DATA7703 Machine Learning for Data Scientists

Metro Interstate Traffic Volume

Group Name: WFTN

Jingxin Nie (46002002)
Linyuan Xing (47163740)
Yiang Gao (47192982)
Ruyun Qi (44506065)

November 8, 2021

Abstract

Traffic Volume is an important consideration for urban transportation authorities. There are a large number of studies on traffic volume, and most of them focus on ferries or airplanes. However, metro interstate traffic volume also affects people's lives and social economy. An object of this paper is to investigate the changes in metro interstate traffic volume in different weather conditions and different times, and establish different models for analysis and prediction. This will provide stakeholders with a viable forecasting method. The results showed that the analysis of traffic volume is a regression problem, and the linear regression model is not suitable for this data, there is a nonlinear relationship between traffic volume and influencing factors. By comparing the four models of linear regression, decision tree, XGBoost, and neural network, the results show that the decision tree model has an average fit to the data, while the neural network model data training time is too long, and the XGBoost model is the best one.

Contents

1 Introduction

1.1 Statement of the problem

Transportation is to provide convenience to people's lives and better help people save time. Taking transportation is an essential part of people's lives, and smooth transportation can provide residents with a good living experience.

However, most people encounter traffic jams in their lives. Whether it is traffic jams on the way to work, or traffic stagnation encountered during travel, it has brought great inconvenience to people's lives. Therefore, the study and prediction of traffic flow is very meaningful, and it may be able to improve people's lives.

In response to this problem, our study proposes to predict the traffic volume at different times. We plan to use data from UCI. After processing the data, we try models such as Linear regression, decision tree, XGBoost and Neural Network. Then we compare the MSE of different models to select the best model, so that the research can achieve the best results.

1.2 Significance of the problem

The analysis and forecast of traffic volume provide stakeholders with a convenient way to improve their lives.

- **Individuals:**
  If residents go to work in the morning, they need to predict the traffic flow to plan when they should get up. In order to prevent being late, they can choose the appropriate model of transportation under different weather conditions according to the traffic forecast.
- **Government:**
  The analysis and forecast of traffic volume can better help the government to make reasonable strategies at different times, such as formulating reasonable decentralization strategies during holidays to prevent traffic congestion.
- **Society:**
  The analysis and prediction of traffic volume is also of great significance. For example, in an emergency, finding the least blocked path is more convenient for ambulances, fire trucks and police cars to save lives, extinguish fires, and arrest fugitives at the fastest speed. This can provide protection for the lives of residents to a certain extent, and is more conducive to maintaining social stability.

2 Data Pre-processing

Before any analysis, data quality is important. Since the initial data set contains attributes such as strings and time, we need to preprocess the data to encode these strings and time. Data preprocessing is a very important step in the project. The quality of the data directly determines the prediction and generalization capabilities of the model. It involves many factors, including accuracy, completeness, consistency, timeliness, credibility, and interpretability. However, in real data, the data we get may contain a lot of missing values, may contain a lot of noise, may be caused by manual input errors, there may be abnormal points, may contain various types of data, which is very unfavorable to the algorithm model. train. The result of data cleaning is to process various complex data in corresponding ways to obtain the standard, clean, and continuous data, and provide our project.

Feature	**Description**
Holiday	Indicates if the date is a holiday and if it specifies the holiday, if not None
Temp	Indicates the temperature in Kelvin
rain_1h	Amount in mm of rain that occurred in the hour
snow_1h	Amount in mm of snow that occurred in the hour
clouds_all	Percentage of cloud cover
weather_main	Short textual description of the current weather
weather_description	Longer textual description of the current weather
date_time	Hour of the data collected in local CST time
traffic_volume	Hourly I-94 ATR 301 reported westbound traffic volume

Table 1: Dataset Overview

2.1 Holidays and Festival Dictionary

Because people travel on holiday differently from people who travel on workdays, we should distinguish between holidays in the dataset and workdays. We'll record both the vacations and their duration. For example, January 1 is New Year's Day, and the New Year's Day holiday has a total of three days, January 1, January 2, and January 3. In the initial dataset, only January 1 is counted as a holiday, but we must record January 2 and January 3 as holidays. The significance of this approach is that holiday traffic is not the same when they belong to workdays, and holiday traffic can be understood as some outliers.

In the festival dictionary, we recorded the start time of each festival. Regardless of the training set or the test set, we need to mark the start time of the holiday. This approach is also used in conjunction with the holiday list above to achieve better-preprocessing effects.

Only by completely separating holidays from workdays could reduce the impact of outliers on the model. Then, we classified dates for holidays and workdays. We classify the whole data into 4 types of data: the day before workday, workday, vacation, and festival. The day before workday means Friday and the last day of the holiday. Workday means normal Monday, Tuesday, Wednesday, Thursday. Vacation means Ordinary National Holidays. Festival means Christmas, Thanksgiving days, and so on.

Special Days	Description
weekday-1	The day before the working day
weekday	Working day
vac	Date collected in the vacation dictionary
fes	Date collected in the festival dictionary

Table 2: Description of Special Days

2.2 Separate Time Data

In this part, we separated the time data into the year, month, day, and hour. Then, we could use these new data sets to match the corresponding festival. For example, month = 1 and day = 1, it is the new year's day festival. Month = 12 and day = 25, it is the Christmas festival.

Time	Year	Month	Day	Hour
$2012-10-02-01:00:00$	2012	10	02	01
$2013-01-01-02:00:00$	2013	01	01	02
$2014-01-01-03:00:00$	2014	01	01	03
$2013-01-02-04:00:00$	2013	01	02	04
$2012-02-02-05:00:00$	2012	02	02	05
$2012-03-01-06:00:00$	2012	03	01	06
$2012-04-01-07:00:00$	2012	04	01	07
$2015-06-01-08:00:00$	2015	06	01	08
$2012-07-01-09:00:00$	2012	07	01	09
$2012-10-10-10:00:00$	2012	10	10	10

Table 3: Examples of Separated Time

2.3 Encode Weather and Weather Description

In the dataset, we take out all the weather descriptions in the data and encode them from good to bad, corresponding to 0-3. Weather codes have been used before, and weather conditions can be judged directly here. Since the two weather descriptions have been previously coded, the two descriptions are categorized directly here. The values you get are put back on our list.

