# Applying Machine Learning to LTE Traffic Prediction: Comparison of Bagging, Random Forest, and SVM

Nikolai Stepanov[†], Daria Alekseeva[‡], Aleksandr Ometov[‡], Elena Simona Lohan[‡]
[†]National Research University Higher School of Economics, Moscow, Russia
[‡]Tampere University, Tampere, Finland
Authors emails: ndstepanov@edu.hse.ru, {name.surname}@tuni.fi

***Abstract*—Today, a significant share of smartphone applications use Artificial Intelligence (AI) elements that, in turn, are based on Machine Learning (ML) principles. Particularly, ML is also applied to the Edge paradigm aiming to predict and optimize the network load conventionally caused by human-based traffic, which is growing each year rapidly. The application of both standard and deep ML techniques is expected to improve the networks' operation in the most complex heterogeneous environment. In this work, we propose a method to predict the LTE network edge traffic by utilizing various ML techniques. The analysis is based on the public cellular traffic dataset, and it presents a comparison of the quality metrics. The Support Vector Machines method allows much faster training than the Bagging and Random Forest that operate well with a mixture of numerical and categorical features.**

***Index Terms*—Machine Learning, traffic analysis, LTE, NGN, optimization**

## I. Introduction

The penetration of modern technology in everyday life is inevitable and globally driven by computerization. At the end of the twentieth century, researchers came up with the idea that if computers could learn from their experience and automatically improve their programs' efficiency, their usefulness would increase [1]. Nowadays, Artificial Intelligence (AI) is already successfully embedded in various human activities [2]–[5]. According to Blumberg Capital Survey, in 2019, 50% of companies implemented AI in their business, and about 26% of ordinary users claim that they are already using AI [6]. Another recent study by McKinsey Global Survey claims that about 63% of the companies where AI has been deployed report revenue increases from it [7], thus, making it a significant research field for both industry and academia.

It is worth noting that the concept of AI is based on Machine Learning (ML) techniques [8]. ML is an approach to the design of intelligent systems, and the success of ML algorithms leads to present growth in AI [9]. It has become difficult to imagine an area in modern technology that lacks ML methods [10].

From the telecommunications perspective, Next Generation Networks (NGN) beyond 5G will inevitably face the challenge of a growing number of users and devices, and, as a result, of the extraordinary growth of mobile network traffic [11], [12]. Such growth results in a higher number of requests to limited network resources and to the increase of the systems architecture complexity. Managing such a large amount of data using conventional network planning methods becomes a problematic task [13], especially for the operation on the network edge, i.e., as close to the user equipment on the network architecture level as possible [14].

Generally, ML technology is one of the modern methods expected to allow for the control of large information flows. The ML algorithm will attempt to find the best solution to a specific problem through the sophisticated use of statistical data [10]. It is expected to provide a higher and smarter level of surveillance, network, and application management. It will also allow finding new patterns in the data to use them in the future that are difficult to detect manually. These methods are the main object of research in this work.

The study's main aim is to predict the traffic in the LTE network utilizing such ML techniques as Random Forest, Bagging, and Support Vector Machines (SVM). The dataset "Predict traffic of LTE network" utilized in this work is obtained from kaggle.com [15]. Data collection took place during one year. When a subscriber used a mobile data service, the mobile device was served by a nearby cell. The data set has information from 57 cells. Programming language Python 3.7 and Jupyter notebook software were applied for modeling the prediction algorithm. Libraries for the data analysis and visualization are NumPy, Pandas, Scikit-learn, Scipy, matplotlib, seaborn.

The rest of this paper is organized as follows. Section II provides a brief overview of models and metrics that were selected for this study. Section III examines the main techniques required for the data preprocessing. Section IV presents the strategies utilized for traffic prediction. Section V describes the results obtained with each of the applied models, i.e., quality metrics and algorithm running time. The last section concludes the paper and provides future work.

