IJARST

International Journal of Advanced Research in Science and Technology

https://doi.org/10.62226/ijarst2024132519

ISSN 2319 – 1783 (Print)
ISSN 2320 – 1126 (Online)

Prediction of Mobile Network Performance Using Supervised Machine Learning Models

[1*]F. U. Didigwu, [2]J. C. Anichi

[1*]*Department of Electrical/Electronic Engineering, University of Port Harcourt, Nigeria*

[2]*Department of Data Science, Coventry University, United Kingdom*

[1*]didigwu.fidelis@uniport.edu.ng, [2]jonnyanichi@gmail.com

ARTICLEINFO

Article history:

Received 06 Nov 2024
Accepted 28 Nov 2024
Available online 13 Dec 2024

Keywords:

Machine Learning,
Random forest Regression,
Downlink Throughput,
LTE, Mobile Network

ABSTRACT

Mobile network management and drive tests provide services that give a clear insight into the quality of mobile network coverage and other wireless networks including identifying areas of poor signal quality and identification of black spots. This research investigates the potential of supervised machine learning models to predict mobile network performance, focusing on Coventry University and Coventry City Center, United Kingdom. The measured download speed represented as Downlink bit rate (DL_bit rate) at these locations are used to obtain user performance data using Lebera mobile virtual network operator SIM card in order to assist with the visualization of the mobile network performance maps. The study explores the application of various machine learning models, including Random Forest, Linear Regression, Gradient Boosting, Support Vector Regression (SVR), and K-Nearest Neighbors (KNN), to forecast mobile network throughput speed (DL_Throughput). Among the models, the Random Forest Regression (RFR) model demonstrated the highest accuracy, with a Mean Absolute Error (MAE) of 6.79 and an R-squared (R^2) value of 0.496, indicating its effectiveness in capturing network performance variability. The findings highlight the considerable promise of machine learning in optimizing mobile networks, enabling telecommunications providers to automate network management, enhance resource allocation, and improve the Quality of user Experience (QoE). The application of predictive analytics in this context offers significant advantages, including increased network efficiency, enhanced user satisfaction, and reduced operational costs. This study underscores the importance of integrating AI-driven predictive models into mobile network optimization strategies to meet the evolving demands of urban environments.

1. Introduction

The measured download speed represented as the DL_bit rate at locations around Coventry city Center and Coventry University is used to obtain user performance data using Lebera mobile virtual network operator SIM card in order to assist with the visualization of the mobile network performance maps. An increasing amount of scholarly works acknowledges the significance of performing drive/walk tests. Drive tests and mobile network management offer services that provide a clear information on the coverage quality of other wireless networks and mobile networks, including the detection of coverage holes and places with low signal quality. A drive test can also give the user information about how well the mobile network complies with applicable technical specifications and laws [1].

There are several key performance indicators observed during mobile network drive tests, including Reference Signal Received Power (RSRP), Reference Signal Received Quality (RSRQ), Signal-to-Noise Ratio (SNR), Channel Quality Indicator (CQI), and Received Signal Strength Indicator (RSSI). These metrics are fundamental Radio Resource Management (RRM) parameters in 4G/5G mobile communication systems. Link adaptation and packet scheduling utilize the Signal-to-Interference plus Noise Ratio (SINR) to assess channel quality, while RSRP and intra-eUTRAN (evolved Universal Terrestrial Radio Access Network) measure network coverage and quality [2]. Due to the strong correlation between these parameters and signal strength, throughput can be used to estimate signal strength, which also depends on factors like location.

Human interaction is necessary for mobile network administration in order to meet user network Quality of

Service (QoS) requirements, diagnose and analyze customer complaints, and conduct drive testing. In addition to being time-consuming, this method requires analyzing vast and complicated amounts of data in order to identify the hidden fluctuation pattern in the network connectivity and provide insightful information about the dataset. Throughout procedure, a significant amount of data is required, including information on nearby cells [3].

Due to the radio frequency tools' inability to perform location-based queries, some locations are not covered during the drive test. As a result of this, machine learning (ML) must be used, since a lot of data is produced by numerous people and that have gadgets linked to the network. Using machine learning (ML), a method of creating a model that is artificially trained to complete a task without explicit coding to do predictive analytics in the telecom industry is very beneficial to the mobile Network operators and mobile virtual network operators making routine planning, monitoring, and optimization tasks easier.

Machine learning technique that understands the connection between input and output is supervised machine learning. The output is commonly referred to as the target or "y variable," and the inputs are known as features or "X variables." Labeled data is the kind of data that includes both the target and the features. It is the primary distinction between the two well-known forms of machine learning, supervised and unsupervised [4].

Smooth experiences and dependable connectivity are dependent on a well-managed network. Networks may be continuously assessed, tracked, and optimized to make the most use of the resources at hand with the incorporation of AI. With the use of AI algorithms, consumers may be guaranteed uninterrupted service by locating network congestion areas, anticipating breakdowns, and proactively rerouting traffic to avoid bottlenecks. In order to precisely forecast future demand, AI algorithms can examine past network data, user behavior, and traffic patterns. This makes it possible for telecom companies to design network capacity more efficiently, build infrastructure strategically, and distribute resources wisely. Providers may now guarantee that resources are distributed where they are needed, improving network performance and reducing costs, by anticipating customer requests [5].

The purpose of this research is to investigate how mobile network performance at Coventry University and Coventry City Center may be predicted using supervised machine learning models. There is a lot of promise for mobile network optimization with machine learning models. Operators can automate network management activities, predict throughput, coverage, improve resource allocation, and evaluate network data by utilizing machine learning and deep learning algorithms. The advantages of AI-driven mobile network optimization are significant, outweighing the difficulties; they include better network efficiency, greater user experience, and lower operating costs.

2. Study Area

Coventry University was founded in 1843 as the Coventry School of Design and has evolved into a major educational institution. Initially focused on arts and design, it expanded into technical and engineering fields, becoming Lanchester Polytechnic in 1970 and Coventry Polytechnic in 1987. It gained university status in 1992. Today, Coventry University is known for its innovative teaching, industry connections, and research excellence, with a diverse student body from over 130 countries.

Coventry, a city with a rich history dating back to medieval times, became a center for industry, particularly in textiles, bicycles, and later, the automotive sector. Over the years, Coventry transitioned from manufacturing to a more diversified economy, embracing education, technology, and services.

Culturally, Coventry is notable for its contributions to the arts and its recognition as the UK City of Culture in 2021. This designation has highlighted the city's cultural diversity and spurred further investment in its cultural and creative industries. Today, Coventry is a dynamic city that balances its historical roots with a forward-looking vision, supported by its educational institutions and vibrant cultural scene [6].

Given its developmental status, ensuring high-quality mobile network performance is essential for seamless connectivity at Coventry University and throughout Coventry city. This will help drive economic growth and support human capital development.