Code	Description
0	weather code - good
1	weather code - mid
2	weather code - bad
3	weather code - too bad

Table 4: Description of Weather Code

Code	Description
0	weather description code - good
1	weather description code - mid
2	weather description code - bad
3	weather description code - too bad

Table 5: Weather Description Code

There are 11 normal kinds of weather, which are Clear, Clouds, Drizzle, Fog, Haze, Mist, Rain, Smoke, Snow, Squall, and Thunderstorm. We plan to encode these 11 kinds of weather into 4 types based on the extent to which the weather affects travel. For example, on a clear day, the weather is not a factor in people's travel considerations, so the code of the clear day is 0. On a fog day, people will consider a little about the weather before their travel behavior. However, on a rainy day, people will consider more about the weather. At last, on a thunderstorm day, people will consider a lot about the weather before their travel behavior. So the summary of the weather code is the weather is worse and the code is bigger.

Weather	**Weather Code**
Clear	0
Clouds	1
Drizzle	1
Fog	1
Haze	1
Mist	1
Rain	2
Smoke	2
Snow	3
Squall	3
Thunderstorm	3

Table 6: Weather Code

In the dataset, we take out all the weather descriptions in the data and encode them from good to bad, corresponding to 0-3. The coding here is done in what we think is a comparison. Test several times to get a better coding order. such a treatment It is also the digitization of the weather description section, which facilitates the identification of our models while adding to our characteristics. Before we deal with the weather.

Weather Description	Weather Description Code
broken clouds	0
few clouds	0
scattered clouds	0
sky is clear	0
Sky is Clear	0
drizzle	1
fog	1
light intensity drizzle	1
light intensity shower rain	1
light rain	1
light snow	1
mist	1
moderate rain	1
freezing rain	2
haze	2
light rain and snow	2
overcast clouds	2
proximity shower rain	2
proximity thunderstorm with drizzle	2
proximity thunderstorm with rain	2
shower snow	2
smoke	2
snow	2
SQUALLS	2
heavy intensity drizzle	3
heavy intensity rain	3
heavy snow	3
thunderstorm	3
thunderstorm with drizzle	3
thunderstorm with heavy rain	3
thunderstorm with light drizzle	3
thunderstorm with light rain	3
thunderstorm with rain	3
very heavy rain	3

Table 7: Weather Description Code

2.4 Additional Weather Features

The past weather situation. Because the data are time-sensitive, past weather conditions are likely to have an impact on current or future weather conditions. For example, after a snow day, snow on the road could take some days to melt. The impact on travel is significant. After a rainy day, the stagnant water on the road will also affect the traffic after it stops raining. We have recorded the weather conditions for the past five days. That is, the current weather conditions and weather conditions within five days.

Features	Description
wcode-1	The weather in the past one days
wcode-2	The weather in the past two days
wcode-3	The weather in the past three days
wcode-4	The weather in the past four days
wcode-5	The weather in the past five days

Table 8: Weather of Past Five Days

Weather weights for the past few hours and the next few hours. We have also recorded the weather code and weather description code for the past 6 hours, 12 hours, 24 hours, and 84 hours. Based on these additional attributes, we could make a better prediction.

Features	Description
wcode-mean-0	The weather in the future 6 hours
wcode-mean-1	The weather in the future 12 hours
wcode-mean-2	The weather in the future 24 hours
wcode-mean-3	The weather in the future 84 hours

Table 9: Weather of Past Few Hours

Features	Description
wcode-mean+0	The weather in the past 6 hours
wcode-mean+1	The weather in the past 12 hours
wcode-mean+2	The weather in the past 24 hours
wcode-mean+3	The weather in the past 84 hours

Table 10: Weather of Future Few Hours

We perform aggregation operations on the data according to different standards, such as the average, maximum, minimum, and median of each month. From these data, generate historical average, historical average weight, historical maximum, historical minimum, historical median, weather average, average of the past hour, and average of the past two hours.

Features	Description
history_mean	Historical average
history_mean-w	Historical average weight
history_max	Historical maximum
history_min	Historical minimum
history_mean	Historical median
history_mean-w	Weather average
history_max	The average of the past an hour
history_min	The average of past two hours

Table 11: Weather of Future Few Hours

3 Predictive Models

3.1 Linear Regression

3.1.1 Basic theory

Linear regression is the simplest regression algorithm in machine learning. Multiple linear regression refers to a linear regression problem in which a sample has multiple characteristics. For a sample with features, its regression result can be written as an equation that is almost familiar to everyone.

3.1.2 Preprocessing

We converted the date into month, week, and hour. We used label encode to encode weather main and weather description.

	holiday	temp	rain_1h	snow_1h	clouds_all	weather_main	weather_description	month	weekday	hour
0	7	264.16	0.0	0.0	75	8	10	1	1	2
1	7	263.95	0.0	0.0	90	1	18	1	1	3
2	7	263.65	0.0	0.0	90	1	18	1	1	4
3	7	262.40	0.0	0.0	40	8	10	1	1	5
4	7	261.42	0.0	0.0	20	1	4	1	1	6

Figure 1: Features of Linear Regression

3.1.3 Performance

Features	**Weight**
holiday	125.0649620342929
temp	15.774365014768039
rain_1h	-35.768652758659655
snow_1h	-499.0617225164676
clouds-all	1.968500125687452
weather_main	-20.27542261686605
weather_description	-8.56525260457146
month	-11.930084212183933
weekday	-163.4615181859234
hour	94.66661627897057

Table 12: Weight of Features

Training MSE	**Training Score**	**Test MSE**	**Test Score**
3365803.54	16.77%	3283215.78	16.41%

Table 13: Prediction Performance (Linear Regression)

According to the scores of MSE and r2 of the linear regression model, we can see that the MSE of linear regression is very high, but r2 is only about 0.18. From these values, it is preliminarily

judged that the linear regression model is not very suitable.

