

ABSTRACT

For a developing city, effective road traffic management is critical. In recent years, the Intelligent Transportation System (ITS) has emerged as a critical field of research for addressing a variety of traffic issues and assisting smart cities with technological solutions. Because roads link people, the need of road safety is emphasized at all times. In order to properly manage traffic congestion on highways, technologies must be able to precisely estimate traffic. This project contains a survey of road data and traffic forecast for the roadways of Bengaluru, India, in order to build an innovative technology strategy to anticipate road traffic flow in terms of congestion levels using regression analysis.

Tom Tom Bengaluru Traffic Index provided the data for regression analysis. To forecast future traffic flow, we used a variety of regression models, including polynomial regression, multiple linear regression, random forest, and multilayer perceptron. Finally, an ensemble model is given that estimates the congestion level using the bagging and boosting techniques. This approach compares the results with multiple regression models using historical data and utilizes ensembling to forecast the average congestion level to anticipate short-term traffic flows. The examination of this data reveals that ensemble mode projections for distinct routes in Bengaluru, India, tend to approach closer to the real congestion level.

Finally, this model is hosted on a cloud platform so that it is available as an open-source website. Users can make well-informed decisions before commencing their road travel by interacting with this user-friendly website. The website provides us with information on the expected congestion levels of the roads in the near future.

Table of Contents

1. Introduction

2. Objective

2.1 Intelligent Transportation systems

3) Literature Review

3.1 Traffic flow prediction

3.2 Regression Analysis

3.2.1 Multiple linear Regression

3.2.2 Polynomial Regression

3.2.3 Random Forest

3.2.3 Multilayer Perceptron

3.3 Related Work

3.4 Ensemble Models

3.4.1 Bagging and Boosting

3.5 Rush Hour Analysis

4. Data collection

4.1 Data Visualisation

4.1.1 Frequency analysis of attributes

5. Solution methodology

5.1 Linear Regression

5.2 Polynomial Regression

5.3 Multilayer Perceptron

5.4 Random Forest

5.5 Ensemble Model

5.6 Rush Hour Modeling

5.7 Flask

5.8 SQLAlchemy

5.9 Heroku

5.10 Deploying Ensemble Model on Website

5.11 Working of Website

6 . Results

7. Conclusions

8. References

1. Introduction

Since the inception of vehicle transportation, traffic management has been a major concern, particularly in congested areas. London, New York, Los Angeles, Paris, and Beijing are among the world's most densely populated cities. Traffic congestion is also an issue in Indian cities such as Bangalore, Chennai, and Mumbai, which has been a source of anxiety for local residents. In fact, Bengaluru's road traffic is ranked 10th around the world in the category of congestion level for the year 2021. People's health is also harmed by the rising number of automobiles on the road. To overcome this problem, several methods and strategies have been attempted, such as traffic monitoring with current technology or classic light controls, but none appears to be enough. More study is needed in this area in order to develop better methods for reducing losses.

Traffic congestion has an impact on the global economy, the environment, and people's lives. It also adds to people's stress levels by causing travel delays. Not only that but halting and restarting autos burns gasoline at a higher rate. This results in air pollution, which contributes to global warming. This fuel use is also quite expensive. To ensure smooth traffic flow, thorough research and monitoring are required. Intelligent Transportation Systems (ITS) have been working with road traffic flow management over the past decade in order to make travel safer and more comfortable. As part of ITS, researchers work with traffic data. To extract information from traffic data, several sensors are employed, including pneumatic road tube counting, video vehicle identification, piezoelectric sensor, magnetic sensor, and others. These sensors are installed along the roadside and continually monitor the roadways, storing the data in a database. As a result, the database grows in size, requiring researchers to cope with large amounts of data. Road traffic data may be utilized to better understand vehicle behavior and better manage traffic. These data may also be used to forecast traffic flow on various city roadways. The ability to estimate traffic flow can be extremely useful in traffic management. There has been a lot of studies done to estimate traffic flow in the short term. However, there are just a few studies on long-term traffic flow forecasts. Nonetheless, because the situation and environment change in

different regions of the world, several studies and methodologies for detecting and forecasting road traffic are necessary. The use of big data enriches this study field and opens the door to further in-depth studies in the future. Making long-term traffic flow predictions will have a huge impact by saving people time and money.

2. Objectives

The objective of this project is to assist the Intelligent Transportation System (ITS) by using various regression techniques to predict the future traffic flow for safer road travel. It consists of three parts. They are:

- (i) To develop several machine learning models and perform a regression analysis to estimate the congestion level in percentage using historic data.
- (ii) To develop a web application that is user-friendly which takes in certain input parameters and displays the output after regression analysis. The application provides road travel recommendations based on the predicted value.
- (iii) To host the developed website on a server, so that the website is open source.

2.1. Intelligent Transportation System (ITS)

With the use of intelligent transportation technology, such as navigation cars, traffic signal control, number plate recognition, speed or CCTV systems, ITS provides innovative solutions to enhance traffic conditions. For decades, experts have been developing strategies to prevent commuters from traveling in the wrong direction to escape traffic congestion. ITS has recently opened a door to finding answers to these difficulties utilising cutting-edge technology. For traffic data mining, floating vehicle data (FCD) and GPS trajectory data have become essential data sources, allowing us to better analyze traffic patterns. For effective traffic management, these understandings are critical. Modern data scientists use these approaches to other machine learning methods to enhance human lives, such as lowering road noise, anticipating traffic demand, and even predicting road accidents. In addition, in the realm of ITS, traffic congestion prediction has become a popular topic. These projections can be utilised for effective route planning in the future to assist individuals in planning their journey. It is undeniable that ITS

systems based on cutting-edge technology enable transportation engineers to improve mobility and safety. As a result, ITS has the potential to save lives while also developing transportation infrastructure and technology for the future.

3. Literature Review

In the field of ITS, traffic data mining provides the foundation for organizing and analyzing road traffic data in order to enhance decision-making. Researchers will be able to provide solutions for cities and communities to improve their performance as a result of this. Much research has focused on training various machine learning algorithms (such as Deep Learning, K-Nearest Neighbor, Support Vector Machine, and others) to create models for forecasting traffic flow in the short and long term. This topic of research, on the other hand, involves the prediction of traffic velocity, gap distance, and traffic oscillation in order to enhance road traffic conditions in smart cities all over the world.

3.1 Traffic flow prediction

Predicting road traffic flow is critical for improved traffic control. To provide a location forecast, a simple traffic flow prediction strategy uses prior time-series traffic flow as input and trains machine learning methods. To increase prediction performance, many machine learning algorithms are constantly applied.

For various sorts of environments, multiple traffic flow forecast models are necessary. One strategy that produces good results in one area may not provide equivalent results in another. As a result, there is no way to come up with a global model that will work anywhere in the globe. As a result, this field of traffic forecasting study should be pursued further. As can be seen, the majority of the studies are focused on short-term traffic flow forecasting, which typically ranges from 10 to 30 minutes. As a result, more models concentrating on long-term traffic flow forecasts should be developed. Five regression approaches are used in this research to estimate traffic flow.

3.2 Regression Analysis

Regression analysis is a well-known technique for determining the relationship between variables. This variable can be used to represent characteristics or traits. A series of statistical processes is used to estimate the dependence or relationship between qualities. The regression models then employ the mathematical relationship between the variables to estimate a value for an independent data point.

3.2.1 Multiple linear Regression

Linear regression is a supervised learning technique that solves prediction problems using a linear approach. Linear regression tries to fit a line between the independent and dependent variables that describes the connection. When the dependent variable rises in tandem with the independent variable, it is said to be positively linked. Negatively connected refers to a connection that is the polar opposite. A linear regression line's purpose is to have the least amount of error for all data points. Multiple linear regression is a kind of linear regression in which there are more than one independent variable.

3.2.2 Polynomial Regression

Polynomial regression is a type of statistical regression analysis in which an n th degree polynomial in x represents the connection between the independent variable x and the dependent variable y . As a result, polynomial regression is regarded as a subset of multiple linear regression. When the Linear Regression Model fails to sufficiently explain the best conclusion and the points in the data are not captured by the Linear Regression Model, Polynomial Regression is utilized.

3.2.3 Random Forest

To produce predictions for classification and regression issues, random forest builds a large number of decision trees. It employs the ensemble learning approach, which entails training

many models or learners and integrating their outcomes to arrive at a prediction. Tin introduced the forest algorithm, which creates a forest with many trees, in 1995. The technique improved the generalization accuracy of a decision tree-based classifier, and its performance for identifying handwritten digits was highly promising. The results show that the method performs well in classification and regression with random characteristics and inputs. Bootstrap aggregating is another name for this approach.

3.2.4 Multilayer Perceptron

The aim of a multilayer perceptron, commonly known as an MLP, is to link many perceptrons together to create multiple outputs. Perceptron creates a weighted sum by feeding inputs into the processor along with weights and a bias. The output is generated by passing the weighted sum through an activation function. Only the linearly separable issue can be solved by a single perceptron. MLP has the capacity to address non-linearly separable real-world problems.

3.3 Related Work

In the field of ITS, traffic data mining provides the foundation for organizing and analyzing road traffic data in order to enhance decision-making. Researchers will be able to provide solutions for cities and communities to improve their performance as a result of this. Training different ML algorithms (such as DL, K-NN, SVM, and others) to construct models for estimating traffic flow in the short and long term has gotten a lot of attention. This field, on the other hand, investigates ways to predict traffic velocity, gap distance, and traffic oscillation in order to improve road traffic conditions in smart cities throughout the world.

Fankar Armash Aslam et al. (2015) described the web as the networking aid that is most commonly used and is claimed to be satisfactory by all users as it is a solution provider to all varieties of puzzles and problems. One of the important criteria to be looked upon during a web portal development is the appearance of the website. The better the appearance of the website, the larger the users are attracted towards it. Hence proper technology is required to implement

such well-structured websites. "python" and "flask" satisfy the technology needs of a good web portal.

Predicting road traffic flow is critical for improved traffic control. To provide a location forecast, a simple traffic flow prediction strategy uses prior time-series traffic flow as input and trains machine learning methods. To increase prediction performance, many machine learning algorithms are constantly applied. This study provides an ensemble model that outperforms the other common models in terms of accuracy.

3.4 Ensemble Models

In the prediction process, an ensemble model is a machine learning strategy that combines numerous different models. Such models are called base estimators. Base estimators are preferred over single estimators due to the following technical difficulties of single estimator:

- High variance: Model is not flexible in a way that it is sensitive only to the provided inputs to the trained features.
- Low accuracy: Entire training data to fit using one model or algorithm might not be good enough to meet expectations.
- Features noise and bias: One or a few features are relied heavily on while making a prediction.

Most ML technologies have constraints, and modeling with high accuracy seems difficult. The overall accuracy of the model might be increased if the merging of numerous models is done and developed. The combination may be accomplished by combining the outputs from each model with two goals in mind: lowering model error while retaining generality. Some strategies may be used to implement this type of aggregate. Ensembling is broadly classified into two types based on the flow of model processing. They are Bagging and Boosting ensembling.

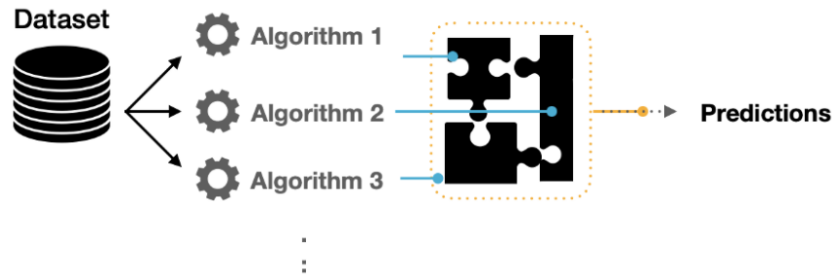


Figure 1: Diversifying the model predictions using multiple algorithms.

