MBTA Bus Service Prediction

Final Project Report

Group 78

Praneith Ranganath
Aditya Bharadwaj Shivapura Guruprasad

ranganth.p@northearsern.edu
shivapuraguruprasa.a@northeastern.edu

Percentage of Effort Contributed by Student 1: 50

Percentage of Effort Contributed by Student 2: 50

Signature of Student 1:_ Praneith Ranganath

Signature of Student 2: Aditya Bharadwaj Shivapura Guruprasad

Submission Date: 4/20/2023

Project Setting:

Everyday millions of passengers use Massachusetts Bay Transportation Authority (abbreviated MBTA and known colloquially as "the T") for their commute. In 2021, the system had a ridership of 160,557,600, or about 689,300 per weekday as of the second quarter of 2022 making it the fourth-busiest transport system in the U.S. The MBTA bus system, the nation's sixth largest by ridership, has over 170 routes. Most routes provide local service in the urban core; smaller local networks are also centered around Waltham, Lynn, and Quincy.

Greater Boston's population has increased 53% in the last 50 years while the MBTA bus fleet has decreased. The MBTA's current operating fleet of 1,121 buses is smaller than it was in 1972, when the agency operated 1,200 buses. While the MBTA has invested significant resources to hire new bus operators, even if they double the current rate of hiring, it could take up to two years to hire and train enough operators to bring bus service up to the pre-pandemic level.

Problem Definition: Apart from the above said challenges, it is also reported that the MBTA needs 300 more drivers to meet its current schedule, plus another 440 for its reshaped route map. Hence with this data analytics project, we are trying to help the organization solve these problems by building a model to answer below questions:

1. What is the predicted load of a bus on a particular date and time?
2. What is the effective way to utilize bus drivers ?
3. Which bus route needs additional resources and to balance them?

Data collection: The dataset is obtained from the official open source data of MBTA on their website **(https://mbta-massdot.opendata.arcgis.com/)**

Data description: The guide to help us understand the data can be found here. It has around 5m records with 10+ attributes. The data contains detailed boarding, alighting, and load for bus. It contains the averages per trip, season, route/line, direction, stop, day type, and time period.

Below is the overview of all the columns that we have in our dataset

Data Dictionary:

Name	Description	Data Type	Example
season	Season and year for which ridership should be returned.	String	Fall 2017
route_id	GTFS-compatible route for which ridership should be returned.	String	34E
route_variant	GTFS compatible route pattern id that identifies the variation from the typical trip taken by the vehicle. Indicates the route, variant number, and direction of trip.	String	34E-6-1
direction_id	GTFS-compatible direction for which ridership should be returned.	Integer	0
trip_start_time	24-hour time at which the one-way trip began. Trips starting after 11:59 PM are represented as 24:00 plus their 24-hour time. For example, a trip starting at 1:30 AM will be 25:30:00.	Time	05:15:00
day_type_id	Shorthand for day identifier.	String	day_type_01
day_type_name	Text description of the id. Weekday, Saturday, or Sunday; holidays are excluded from the data.	String	Weekday
stop_name	GTFS-compatible stop name for which ridership should be returned.	String	WASHINGTON ST OPP RUGGLES ST
stop_id	GTFS-compatible stop for which ridership should be returned.	String	1
stop_sequence	The ordinal identifier of the stop on the particular trip.	Integer	9
average_boardings	The average count of passengers getting on vehicles.	Double	435.6
average_alightings	The average count of passengers leaving vehicles.	Double	65.4
average_load	The average number of passengers on the vehicle upon leaving a stop, summed by the aggregated fields; represented as cumulative average load out.	Double	17.8
sample_size	The number of trips scheduled at the start time used to calculate the average boardings, alightings, and loads.	Integer	18

Data exploration, visualization, and processing

The "load_" column in our dataset provides valuable information about the number of passengers on a bus at the time it leaves a stop. This data can be used to evaluate the efficiency and effectiveness of bus services in terms of passenger demand.

Analyzing the average load data can help us identify peak hours when buses are most crowded, and take appropriate measures such as adding more buses or adjusting schedules to accommodate passenger needs.

Additionally, tracking average load over time can help us monitor trends in passenger demand and adjust bus routes or services accordingly.

Furthermore, the average load data can be used in combination with otherfactors such as travel time, distance between stops, and overall route efficiency to identify areas where improvements can be made to enhance the overall quality of bus services.

Overall, the "load_" column is a crucial piece of information for anyone looking to evaluate and improve the performance of bus services. Itcan help us identify problem areas and take targeted steps to enhance passenger experience, increase ridership, and ultimately, improve the overall sustainability of public transportation.

The dataset is from MBTA Opend Data Portal and is a collection of over 5 million records containing records of over 5 years and is a combination of numerical and categorical attributes which makes this dataset complete and diverse. The attribute or column route_id uniquely identifies each bustravelling a specific route through the years and the data is divided into three unique day types namely Weekday, Saturday and Sunday and we will be considering the same for our model. The stop_name uniquely associates with route_id forming a composite key attribute.

