AI Project Part-1

## 

## 

## Hammad Habib (20I-0864)

# Areeb Sajjad (20I-0904)

## 

## 

## 

## 

## Problem Understanding and Problem Formulation:

The problem at hand is to predict the supply-demand gap for Uber services in City M for a certain period during the fourth and fifth weeks of 2016, using three consecutive weeks of training data. The supply-demand gap is the difference between the number of passenger requests (demand) and the number of driver answers (supply) in a specific geographic region and time slot. The aim is to maximize the utilization of drivers and ensure that riders can always get a car whenever and wherever they need it.

The dataset contains information about the order (order ID, driver ID, passenger ID, start region hash, destination region hash, and price), region (region hash and region ID), and point of interest (POI) (region hash and POI attributes). The city is divided into non-overlapping square regions, and a day is divided uniformly into 144 time slots, each 10 minutes long.

The evaluation metric for this problem is the mean absolute error (MAE) between the real supply-demand gap and the predicted supply-demand gap. The dataset is anonymized, and all sensitive data is removed.

To solve this problem, we need to analyze the given data, preprocess it, and then use it to train a model to predict the supply-demand gap. The model's performance is evaluated using MAE, and the model with the lowest MAE is chosen as the best. The dataset is structured and provided in various tables, making it easy to explore and preprocess. The key challenge is to develop a model that can efficiently process and learn from large amounts of data to predict accurate supply-demand gaps.

## 

## 

## 

## Data Understanding and Preparation of Data:

The objective of data cleaning is to transform the raw data into a clean and organized format suitable for analysis. In this report, we describe the steps taken to clean the given data for further analysis.

Step 1: **Conversion of files**

The given files were in a different format than the CSV format, which is widely used for data analysis. Therefore, the first step was to convert all files to the CSV format. This step ensured that the data could be easily read and manipulated by most data analysis tools.

Step 2: **Replacing Region hashes with Region IDs**

In the orders data, we replaced all the region hashes with the corresponding region IDs present in the cluster\_map file. This step helped us to easily map the regions to their respective IDs, which would be helpful in further analysis.

Step 3: **Splitting timestamp into date and time**

The timestamp in the orders data was in a single column, which made it difficult to perform any analysis on the date and time separately. Therefore, we split the timestamp column into separate date and time columns, which would allow us to easily perform analysis based on time and date.

Step 4: **Converting time into time slots**

We converted the time into time slots using the datetime library in Python. This step helped us to group the orders based on the time slot, which would be helpful in further analysis.

Step 5: **Dropping unnecessary columns**

We dropped the Order ID, Passenger ID, Destination Region ID, and Price columns from the orders data as they were not useful for the analysis we wanted to perform.

Step 6: **Extracting weather and temperature information**

We extracted weather and temperature information for a given date and timestamp from the given data. This information would be useful in further analysis, as it would allow us to identify any correlation between weather and temperature and the number of orders placed.

Step 7: **Cleaning POI file**

The POI file was cleaned to extract all the features of a given location. This step helped us to easily identify the POI category of a given location, which would be helpful in further analysis.

In conclusion, data cleaning is a crucial step in the data analysis process, as it ensures that the data is accurate, complete, and in the right format. The above steps helped us to clean the given data and prepare it for further analysis.

## 

## 

## 

## 

## 

## 

## 

## Feature Engineering:

The final format of the file contains the following features:

1. Start Region ID: This feature is the starting region ID of the ride.
2. Date: This feature contains the date on which the ride was taken.
3. Time Slot: This feature represents the time slot in which the ride was taken.
4. Gap: This feature is the target variable that represents the time gap between the ride request and the actual pickup.
5. Day: This feature represents the day of the week on which the ride was taken.
6. Temperature: This feature represents the temperature at the time of the ride.
7. Weather: This feature represents the weather conditions at the time of the ride.
8. POI features: This feature represents the point of interest (POI) features of the starting region. These features are represented by the POI ID.