3.1.4 Prediction results

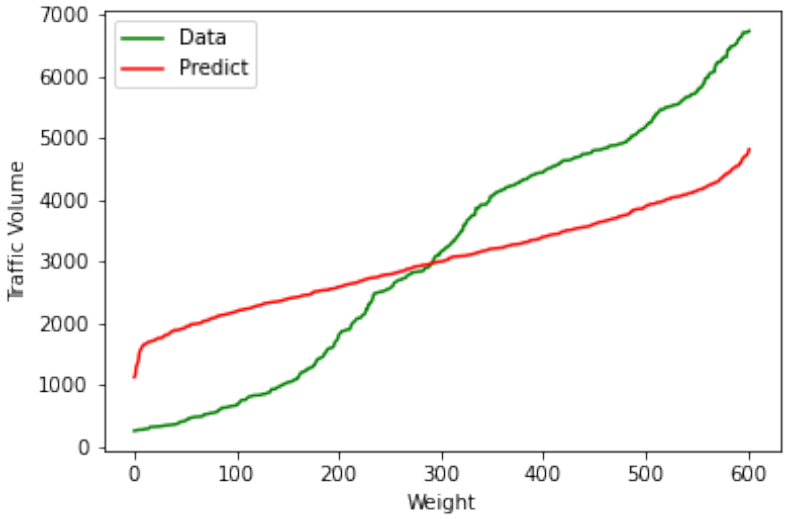

Figure 2: Weight vs. Traffic Volume for Linear Regression

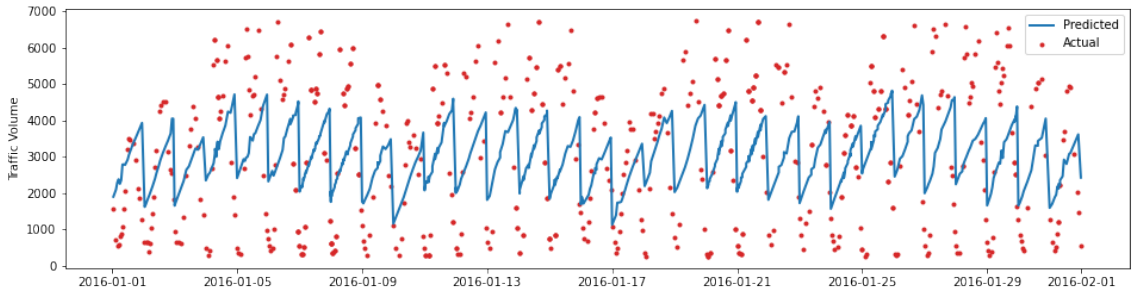

Figure 3: Predicted & Actual Traffic Volume in Jan 2016 (Linear Regression)

The green line is the line generated by the test set in the data set, and the red line is the line generated based on the prediction of the linear regression model. From the image perspective, it is obvious that the accuracy of the prediction is not high.

We also used a lot of different parameters to make a linear regression model, but the results are all similar. According to the synthesis of MSE and r2 and the generated images, the results of prediction through linear regression will not be very good.

3.2 Decision Tree

3.2.1 Introduction

- **Reasons for model selection:**
  Because the linear regression model is mainly suitable for continuous variables, but we browsed the data and found that many data in the data are not continuous. Therefore, we try to make the decision tree model explore the possible relationships of the data.

- **Basic theory:**
  The decision tree can be used as a classification algorithm, can also be used as a regression algorithm. It is a tree structure in which each internal node represents a judgment on an attribute, each branch represents the output of a judgment result, and finally each leaf node represents a classification result.

- **Pros and cons:**
  The decision tree is easy to understand. Each feature of it is processed separately, so the division of data will not be affected by data scaling. But a single decision tree is prone to overfitting.

3.2.2 Feature importance

Feature importance is the importance of each feature to the decision tree. They are between 0 and 1. The closer to 1, the higher the importance. The following table shows the importance of each feature in this decision tree:

Features	**Importance**
temp	0.41888
rain/1h	0.02683
clouds/all	0.10481
weather$_{d}escription$	0.05337
month	0.16639
day	0.12447
hour	0.07396
holiday	0.00071
weather/main	0.02904
snow/1h	0.00152

Table 14: Feature importance for Decision Tree

This table shows that 'temp' is the most significant feature, which is the most important in the decision tree, followed by time-related features, like 'month', 'day'. The amount of 'holiday' has a very low degree of influence in the decision tree, and it also show that whether it snow or not does not matter.

3.2.3 Model explanation

Since the actual decision tree is very large and cannot be fully displayed with the graph, we built it and set the maximum depth to 2 so that such a decision tree can be displayed more easily.

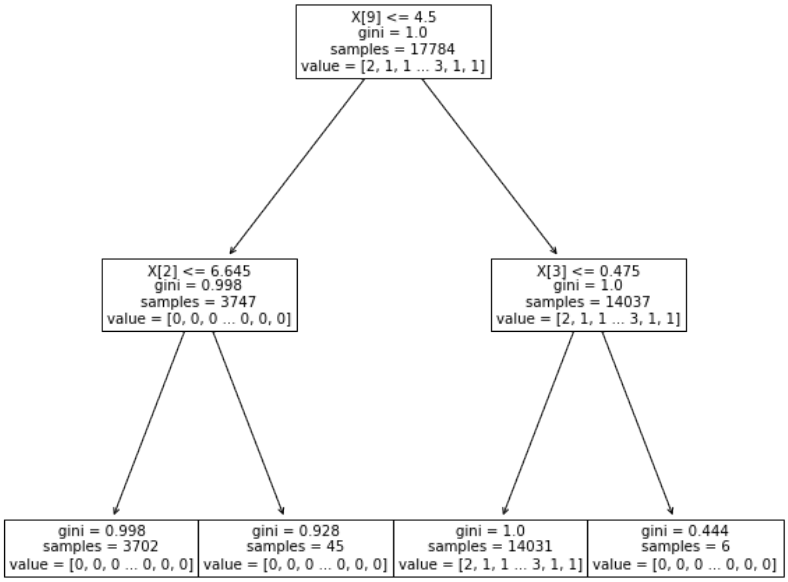

Figure 4: Decision Tree

From this figure, we can see that at the beginning, x(9) is screened. It is about that if the figure for 'day' is greater than 4.5, then the x(3) 'rain/1h' will be compared. If the figure for 'day' is smaller or equal to 4.5, then the x(2)'temp' will be screened. Figure 3 just show the decision tree with the max depth is 2. In fact, the best decision tree model we get is not just 2 depth, but the analysis has the same principle.It will continue to split so, until a regression value is obtained according to different characteristics. About decision tree model, there are the following explanations:

- **Pruning:**

  The decision tree model has several key variables that need to be set. Generally, the decision tree has over-fitting, or the accuracy of the training set is too high. Therefore, we will consider doing pruning. In the initial model, we did not limit the maximum depth. Sure enough, the scores of the model obtained on the training set is 1, which indicates that the model may have generalization or overfitting. Then we changed the value of maxdepth many times and selected the model when maxdepth was 42.