## II. Selected Models and Metrics

ML is a field of study in computer science aimed at using algorithms for massive data set processing to improve efficiency through the experience automatically. Essentially, ML has two parts – classic and deep learning [16]. The first includes Linear Models, such as Linear Regression, Logistic Regression, SVM, and Ensemble Models (based on such algorithms as Bagging and Gradient Boosting). It should be noted that ML methods have their roots in the second half of the $20^{th}$ century, but due to the lack of computing power, they did not find their application until the beginning of the $21^{st}$ century [17]. A second group, deep learning, was a $2012^{th}$ breakthrough followed by George E. Dahl's team's victory in the Merck Molecular Activity Challenge. This group utilizes various neural networks that are mainly used to find more complex relationships in the data [18].

The next subsection describes the selected forecasting models: Random Forest, Bagging, and SVM. Then, definitions of quality metrics are given to determine the model achieving the best results.

### *A. Applied Models*

In this work, we use classical ML models to predict traffic in LTE network: two ensemble models, one of which is a Random Forest that does not require special choices of hyperparameters and Bagging, which requires this procedure, and one linear model, in order to understand, comparing with previous models, how linear the dependencies in the data are.

*a) Random Forest:* The Random Forest uses a random approach to constructing decision trees: each tree is trained on randomly selected objects and randomly selected features, the so-called random subspace method. Then, based on the result obtained for each tree, a prediction can be obtained. The result can be formed in various ways, such as using a fast majority vote, known from trailblazing work by J.S. Moore [19]. Using such a random approach, it is feasible to reduce the model's error, namely, to reduce the spread of model predictions.

*b) Bagging:* Bagging and Random Forest are very similar algorithms. In the case of Bagging, we train $N$ independent decision trees on subsamples with return (bootstrap) but using the entire feature data space simultaneously. In a Bagging, each tree learns without knowing what results in another tree are provided. Then, as described in the Random Forest algorithm, we obtain an answer at each object based on the result from each tree.

*c) Support Vector Machines:* SVMs are based on maximizing the distance from the dividing plane. For the regression and classification problem, the model uses reference objects (located on the class boundary) [20]. The solution is not based on objects but on the dot products of objects, and it is permissible to use kernels. These approaches allow us to train the model much faster than the approaches that Bagging and Random Forest are based on.

The next subsection discusses our metrics that are qualitative indicators for evaluating models.

### *B. Used Metrics*

In this work, we define metrics as quality or quantity indexes, which give information about a particular process. This subsection presented three selected ones: Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Coefficient of Determination ($R^2$).

*a) Root Mean Square Error:* RMSE is a relative measure of how well the model fits predicted variables. It is calculated as

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

where $y_i$ is a real output, $\hat{y}_i$ is a predicted output, $N$ is the number of variables. Generally, it represents the average of the square of the prediction error (the difference between real and predicted output), i.e., the larger number corresponds to the larger error.

*b) Mean Absolute Error:* MAE is similar to the RMSE, but it is taking the sum of the absolute value, such that

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

which shows the average over the data set of the absolute differences between prediction and real observation. MAE can range from 0 to infinity, and the lower value corresponds with the best testing model.

*c) Coefficient of Determination:* The coefficient of determination ($R^2$) allows estimating how much of the variance the algorithm could predict from the total variance in the data, which is defined as

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

where $y_i$ is a real output, $\hat{y}_i$ is a predicted output, $\bar{y}$ is the mean of the observed data.

The $R^2$ is the proportion of the variance in the predicted variable, which varies between 0 % and 100 %. That means that the higher percentage shows a better correlation between predicted and observed variables ($R^2$ = 100 % means that all variance in the data is explained by the model).

The next section defines several techniques of data preprocessing for the proper calculation of the algorithm.

## III. Data Preparation Techniques

Data for analysis must be presented in an accessible form suitable for the ML application. It can be numerical data, e.g., example, date, time or nominal, e.g., categories. Before applying the algorithms' model, it is necessary to present them in the required form for processing (for example, binary numbers). Data preprocessing is demanded the algorithm to work correctly. This section describes such methods as One-Hot Encoding (OHE) and Min-Max Scaling.

*a) One-Hot Encoding:* It was already mentioned before, data can be a categorical (nominal) or numerical type. Some algorithms, such as an XGBoost or Catboost, can work with categories, while most ML algorithms operate only with numerical data.

