Machine Learning-Based Predictive Modeling for 4G Long Term Evolution (LTE) Traffic Prediction

Snehil Sharma

School of Computer Engineering,
KIIT Deemed to be University,
Bhubaneswar 751024, India;
2005545@kiit.ac.in

Abstract

Smartphones are increasingly using artificial intelligence (AI), which is based on machine learning (ML). ML is also being used in the Edge paradigm of next-generation networks (NGNs) to predict and optimize network load, which is growing rapidly due to human traffic. Both standard and deep ML techniques are being used to improve NGN operation in complex heterogeneous environments. This paper proposes a method to predict traffic on the LTE network edge using the ML techniques Random Forest, Bagging, Support Vector Machines (SVMs), Bayes algorithm, Tweedie Regressor, RANSAC Regressor and Huber Regressor. We developed a corresponding ML environment based on a public cellular traffic dataset and compared the quality metrics and algorithm running time for each model. The SVM method allows the model to train much faster than the bagging and random forest algorithms. While bagging and random forest operate well with a mixture of numerical and categorical features, the SVM requires scaling the dataset and has difficulty finding nonlinear dependencies in the data.

Keywords – Machine Learning, 4G, Random Forest, Bagging, Support Vector Machines, Bayes algorithm, Tweedie Regressor, RANSAC Regressor, Huber Regressor, traffic analysis, optimization, LTE, NGN

I. Introduction

The inevitability of modern technology's integration into our daily lives is a global phenomenon driven by computerization. In the course of technological evolution, one key notion emerged: the idea that computers, if capable of learning from their experiences and autonomously enhancing program efficiency, could significantly boost their utility by the end of the 20^{th} century. Today, researchers have effectively integrated Artificial Intelligence (AI) into various human activities. According to the Blumberg Capital Survey in 2019, 50% of companies had implemented AI in their business operations, and approximately 26% of regular users claimed to use AI at least once daily. Another recent study by McKinsey Global Survey asserted that around 63% of companies that implemented AI reported increased revenues, making AI a pivotal domain for both industry and academia [1].

It's crucial to recognize that the foundation of AI lies in Machine Learning (ML) techniques. ML represents an approach to crafting intelligent systems, and the accomplishments of ML algorithms have driven the current surge in AI development. It is now challenging to identify any area within modern technology that doesn't leverage ML methods [2].

In the context of telecommunications, Next Generation Networks (NGN) are inevitably confronted with the challenge of accommodating a growing number of users and devices, leading to an extraordinary surge in mobile network traffic. This surge results in an increased volume of requests for limited network resources and adds complexity to the system's architecture. Managing such vast data using conventional network planning methods becomes a daunting task, particularly at the network Edge, which is as close as possible to the user equipment within the network architecture [3].

ML technology emerges as one of the modern methods poised to handle these vast data streams. ML algorithms aim to discover optimal solutions for specific problems through the sophisticated analysis of statistical data. They are expected to offer higher-level, smarter surveillance, network and application management, and the ability to identify elusive data patterns for future use that would be challenging to detect manually. These methods are the primary focus of this research.

This study's objective is to forecast traffic in the LTE network using ML techniques like Random Forest, Bagging, and Support Vector Machines (SVM). The dataset, "Predict traffic of LTE network," [4] employed in this study, was sourced from Kaggle.com and covers a one-year data collection period. During this time, mobile devices were served by nearby 4G cells whenever a subscriber used a mobile data service. The dataset contains information from 57 cells, and the Python 3.11 programming language, in conjunction with VScode software, was used to model the prediction algorithm. Data analysis and visualization were facilitated by libraries such as NumPy, Pandas, Scikit-learn, Scipy, Matplotlib, and Seaborn.

The remainder of this paper is structured as follows: Section II provides a concise overview of the models and metrics chosen for this study. Section III delves into the essential techniques required for data preprocessing. Section IV outlines the strategies employed for traffic prediction. Section V details the results achieved with each of the applied models, encompassing quality metrics and algorithm execution times. The final section concludes the paper and outlines future directions for research.