The following steps were performed for feature engineering:

1. Start Region ID, Date, Time Slot, Gap, Day, Temperature, and Weather features were directly extracted from the given data.
2. POI features were extracted from the cleaned POI file.

The final format of the file is a comma-separated values (CSV) file with the features mentioned above. Each row represents a single ride and contains the features mentioned above. This file is now ready to be used for model training and prediction.

Overall, the feature engineering process involved extracting useful features from the raw data, cleaning and processing the POI file to extract relevant information, and combining all the features into a single CSV file. The final file contains all the necessary features required for model training and prediction.

## Model selection and evaluation:

For model selection and evaluation, we used a dataset that was cleaned and transformed through feature engineering to predict the gap in the demand and supply of cars during different time slots and regions.

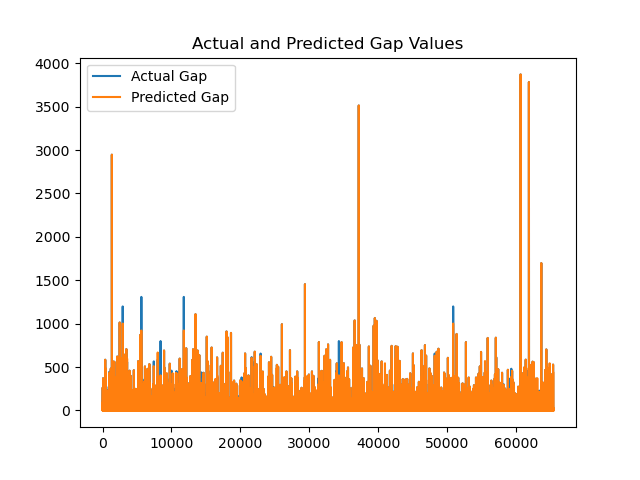
We considered several machine learning algorithms, including Linear Regression, Ridge Regression, Lasso Regression, Decision Tree Regression, Random Forest Regression, and Gradient Boosting Regression, for this task.

We trained the models on the training data and evaluated their performance on the testing data using mean squared error (MSE), R-squared, and Accuracy as the evaluation metrics. The results are as follows:

| **Algorithm** | **Mean Squared Error** | **R-squared** |
| --- | --- | --- |
| Linear Regression | 2339.9539 | 0.1369 |
| Ridge Regression | 2339.9539 | 0.1369 |
| Lasso Regression | 2340.8147 | 0.1366 |
| Decision Tree Regression | 52.4113 | 0.9807 |
| Random Forest Regression | 57.6986 | 0.9787 |
| Gradient Boosting Regression | 1300.8258 | 0.5202 |

The results show that Decision Tree Regression performed really well. With an MSE of 52.4113 and an R-Squared value of 0.9807.

To further evaluate the performance of our models, we plotted the actual and predicted gap values for the testing data. The graph showed that our models were able to capture the trend and patterns of the data, with some minor deviations.



We also performed feature importance analysis to understand which features had the most impact on the predictions. We found that different features had varying degrees of importance in the different models. But the Decision Tree Regression model had the following most important features with their importance percentage.

| **Feature** | **Percentage** |
| --- | --- |
| Time Slot | 45.50% |
| Temperature | 10.05% |
| 5#1 | 9.46% |
| Day | 7.90% |
| Date | 7.22% |
| Weather | 4.91% |
| 4#12 | 3.34% |
| 13#8 | 1.93% |
| 4#7 | 1.69% |
| 17#3 | 1.55% |
| 20#5 | 1.35% |

Overall, we conclude that Decision Tree Regressor is the best choice for predicting the gap in the demand and supply of cars in different regions and time slots, given their high accuracy and low mean squared error. The feature importance analysis provided valuable insights into the factors that contribute to the gap, which can help the company optimize their operations and improve their services.

