Capabl.

Summer Internship program

Taxi Trip Duration prediction

TABLE OF CONTENTS

- Objective
- Data Gathering
- Data Pre-processing
    - Data Cleaning
    - Missing value analysis
    - Exploratory Data analysis
    - Feature selection
- Train Model
- Test model
- Conclusion

OBJECTIVE

To build a Machine Learning model that predicts the total ride duration of taxi trips in New York City with minimum error.

To Business Goal : Improve the efficiency of taxi systems to be able to predict how long a driver will have his taxi occupied

Data Gathering

Dataset: NYC_taxi

The NYC taxi trip prediction dataset is a collection of data that captures various attributes and information about taxi trips in New York City. It provides insights into the duration and distance of the trips

The dataset includes

id	a unique identifier for each trip
vendor_id	a code indicating the provider associated with the trip record
pickup_datetime	date and time when the meter was engaged
dropoff_datetime	date and time when the meter was disengaged
passenger_count	the number of passengers in the vehicle (driver entered value)
pickup_longitude	the longitude where the meter was engaged
pickup_latitude	the latitude where the meter was engaged
dropoff_longitude	the longitude where the meter was disengaged
dropoff_latitude	the latitude where the meter was disengaged
store_and_fwd_flag	This flag indicates whether the trip record was held in vehicle memory before sending to the vendor because the vehicle did not have a connection to the server Y=store and forward; N=not a store and forward trip
trip_duration	duration of the trip in seconds

PROJECT TIMELINE

Missing value analysis

Feature creation

Data Cleaning

Exploratory data analysis

Training and testing

Data preprocessing

Data Cleaning

id, vendor_id, pickup time and drop off time doesn't contribute to our target variable

They could be dropped from the training set

[3]:
```
(1458644, 11)
```

```
#Khushi 21BCE1282
df=df.drop(['id','vendor_id','pickup_datetime','dropoff_datetime'],axis=1)
df.dtypes
```

[5]:
```
passenger_count         int64
pickup_longitude      float64
pickup_latitude       float64
dropoff_longitude     float64
dropoff_latitude      float64
store_and_fwd_flag     object
trip_duration           int64
dtype: object
```

+ Code + Markdown

[4]:
```
#Khushi 21BCE1282
```

Missing vales

- Missing values can be identified by

  df.isnull().sum()

```
dropoff_latitude      float64
store_and_fwd_flag     object
trip_duration           int64
dtype: object
```

```
#Khushi 21BCE1282
df.isnull().sum()
```

```
[6]: passenger_count      0
pickup_longitude      0
pickup_latitude       0
dropoff_longitude     0
dropoff_latitude      0
store_and_fwd_flag    0
trip_duration         0
dtype: int64
```