One of the options to represent categorical data in numerical format is to use the OHE technique, i.e., each category is encoding with a binary number. A 1 value is placed in the binary variable for the category and 0 values for the others. For example, data consist the string $Devices$ with labels $laptop$, $computer$, and $cellphone$. Then, each unique category value is assigned with a binary number – $laptop$ is "100", $computer$ – "010", $cellphone$ – "001".

The number of binary variables depends on the number of categories. For example, the $Device$ has 3 labels, which means that it has 3 binary variables. It caused that OHE is applicable with a finite set of label values. Many labels make it inapplicable and impractical for devices as it requires a considerable number of triggers.

*b) Min-Max Scaling:* Min-Max Scaling is a normalization technique for the input variables. This technique is applicable to the input data measured on different scales, which might cause bias during the modeling. The equation for the min-max scaling is as follows

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

where $x_s$ is a scaled variable, $x$ is an input variable, $min(x)$ and $max(x)$ are the minimum and maximum values of all inputs accordingly.

Section II and Section III underline chosen models, metrics, and data preparation techniques. The next section presents the prediction of the traffic based on the selected algorithms.

## IV. Models For Predicting Traffic in the Network

As a rule, the development of a model implies several stages: data preparation, data analysis, application of the models, and analysis of the results. The following provides the details on those stages.

### A. Data Preparation

*a) Gaps:* In reality, some data may disappear or be missed from the record at the time of its arrival. Different reasons cause gaps in the collected infocommunication data: problems with the system, packet loss, interference, and others. Data that is taken for the analysis in this work is *clean*, i.e., it does not have any gaps.

*b) Categorical features:* The algorithms that are used in this work do not operate with nominal (categorical) variables. Therefore, it is necessary to cast all signs to a numeric value. The $Hour$ when the traffic arrived initially has a numeric data type. The $Date$ column is of string type, so this column splits into three: $Month$, $Day$, and $Year$. Each of them is converted to a numeric data type. The original $Date$ string is removed. Thus, our feature space has increased by two. The column $CellName$ is converted using the OHE method.

*c) Feature transformation:* The work uses a linear SVM algorithm. Data must be normalized for linear models, so the Month and Day features were scaled by the interval $[0, 1]$ using min-max normalization. Also, linear models' accuracy can increase if the feature is distributed similarly to the normal law. Initially, the traffic was distributed according to the exponential distribution, but assuming that $Traffic_j = \ln(Traffic_j)$, we obtain a normal distribution.

The time dependence of the average traffic for all cells was considered in detail, see Fig. 1. It can be observed that the dependence, with errors, resembles a sinusoidal one. Based on these conclusions, the $Hour$ feature was coded according to $h_j = \sin(h_j)/T$, where $h_j$ is the day of the month and $T$ is set to 30.

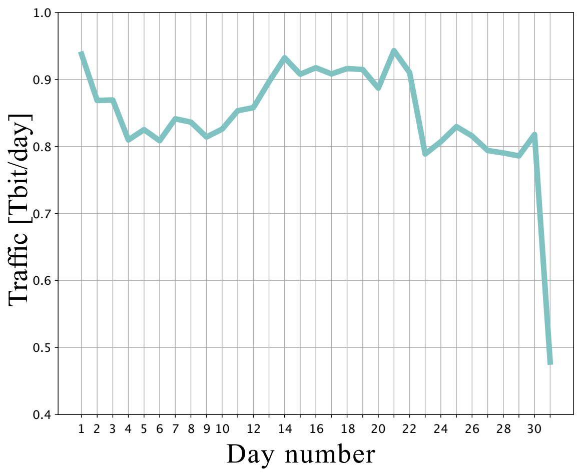

Fig. 1. The distribution of cumulative traffic on a specific day of the month from all subscribers

### B. Data Analysis

It was important to analyze the graphical presentation of the data in order to evaluate the distributions. It became possible to analyze the dependence of the target variable on the characteristic, evaluate this dependence, find out if there is a correlation between the characteristics, and so on.

