quantitatively compare different data preprocessing strategies (including anomaly filtering and aggregation).
(3) We introduce a simple, interpretable Weather Index and integrate real-time satellite density at each measurement point.
(4) We evaluate our models against established baselines and report generalization to new locations.

In our paper, we will not use the server data from MLab because the server location often reflects the nearest available measurement server rather than the actual geographic origin of the client data, leading to inconsistencies when analyzing ISP-specific performance metrics, especially for decentralized networks like Starlink. Additionally, we will focus on the download measurements, the upload ones having similar computations.

2 Background

Understanding the performance of Starlink, the leading low Earth orbit (LEO) satellite Internet service, has become a rapidly evolving research area. Initial studies primarily benchmarked Starlink's throughput and latency compared to terrestrial ISPs, but there is growing attention on environmental factors, especially weather, that affect user experience.

Starlink Network Architecture. Starlink connectivity is achieved through a phased-array user terminal (popularly known as "Dishy"), which communicates with passing LEO satellites. Each user's traffic is routed via satellite to a Starlink ground station or "point of presence" (POP), which connects to the public internet [4, 15]. The path between user and internet may therefore be affected by weather both at the user terminal and at the POP location, though most measurement datasets—including ours—primarily reflect conditions at the user site.

Weather and Satellite Internet. Several recent works provide empirical evidence that weather conditions can significantly degrade Starlink performance. Kassem et al. [22] conducted a browser-based measurement campaign and reported noticeable increases in page load times during rain events. Ma et al. [29] found up to 50% throughput reduction during heavy precipitation, particularly for download traffic, highlighting Starlink's susceptibility to adverse weather.

The WetLinks Dataset. A major advance was the publication of the WetLinks dataset [27], which contains over 140,000 Starlink measurements collected alongside high-resolution weather data from professional-grade stations colocated with user terminals. Analysis of WetLinks confirmed that rain and cloud cover are the dominant drivers of download throughput degradation, with latency and upload speeds less affected. The dataset enables controlled studies of environmental impacts on Starlink links and provides a benchmark for further modeling work.

Predictive Modeling of Weather Effects. Building on this foundation, Lanfer et al. [26] developed weather-aware prediction

models for Starlink performance. Using the WetLinks dataset, they trained Random Forest and Support Vector Regression (SVR) models to forecast throughput and RTT under varying meteorological conditions. Their results demonstrate that machine learning can effectively capture the relationship between weather and Starlink network quality, paving the way for data-driven forecasting.

Gaps and Motivation. Despite these advances, most prior work has focused on relatively limited geographies (primarily Europe and North America) or on professionally instrumented terminals [22, 26, 27] which may not reflect typical user conditions. Furthermore, few studies have systematically explored spatial generalization, forecast-based prediction, or the integration of crowdsourced measurements with large-scale weather datasets. The current work addresses these gaps by leveraging globally distributed crowdsourced NDT measurements, enriched with both historical and forecasted weather features, and by proposing a reproducible pipeline for spatiotemporal Starlink performance prediction.

3 Training Methodology - Data Acquisition

This section describes the data collection, preprocessing, feature enrichment, and modeling pipeline used to analyze Starlink network performance and its relationship to meteorological conditions and satellite densities. Figure 1 summarizes the entire training data pipeline and is also relevant for the methodology in the next section.

3.1 Measurement Data Acquisition

The core dataset for this study was obtained from the Measurement Lab (MLab) NDT7 platform [25], which collects and provides open-access internet performance data globally. MLab's Network Diagnostic Tool (NDT7) [24] is designed to assess end-to-end network characteristics such as throughput, round-trip time (RTT), and jitter between clients and measurement servers. Each NDT7 measurement session consists of a sequence of multiple active probes, yielding a short time series rather than a single value per test. Raw measurements were queried from Google BigQuery using the following parameters:

- Only client measurements where `client.Network.ASNumber = 14593` (Starlink);
- Measurements dated between April 1, 2025 and May 19, 2025 both inclusive;
- Exclusion of zero-throughput results and incomplete geolocation data.

The raw MLab records used here report per-session metrics. Notably, **each row in the database represents one complete measurement session and contains a time series of 10 entries for each value**.

For preprocessing, two alternative strategies are explored and compared in this work:

- **Last Measurement:** Use only the final (last) entry value recorded in each session;
- **Median Aggregation:** Use the statistical median (`PERCENTILE_DISC`) across the 10 measurement values in each session.

We show in Section 6 that the choice between these approaches has important consequences for robustness to outliers and prediction accuracy and the median is the end choice.

No temporal or spatial aggregation is performed at the SQL stage; all further preprocessing and aggregation into coarser spatiotemporal bins is conducted post-query. The complete BigQuery SQL query is provided in Appendix A.

3.2 Geolocation Standardization

For each measurement, the city and country codes were extracted. To ensure consistency in geolocation mapping, we referenced the `worldcities.csv` dataset, which contains standardized latitude and longitude for over 40,000 global cities [2]. When city information was ambiguous or unavailable, the country's centroid was used as a fallback, ensuring no measurement was dropped due to missing fine-grained coordinates. In practice, nearly all Starlink measurements included valid city information, so the fallback was very rarely needed.

3.3 Weather Data Enrichment

To account for environmental factors affecting Starlink performance, we enriched each measurement with hourly weather data:

- **Meteostat API** [11, 31]: For each location and hour, we retrieved the nearest five weather stations (using the `.nearby()`) and extracted key hourly features: cloud cover (`coco`), precipitation, wind speed, temperature, snow, and relative humidity. Meteostat provides **hourly observations and reanalysis** with spatial granularity down to a few kilometers in populated areas, making it suitable for fine-grained network-level studies [11].
- **Fallback:** If Meteostat data was unavailable for a specific location-hour, a default value was assigned. This fallback was required for only 0.575% of records.

To enhance the analysis of Starlink network performance data, we incorporated weather conditions as a contextual factor. The enrichment pipeline processes large CSV datasets, extracting city and country information from each measurement record. To ensure accuracy, we used the before-mentioned global city coordinates dataset to map city names to geographical coordinates.

Weather Data Sources and Forecast Integration: Historical weather enrichment performed using the Meteostat API matches the most significant weather conditions impacting Starlink connectivity—cloudiness, precipitation, wind speed, snowfall, and temperature variation. These are captured both retrospectively and also in forecasts, as we are presenting in Section 5. Prior studies confirm that heavy clouds and adverse weather can substantially degrade satellite signal quality [40, 42].

Weather Index Formula: Traditional weather indices such as the Weather Severity Index (WSI)[36] or Risk Index[12] are designed for public safety, infrastructure planning, or transportation resilience, and do not align with the needs of high-resolution, satellite-specific connectivity modeling. These indices often provide categorical or daily summaries and lack the temporal granularity and quantitative semantics required for link-level network performance prediction. Moreover, they are typically computed at regional scales and are not designed to be forecastable at the hourly level across arbitrary geographic coordinates, making them incompatible with our goal of real-time, spatiotemporal performance prediction. To bridge this gap, we introduce a custom, normalized Weather Index (WI), explicitly tailored to the attenuation characteristics of Starlink. The

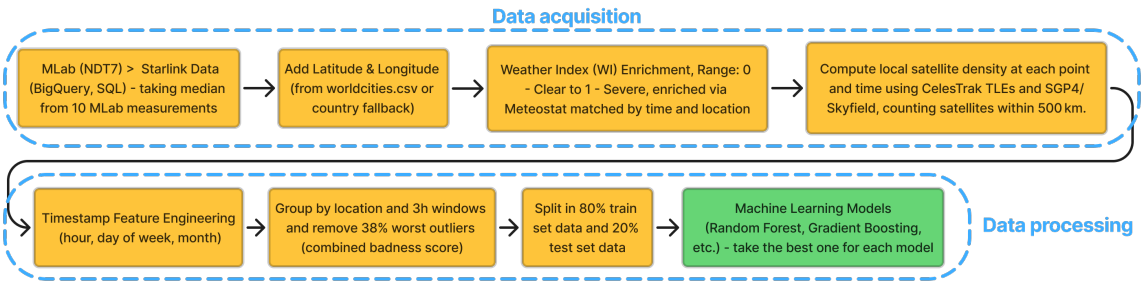

Figure 1: Overview of the data acquisition and processing pipeline, also referred as training pipeline.