II. Models and Metrics

Machine Learning (ML) is a branch of computer science focused on harnessing algorithms to process vast datasets automatically and enhance efficiency through learning from experience [5]. ML essentially comprises two components: classical and deep learning. The former encompasses Linear Models, such as Linear Regression, Logistic Regression, SVM, and Ensemble Models, which are founded on algorithms like Bagging and Gradient Boosting. It's worth mentioning that ML methods trace their origins to the latter half of the 20^{th} century but didn't find practical application until the early 21^{st} century, largely due to computational limitations. The second category, deep learning, marked a significant breakthrough in 2012, led by George E. Dahl's team's [6] triumph in the Merck Molecular Activity Challenge. This category employs various neural networks, primarily to uncover more intricate patterns within data.

The subsequent section outlines the chosen prediction models: random forest, bagging, and SVM. Following that, definitions of quality metrics are provided to identify the model that yields the most favorable results.

A. Applied Models

In this study, we employ traditional machine learning models to forecast LTE network traffic. We utilize two ensemble models, one being a Random Forest that doesn't necessitate specific hyperparameter selections, and Bagging, which does entail this process. Additionally, we incorporate a linear model to gain insights into the linearity of data dependencies, particularly when compared to the previous models.

a) Random Forest: The random forest employs a stochastic method for building decision trees. Each tree is trained on a random selection of data points and a random subset of features, a technique known as the random subspace method. Subsequently, based on the outcome obtained for each tree, predictions can be generated. This result can be aggregated in multiple ways, such as utilizing a rapid majority vote, as originally introduced in the pioneering work by J.S. Moore [7]. This random approach enables the potential reduction of model errors, specifically in diminishing the variability in model predictions.

b) Bagging: Bagging and Random Forest are closely related algorithms. In bagging, we train N distinct decision trees on subsamples drawn with replacement (bootstrap), while utilizing the entire set of features concurrently. In bagging, each tree learns independently without being aware of the outcomes of other trees. Subsequently, as outlined in the Random Forest algorithm, we derive a prediction for each data point based on the results obtained from each individual tree.

c) Support Vector Machines: Support Vector Machines (SVMs) operate by maximizing the margin from the separating plane. In both regression and classification tasks, the model relies on reference objects situated at the class boundary. The solution is not founded on the objects themselves but on the dot products between objects, and it also permits the utilization of kernels. These techniques enable us to train the model considerably faster than the methods upon which Bagging and Random Forest are built.

The subsequent section delves into our metrics, which serve as qualitative measures for assessing the models.

B. Metrics Used

In this study, we characterize metrics as measures, whether in terms of quality or quantity, that provide insights into a specific process. In this section, we have introduced three chosen metrics: Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Coefficient of Determination (R^2)

a) Root Mean Square Error: RMSE serves as a comparative gauge of how effectively the model aligns with predicted variables. It is computed as

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

Here, y_i denotes the actual output, $\hat{y}_i$ represents the predicted output, and N stands for the total number of variables. Essentially, it signifies the mean of the squared prediction errors, which are the differences between the actual and predicted outputs. A higher value corresponds to a greater prediction error.

b) Mean Absolute Error: MAE shares similarities with RMSE, but it involves summing the absolute values, as shown by the formula

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{2}$$

This formula illustrates the mean across the dataset of the absolute disparities between predictions and actual observations. MAE can vary from 0 to ∞, and a lower value indicates a superior performance of the testing model.

c) Coefficient of Determination: The coefficient of determination (R^2) enables us to assess the extent to which the algorithm can predict the variance from the overall variance in the data. This is defined as:

$$R^2 = 1 - \left(\frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2} \right) \tag{3}$$

In this equation, y_i represents the actual output, $\hat{y}_i$ is the predicted output, and $\bar{y}$ is the average of the observed data.

R^2 is the fraction of the variance in the predicted variable, ranging from 0% to 100%. In other words, a higher percentage signifies a stronger correlation between the predicted and observed variables. An R^2 value of 100% indicates that the model accounts for all the variance in the data.

The following section outlines various data preprocessing methods necessary for accurately computing the algorithm.

III. Data Preparation Techniques

Data intended for analysis should be in a format compatible with the ML application. It can take the form of numerical data (such as dates and times) or categorical data (representing categories). Before applying the algorithm model, it is essential to convert them into the necessary format for processing, which might involve encoding them as binary values. Data preprocessing is imperative to ensure the algorithm functions correctly. This section discusses techniques like One-Hot Encoding (OHE) and Min-Max Scaling.

a) One-Hot Encoding: As mentioned earlier, data can exist in categorical (nominal) or numerical forms. Some algorithms, like XGBoost or Catboost, are capable of handling categorical data, whereas most ML algorithms primarily operate with numerical data.

