Markus Müller

Task for Course: DLBDSME01 – Model Engineering

Task 2: Development of a Demand Forecasting Model for Public Transport

## Table of Contents

# 1. Introduction

## 1.1. Background and Problem Statement

Importance of demand forecasting for public transport:

When dealing with public transportation, an organisation must be aware of the demand at all times. They need to prepare the logistics to solve the transportation problem and have a deep understanding of their time management. When demand is high, more public transportation should be available, when demand is low less public transportation should be available.

To solve these two problems an organization must have some kind of system to forecast the demand for public transportation. This model also needs to be highly accurate to satisfy the customer's needs to successfully get them from point a to point b and on time.

## 1.2. Objective of the Study

The goal of this case study is to write a regression model to predict the average demand of taxis per hour over the day in the ten most important clusters. To identify these important clusters the data needs to be evaluated using clustering algorithms such as K-Nearest-Neighbour

## 1.3. Project Structure and Repository Structure

As one of the data scientists from the American transportation company addressing this issue, one must also understand the importance of a well-organized project structure, for collaborative endeavours and reproducibility.

A shared GitHub repository can help in solving the issues when working in bigger teams and complex code with different versions.

Proposed git repository structure:

- "data/" – raw & processed data files
- "models/" – trained models
- "plots/" – generated visualizations
- "docs/" - documentation and related texts
- "maps/" – heatmaps and maps

**GitHub URL: "https://github.com/kmeans27/model_engineering__demand_forecasting"**

# 2. Data Understanding

## 2.1. Initial Data Exploration

Now that the GitHub repository is created, we need to get a first look at the data. The data for this task, which can be downloaded from the IU website contains the following files:

- Apr14.csv
- May14.csv
- Jun14.csv
- Jul14.csv
- Aug14.csv
- Sep14.csv
- Data-on-map.py

The data for this task consists of 7 files, 6 of these files are raw CSV data files in the following structure: “Date/Time”, “Lat”, “Lon”, “Base”. After opening the apr14.csv file in excel and performing the text to column functionality to separate the data by columns and adding a filter on top of it, one can see that the “Base” column only has 5 different values. The “Base” column is an identifier for the taxi dispatching base or company. We can use the “Base” column in the following ways:

- Categorical feature
- Analyse demand by base
- Temporal analysis by base
- Geospatial analysis by base

The data-on-map.py script simply does what the name of the script says. It uses the pandas and folio libraries to display part of the data on the map which is then saves as a .html file.  The script also gives an overview of how many rows are in the dataset “apr14.csv” which equals to 564516.

### 2.2. Data Quality Assessment

The data appears to be comprehensive and devoid of missing values from the perspective of data quality. With the use of the available data, one can identify bases with the highest or lowest demand as well as high demand regions, times, and days. However, it would have been better if there had been more columns. Demand is not primarily determined by the place or the time of day. There are more crucial aspects to take into account as well, such as:

- Weather-related (precipitation, temperature)
- Events and holidays
- Socio-Economic and demographic data (income levels, age distribution)
- Infrastructure (public transport accessibility, parking availability)
- Fare prices

# 3. Data Preparation

## 3.1. Data Cleaning

### 3.1.1. Get data into right structure

To get the data into the right structure for further analysis, one must first combine all the csv files into one single csv file – to make some first assumptions and evaluations. This can be achieved by using the python library pandas.
*df = pd.concat((pd.read_csv(file) for file in files), ignore_index=True)*
*df.to_csv('combined_data.csv', index=False)*

These two combine all the csv files into a new combined_data.csv file. One cannot opt for excel to achieve this because excel has a maximum date rows limit which is reached in this project.

### 3.1.2. Outliers’ detection and treatment

To keep the data as accurate as possible only the most extreme values need to be removed. One can see in the scatter plot, that some extreme values are included in this dataset. These can be quickly filtered out by using df.quantile(%). The boundaries for latitude and longitude are set to 0.1% and 99.9%. This removed about 16.000 rows. This process helped immensely, to focus on the most important clusters and leave out the ones with weak demand.