WI linearly combines normalized features—cloud cover, wind speed, snow presence, and temperature deviation—weighted according to empirical findings on signal degradation severity [22, 27]. This index is lightweight, interpretable, and compatible with many weather APIs. Thus, also compatible with both historical measurements (via Meteostat) and the 48-hour forecasts (via OpenWeather API) that will be presented in the next section. The formula for the Weather Index (WI) is as follows:

$$\begin{aligned} \text{WI} = 0.7 \cdot \text{cloudiness} + 0.15 \cdot \text{wind\_speed} \\ +0.1 \cdot \text{snow} + 0.05 \cdot \text{temp} \end{aligned} \tag{1}$$

Where:

- **Cloudiness**: Normalized cloud cover (0 to 1) calculated from the `coco` field using a mapping function.
- **Wind Speed:** wind_speed = `wind.speed`/20 (assuming 20 m/s as the upper bound for normalization)
- **Snow:** snow = `snow.3h`/10 (assuming 10 mm/3h represents heavy snowfall)
- **Temperature:** temp = (`main.temp` + 50)/100 (maps temperatures from $[-50°C, +50°C]$ to $[0, 1]$)

Justification of Weights. The weighting of features in our Weather Index is informed by recent studies that highlight cloud cover as the dominant factor impacting Starlink and other LEO satellite performance. For example, Laniewski et al. [27] and Kassem et al. [22] both find that cloud cover and precipitation are strongly correlated with throughput degradation, with cloud cover responsible for increases in latency and jitter, and rain causing a 37–52% reduction in Starlink throughput. Wind and snow were found to have secondary and less pronounced effects [28, 29]. These findings support the assignment of a higher weight to cloudiness (0.7) and smaller weights to wind speed (0.15), snow (0.1), and temperature (0.05) in our index. As a robustness check, we also performed a sensitivity analysis by varying the weights and observed that the model's performance was relatively stable. A more exhaustive analysis is left for future work.

Handling Missing Data: If the Meteostat `coco` field is missing (`NaN`), we attempt to retrieve data from the five nearest stations; if still unavailable, we default the Weather Index to 0.1 (clear).

Cloudiness and extreme weather are designed to emphasize their increased impact during severe conditions, as observed in studies on Starlink's performance in adverse weather [40, 42]. The Weather Index is higher in poor weather conditions and lower when the weather is clear. Case studies of how the performance is correlated to our WI can be found in Appendix C and Appendix D.

Mapping Cloudiness Codes: The `coco` field from Meteostat is converted to corresponding cloudiness values. The full mapping and detailed explanation are provided in Appendix K.

This mapping is designed to reflect the impact of different weather conditions on satellite communication quality. Severe weather conditions, such as thunderstorms and heavy snowfall, are given higher cloudiness scores, reflecting their bad effects on signal quality.

3.4 Satellite Density Enrichment

To further contextualize each measurement with respect to the available Starlink infrastructure, we enriched the dataset with a **satellite density** feature. This attribute quantifies the number of Starlink satellites physically present within a given radius of each measurement point at the precise test timestamp.

Data Source and Motivation. The raw satellite position data, including two-line element (TLE) sets for all operational Starlink satellites, was sourced periodically from CelesTrak and provides a continuous record of Starlink's orbital state vectors from May 14, 2023 to June 5, 2025. The Two-Line Element set (TLE) is a standardized data format for encoding satellite orbital elements, maintained and distributed by organizations such as CelesTrak [23].

Density Computation Pipeline. For each measurement, the following steps were performed:

(1) **Timestamp alignment:** The TLE file closest in time to the measurement's UTC timestamp was selected for orbit propagation, same day precision was considered enough for the orbit data.
(2) **Orbit propagation:** Starlink satellite orbits were propagated to the measurement timestamp using the SGP4 algorithm [38, 44], via a threaded Python implementation. [1]
(3) **Geodetic mapping:** Satellite positions were converted to geodetic (latitude, longitude) coordinates using the Skyfield library [37], which provides high-precision transformations from TEME to ITRF/ITRS [32, 44].
(4) **Spatial counting:** The geodesic distance between each satellite and the test measurement location was computed. The number of satellites within a $r = 500$ km radius was recorded as the satellite density for that measurement.

Implementation. The python implementation processed grouped measurements efficiently by loading each day's TLEs only once.

[1] Parallel SGP4 propagation used Python's `ThreadPoolExecutor` and `sgp4` [38]. Code: https://github.com/TUD-BScResearchProject-6079/model-training/blob/master/_develop-sat-density-predictor/SG4-Satellite-Density-Enrichment.py

The pipeline used the `sgp4` package for orbit propagation [38] and the `skyfield` library for geodetic conversion [37].

Scientific Motivation. Satellite density is a critical contextual factor for LEO networks. A higher local satellite density generally indicates more available line-of-sight paths and lower probability of terminal congestion, thus influencing achievable throughput and latency [28]. Including this dynamic spatial feature enables the model to capture short-term, location-specific variations in Starlink performance.

The Satellite Density Index (*SDI*) quantifies the number of Starlink satellites per unit area within a radius r which we set to 500km:

$$SDI = \frac{N}{A}, \quad A = \pi r^2$$

where N is the number of satellites within radius r, and A is the corresponding area. A higher SDI indicates a higher concentration of satellites, which can correlate with better connectivity.

4 Training Methodology - Data Processing

4.1 Temporal Feature Engineering

All measurements were timestamped in UTC. For each observation, we extracted the following temporal features:

- **Hour of day**, to capture day/night effects and peak usage.
- **Day of week** (0=Monday, 6=Sunday), enabling the model to distinguish between weekdays and weekends.
- **Month**, to account for possible seasonal variation.

This feature extraction allows the model to learn and exploit regular temporal structures in the data—such as higher congestion during certain hours or on weekends—enabling improved prediction of network performance. By including these features, we also decorrelate satellite density effects from simple timestamp effects, ensuring that our model's predictions are not confounded by satellite passes or UTC time alignment.

Figure 2 illustrates the utility of temporal features: the left panel shows a clear diurnal loaded latency pattern at a sample site (Manila, Philippines), while the right panel shows the corresponding diurnal packet loss pattern at another site (Seattle, USA). Both plots cover the full 24-hour day–night cycle. Notably, the lowest latency values in Manila occur at night (around 20 UTC, which corresponds to nighttime locally), while in Seattle, the minimum packet loss and sample count are both observed around 10 UTC—corresponding to local nighttime hours as well. These patterns underscore the importance of modeling temporal effects. Feature engineering of hour and day-of-week enables the model to exploit these repeatable diurnal and weekly effects. Appendix F presents an analysis of weekday versus weekend effects, demonstrating that although differences can be observed, their direction and magnitude vary by site, precluding a simple, general conclusion.

4.2 Outlier Removal and Anomaly Filtering

Crowdsourced measurements are inherently noisy, so robust outlier filtering is essential for reliable modeling. For each unique latitude and longitude, we grouped measurements into rolling 3-hour windows and computed a composite *badness* score for each record:

$$\text{badness} = \text{PL} + \frac{\text{Jitter}}{100} + \frac{\text{Latency}}{100} + (1 - \text{NormalizedThroughput})$$

where *NormalizedThroughput* is the throughput divided by the maximum observed in the respective window. Within each window, we removed the worst 25% of records (those with the highest badness), retaining the most representative 75%. This method, based on established practices for crowdsourced network measurement [18, 20], effectively reduces noise while preserving data diversity. The choice of a 25% removal threshold is supported by empirical analysis (see Appendix E), which found similar performance to more aggressive filtering but with less data loss.

4.3 Feature Selection and Model Inputs

After preprocessing, the following features were used as model inputs: Latitude, Longitude, Hour, Day of Week, Month, Weather Index (WI), Satellite Density. The model targets were: Packet Loss Rate, Download Jitter (ms), Loaded Latency (ms) and Throughput (Mbps).

4.4 Model Training and Validation

We employ ensemble learning combining Random Forest and Gradient Boosting regressors for predicting packet loss, download jitter, latency and throughput inspired by the approach in [26]. The ensemble method leverages the strengths of both models, improving prediction accuracy and robustness. Model hyperparameters were optimized using cross-validation techniques.

To rigorously assess model generalization, we split the cleaned and anomaly-filtered dataset into 80% for training and 20% for testing. This 80–20 division was done randomly, ensuring that evaluation metrics reflect the model's ability to predict on unseen data. A random 80–20 split is a widely accepted standard for supervised learning evaluation [21].

We trained both Random Forest and Gradient Boosting regression models using scikit-learn [35]. The best-performing model for each metric (including a 60:40 ensemble) was selected based on validation RMSE. Feature normalization was performed using RobustScaler also from `scikit-learn` [35] to reduce the influence of outliers.

5 Prediction Methodology - Grids and Cities

To enable interpretable and global prediction, we generate a regular grid of locations using the H3 hexagonal system [1]. For each hour in a 48-hour prediction window, we predict performance at the centroid of each permitted level-2 H3 hexagon. Hexes are excluded if they are more than 710 km from any training measurement (to avoid unreliable extrapolation and keep the number of hexagons low), fall in oceans or countries where LEO operation is banned. This produces roughly 800 hexes, visualized in the web interface (see appendix Fig. H). Locations outside this mask are omitted.

We visualize predictions across space and time using color-coded hexagons representing network quality classes. A color logic scheme maps PacketLossRate, DownloadJitter, LoadedLatency and Throughput to discrete color bands ranging from green (excellent) to dark red (severe degradation), with full color thresholds provided in Appendix I. Figure 4 illustrates predicted Starlink network quality for two selected hours (2025-06-12, 15:00 and 23:00 UTC) as rendered on the global H3 grid and as points for key cities, respectively.