- **Performance:**

  We choose the decision tree model with max depth is 42, the result has showed below:

Training MSE	**Test MSE**
15902.38	538533.24

Table 15: Prediction Performance (Decision Tree)

3.2.4 Prediction results

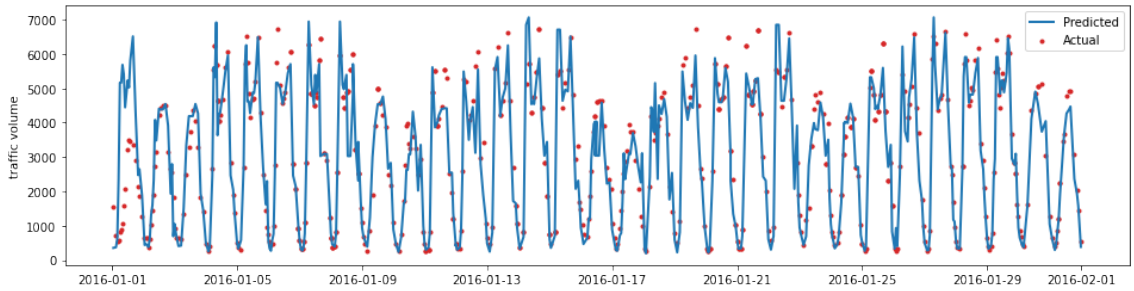

Figure 5: Predicted Actual Traffic Volume in Jan 2016 (Decision Tree)

The figure shows that most actual values fall on the forecast, but there are still many plots that do not fall on it. We believe that the decision tree model is closer to this data than the linear regression model. However, considering the model's score on the test set is low and the points that are still off the line in the figure, we will continue to explore other models.

3.3 XGBoost

3.3.1 Introduction

GBDT uses boosting technique to create an ensemble learner. Decision trees are connected sequentially to obtain a strong learner. Decision trees in GBDT are not fit to the entire dataset. The goal is to minimize the errors of previous tree. Thus, each tree fits to the residuals from the previous one. As a result, the overall accuracy and robustness of the model gradually increase.

As we know the success of a random forest model highly depends on using uncorrelated decision trees. Bootstrapping plays a key role in creating uncorrelated decision trees. GBDT does not use or need bootstrapping. Since each decision tree is fit to the residuals from the previous one, we do not need to worry about having correlated trees.

We can find the difference between random forest and GBDT on the two figures.

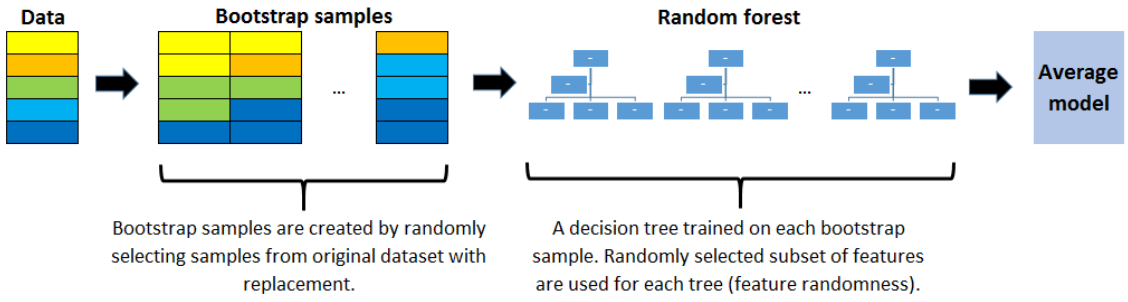

Figure 6: Random Forest

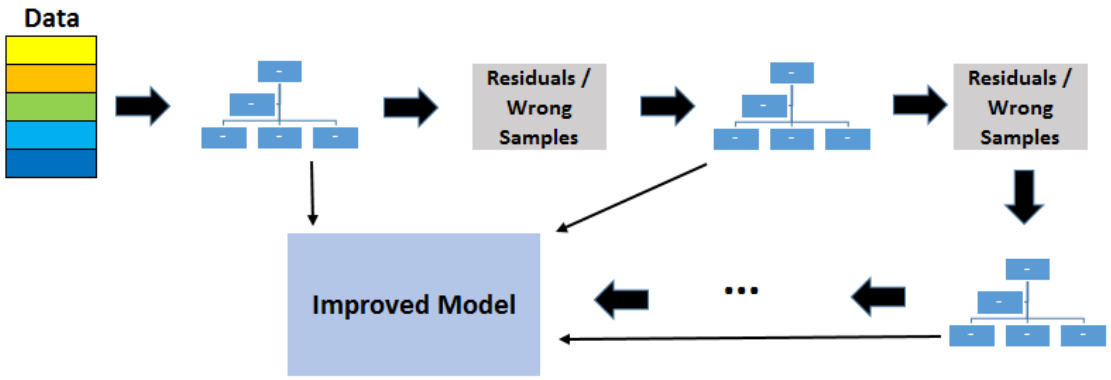

Figure 7: Gradient Boosted Decision Tree

XGBoost is an implementation of gradient boosted decision trees designed for speed an performance. The library is laser focused on computational speed and model performance, as such there are few frills. Nevertheless, it does offer a number of advanced features.