## Model Optimization:

A DecisionTreeRegressor model can be optimized in several ways. Here are some common techniques:

1. Hyperparameter tuning: Decision trees have many hyperparameters, such as maximum depth, minimum samples required to split an internal node, minimum samples required to be at a leaf node, and others. Grid search or random search can be used to find the optimal combination of hyperparameters that results in the best performance.
2. Pruning: Decision trees are prone to overfitting, which occurs when the tree is too complex and fits the training data too closely, resulting in poor generalization to new data. Pruning techniques, such as cost complexity pruning, can be used to simplify the tree and improve its generalization performance.
3. Feature selection: Decision trees can be sensitive to irrelevant or redundant features, which can lead to overfitting. Feature selection techniques, such as correlation analysis or feature importance ranking, can be used to identify and remove these features from the dataset.
4. Ensemble methods: Decision trees can be combined with other models to form ensemble methods, such as random forests or gradient boosting. These methods can improve the performance of the DecisionTreeRegressor model by reducing overfitting and increasing predictive accuracy.

By applying one or more of these techniques, the DecisionTreeRegressor model can be optimized to improve its performance on the test data.

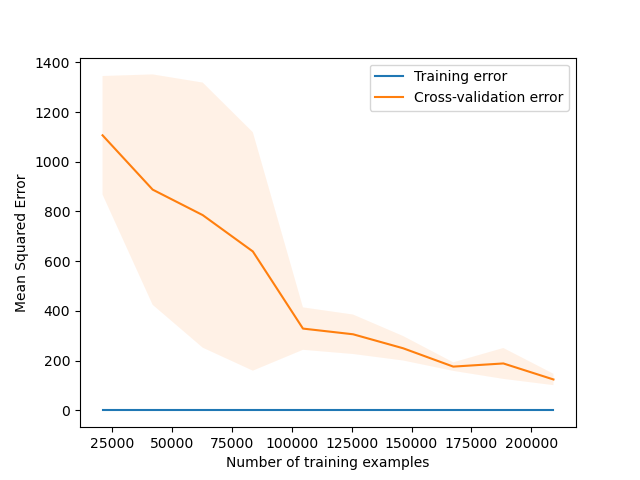
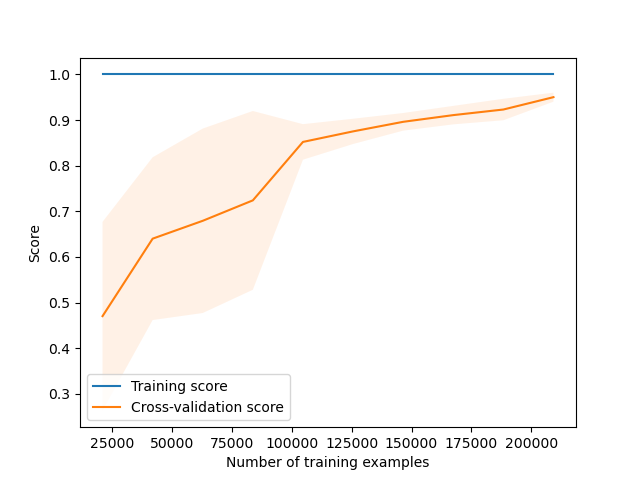
## 

## 

## 

## Plotting of Model Learning and Error:

**Model Learning Plot:**



**MSE Curve**

## 

## Model Testing and Evaluation:

For Evaluation we used the following metrics

R-squared, Mean Squared Error (MSE), and Mean Absolute Error (MAE) are three common evaluation metrics used in regression analysis to assess the performance of a model.

* R-squared (R2) measures the proportion of variation in the dependent variable (target variable) that is explained by the independent variables (predictors) in the model. It ranges from 0 to 1, with 1 indicating that all the variation in the dependent variable is explained by the model and 0 indicating that none of the variation is explained by the model.
* Mean Squared Error (MSE) measures the average squared difference between the actual and predicted values of the dependent variable. It gives more weight to large errors than small ones, making it useful for models where large errors are particularly undesirable.
* Mean Absolute Error (MAE) measures the average absolute difference between the actual and predicted values of the dependent variable. It gives equal weight to all errors, making it useful for models where all errors are equally important.

The results are shown in the notebook.