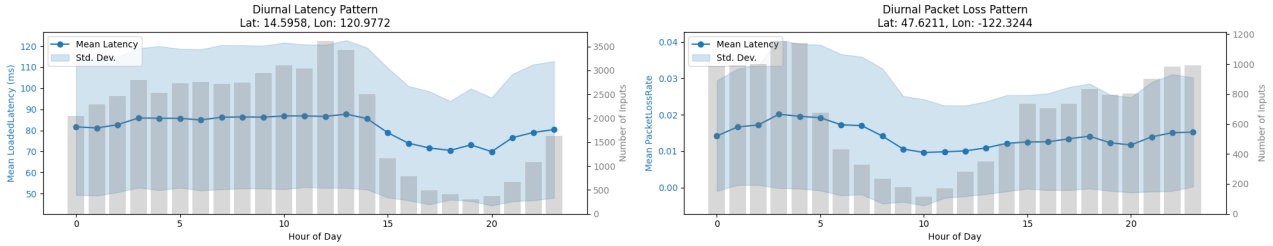

Figure 2: Examples of diurnal variation in loaded latency (left, Manila, Philippines) and packet loss (right, Seattle, Washington, USA) at distinct sites. Shaded regions indicate standard deviation, while bars denote the number of samples per hour.

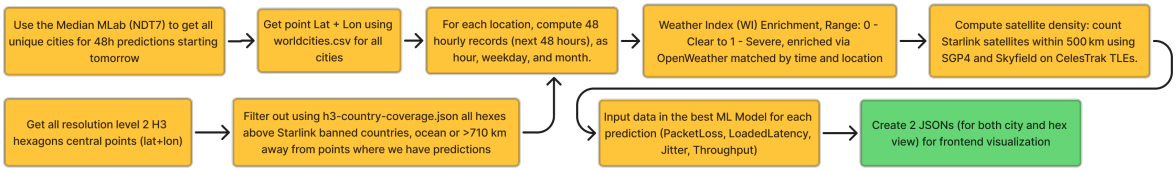

Figure 3: Overview of the prediction pipeline.

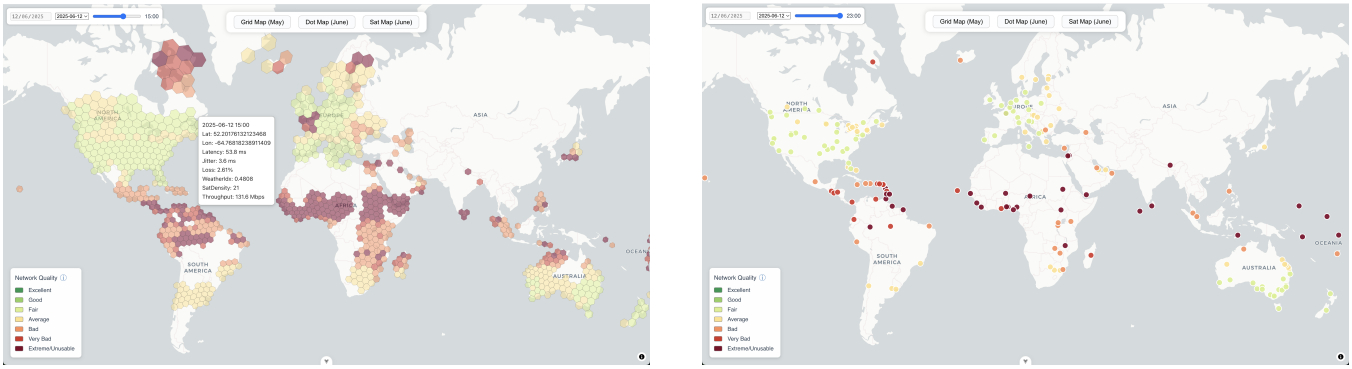

Figure 4: Spatiotemporal predictions of Starlink network quality for June 12, 2025. Left: Global predictions at 15:00 UTC, visualized using 802 H3 hexagons. Right: Predictions at 23:00 UTC for 196 major city locations. Both panels use the same color scale, mapping excellent quality to green and severe degradation to dark red (see Appendix I). The total number of hexes and cities is chosen to stay just under the daily 1,000-call OpenWeatherMap API limit.

For each hex-hour, weather features are generated using real-time OpenWeatherMap forecasts, providing the required inputs to our Weather Index formula. The process is fully automated and runs daily, enabling both next-day and day-after-tomorrow forecasts (see Fig. 3).[2]

Weather mapping details: To derive a cloudiness value in the range [0, 1] from OpenWeather's discrete weather codes, we use a mapping approach consistent with our Meteostat-based training. For example, Clear (800) is mapped to 0.0, Few clouds (801) to 0.1, Scattered clouds (802) to 0.5, Broken/overcast (803/804) to 0.9, and mist, smoke, haze, or dust (700–781) to 0.7. A complete mapping, as well as a detailed comparison between the OpenWeather and Meteostat code-to-cloudiness mappings used during training and prediction, can be found in Appendix K. Wind speed, snow amount, and temperature are also normalized as described in our code. Specifically, each feature is scaled to $[0, 1]$ as in the training part. (See 3.3) Hourly values are linearly interpolated from the 3-hourly OpenWeather endpoint. In rare cases where OpenWeatherMap data is unavailable for a given entry, we assign a default value equal to 0.1 - clear weather, similar to Section 3.3. In practice, missingness was negligible.

Satellite density: A simplified non-threaded algorithm is used for gathering densities as less points require computation compared to training. The most recent CelesTrak TLE file is fetched and satellite future positions are computed towards finding densities for each point.

6 Evaluation & Results

This section presents four evaluation approaches. First, we tested a pipeline using the last recorded value of each measurement session from the 10-measurement MLab, but found this led to inflated

[2]Weather enrichment, satellite enrichment, and prediction scripts are present in Appendix G.

Table 1: Prediction accuracy for all targets under 25% anomaly filtering. Models compared: baselines, decision trees (RF, GBR), and their ensemble. Lower MAE/RMSE is better; higher R^2 is better. Best value is green-colored.

Target	Model	MAE	RMSE	R^2
PacketLossRate (0-1)	Dummy (Mean)	0.0377	0.0547	-0.0000
	Dummy (Median)	**0.0325**	0.0591	-0.1652
	LinearRegression	0.0366	0.0537	0.0358
	Shallow Tree	0.0355	0.0526	0.0752
	KNN (k=5)	0.0372	0.0562	-0.0563
	Gradient Boosting	0.0351	**0.0523**	**0.0876**
	Random Forest	0.0370	0.0562	-0.0534
	Ensemble (GBR+RF)	0.0355	0.0531	0.0590
DownloadJitter (ms)	Dummy (Mean)	2.7944	5.1023	-0.0000
	Dummy (Median)	**2.5405**	5.2467	-0.0574
	LinearRegression	2.7243	4.9730	0.0500
	Shallow Tree	2.6300	4.8420	0.0994
	KNN (k=5)	2.8001	5.2938	-0.0765
	Gradient Boosting	2.5627	**4.7399**	**0.1370**
	Random Forest	2.7973	5.3744	-0.1095
	Ensemble (GBR+RF)	2.6254	4.9208	0.0699

Target	Model	MAE	RMSE	R^2
LoadedLatency (ms)	Dummy (Mean)	26.4925	38.8452	-0.0000
	Dummy (Median)	24.9221	40.1221	-0.0668
	LinearRegression	24.9719	36.1212	0.1353
	Shallow Tree	22.0922	32.3836	0.3050
	KNN (k=5)	18.3240	25.8776	0.5562
	Gradient Boosting	17.5710	24.9281	0.5882
	Random Forest	18.2761	25.9947	0.5522
	Ensemble (GBR+RF)	**17.4621**	**24.6096**	**0.5986**
Throughput (Mbps)	Dummy (Mean)	65.7713	79.6644	-0.0000
	Dummy (Median)	64.0664	82.0388	-0.0605
	LinearRegression	61.4818	75.8038	0.0946
	Shallow Tree	58.9972	73.6828	0.1445
	KNN (k=5)	57.8665	74.7639	0.1192
	Gradient Boosting	55.9662	**70.1280**	**0.2251**
	Random Forest	57.7700	75.1978	0.1090
	Ensemble (GBR+RF)	**55.6766**	70.9080	0.2077

predictions across all regions; a comparison of preprocessing strategies is given in Section 6.1. Next, in Section 6.2, we evaluate model performance on a 20-80 train-test split and benchmark against standard baselines. We then assess the model's predictive capabilities on real-world data. Finally, we directly compare the enriched models to their bare counterparts to evaluate the impact of Satellite Density and Weather Index enrichment.