Analyzing missing values:

The dataset was explored to verify if there are any null values and found out that there are no null values by checking the sum of null values.

```
season             0
route_id           0
route_variant      0
direction_id       0
trip_start_time    0
day_type_id        0
day_type_name      0
stop_name          0
stop_id            0
stop_sequence      0
boardings          0
alightings         0
load_              0
sample_size        0
ObjectId           0
dtype: int64
```

Descriptive statistics of the predictors for each season:

```
-----------------Descriptive Statistics of route_id for each season----------------

            count  unique  top   freq
season
Fall 2016  859425     172   77  19305
Fall 2017  864428     173   77  19336
Fall 2018  864850     174   77  20267
Fall 2019  863040     169   57  20166
Fall 2020  854771     150   34  24466
Fall 2021  858288     147   34  24570
```

```
------------------Descriptive Statistics of route_variant for each season-----------------

           count unique     top   freq
season
Fall 2016  859425    878  28-_-1  10582
Fall 2017  864428    890  28-_-1  10582
Fall 2018  864850    892  28-_-1  10064
Fall 2019  863040    824  28-_-1  10064
Fall 2020  854771    700  28-_-1  10505
Fall 2021  858288    672  28-_-1  11396

------------------Descriptive Statistics of direction_id for each season-----------------

              count      mean       std  min  25%  50%  75%  max
season
Fall 2016  859425.0  0.499733  0.500000  0.0  0.0  0.0  1.0  1.0
Fall 2017  864428.0  0.499979  0.500000  0.0  0.0  0.0  1.0  1.0
Fall 2018  864850.0  0.499595  0.500000  0.0  0.0  0.0  1.0  1.0
Fall 2019  863040.0  0.500564  0.500000  0.0  0.0  1.0  1.0  1.0
Fall 2020  854771.0  0.501932  0.499997  0.0  0.0  1.0  1.0  1.0
Fall 2021  858288.0  0.500210  0.500000  0.0  0.0  1.0  1.0  1.0


------------------Descriptive Statistics of trip_start_time for each season-----------------

           count unique       top  freq
season
Fall 2016  859425   1252  18:00:00  5079
Fall 2017  864428   1250  18:00:00  4945
Fall 2018  864850   1270  18:00:00  4845
Fall 2019  863040   1264  08:00:00  4334
Fall 2020  854771   1308  15:00:00  3801
Fall 2021  858288   1292  17:00:00  4005


------------------Descriptive Statistics of day_type_id for each season-----------------

            count unique          top    freq
season
Fall 2016  859425      3  day_type_01  423568
Fall 2017  864428      3  day_type_01  422725
Fall 2018  864850      3  day_type_01  418799
Fall 2019  863040      3  day_type_01  414230
Fall 2020  854771      3  day_type_01  405579
Fall 2021  858288      3  day_type_01  410659


------------------Descriptive Statistics of day_type_name for each season-----------------

           count unique      top    freq
season
Fall 2016  859425      3  weekday  423568
Fall 2017  864428      3  weekday  422725
Fall 2018  864850      3  weekday  418799
Fall 2019  863040      3  weekday  414230
Fall 2020  854771      3  weekday  405579
Fall 2021  858288      3  weekday  410659
```

```
------------------Descriptive Statistics of stop_name for each season-----------------

            count unique             top  freq
season
Fall 2016  859425   6690  DUDLEY STATION  5383
Fall 2017  864428   6758  DUDLEY STATION  5328
Fall 2018  864850   6726  DUDLEY STATION  5267
Fall 2019  863040   6586  DUDLEY STATION  5439
Fall 2020  854771   6178  NUBIAN STATION  5540
Fall 2021  858288   6170  NUBIAN STATION  5678
```

```
------------------Descriptive Statistics of stop_id for each season-----------------

             count          mean           std  min     25%     50%     75%  \
season
Fall 2016  859425.0   8932.587651  16218.577337  1.0  1436.0  3562.0  7411.0
Fall 2017  864428.0  10086.908461  19353.605483  1.0  1449.0  3633.0  7624.0
Fall 2018  864850.0  10425.595543  19823.279444  1.0  1470.0  3694.0  7956.0
Fall 2019  863040.0  10512.415562  19901.137829  1.0  1473.0  3733.0  8162.0
Fall 2020  854771.0  10390.038353  19702.339183  1.0  1441.0  3714.0  7881.0
Fall 2021  858288.0  10616.423190  19905.827793  1.0  1468.0  3922.0  8241.0

              max
season
Fall 2016   99991.0
Fall 2017  119913.0
Fall 2018  119913.0
Fall 2019  119913.0
Fall 2020  119913.0
Fall 2021  119913.0
```

```
------------------Descriptive Statistics of stop_sequence for each season-----------------

             count       mean        std  min   25%   50%   75%      max
season
Fall 2016  859425.0  24.847236  31.544052  1.0  10.0  20.0  33.0  13600.0
Fall 2017  864428.0  24.656082  31.394996  0.0  10.0  20.0  33.0  13600.0
Fall 2018  864850.0  25.169322  20.235399  0.0  10.0  21.0  34.0    140.0
Fall 2019  863040.0  24.543175  19.816301  0.0  10.0  20.0  33.0    138.0
Fall 2020  854771.0  23.253278  17.902752  0.0  10.0  19.0  32.0    131.0
Fall 2021  858288.0  23.314379  17.963343  0.0  10.0  19.0  32.0    132.0
```

```
------------------Descriptive Statistics of boardings for each season-----------------

             count      mean       std   min  25%  50%  75%   max
season
Fall 2016  859425.0  0.850347  2.478084   0.0  0.0  0.0  0.7  69.0
Fall 2017  864428.0  0.869022  2.475255   0.0  0.0  0.1  0.8  66.7
Fall 2018  864850.0  0.862901  2.406029   0.0  0.0  0.1  0.8  54.0
Fall 2019  863040.0  0.879158  2.413114   0.0  0.0  0.2  0.8  60.0
Fall 2020  854771.0  0.425046  1.189049   0.0  0.0  0.1  0.4  29.3
Fall 2021  858288.0  0.595673  1.570235 -11.2  0.0  0.1  0.5  58.7
```

```
------------------Descriptive Statistics of alightings for each season-----------------
```

