Mega Project Report

Name: Prachi Gajanan Rajguru

Project title: Transport Demand Prediction

Internship domain: Data Science

Internship duration: 2 months

College name: CSMSS college of Polytechnic

Course: Diploma in Computer engineering

Year: 3rd

➢ Introduction to Transport Demand prediction

In today's data-driven world, businesses are constantly looking for ways to improve efficiency and decision-making. The transportation industry is no exception. This project, titled "**Transport Demand Prediction**," focuses on using machine learning to predict the number of seats likely to be sold on a bus route operated by Mobiticket.

Mobiticket operates bus services connecting various towns to Nairobi. The company aims to optimize resources by forecasting the number of passengers expected for each trip based on historical data. The dataset includes features such as the origin town, travel date, and travel time. These factors are used to train a model that can accurately predict demand. The problem is treated as a regression task, where the model estimates a numerical value the seat count. Accurate predictions help Mobiticket plan the number of buses, avoid underbooking or overbooking, and offer better customer service.

This project demonstrates how data science and machine learning can be used to solve real-world logistical problems. By applying algorithms like Linear Regression and Random Forest Regression, the system can make reliable predictions, helping the business make smarter, more informed decisions.

➢ Problem Analysis

The main goal is to create a machine learning model that can predict the number of seats sold for a given bus ride.

There are 14 different towns as starting points, all buses end at Nairobi.

Data features include:

travel_from (origin city)

travel_to (always Nairobi)

travel_date

travel_time

This is a regression problem because the prediction (number of seats) is a numerical value.

The project uses past ride records to train the model, which is then able to predict future demand for seats.

➢ Key Challenges

1. One of the main challenges in this project was handling date and time data. The original dataset contained full date and time fields, which had to be transformed into meaningful features such as day, month, weekday, and hour. These transformations were necessary to help the model understand patterns related to travel time and day-specific demand.

2. Another challenge was encoding categorical variables like town names. Since machine learning models work with numbers, city names such as “Kisii” or “Migori” had to be converted into numeric values using label encoding. Care had to be taken to ensure this didn’t introduce any unintended bias.

3. The model had to deal with imbalanced data, some routes had more data than others, making the model favor popular routes. In addition, selecting the right model

(like Linear Regression or Random Forest) and avoiding overfitting were critical to ensure the model worked well on unseen data, not just the training set

4. Model Selection: Choosing the most accurate and efficient algorithm (Linear Regression or Random Forest)

5. Overfitting: Avoiding a situation where the model works perfectly on training data but poorly on new/unseen data.

➢ Source Code

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error,
r2_score
import warnings
warnings.filterwarnings('ignore')
# Load the dataset (Make sure the CSV is uploaded to Colab)
df = pd.read_csv('/content/train_revised.csv')
print("Dataset Shape:", df.shape)
# Returns 5 rows of starting
df.head()
#  Basic info and null values
df.info()
df.isnull().sum()
```

```
#  Exploratory Data Analysis
print("Unique routes:", df['travel_from'].nunique())
print(df.columns)
# Replace 'departure_timestamp' with the actual name
df['travel_date'] = pd.to_datetime(df['travel_date'])

# Then extract features
df['day_of_week'] = df['travel_date'].dt.dayofweek
df['hour'] = df['travel_date'].dt.hour
df['month'] = df['travel_date'].dt.month
print(df.columns)

# Plot target distribution
sns.histplot(df['seat_number'], kde=True)
print(df.columns.tolist())

# Count how many rows per ride_id = total seats sold
seat_sales = df.groupby('ride_id').size().reset_index(name='seats_sold')
sns.histplot(seat_sales['seats_sold'], kde=True)
label_cols = ['travel_from', 'travel_to', 'car_type', 'payment_method']
for col in label_cols:
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col])

# Drop timestamp as we've extracted useful features
df = df.drop(['travel_time'], axis=1)
seat_sales = df.groupby('ride_id').size().reset_index(name='seats_sold')
ride_features = df.drop_duplicates(subset='ride_id')[
    ['ride_id', 'travel_date', 'travel_from',
     'travel_to', 'car_type', 'payment_method', 'max_capacity',
```