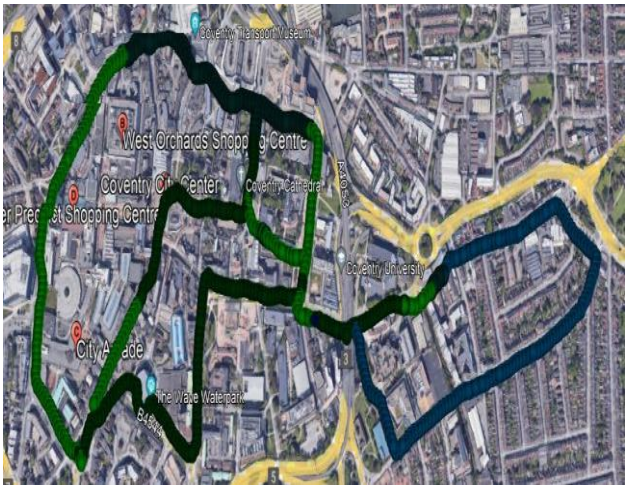

Fig.1: Drive test route

3. Literature Review

The area of mobile network performance is examined in this paper, with a particular emphasis on throughput speed and

the use of supervised machine learning models in predictive analytics. Previous researches have shown that there are many approaches and techniques that are been developed over time to forecast and optimize mobile network performance in this field.

[7] created a machine learning (ML) model to forecast uplink (UL) traffic for 4G and 5G mobile networks. Drive tests was carried out on 4G LTE networks in Melbourne, Florida; Batman, Turkey; and Houston, Texas to collect data. For the purpose of estimating UL throughput, the effectiveness of machine learning (ML) techniques was assessed, including k-nearest neighbor (KNN), gradient boosting regression, decision tree regression (DTR), and linear regression. For each model, the coefficient of determination (R^2) and Root Mean Square Error (RMSE) were computed, DTR and KNN had the highest R^2 values, both at 0.92.

[8] researched on predicting user data transmission rates (throughput) in a mobile network using machine learning. After comparing several models, it was discovered that on a slower, less crowded road, K-Nearest Neighbors (K-NN) fared the best. The authors agreed that it is more difficult to forecast throughput on a freeway due to faster speeds and more frequent switching between cell towers. This study presents a novel approach to identify problematic cells in mobile networks that experience low throughput, or slow data transmission speeds. As network complexity increases and data privacy laws restrict data collecting, this is becoming increasingly important. This implies that these issues should be the focus of future research.

[9] presented a novel approach to identify problematic cells in mobile networks that experience low throughput, or slow data transmission speeds. This is getting more and more crucial as data privacy regulations limit data collection and network complexity rises. To solve this issue, a combination of deep learning and clustering approaches is used in the developed solution. With a small amount of tagged data (useful for privacy reasons) collected from a 4G network, it excels at classifying cells. Their strategy produced outstanding results, with an F_1-Score of 0.67 which is much higher than the 0.25 of the baseline method. This result was a significant boost in the ability to identify faulty cells in the network. Also, in order to identify the optimal model for throughput prediction, several methods suggest a machine learning approach for forecasting mobile network performance. The model with the highest R^2 score and lowest evaluation metric error for downlink throughput prediction was the random forest model. Its extra degree of unpredictability contributes to accuracy gains and variance reduction. The authors concluded that two critical elements influencing network performance are the Signal-to-Noise Ratio (SNR) and measurement location.

In another study by [10] reviewed multiple machine learning (ML) models that forecast mobile network performance in diverse situations and frequency bands using Reference Signal Received Power (RSRP) which was used in predicating the network coverage over the given area. The study concentrated on a number of machine learning models, such as Gaussian Process Regression (GPR), Regression Trees (RT), Extremely Randomized Trees (ET), Support Vector Machines (SVM), Artificial Neural Networks (ANN), and Linear Regression (LR). Measurement data from 4G LTE frequency bands (1800 MHz and 2600 MHz) in various locations surrounding Putrajaya, Malaysia, were used to train the models. In terms of Root Mean Square Error (RMSE) and R^2 for RSRP prediction, the GPR model was determined to be the most accurate. It is less useful, nevertheless, because of its lengthy training times and slow prediction speed. As the second most accurate model, Random Forest (RF) was suggested as the most useful machine learning (ML) model for creating rigorous RSRP prediction models in various contexts and bands.

In order to improve the quality of service, [11] reviewed machine learning approach to control the enormous demand for upload, Uplink (UL) transmission. The primary supervised learning techniques which are Random Forest and Support vector machine were used to forecast UL bandwidth in the range of 88% to 94% with various scenarios. The various key parameters of their investigation were signal strength, signal quality, noise levels and location. The study shows how resilient the models are to different factors.

[12] reviewed the issue of using throughput prediction algorithms in 4G LTE networks. Through the usage of the QXDM (Qualcomm extensible Diagnostic Monitor) toolkit, they gathered data and created Link Forecast, a machine learning-based framework for throughput prediction. The approach predicts instantaneous link bandwidth using lower-layer data.

[13] employed the Random Forest method to predict throughput. In addition to the data gathered from crowdsourcing, the authors also relied on additional data from network operators for throughput prediction. Measurements of the radio access network, including average cell throughput, average number of users in each cell, and connection success rate, were included in the extra data. The authors came to the conclusion that the accuracy of the throughput prediction was enhanced by these extra data.

The Random Forest (RF) model for throughput prediction in the LTE network carried out by [14] emphasized the model's use in preserving an autonomous vehicle's dependable connection to infrastructure.

A RF model was also used by [15] to forecast the throughput of video streaming. The result showed high accuracy of the model.

Different models based on Deep Neural Network (DNN) techniques used in another research by [16] allowing for throughput prediction in places without prior knowledge of mobile network performance. Uplink connection throughput frequently fluctuates when using a live streaming service,

especially at high vehicle speeds, which results in service delays.

Stationary LTE measurements experiment of downlink (DL) mobile network performance was carried out by [17]. DL throughput is predicted using data collected from pedestrian lanes, highways, and regional routes. Using ML algorithms on two US cellular carriers, the study obtains an error rate ranging from 4 to 17%. The study achieved high accuracy in downlink (DL) prediction, but there remains potential for improvement to fully eliminate error variations.

In order facilitate network performance, [18] conducted research on the Self-Organizing Network (SON) to ascertain the Quality of Service of LTE network. Key performance indicators (KPIs) used was learning-based SON as estimation parameters, which dynamically adjusts to the environment in order to improve QoS. Using 95% of the best network performance, the researchers employed LTE features including dynamic frequency and bandwidth assignment.

Also, [19] conducted research which showed that network operators can optimize capacity, dynamically assign resources, and enhance end-user Quality of Service (QoS) by precisely anticipating network demand by avoiding pointless maintenance and improving network performance with focused interventions, this can lower operating costs.