season	count	mean	std	min	25%	50%	75%	max
Fall 2016	859425.0	0.856404	2.607607	-5.0	0.0	0.0	0.8	77.0
Fall 2017	864428.0	0.875009	2.601068	-5.0	0.0	0.1	0.8	69.0
Fall 2018	864850.0	0.865102	2.521548	0.0	0.0	0.1	0.8	67.5
Fall 2019	863040.0	0.881275	2.542070	0.0	0.0	0.2	0.8	63.2
Fall 2020	854771.0	0.426820	1.184171	0.0	0.0	0.1	0.4	33.6
Fall 2021	858288.0	0.597633	1.595038	-5.3	0.0	0.1	0.5	59.6

```
------------------Descriptive Statistics of load_ for each season-----------------
```

season	count	mean	std	min	25%	50%	75%	max
Fall 2016	859425.0	11.666017	9.393657	-2.6	4.7	9.4	16.3	105.0
Fall 2017	864428.0	11.802815	9.371054	-2.5	4.8	9.6	16.5	78.7
Fall 2018	864850.0	11.562608	9.221699	-2.0	4.7	9.4	16.1	71.8
Fall 2019	863040.0	11.623137	9.177135	0.0	4.8	9.5	16.1	77.9
Fall 2020	854771.0	6.056965	4.371077	0.0	2.8	5.2	8.3	35.5
Fall 2021	858288.0	7.802133	5.851568	-17.9	3.4	6.6	10.8	58.7

Visualizing the insights:

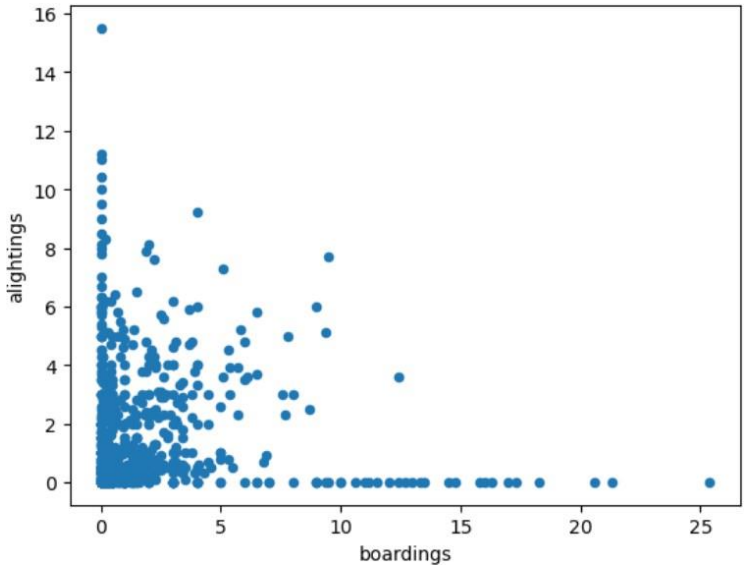


Fig. 1a

The scatter plot (Fig. 1a) displays the relationship between the number of passengers boarding and alighting. There is a weak relationship between the two variables, as there are many data points with low values and the data is spread out. This suggests that there is no clear pattern or trend in the data. The large number of points with low values indicates that many of the stops have very few passengers boarding or alighting. This could be due to factors such as the location of the stop or the time of day. Overall, the scatter plot suggests that there is no strong correlation between the

number of passengers boarding and alighting, and that further analysis may be needed to uncover any underlying patterns or trends.