3.4.1. Bagging and Boosting Ensembling

Ensemble methods work on the concept of a combination of several decision tree classifiers to produce better performance of prediction which is not possible by a single decision tree classifier. The main principle lies in: a bunch of weak learners combining together to form a strong learner, which increases the accuracy of the model. Ensemble Bagging decreases the variance in the prediction by producing multi-sets of the original data to train using combinations and repetitions. Ensemble Boosting is an iterative technique that uses the principle of adjusting the weights of the observation based on its last predicted value. The major difference between the two techniques lies in the process flow wherein, the bagging technique undergoes parallel flow whereas boosting technique undergoes sequential flow.

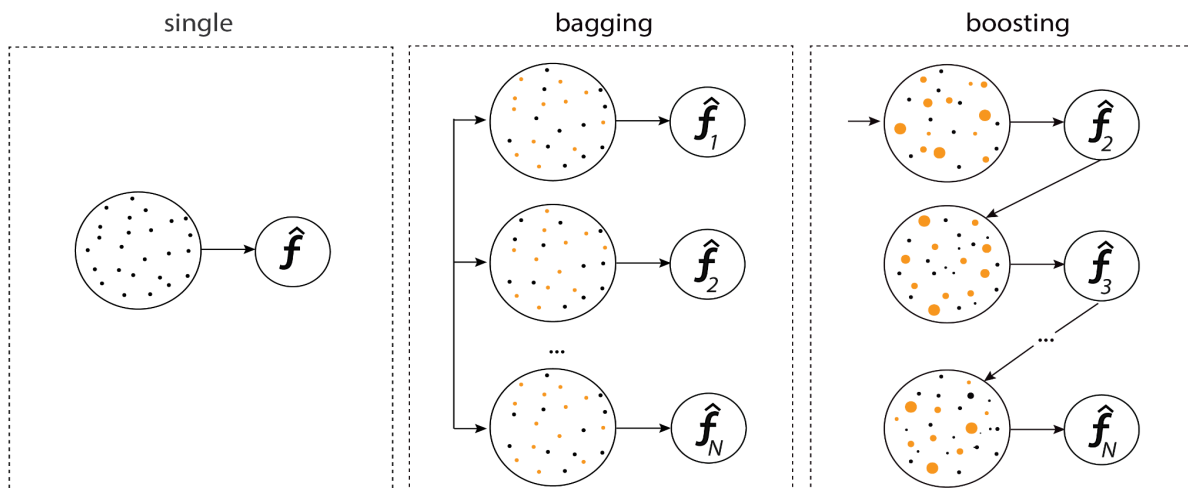


Fig 1.1 : Ensemble Bagging and Boosting Techniques

3.5 Rush Hour Analysis

Traffic's congestion level is not constant as it varies with time. With more and more increase in the working population, the more usage of roads and hence increase in congestion levels. The roads are most congested when people go to the workplace/return from the workplace. For the roads of Bengaluru, a similar pattern of traffic is evident. During the past 18th months, the pandemic has affected the roads of the world. There has been a change in the mobility patterns when compared to the pre-pandemic state. The statistics show that the roads were most congested from 9 :00 till 11: 00 in the morning and 17:00 till 20:00 in the evening. Moreover, Saturday seemed to be the most congested day of the week throughout the year. In 2021, a decrease in the congestion level was seen in Bengaluru. The data suggests that the average travel time was reduced by 2 mins per day.

For the roads of Bengaluru, the relationship between the time and the congestion levels over a week for the year 2021 is shown below. A 48% congestion level means that on average, travel times were 48% longer than during the baseline non-congested conditions. This means that a 30 min trip will take 14 mins longer to complete when the congestion level is at 48%.

	Sun	Mon	Tue	Wed	Thu	Fri	Sat
12:00 AM	3%	0%	0%	0%	0%	0%	1%
	0%	0%	0%	0%	0%	0%	0%
02:00 AM	0%	0%	0%	0%	0%	0%	0%
	0%	0%	0%	0%	0%	0%	0%
04:00 AM	0%	0%	0%	0%	0%	0%	0%
	1%	1%	1%	0%	1%	1%	2%
06:00 AM	6%	6%	7%	7%	7%	7%	8%
	16%	17%	17%	17%	17%	17%	18%
08:00 AM	22%	33%	31%	33%	32%	31%	29%
	26%	52%	49%	51%	49%	47%	41%
10:00 AM	33%	57%	54%	57%	54%	51%	49%
	42%	55%	52%	55%	53%	51%	60%
12:00 PM	47%	53%	52%	54%	53%	51%	69%
	45%	51%	49%	51%	50%	48%	69%
02:00 PM	41%	49%	45%	48%	47%	45%	65%
	41%	51%	47%	50%	49%	48%	64%
04:00 PM	45%	55%	50%	54%	53%	53%	68%
	48%	61%	56%	60%	59%	60%	73%
06:00 PM	55%	75%	69%	74%	73%	75%	85%
	59%	73%	67%	74%	71%	75%	86%
08:00 PM	47%	52%	47%	53%	52%	55%	65%
	36%	34%	32%	35%	35%	37%	45%
10:00 PM	24%	19%	18%	22%	21%	23%	28%
	10%	7%	7%	8%	8%	10%	13%

Fig 1.2 : Rush hour statistics over a week

4. Data collection

Bengaluru, India was used to collect real-time road traffic data. Bengaluru features a number of one-way streets, which are generally crowded. Bengaluru is ranked the world's tenth most crowded city as of 2021. As a result, this highway sees a considerable number of cars every day. Inductive loop detectors, also known as loop sensors, are often used to collect traffic data. The primary purpose of this study is to forecast traffic flow using historical traffic data. The real time traffic data was collected from TomTom Traffic index(https://www.tomtom.com/en_gb/traffic-index/bengaluru-traffic#statistics).

Congestion level, average vehicle speed, the total volume of light vehicles, the total volume of heavy vehicles, average number of lanes occupied by the vehicle along its travel route, and date and time characteristics are among the attributes considered for traffic forecast.

The data was collected from the 1st of January 2021 to the 25th of April in 2022. The average vehicle speed, the number of light and heavy vehicles on the roadways and the average lane utilisation of vehicles were all gathered as input data. Using past data, the model estimates the amount of congestion during the above-mentioned time frame.

There were 480 data points to map to congestion levels, corresponding to data points from January 1st, 2021 to April 25th, a total of 480 days. The tom tom traffic index was used to calculate the levels of congestion. The data for testing was gathered from April 1st to April 25th. The model predictions were based on these 25 data points, which correspond to 25 days between April 1st and 25th.

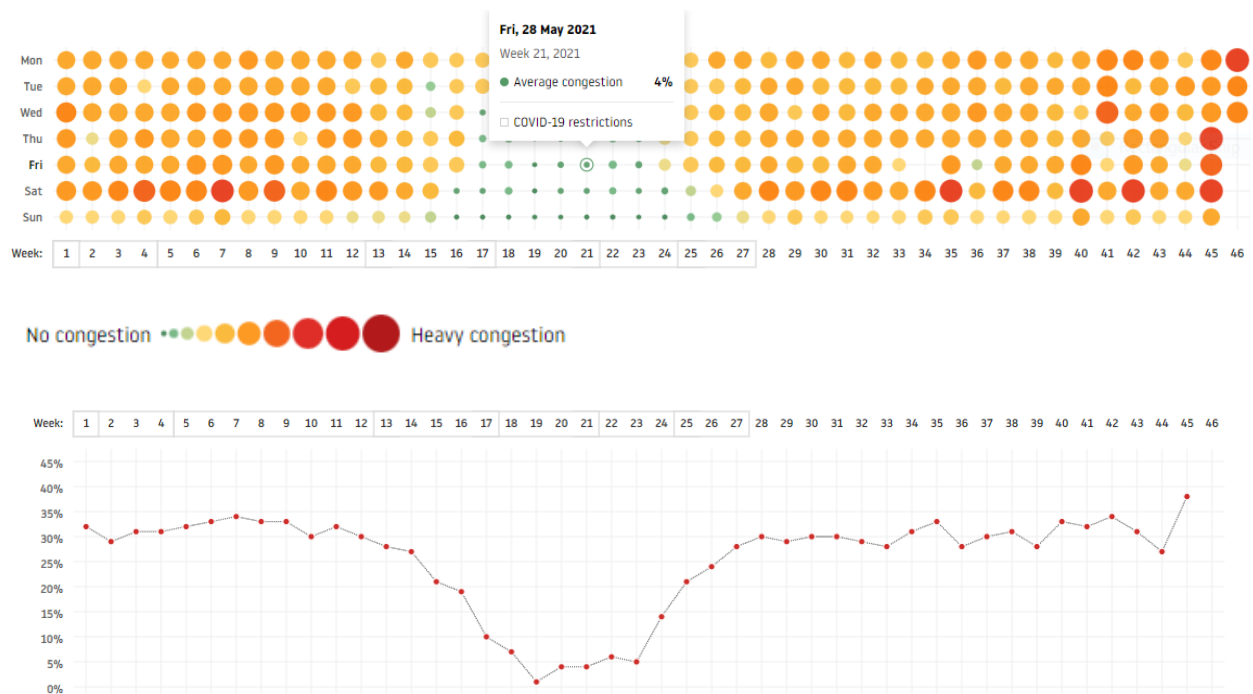


Fig 2: Average congestion levels on weekly basis starting from Jan 2021

4.1 Data Visualisation

For statistical data computation, the study was carried out using the Python programming language. The Pearson Correlation Coefficient was used to examine the correlation between each independent variable, and a histogram was displayed to examine the frequency characteristics of the input data.

4.1.1 Frequency analysis of attributes

Histograms of the traffic data attributes are presented below .