3.3.2 Features

The implementation of the model supports the features of the scikit-learn and R implementations, with new additions like regularization. Three main forms of gradient boosting are supported:

- *Gradient Boosting* algorithm also called gradient boosting machine including the learning rate.
- *Stochastic Gradient Boosting* with sub-sampling at the row, column and column per split levels.
- *Regularized Gradient Boosting* with both L1 and L2 regularization

System Features:

The library provides a system for use in a range of computing environments, not least

- *Parallelization* of tree construction using all of your CPU cores during training.
- *Distribution Computing* for training very large models using cluster of machines.
- *Out-of-Core Computing* for very large datasets that do not fit into memory.
- *Cache Optimization* of data structures and algorithm to make best use of hardware.

Algorithm Features:

The implementation of the algorithm was engineered for efficiency of compute time and memory resources. A design goal was to make the best use of available resources to train the model. Some key algorithm implementation features include:

- *Sparse Aware* implementation with automatic handling of missing data values.
- *Block Structure* to support the parallelization of tree construction.
- *Continued Training* so that you can further boost an already fitted model on new data.

3.3.3 Execution Speed

Generally, XGBoost is fast. Really fast when compared to other implementations of gradient boosting. Szilard Pafka performed some objective benchmarks comparing the performance of XGBoost to other implementations of gradient boosting and bagged decision trees. He wrote up his results in May 2015 in the blog post titled "Benchmarking Random Forest Implementations".

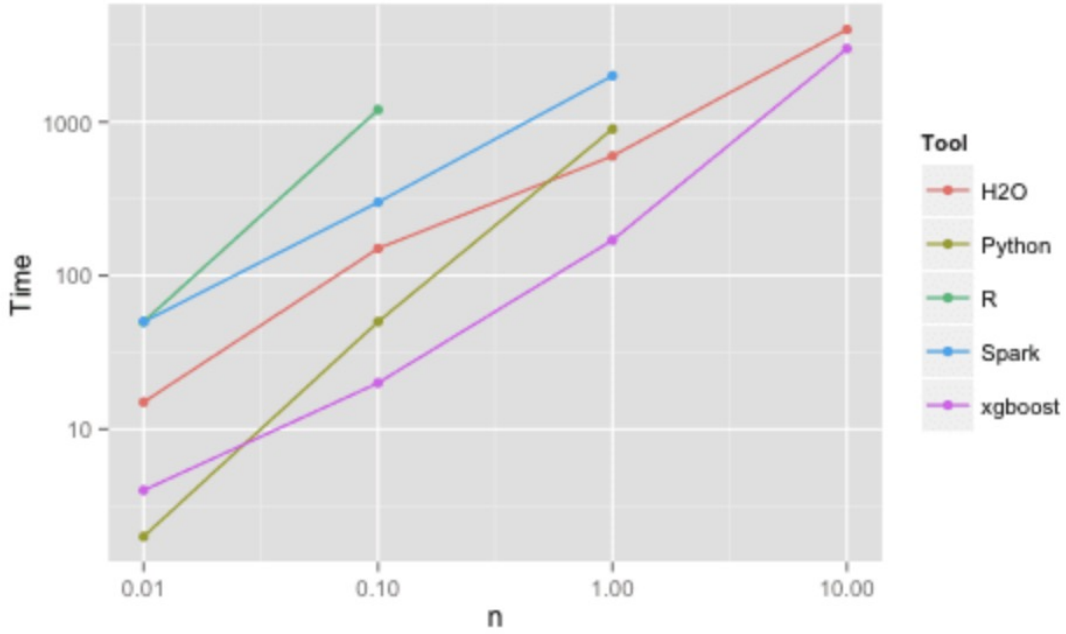

Figure 8: Benchmark Performance of XGBoost

His results showed that XGBoost was almost always faster than the other benchmarked implementations from R, Python Spark and H2O.

3.3.4 Optimization

Machine learning algorithms have hyperparameters that allow the algorithms to be tailored to specific datasets.

Although the impact of hyperparameters may be understood generally, their specific effect on a dataset and their interactions during learning may not be known. Therefore, it is important to tune the values of algorithm hyperparameters as part of a machine learning project.

It is common to use naive optimization algorithms to tune hyperparameters, such as a grid search and a random search. An alternate approach is to use a stochastic optimization algorithm, like a stochastic hill climbing algorithm.

Manual Hyperparameter Optimization:

Machine learning models have hyperparameters that you must set in order to customize the model to your dataset.

Often, the general effects of hyperparameters on a model are known, but how to best set a hyperparameter and combinations of interacting hyperparameters for a given dataset is challenging.

A better approach is to objectively search different values for model hyperparameters and choose a subset that results in a model that achieves the best performance on a given dataset. This is called hyperparameter optimization, or hyperparameter tuning.

A range of different optimization algorithms may be used, although two of the simplest and most common methods are random search and grid search.

- **Random Search:**
  Define a search space as a bounded domain of hyperparameter values and randomly sample points in that domain.
- **Grid Search:**
  Define a search space as a grid of hyperparameter values and evaluate every position in the grid.

Grid search is great for spot-checking combinations that are known to perform well generally. Random search is great for discovery and getting hyperparameter combinations that you would not have guessed intuitively, although it often requires more time to execute.

Hyperparemeter:

Hyperparameter	Description
max_depth	The depth of the tree
learning_rate	The weight of learning rate
n_estimators	The number of the weak estimators
objective	The loss function
n_jobs	The number of parallel models
reg_alpha	L1 regularization
reg_lambda	L2 regularization

Table 16: Hyperparemeter for XGBoost

Hyperparameter Optimization:

XGBoost is short for Extreme Gradient Boosting and is an efficient implementation of the stochastic gradient boosting machine learning algorithm.

The stochastic gradient boosting algorithm, also called gradient boosting machines or tree boosting, is a powerful machine learning technique that performs well or even best on a wide range of challenging machine learning problems.

For this project, we will focus on four key hyperparameters; they are:

- Learning Rate(learning_rate)
- Number of Trees(n_estimators)
- Subsample Percentage(subsample)
- Tree Depth(max_depth)

The *learning rate* controls the contribution of each tree to the ensemble. Sensible values are less than 1.0 and slightly above 0.0 (e.g. 1e-8).