+ Code + Markdown

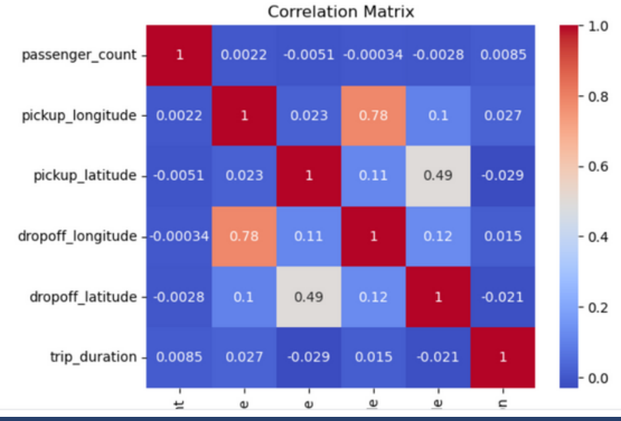
- No null elements found in any column

Exploratory data analysis

A correlation matrix is a table that shows the correlation coefficients between multiple variables.

```
#Khushi 21BCE1282
import seaborn as sns
# Correlation matrix
corr_matrix =df.corr()
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```

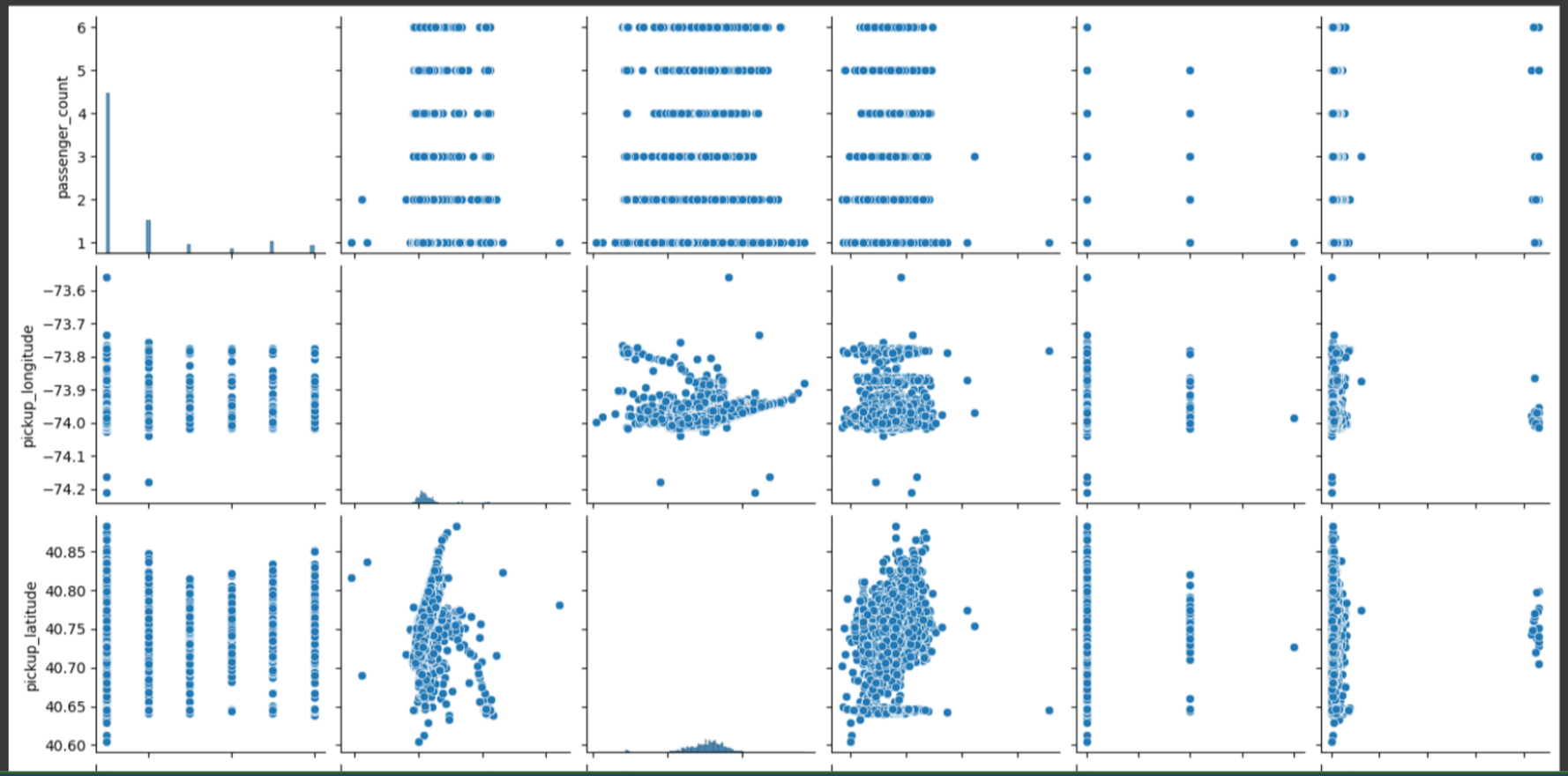
```
/tmp/ipykernel_32/3533226130.py:4: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid co
mns or specify the value of numeric_only to silence this warning.
  corr_matrix =df.corr()
```



Pairplot

A pairplot is a graphical representation that shows the relationships between pairs of variables in a dataset.