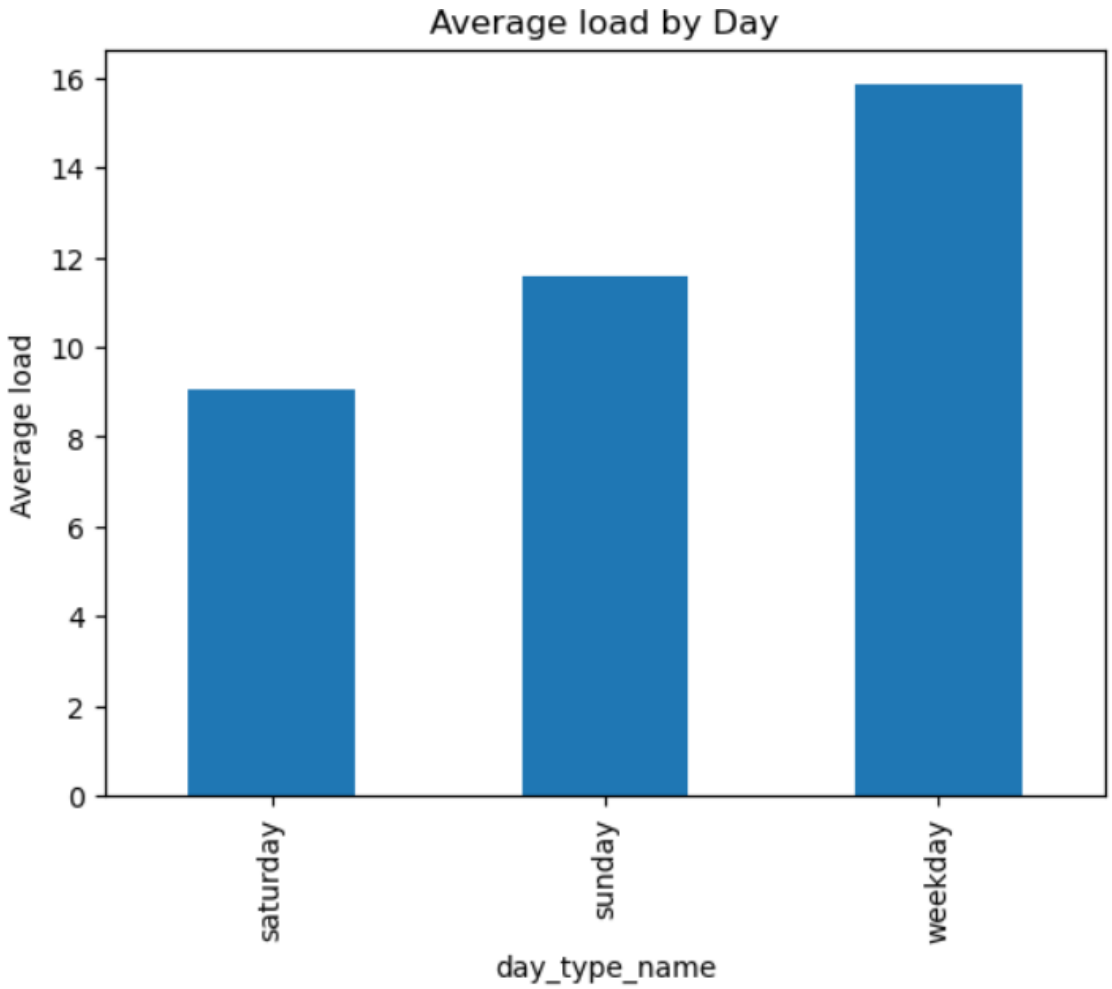


Fig 2a

The bar graph(Fig 2a) depicts the average load on buses on different days of the week. The graph shows that the average load on buses is highest on weekdays, which indicates that the bus services are used more during the weekdays than on weekends. This information can be used to predict the load on buses during weekdays, which can help in the efficient management of buses.

Furthermore, the graph shows that the load on buses is relatively higher on Sundays as compared to Saturdays. This information can be useful for bus operators to adjust their schedules and services on weekends to meet the demand.
Overall, the data presented in the graph can be used to optimize the bus services by providing more buses during the weekdays to accommodate the high load, and adjusting the schedules and services on weekends to meet the increased demand on Sundays. This can help in providing a better and more efficient transportation system for the public.

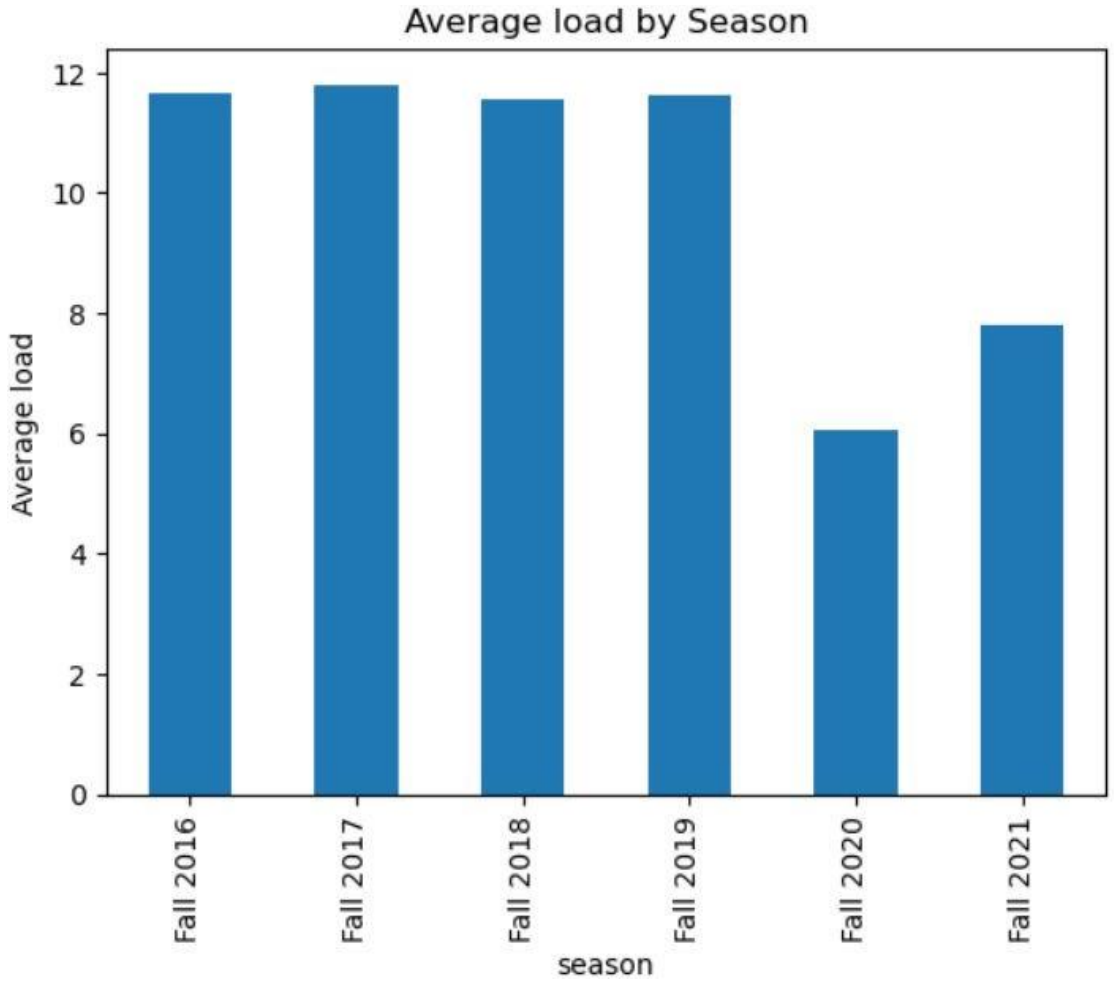


Fig 3a

The bar chart(Fig 3a) provided shows the load on MBTA, which remained consistent over the years until it suddenly dropped to record lows in 2020. This drop can be attributed to the COVID-19 pandemic and the resulting restrictions put in place to stop the spread of the virus. The sudden decrease in load is a significant concern for the MBTA and suggests that they may need to reevaluate their operations and plan for potential future disruptions.