The *number of trees* controls the size of the ensemble, and often, more trees is better to a point of diminishing returns. Sensible values are between 1 tree and hundreds or thousands of trees.

The *subsample* percentages define the random sample size used to train each tree, defined as a percentage of the size of the original dataset. Values are between a value slightly above 0.0 (e.g. 1e-8) and 1.0

The *tree depth* is the number of levels in each tree. Deeper trees are more specific to the training dataset and perhaps overfit. Shorter trees often generalize better. Sensible values are between 1 and 10 or 20.

After we define the *objective* function and *step* function, we can use the hill climbing algorithm to tune the hyperparameters of the XGBoost.

3.3.5 Performance

Validation MSE	Test MSE
73373.521	459059

Table 17: Prediction Performance (XGBoost)

3.3.6 Prediction results

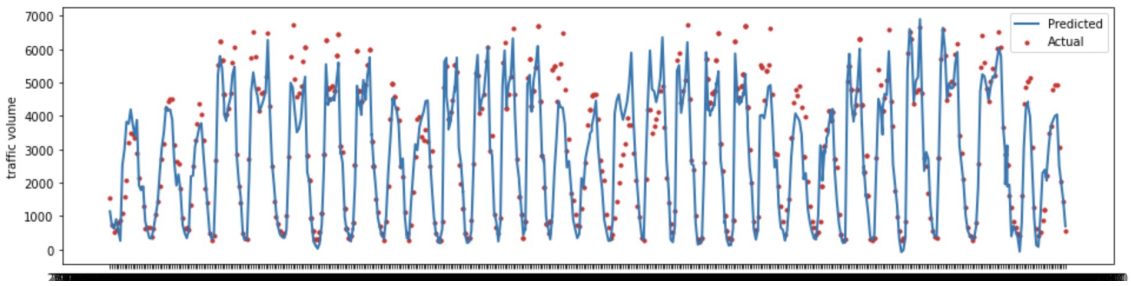

Figure 9: Predicted & Actual Traffic Volume in Jan 2016 (XGBoost)

3.4 Neural Networks

3.4.1 Basic theory

Neural Networks consider variable interactions and create a non-linear prediction model. The hierarchical or multi-layered structure of neural networks contributes to their predictive abilities. Multilayer Perceptron (MLP) which is known as a multilayer feedforward neural network is selected in our prediction of traffic volume.

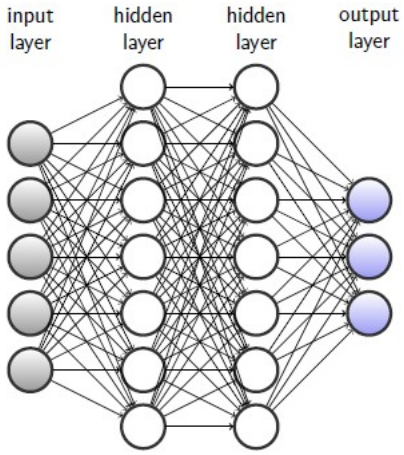

Figure 10: Sample Multi-Layer Perception (DATA7703 Lecture: Multilayer Perceptrons)

Below are the variables and functions used in our modelling:

Notation	Description
x_{ij}	input of pattern i and input j
y_{ik}	actual output value of pattern i and ouput k
w_{jq}	weight on connection from jth input to qth hidden unit
w_{qk}	weight on connection from qth hidden unit to kth output unit
$a(u) = \max(0, u)$	rectifer activation function
$u_{iq} = \sum_{j=0}^{p} w_{jq} x_{ij}$	weighted input for qth hidden unit in response to ith input pattern
$h_{iq} = g(u_{iq})$	hidden unit value of pattern i and hidden unit q
$z_{ik} = \sum_{q=0}^{l} w_{qk} h_{iq}$	weighted input for kth output unit in response to ith input pattern
$o_{ik} = g(z_{ik})$	output unit value of pattern i and output k
$E_{ik} = [y_{ik} - O_{ik}]^2$	squared error

Table 18: MLP Notations

3.4.2 Pre-processing

Except the features in raw dataset, the features of hour, weekday and month were generated from the date time. For example, the following is the features of our training set (X_{train}).

	holiday	temp	rain_1h	snow_1h	clouds_all	weather_main	weather_description	month	weekday	hour
0	7	264.16	0.0	0.0	75	8	10	1	1	2
1	7	263.95	0.0	0.0	90	1	18	1	1	3
2	7	263.65	0.0	0.0	90	1	18	1	1	4
3	7	262.40	0.0	0.0	40	8	10	1	1	5
4	7	261.42	0.0	0.0	20	1	4	1	1	6
...	...	...	...	...	...	...	...	...	...	...
17779	7	264.89	0.0	0.0	90	8	15	12	3	16
17780	7	265.32	0.0	0.0	90	8	15	12	3	19
17781	7	265.51	0.0	0.0	75	8	15	12	3	21
17782	7	265.78	0.0	0.0	75	8	15	12	3	23
17783	6	265.94	0.0	0.0	90	4	7	1	4	0

17784 rows × 10 columns

Figure 11: X_{train} for MLP

Furthermore, since the stochastic optimizer is sensitive to the feature values, it is necessary to standardize the features before training the models by using StandardScaler.

3.4.3 Optimization

In our regression scenario, the mean square error (MSE) is the loss function which reflects the performance of our training model. Therefore, the best solver could be chosen from their training MSEs.

SDG optimizer with logistic activation function was the fastest in our case which required the least iterations for convergence , but its training MSE which is over 4 million is much higher than the other two optimizers. Thus, we continued to compare the LBFGS optimizer and ADAM optimizer.

Rectifer activation function and hidden layer size of (100,) were found to be the best for both LBFGS and ADAM optimizer by implementing GridSearchCV. To display their performances, the figure of max iterations versus training MSEs was plotted as bellow shows.