We evaluated the prediction model on 3-hour binned ground truth measurements aggregated from MLab [6]. For temporal alignment, timestamps were grouped into hourly bins. Evaluation metrics included the **Mean Absolute Error (MAE)** [45], **Root Mean Squared Error (RMSE)** [10], and the **coefficient of determination (R^2)** [13], computed across all prediction targets. MAE measures the average magnitude of prediction errors, providing a clear and interpretable assessment of model accuracy. RMSE penalizes larger errors more strongly, highlighting sensitivity to outliers. The R^2 score indicates the proportion of variance in the ground truth explained by the model, serving as a standard measure of predictive power. [19, 21] Collectively, these metrics enable a comprehensive evaluation of both the precision and reliability of the model's predictions. For full metric definitions see Appendix L.

6.1 Comparison of Preprocessing Strategies

We evaluated two data preprocessing strategies for constructing our predictive models: Last Measurement Model, Median Aggregation Model.

Table 2 summarizes the model performance for both strategies, evaluated on April 1st data, with mean absolute error (MAE), root mean squared error (RMSE), and the coefficient of determination (R^2). The results are reported for each target variable. This supports prior findings that aggregation reduces measurement noise in crowdsourced network datasets [18, 27].

The median aggregation model demonstrates **notably lower error** for download latency and jitter compared to the last measurement model, which tends to capture transient spikes at the session end. While both strategies yield nearly identical results for packet loss and throughput, the median approach consistently achieves better or similar R^2 across all metrics.

It is important to note that **no anomaly filtering or outlier removal** was applied in this evaluation: models were trained and tested on fully raw data as collected, including any transient or extreme measurement values. Additionally, **no weather feature enrichment or satellite density enrichment** was performed in these experiments. The results therefore reflect the predictive difficulty in this purely data-driven setting and explain the relatively high errors and negative R^2 observed for some metrics, especially jitter.

Despite the "last" strategy achieving slightly lower error in packet loss (identical MAE and RMSE), it produces much higher errors for latency and jitter, indicating susceptibility to outliers and session-end artifacts. The median aggregation offers more **robustness** and **generalizability** to unseen conditions and noisy measurements. We thus recommend median aggregation for spatiotemporal forecasting tasks in this domain.

6.2 Baseline and Model Comparison

Table 2: Model preprocessing performance: Median vs. Last aggregation.

Measurement	Metric	Median	Last
Packet Loss (%)	MAE	0.0304	0.0304
	RMSE	0.0508	0.0508
	R^2	0.0263	0.0266
Jitter (ms)	MAE	8.07	11.05
	RMSE	42.79	50.50
	R^2	−0.0021	−0.0012
Latency (ms)	MAE	33.44	56.94
	RMSE	91.75	140.94
	R^2	0.0531	0.0382
Throughput (Mbps)	MAE	48.16	48.06
	RMSE	65.26	65.21
	R^2	0.0711	0.0702

We compare our smart models to several baselines—Dummy Mean and Median, Linear Regression, Shallow Decision Tree, and KNN—all evaluated with 25% anomaly filtering to ensure robust results. Table 1 summarizes predictive accuracy for four network metrics

(PacketLossRate, DownloadJitter, LoadedLatency, Throughput) on the filtered evaluation set. The ensemble model (combining Gradient Boosting and Random Forest) consistently outperforms these baselines.

Each baseline serves a distinct purpose in assessing model performance. The **mean** and **median** dummy predictors offer naive reference points, always predicting the mean or median value from the training set; the mean predictor is especially important for interpreting R^2, as R^2 quantifies variance explained over this baseline [16]. **Linear regression** is included for its interpretability and to test whether simple relationships suffice [21]. A **shallow decision tree** baseline allows us to capture basic nonlinear effects and feature interactions with low complexity [7]. The ***k*-nearest neighbors (KNN)** approach leverages local similarity for prediction and provides a non-parametric view of the data [5]. By including results both with and without anomaly filtering, we ensure that performance improvements from advanced models are meaningful, robust, and not driven by outliers or noise.

6.3 Model Performance Analysis and Discussion

Our models demonstrate varying predictive success across the four main Starlink network metrics, reflecting underlying differences in signal dynamics and data complexity.

Packet Loss Rate remains particularly challenging: all models, including the ensemble, show MAE values around 0.03–0.04 and R^2 scores close to zero or negative (Table 1), consistent with prior findings that loss is highly stochastic and poorly captured by available features [34]. Even so, ensemble methods achieve slightly lower MAE than naive baselines, suggesting some incremental benefit.

Download Jitter predictions show moderate improvement, with the ensemble model reaching an R^2 of 0.14 and reducing MAE to 2.56 ms, compared to 2.79 ms for the mean baseline. Anomaly filtering improves robustness for jitter and other metrics, underscoring the importance of removing outliers [9].

Loaded Latency stands out for its high predictability. Here, the ensemble achieves an MAE of 17.46 ms and an R^2 of 0.60, greatly outperforming both linear regression (MAE 24.97 ms, R^2 0.14) and the mean predictor (MAE 26.49 ms, R^2 0.00). This aligns with literature showing that latency in LEO systems is more strongly linked to weather, satellite density, and time features [43].

Throughput is moderately predictable, with the ensemble lowering MAE to 55.7 Mbps and R^2 to 0.21, compared to 65.8 Mbps MAE and near-zero R^2 for the mean. External factors, including congestion and terminal variability, still limit prediction accuracy [17].

Overall, the use of MAE, RMSE, and R^2 allows for nuanced performance evaluation. Ensemble methods based on Gradient Boosting and Random Forest consistently outperform simpler models for most targets, especially after anomaly filtering, while packet loss remains inherently difficult to model.

These results emphasize the value of integrating meteorological and temporal context, as well as careful preprocessing, to enable reliable network quality forecasting—paving the way for improved user experience and network management.

6.4 Model Evaluation via Time-Window Ground Truth Matching

We evaluate our trained model on two distinct days: May 20th (a weekday) and May 24th (a weekend), using a robust comparison between predictions and real-world ground truth. The overall process is summarized in Figure 5, which visualizes the enrichment of predictions and their matching to ground truth measurements.

The model is trained on data collected between April 1st and May 19th, using the filtered subset obtained from the 25% outlier removal process (Section 4.2). In Appendix J we presented the 38% version that leads to similar results.

Time-Window Median Matching. For each prediction record—defined by latitude, longitude, and hour—we construct a timestamp and match it against all available ground truth measurements within a ±1.5 hour window grouping by latitude and longitude.

If at least one match is found, we compute the median value of each of the four ground truth metrics: PacketLossRate, DownloadJitter, LoadedLatency, and Throughput. This window-based median provides a more stable and representative target value, especially in the presence of measurement noise.

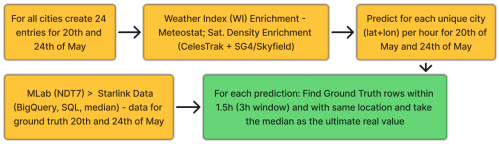

Figure 5: Overview of the spatiotemporal evaluation pipeline. Predictions on May 20 and 24 are enriched with Weather Index and Satellite Density, and matched to median ground truth measurements from MLab within a ±1.5h window

Table 3 highlights the results we collected in this approach. We find that latency and throughput can be predicted with moderate accuracy—particularly on weekdays—while packet loss and jitter remain more difficult due to their bursty and unpredictable nature. Notably, we see a performance drop on the weekend day, reflecting shifts in user behavior, but most probably the gap between the last training day (May 19) and the evaluation day (May 24). This highlights the importance of regular model retraining to maintain accuracy when forecasting more than two days ahead.

Table 3: Model performance on May 20th and May 24th

Date	Metric	MAE	RMSE	R^2
20 May	Packet Loss	0.0279	0.0379	−0.0018
	Download Jitter	2.73 ms	9.11 ms	0.0154
	Loaded Latency	17.4 ms	34.2 ms	0.5727
	Throughput	41.5 Mbps	58.6 Mbps	0.2640
24 May	Packet Loss	0.0280	0.0405	−0.0083
	Download Jitter	4.48 ms	41.61 ms	0.0069
	Loaded Latency	20.8 ms	78.8 ms	0.2167
	Throughput	38.2 Mbps	53.9 Mbps	0.2835

6.5 Evaluation Results and Enrichment Impact

To address our research question—whether enriching the model with Satellite Density and Weather Index (WI+SD) improves performance—we evaluated both enriched and baseline models on unseen data from two representative days (Table 4).

Table 4: Model evaluation for May 20 and 24, with and without Satellite Density + Weather Index enrichment (WI+SD).

		20 May		24 May	
Target	Metric	**WI+SD**	**Bare**	**WI+SD**	**Bare**
Latency	MAE	17.31	18.38	20.83	21.95
	RMSE	34.60	38.13	78.79	81.07
	R^2	0.5731	0.4816	0.2167	0.1707
Throughput	MAE	40.82	38.37	38.16	35.67
	RMSE	58.04	53.02	53.86	49.12
	R^2	0.2699	0.3907	0.2835	0.4039

The results show that **WI+SD enrichment consistently improves latency prediction on both test days, reducing MAE and RMSE while increasing R^2—with the largest gains observed on May 20, where latency MAE drops by over 1 ms and the explained variance increases by nearly 10%.** This demonstrates that incorporating spatial and meteorological context enables the model to better generalize to new time periods, directly supporting our main hypothesis.