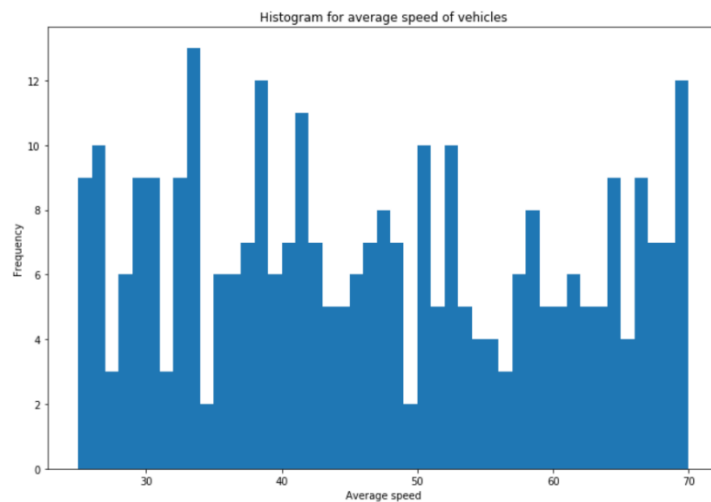


Fig 3: Histogram for average speed of vehicles

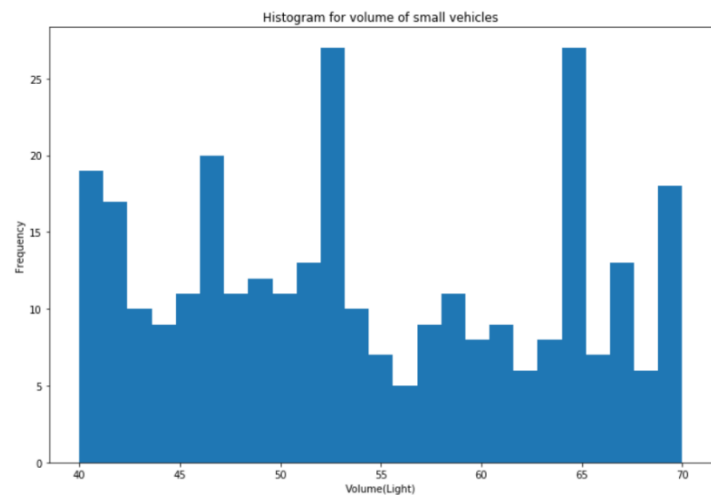


Fig 4: Histogram for volume of light vehicles

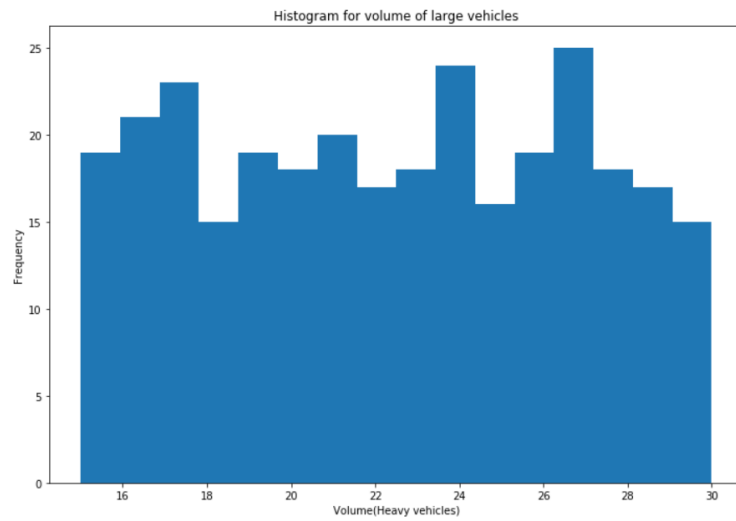


Fig 5: Histogram for volume of light vehicles

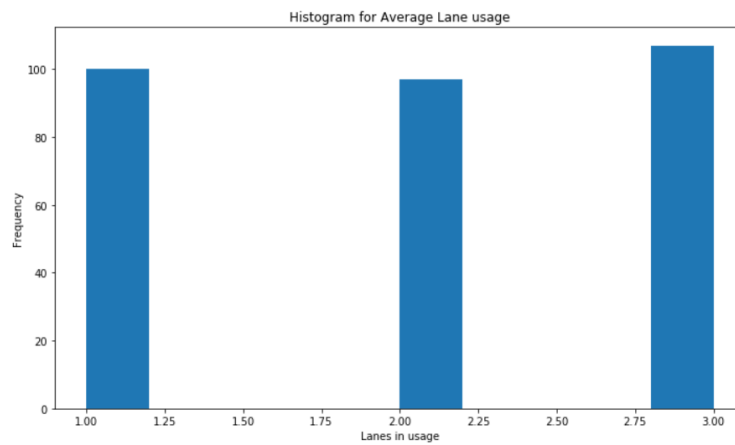


Fig 6: Histogram for volume of light vehicles

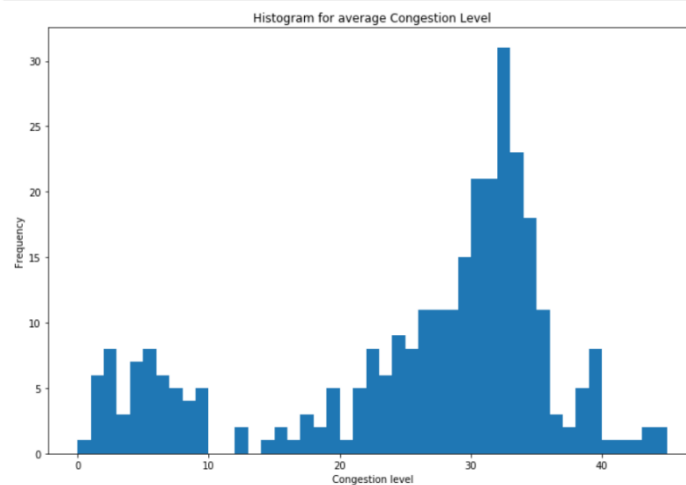


Fig 7: Histogram for average congestion level

Pearson correlation

The Pearson correlation coefficient is a measure of how closely two sets of data are related. It is the ratio of two variables' covariance to the product of their standard deviations; consequently, it is effectively a normalised measurement of covariance, with the result always falling between -1 and 1. The metric can only indicate a linear correlation of variables, similar to covariance, and excludes many other forms of interaction or association.

The Pearson interactions are shown in the below table.

Y1=Average speed of vehicles

Y2=Average volume of light vehicles

Y3=average volume of heavy vehicles

Y4=Average no of lanes used by the vehicle

Y5=The day of the month

Y6= Month of the year

	Y1	Y2	Y3	Y4	Y5	Y6	X
Y1	1	-0.098	-0.055	0.030	0.025	0.030	-0.076
Y2	-0.098	1	0.051	-0.183	-0.101	0.124	0.123
Y3	-0.055	0.051	1	0.018	-0.092	-0.023	0.026
Y4	0.030	-0.183	0.018	1	-0.028	-0.065	-0.091
Y5	0.025	-0.101	-0.092	-0.028	1	0.017	-0.007
Y6	0.030	0.124	-0.023	-0.065	0.017	1	0.040
X	-0.076	0.123	0.026	-0.091	-0.007	0.040	1

5. Solution methodology

Researchers can use road traffic data for anticipating road traffic flow thanks to ITS-related research. This is especially important in congested cities. This project seeks to make a contribution to the field of ITS for the creation of smart cities. This research also introduces a novel way to long-term traffic flow forecasting using regression models. The research also introduces an ensemble regression approach and compares the effectiveness of several regression models using real-world traffic data. The real-world data was gathered on a street in Bengaluru, India. Application of regression models to various city roadways in order to precisely determine the best-performing regression model in terms of long-term traffic flow forecast. The major objectives are summarized below:

1. Big data traffic may be seen and analysed.
2. From historical traffic data, determine the traffic pattern.
3. Apply regression analysis methods (such as Linear Regression, Polynomial Regression, Multilayer Perceptron, Random Forest, Boosting and Bagging ensemble) to real-time traffic data and evaluate their performance.
4. Propose an ensemble model and forecast the model output using the best ensemble approach.
5. To develop a web application which is user-friendly that takes in certain input parameters and displays the output after the regression analysis.
6. To host the developed website on a server, so that the website is open source.

5.1 Linear Regression

Linear regression is a supervised learning technique that solves prediction problems using a linear approach. Linear regression tries to fit a line between the independent and dependent

variables that describes the connection. When the dependent variable rises in tandem with the independent variable, it is said to be positively linked. Negatively connected refers to a connection that is the polar opposite. A linear regression line's purpose is to have the least amount of error for all data points.

Calculation of Multiple Linear Regression:

The general multiple linear regression model is given by:

$$\begin{aligned}
 y_i &= \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_k x_k + \epsilon_i \\
 y_i &= [x_1, x_2, \dots, x_k] \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_k \end{bmatrix} + \epsilon_i \\
 y_i &= \underbrace{x^T}_{(1 \times k)} \underbrace{\beta}_{(k \times 1)} + \epsilon_i
 \end{aligned}$$

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,k} \\ x_{2,1} & x_{2,2} & \dots & x_{2,k} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \dots & x_{n,k} \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_k \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{bmatrix}$$

$$\mathbf{y} = \mathbf{X}\mathbf{b} + \mathbf{e}$$

Finally the coefficients of the independent variables i.e. matrix b is calculated as follows:

$$\mathbf{b} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}.$$

The model is then fitted and the prediction is made on each new dataset using the coefficients for the calculated matrix elements of b and the about mentioned general equation.

5.2 Polynomial Regression

Polynomial Regression is a regression method that represents the relationship between a dependent (y) and independent (x) variable using an nth degree polynomial (x). Using polynomial regression, we want to find an nth degree polynomial function that is the closest approximation of our data points. The Polynomial Regression equation is illustrated below.

The general equation for m degree polynomial with n unknown variables is given by,

$$\begin{bmatrix} \sum_{i=0}^n x_i^{2m} & \dots & \sum_{i=0}^n x_i^{(m+2)} & \sum_{i=0}^n x_i^{(m+1)} & \sum_{i=0}^n x_i^m \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \sum_{i=0}^n x_i^{(m+2)} & \dots & \sum_{i=0}^n x_i^4 & \sum_{i=0}^n x_i^3 & \sum_{i=0}^n x_i^2 \\ \sum_{i=0}^n x_i^{(m+1)} & \dots & \sum_{i=0}^n x_i^3 & \sum_{i=0}^n x_i^2 & \sum_{i=0}^n x_i \\ \sum_{i=0}^n x_i^m & \dots & \sum_{i=0}^n x_i^2 & \sum_{i=0}^n x_i & n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \dots \\ a_m \end{bmatrix} = \begin{bmatrix} \sum_{i=0}^n y_i x_i^m \\ \dots \\ \sum_{i=0}^n y_i x_i^2 \\ \sum_{i=0}^n y_i x_i \\ \sum_{i=0}^n y_i \end{bmatrix}$$

The matrix 'A' is computed using matrix operations, and the values of $a_0, a_1, a_2 \dots a_m$ are the coefficients of the independent variables.

In this model, the degree of polynomial regression is 2. This means the highest power of any independent variable is 2. The number of input variables is 6. So, the number of coefficients of independent variables will be equal to 28 (which corresponds to 6 variables in their highest power + 6 variables taken two at a time + 6 variables of degree 1 + y-intercept).

5.3 Multilayer Perceptron

The aim of a multilayer perceptron, commonly known as an MLP, is to link many perceptrons together to create multiple outputs. Perceptron creates a weighted sum by feeding inputs into the processor along with weights and a bias. The output is generated by passing the weighted sum

through an activation function. Only the linearly separable issue can be solved by a single perceptron. MLP has the capacity to address non-linearly separable real-world problems. The above-mentioned activation function is a sigmoid function. This is denoted by sigma, and the formula is as follows: S(x) is the sigmoid activation:

$$S(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{e^x + 1}$$

The weighted total determined from the bias and the connectivity of inputs is x for a conventional multilayer perceptron.

Calculation of Multilayer Perceptron:

The inputs (X1, X2,..., Xn) can be thought of as a single-dimensional vector. The weighted matrix W = (W11, W12..., Wnn) is then multiplied by the input and the bias is calculated (B).

$$\begin{bmatrix} W_{11} & W_{12} & \dots & W_{1n} \\ W_{21} & W_{22} & \dots & W_{2n} \\ \vdots & & & \\ W_{N1} & W_{N2} & \dots & W_{Nn} \end{bmatrix} * \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{bmatrix}$$

The equation can be expressed as (W_{ij} * I + B)

Where I is the input vector and I = 1, 2,..., n; and j is the output vector and j = 1, 2,..., n. Over the weighted total, the activation function, often known as the sigmoid function, is applied. This generates the output, which processes certain hidden outputs in the case of several layers (H).

Hidden outputs, H = Sigma ((W_{ij} *I + B))

The equation's hidden outputs are subsequently processed by neurons or perceptrons to produce the final output. The final result is produced using a different bias and the sigmoid function.

The final output, O = Sigma((W_{ij} *H + B))

5.4 Random Forest

To produce predictions for classification and regression issues, random forest builds a large number of decision trees. It employs the ensemble learning approach, which entails training many models or learners and integrating their outcomes to arrive at a prediction. Tin provided the first technique for building a forest with many trees in 1995. The technique improved the generalization accuracy of a decision tree-based classifier, and its performance for identifying handwritten digits was highly promising. Leo updated the random forest method in order to increase its performance. The results show that the method performs well in classification and regression with random characteristics and inputs. For training or learning models, Random Forest uses the bagging approach. Bootstrap aggregating is another name for this approach.