One approach for representing categorical data in numerical form is through the use of the One-Hot Encoding (OHE) technique. This technique assigns a binary number to each category. It places a 1 in the binary variable corresponding to the category and 0 values for the others. For instance, if the data includes the category "Devices" with labels "*laptop*," "*computer*," and "*cellphone*," each unique category value is mapped to a binary number – "*laptop*" becomes 100, "*computer*" is represented as 010 and "*cellphone*" is 001.

The number of binary variables needed depends on the number of categories. For example, if there are three labels for "Device," it means there are three binary variables. However, OHE becomes impractical with a large number of labels, as it requires a substantial number of binary representations.

b) Min-Max Scaling: Min-Max Scaling is a method for standardizing input variables. It is used when input data is measured on various scales, as this discrepancy can introduce bias in the modeling process. The formula for Min-Max scaling is as follows

$$x_s = \left(\frac{x - min(x)}{max(x) - min(x)} \right), \tag{4}$$

In this equation, x_s represents the scaled variable, while x stands for the input variable. The terms $min(x)$ and $max(x)$ refer to the minimum and maximum values of all input variables, respectively.

Sections IV and V emphasize the selected models, metrics, and data preparation methods. The subsequent section focuses on making traffic predictions using the chosen algorithms.

IV. Models For Traffic Prediction in the Network

Typically, the construction of a model involves several phases: data preprocessing, data examination, model implementation, and result analysis. The subsequent sections elaborate on each of these stages.

A. Data Preparation

a) Gaps: In practice, certain data can sometimes become unavailable or be omitted from the record when it is received. Various factors can lead to gaps in the collected infocommunication data, such as system issues, packet loss, interference, and more. However, it's important to note that the data used for analysis in this study is considered *clean*, meaning it is devoid of any gaps or missing information.

b) Categorical features: The algorithms employed in this study do not accept nominal (categorical) variables. Therefore, it's essential to convert all attributes into a numeric format. The *Hour* variable, which denotes the time the traffic initially arrived, is already in numeric format. However, the *Date* column is currently in string format, so it is divided into three separate columns: *Month*, *Day*, and *Year* with each of them being converted into a numeric format. Consequently, the original *Date* string column is discarded. This results in an expansion of our feature space by two additional columns. The *CellName* column is transformed using the One-Hot Encoding (OHE) method to accommodate the algorithm's requirements.

c) Feature transformation: This study applies a linear SVM algorithm. For linear models, it is essential to normalize the data. To achieve this, the *Month* and *Day* features were scaled to the interval [0,1] through min-max normalization. Furthermore, the accuracy of linear models can be enhanced when the features are distributed in a manner similar to the normal distribution. Initially, the traffic data followed an exponential distribution, but after transforming it by taking the natural logarithm, i.e., assuming that $Traffic_j = ln(Traffic_j)$, the data now conforms to a normal distribution.

The temporal variation in the average traffic for all cells was thoroughly examined, as depicted in the figure. It's evident that this dependence, albeit with some errors, exhibits a sinusoidal pattern. Drawing on these observations, the *Hour* feature was encoded by transforming it as follows

$$Hour_j = \frac{\sin(Hour_j)}{T} \tag{5}$$

where h_j represents the day of the month, and T is set to 30.