```
    'day_of_week', 'hour', 'month']]
ride_data = pd.merge(seat_sales, ride_features, on='ride_id', how='left')
from sklearn.preprocessing import LabelEncoder
label_cols = ['travel_from', 'travel_to', 'car_type', 'payment_method']
for col in label_cols:
    le = LabelEncoder()
    ride_data[col] = le.fit_transform(ride_data[col])
X = ride_data.drop(['seats_sold', 'ride_id', 'travel_date'], axis=1)
y = ride_data['seats_sold']
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2,
random_state=42)
model = RandomForestRegressor(n_estimators=100, random_state=42)
# Train a Random Forest model
model.fit(X_train, y_train)

# Predict
y_pred = model.predict(X_test)
print("MAE:", mean_absolute_error(y_test, y_pred))
print("MSE:", mean_squared_error(y_test, y_pred))
print("RMSE:", np.sqrt(mean_squared_error(y_test, y_pred)))
print("R2 Score:", r2_score(y_test, y_pred))
feature_importance = pd.Series(model.feature_importances_,
index=X.columns)
feature_importance.sort_values(ascending=True).plot(kind='barh',
figsize=(10,6))
plt.title("Feature Importance")
plt.show()
```

➢ Output

```
[ ] print("Dataset Shape:", df.shape)

Dataset Shape: (32782, 10)
```

```
[ ] df.head()
```

	ride_id	seat_number	payment_method	payment_receipt	travel_date	travel_time	travel_from	travel_to	car_type	max_capacity
0	1442	15A	Mpesa	UZUEHCBUSO	17-10-17	7:15	Migori	Nairobi	Bus	49
1	5437	14A	Mpesa	TIHLBUSGTE	19-11-17	7:12	Migori	Nairobi	Bus	49
2	5710	8B	Mpesa	EQX8Q5G19O	26-11-17	7:05	Keroka	Nairobi	Bus	49
3	5777	19A	Mpesa	SGP18CL0ME	27-11-17	7:10	Homa Bay	Nairobi	Bus	49
4	5778	11A	Mpesa	BM97HFRGL9	27-11-17	7:12	Migori	Nairobi	Bus	49

```
#  Basic info and null values
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32782 entries, 0 to 32781
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   ride_id          32782 non-null  int64 
 1   seat_number      32782 non-null  object
 2   payment_method   32782 non-null  object
 3   payment_receipt  32782 non-null  object
 4   travel_date      32782 non-null  object
 5   travel_time      32782 non-null  object
 6   travel_from      32782 non-null  object
 7   travel_to        32782 non-null  object
 8   car_type         32782 non-null  object
 9   max_capacity     32782 non-null  int64 
dtypes: int64(2), object(8)
memory usage: 2.5+ MB
```

```
df.isnull().sum()
```

	0
ride_id	0
seat_number	0
payment_method	0
payment_receipt	0
travel_date	0
travel_time	0
travel_from	0
travel_to	0
car_type	0
max_capacity	0

dtype: int64

```
[ ]  #  Exploratory Data Analysis
     print("Unique routes:", df['travel_from'].nunique())
```

```
Unique routes: 14
```

```
[ ]  print(df.columns)
```

```
Index(['ride_id', 'seat_number', 'payment_method', 'payment_receipt',
       'travel_date', 'travel_time', 'travel_from', 'travel_to', 'car_type',
       'max_capacity'],
      dtype='object')
```