*a) Three groups of cells were identified:* Yellow – with high traffic (10% of the total number of cells), red – with low traffic (also 10% of all cells) and blue (80% of cells), which have medium traffic. Some of these cells are shown in Fig. 2. From it, one can conclude, for example, about the radio transmitter's location – yellow ones are most likely located in the city center or, for example, in places of large crowds, where various public events can be held. Besides, this is a clear indicator that more control is needed for the yellow cells, since if they go out of operation, then a large number of users will be "cut off" from the network.

*b) Correlation:* For further work with features and the development of new ones, it is necessary to assess their correlation between each other and the target variable. For this, a heat map was built and depicted in Fig. 3. Note, the correlation between the month and the year, in modulus, is very high, which is logical, since in 2017, there are only three months in the data, and in 2018 – ten. This is an unnecessary correlation; it was decided to remove this dependence using the OHE of the $Year$ feature.

## V. Numerical Results

In this work, the following modeling environments were used. Programming language Python 3.7 and Jupyter notebook software were applied for modeling the prediction algorithm.

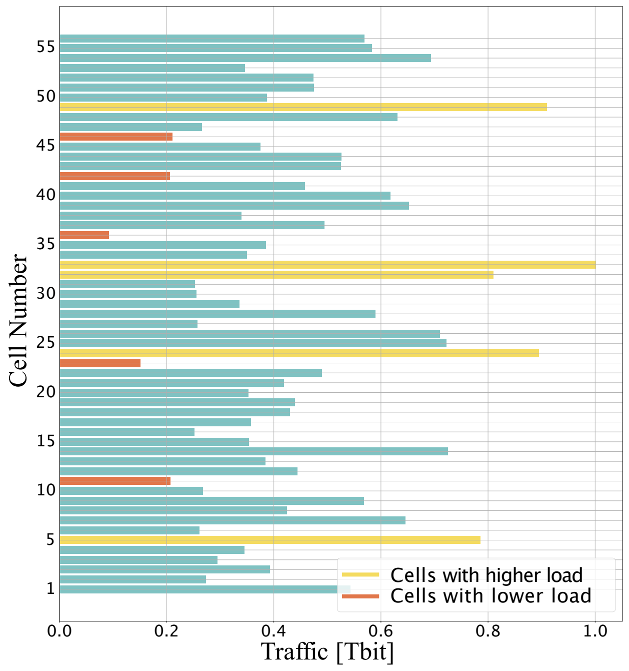

Fig. 2. Part of cells traffic for the entire time period

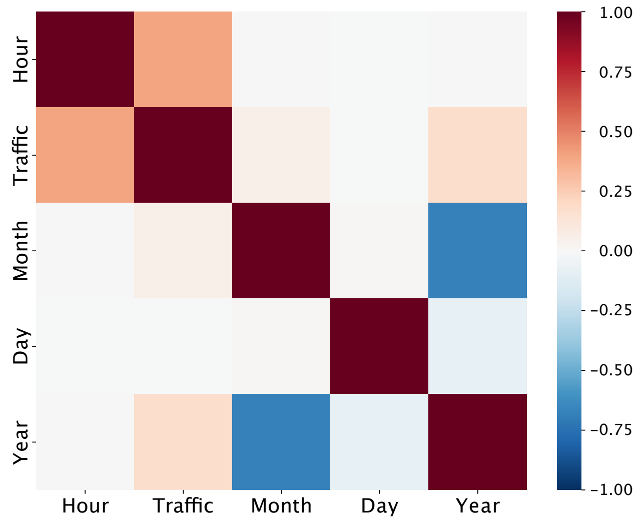

Fig. 3. Correlation between traits and target variable

Libraries for the data analysis and visualization are Numpy, Pandas, Scikit-learn, Scipy, matplotlib, and seaborn.

The problem is formulated in ML terms and the results of the algorithms that were applied are given in Table I.

1) Problem type – regression.
2) Target variable – traffic on one specific cell.
3) Features by which the prediction is made:
   - Time of day ($Hour$), month ($Month$), day ($Day$)
   - Binary features of the cellular communication cells ($CellName$) and binary features of the $Year$ (2017 and 2018)
4) Metrics – RMSE, MAE and $R^2$.