Calculation of Random Forest:

Random forest algorithm uses feature bagging as an ensemble learning approach.

$S = s_1, s_2, s_3, \dots, s_n$ and $R = r_1, r_2, r_3, \dots, r_n$ are two examples of training sets and replies. By bagging samples over and again, the algorithm picks samples at random (T times).

For, $t = 1, 2, \dots, T$:

- Select S_t, R_t samples from S and R .
- Train the regression tree, f_t over S_t, R_t .

Following the learning process, predictions may be made using regression trees on s' .

$$\hat{f} = \frac{1}{T} \sum_{t=1}^T f_t(s')$$

In addition, the algorithm uses the majority votes based on the created trees for classifying. To learn about the prediction uncertainty, a standard deviation (SD) based on the predictions from created trees may also be determined. T is an independent parameter that may be calculated through cross-validation and represents the number of trees. The training and testing error begins to saturate after a certain number of trees have been fitted. By building a predictor that can distinguish between actual and synthetic data, the random forest may also be utilised for unsupervised learning. The approach is ideal for big volumes of data due to its ability to handle mixed attribute types and implicit variable selection.

5.5 Ensemble Model

In this report, we present an ensemble model for forecasting the traffic flow of Bengaluru city for a short period of time by incorporating all six regression techniques. In essence, the model estimates the weighted average projected value of several regression models and makes forecasts for future city traffic.

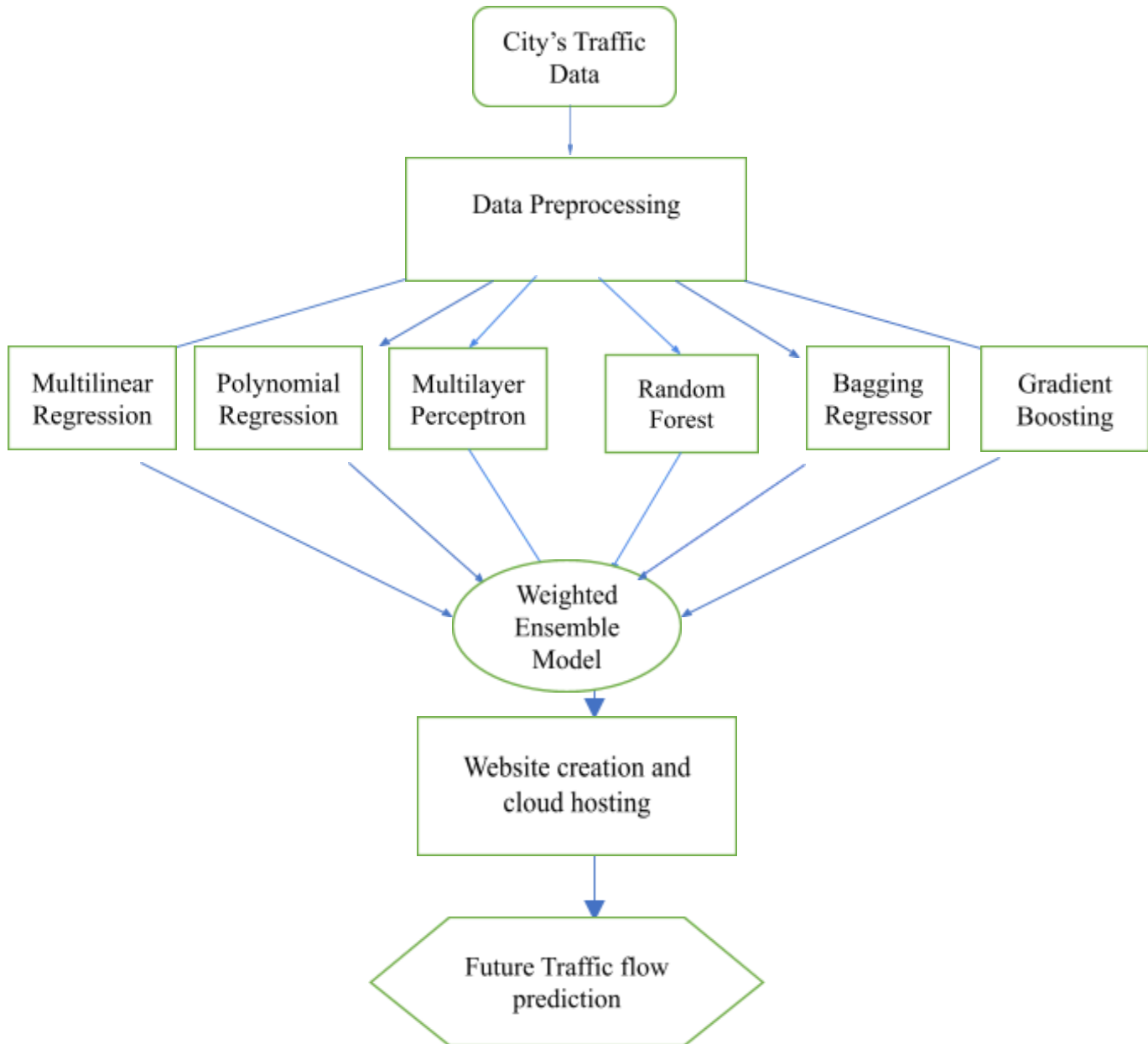


Fig 8: Solution Architecture, The model that has been proposed is based on Ensemble Learning Technique.

5.6 Rush Hour Modelling

It is difficult to model the mapping of congestion levels to all-time intervals of the day. It makes this project computationally harder and more intelligent and complex algorithms are needed to regress such time attributes. For modeling the rush hours, there were two splits done on the time interval after a detailed analysis of the time attributes, for the computation to become easier. Since most of the traffic was observed during the morning 9-11 and evening 5-8 time intervals, the splits were also 9-11 am and 5-8 pm. The average congestion level is taken as constant throughout the day. For the two time splits, the congestion level were slightly higher than the average congestion level on that day. This is evident from the data collected . Upon analysis, the congestion level of the morning split is almost 10% more than the average congestion level and almost 15 % higher for the evening split . This made the model be more precise in the predictions during these two time interval splits.

Rush hour modeling is not separately done as a whole part of the project. Instead, these two time interval splits are taken care of during the website creation and cloud hosting, where the time input is manually entered by the user. The usage of this time split comes in handy when the user plans an itinerary in the future when he wants to know the approximate congestion level during a particular time of the day. This part of the modeling was taken care of by adding a simple if-else statement to the program code. For example, if the user enters the input data as 9:15 am , then the program will automatically increase the predicted congestion level percentage by 10% . Similarly, if the input time is 7:15 pm then the program will automatically increase the predicted congestion level by 15%. If the time interval entered is other than these two splits, the value of the corresponding input variables is unchanged and hence the average congestion level is predicted without any changes. The prediction of the output congestion level is made after doing these small modifications to the input data.

5.7 Flask

For deploying the Machine learning model onto a website we will be concentrating the things centered on Python programming language and the deployment tools for this will be Flask and Heroku. The first and foremost thing to do is to create a Machine learning model in Python.

After training, The generated model file is stored as a pickle file which is a serialized format for storing objects using either Pickle or Joblib. Python pickle module's usage is mainly used for serializing and de-serializing a Python object structure. Pickling is a method to change a python object to a character stream. This concept is to contain all useful information of the character stream that may be used to reconstruct in another python script.

Then we can make the inference call (.predict()) by entering the required input parameters to predict the output value.

The model is queried to get the desired output for a test input after the trained model file is created. Calling the predict() function using the input data predicts the output. This model is then hosted on a web service. Hence the usage of Flask is done to create a web app.

Flask is a web application framework written in Python. Web Server Gateway Interface (WSGI) is termed as the universal standard for web application development in python. It provides us with a set of universal rules for the interface between the web applications and the web server. Werkzeug is a WSGI toolkit. It processes requests, response objects, and implements other utility functions. Web framework is built on top of it as it enables us to do so. One of the bases that the Flask framework uses is the Werkzeug. Combining templates with a certain data source to render dynamic web pages is done using a web templating system combines.

There are many more modules and frameworks which allow the users to build webpages using python. Such examples are Django, Flask, Web2py, Grok, TurboGears, etc. Flask and Django are the most commonly used. When compared to Flask, Django is easier to use but Flask provides us with the versatility to program with. The reason for selecting Flask is that it is a very light web framework that helps in creating web apps with minimal lines of code. Flask helps us in creating apps with less involvement of time and also is a good tool for beginners who want to learn to build web applications. Also, the framework depends completely on Python for coding-related stuff rather than relying on other dependent tools.

5.8 SQLAlchemy

Large data are stored using a database management system (DBMS). A database system includes the data of the DBMS and all the applications that go along with them. These collectively are sometimes abbreviated to just database. Most data in today's world is presented in rows and columns in a series of tables. This makes processing and analyzing the data much easier. Hence the data is mostly processed efficiently. Data is then updated, regulated, accessed, managed, and organized efficiently and easily. Structured Query Language (SQL) is the most widely used language for querying data. It is a programming language that most relational databases use to manipulate, query, and analyze data. A relational database's items are structured into tables with columns and rows. Relational database technology is the most structured and versatile approach to accessing structured data.

An object-relational mapper (ORM) provides an object-oriented layer between relational databases and object-oriented programming languages without having to write SQL queries. It standardizes interfaces, which reduces boilerplate and speeds up development. SQLAlchemy is a Python SQL toolkit and Object Relational Mapper that provides complete SQL capability and flexibility to application developers. It includes a full set of well-known enterprise-level persistence patterns, which are designed for fast and efficient database access and have been adapted into a simple Pythonic domain language.

SQLAlchemy is widely known for its object-relational mapper. object-relational mapper is an optional component that implements the data mapper pattern. It does with the ease in which classes get mapped to the database in open-ended, multiple ways, allowing the object model and database schema to be separated from the start.

SQLAlchemy's approach to problems is completely unique from that of most other SQL / ORM tools. Instead of hiding away SQL and object-relational details and all processes that concern it are fully exposed within a series of transparent and composable tools. The library takes on the control of automating any kind of unnecessary processes.

5.9 Heroku

Delivery of different services over the internet is most commonly called as cloud computing. It is closely related to the internet of things (IoT). Several services offered by the cloud include data storage, dedicated cloud servers, databases, software, and networking. Cloud-based storage makes it possible to save data to a remote database rather than storing the data on local storage devices or hard drives. As long as an electronic device has access to the web, it has access to the data and the software programs to run it. For large businesses that involve increasing efficiency, security, productivity, and cost savings the most commonly preferred option is cloud computing due to its highly valuable characteristics

Heroku is a container-based cloud Platform as a Service. Several service architectures platforms typically include programming-language execution environments, web servers, libraries, databases, operating systems, and connectivity to some platforms. For most of the developing platforms, Heroku is the most chosen cloud platform. It is a virtual environment that allows developers to deploy, manage, and scale modern technologies and applications. Its main focus lies in customer facilitation as it offers simple application development and deployment. Heroku gives developers the full freedom to list down their core ideas and helps them develop such technology products rather than worrying users about server maintenance, infrastructure, or hardware. More importance is given to user experience and the quality of the workspace the user is working in. Heroku supports a variety of programming languages for web application deployment, including Java, Node.js, Ruby, Scala, Clojure, Python, PHP, and Go. For small applications, Heroku offers a free service model. In circumstances where more complicated company demands must be met, tiered service packages are also available.

5.10 Deploying ensemble learning model on website

Firstly the ensemble model to estimate congestion level is developed. Once confident predictions were obtained, it was integrated with a web application framework, i.e, Flask. While developing the framework, the communication between the frontend part and the backend part had been taken care of. The frontend part deals with end user experience and the backend is where the logic works. Finally, it was deployed on a server and here the server that was used is Heroku.