In contrast, throughput prediction does not benefit from WI+SD features: the bare model performs slightly better. This suggests that, while latency is strongly influenced by satellite coverage and atmospheric conditions, throughput depends on additional factors not captured by our enrichment.

This evaluation confirms that augmenting input features with satellite density and weather index substantially enhances latency prediction accuracy on unseen data, while alternative strategies are needed to improve throughput prediction. This directly answers our research question and establishes both the value and the limits of spatiotemporal enrichment for LEO performance modeling.

7 Future Work

Several avenues remain open to enhance the predictive performance and scientific rigor of our forecasting framework.

First, more advanced feature engineering could improve model expressiveness. For example, representing temporal variables such as hour of day or day of week using sine and cosine transformations would better capture their cyclical, periodic nature. This approach could help models more effectively learn daily and weekly patterns.

Second, we currently cannot evaluate predictions for all H3 cells, especially those without ground truth. Future work should explore methods for handling regions with missing data, such as generating synthetic data, using spatial transfer learning, or quantifying prediction uncertainty.

Our present satellite density metric counts Starlink satellites within a 500km circle above each site. However, the user terminal ("Dishy") actually sees an **elliptical** sky region, not a perfect circle[8]. Refining this metric using elliptical projections could yield more accurate density estimates.

The model also does not yet consider **solar activity** (e.g., flares, geomagnetic storms), which can disrupt satellite communications or destroy individual satellites [14, 30, 46]. Adding a "space weather risk" feature, derived from real-time solar indices, could improve predictions during adverse space conditions.

Another key direction is to explicitly incorporate the distance to the nearest Point-of-Presence (POP) into our models. Because NDT7 measurements include the server IP and geolocation, we can infer which POP served each test. Introducing a "distance-to-POP" feature would allow us to account for the influence of network backhaul and terrestrial routing on latency and throughput—factors not captured by satellite or weather context alone. This feature would also enable the use of a much broader historical dataset, including time periods or regions where local POPs were absent, thereby extending our training window from months to years. Implementing this enhancement requires updating preprocessing scripts to extract POP locations from server data and calculate client-to-POP distances for every measurement.

Finally, as Starlink expands to new countries (e.g., India [41]), future studies should test model generalization and network quality in these diverse regions.

8 Conclusion

Our results demonstrate that Starlink network latency and throughput can be predicted at fine spatiotemporal resolution using open, crowdsourced data enriched with contextual features. However, accurate forecasting for a target day requires both a substantial history of measurements and a rolling retraining approach—models must be updated daily with recent data to capture temporal dynamics and seasonal effects.

Crucially, our experiments highlight that **aggressive anomaly filtering** is essential to achieve reliable predictions, especially for noisy metrics such as latency and jitter. Without robust data cleaning, models are prone to overfit transient spikes and measurement artifacts, leading to poor generalization.

Evaluation using a robust matching methodology (ground truth medians within ±1.5-hour windows) confirms that latency and throughput are the most predictable metrics, while packet loss rate and jitter remain more challenging. Our findings suggest that consistent data collection, rigorous filtering, and frequent model updates are critical for actionable forecasting in dynamic LEO satellite networks.

The best model for loaded latency uses Satellite Density and Weather Index (WI+SD) enrichment, while the best throughput results are achieved with the bare model. In future work, we will explore additional enrichment features aimed specifically at improving throughput prediction to match the gains achieved for latency.

Overall, this thesis provides a robust framework and a benchmark for future predictive analytics in LEO satellite networks, setting a foundation for smarter, more adaptive Internet service management as coverage continues to expand.

9 Responsible Research

This work fully complies with the ethical guidelines of TU Delft and ACM for responsible research. All data used are either aggregated, anonymized, or openly available. No individual or personal information is processed or published: MLab measurements are crowdsourced and de-identified at source, while all weather and satellite data are public and used solely for scientific analysis.

Reproducibility is ensured through the availability of all code and scripts for data preprocessing, model training, and prediction in the supplementary materials. The entire pipeline, from raw MLab queries to final hourly predictions for arbitrary locations, is documented and reproducible. No proprietary or privacy-sensitive data are handled. Future work should continue to monitor potential risks of geographical re-identification and adhere to all relevant data provider terms.

For full reproducibility, all source code is openly available:

- Model creation and training scripts: https://github.com/TUD-BScResearchProject-6079/model-training
- Application (frontend and backend): https://github.com/TUD-BScResearchProject-6079/leo-viewer

This paper benefited from language improvements and rephrasing with the assistance of ChatGPT 4.1.

References

[1] 2023. H3: Uber's Hexagonal Hierarchical Spatial Index. https://h3geo.org/.

[2] 2023. World Cities Dataset. https://simplemaps.com/data/world-cities.

[3] 2024. Meteostat: Weather Condition Codes. https://dev.meteostat.net/formats.html#weather-condition-codes.

[4] Jasmul Alam, Md.Sakir Hossain, Jauwad Ansari, AbuZafarMd. Imran, and Imtiaz Kamrul. 2021. Estimation of Rain Attenuation of Earth-to-Satellite Link over Nepal for Ku & Ka Bands. *arXiv preprint arXiv:2109.08032* (2021). https://arxiv.org/abs/2109.08032 Submitted Aug 2021; Focuses on spatial/temporal fade margins of 24–80dB for Ku and 40–80dB for Ka bands across Nepalese regions.

[5] Naomi S. Altman. 1992. An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression. *The American Statistician* 46, 3 (1992), 175–185.

[6] Robert et al. Beverly. 2010. Measurement Lab (M-Lab): Infrastructure for the Measurement and Analysis of Internet Connectivity. In *Passive and Active Measurement Conference (PAM)*.

[7] Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. 1984. *Classification and Regression Trees*. Wadsworth.

[8] Shkelzen Cakaj, Bexhet Kamo, Algenti Lala, and Alban Rakipi. 2014. The Coverage Analysis for Low Earth Orbiting Satellites at Low Elevation. *International Journal of Advanced Computer Science and Applications* 5, 6 (2014). doi:10.14569/IJACSA.2014.050602

[9] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly Detection: A Survey. *Comput. Surveys* 41, 3 (2009), 1–58. doi:10.1145/1541880.1541882

[10] Davide Chicco and Giuseppe Jurman. 2021. The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC Genomics* 21, 6 (2021), 1–13. RMSE discussion included.

[11] Meteostat Contributors. 2025. Meteostat: Free and Open Weather and Climate Data with Hourly Observations. https://dev.meteostat.net/. Accessed: 2025-06-22.

[12] Federal Emergency Management Agency. 2021. *National Risk Index Technical Documentation*. Technical Report. U.S. Department of Homeland Security. https://www.fema.gov/sites/default/files/documents/fema_national-risk-index_technical-documentation.pdf Describes regional hazard indices for public safety and infrastructure, including limitations of granularity and forecastability.

[13] Jian Gao. 2024. R-Squared (Rˆ2) – How much variation is explained? *Research Methods in Medicine & Health Sciences* 5, 4 (2024), 104–109. doi:10.1177/26320843231186398

[14] J. C. Green, J. Likar, and Y. Shprits. 2017. Impact of Space Weather on the Satellite Industry. *Space Weather* 15, 6 (2017), 804–818. doi:10.1002/2017SW001646

[15] Mark Handley. 2018. Delay is Not an Option: Low Latency Routing in Space. In *Proceedings of the 17th ACM Workshop on Hot Topics in Networks (HotNets '18)*. 85–91. doi:10.1145/3286062.3286075

[16] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2009. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (2 ed.). Springer. doi:10.1007/978-0-387-21606-5

[17] Henrik Hellström, José Mairton Barros da Silva, Mohammad Mohammadi Amiri, Mingzhe Chen, Viktoria Fodor, H. Vincent Poor, and Carlo Fischione. 2022. Wireless for Machine Learning: A Survey. *Foundations and Trends® in Signal Processing* 15, 4 (2022), 290–399. doi:10.1561/2000000114

[18] Matthias Hirth, Tobias Hoßfeld, Marco Mellia, Christian Schwartz, and Frank Lehrieder. 2015. Crowdsourced Network Measurements: Benefits and Best Practices. *Computer Networks* 85 (2015), 20–36. doi:10.1016/j.comnet.2015.07.003

[19] Rob J Hyndman and Anne B Koehler. 2006. Another look at measures of forecast accuracy. *International Journal of Forecasting* 22, 4 (2006), 679–688. doi:10.1016/j.ijforecast.2006.03.001

[20] ITU-T. 2011. ITU-T Recommendation G.1010: End-user multimedia QoS categories. https://www.itu.int/rec/T-REC-G.1010.