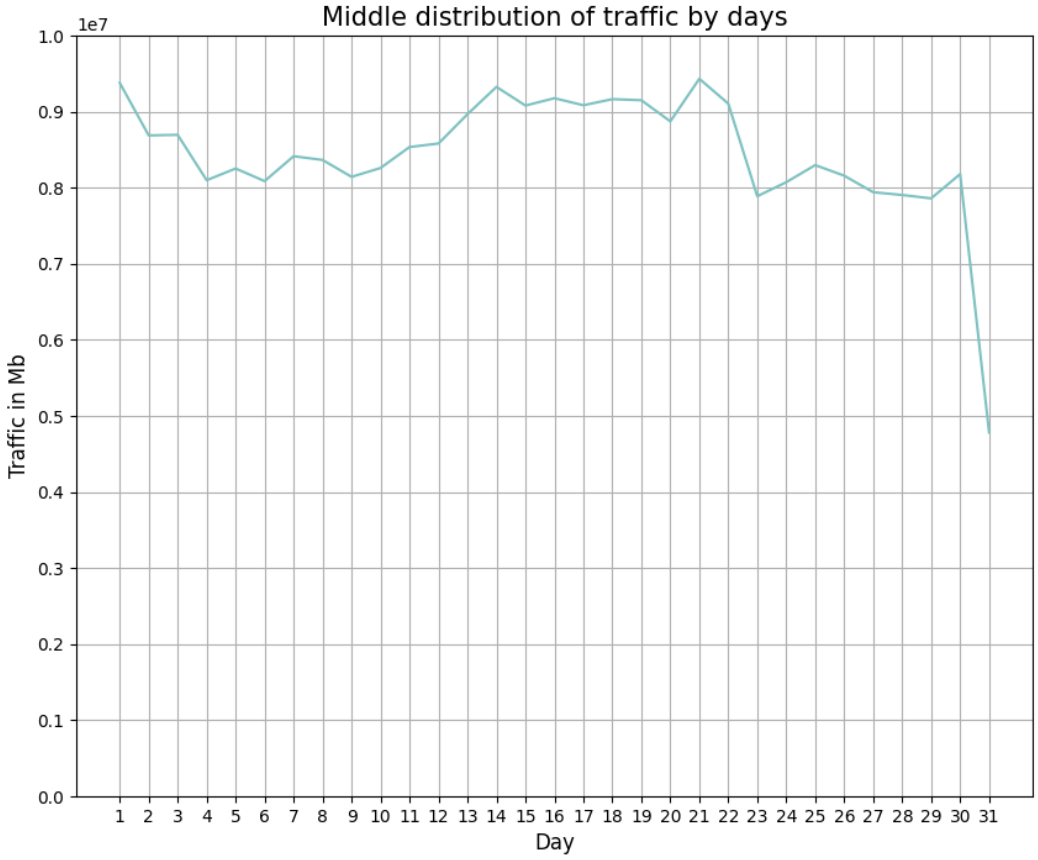

Fig. 1. The spread of total data usage on a particular day of the month from all users

B. Data Analysis

Examining the visual representation of the data was crucial to assess the data distributions. This analysis allowed us to explore the relationship between the target variable and the characteristics, assess this relationship, identify potential correlations between the characteristics, and so forth.

a) Three groups of the cells were identified: The cells are categorized into three groups: yellow cells, representing high traffic (10% of all cells); red cells, indicating low traffic (another 10% of all cells); and bluish green cells, which account for the majority (80% of all cells) and have moderate traffic levels. Some of these cells are depicted in Figure 2. This visual representation allows us to make several inferences. For instance, it suggests that the yellow cells are likely situated in the

city center or areas where large crowds gather, possibly during public events. Additionally, it underscores the importance of maintaining robust control over the yellow cells, as their disruption could result in a significant number of users being disconnected from the network.