4. Method

In this research a machine learning models was created to forecast mobile network throughput in Coventry University and City Center. Data on mobile network throughput speeds was gathered while conducting a walk test following pre designated routes through Coventry University and the city center using 4G/5G Samsung Galaxy A25 phone containing lebara (Vodafone) mobile network SIM cards and G-NetTrack Pro which is an open-source application on google play store [20]. There are 5178 rows and 270 characteristics in the dataset. The target variable for the machine learning models was represented by this data. Alongside throughput speeds, the walk test gathered more information that may impact network performance. This involved measurements of the signal strength (RSRP), Signal Quality (RSRQ), Signal to noise ratio (SNR), GPS coordinates to determine the exact location of each measurement, Serving Cell site details (Physical Cell ID, type of network) measured from the broadcast information on the network over the phone.
Python is a widely used programming language, chosen for this work due to its ease of learning, fast execution, capabilities, readability, and large supportive programming community. It is a popular programming language used in many domains, such as machine learning, web development, scientific computing, and data analysis [21]. Strong modules like NumPy, Pandas, and scikit, in particular, make Python an ideal tool for handling, processing, and analyzing massive volumes of data.

Figure 2 shows design flow diagram of the research process, indicating each stage of data preparation till output is achieved. It employed top-down design methodology. It is applied to systems, and denotes several stages in the process of creating a distributed diagrammatic representation of a research from the top to bottom. It provides a clearer understanding of the research design, as each module is thoroughly explained.

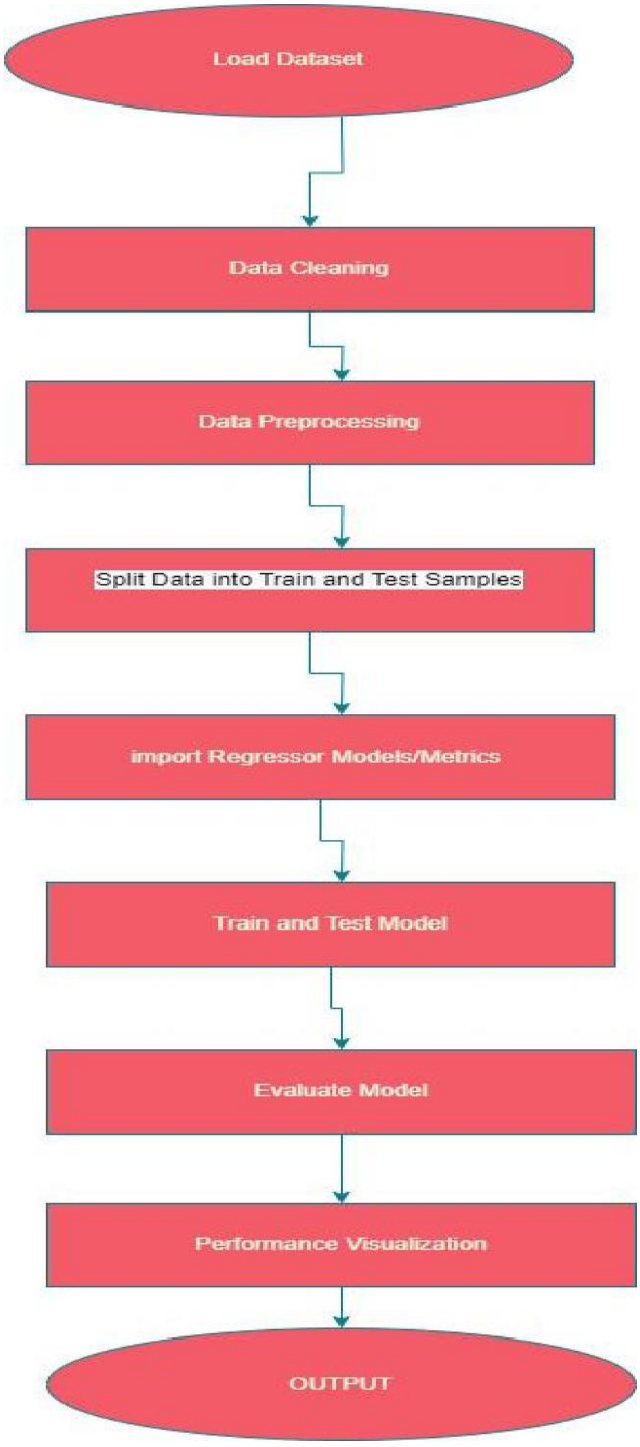

Fig. 2: Design Activity Flow

5. Model Evaluations

The preprocessed data was used to create the training and testing sets. The training set was employed to train the selected machine learning models, and the evaluation process involved analyzing the performance of several regression models on the prepared dataset. Five regression models were trained and evaluated: K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Gradient Boosting, Random Forest, and Linear Regression. Various metrics, such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and R-squared (R^2) score, were calculated to assess the models' performance. These metrics provided insights into the reliability and accuracy of the models. Additionally, scatter plots comparing actual and predicted values were generated to visually demonstrate each model's performance.

6. DATA PROCESSING

The first step in data preprocessing is to look for and handle missing values. Next, any columns that have missing data are removed. Certain columns are filtered to remove non-numeric rows, and pertinent columns are changed to integer types. Data quality is ensured by identifying and removing duplicate records. To cut down on noise, only the numerical columns are chosen for analysis and any extraneous features removed. In order to see and comprehend the associations between features, a heatmap is created and a correlation matrix is calculated.

Figure 3 shows correlation heatmap which is a useful tool for visualizing correlations among several variables. There is a significant positive correlation between DL_bitrate and Qual (0.2), CQI (0.1), and SNR (0.2). This suggests that SNR, CQI, and Qual are good predictors of DL_bitrate since they are correlated with greater DL_bitrate values. Conversely, DL_bitrate shows negative correlations with both Level (-0.5) and PSC (-0.1), suggesting that increase in Level and PSC are associated with DL_bitrate declines. Also, the correlation heatmap makes it simpler to comprehend the relationships between the various dataset pieces. It assists in identifying possible interactions between features and highlights the variables that have the strongest correlation with the target variable, DL_bitrate. With regard to mobile network performance, this data is essential for feature selection and enhancing the precision of predictive models.