Users then access the web application using the URL. The pipeline to deploy the ensemble learning model on website and hosting the website on the cloud platform is shown.

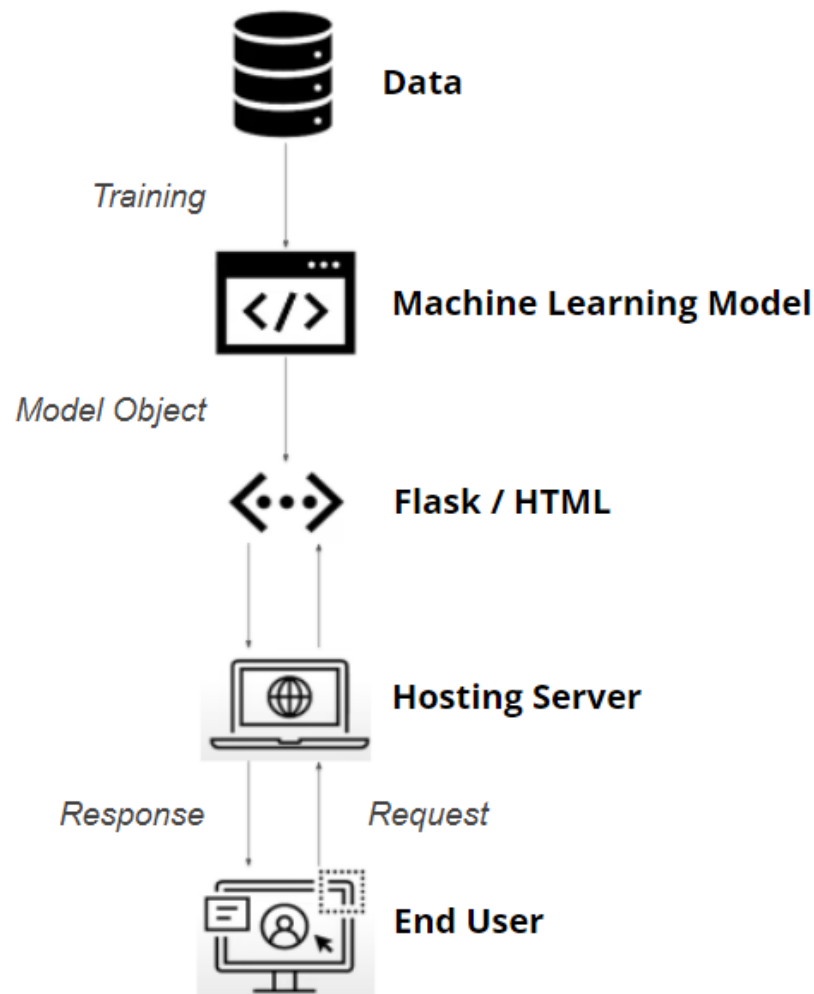


Figure 5.9.1: Pipeline to deploy machine learning(ensemble) model on website.

The project structure includes three main files. `model.py` file contains code for the machine learning model to predict the soil nutrients. `app.py` contains Flask application (backend) that receives input from the frontend, computes the predictions based on the machine learning models, and returns it. HyperText Markup Language (HTML) files contain the frontend templates to allow users to enter the inputs and display the output.

After a web service that runs on the local system was developed, it was necessary to host the web service on the Heroku server so that someone else who wants to consume the service can. The following steps were followed to deploy the local website on Heroku:

1. Created a “Procfile” which contains the following code inside it:

```
web: gunicorn app:app
```

2. Created a "requirements.txt" file using the following code:

```
pip freeze > requirements.txt
```

3. Installed the Heroku CLI (Command-Line Interface).

4. Performed login with the following command:

```
heroku login
```

5. Created a new empty application on Heroku, along with an associated empty Git repository. Running this command from the app’s root directory, the empty Heroku Git repository was automatically set as a remote for the local repository.

```
heroku create -a traffic-congestion-predictor
```

6. Checked if the remote had been set successfully by running the following command:

```
git remote -v
```

The following response appeared in the terminal:

```
heroku https://git.heroku.com/traffic-congestion-predictor.git (fetch)
```

```
heroku https://git.heroku.com/traffic-congestion-predictor.git (push)
```

7. Pushed branch to the new heroku remote:

```
git init
```

```
git add .
```

```
git commit -am "make it better"
```

```
git push heroku main
```

8. Tracked the progress of the deployment being displayed on the terminal. The website was successfully hosted on the Heroku server with the URL, and can be accessible any time:

<https://traffic-congestion-predictor.herokuapp.com/>

5.11 Working of the Website

The website contains a single page where the user interface will contain 4 slider bars that control four input parameters i) Average speed of the vehicles ii) Average volume of light vehicles iii) Average volume of heavy vehicles and iv) Average number of lanes used by the vehicle. Along with these input parameters, date and time inputs are collected manually from the user. The prediction of the congestion level at that particular time of the day is displayed as output instantly.

This website also gives recommendations on whether to commence our travel on that particular date and time. If the predicted congestion level is less than 20 %, the roads are slightly congested and the website recommends us to travel at our convenience. If the predicted congestion level is more than 20 % but less than 25 %, then the roads are moderately congested. If the predicted congestion level is more than 25%, the roads are heavily congested, and the website recommends us to travel on a later date or time.

6. Results

Evaluation criteria

A forecasting method's prediction accuracy is measured by the mean absolute percentage error (MAPE). It is also known by the name of mean absolute percentage deviation (MAPD) sometimes in statistics. The accuracy is usually stated as a ratio, that can be calculated using the following formula:

$$\text{MAPE} = \frac{100}{n} \sum_{t=1}^n \left| \frac{A_t - F_t}{A_t} \right|$$

where A_t is the actual value and F_t is the forecast value. Their difference is divided by the actual value A_t . The absolute value in this ratio is summed for every forecasted point in time and divided by the number of fitted points n .

This research looks on the suitability and capabilities of regression models for predicting short-term traffic flow. Existing python library tool tests were used to represent them. Multiple

Linear Regression, Polynomial Regression, Multilayer Perceptron, Random Forest, and Proposed Ensemble Model were employed as regression models.

Multiple linear regression model

The data was fitted to a multiple linear regression model. The following images show the data visualisation of predicted VS actual data points.

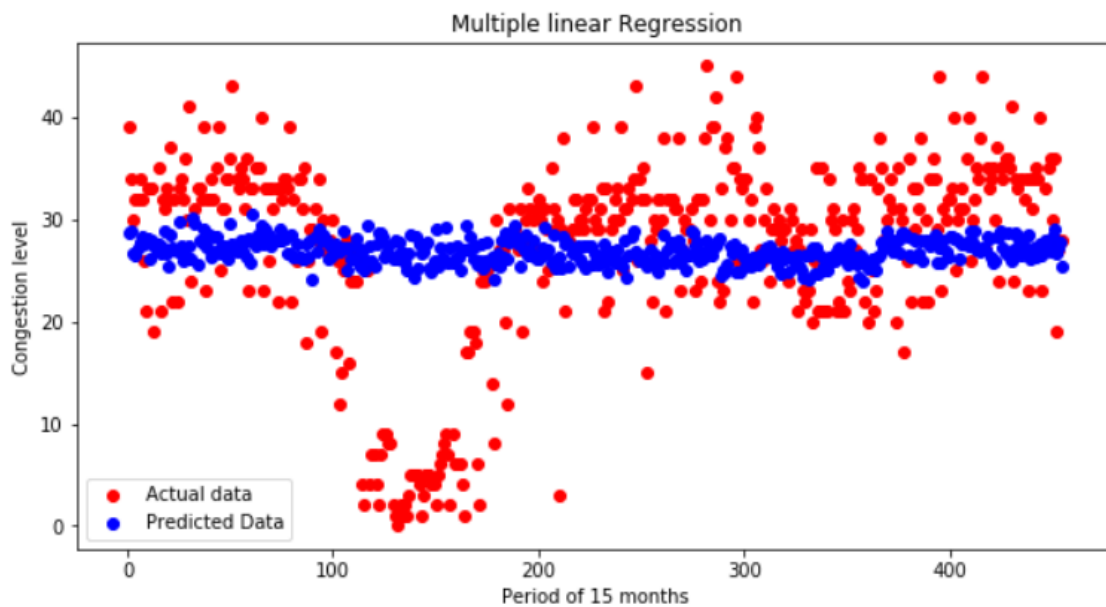


Fig 8 : Multiple Linear Regression (Actual vs predicted Jan 2021 till March 2022)

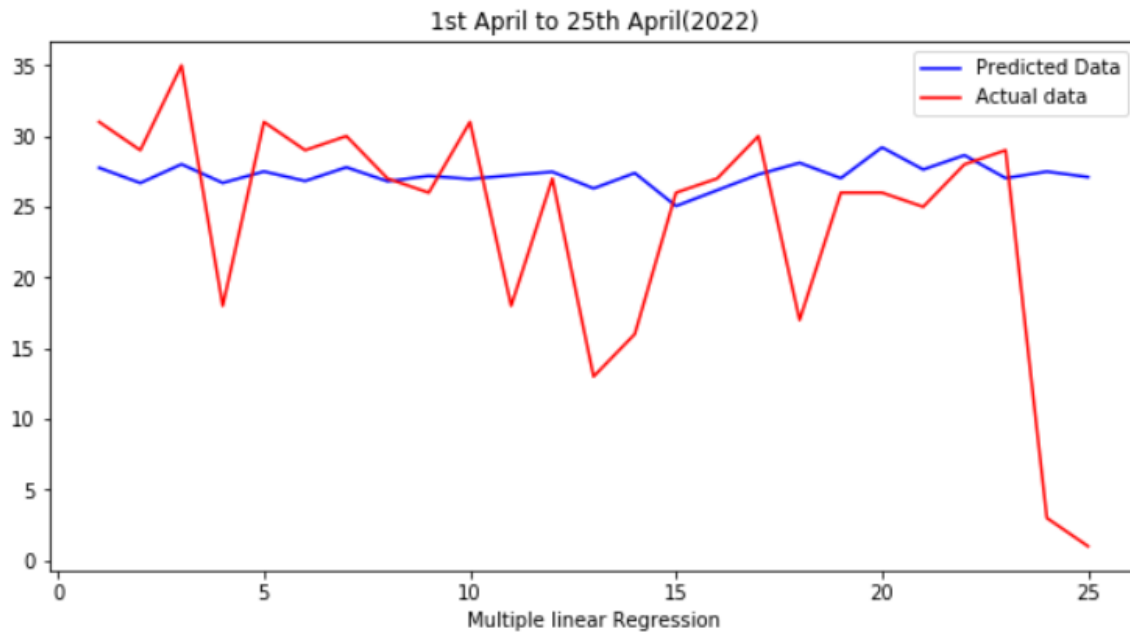


Fig 9 : Multiple Linear Regression (Actual vs predicted Apr 1st till Apr 25th 2022)

The Mean Absolute Percentage Error came out to be 19.88683923 %.

Polynomial regression model

The data was fitted to a polynomial regression model of degree 2. The following images show the data visualisation of predicted VS actual data points.