```
#Khushi 21BCE1282
sns.pairplot(df)
plt.title('Pair Plot')
plt.show()
```



Feature Selection

Label Encoder

Label Encoder is a preprocessing technique used in machine learning to convert categorical variables into numerical form.

- In the given dataset, the column 'store_and_fwd_flag' contains two variables i.e'N','Y'
- It could be change into numerical values using Label encoder

```
#Khushi 21BCE1282
#Label encoder
from sklearn import preprocessing
label_encoder= preprocessing.LabelEncoder()
# Encode labels in column 'store_and_fwd_flag'.
df['store_and_fwd_flag']= label_encoder.fit_transform(df['store_and_fwd_flag'])
counts = df['store_and_fwd_flag'].value_counts()
print(counts)
```

```
0    1450599
1       8045
Name: store_and_fwd_flag, dtype: int64
```

+ Code + Markdown

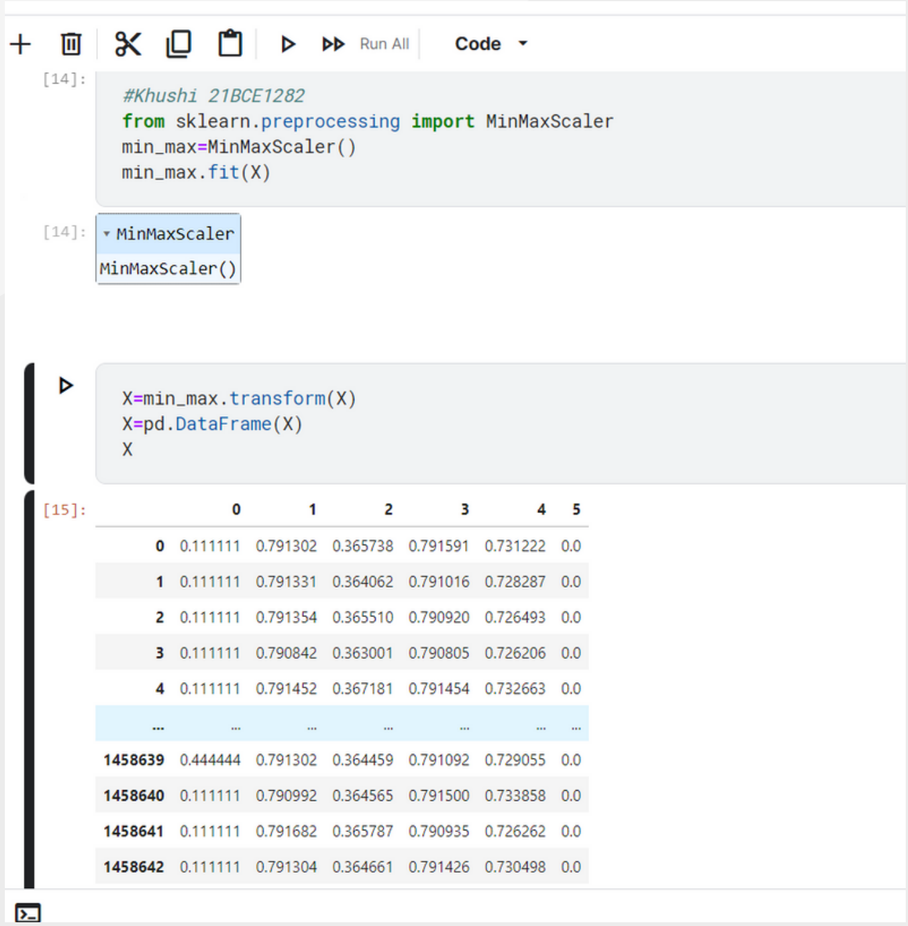
```
[10]:
#Khushi 21BCE1282
import seaborn as sns
```

Seperating Target variable and Features

Min Max Scaling

- It scales the values of a feature to a fixed range, typically between 0 and 1,

The values in every column differ too much in range
Thus Min- Max scaling is used to range them between 0-1
This reduces non-linearity and model fits well to the dataset