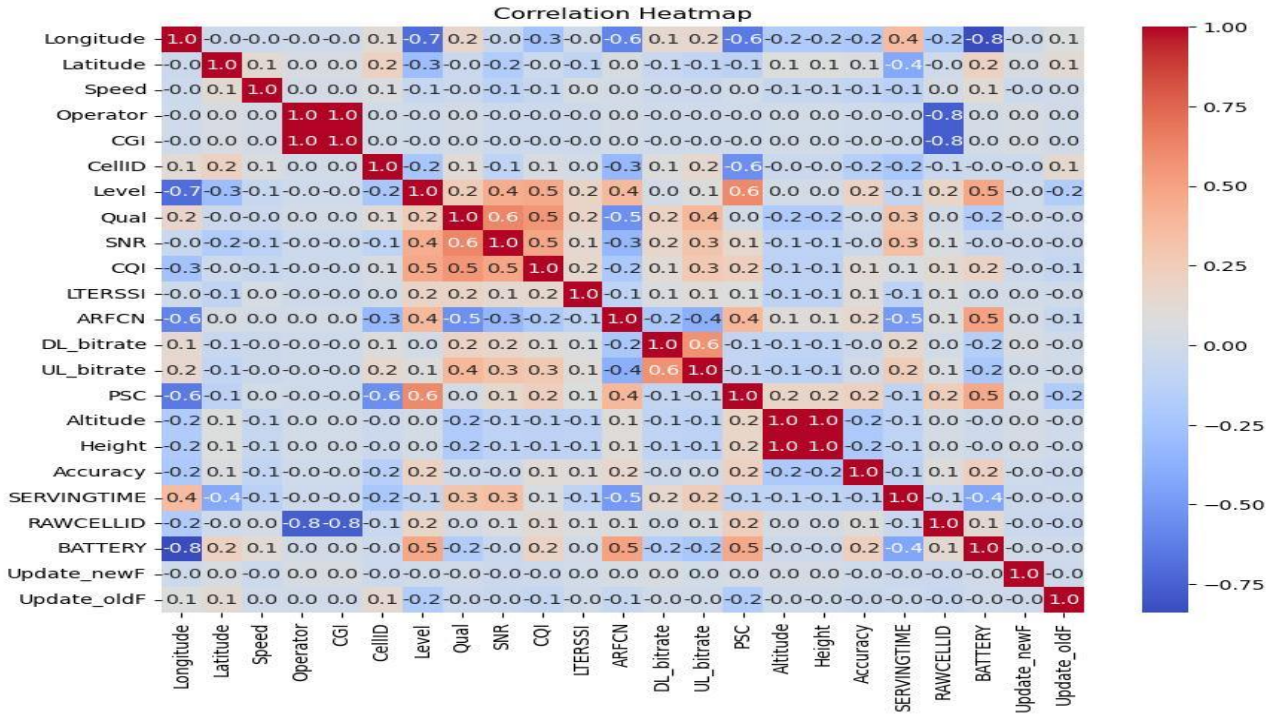

Fig. 3: Correlation Heatmap

7. DATA PREPARATION

Importing libraries that are necessary for data processing, statistical analysis, and visualization like pandas, seaborn, matplotlib, numpy, and plotly.express is the first step in data preparation.

Figure 4 shows the import of all the libraries that are crucial for this process.

```
In [ ]: # Importing necessary libraries
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
import warnings
warnings.filterwarnings('ignore')
import plotly.express as px

# Load the dataset
df = pd.read_csv("Lebara_2024.06.25_11.19.10.csv")
# Display the entire DataFrame
df
# Get the number of rows and columns in the dataset
df.shape
# Check for missing values in each column
df.isna().sum()
# Get all unique values in the 'DL_bitrate' column
df["DL_bitrate"].unique()
```

Fig. 4: Snapshot of python codes for data preparation.

The Python code in figure 4 starts by importing libraries for scientific computation (NumPy), data management (Pandas), visualization (Seaborn, Matplotlib, Plotly), and handling errors (warnings). For structured data analysis, the code loads information from a CSV file into a Pandas DataFrame. Throughout the script, the DataFrame is conveniently referenced by its variable name, df. To quickly view the data within the DataFrame, the code shows the whole DataFrame. Next, it uses df.shape to expose the number of rows and columns in order to verify the size of the data. Lastly, the code uses df.isna() to check for missing values.sum(). The purpose of this code snippet is to load, check, and prepare the data for further analysis.

8. DATA CLEANING

One of the most important steps in the data preparation process is data cleansing, it removes unwanted data, and deletes empty columns in the data set.

To generate a cleaned DataFrame, it starts by using the isna().sum() method to find and count any missing values in each column. After that, any missing values are dropped from the columns. Subsequently, it purges rows containing certain values denoted by hyphens ("-") from particular columns and, for consistency, these columns are changed to integer data types. The drop_duplicates() function is used in the code to eliminate duplicate rows from the DataFrame. It is necessary to follow this procedure to ensure that the dataset is consistent and clean, which is necessary for additional modeling and analysis. Figure 5 shows python codes for data cleaning.

```
In [ ]: # Handling Missing Values
nan_count = df.isna().sum()
print("\nCount of NaN entries in each column:")
print(nan_count)
df_cleaned = df.dropna(axis=1)
df = df_cleaned
df.isna().sum()

# Cleaning Specific Columns
df = df[df["Qual"] != "-"]
df = df[df["SNR"] != "-"]
df = df[df["CQI"] != "-"]
df = df[df["LTERSSI"] != "-"]
df["Qual"] = df["Qual"].astype(int)
df["SNR"] = df["SNR"].astype(int)
df["CQI"] = df["CQI"].astype(int)
df["LTERSSI"] = df["LTERSSI"].astype(int)
df.info()

# Removing Duplicates
df.duplicated().sum()
df = df.drop_duplicates()
```

Fig.5: Python codes for data cleaning

9. EXPLORATORY DATA ANALYSIS

Exploratory data analysis gives a quick overview of the data distribution, the dataset is first shown using histograms created with plt.hist. Figure 6 is the summary of the features in the dataset.

```
# Visualization of the dataset for a brief insight
plt.figure(dpi=120)
df.hist(figsize=(20,15))
plt.show()

# Correlation Analysis
corr_matrix = df_num.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, annot=True, cmap="coolwarm", fmt=".1f", annot_kws={"size": 10})
plt.title("Correlation Heatmap")
plt.show()
```

Fig.6: Python codes for visualization dataset

The next step involves doing a correlation analysis to determine the links between numerical variables. To do this, a correlation matrix is created using df_num.corr(). The correlation matrix is then shown as a heatmap with comments using sns.heatmap, giving a clear and succinct picture of the relationships between the various variables. Figure 7 shows histogram representation of the features in the dataset.