For comparison, it must be said that in MAE and RMSE metrics, the ideal algorithm reaches zero, and in the determination coefficient $R^2$ = 100%.

TABLE I
ALGORITHMS PERFORMANCE EVALUATION

| | **Bagging** | **Random Forest** | **SVM for regression** |
|---|---|---|---|
| RMSE [Mbit] | 2.59 | 3.38 | 2.68 |
| MAE [Mbit] | 1.66 | 2.19 | 1.69 |
| $R^2$, % | 50.8 | 34.2 | 48.8 |
| Wall time of learning [s] | 116 | 122 | 6 |

Table I summarizes the results from where the following conclusions can be drawn. Support vector machines perform slightly worse than ensembles and require additional preprocessing in the form of standardization, but their training requires less time. Therefore, the question of which is better – good quality or the learning rate remains open, since sometimes it is worth having a fast model, and, for example, training it online thereby correcting its errors.

Of all nonlinear models, Bagging showed the best result, but training takes time (wall time of learning is 116 seconds).

Random Forest showed the worst result, but it should not be ruled out, since it depends on the number of features, and our dataset has a small number of them (57 features, of which only 2 are not categorical – this is not enough for this algorithm). Therefore, if we solve the problem of generating new features, then the Random Forest can improve its metrics.

The selection of hyperparameters for Bagging is a rather laborious task. It is worth noting that using Grid Search for the support vector machine, the optimal parameters were selected and it took only a few seconds, but for Bagging it was not possible to do this since this selection takes a long time, which in the context of NGN is very little since everything should work moderately fast. Grid search is an approach to parameter tuning that will methodically build and evaluate a model for each combination of algorithm parameters specified in a grid [21].

## VI. Conclusion and Future Work

This work has shown that it is feasible to extract prediction-suitable information from having even very simple traffic-related data using ML methods to solve optimization problems in infocommunication systems from the traffic prediction perspective. In particular, the traffic was predicted for the LTE edge operation based on the dataset from 57 cells.

Algorithms used in this work are detached from the entire system. A challenge that needs to be addressed further is the fact that algorithms should not work separately from the network. For example, it is possible to propose to use an additional overlay, where problems are solved by voting for the best result if the algorithms "argu" with each other. In addition, if there is no disagreement between the algorithms, then the conclusion is made at the place where the data came.

In this work, three algorithms for solving the problem of predicting traffic in the LTE network were demonstrated: Bagging, Random Forest, and SVM. The emphasis was on three parts: primary data processing, visualization, and forecasting.

Based on the obtained results, we can conclude the real application of machine learning algorithms for predicting traffic in the network. The data that the algorithm should receive as input must either be pre-processed quickly before the prediction itself, or this process should be taken out in a separate part of the system, which will make it possible to be confident in the quality of the data for the algorithm, but excludes the possibility of training the model in real-time.

Moreover, the second point is rather intended to interpret the data for better processing. After all, the ML algorithm is a black box that works based on mathematical transformations that are not always understandable to humans. So data interpretation is also an important machine learning process, which allows predicting a little how the algorithm will work and find patterns in the data before obtaining model results.

The third point is the most important. It shows that algorithms taken from various programming languages libraries cannot immediately give a reasonably good result. They require tuning hyper-parameters for a specific task and creating new features to approximate the true relationship in the data better. It is necessary to understand that training the model is not a fast process. Therefore it is necessary to assess the importance of features and choose the right hyper-parameters to balance quality and speed, which is also an essential factor in NGN beyond 5G.

The undoubted advantage is that the work demonstrated in this paper has good prospects planned to be implemented in the future. First, to train the Bagging on Grid Search, which will improve the prediction quality. Second, to implement and compare other machine learning algorithms. Third, to provide more details and impact from more features, e.g., to apply other approaches to coding CellName, find other transformations of the feature space, and generate new features based on the existing ones. Fourth, to combine different data scaling approaches, such as using standardization, which may improve the linear SVM algorithm. Fifth, to train not on all objects, but, e.g., on those cells where the traffic has a total high, medium, and low values, respectively, separately, which will improve the quality of prediction for each group. A specific action point will be given to the research question if it is possible to solve the inverse problem – classifying a cell by time, date, and traffic that came to it. For example, if there is a need to find out about service radio stations quickly.