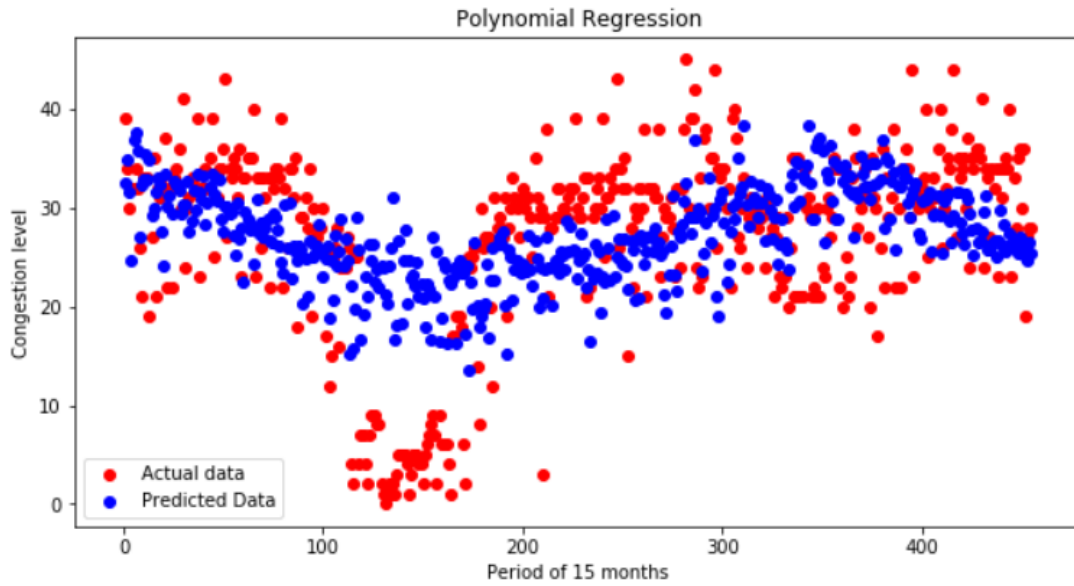


Fig 10 : Polynomial Regression (Actual vs predicted Jan 2021 till March 2022)

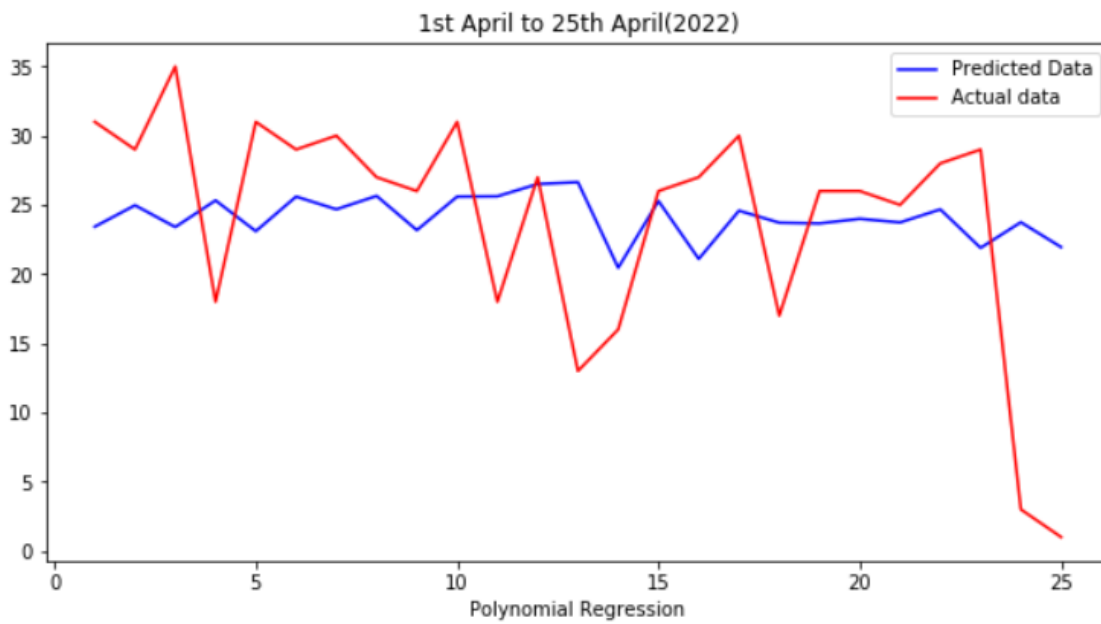


Fig 11 : Multiple Linear Regression (Actual vs predicted Apr 1st till Apr 25th 2022)

The Mean Absolute Percentage Error came out to be 19.8704253%.

Multilayer Perceptron regression model

The data was fitted to a multilayer perceptron regression model with 4 hidden layers of 64 neurons each. The following images show the data visualisation of predicted VS actual data points.

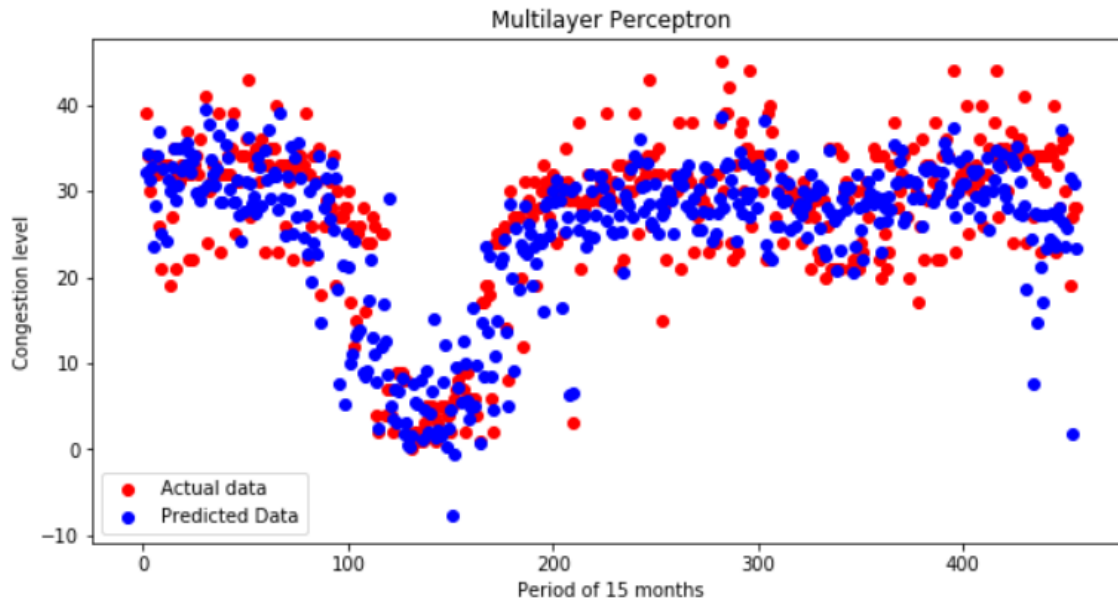


Fig 12 : MLP Regression (Actual vs predicted Jan 2021 till March 2022)

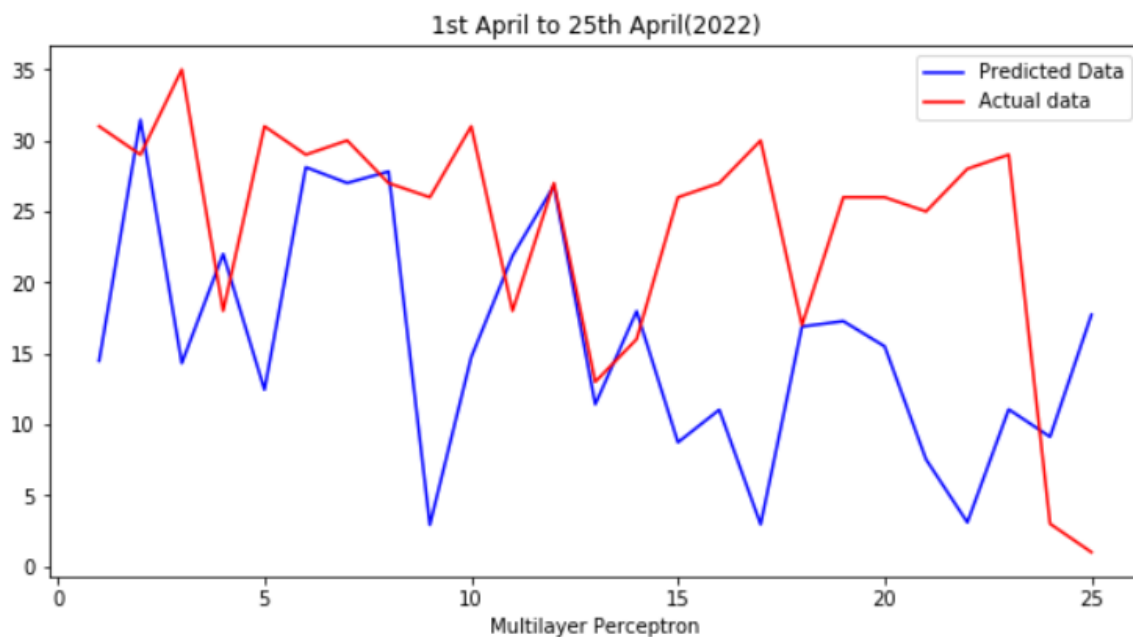


Fig 13 : MLP Regression (Actual vs predicted Apr 1st till Apr 25th 2022)

The Mean Absolute Percentage Error came out to be 20.51858027 %.

Random Forest regression model

The data was fitted to a Random Forest regression model with 600 branch nodes. The following images show the data visualisation of predicted VS actual data points.

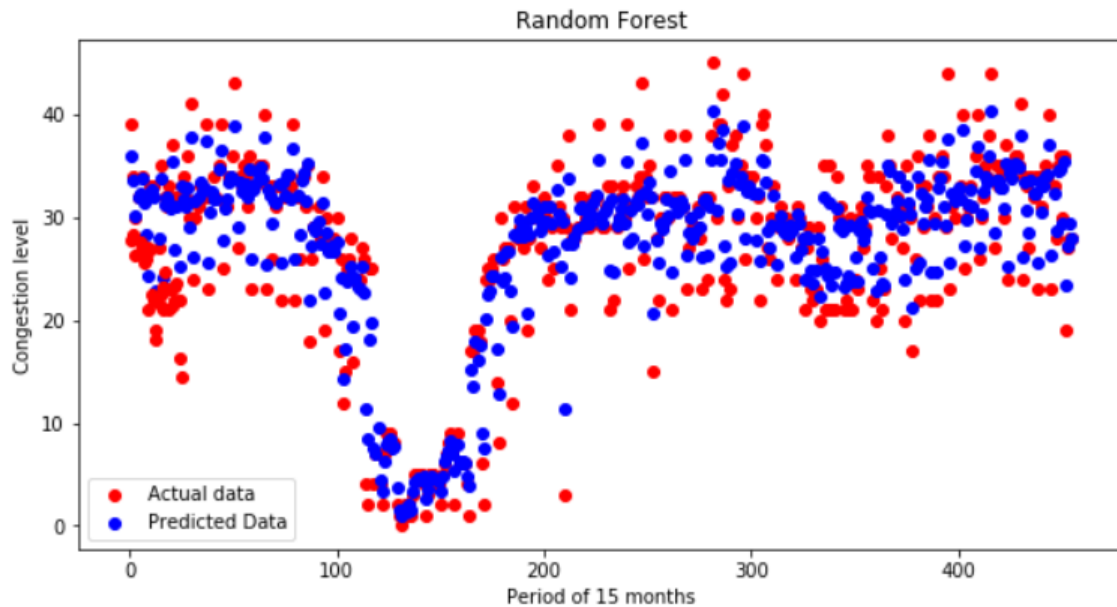


Fig 14 : Random Forest Regression (Actual vs predicted Jan 2021 till March 2022)