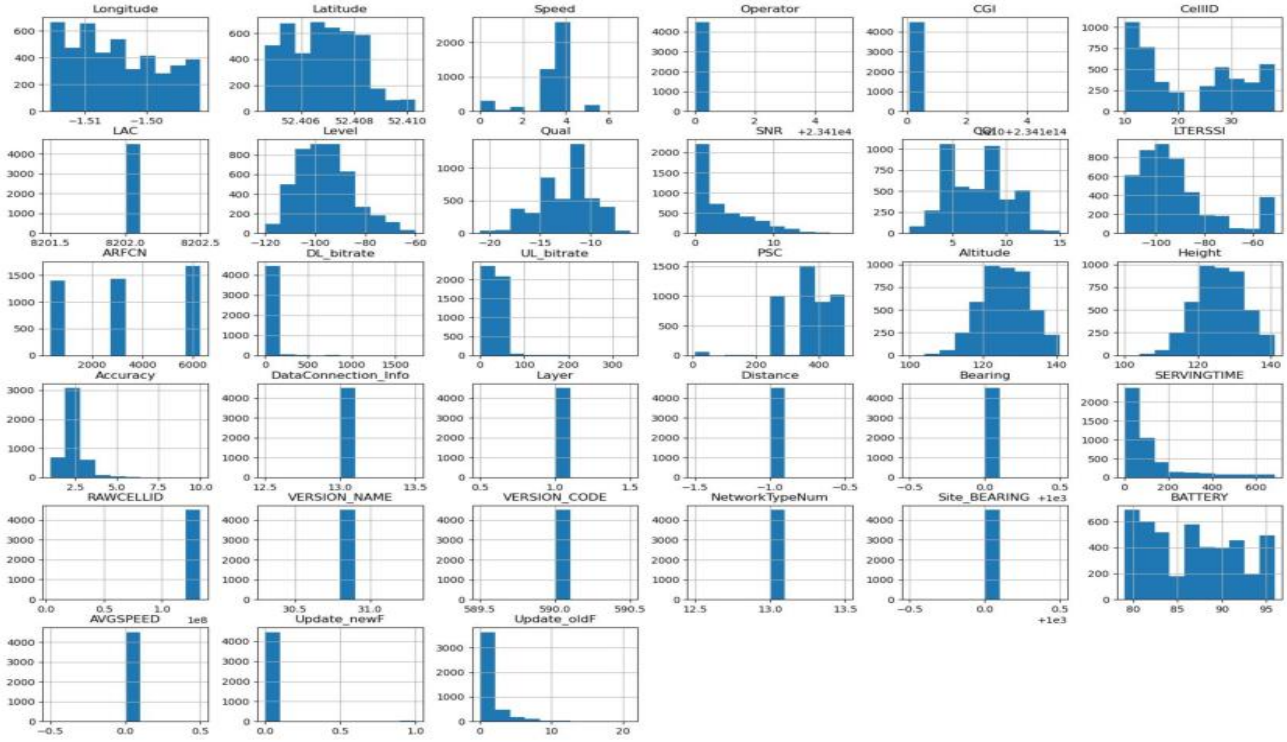

Fig. 7: Histogram representation Dataset

Figure 7 provides a visual representation of the distribution for each variable by displaying histograms for the different features in the dataset. A distinct attribute, such as longitude, latitude, speed, and DL_bitrate, is represented by each subplot. The frequency of data points inside designated ranges for each attribute is shown by the histograms, which aid in the identification of trends, outliers, and the general distribution of the data. For instance, the distribution of DL_bitrate is skewed, with the majority of values concentrated at lower ranges; in contrast, the distribution of features like latitude and longitude is more uniform. These representations are essential for comprehending the properties of the data.

10. FEATURE ENGINEERING

A crucial step in getting the data ready for modeling is feature engineering which removes every superfluous and irrelevant data to the model.

Figure 8 is a python code which was used to remove unwanted columns in the work.

```
# Selecting Numerical Columns
df_num = df.select_dtypes(include=[int, float])

# Dropping Unnecessary Columns
columns_to_drop = ["LAC", "DataConnection_Info", "Layer", "Distance", "Bearing", "VERSION_NAME", "VERSION_CODE",
                   "NetworkTypeNum", "Site_BEARING", "AVGSPEED"]
df_num = df_num.drop(columns=columns_to_drop)

# Preparing Data for Modeling
X = df_num.drop(["DL_bitrate"], axis=1)
y = df_num["DL_bitrate"]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
scaler = MinMaxScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

Fig.8: Python codes to remove irrelevant columns for the model

Using df.select_dtypes (include=[int, float]), it first selects only the dataset's numerical columns. Next, it removes columns like "LAC," "DataConnection_Info," and others that are superfluous and irrelevant to the model. Next, the data is divided into two variables: the target variable (y), which is DL_bitrate, and the characteristics (X). Using train_test_split, the dataset is further split into training and test sets in an 80:20 ratio. Lastly, to normalize the data and enhance the performance of machine learning models, the features are scaled using MinMaxScaler.

11. DEVELOPING AND EVALUATING THE MACHINE LEARNING MODELS

Through the use of data and machine learning, computer systems make better decisions on a given task and perform better. For both regression tasks, a variety of supervised machine learning algorithms were used in this research.

11.1Random Forest Regression Mode

The Random Forest Regressor was first initialized with 100 estimators and a random state of 42 after importing the required libraries. After being trained on the training set of data (X_train, y_train), the model is applied to the test set of data (X_test) to generate predictions. To assess the correctness of the model, a number of performance measures are computed and printed, including Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and R-squared (R^2). In order to visually evaluate the model's performance, the code finally plots the actual values against the anticipated values. Figure 9 shows the python codes for creating Random Forest Regression Model.

```
from sklearn.ensemble import RandomForestRegressor
rf_reg = RandomForestRegressor(n_estimators=100, random_state=42)
rf_reg.fit(X_train, y_train)
y_pred_rf_reg = rf_reg.predict(X_test)
from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_error
print("Random Forest Regression MSE:", mean_squared_error(y_test, y_pred_rf_reg))
print("Random Forest Regression RMSE:", mean_squared_error(y_test, y_pred_rf_reg, squared=False))
print("Random Forest Regression MAE:", mean_absolute_error(y_test, y_pred_rf_reg))
print("Random Forest Regression R2:", r2_score(y_test, y_pred_rf_reg))

# Plot actual vs predicted values
plt.figure(figsize=(10, 6))
plt.scatter(y_test, y_pred_rf_reg, color='blue', label='Actual vs Predicted')
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], color='red', linestyle='--', label='Perfect Prediction')
plt.xlabel('Actual DL_Throughput')
plt.ylabel('Predicted DL_Throughput')
plt.title('Actual vs Predicted DL_Throughput (Random Forest Regression)')
plt.legend()
plt.show()
```

Fig.9: Random Forest Regression Model

11.2Linear Regression

This machine learning model adopts a linear relationship between an independent and dependent variable to forecast future events. Here it was used to evaluate throughput speed and the key performance index of the network.

Figure 10 below shows how to use scikit-learn to create a linear regression model.

```
from sklearn.linear_model import LinearRegression
# Linear Regression Model

# Initialize and Train Linear Regressor
linear_reg = LinearRegression()
linear_reg.fit(X_train, y_train)

# Make Prediction
y_pred_linear = linear_reg.predict(X_test)
# Test and calculate the evaluation metrics

print("Linear Regression MSE:", mean_squared_error(y_test, y_pred_linear))
print("Linear Regression RMSE:", mean_squared_error(y_test, y_pred_linear, squared=False))
print("Linear Regression MAE:", mean_absolute_error(y_test, y_pred_linear))
print("Linear Regression R2:", r2_score(y_test, y_pred_linear))
# Plot actual vs predicted values
plt.figure(figsize=(10, 6))
plt.scatter(y_test, y_pred_linear, color='blue', label='Actual vs Predicted')
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], color='red', linestyle='--', label='Perfect Prediction')
plt.xlabel('Actual DL_Throughput')
plt.ylabel('Predicted DL_Throughput')
plt.title('Actual vs Predicted DL_Throughput (Linear Regression)')
plt.legend()
plt.show()
```

Fig. 10: Python codes creating Regression model

11.3Gradient Boosting

It is a kind of machine learning that is based on combining the best subsequent model with earlier models to minimize total prediction error. The main concept is to reduce the error by setting the target outcomes for the subsequent model. Figure 11 below shows how to use scikit-learn to train and assess a Gradient Boosting Regression model.