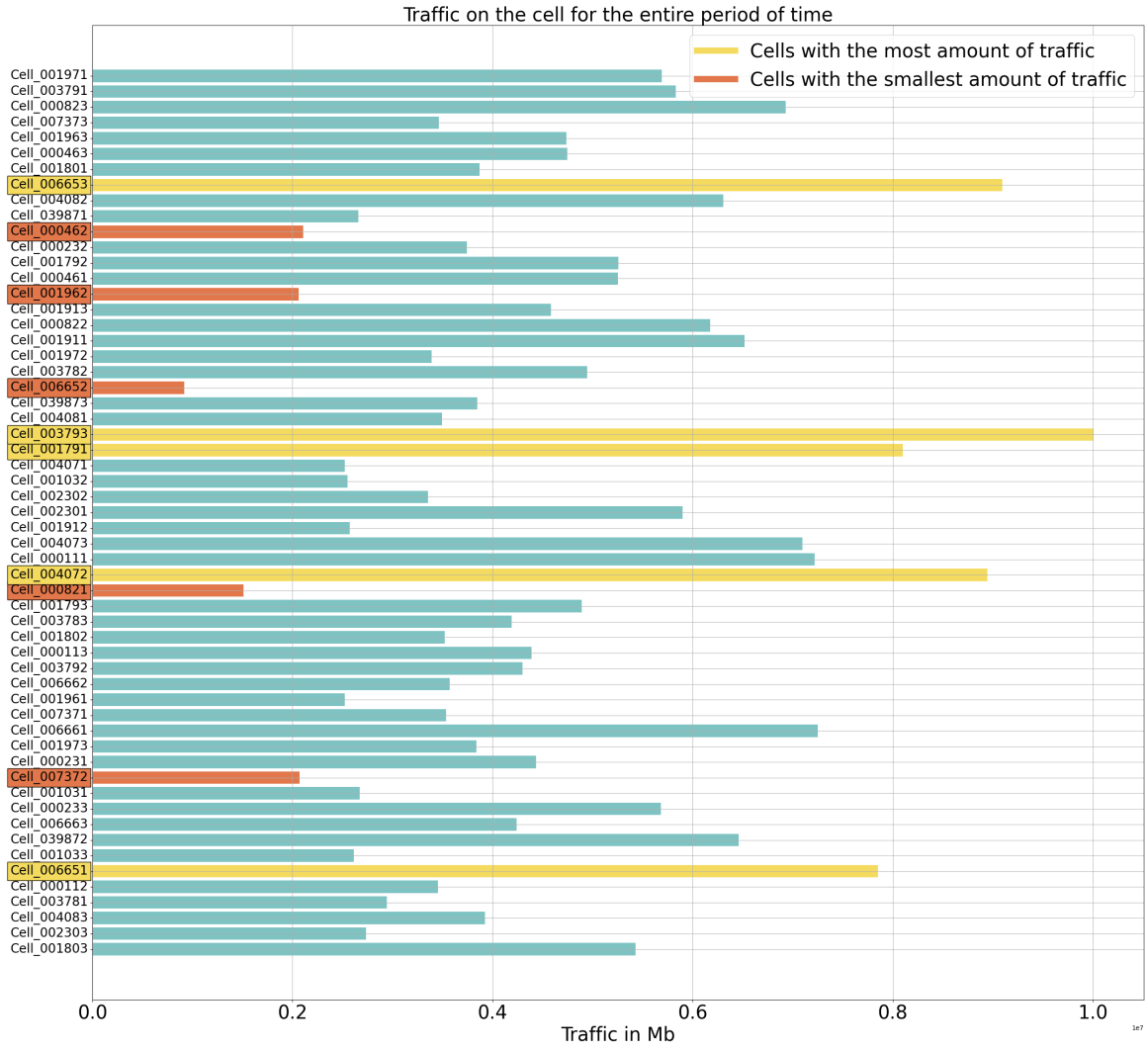

Fig. 2. Traffic from a subset of cells over the entire time span

b) Correlation: To continue working with the features and creating new ones, it's essential to evaluate their correlations among themselves and with the target variable. To accomplish this, a heatmap was generated and is displayed in the figure. Notably, the absolute correlation between the *Month* and *Year* features is very high, which is expected, as the data contains only three months in 2017 and ten months in 2018. This represents an unnecessary correlation, and it was decided to eliminate this dependency by employing One-Hot Encoding (OHE) for the *Year* feature.

V. Result Analysis

In this study, the modeling was conducted within the following environments: Python 3.11 programming language and the VScode software were utilized for building the prediction algorithm. Additionally, various libraries for data analysis and visualization, including Numpy, Pandas, Scikit-learn, Scipy, Matplotlib, and Seaborn, were employed.