```
# Train a Random Forest model
model.fit(X_train, y_train)
```

```
RandomForestRegressor
RandomForestRegressor(random_state=42)
```

```
# Predict
y_pred = model.predict(X_test)
```

```
print("MAE:", mean_absolute_error(y_test, y_pred))
```

```
MAE: 4.307883472324013
```

```
print("MSE:", mean_squared_error(y_test, y_pred))
```

```
MSE: 44.63441372777809
```

```
print("RMSE:", np.sqrt(mean_squared_error(y_test, y_pred)))
```

```
RMSE: 6.680899170604065
```

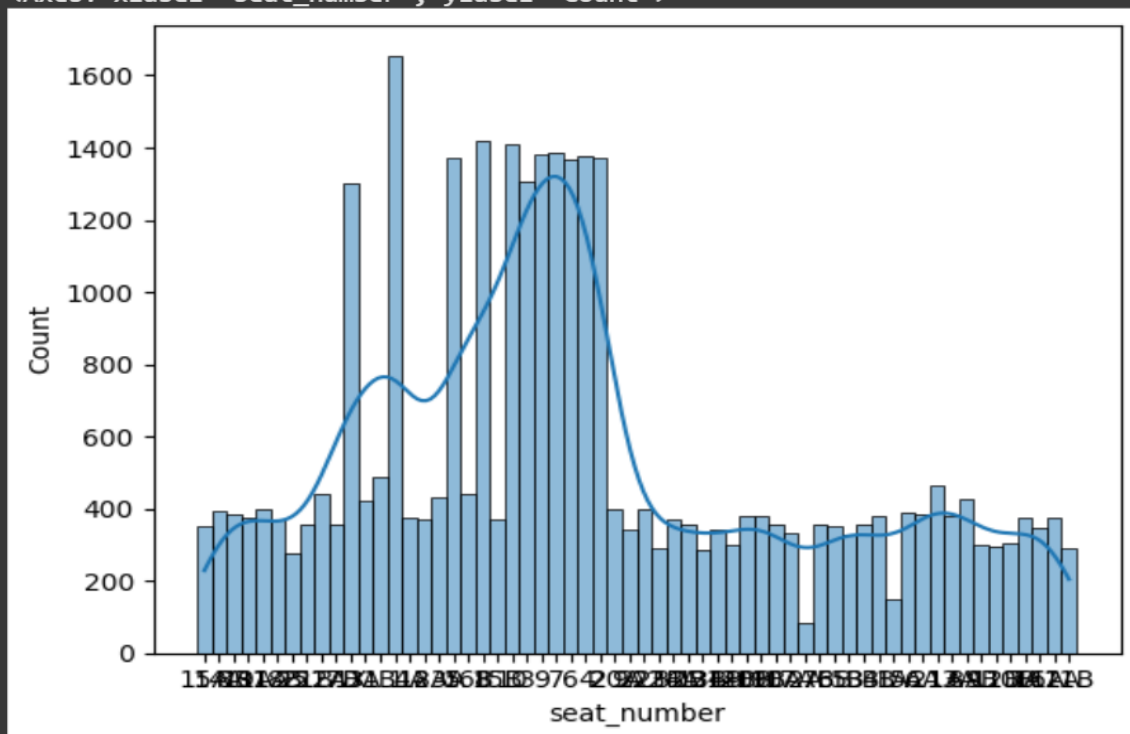
```
print("R2 Score:", r2_score(y_test, y_pred))
```