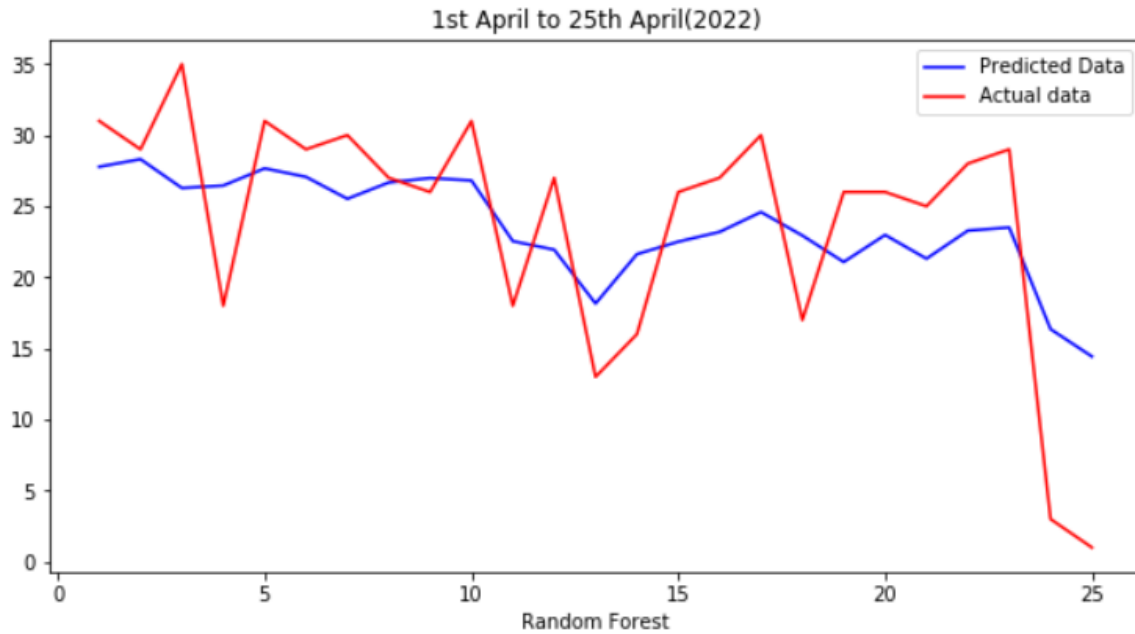


Fig 15 : Random Forest Regression (Actual vs predicted Apr 1st till Apr 25th 2022)

The Mean Absolute Percentage Error came out to be 16.69985031%.

Bagging regression model

The data was fitted to a bagging regressor model which had the base model as XGB Regressor. The following images show the data visualisation of predicted VS actual data points.

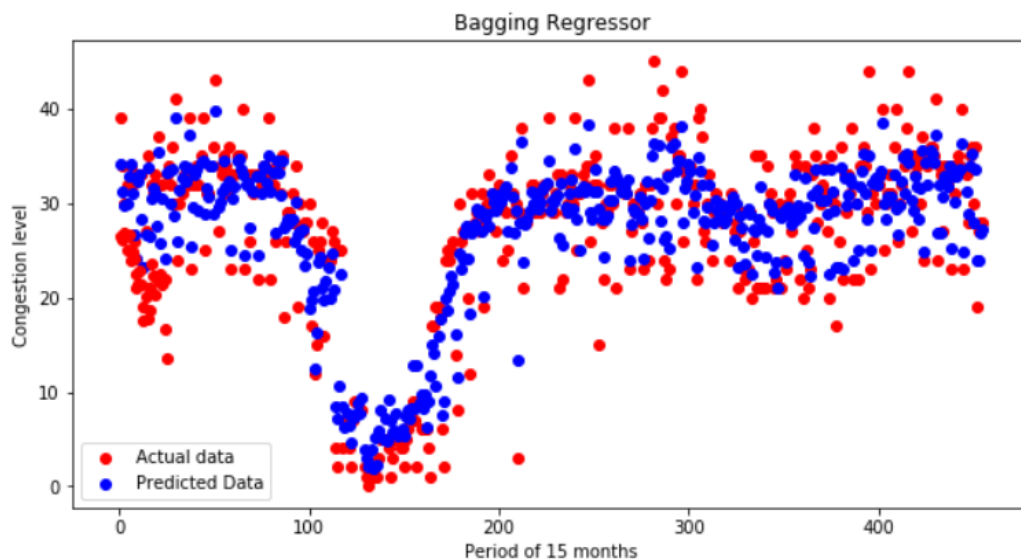


Fig 16 : Bagging Regression (Actual vs predicted Jan 2021 till March 2022)

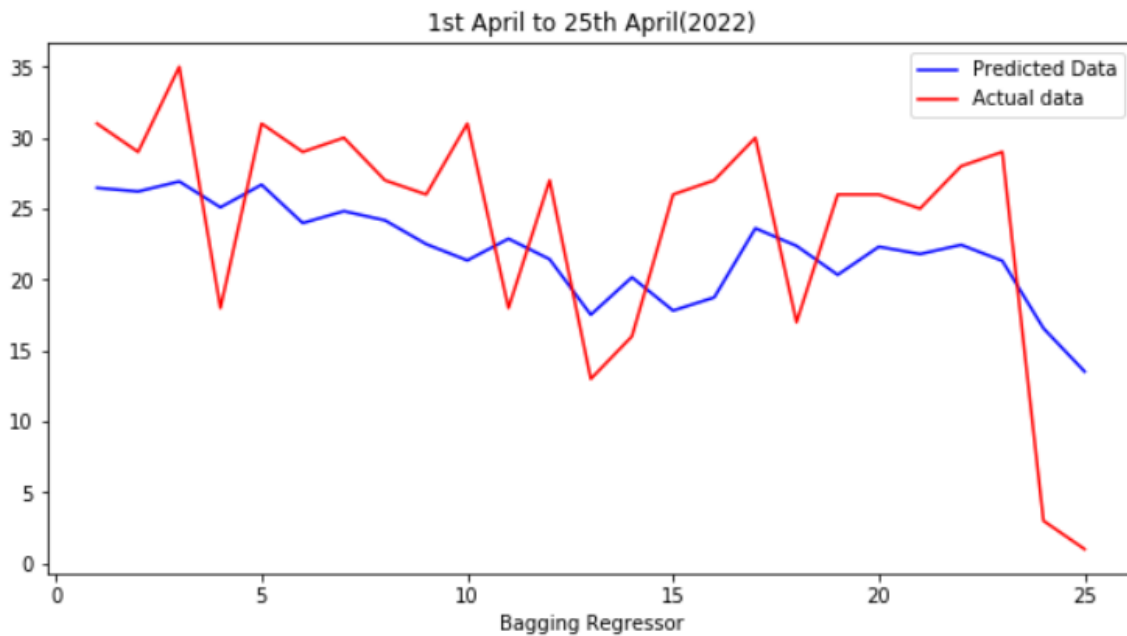


Fig 17 : Bagging Regressor (Actual vs predicted Apr 1st till Apr 25th 2022)

The Mean Absolute Percentage Error came out to be 19.85185682%.

Gradient Boosting Regression Model

The data was fitted to a boosting regressor model which uses the logic of gradient boosting. The following images show the data visualisation of predicted VS actual data points.

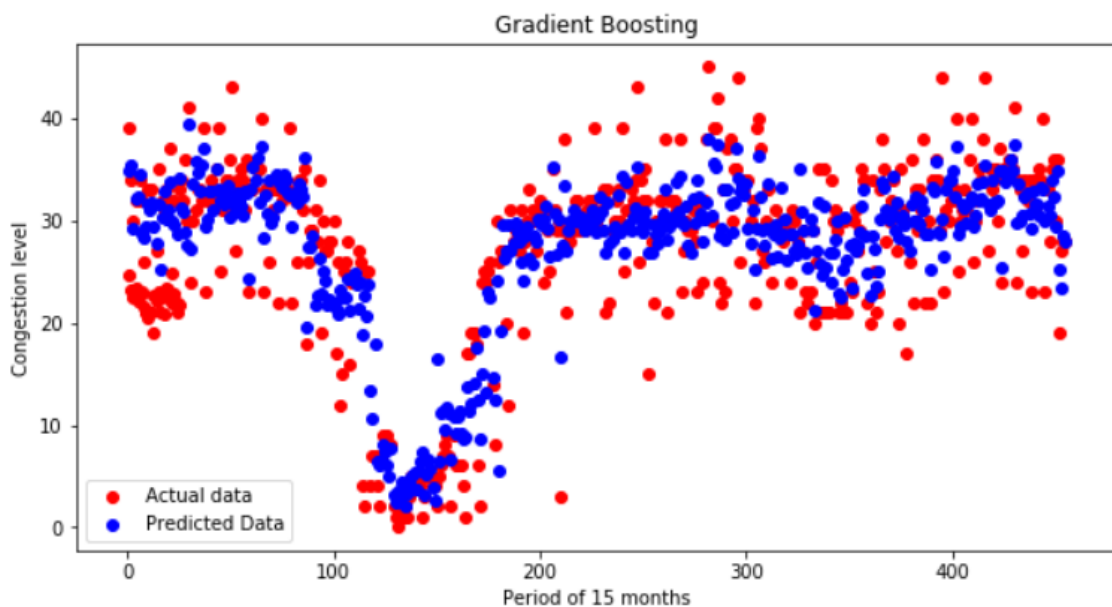


Fig 19 : Boosting Regression (Actual vs predicted Jan 2021 till March 2022)

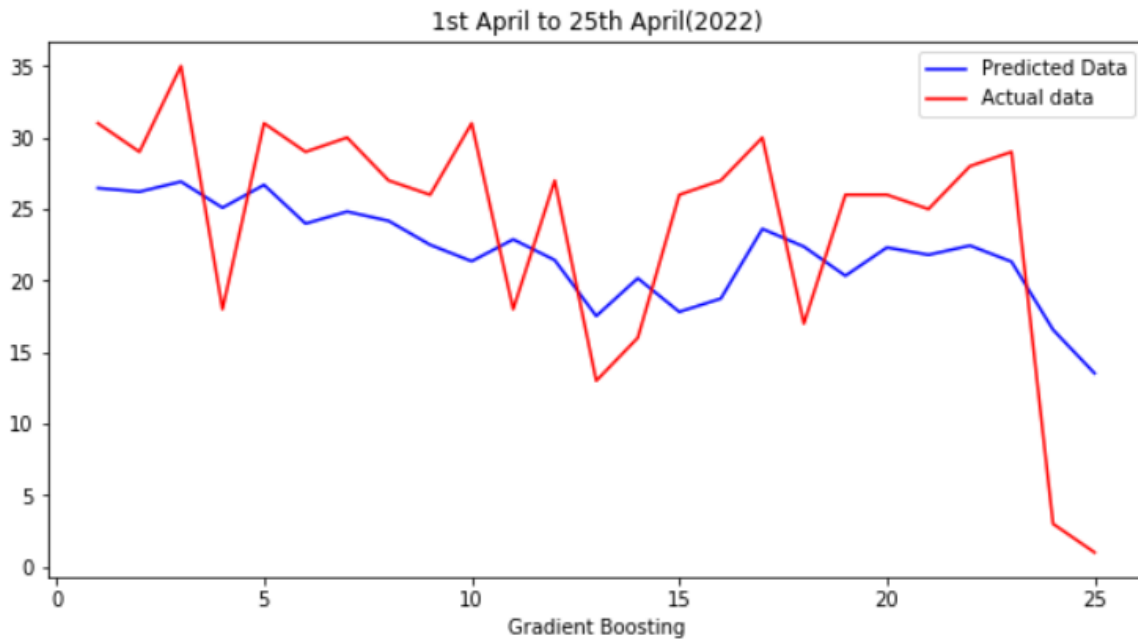


Fig 20 : Boosting Regression (Actual vs predicted Apr 1st till Apr 25th 2022)

The Mean Absolute Percentage Error came out to be 17.36173638%.