[21] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2013. *An introduction to statistical learning*. Springer.

[22] Mohamed M. Kassem, Ashwin Raman, Dario Perino, and Naveen Sastry. 2022. A Browser-side View of Starlink Connectivity. In *Proceedings of the 22nd ACM Internet Measurement Conference (IMC '22)*. 151–158. https://dl.acm.org/doi/10.1145/3517745.3561457

[23] T.S. Kelso. 2024. CelesTrak: Satellite Two-Line Element Set Format. https://celestrak.org/columns/v04n03/.

[24] Measurement Lab. 2023. NDT7: Network Diagnostic Tool v7 Protocol Specification. https://www.measurementlab.net/tests/ndt/ndt7/.

[25] Measurement Lab. 2024. Measurement Lab (M-Lab) NDT7 Dataset. https://www.measurementlab.net/tests/ndt/.

[26] Eric Lanfer, Dominic Laniewski, Daniel Otten, and Nils Aschenbruck. 2024. Weather-Based Link Prediction for LEO-Satellite Networks using the WetLinks Dataset. In *Proceedings of IFIP Networking Conference*. https://doi.org/10.23919/IFIPNetworking62109.2024.10619789

[27] Dominic Laniewski, Eric Lanfer, Bernd Meijerink, Roland van Rijswijk-Deij, and Nils Aschenbruck. 2024. WetLinks: A Large-Scale Longitudinal Starlink Dataset with Contiguous Weather Data. *Proceedings of the 8th Network Traffic Measurement and Analysis Conference (TMA)* (2024). https://doi.org/10.23919/TMA62044.2024.10558998

[28] Sami Ma, Yi-Ching Chou, Haoyuan Zhao, Long Chen, Xiaoqiang Ma, and Jiangchuan Liu. 2022. Network Characteristics of LEO Satellite Constellations: A Starlink-Based Measurement from End Users. *arXiv preprint arXiv:2212.13697* (2022). https://arxiv.org/abs/2212.13697 Accessed 2025-06-22.

[29] Shuai Ma, Yung-Chih Chou, Haoyi Zhao, Lei Chen, Xiaoyong Ma, and Jian Liu. 2023. Network Characteristics of LEO Satellite Constellations: A Starlink-Based Measurement from End Users. *Proceedings of IEEE INFOCOM 2023* (2023), 1–10.

[30] Mark MacAlester. 2021. *Space Weather Effects on Communications Systems*. Technical Report. NOAA Space Weather Prediction Center. https://www.swpc.noaa.gov/sites/default/files/images/u63/01.%20MacAlester_Space%20Weather%20Effects%20on%20Communications%20Systems.pdf Technical report.

[31] Meteostat. 2025. meteostat Python Package (v1.7.4). PyPI. https://pypi.org/project/meteostat/1.7.4 Provides bulk access to historical weather station data.

[32] Oliver Montenbruck and Eberhard Gill. 2000. *Satellite Orbits: Models, Methods and Applications*. Springer.

[33] National Oceanic and Atmospheric Administration (NOAA). 2024. Hurricane Helene Makes Landfall: Rainfall Totals and Impacts for Atlanta, Georgia. NOAA Event Report. https://www.weather.gov/ Atlanta experienced 11.12 inches of rain over 48 hours, breaking a 104-year record and leading to widespread flooding..

[34] Jitendra Padhye, Vlad Firoiu, Don Towsley, and James Kurose. 2000. Modeling TCP throughput: A simple model and its empirical validation. *ACM SIGCOMM Computer Communication Review* 28, 4 (2000), 303–314. https://dl.acm.org/doi/10.1145/285243.285291

[35] F. Pedregosa et al. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830. https://jmlr.org/papers/v12/pedregosa11a.html

[36] Paul Pisano, Andrew Stern, and Lance Goodwin. 2011. *Clarus Regional Demonstration Program*. Technical Report. US Department of Transportation. https://ams.confex.com/ams/pdfpapers/163627.pdf.

[37] Brandon Rhodes. 2019. Skyfield: High precision astronomy for Python. https://rhodesmill.org/skyfield/, Accessed June 2025.

[38] Brandon Rhodes and David A. Vallado. [n. d.]. sgp4: Satellite Orbit Propagation in Python. https://pypi.org/project/sgp4/, Accessed June 2025.

[39] Space Exploration Technologies Corp. 2025. *Starlink: High-Speed Internet From Space*. https://www.starlink.com/ Accessed: 2025-06-22.

[40] Starlink Support. 2025. Does weather impact my service quality? https://www.starlink.com/support/article/529bf751-3cad-f460-d653-4af162f195da. Accessed: 2025-05-16.

[41] Shaurya Tomer and Gadgets 360 Staff. 2025. *Elon Musk's Starlink to Launch in India With Plans Priced Under Rs.850 Per Month*. https://www.gadgets360.com/internet/news/starlink-india-launch-plans-price-benefits-elon-musk-spacex-report-8510287 Accessed: 2025-06-22.

[42] M. A. Ullah et al. 2025. Impact of Weather on Satellite Communication: Evaluating Starlink Resilience. *arXiv* (2025). https://arxiv.org/abs/2505.04772 Accessed: 2025-05-16.
[43] Aiden Valentine and George Parisis. 2021. Developing and Experimenting with LEO Satellite Constellations in OMNeT++. In *8th OMNeT++ Community Summit*. 1–8. https://arxiv.org/abs/2109.12046
[44] David A. Vallado, Paul Crawford, Richard Hujsak, and T. S. Kelso. 2006. Revisiting Spacetrack Report #3: Rev 1. *AIAA/AAS Astrodynamics Specialist Conference* (2006). https://celestrak.org/publications/AIAA/2006-6753/ AIAA 2006-6753.
[45] Cort J. Willmott and Kenji Matsuura. 2005. Advantages of the Mean Absolute Error (MAE) over the Root Mean Square Error (RMSE) in Assessing Average Model Performance. *Climate Research* 30, 1 (2005), 79–82. doi:10.3354/cr030079
[46] Charlie Wood. 2024. Solar Storm Knocks Out 40 Starlink Satellites in Orbit. https://www.smithsonianmag.com/smart-news/solar-storm-knocks-40-spacex-satellites-out-of-orbit-180979566/. Accessed: 2025-06-18.

A MLab BigQuery SQL Query - Median

```
SELECT
  a.UUID as uuid,
  DATETIME(TIMESTAMP(a.TestTime), "UTC") AS test_time,
  client.Geo.City AS city,
  client.Geo.CountryCode AS country_iso,
  a.LossRate AS packet_loss_rate,
  ROUND(a.MeanThroughputMbps, 5) AS throughput_mbps,
 ROUND((SELECT PERCENTILE_DISC(TCPInfo.RTT, 0.5) OVER()
 FROM UNNEST(m.raw.Download.ServerMeasurements) LIMIT 1)
  / 1000, 5)
  AS download_latency_ms,
 ROUND((SELECT PERCENTILE_DISC(TCPInfo.RTTVar, 0.5) OVER()
 FROM UNNEST(m.raw.Download.ServerMeasurements) LIMIT 1)
  / 1000, 5)
  AS download_jitter_ms,
 ROUND((SELECT PERCENTILE_DISC(TCPInfo.RTT, 0.5) OVER()
 FROM UNNEST(m.raw.Upload.ServerMeasurements) LIMIT 1)
  / 1000, 5)
  AS upload_latency_ms,
 ROUND((SELECT PERCENTILE_DISC(TCPInfo.RTTVar, 0.5) OVER()
 FROM UNNEST(m.raw.Upload.ServerMeasurements) LIMIT 1)
  / 1000, 5)
  AS upload_jitter_ms
FROM measurement-lab.ndt.ndt7 m
WHERE date >= '2025-04-01' AND date <= '2025-05-19'
  AND client.Geo.CountryCode IS NOT NULL
  AND a.MeanThroughputMbps <> 0.0
  AND client.Network.ASNumber = 14593
```