```
[14]: #Khushi 21BCE1282
      from sklearn.preprocessing import MinMaxScaler
      min_max=MinMaxScaler()
      min_max.fit(X)

[14]: MinMaxScaler()
      MinMaxScaler()

      X=min_max.transform(X)
      X=pd.DataFrame(X)
      X
```

[15]:

	0	1	2	3	4	5
0	0.111111	0.791302	0.365738	0.791591	0.731222	0.0
1	0.111111	0.791331	0.364062	0.791016	0.728287	0.0
2	0.111111	0.791354	0.365510	0.790920	0.726493	0.0
3	0.111111	0.790842	0.363001	0.790805	0.726206	0.0
4	0.111111	0.791452	0.367181	0.791454	0.732663	0.0
...	...	...	...	...	...	...
1458639	0.444444	0.791302	0.364459	0.791092	0.729055	0.0
1458640	0.111111	0.790992	0.364565	0.791500	0.733858	0.0
1458641	0.111111	0.791682	0.365787	0.790935	0.726262	0.0
1458642	0.111111	0.791304	0.364661	0.791426	0.730498	0.0

Features are converted to data frame after fitting with min max scalar for training them model

Target variable was first converted to dataframe since data in 2D format can only go under Min-Max scaling

Target variable are converted to data frame after fitting with min max scalar for training them model

```
y=pd.DataFrame(y)
min_max.fit(y)
y=min_max.transform(y)
y=pd.DataFrame(y)
y
```

[16]:

	0
0	0.000129
1	0.000188
2	0.000602
3	0.000121
4	0.000123
...	...
1458639	0.000220
1458640	0.000185
1458641	0.000216
1458642	0.000105
1458643	0.000056

1458644 rows × 1 columns

Training the model

- **Linear Regression**

Linear regression is a statistical modeling technique used to analyze the relationship between a dependent variable and one or more independent variables

[18]:
```
#Khushi 21BCE1282
#splitting into training and testing data for creating model
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=3)
```

[19]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(X_train,y_train)
```

[19]:
```
LinearRegression
LinearRegression()
```

+ Code + Markdown

```
y_pred=lr.predict(X_test)
```

Predicted values using the Linear Regression based ML model

```
#Khushi 21BCE1282
#predicted values
y_pred
```

```
[21]: array([[0.00028499],
             [0.00023014],
             [0.00025797],
             ...,
             [0.00027275],
             [0.00024765],
             [0.00023225]])
```

+ Code + Markdown

Testing of the model

The primary goal of testing an ML model is to assess its performance and evaluate how well it generalizes to unseen data.

R2 Score

It measures the proportion of the variance

Mean squared error

It measures the average squared difference between the predicted values and the actual values

Mean absolute error

It measures the average absolute difference between the predicted values and the actual values

```
        [0.00024765],
        [0.00023225]])
```

+ Code + Markdown

```python
#Khushi 21BCE1282
from sklearn.metrics import r2_score,mean_squared_error,mean_absolute_error
score = r2_score(y_test,y_pred)
print("The accuracy of our model is {}%".format(score, 2))
mae = mean_absolute_error(y_test,y_pred)
print("Mean absolue error:",mae)
mse = mean_squared_error(y_test,y_pred)
print("Mean squareed error: ",mse)
```

```
The accuracy of our model is 0.0015590513645580284%
Mean absolue error: 0.0001707185148601693
Mean squareed error:  1.8471075927914924e-06
```

+ Code + Markdown

Error to be minimum.
Thus our model fits the dataset accurately!!

Conclusion

- **Thus Regression based ML model is trained with the given NYC taxi data set with minimum error.**

THANK YOU

Created by
Khushi Singh
21BCE1282

Vellore Institute of technology,Chennai

7396207675

khushi.singh2021n@vitstudent.ac.in