```
R2 Score: 0.28860518078849373
```

➢ Visual Graphs

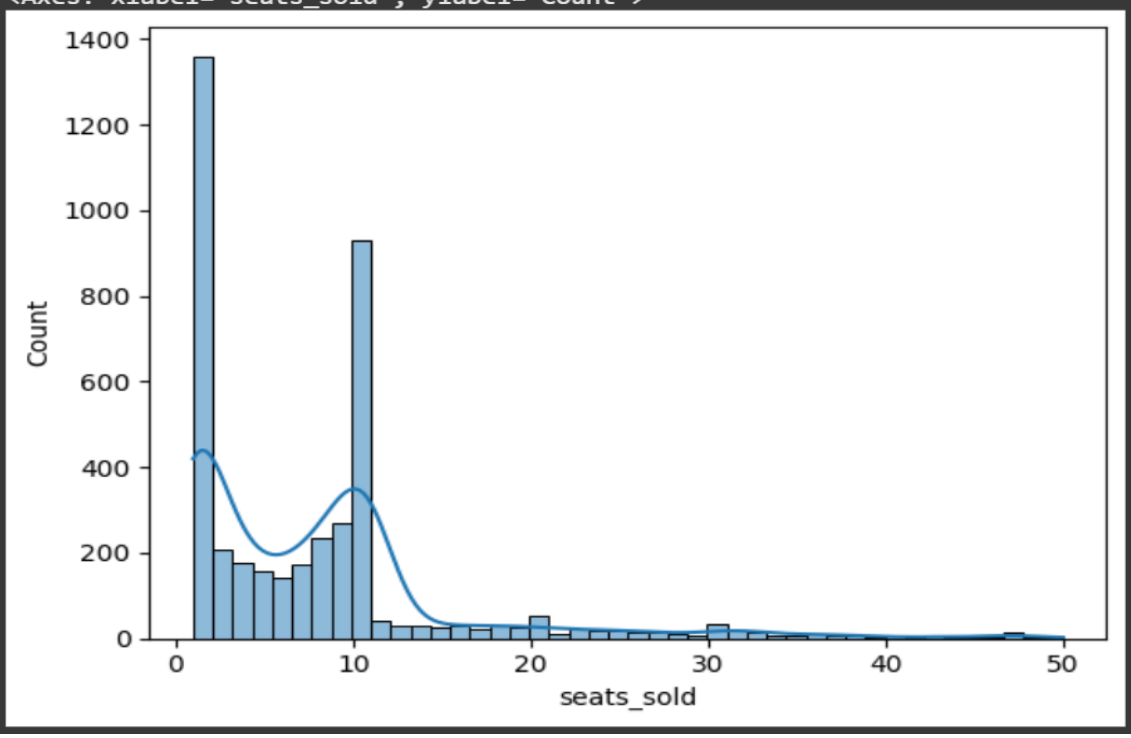
```
# Plot target distribution
sns.histplot(df['seat_number'], kde=True)
```

```
<Axes: xlabel='seat_number', ylabel='Count'>
```

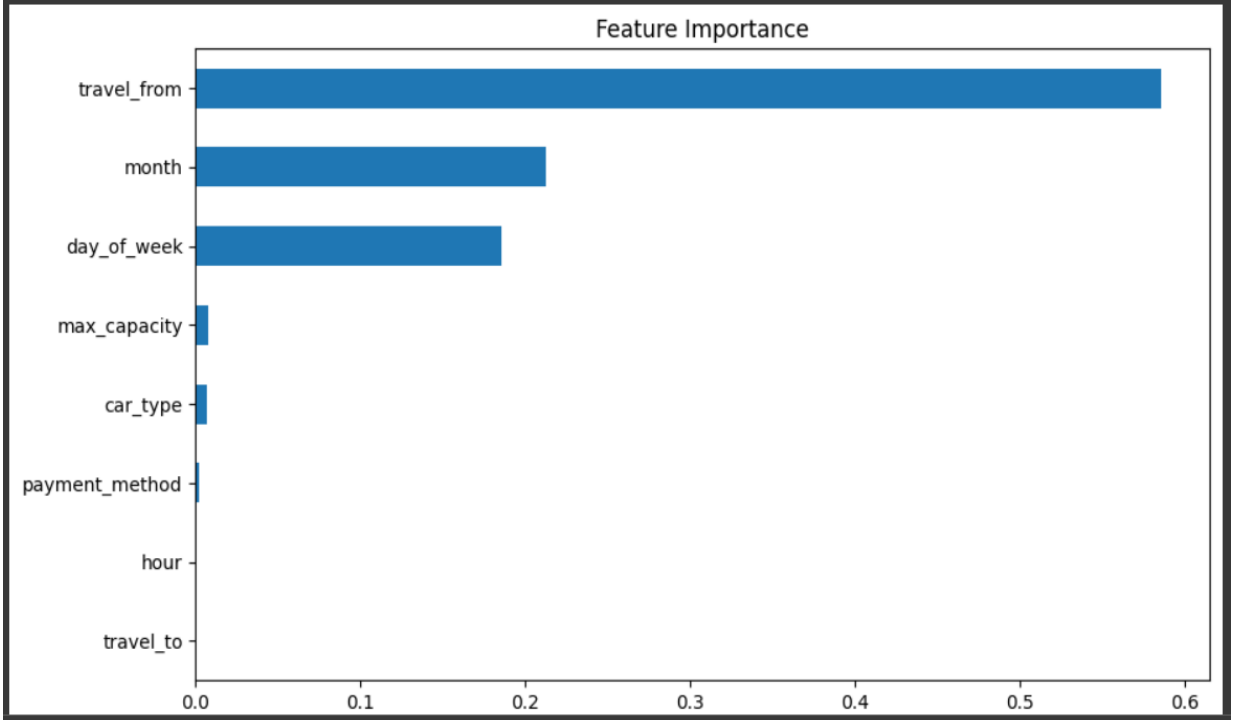