B MLab BigQuery SQL Query - Last Measurement

```
WITH latest_measurements AS (
  SELECT
    a.UUID AS MeasurementUUID,
    MAX(GREATEST(
      IFNULL(download_measurement.TCPInfo.ElapsedTime, 0),
      IFNULL(upload_measurement.TCPInfo.ElapsedTime, 0)
    )) AS MaxElapsedTime
  FROM `measurement-lab.ndt.ndt7`
  LEFT JOIN UNNEST(raw.Download.ServerMeasurements)
  AS download_measurement
  LEFT JOIN UNNEST(raw.Upload.ServerMeasurements)
  AS upload_measurement
  WHERE date >= '2025-04-01' AND date <= '2025-05-19'
    AND client.Network.ASNumber = 14593
  GROUP BY a.UUID
)
SELECT
  a.UUID AS uuid,
  a.TestTime AS test_time,
  client.Geo.City AS city,
  client.Geo.CountryCode AS country_iso,
  a.LossRate AS packet_loss_rate,
  a.MeanThroughputMbps AS throughput_mbps,
  -- Unnesting the raw download measurements
  ROUND(download_measurement.TCPInfo.RTT / 1000, 5)
  AS download_latency_ms,
  ROUND(download_measurement.TCPInfo.RTTVar / 1000, 5)
  AS download_jitter_ms,
  -- Unnesting the raw upload measurements
  ROUND(upload_measurement.TCPInfo.RTT / 1000, 5)
  AS upload_latency_ms,
  ROUND(upload_measurement.TCPInfo.RTTVar / 1000, 5)
  AS upload_jitter_ms,
FROM `measurement-lab.ndt.ndt7`
  LEFT JOIN UNNEST(raw.Download.ServerMeasurements
  ) AS download_measurement
  LEFT JOIN UNNEST(raw.Upload.ServerMeasurements)
  AS upload_measurement
 JOIN latest_measurements lm ON a.UUID = lm.MeasurementUUID
    AND GREATEST(
      IFNULL(download_measurement.TCPInfo.ElapsedTime, 0),
      IFNULL(upload_measurement.TCPInfo.ElapsedTime, 0)
    ) = lm.MaxElapsedTime
WHERE date >= '2025-04-01' AND date <= '2025-05-19'
  AND client.Network.ASNumber = 14593
ORDER BY test_time DESC;
```

C Impact of Hurricane Helene on Atlanta's Network Performance

Hurricane Helene, a Category 4 storm, made landfall in Florida on September 26, 2024, and subsequently moved into Georgia, causing significant damage. Atlanta experienced its heaviest 3-day rainfall totals in 104 years, with 11.12 inches recorded over 48 hours, leading to widespread flooding and power outages [33].

To assess the impact of Hurricane Helene on network performance in Atlanta, we compare two data entries: one from September 4, 2024 (pre-hurricane), and another from September 27, 2024 (during the hurricane).

As shown in Table 5, network performance in Atlanta deteriorated significantly on September 27. The throughput fell sharply from 266.73 Mbps to 0.716 Mbps, while the packet loss rate increased tenfold. The computed Weather Index (WI) rose from 0.1209 to 0.6088, reflecting worsening atmospheric conditions — including cloud cover, wind, and possible rain. We can also conclude that our WI gets very bad only for clouds, reaching high values for hurricanes, but not maximal ones.

Table 5: Network performance in Atlanta before and during the remnants of Hurricane Helene

Metric (Download)	Pre-Storm	During Storm
Test Time (UTC)	06:39	03:16
Packet Loss Rate	1.98%	20.20%
Throughput (Mbps)	266.73	0.716
Loaded Latency (ms)	56.734	224.705
Jitter (ms)	1.609	15.791
Weather Index	0.1209	0.6088

Although WI provides a useful signal for interpreting environmental stress on satellite Internet, the partial degradation seen in this case confirms that WI alone may not capture the full complexity of performance drops — especially when infrastructure or congestion-related factors are involved.

These findings illustrate the substantial impact of Hurricane Helene on Atlanta's network infrastructure, highlighting the vulnerability of communication systems during extreme weather events.

D Weather Index Sensitivity in Tokyo and Toronto

To better understand the correlation between local weather and Starlink network performance, we compare high and low Weather Index (WI) conditions for two major cities: Tokyo and Toronto.

Tokyo (April 2025)

Table 6: Tokyo: Weather Index vs Network Performance

Metric	Good WI, Poor Perf	Bad WI, Poor Perf	Bad WI, Good Perf
Test Time (UTC)	Apr 15, 09:53	Apr 11, 11:42	Apr 11, 12:16
Packet Loss Rate	0.2140	0	0.0024
Throughput (Mbps)	8.07	0.80	114.51
RTT (ms)	197.34	1092.12	44.12
Jitter (ms)	17.64	356.97	1.86
Weather Index	0.1150	0.7373	0.7374

Table 6 reveals that although high Weather Index values often correlate with degraded performance (as seen on April 11, 11:42), **Tokyo's robust infrastructure can still yield excellent throughput even during bad weather** (April 11, 12:16). On the other hand, **low WI is not a guarantee of good performance** either—as demonstrated by the April 15 test, which suffered from severe packet loss despite favorable weather.

This confirms that **Weather Index is a significant factor, but not the only one** influencing Starlink performance in urban areas like Tokyo.

Toronto (April 2025)

Toronto shows a **clear degradation** in both throughput and packet loss as WI increases, matching expectations. The drop in throughput and rise in jitter suggest that **WI is a useful predictor** of performance drops in this location.

These comparisons demonstrate that the **Weather Index is a helpful indicator**, but its impact may vary by region. While Toronto's performance degrades predictably with WI, Tokyo's results show that **non-weather factors** may sometimes dominate, especially in urban or high-traffic environments.

Table 7: Toronto performance under varying Weather Index

Metric	Low WI (Apr 29)	High WI (Apr 2)
Test Time (UTC)	00:33	20:05
Packet Loss Rate	0.0058	0.1944
Throughput (Mbps)	48.62	3.73
Latency (ms)	94.65	60.83
Jitter (ms)	5.51	12.88
Weather Index	0.0415	0.7216

E Justification for the 25% / 38% Outlier Removal Threshold

To robustly evaluate the effect of outlier filtering on our prediction model, we empirically compared multiple outlier removal thresholds using crowdsourced network measurement data collected during the week of May 20$^{\text{th}}$, 2025. The focus of this analysis is to maximize prediction accuracy for **throughput** and **latency**, which are the primary targets of our model, while also reporting on auxiliary metrics such as packet loss and jitter.

We systematically evaluated four plausible thresholds for the percentage of most error-prone samples to remove: 15%, 20%, 25%, 30%, 38%, and 45%. For each threshold, the model was retrained and evaluated using a real-world test set from May 20$^{\text{th}}$, with a temporal matching window of ±1.5 hours to account for network measurement variance.

Table 8: Evaluation metrics for varying outlier removal thresholds (20$^{\text{th}}$ May, ±1.5h window).

Threshold	Metric	MAE ↓	RMSE ↓	R^2 ↑
15%	Throughput	43.30	61.45	0.225
	Latency	15.58	34.84	0.536
20%	Throughput	41.13	58.12	0.245
	Latency	18.28	34.48	0.569
25%	Throughput	41.49	58.63	0.264
	Latency	17.45	34.22	0.573
30%	Throughput	41.77	58.80	0.297
	Latency	17.00	34.35	0.563
38%	Throughput	41.93	58.71	0.337
	Latency	16.00	33.86	0.572
45%	Throughput	42.74	59.63	0.359
	Latency	15.65	34.39	0.552

As shown in Table 8, the 25% and 38% thresholds yields the most balanced and robust results across our target metrics:

- For **throughput**, the 38% setting provides the highest R^2, indicating better variance explanation, with competitive MAE/RMSE.
- For **latency**, 38% yields the lowest RMSE and highest R^2 of all thresholds, confirming strong generalization.

- The **25%** threshold offers a strong compromise, with R^2 values for both throughput (0.264) and latency (0.573) nearly matching the top-performing 38% setting, and with stable MAE and RMSE—demonstrating that substantial robustness can be achieved without excessive data loss.
- While 15% or 45% filtering slightly improves MAE, they lead to worse RMSE or R^2, suggesting overfitting or excessive pruning.

Removing the top 38% of worst samples using composite metrics thus provides the optimal balance between robustness and information retention. This threshold is adopted in all evaluations and final predictions. For further implementation details, see Section 6.

F Weekday–Weekend Comparison

Diurnal patterns in Starlink latency are influenced by day-of-week, reflecting differences in user activity and network load between weekdays and weekends. Figure 6 compares median loaded latency for all cities grouped together, revealing that Sundays (blue) tend to exhibit marginally lower latency than Wednesdays (red) for most hours, especially during late night and early morning periods. This suggests reduced network congestion on weekends.

To illustrate the site-specific nature of this effect, Figure 7 shows the comparison for Paris, France. Here, the Sunday median latency is consistently lower than on Wednesday across the day, most notably during off-peak hours, highlighting the impact of reduced weekend demand even in dense European urban environments.

In contrast, Figure 8 presents results for Antananarivo, Madagascar (Lat: −18.91, Lon: 47.52). At this site, Tuesday (weekday) median latency is lower than Saturday (weekend) for most hours, indicating that weekday/weekend patterns are not universal and may depend on regional usage patterns or local Starlink capacity.

In all panels, shaded regions indicate the interquartile range (IQR) and bars represent the number of measurements per hour. These results emphasize the value of temporal feature engineering for accurate latency modeling, as both hour-of-day and day-of-week effects can meaningfully influence Starlink network performance in a location-dependent manner.

These analyses demonstrate that Starlink latency patterns depend strongly on location, and no universal statement can be made that performance is always better on weekends or weekdays. The aggregate view across all cities (Figure 6) shows only marginal differences between weekdays and weekends, further underscoring that the effect varies by region and context. This highlights the importance of site-specific analysis and careful temporal modeling when assessing or predicting satellite network performance.