Ensemble regression model

The data was fitted to an ensemble regression model with a weighted average . The weights of 0.75,1,0.25,7,0.5,0.5, were assigned to the linear, random forest, multiple, MLP regression, Bagging and Boosting regression models respectively, based on MAPE. The following images show the data visualization of predicted VS actual data points.

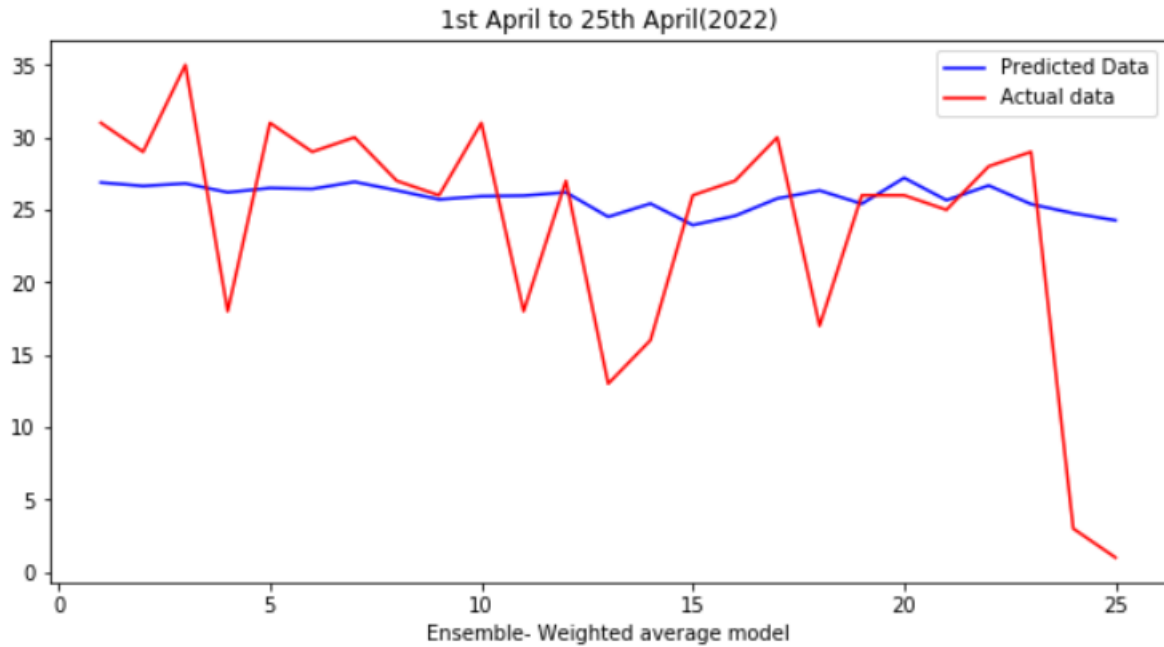


Fig 21 : Weighted Ensemble model Regression (Actual vs predicted Apr 1st till Apr 25th 2022)

The Mean Absolute Percentage Error came out to be 15.73506292%.

Overall Analysis

The comparison of all the models were made and the results are shown in the below line graph.

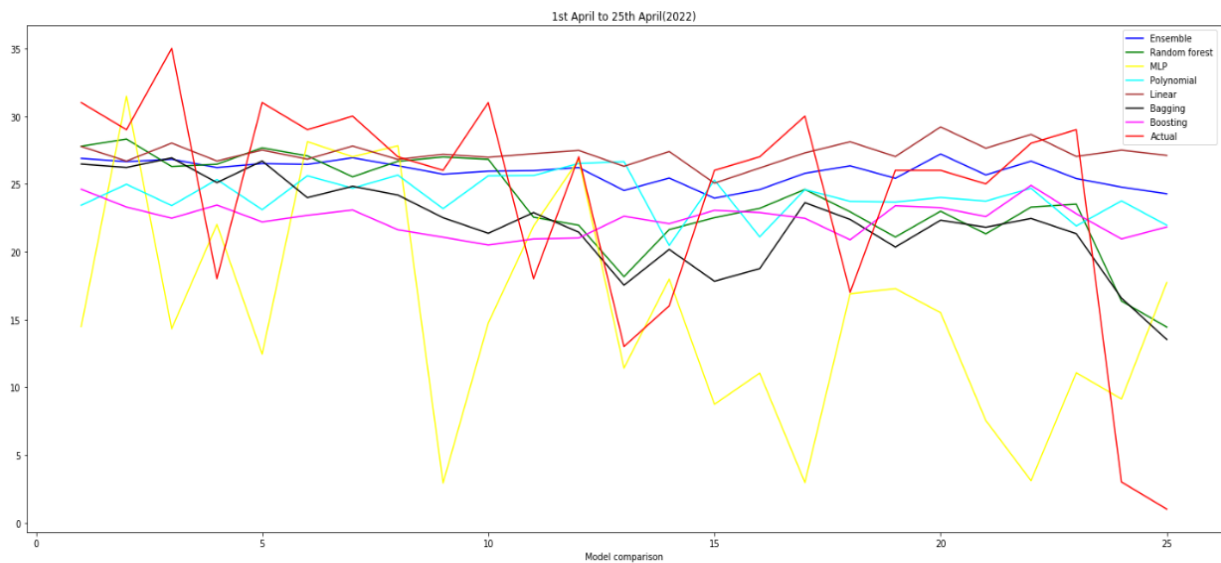


Fig 22 : All Model comparison (Actual vs predicted Apr 1st till Apr 25th 2022)

Website Outputs

HTML was used for frontend development. It is the standard markup language for documents designed to be displayed in a web browser. The structure, design, behavior, and content of everything seen on browser screens when the web application is opened up, is implemented in the frontend. The sliding bars that control the input parameters were implemented on HTML .

```
<div class="form-group">
  <label for="speed">Average Speed of Vehicles: <span id="speed_val">45</span></label>
  <input type="range" class="form-control-range slider" name="speed" id="speed" min="30" max="60">
</div>

<div class="form-group">
  <label for="avg_light">Average Volume of Light Vehicles: <span id="avg_light_val">55</span> </label>
  <input type="range" class="form-control-range slider" name="avg_light" id="avg_light" min="45" max="65">
</div>

<div class="form-group">
  <label for="avg_heavy">Average Volume of Heavy Vehicles: <span id="avg_heavy_val">24</span> </label>
  <input type="range" class="form-control-range slider" name="avg_heavy" id="avg_heavy" min="18" max="30">
</div>

<div class="form-group">
  <label for="avg_lanes">Number of Lanes used: <span id="avg_lanes_val">2</span> </label>
  <input type="range" class="form-control-range slider" name="avg_lanes" id="avg_lanes" min="1" max="3">
</div>
```

Fig 23: HTML implementation of the sliding bars

The Website also recommends whether to plan our road travel or not. It is done based on the predicted congestion levels. The following table shows the criteria that are being used for recommendation.

Congestion Level (L)	Recommendation
$L \leq 20$	Slightly congested, but the road journey can commence.
$20 < L < 25$	Moderately congested.
$L \geq 25$	Highly congested, Plan your journey at a later date or time.

Table 1: Recommendations based on congestion levels

Heroku Website Outputs

Congestion Level Prediction

Average Speed of Vehicles: **40**



Average Volume of Light Vehicles: **58**




Average Volume of Heavy Vehicles: **21**



Number of Lanes used: **2**



Date and Time: 

Submit

Congestion Level in percentage(%): **36.28**

Highly Congested. Plan your journey on a later date or time.

Fig 24: Website interface, output, and recommendations

7. Conclusions

Intelligent transportation will make our everyday transit safer, greener, and more convenient in the not-too-distant future. ITS can help to minimize the number of traffic accidents, avoid injuries and fatalities, improve traffic flow, and save money on gas. As a result, we must come up with innovative ways to strengthen and improve future transportation infrastructure. This article presents a brief overview of traffic flow forecasting in recent years and identifies a simple yet successful method for forecasting future traffic flow in Bengaluru, India. In this paper, we have analyzed the traffic data from the month of January 1st, 2021 till April 25th 2022 that was collected from tom tom traffic index. We used regression models to anticipate traffic flow using pre-processed road traffic data. Each model produced patterns in order to forecast future traffic flow. Five different regression models were examined in terms of performance. The weighted ensemble regression model, which was developed in the study, performed the best among them, with the error rate as low as 15.7 %. This ensemble model is then hosted on a website.

Further more, the website has also been a success. The user-friendliness of the website allows the user to plan his road travel beforehand so that no time is lost unnecessarily during the travel. Since the website is open source, anyone can access it and people can be well informed about the road conditions in the upcoming future.

The future scope of this project is vast and can be generalised further. This project mainly focuses on the roads of Bengaluru within a 200 km loop. The area of this study can be expanded and can be generalised for other cities and even countries worldwide. The users of this website can benefit by planning their upcoming itinerary very well and can get to know about their surrounding roads. This project I believe will contribute to the ITS.

8. References

1. Alam, Ishteaque, et al. "Pattern mining from historical traffic big data." *2017 IEEE Region 10 Symposium (TENSYP)*. IEEE, 2017.
2. Baratian-Ghorghi, Fatemeh, and Huaguo Zhou. "Traffic control devices for deterring wrong-way driving: Historical evolution and current practice." *Journal of traffic and transportation engineering (English edition)* 4.3 (2017): 280-289.
3. Altintasi, Oruc, Hediye Tuydes-Yaman, and Kagan Tuncay. "Detection of urban traffic patterns from Floating Car Data (FCD)." *Transportation research procedia* 22 (2017): 382-391.
4. Zhao, Shuangming, Pengxiang Zhao, and Yunfan Cui. "A network centrality measure framework for analyzing urban traffic flow: A case study of Wuhan, China." *Physica A: Statistical Mechanics and its Applications* 478 (2017): 143-157.
5. Bravo-Moncayo, Luis, et al. "Neural based contingent valuation of road traffic noise." *Transportation Research Part D: Transport and Environment* 50 (2017): 26-39.
6. He, Yongxiu, et al. "Urban long term electricity demand forecast method based on system dynamics of the new economic normal: the case of Tianjin." *Energy* 133 (2017): 9-22.
7. M. Gaber, A. M. Wahaballa, A. M. Othman, and A. Diab, "Tra_c accidents prediction model using fuzzy logic: Aswan desert road case study,"
8. Gaber, Mohammed, et al. "Traffic accidents prediction model using fuzzy logic: Aswan desert road case study." *JES. Journal of Engineering Sciences* 45.1 (2017): 28-44.
9. Hou, Zhongsheng, and Xingyi Li. "Repeatability and similarity of freeway traffic flow and long-term prediction under big data." *IEEE Transactions on Intelligent Transportation Systems* 17.6 (2016): 1786-1796.
10. Wang, Hongnian, and Han Su. "Star: A concise deep learning framework for citywide human mobility prediction." *2019 20th IEEE International Conference on Mobile Data Management (MDM)*. IEEE, 2019.
11. Duan, Yanjie, Yisheng Lv, and Fei-Yue Wang. "Performance evaluation of the deep learning approach for traffic flow prediction at different times." *2016 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI)*. IEEE, 2016.
12. Zhou, Teng, et al. "δ-agree AdaBoost stacked autoencoder for short-term traffic flow forecasting." *Neurocomputing* 247 (2017): 31-38.
13. Koesdwiady, Arief, Ridha Soua, and Fakhreddine Karray. "Improving traffic flow prediction with weather information in connected cars: A deep learning approach." *IEEE Transactions on Vehicular Technology* 65.12 (2016): 9508-9517.
14. Zhao, Zheng, et al. "LSTM network: a deep learning approach for short-term traffic forecast." *IET Intelligent Transport Systems* 11.2 (2017): 68-75.