The chart also shows an increase in rider percentage, which could be a positive sign for the MBTA as they work to recover from the pandemic. This increase will be helpful in determining if the MBTA needs to be prepared for future disruptions or if the current trend will continue.

Overall, the chart provides valuable insights into the impact of the pandemic on the MBTA and the potential implications for future operations. By analyzing the data available, the MBTA can make informed decisions about how to best serve their customers and adapt to changing circumstances.

Correlation Analysis of the predictors:

Using the corr() function from pandas library to demonstrate correlationheatmap of the dataset :

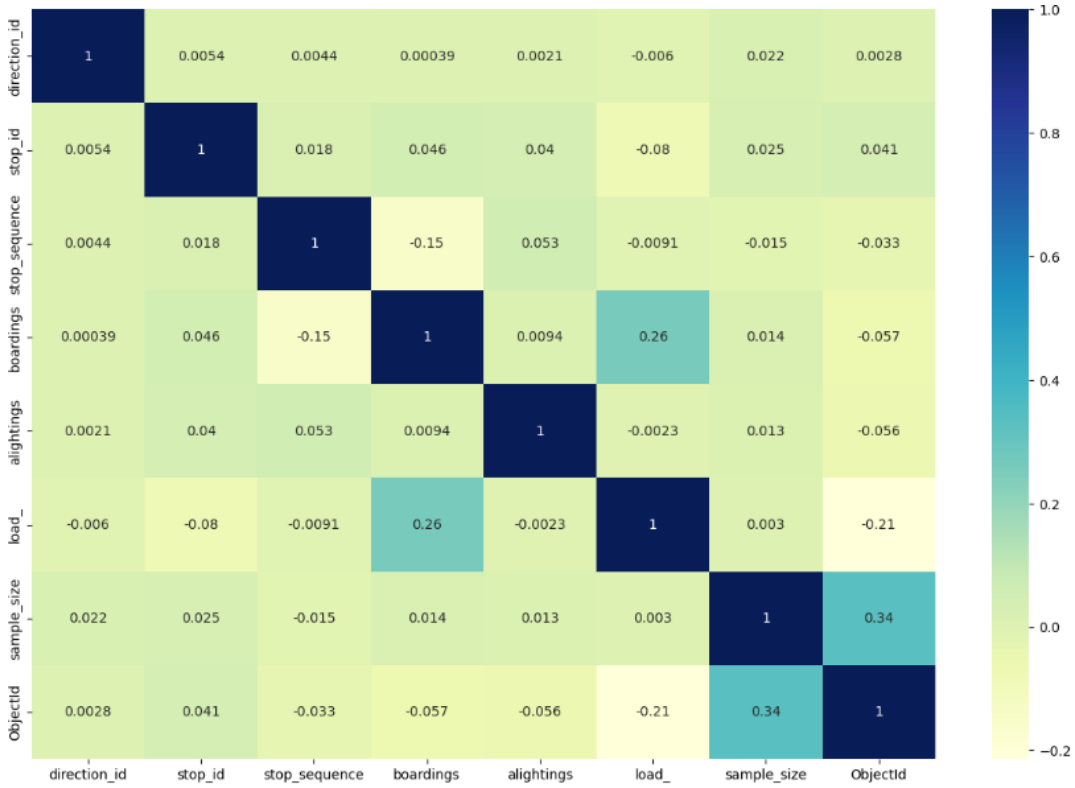


Fig 4a

The heatmap represents the correlation between different predictors. A lower correlation value indicates that the attributes are less related to each other. When all the predictors have less correlation values, it suggests that each attribute is important and can contribute unique information to the model. This information can be valuable in developing predictive models or understanding the relationships between different variables. It is important to consider all the predictors in the model, rather than removing any based on their correlations with each other, to ensure that the model captures all relevant information and produces accurate results.