## 3.2. Feature Engineering

### 3.2.1. Time-based features (e.g., hour of the day, day of the week)

By breaking down the “Date/Time” column into smaller units like hour, day and month, one could detect hourly, daily or monthly (seasonal) demand patterns. Examples for time-based features include:

- **Hourly patterns**: People living in cities are rushing to their workplaces in the morning and heading back home in the evening. We would expect taxi demand to fall during the lazy afternoon hours, between 13:00-16:00. Otherwise, we could expect taxi demand to surge at around 09:00 and 17:00. Sunday morning between 00:00-04:00 could also see huge demand as people are heading home after a night out.
- **Daily & Weekly Patterns**: The weekends can be very busy in big cities like New York. There will most likely be a higher demand for taxis on Friday, when people are heading out. This also goes for Saturday, where people are not rushing to work but getting out in the city to experience the nightlife.
- **Seasonal Variations**: People may choose to use taxis on chilly winter evenings rather than face the weather since they are more comfortable indoors. In contrast, demand may decline in the summer when people choose to enjoy the sun by riding or walking. Nevertheless, because this course does not offer a full year's worth of data, it may be quite difficult to identify the seasonal variations.

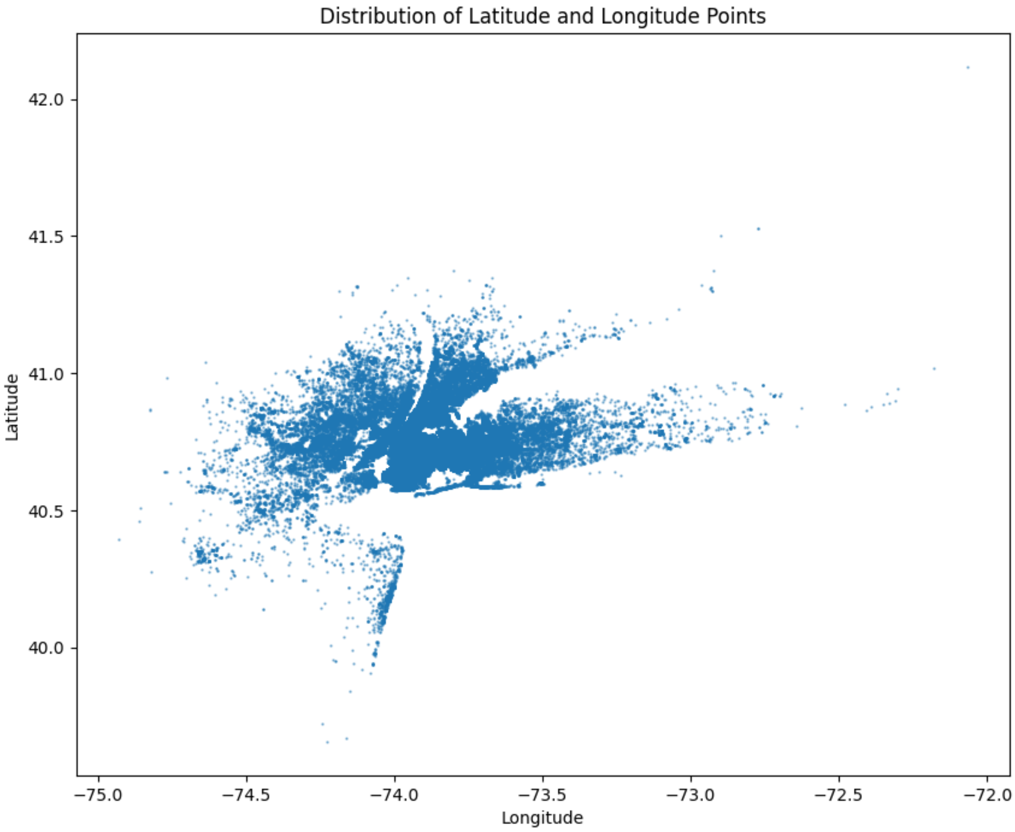
### 3.2.2. Geographical features (e.g., latitude and longitude)

Latitude and longitude can be analysed in different ways to detect high or similar regions of demand:

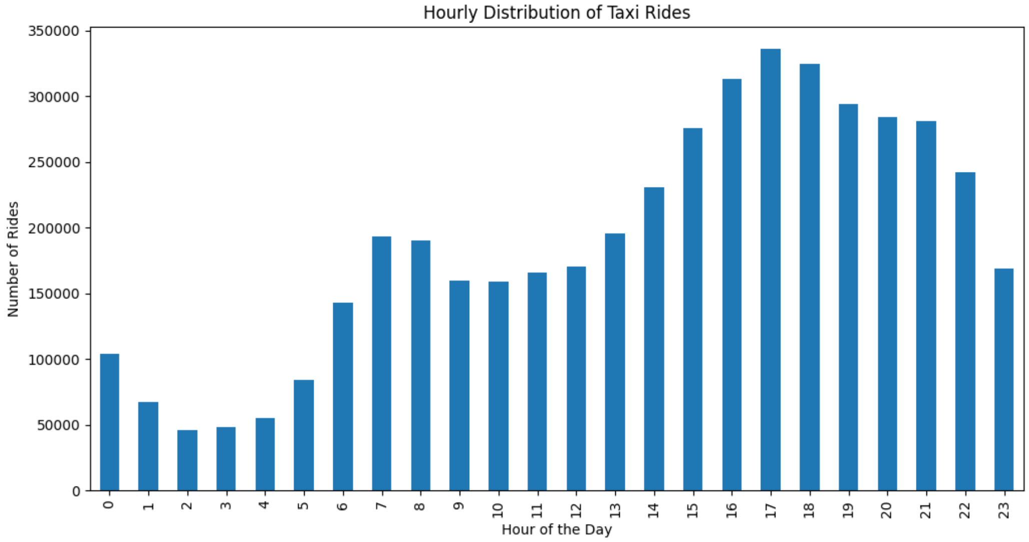
- **Clustering using Latitude and Longitude**: By implementing clustering algorithms such as K-Nearest-Neighbour on our latitude and longitude data, we can identify areas with similar demand, which can be used to identify these hotspots of taxi activity.
- **Distance to important Landmarks**: Major city hubs can be magnets for taxi demand. Areas like the central park on a sunny day can surge taxi demand even more on days of popular events. One may estimate the potential demand pull these landmarks may have by computing the haversine distance to these landmarks.

## 3.3. Data Visualization

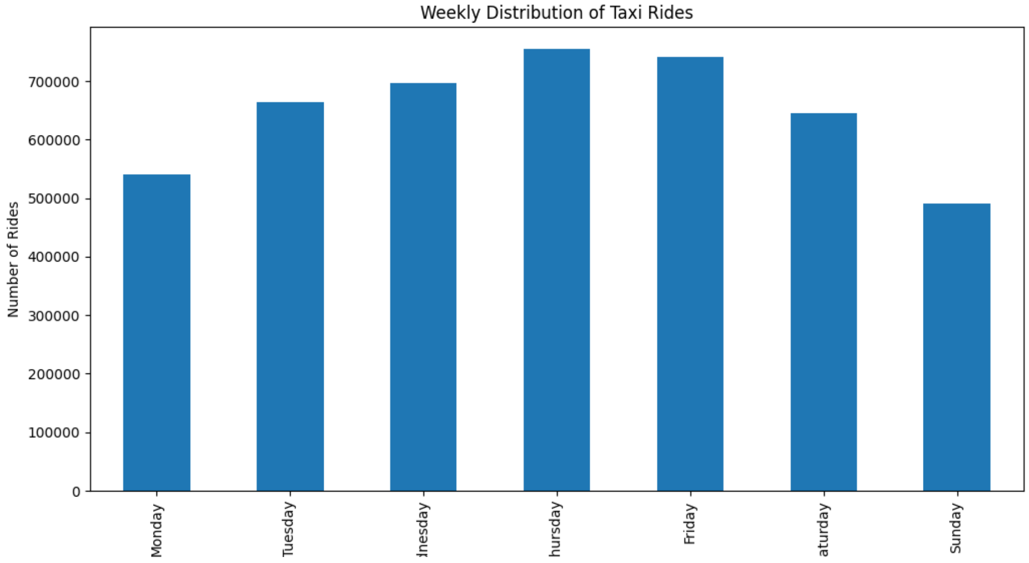
### 3.3.1. Distribution of taxi demand over time