```
from sklearn.ensemble import GradientBoostingRegressor
# Gradient Boosting (GB) Regression Model

# Initialize and Train GB Regressor
gb_reg = GradientBoostingRegressor(n_estimators=100, learning_rate=0.1, max_depth=3, random_state=42)
gb_reg.fit(X_train, y_train)

# Make predictions
y_pred_gb = gb_reg.predict(X_test)
# Test and calculate the evaluation metrics

print("Gradient Boosting Regression MSE:", mean_squared_error(y_test, y_pred_gb))
print("Gradient Boosting Regression RMSE:", mean_squared_error(y_test, y_pred_gb, squared=False))
print("Gradient Boosting Regression MAE:", mean_absolute_error(y_test, y_pred_gb))
print("Gradient Boosting Regression R2:", r2_score(y_test, y_pred_gb))

# Plot actual vs predicted values
plt.figure(figsize=(10, 6))
plt.scatter(y_test, y_pred_gb, color='blue', label='Actual vs Predicted')
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], color='red', linestyle='--', label='Perfect Prediction')
plt.xlabel('Actual DL_Throughput')
plt.ylabel('Predicted DL_Throughput')
plt.title('Actual vs Predicted DL_Throughput (Gradient Boosting Regression)')
plt.legend()
plt.show()
```

Fig. 11: Gradient Boosting Regression Model

Initialization of Gradient Boosting Regression model with predetermined hyperparameters is followed by training on the training dataset and testing it on the test dataset using metrics like MSE, RMSE, MAE, and R-squared. In order to visually compare the actual and anticipated DL_Throughput numbers and assess the performance of the model, a graph was also created.

11.4Support Vector Regression Model

This model was tested using test data and trained on scaled training data. It was evaluated to ascertain the correctness of the model through the performance metrics such as MSE, RMSE, MAE, and R-squared. A plot comparing the real and predicted DL_Throughput values obtained which shows the correctness of the model. A Support Vector Regression (SVR) model is implemented in figure 12.

```
from sklearn.svm import SVR
# Split data into features (X) and target (y)

# Initialize and Train SVM Regressor
svm_reg = SVR()
svm_reg.fit(X_train_scaled, y_train)

# Make Predictions with SVM Regressor
y_pred_svm_reg = svm_reg.predict(X_test_scaled)
# Function to evaluate and print metrics
def print_metrics(model_name, y_test, y_pred):
    print(f"{model_name} MSE:", mean_squared_error(y_test, y_pred))
    print(f"{model_name} RMSE:", mean_squared_error(y_test, y_pred, squared=False))
    print(f"{model_name} MAE:", mean_absolute_error(y_test, y_pred))
    print(f"{model_name} R2:", r2_score(y_test, y_pred))
    print()
# Evaluate SVM Regressor
print_metrics("SVM Regression", y_test, y_pred_svm_reg)

# Function to plot actual vs predicted values
def plot_actual_vs_predicted(y_test, y_pred, model_name):
    plt.figure(figsize=(10, 6))
    plt.scatter(y_test, y_pred, color='blue', label='Actual vs Predicted')
    plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], color='red', linestyle='--', label='Perfect Prediction')
    plt.xlabel('Actual DL_Throughput')
    plt.ylabel('Predicted DL_Throughput')
    plt.title(f'Actual vs Predicted DL_Throughput ({model_name})')
    plt.legend()
    plt.show()
```


Fig.12: Support Vector Regression Model

11.5K-Nearest Neighbor (KNN)

It is a non-parametric supervised learning classifier that classifies or predicts how to group a single data point using proximity. The K-Nearest Neighbors (KNN) regression model's training and evaluation process is shown in figure 13.