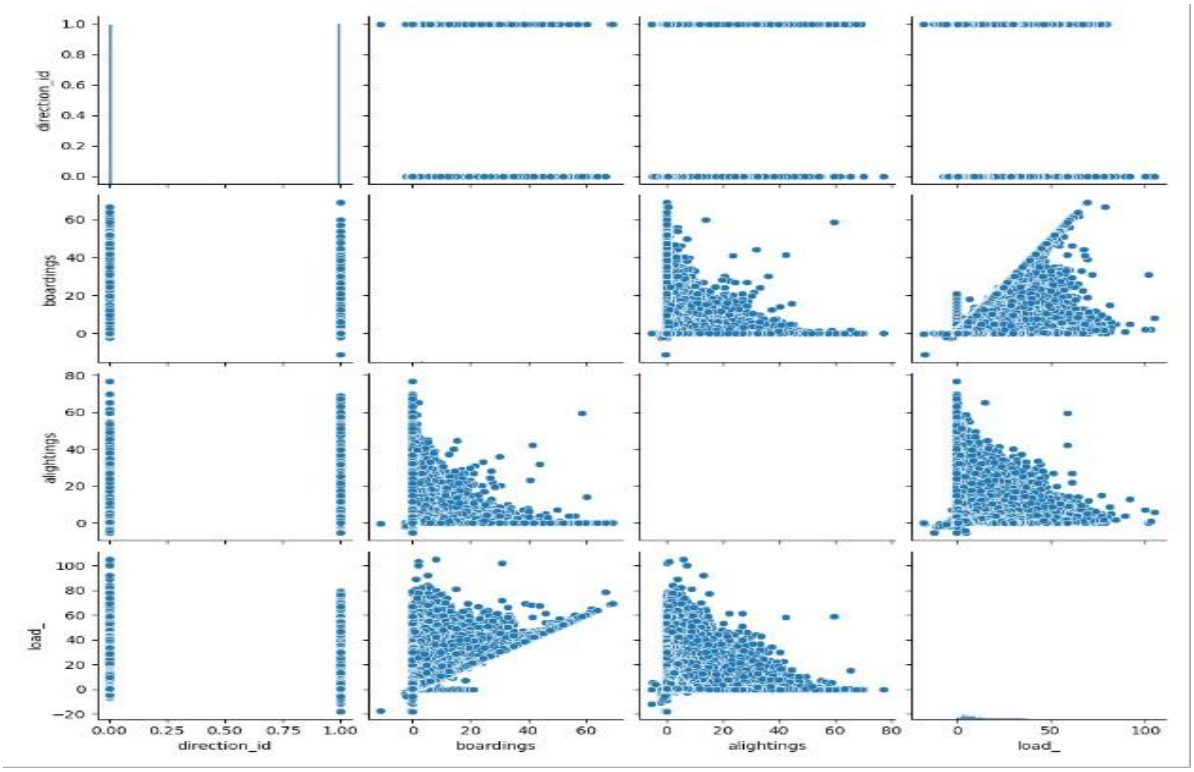


Fig 4b

In the given dataset, a correlation matrix was used to understand the relationship between the four predictors. The analysis showed that the attributes were not highly correlated with each other. However, there was some correlation between the "load" and "boardings" attributes. This indicates that these two attributes may have some relationship, and they should be considered together in any predictive model.

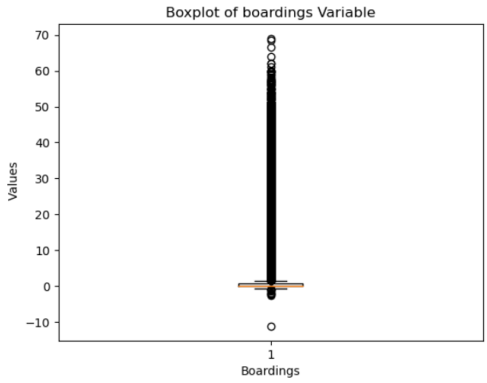


Fig 5a

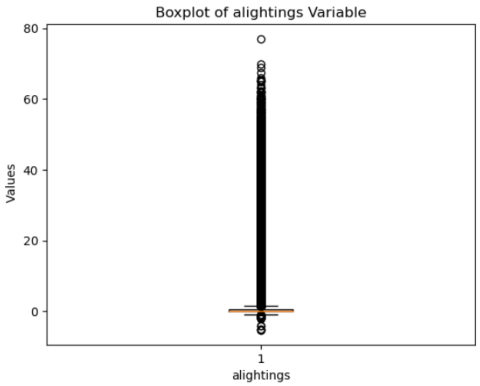


Fig 5b

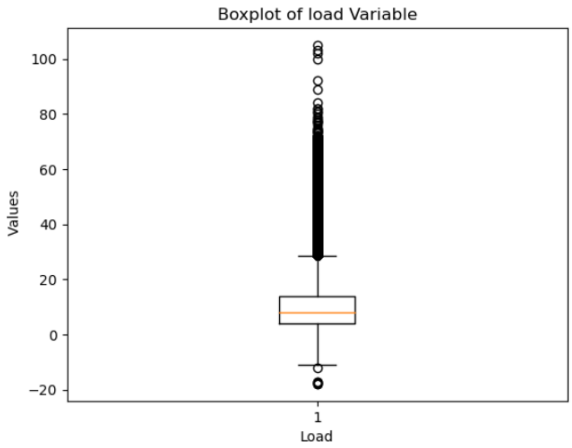


Fig 5 c

Fig 5 a,b and c shows the distribution of the numerical variables in our data. Its is evident that most of the data is centered within a range. Also with the load data we could see that there are few outliers present.

Dimension reduction and variable selection

Based on this analysis above, it was concluded that dimension reduction was not required for this dataset. Each attribute had some level of importance in predicting the load of the bus, and keeping all the attributes would enable the model to capture all the relevant information and provide accurate predictions. Therefore, it was recommended to include all the predictors in any model developed to predict the load of the bus.

In summary, the correlation matrix analysis provided valuable insights into the relationship between the predictors and helped to determine the importance of each attribute in predicting the load of the bus. This information is crucial in developing an accurate predictive model that can be used to improve the efficiency of bus operations.

Model exploration and model selection

Based on the dataset what we have, it is best suitable to build a regression model to predict the average number of people on a bus at a particular time. We can use regression for this problem as we are predicting a numerical value.

There are many regression models. Among them we ran our data in the below models and check the performance.