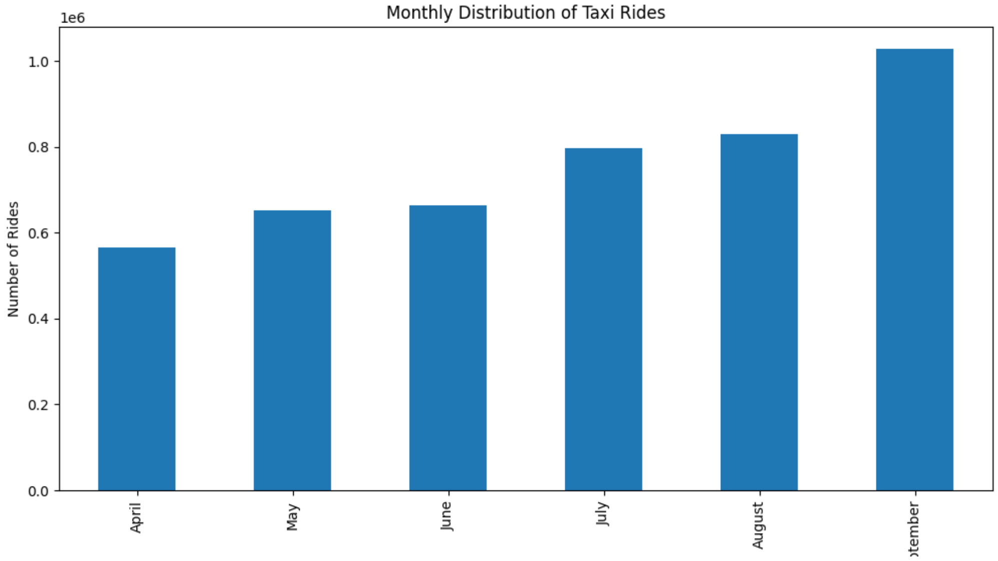
This scatter plot provides a basic visualization of taxi pickup location distribution on latitude and longitude. This is the original dataset so no extreme values have been removed to better demonstrate the full point of view.



By looking at this plot, one can clearly confirm some of the assumptions made earlier. Taxi demand peaks at 17 o’clock and the demand is noticeably higher in the evening than in the morning. Also, between 7 and 8 o’clock demand is also high because people are rushing to work.



By observing the weekly distribution, one can see that the most demand is on Thursday across all 6 months included in this dataset, but demand on Friday is almost as high. Sunday has the least demand.



Analysing the monthly distribution, one cannot identify any other trends than that demand increases over time.

# 4. Modelling

## 4.1. Clustering Approach

The task description states that we can use any clustering algorithm. The script also suggests using the K-Nearest-Neighbour clustering algorithm, but the algorithms is often used for classification and regression tasks, not for clustering. So, for this reason, the k-means algorithm is used for this task.