```
from sklearn.neighbors import KNeighborsRegressor
# Initialize and Train KNN Regressor
knn_reg = KNeighborsRegressor()
knn_reg.fit(X_train_scaled, y_train)
# Make Predictions with KNN Regressor
y_pred_knn_reg = knn_reg.predict(X_test_scaled)
# Function to evaluate and print metrics
def print_metrics(model_name, y_test, y_pred):
    print(f"{model_name} MSE:", mean_squared_error(y_test, y_pred))
    print(f"{model_name} RMSE:", mean_squared_error(y_test, y_pred, squared=False))
    print(f"{model_name} MAE:", mean_absolute_error(y_test, y_pred))
    print(f"{model_name} R2:", r2_score(y_test, y_pred))
    print()
# Evaluate KNN Regressor
print_metrics("KNN Regression", y_test, y_pred_knn_reg)
# Function to plot actual vs predicted values
def plot_actual_vs_predicted(y_test, y_pred, model_name):
    plt.figure(figsize=(10, 6))
    plt.scatter(y_test, y_pred, color='blue', label='Actual vs Predicted')
    plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], color='red', linestyle='--', label='Perfect Prediction')
    plt.xlabel('Actual DL_Throughput')
    plt.ylabel('Predicted DL_Throughput')
    plt.title(f'Actual vs Predicted DL_Throughput ({model_name})')
    plt.legend()
    plt.show()
```


Fig.13: K-Nearest Neighbors (KNN) regression model

Scaled test data is used to make predictions after the model has been trained using scaled training data. MSE, RMSE, MAE, and R-squared metrics are used to evaluate the model's performance. The correctness of the model is shown in a graph that compares the actual and predicted DL_Throughput values.

12. RESULTS

The goal of this study was to use supervised machine learning models to predict the throughput speed of mobile network. In order to capture real-time network performance indicators, user experience data, and stakeholder feedback, extensive walk tests were conducted. A variety of supervised learning models were employed, such as K-Nearest Neighbors (KNN) Regression, Support Vector Regression (SVR), Gradient Boosting Regression, Random Forest Regression, and Linear Regression. The processed dataset was used to train and test these models in order to predict the download bitrate (DL_bitrate) which is a crucial sign of network performance. The results obtained from different models are presented. Table 1 shows the evaluation of Random Forest regression model results.

Table 1: Summarized Random Forest Regression Evaluation Result

S/N	Evaluation Metrics	Results
1	Mean squared Error (MSE)	1060.40
2	Root Mean Squared Error (RMSE)	32.56
3	Mean Absolute Error (MAE)	6.79
4	R-squared (R^2) score	0.496

Figure 14 shows the simulated down link throughput (DL_Throughput) result from Random Forest Regression Model.

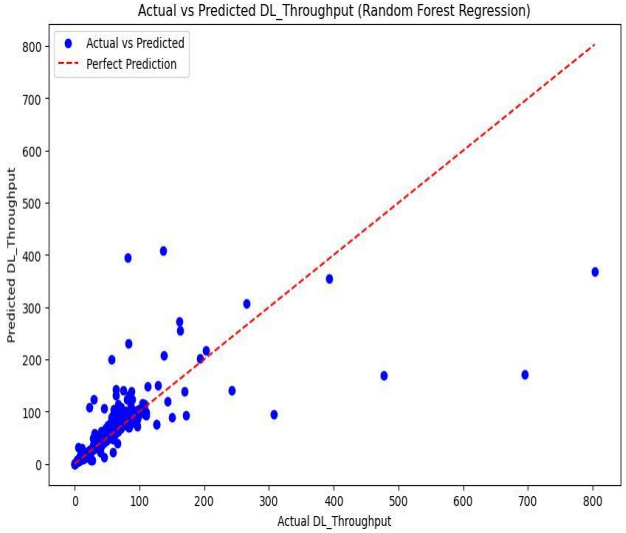

Fig.14: Plot of Actual Vs Predicted Down Link Throughput (DL_Throughput) from Random Forest Model

Table 2 is an evaluation result from Linear Regression Model.

Table 2: Linear Regression Evaluation Result

S/N	Evaluation Metrics	Results
1	Mean squared Error (MSE)	1341.61
2	Root Mean Squared Error (RMSE)	36.63
3	Mean Absolute Error (MAE)	8.38
4	R-squared (R^2) score	0.36

Figure 15 is simulated result of Actual Vs Predicted DL_Throughput for Linear Regression Model.

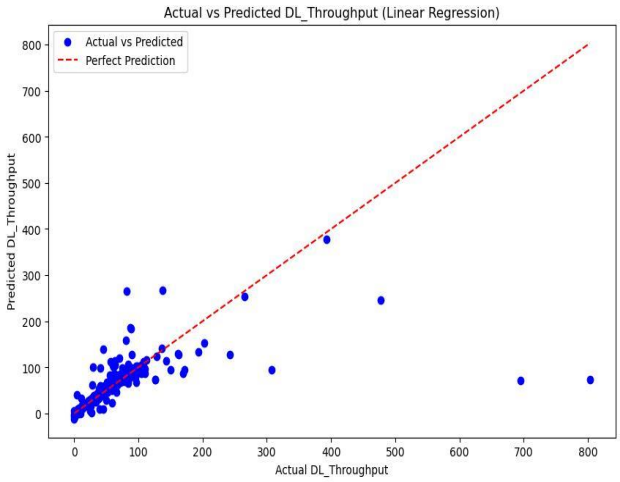

Fig.15: Simulated result of Actual Vs Predicted DL_Throughput for Linear Regression Model.

Table 3 is an evaluation result from Gradient Boosting Regression model. The focus is on the Mean Absolute Error (MAE) because it helps to determine which model produces the most consistent and accurate prediction.

Table 3: Gradient Boosting Regression Model result

S/N	Evaluation Metrics	Results
1	Mean squared Error (MSE)	1518.65
2	Root Mean Squared Error (RMSE)	38.97
3	Mean Absolute Error (MAE)	7.74
4	R-squared (R^2) score	0.28

Figure 16 is a graphical representation of the Actual against the Predicted Down Link throughput (DL_Throughput) for Gradient Boosting Regression Model. The graph displays the DL_Throughput prediction performance of the model.

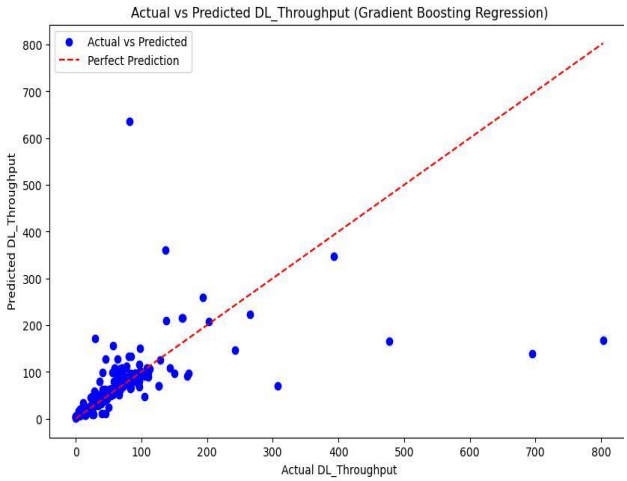

Fig. 16: Actual Vs Predicted DL_Throughput for Gradient Boosting Regression Model

Table 4 shows result evaluation from Support Vector Regression. It uses MAE to evaluate the accuracy of the model.

Table 4: Support Vector Regression results

S/N	Evaluation Metrics	Results
1	Mean squared Error (MSE)	1771.73
2	Root Mean Squared Error (RMSE)	42.09
3	Mean Absolute Error (MAE)	12.04
4	R-squared (R^2) score	0.16

Figure 17 is a simulated result from Support Vector Regression Model.

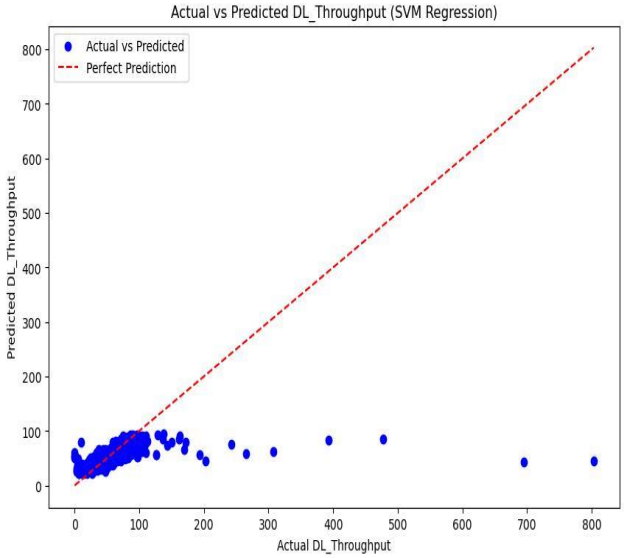

Fig. 17: Actual Vs Predicted Down Link Throughput (DL_Throughput) from SVM Regression

Table 5 is KNN Regression Result. The model forecasted the throughput speed of mobile networks using MSE, RMSE, MAE, and R^2.

Table 5: KNN Regression Evaluation Result

S/N	Evaluation Metrics	Results
1	Mean squared Error (MSE)	1860.67
2	Root Mean Squared Error (RMSE)	43.14
3	Mean Absolute Error (MAE)	13.25
4	R-squared (R^2) score	0.12

Figure 18 is a plot of the Actual Vs Predicted Downlink throughput speed.

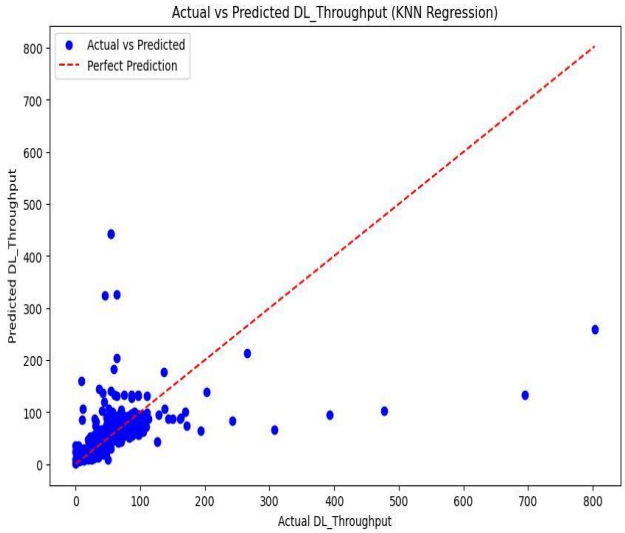

Fig.18: Actual vs Predicted DL_Throughput of KNN Regression model

13. DISCUSSION

The evaluation results of several regression models, including Random Forest, Linear Regression, Gradient Boosting, Support Vector Regression (SVR), and K-Nearest Neighbors (KNN), were analyzed to determine their effectiveness in predicting mobile network throughput speed (DL_Throughput) in Coventry University and Coventry city. Figure 14, Random Forest Regression (RFR) model demonstrates a high level of accuracy, with a Mean Absolute Error (MAE) of 6.79 and an R-squared (R^2) of 0.496, indicating 49.6% of the variance in the target variable. The model's performance is visualized in a plot where predicted values closely align with actual values, particularly at lower throughput levels, though accuracy diminishes at higher values.

Figure 15 is a Linear Regression model, while simpler, also performs adequately, with a plot showing good prediction alignment at lower throughput levels but reduced accuracy as throughput increases. Figure 16 is a Gradient Boosting Regression model, and is notable for handling complex data patterns, achieving an MAE