We tried to analyze few of the regression models and below are the observations:

1. Linear Regression

Pros:

- It is simple to implement and easier to interpret the output coefficient
- It is less complex when compared to other algorithms
- It is susceptible to over-fitting but it can be avoided using some dimensionality reduction techniques, regularization (L1 and L2) techniques and cross-validation

Cons:

- Outliers can have huge effects on the regression and boundaries are linear in this technique
- linear regression assumes a linear relationship between dependent and independent variables. That means it assumes that there is a straight-line relationship between them. It assumes independence between attributes which might not be true all the times.
- linear regression also looks at a relationship between the mean of the dependent variables and the independent variables. Just as the mean is not a complete description of a single variable, linear regression is not a complete description of relationships among variables.

Performance of Linear regression:

R2 value: 0.17
RMSE: 59.17

2. K- Nearest Neighbours (KNN) Algorithm

Pros:

- K-NN algorithm is very simple to understand and equally easy to implement. To classify the new data point K-NN algorithm reads through whole dataset to find out K nearest neighbors.
- is a non-parametric algorithm which means there are assumptions to be met to implement K-NN. Parametric models like linear regression has lots of assumptions to be met by data before it can be implemented which is not the case with K-NN.
- K-NN does not explicitly build any model, it simply tags the new data entry based learning from historical data. New data entry would be tagged with majority class in the nearest neighbor

Cons:

- K-NN might be very easy to implement but as dataset grows efficiency or speed of algorithm declines very fast.
- KNN works well with small number of input variables but as the numbers of variables grow K-NN algorithm struggles to predict the output of new data point.
- K-NN algorithm is very sensitive to outliers as it simply chose the neighbors based on distance criteria.
- K-NN inherently has no capability of dealing with missing value problem.

Performance of KNN regression:

R2 value: 0.68
RMSE: 24.38

3. Decision Tree Algorithm

Pros:

- Compared to other algorithms decision trees requires less effort for data preparation during pre-processing
- A decision tree does not require normalization of data
- Missing values in the data also do NOT affect the process of building a decision tree to any considerable extent
- It is very intuitive and easy to explain to technical teams as well as stakeholders

Cons:

- A small change in the data can cause a large change in the structure of the decision tree causing instability
- Decision tree often involves higher time to train the model
- Decision tree training is relatively expensive as the complexity and time has taken are more

Performance of Decision Tree:

R2 value: 0.79
RMSE: 15.2

4. Random forest Algorithm

Pros:

- Handles both linear and nonlinear relationships between features and target variables.
- Can handle large datasets with high dimensionality and many features.
- Can handle missing values and imbalanced datasets.

- Provides feature importance measures, which helps in feature selection and understanding the importance of each feature in the model.
- Can handle both continuous and categorical features.

Cons:

- Can overfit the training data, especially if the number of trees is too large.
- May not work well with small datasets as it needs a large number of samples to build an accurate model.
- Can be slow and computationally expensive to train, especially with large datasets.
- The model is not interpretable in terms of the relationships between features and target variables.
- May not be able to capture complex interactions between features.

  Performance of Random forest regression:

  R2 value: 0.88

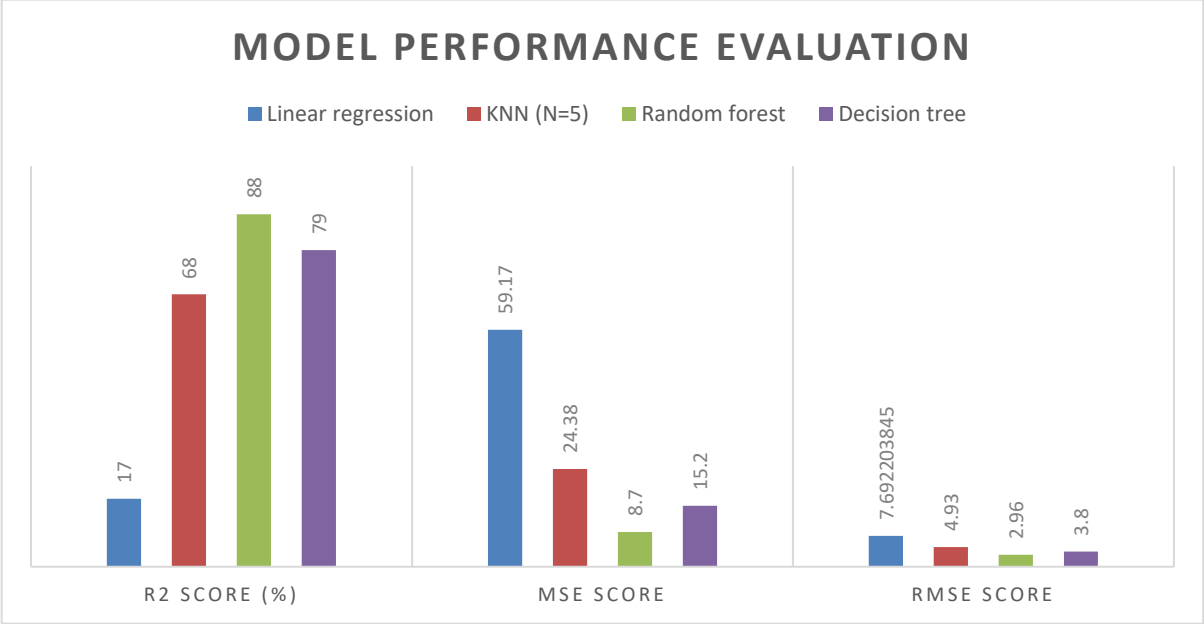
  RMSE: 8.7

Model Performance Evaluation and visualization

We pre-processed the data to make sure its cleaned. Then we encoded the data using Label Encoder and Frequency encoder to convert the categorical data into numeric.
Then, we fit our data with different Machine learning models and analyzed their performance.

Below is the summary of all the models that we tested and their performance values.