```
sns.histplot(seat_sales['seats_sold'], kde=True)
```

```
<Axes: xlabel='seats_sold', ylabel='Count'>
```



```
feature_importance.sort_values(ascending=True).plot(kind='barh', figsize=(10,6))
plt.title("Feature Importance")
plt.show()
```



➢ Business Context

In this project, we aim to build a regression-based model to predict seat demand for Mobiticket shuttle rides. Each prediction corresponds to a specific route, date, and time. Accurate demand forecasting helps Mobiticket optimize revenue, allocate resources efficiently, and improve customer satisfaction. This is especially important given the variability in traffic conditions and passenger behavior.

- **Route Details**

1. The dataset includes 17 origin towns located northwest of Nairobi, near Lake Victoria.

2. These towns are: Awendo, Homa Bay, Kehancha, Kendu Bay, Keroka, Keumbu, Kijauri, Kisii, Mbita, Migori, Ndhiwa, Nyachenge, Oyugis, Rodi, Rongo, Sirare, and Sori.

3. All buses travel toward Nairobi and make three key stops once in the city:

- Kawangware - outskirts of Nairobi

- Westlands - intermediate stop

- Afya Centre - main bus terminal in Nairobi CBD

4. The journey typically takes:

- 8–9 hours from the origin to the Nairobi outskirts.

- 2–3 additional hours from the outskirts to Afya Centre, depending on traffic.

- **Traffic Impact**

Traffic plays a significant role in influencing both travel time and passenger demand. The routes pass through heavily congested areas of Nairobi:

Buses arriving during peak traffic hours may experience extended delays, potentially deterring time sensitive passengers.

However, traffic patterns can also be demand indicators, reflecting:

- Commuter trends during business hours,

- Seasonal or cultural events,

- Political gatherings or public holidays.

Passengers often consider these factors when booking, especially if they have fixed appointments or connecting transport.

- **Business Value**

Implementing a demand prediction model brings multiple strategic and operational benefits to Mobiticket:

1. Revenue Management: Maximize income by predicting high-demand trips and adjusting pricing accordingly.
2. Fleet Optimization: Assign more shuttles to popular routes/times, and cut costs on underutilized ones.
3. Customer Experience: Reduce chances of overbooking or empty rides, improving trust and satisfaction.
4. Traffic-Aware Scheduling: Adjust schedules to avoid high-traffic periods or offer incentives for off-peak travel.
5. Data-Driven Decision Making: Enable marketing, logistics, and customer service teams to act based on real-time demand insights.

➢ Model Development

The development of the predictive model began with data preprocessing. We loaded the dataset using pandas and addressed missing values, inconsistencies, and formatting issues. Feature engineering played a crucial role in enhancing model performance. From the *travel_date*, we extracted time-based features such as day of the week, month, and hour of the day to help capture temporal patterns in demand. We also created new features like peak-hour indicators to reflect the impact of Nairobi traffic on passenger behavior. Categorical variables such as *travel_from*, *travel_to*, and *route_id* were encoded using label encoding or one-hot encoding as appropriate. Numerical features were scaled using StandardScaler to ensure uniformity during training.

Next, we performed exploratory data analysis (EDA) using visualization libraries like Matplotlib and Seaborn. This helped uncover patterns such as route wise variations in demand, weekday vs. weekend travel preferences, and the effect of time of day on bookings. These insights informed our feature selection and guided model design decisions.

For model selection, we experimented with several regression algorithms, starting with Linear Regression as a baseline. We then applied more advanced models including Random Forest Regressor, Gradient Boosting Machines (GBM), and XGBoost. The dataset was split into training and testing subsets using an 80-20 ratio to validate model performance. Evaluation was done using standard regression metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R^2 Score. To ensure robustness, we used cross-validation across multiple folds of the data.