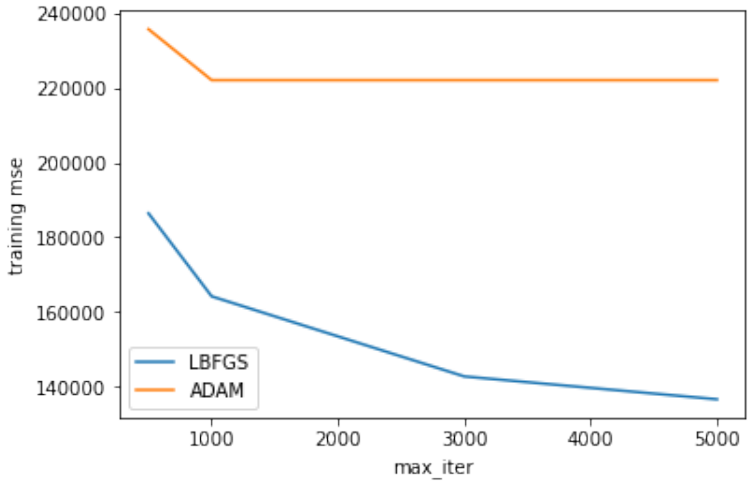

Figure 12: max iterations v.s. training mses for LBFGS & ADAM optimizer

It is obvious that LBFGS has better perforce especially with larger iterations. Its training MSE could be reduced to 136615.58 with max iterations of 5000. Then, tuning alpha which is the L2 penalty parameter to optimize this model.

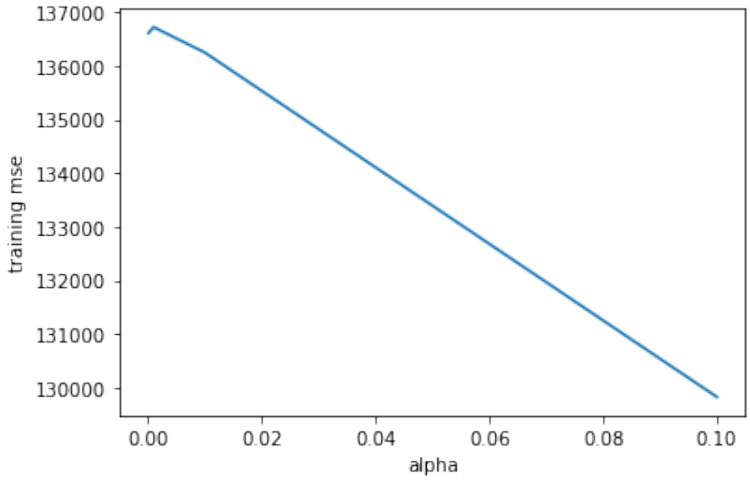

Figure 13: alpha v.s. training mses for LBFGS optimizer

Therefore, the optimized training model has the bellow parameters.

Parameter	**Value**
hidden_layer_sizes	(100,50)
activation	relu
solver	lbfgs
alpha	0.1
max_iter	5000

Table 19: Parameters for MLPRegressor

3.4.4 Performance

Training Time	Training MSE	Test MSE	Training Score	Test Score
411.28 sec	129631.84	1075083.39	96.79%	72.63%

Table 20: Prediction Performance (MLP)

It took a long time to train this MLP model since the training set is large and the max iterations is large. After the long training, the training score indicates the high performance of predictions.

3.4.5 Prediction results

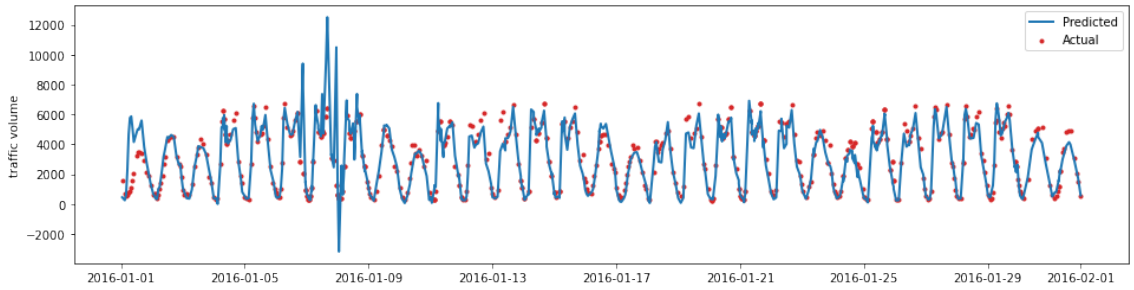

Figure 14: Predicted & Actual Traffic Volume in Jan 2016 (MLP)

The actual values are well-matched by the forecast line. However, some outliers of negative or extremely large predicted traffic volume exist a few days before the 9th of January, 2019.

3.5 Model Discussions

In this project, we try to use different models to find the best prediction. Initially, the linear regression model became our first try. Even we do a lot of work to find the relationship between the features, and the linear regression model can not do well in this dataset. As we know, the linear model works well on the dataset whose features and the results have a strong relationship. Considering the low relationship between those features, the tree model, especially the forest model, could work well on this dataset. So, we will try two different ensemble models, the bagging one and the boosting one. For the decision tree and random forest, we find the minimum mse of the random forest model is around 600,000. On the other hand, the minimum mse of the XGBoost, which belongs to the boosting, is 400,000. We think that the tree model can fit the data well, and it is a fast model. Then we try the neural network. When the network becomes deeper, the network can fit the data well, which could perform well on the dataset. However, the result shows that the MLP does not do that well on this dataset, and it cost too much time on the training process. If the network could go deeper, the result could be better. But this action could bring more time and computer cost.

Model	**Test MSE**
Linear Regression	3283215
Decision Tree	538533
XGBoost	459059
Neural Network	1075083

Table 21: Test MSEs for Different Models

4 Limitations

4.1 Dataset

Our dataset's features are mostly related to weather and holidays. Other factors, such as the population of the neighboring suburbs and the station's route alterations, may have an impact on the traffic volume at Station 301. Our models' performance could be improved if these features' data could be collected.

4.2 Data Pre-processing

In this project, we have used a variety of data preprocessing methods. In order to adapt to the establishment of various models, different models require different data preprocessing methods. In the end, we chose the best performing model among all the models we tried. Data preprocessing is a place worthy of our lifelong learning. There are too many better data preprocessing methods waiting for us to discover and learn.