Model	R2 score (%)	MSE score	RMSE score
Linear regression	17	59.17	7.692204
KNN (N=5)	68	24.38	4.93
Random forest	88	8.7	2.96
Decision tree	79	15.2	3.8



From the above, we can infer that Random forest regressor gave the best prediction of the bus load. It also has a higher R2 value indicating proportion of variance in the response variable that is explained by the predictor variables in the model. It also has relatively less MSE and RMSE values showing less error values. Hence, random forest regressor is the better predicting model for our dataset.

Model hyper parameter tuning:

Hyperparameter tuning is the process of selecting the optimal set of hyperparameters for a machine learning algorithm to improve its performance on a given dataset. Hyperparameters are parameters that are not learned during the training process, but instead are set before training begins.

For example, in the case of a random forest algorithm, the number of trees in the forest, the maximum depth of each tree, and the number of features to consider at each split are all hyperparameters.

Hyperparameter tuning is done because the default hyperparameters of a machine learning algorithm may not be optimal for a specific dataset, and therefore may result in suboptimal performance. By tuning the hyperparameters, we can find the best combination of hyperparameters for a given problem, which can lead to significant improvements in model accuracy and generalization.

We tried different parameters for our random forest model using RandomizedsearchCV function by the Sklearn library since the dataset is very huge.

These are the parameters tested:

```
param_grid = {
    'n_estimators': [10, 20, 30],
    'max_depth': [None, 5, 10, 20],
   # 'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}
```

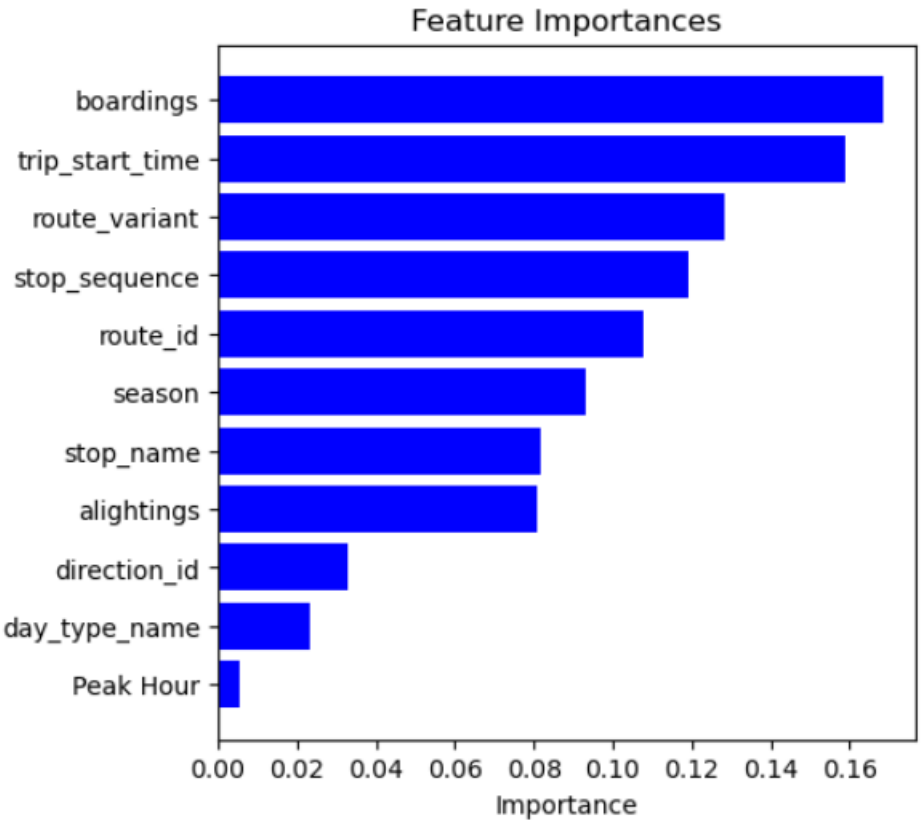
Below is the result from the test:

```
Fitting 3 folds for each of 10 candidates, totalling 30 fits
Best Parameters: {'n_estimators': 30, 'min_samples_leaf': 4, 'max_depth': None}
Best Score: 0.8481635887669707
```

We can see that the combination of 30 trees gave us the optimal performance results.

Model understanding

A simple way of understanding the results of a model is to look at feature importances. Feature importances indicate the importance of a feature within the predictive model, there are several ways to calculate feature importance, but with the Random Forest classifier, we're able to extract feature importances using the built-in method on the trained model. In the Random Forest case, the feature importance represents the number of times each feature is used for splitting across all trees.



From the above chart, we can observe the following points:

- Number of passengers boarding the bus is the top driver for the load as expected.
- Trip start time is also an important influencer of the load
- Surprisingly, day of the week does not have that impact on the load variable.

Impact of Project outcomes:

The main goal of the project is to accurately predict the load on a bus on a given trip day. This is done by using the real-world data published by MBTA. By leveraging the predictive power of the above model, we hope to help the organization make better business decisions. This is expected to help them with making better scheduling and also to cut down on losses due to low turnover of the passengers. Out of all the regression models we tested, random forest predicted the load better. It could also manage the large volume of data that we provided with good performance.

References:

1. MBTA ridership information: https://mbtamassdot.opendata.arcgis.com/search?tags=bus
2. Pandas functionality: https://levelup.gitconnected.com/20-pandas-functions-for-80-of-your-data-science-tasks-b610c8bfe63c
3. RandomizedsearchCV: https://towardsdatascience.com/machine-learning-gridsearchcv-randomizedsearchcv-d36b89231b10