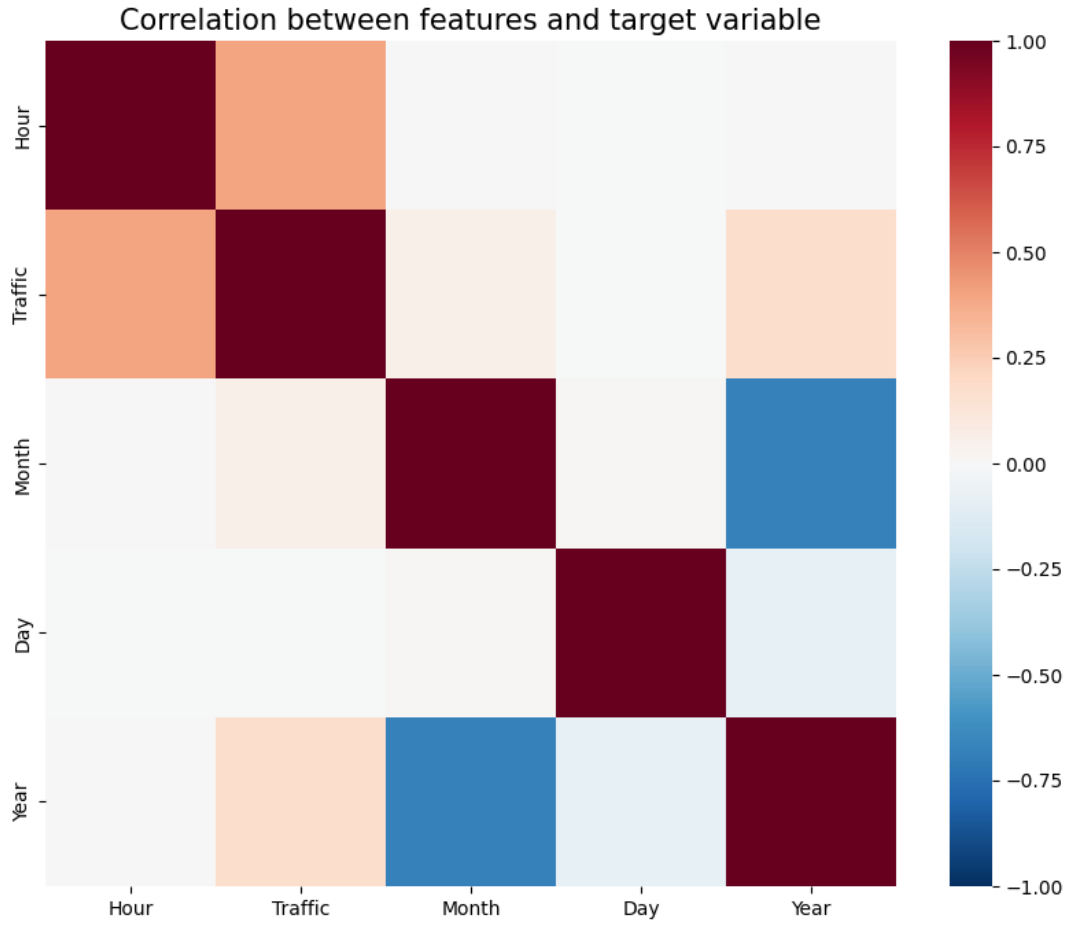

Fig. 3. Correlation between features and target variable

The problem is framed within the domain of machine learning, and the outcomes of the applied algorithms are presented in the table. The problem is characterized by the following:

1) Problem type: Regression
2) Target Variable: Traffic for a specific cellular cell.
3) Features used for prediction:
   - Time of day ($Hour$), month ($Month$) and day (Day)
   - Binary features representing cellular communication cells ($CellName$) and binary features for the $Year$ (2017 and 2018)
4) Evaluation Metrics: RMSE, MAE and R^2.

For the sake of comparison, it's important to note that in both MAE and RMSE metrics, an ideal algorithm would achieve a score of zero. Additionally, in the coefficient of determination R^2, a perfect model would attain a value of 100%.

Table 1 provides a summary of the results, from which several conclusions can be drawn. Support vector machines exhibit slightly inferior performance compared to ensemble methods and necessitate extra preprocessing through standardization. However, they require less training time. As a result, the debate regarding what is more important - superior quality or a quicker learning pace - remains unanswered. There are instances where having a swift model and the ability to train it online to rectify errors may be preferable.

Among all the nonlinear models, Bagging demonstrated the best results, albeit with a longer training time (116 seconds).

TABLE I
PERFORMANCE METRICS OF DIFFERENT ALGORITHMS

	Bagging	**Random Forest**	**SVM for Regression**	**Bayes Algorithm**	**Tweedie Regressor**	**RANSAC Regressor**	**Huber**
RMSE [Mbit]	1.27	1.47	1.30	1.66	1.55	1.51	1.68
MAE [Mbit]	0.97	1.15	0.98	0.99	1.21	1.15	0.98
R^2 [%]	50.9	34.1	48.7	49.7	26.7	30.4	49.1
Wall time of learning [s]	19.1	54.1	3.3	2.4	0.5	5.0	11.7

On the other hand, Random Forest yielded the least favorable results, but it's important not to dismiss it entirely. Its performance is closely tied to the number of features, and our dataset contains a relatively small number of them (57 features, with only 2 being non-categorical). This might not be sufficient for Random Forest to shine. Therefore, by addressing the challenge of generating new features, Random Forest's metrics could be enhanced.

The hyperparameter selection process for Bagging is a rather labor-intensive task. It's worth noting that for the Support Vector Machine, optimal parameters were efficiently chosen using Grid Search in a matter of seconds. However, this wasn't feasible for Bagging due to the considerable time it consumes. This is particularly relevant in the context of Next Generation Networks, where efficiency and speed are paramount. Grid Search is an approach to fine-tuning parameters that systematically evaluates and builds models for each combination of algorithm parameters specified in a grid [8].