4.3 Models

4.3.1 Linear Regression

To establish a linear regression model, we tried many combinations of eigenvalues. At first, when all the parameters in the original data set were added to the model, we got a very unsatisfactory result. We subsequently adjusted the data preprocessing method several times, but the results were not very good. We even tried to use only a few key features to make a linear regression model, but the results were still not much different from the initial results, and the prediction results were very poor. In the end, everyone thought that this data set was not suitable for linear regression models.

4.3.2 Decision Tree

The decision tree is not suitable for data with high feature dimension but less data in each category. Such segmentation will cause the data to be divided into a lot of fragmented smaller spaces instead of being divided into two larger spaces. If the depth of the tree is not limited, the decision tree will keep splitting. The total amount of data we used is large. However, after coding, we find that some features have more categories, and some categories have fewer data. In addition, the decision tree is used on a large number of data samples. Therefore, it leads to some deviations.

4.3.3 XGBoost

Xgboost adopts pre sorting. Before iteration, it pre sorts the characteristics of nodes, traverses and selects the optimal segmentation point. When the amount of data is large, the greedy method takes time. Xgboost also adopts level-wise to generate decision trees and split the leaves of the same layer at the same time, so as to carry out multi-threaded optimization, which is not easy to over fit. However, the splitting gain of many leaf nodes is low, so it is not necessary to split further, which brings unnecessary overhead.

4.3.4 Neural Network

It had trouble in converging to a solution even after 5000 iterations. In fact, our final solution is not optimal for each training pattern. To avoid missing any patterns, a high number of training iterations is required. Furthermore, since our case required a large number of hidden layers for optimal performance, the overall number of parameters increased dramatically, resulting in inefficiency owing to redundancy in such high dimensions.

5 Conclusion

5.1 Summary

The Metro Interstate Traffic Volume dataset is a regression problem that requires us to forecast the value of a continuous variable. The objective of this project is to build prediction models and compare their performance using a variety of machine learning algorithms.

Simple linear regression showed the lowest performance with its large MSE. Its forecast line did not fit well with the actual volume values. It is ideal that XGBoost outperforms Decision Tree because it is like a more advanced version of Decision Tree that utilizes a gradient boosting framework. One of the most important findings is that by implementing its linear combined model, the MSE of the XGBoost model could be reduced by 7.6%, implying that a well-designed combination model could improve prediction performance. Another is that the hyperparameter tuning of a Multilayer Perceptron (MLP) requires precision and patience. As a result, better optimization may contribute to enhancing MLP performance.

In our prediction case, XGBoost has the best outcomes, as evidenced by the results presented for each model. It is effective because it utilizes complicated weighting features and a combination of two training models, whereas the other prediction models we considered did not.

5.2 Future Recommendations

Linear regression is one of the most basic types of regression in machine learning, but it does not perform well in our case. Other regression models, such as logistic regression, ridge regression, lasso regression, and Bayesian linear regression, may be applied and fit better in this scenario because there are only a few features to predict traffic volume. If the dataset is high-dimensional, it would be helpful to implement principal component analysis (PCA) for dimension reduction. Furthermore, Support Vector Regression (SVR) works similarly to Support Vector Machines (SVMs), attempting to fit the best line inside a threshold value, rather than minimizing the error between the actual and projected value in other regression models. The distance between the hyperplane and the boundary line is its threshold value. For large datasets, such as ours, Linear SVR or SGD Regressor should be functional.

We found that traffic is heavier in rush hour in the morning from 6 am to 8 am and afternoons from 3 pm to 6 pm because of commuting. Since the traffic flow is complicated and stochastic, there exist a lot of challenges for short-term prediction. To solve this problem, it would be helpful to implement specific models for time-based issues. For example, a combined short-term traffic flow prediction system that relies on the autoregressive integral moving average (ARIMA) model and long short-term memory (LSTM) neural network were proposed as a solution by Lu et al. in 2021. In addition, Zheng et al. (2020) has presented DSTO-GBRT, a dynamic spatial-temporal feature optimization technique based on a gradient–boosted regression tree for short-term traffic flow prediction using ERI extensive data. These solutions are considered more for the characteristics of time which are not mainly addressed in this project. Thus, future studies should pay more attention to these specific aspects.

A References

D. Zhang and M. R. Kabuka, "Combining weather condition data to predict traffic flow: a GRU-based deep learning approach," *IET intelligent transport systems*, vol. 12, no. 7, pp. 578–585, 2018, doi: 10.1049/iet-its.2017.0313.

Jason Brownlee, "A Gentle Introduction to XGBoost for Applied Machine Learning", 2021. [Online]. Available: `https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/`

Jason Brownlee, "How to Manually Optimize Machine Learning Model Hyperparameters", 2021. [Online].Available: `https://machinelearningmastery.com/manually-optimize-hyperparameters/`

J. Brownlee, "Linear Regression for Machine Learning", Machine Learning Mastery, 2016. [Online].Available: `https://machinelearningmastery.com/linear-regression-for-machine-learning/`.

L. Zheng, J. Yang, L. Chen, D. Sun, and W. Liu, "Dynamic spatial-temporal feature optimization with ERI big data for short-term traffic flow prediction," *Neurocomputing*, vol. 412, pp. 339–350, 2020.

P. Pandey, "Data Preprocessing : Concepts", Medium, 2019. [Online].Available: `https://towardsdatascience.com/data-preprocessing-concepts-fa946d11c825`.

S. Lu, Q. Zhang, G. Chen, and D. Seng, "A combined method for short-term traffic flow prediction based on recurrent neural network," *Alexandria Engineering Journal*, vol. 60, no. 1, pp. 87–94, 2021.

Soner Yildirim, "3 Key Differences Between Random Forests and GBDT", 2021. [Online].Available: `https://towardsdatascience.com/3-key-differences-between-random-forests-and-gbdt-cfc48093200b`

W. Mengel,"Short-Term Traffic Forecasting on the Pacific Motorway by Multiple Linear Regression using coefficients which vary with time," The University of Queensland, School of Engineering, 2002.