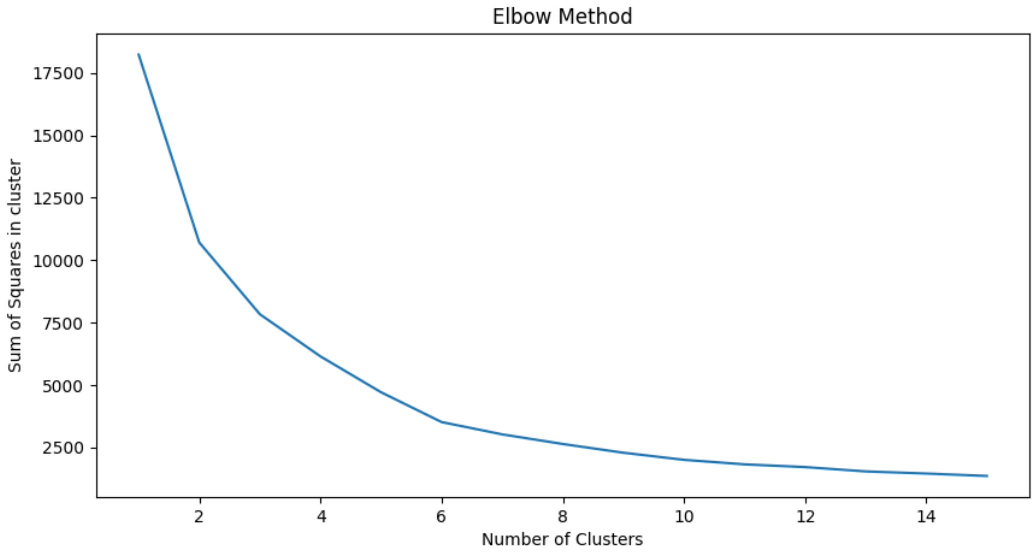
### 4.1.1. K-Means Clustering

To use the k-means algorithm in python, the scikit-learn library can be used.
*from sklearn.cluster import KMeans*

But before clustering the dataset and continuing with the training of the regression model, the numbers of clusters must be defined. To find the optimal numbers of clusters the elbow method can be used.

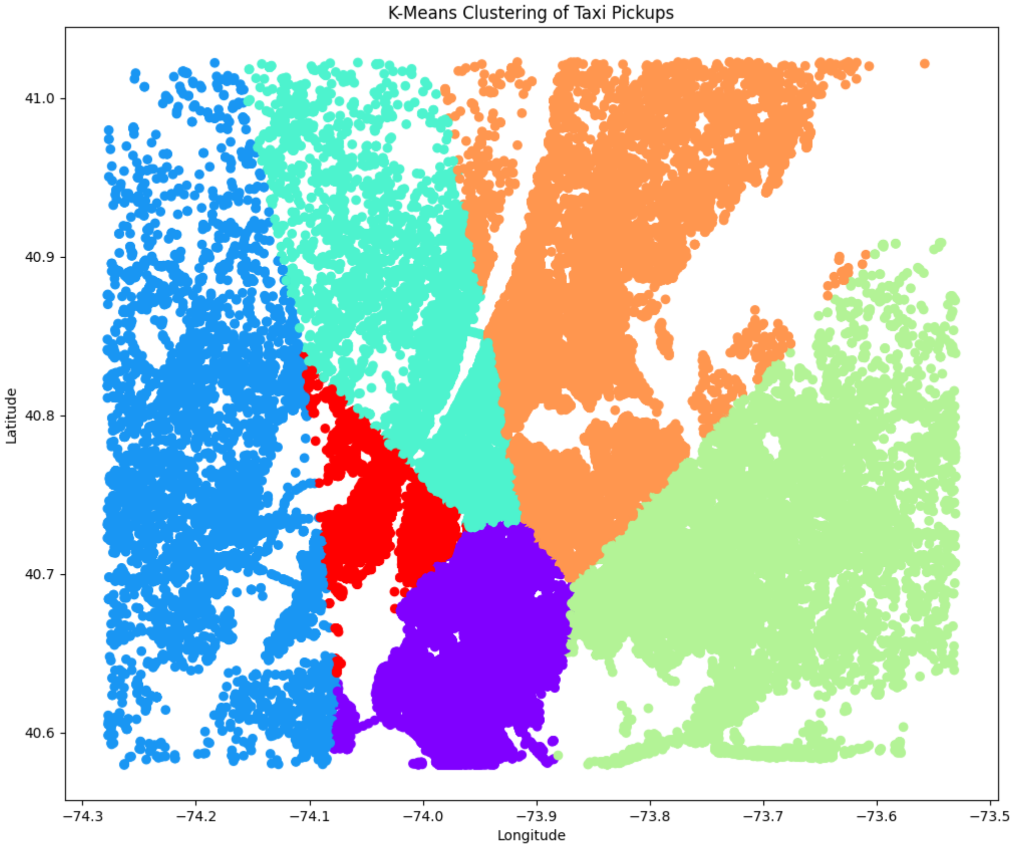
### 4.1.2. Determining the Number of Clusters

The elbow method runs k-means clustering on the dataset for a range of values(k), then computes the sum of squared distances from each point to its classified centre. As **k** increases, the sum of squared distances tends to zero. The goal is to find a small value for k, that has a low sum of squared distances.