of 7.74, though still slightly less accurate than Random Forest Regression (RFR).

Support Vector Regression (SVR) which is figure 17 struggles with an MAE of 12.04 and a lower R^2 of 0.16, indicating poor prediction accuracy, especially at higher throughput levels. The KNN model in figure 18, shows some effectiveness in predicting DL_Throughput, though its accuracy diminishes with higher values, as reflected in its plot.
Overall, the Random Forest Regression model outperforms the others, making it the most reliable for predicting mobile network throughput speed in the Coventry University, and Coventry city.

14. CONCLUSION

This research demonstrated the significant potential of using supervised machine learning models to predict mobile network performance at Coventry University and Coventry City Center. Among the models evaluated, the Random Forest Regression (RFR) model emerged as the most reliable, offering the highest accuracy in predicting mobile network throughput speed, with a Mean Absolute Error (MAE) of 6.79 and explaining 49.6% of the variance in the target variable. This variance could be attributed to the tools used during data collections as some was omitted. This can be made better by utilizing a more sophisticated tool in data collection to ensure a more precise network prediction.

The findings underscore the value of predictive analytics in enhancing the quality and reliability of network services. By leveraging on machine learning, telecommunications providers can automate network management, improve resource allocation, and optimize performance, ultimately leading to better network efficiency, enhanced user experience, and reduced operating costs.

REFERNCES

1. Wahid H.; Abdul R. N.; Afzal S. A; (2022): Machine Learning for Performance Prediction in Mobile Network Management. November 2022, 21(OCT2022):101-107. DOI: 10.24191/jeesr.v21i1.013
2. Afroz, F., Subramanian, R., Heidary, R., Sandrasegaran, K., & Ahmed, S. (2015). SINR, RSRP, RSSI and RSRQ measurements in long term evolution networks. International Journal of wireless & Mobile Networks.
3. Thrane, J., Artuso, M., Zibar, D., & Christiansen, H. L. (2018). Drive test minimization using deep learning with Bayesian approximation. In *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)* (pp. 1–5). IEEE. https://doi.org/10.1109/VTCFall.2018.8690911
4. Moez All. (August, 2022). https://www.datacamp.com/blog/supervisedmachine-learning?dc_referrer=https%3A%2F%2Fwww.google.com%2F
5. Mold Stud, n.d. (2018). Telecommunications and machine learning: Improving network performance, from https://moldstud.com/articles/p-elecommunications-and-machine-learning-enhancing-network-performance.
6. "Coventry University." (2021). *Encyclopedia Britannica*. Retrieved from https://www.britannica.com
7. Elsherbiny, H., Abbas, H. M., Abou-zeid, H., Hassanein, H. S., & Noureldin, A. (2020, December). 4G LTE network throughput modelling and prediction. in GLOBECOM 2020-2020 IEEE Global Communications Conference (pp. 1-6). IEEE
8. Ćurguz, Z., Banjanin, M., & Stojčić, M. (2022). Prediction of user throughput in the mobile network along the motorway and trunk road. Science, Engineering and Technology, 2(2), 23-30.
9. Alho, L., Burian, A., Helenius, J., & Pajarinen, J. (2021, March). Machine learning based mobile network throughput classification. In 2021 IEEE Wireless Communications and Networking Conference (WCNC) (pp. 1-6). IEEE
10. Fauzi, M. F. A., Nordin, R., Abdullah, N. F., & Alobaidy, H. A. (2022). Mobile network coverage prediction based on supervised machine learning algorithms. IEEE Access, 10, 55782-55793.
11. Oussakel, I., Owezarski, P., & Berthou, P. (2019, November). Cellular uplink bandwidth prediction based on radio measurements. In Proceedings of the 17th ACM International Symposium on Mobility Management and Wireless Access (pp. 111-118).
12. Jin, R. (2015). Enhancing upper-level performance from below: Performance measurement and optimization in LTE networks.
13. Samba, A., Busnel, Y., Blanc, A., Dooze, P., & Simon, G. (2016, May). Throughput prediction in cellular networks: Experiments and preliminary results. In 1ères Rencontres Francophones sur la Conception de Protocoles, l'Évaluation de Performance et l'Expérimentation des Réseaux de Communication (CoRes 2016).
14. Jomrich, F., Herzberger, A., Meuser, T., Richerzhagen, B., Steinmetz, R., & Wille, C. (2018). Cellular Bandwidth Prediction for Highly Automated Driving. In Proceedings of the 4th International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS 2018) (pp. 121-132).

15. Raca, D., Zahran, A. H., Sreenan, C. J., Sinha, R. K., Halepovic, E., Jana, R., ... & Varvello, M. (2019, June). Empowering video players in cellular: Throughput prediction from radio network measurements. In Proceedings of the 10th ACM Multimedia Systems Conference (pp. 201-212).
16. Schmid, J., Schneider, M., HöB, A., & Schuller, B. (2019, November). A deep learning approach for location independent throughput prediction. In 2019 IEEE International Conference on Connected Vehicles and Expo (ICCVE) (pp. 1-5). IEEE.
17. Yue, C., Jin, R., Suh, K., Qin, Y., Wang, B., & Wei, W. (2018). LinkForecast: Cellular link bandwidth prediction in LTE networks. IEEE Transactions on Mobile Computing, 17(7), 1582-1594.
18. Bojović, B., Meshkova, E., Baldo, N., Riihijärvi, J., & Petrova, M. (2016). Machine learning-based dynamic frequency and bandwidth allocation in self-organized LTE dense small cell deployments. EURASIP Journal on Wireless Communications and Networking, 2016, 1-16.
19. Louis, F., & Mohamed, S. (2024). AI in Mobile Network Optimization: Enhancing Efficiency and Automation.
20. G-NetTrack Pro manual, 2024, https:// gyokovsolutions.com/manual-g-nettrack/
21. Hiren.(January 2024). https://www.tops-int.com/blog/why-has-python-become-popular-programming-language#:~:text=Python%20is%20a%20flexible%20programming,variety%20of%20enterprises%20and%20applications.