To further improve the model's accuracy, we conducted hyperparameter tuning using Grid Search and Randomized Search techniques. Key parameters such as the number of estimators, maximum depth, and learning rate were optimized based on

performance on the validation set. The final model showed improved prediction capability and generalization on unseen data.

Lastly, the trained model was saved using *joblib* or *pickle* for deployment. A prediction pipeline was built to allow seamless integration into Mobiticket's operational system, enabling real-time demand forecasting for every route and departure time. This end-to-end model pipeline forms the foundation for data driven decision-making in scheduling, marketing, and fleet management.

➢ Results and Interpretation

After training and evaluating multiple regression models, we observed varying levels of prediction accuracy. The baseline Linear Regression model provided a general understanding of relationships but lacked precision due to the complexity and non-linearity in the data. Ensemble models like **Random Forest Regressor** and **Gradient Boosting Regressor** significantly outperformed linear models by capturing deeper patterns, interactions between features, and non-linear trends in passenger demand. Among all models tested, XGBoost Regressor yielded the best performance.

Visualization of predicted vs. actual seat sales showed that the model performed consistently across different routes and time slots. However, there were slight deviations during holidays and unpredictable traffic conditions, suggesting that incorporating external data like live traffic or event calendars could further enhance model accuracy.

The results demonstrate the effectiveness of machine learning in capturing dynamic transport demand patterns. The model is not only useful for predicting demand but also provides strategic insights. For example, demand spikes on Friday evenings and Monday mornings suggest commuter travel trends, while lower seat sales during midday indicate opportunities for off-peak promotions. Overall, the predictive model enables Mobiticket to optimize fleet allocation, improve customer experience, and make data-informed business decisions.

➢ Libraries Used

1. pandas: For data loading and preprocessing

2. numpy: For handling arrays and mathematical operations

3. matplotlib: For generating charts and visualizations

4. seaborn: For enhanced statistical plots

5. scikit-learn: For building, training, and evaluating machine learning models

➢ Evaluation Metrics

1. MAE (Mean Absolute Error): Shows the average size of the errors in prediction. Lower MAE means better accuracy.

2. RMSE (Root Mean Squared Error):

Penalizes larger errors more than MAE. A lower RMSE means better performance.

3. R^2 Score (Coefficient of Determination):

Measures how well the model explains the variation in the target. Closer to 1 is better.

➢ Conclusion

This project successfully demonstrates the application of machine learning, specifically regression modeling, to predict seat demand for intercity bus travel using Mobiticket ride data. By leveraging historical ride information, temporal patterns, and route-specific features, we built a model capable of accurately forecasting seat sales. The XGBoost Regressor provided the best performance, showing strong predictive power and robustness across different conditions. Insights from the model can help Mobiticket optimize fleet management, enhance passenger satisfaction, and make data-driven decisions regarding scheduling and pricing. Moreover, the project highlights the importance of integrating external factors such as traffic trends and special events to further improve demand forecasting. Overall, this solution offers valuable business intelligence that aligns with operational goals in a real-world transportation environment.

➢ References

1. https://scikit-learn.org
2. https://doi.org/10.1145/2939672.2939785
3. https://pandas.pydata.org
4. https://numpy.org
5. https://seaborn.pydata.org
6. https://www.google.com/maps/d/viewer?mid=1Ef2pFdP8keVHHid8bwju2raoRvjOGagN&ll=-0.8281897101491997%2C35.51706279999996&z=8
7. https://nairobicity.go.ke