VI. CONCLUSION

This study has demonstrated the feasibility of extracting predictive insights from even basic traffic-related data using machine learning techniques to address optimization challenges within infocommunication systems, particularly from the perspective of traffic prediction. The primary focus was on predicting traffic for LTE edge operations, using a dataset derived from 57 cells.

The algorithms employed in this study operate independently from the broader system. An ongoing challenge is ensuring that algorithms don't work in isolation from the network. One proposal is to introduce an additional layer where problems are resolved through a consensus mechanism among algorithms when they have differing opinions. When there's agreement among the algorithms, the data is accepted in its original form.

Seven algorithms were showcased for solving the LTE network traffic prediction problem: Bagging, Random Forest, Support Vector Machines (SVM), Bayes algorithm, Tweedie Regressor, RANSAC Regressor and Huber Regressor. Emphasis was placed on three key aspects: initial data preprocessing, visualization, and prediction.

Based on the obtained results, several important conclusions can be drawn regarding the practical application of machine learning algorithms for traffic prediction. The data that algorithms receive as input should either be rapidly preprocessed before prediction or separated into a distinct part of the system. This separation ensures the data quality for the algorithm while eliminating the possibility of real-time model training.

The second point underscores the significance of interpreting data to enhance the machine learning process. Machine learning algorithms often operate as black boxes, relying on mathematical transformations that may not always be comprehensible to humans. Data interpretation aids in understanding how the algorithm is likely to work and identifying data patterns before obtaining model results.

The third point is of paramount importance. It highlights that algorithms sourced from various programming language libraries may not provide excellent results immediately. They necessitate hyperparameter tuning tailored to the specific task and the creation of new features to better capture the true data relationships. It's crucial to recognize that model training is not a swift process. Therefore, it's essential to assess feature importance and choose the appropriate hyperparameters to strike a balance between quality and speed, a critical factor in Next Generation Networks (NGN).

The work presented in this paper holds promising prospects for future implementation. These include training Bagging with Grid Search to enhance prediction quality, implementing and comparing other machine learning algorithms, exploring more details and the impact of additional features (e.g., alternative CellName encoding approaches, feature space transformations, and feature generation), combining various data scaling methods, such as standardization to potentially improve the linear SVM algorithm, and conducting separate training for cells with high, medium, and low total traffic values, respectively, to enhance prediction quality for each group. Furthermore, a specific research question to be addressed is whether it is possible to tackle the inverse problem - classifying cells based on time, date, and incoming traffic. This could be beneficial in scenarios where quick identification of service radio stations is necessary.

REFERENCES

[1] Z. Ullah, F. Al-Turjman, L. Mostarda, and R. Gagliardi, "Applications of Artificial Intelligence and Machine Learning in Smart Cities," Computer Communications, 2020.

[2] E. Alpaydin, Introduction to Machine Learning. MIT press, 2020.

[3] N. Makitalo, A. Ometov, J. Kannisto, S. Andreev, Y. Koucheryavy, and T. Mikkonen, "Safe, Secure Executions at the Network Edge: Coordinating Cloud, Edge, and Fog Computing," IEEE Software, vol. 35, no. 1, pp. 30–37, 2017.

[4] "Predict traffic of LTE network," [Online] https://www.kaggle.com/naebolo/predict-traffic-of-lte-network, (accessed on October 16, 2023).

[5] L. Zhang, J. Tan, D. Han, and H. Zhu, "From Machine Learning to Deep Learning: Progress in Machine Intelligence for Rational Drug Discovery," Drug Discovery Today, vol. 22, no. 11, pp. 1680–1685, 2017.

[6] J. Ma, R. P. Sheridan, A. Liaw, G. E. Dahl, and V. Svetnik, "Deep Neural Nets as a Method for Quantitative Structure–Activity Relationships," Journal of Chemical Information and Modeling, vol. 55, no. 2, pp. 263–274, 2015.

[7] J. S. Moore, "A Fast Majority Vote Algorithm," Automated Reasoning: Essays in Honor of Woody Bledsoe, 1981.

[8] M. Mohri, A. Rostamizadeh, and A. Talwalkar, Foundations of Machine Learning. MIT press, 2018.