Looks like the best number of clusters is k=6, according to this visualization. But the result is not perfectly clear. K = 3 also states a clear decline in sum of squares within cluster, but the sum of squares within a cluster would be too high, which is not what we want to achieve in this implementation, so 6 clusters seem promising.

### 4.1.3. Cluster Visualization



Here is the clustering result for a number of 6 clusters. As we determined by the elbow method before, that 6 clusters would be the most effective approach for this exact task. This looks very informative, as the cluster are clearly divided into straight lines. 4 of these clusters are covering big areas, whereas two cover more dense areas (purple, red).

## 4.2. Regression Models

Before training the regression models, it is important to have the data aggregated to reflect the average demand (vehicle count) for each cluster per hour. This pre-processing procedure simplifies the complicated dataset into a format that is easier to handle for predictive modelling. Effective data aggregation is achieved by using the” groupby” function from the pandas library.

### 4.2.1. Model selection (e.g., linear regression, decision trees, etc.)

It's crucial to choose the right regression algorithm. Several factors led to the selection of the RandomForestRegressor:

- It's well-suited for capturing the non-linear relationships that are characteristic of demand data, where time and location variables interact in complex ways.
- RandomForest requires minimal preprocessing, which is advantageous given the nature of the taxi dataset.

### 4.2.2. Training a Regression Model per Cluster

To estimate the average_demand, the data must be aggregated again before the model is trained. The features "Cluster" and "Hour" are used in this stage in accordance with the task description.

In order to prevent overfitting and guarantee the generalizability of the model, the dataset was then divided using an 80-20 train-test ratio. This splitting is achieved by using the "train_test_split" method from sklearn.model_selection.

After initializing the RandomForestRegressor from sklearn.ensemble, the model is trained and predictions are made using the .fit() and .predict() methods.

# 5. Evaluation

Before deploying a model to production to provide predictions for a real-world use case, it must first be evaluated. Metrics like the mean squared error (MSE) and the coefficient of determination (R) are useful to determine a model's accuracy and reliability

The results are as follows:

- **Mean Squared Error: 65119024**
  A mean squared error which analyses a column which is in the thousands does not indicate that much but nevertheless the mean squared error is quite high.
- **R² score: 0.955***
  An R² score of 0.955 indicates about 95,5% of the variability in the predicted variable. the model is performing well, however it's also possible that the model just matches the training set of data well.

## 5.1. Parameter Tuning and Feature Enhancements

As for parameter tuning, it looks like the random forest regressor has already found quite suitable parameters for the dataset, to further increase the models accuracy one could use more than just the “cluster” and “hour” features for predicting the average demand.

In the data quality evaluation above, is already mentioned which features could be used to make the model as accurate as possible, but these features requires new dataset outside of the tasks directory. Some features one could implement nevertheless:

- Day of week
  - Weekdays
  - Weekends
- Month (Seasonal variations)

Before the feature set for training the model only included the “hour” and “cluster” column. Training the model based on these two features resulted in a MSE of: 65119024 and R² score: 0.955. After adding three other features: weekday, weekend, month the MSE reduced to: 218908 and R² score: 0.988.

As this model is quite suitable for the dataset with a 98,8% variability in the predicted variable, the model doesn’t needs further tuning.

# 6. Integrating the Model into Workflow

## 6.1. Introduction

This model is set to improve the workflow of the logistics team, by forecasting the demand in public transportation. The deployment of this model into the logistics team's workflow is pivotal for real-time decision-making. To give hourly demand projections, the model should be connected with the current transportation management system. The integration could be achieved through a REST API interface, which allows the model to receive input and return predictions in real-time.

## 6.2. GUI Design Proposal

Logistics managers should find it easy to comprehend demand forecasts and make data-driven choices by using a graphical user interface that is straightforward and easy to use.