## Acknowledgment

This paper is supported by the Doctoral training network in ELectronics, Telecommunications and Automation (DELTA), and by the Academy of Finland ULTRA project (328226/328214).

## References

[1] D. Michie, D. J. Spiegelhalter, C. Taylor *et al.*, "Machine Learning," *Neural and Statistical Classification*, vol. 13, no. 1994, pp. 1–298, 1994.

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

[3] R. Pirmagomedov, D. Moltchanov, A. Ometov, K. Muhammad, S. Andreev, and Y. Koucheryavy, "Facilitating mmWave Mesh Reliability in PPDR Scenarios Utilizing Artificial Intelligence," *IEEE Access*, vol. 7, pp. 180 700–180 712, 2019.

[4] M. Usama, J. Qadir, A. Raza, H. Arif, K.-L. A. Yau, Y. Elkhatib, A. Hussain, and A. Al-Fuqaha, "Unsupervised Machine Learning for Networking: Techniques, Applications and Research Challenges," *IEEE Access*, vol. 7, pp. 65 579–65 615, 2019.

[5] P. Masek, P. Sedlacek, A. Ometov, J. Mekyska, P. Mlynek, J. Hosek, and M. Komarov, "Improving the Precision of Wireless Localization Algorithms: ML Techniques for Indoor Positioning," in *Proc. of 42nd International Conference on Telecommunications and Signal Processing (TSP)*. IEEE, 2020, pp. 1–6.

[6] "Artificial Intelligence In 2019: Getting Past The Adoption Tipping Point," [Online] https://blumbergcapital.com/ai-in-2019/, (accessed on October 26, 2020).

[7] "Global AI Survey: AI Proves Its Worth, But Few Scale Impact," [Online] https://www.mckinsey.com/featured-insights/artificial-intelligence/global-ai-survey-ai-proves-its-worth-but-few-scale-impact, (accessed on October 26, 2020).

[8] G. S. Collins and K. G. Moons, "Reporting of Artificial Intelligence Prediction Models," *The Lancet*, vol. 393, no. 10181, pp. 1577–1579, 2019.

[9] M. O. Riedl, "Human-Centered Artificial Intelligence and Machine Learning," *Human Behavior and Emerging Technologies*, vol. 1, no. 1, pp. 33–36, 2019.

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

[11] A. Ometov, E. Olshannikova, P. Masek, T. Olsson, J. Hosek, S. Andreev, and Y. Koucheryavy, "Dynamic Trust Associations Over Socially-Aware D2D Technology: A Practical Implementation Perspective," *IEEE Access*, vol. 4, pp. 7692–7702, 2016.

[12] G. Kalfas, C. Vagionas, A. Antonopoulos, E. Kartsakli, A. Mesodiakaki, S. Papaioannou, P. Maniotis, J. S. Vardakas, C. Verikoukis, and N. Pleros, "Next Generation Fiber-Wireless Fronthaul for 5G mmWave Networks," *IEEE Communications Magazine*, vol. 57, no. 3, pp. 138–144, 2019.

[13] S. A. Kumar and K. S. Murthy, "An Efficient Operations and Management Challenges of Next Generation Network (NGN)," *Global Journal of Computer Science and Technology*, 2014.

[14] N. Mäkitalo, 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.

[15] "Predict traffic of LTE network," [Online] https://www.kaggle.com/naebolo/predict-traffic-of-lte-network, (accessed on October 26, 2020).

[16] 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.

[17] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, "Quantum Machine Learning," *Nature*, vol. 549, no. 7671, pp. 195–202, 2017.

[18] 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.

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

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

[21] G. Ranjan, A. K. Verma, and S. Radhika, "K-nearest Neighbors and Grid Search CV Based Real Time Fault Monitoring System for Industries," in *Proc. of IEEE 5th International Conference for Convergence in Technology (I2CT)*. IEEE, 2019, pp. 1–5.