G Script Sources

Weather enrichment: https://github.com/TUD-BScResearchProject-6079/leo-viewer/blob/main/backend/src/enrich_with_weather.py

Satellite enrichment: https://github.com/TUD-BScResearchProject-6079/leo-viewer/blob/main/backend/src/enrich_with_satellites.py

Prediction: https://github.com/TUD-BScResearchProject-6079/leo-viewer/blob/main/backend/src/predict.py

H Allowed Hexagons for Predictions

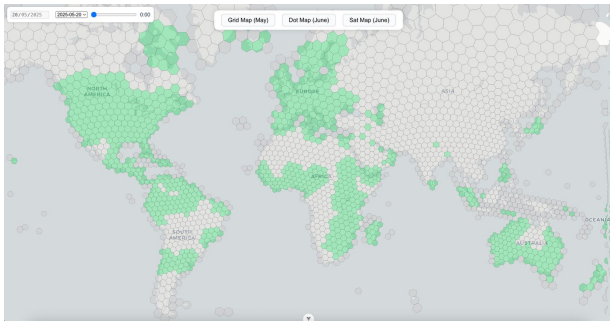

Figure 9: Allowed hexagons are marked with green.

I Interpretation of Map Colors for RTC Quality

Hexagons on all RTC prediction maps are colored according to predicted network quality, using the following thresholds:

Table 9: RTC map color thresholds by predicted metric. Color is assigned by the lowest-quality metric (packet loss, latency, jitter, or throughput). Throughput in Mbps. *If throughput <30 Mbps, color is always set to Dark Red, regardless of other metrics.

	PacketLoss	Latency	Jitter	Throughput
	≤0.005	≤50	≤6	≥150
	>0.005	>50	>6	<150
	>0.015	>70	>11	<100
	>0.03	>90	>18	<70
	>0.05	>110	>25	<50
	>0.07	>130	>35	<40
	>0.10	>150	>45	<30*
	–	–	–	<15

J Evaluation Table 38% anomaly removal

Table 10: Model performance on May 20th (Weekday) and May 24th (Weekend)

Date	Metric	MAE	RMSE	R^2
20 May	Packet Loss	0.0253	0.0355	0.0590
	Download Jitter	2.25 ms	7.29 ms	0.0114
	Loaded Latency	16.0 ms	33.9 ms	0.5723
	Throughput	41.9 Mbps	58.7 Mbps	0.3371
24 May	Packet Loss	0.0254	0.0384	0.0352
	Download Jitter	4.30 ms	41.9 ms	0.0025
	Loaded Latency	20.2 ms	79.2 ms	0.2136
	Throughput	38.7 Mbps	53.7 Mbps	0.3595

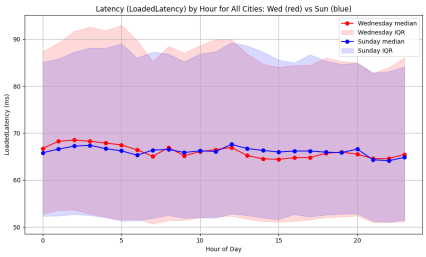

Figure 6: Diurnal loaded latency for all cities (Wednesday vs Sunday). Sunday latency (blue) is marginally lower than Wednesday (red), especially during night and early morning, indicating reduced weekend congestion.

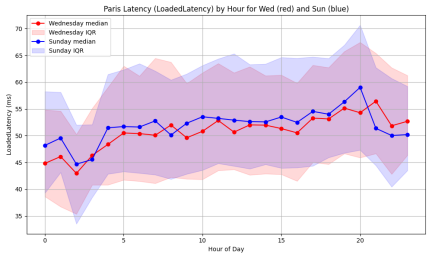

Figure 7: Paris, France: Sunday (blue) median latency is consistently lower than Wednesday (red), especially in off-peak hours, showing a clear weekend benefit in urban Europe.

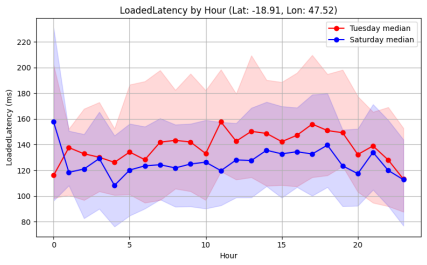

Figure 8: Antananarivo, Madagascar: Tuesday (red) median latency is mostly lower than Saturday (blue), suggesting that weekday/weekend latency patterns vary regionally.

Table 11: OpenWeatherMap (OWM) weather codes to cloudiness mapping.

OWM Code(s)	Cloudiness Value
800	0.0 (Clear sky)
801	0.1 (Few clouds)
802	0.5 (Scattered clouds)
803, 804	0.9 (Broken/overcast clouds)
700–781	0.7 (Mist, smoke, haze, etc.)
300–321	0.6 (Drizzle)
500, 520, 521	0.6 (Light rain)
502–531	0.8 (Heavier rain)
600, 601	0.85 (Light snow)
602	0.95 (Heavy snow)
611–622	0.75 (Sleet/freezing rain)
200–232	1.0 (Thunderstorm)
≥900	1.0 (Extreme)
Otherwise	0.1

Table 12: COCO-to-cloudiness mapping, with concise reasoning. Based on official codes [3].

COCO	Condition	Reasoning
1	Clear	0.0 (No clouds)
2	Fair	0.1 (Few clouds)
3	Cloudy	0.5 (Partly cloudy)
4	Overcast	0.9 (Nearly full cover)
5	Fog	0.7 (Cloud-like effect)
6	Freezing Fog	0.7 (Same as above)
7	Light Rain	0.6 (Rain, not full cover)
8	Rain	0.6 (Same as above)
9	Heavy Rain	0.8 (Likely overcast)
10	Freezing Rain	0.8 (Likely overcast)
11	Heavy Freezing Rain	0.8 (Likely overcast)
12	Sleet	0.75 (Usually overcast)
13	Heavy Sleet	0.75 (Same as above)
14	Light Snowfall	0.85 (Snow = full cover)
15	Snowfall	0.85 (Same as above)
16	Heavy Snowfall	0.95 (Extreme cover)
17	Rain Shower	0.6 (Short, still cloudy)
18	Heavy Rain Shower	0.6 (Same as above)
19	Sleet Shower	0.75 (Short, still cloudy)
20	Heavy Sleet Shower	0.75 (Same as above)
21	Snow Shower	0.85 (Snow = full cover)
22	Heavy Snow Shower	0.85 (Same as above)
23	Lightning	1.0 (Thunderstorm)
24	Hail	1.0 (Thunderstorm)
25	Thunderstorm	1.0 (Full cloud cover)
26	Heavy Thunderstorm	1.0 (Same as above)
27	Storm	1.0 (Full cloud cover)

K Cloudiness Mapping Comparison

OpenWeatherMap Code-to-Cloudiness Mapping (Prediction Phase)

Explanation and Rationale for Meteostat COCO-to-Cloudiness Mapping

Table 12 lists each COCO weather code (per [3]) alongside our assigned cloudiness value and a brief explanation.

Note: Both mappings use custom groupings to convert discrete weather categories into a continuous cloudiness metric in [0, 1]. The COCO codes correspond to weather phenomena defined in the Meteostat dataset, whereas OWM codes are from OpenWeatherMap. For details, see the implementation in our training and backend scripts.

Note on Mapping Consistency

Both the OpenWeatherMap (OWM) and Meteostat COCO mappings translate discrete weather categories (codes) into a continuous cloudiness value in [0, 1], tailored for our Weather Index formula.

While the underlying weather phenomena may differ slightly between datasets, both mappings follow similar logic: - Clear/fair weather maps to low cloudiness, - Rain, snow, fog, and storms to high cloudiness, - Transitional/ambiguous categories are given intermediate values.

Empirical comparison: In practice, predicted maps generated using either mapping (on identical test dates and locations) exhibit negligible visual and statistical differences. The largest sources of prediction variance stem from actual weather, not the mapping. For more details and code, see our repository and backend scripts.

For a full mapping table and explanations, see Appendix 12.

L Evaluation Formulas

- **Mean Absolute Error (MAE):**

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

  where y_i is the ground truth value for the i-th prediction, $\hat{y}_i$ is the predicted value, and n is the total number of samples. MAE is measured in the same units as the target variable (e.g., ms, Mbps) and provides a straightforward interpretation of average error.

- **Root Mean Squared Error (RMSE):**

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

  where y_i is the ground truth value, $\hat{y}_i$ is the prediction, and n is the total number of samples. RMSE, also in the same units as the target, emphasizes larger errors and is sensitive to outliers.

- **Coefficient of Determination (R^2):**

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

  where y_i is the ground truth, $\hat{y}_i$ is the predicted value, $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ is the mean of the ground truth values, and n is the number of samples. R^2 is unitless and ranges from $-\infty$ to 1, with higher values indicating better model fit.

For further details, see [10, 13, 19, 21, 45].