### 6.2.1. Features and functionality

- **Dashboard**
  a centralized dashboard showing the Key Performance Indicators with access to real-time demand forecasts and interactive demand heatmaps.
- **Interactive Map**
  Show demand hotspots that highlight clusters based on expected demand levels used to support the users in managing their locations remotely.
- **Alert system**
  Automated notifications from the alert system for unusual demand patterns, enabling pro-active improvements to logistics.
- **Custom Queries**
  Users should be able to input specific information for informed demand projections by using the before outlined API. (public events, weather conditions, etc)

### 6.2.2. User interaction and experience

- **User-Friendly Interface**
  A simple design that makes all the information easy to see and understand, guaranteeing that users can efficiently use the interface.
- **Mobile Compatibility**
  Since the majority of the logistics staff works remotely, they ought to have mobile access to the demand projections.
- **Customizable views**
  Users are able to personalize their display to highlight the metrics that are most important to them.

## 6.3. Continuous Integration Process for New Data

To assure that the model stays accurate and relevant, it is crucial to continually train it with new data. By implementing continuous integration for data ingestion, the model with increase its accuracy over time.
To keep the model's predictive accuracy high, regular reviews and updates should be planned.

# 7. Conclusion

## 7.1. Summary of Findings

This case study's main goal was to create a regression model that could forecast the average hourly demand for taxis in various city clusters. The study's conclusions are noteworthy in a number of ways.

With an R² score of 0.988, the regression model—which used the RandomForestRegressor algorithm—showed good predictive accuracy. This suggests that the model is very effective for forecasting, as it could account for approximately 98.8% of the variance in taxi demand.

While considering the other features like day of the week and month, incorporation in the model made the Mean Squared Error (MSE) to reduce considerably, putting weight on the importance of incorporating the temporal elements for accurate demand forecast.

This study identifies the relevant feature selection aspect of the model. When building the model at first, a lot of its primitive nature was dependent on simple attributes like hour and cluster giving good primary results. However, the model improved noticeably when more temporal characteristics such as day of the week and month were included.

This, therefore, would suggest that besides being a function of time and position, the demand for taxis is also a function of more general spatial-temporal patterning that could conceivably be determined by seasonal conditions, social activities and work practices.

An important step was that of first use K-means clustering to find significant clusters within the city at first. By so doing pre-processing, the model could take into account the fact that demand patterns can significantly differ between clusters and hence treating different parts of the city differently.

This broadens the application of the study while trying to understand complex urban dynamics within which segmentation is adopted in order to forecast taxi demand. It also helps in adding value towards traffic management, understanding socio-economic analysis and urban planning since through such kind of segmentation it becomes possible to carry out pointing out these hotspots and cold spots in demand. This approach informs more than the strategies for the infrastructure and public safety improvements, but also environmental impacts and emerging city trends hence becomes very instrumental in the understanding of it and dealing appropriately with challenges of any nature that come up when it comes to urban areas.

## 7.2. Recommendations for the Logistic Team

The logistic team should incorporate the model's insights into their operational strategies to ensure the best possible use of available resources. Important suggestions include giving high-demand clusters priority and allocating taxis dynamically based on hourly demand predictions. Fleet efficiency can also be improved by using data analytics for predictive maintenance. Improving customer satisfaction by cutting wait times in areas with high demand can also improve the quality of the services provided. Service delivery will be further optimized by regular model revaluations to adjust to changing urban dynamics and by incorporating real-time data for on-the-fly adjustments.

## 7.3. Potential Improvements and Future Work

In order to improve demand forecasts, future improvements should concentrate on incorporating more datasets, such as meteorological patterns, event schedules, and traffic data. Investigating cutting-edge machine learning methods, such as neural networks, may provide more accurate demand forecasting. Accuracy of the model can also be increased by working together with city planners to share data on urban development projects. In the long run, investigating real-time data integration for dynamic demand prediction may be a big step forward, in addition to regular model updates to keep up with changing transportation trends and cityscapes.

# 8. References

GitHub: “https://github.com/kmeans27/model_engineering__demand_forecasting”

# 9. Appendices

## 9.1. Detailed Model Evaluation Results

**Mean Squared Error**: 218908.2551800578
**R2 Score**: 0.9881488369835723
**Mean Absolute Error**: 223.8799710982659
**Root Mean Squared Error**: 467.87632466289404
**Explained Variance Score**: 0.9882713277606472