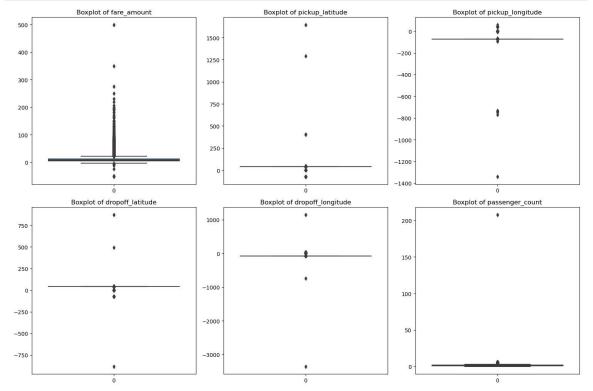
```
In [1]: import pandas as pd
         import numpy as np
         import matplotlib.pyplot as plt
         import seaborn as sns
         from sklearn.model_selection import train_test_split
         from sklearn.linear_model import LinearRegression
         from sklearn.ensemble import RandomForestRegressor
         from sklearn.metrics import mean squared error, r2 score
In [2]: | df = pd.read csv('uber.csv')
         df.head()
In [3]:
Out[3]:
             Unnamed:
                                    key fare_amount pickup_datetime pickup_longitude pickup_lat
                    0
                              2015-05-07
                                                          2015-05-07
             24238194
                                                 7.5
                                                                          -73.999817
                                                                                         40.73
                         19:52:06.0000003
                                                        19:52:06 UTC
                              2009-07-17
                                                          2009-07-17
             27835199
                                                 7.7
                                                                          -73.994355
                                                                                         40.72
                         20:04:56,0000002
                                                        20:04:56 UTC
                              2009-08-24
                                                          2009-08-24
             44984355
                                                12.9
                                                                          -74.005043
                                                                                         40.74
                        21:45:00.00000061
                                                        21:45:00 UTC
                              2009-06-26
                                                          2009-06-26
             25894730
                                                 5.3
                                                                          -73.976124
                                                                                         40.79
                         08:22:21.0000001
                                                        08:22:21 UTC
                              2014-08-28
                                                          2014-08-28
             17610152
                                                16.0
                                                                          -73.925023
                                                                                         40.74
                       17:47:00.000000188
                                                        17:47:00 UTC
In [4]: |df['pickup_datetime'] = pd.to_datetime(df['pickup_datetime'])
In [5]: |df['hour']=df['pickup_datetime'].dt.hour
         df['day']=df['pickup_datetime'].dt.day
         df['day_of_week']=df['pickup_datetime'].dt.dayofweek
         df['month']=df['pickup_datetime'].dt.month
In [6]: |df=df.dropna()
In [7]: | df=df.drop(['pickup datetime'], axis=1)
```

```
In [8]: numeric_features=['fare_amount','pickup_latitude','pickup_longitude','dropo
    plt.figure(figsize=(15,10))
    for i,feature in enumerate(numeric_features):
        plt.subplot(2,3,i+1)
        sns.boxplot(df[feature])
        plt.title(f'Boxplot of {feature}')
    plt.tight_layout()
    plt.show()
```



```
In [9]: df = df[(df['fare_amount']>0) & (df['fare_amount']<100)]
df = df[(df['passenger_count']>0) & (df['passenger_count']<=6)]</pre>
```

```
In [10]:
          numeric_df = df.select_dtypes(include=['number'])
           corr_matrix = numeric_df.corr()
           plt.figure(figsize=(10,8))
           sns.heatmap(corr matrix, annot=True,cmap='coolwarm')
           plt.title('Correlation Matrix')
           plt.show()
                                               Correlation Matrix
                                                                                          1.00
               Unnamed: 0
                              4.9e-050.000180.000290.000170.000380.0017 3.7e-060.00052-0.0045 0.0013
               fare_amount -4.9e-05
                                   0.0072 -0.0059 0.0063 -0.0083 0.012 -0.021 0.0017 0.0056 0.024
                                                                                          0.75
            pickup longitude -0.00018 0.0072
                                                    -0.85 -0.00018 0.0028 0.0055 0.00053 -0.0047
                                                                                          - 0.50
             pickup_latitude -0.00029-0.0059 -0.82
                                                         -0.0019 -0.0041 -0.0085 -0.0022 0.0047
                                                                                          - 0.25
                                                    0.92 0.00015 0.0039 0.0052-0.00025-0.0035
            dropoff_longitude -0.00017 0.0063
             dropoff_latitude -0.00038-0.0083 -0.85
                                               -0.92
                                                        0.00081-0.003 -0.008 -0.0021 0.0037
                                                                                          - 0.00
            0.015 0.0022 0.036 0.0086
                                                                                           -0.25
                    hour -3.7e-06 -0.021 0.0028 -0.0041 0.0039 -0.003 0.015
                                                                   0.0049 -0.087 -0.004
                     day -0.00052 0.0017 0.0055 -0.0085 0.0052 -0.008 0.0022 0.0049
                                                                         0.0057 -0.018
                                                                                           -0.50
In [11]: X = df[['pickup_latitude','pickup_longitude','dropoff_latitude','dropoff_lo
           y = df['fare_amount']
In [12]: |X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_stall)
In [17]: | lr_model = LinearRegression()
           lr_model.fit(X_train,y_train)
           y_pred_lr = lr_model.predict(X_test)
In [18]:
          rf_model = RandomForestRegressor(n_estimators=100,random_state=42)
           rf_model.fit(X_train,y_train)
           y pred rf = rf model.predict(X test)
In [19]: | def evaluate_model(y_test,y_pred,model_name):
               r2 = r2_score(y_test,y_pred)
               rmse = np.sqrt(mean_squared_error(y_test,y_pred))
               print(f'{model_name}-R2Score:{r2:2f},RMSE:{rmse:.2f}')
           evaluate_model(y_test,y_pred_lr,'Linear Regression')
           evaluate_model(y_test,y_pred_rf,'Random Forest Regression')
           Linear Regression-R2Score:0.001059,RMSE:9.45
```

Random Forest Regression-R2Score:0.803208,RMSE:4.19

In [20]: print("Linear Regression vs Random Forest Regression:")
 print("Linear Regression R2 Score: ",r2_score(y_test,y_pred_lr))
 print("Linear Regression RMSE:",np.sqrt(mean_squared_error(y_test,y_pred_lr))
 print("Random Forest Regression R2 Score: ",r2_score(y_test,y_pred_rf))
 print("Random Forest Regression RMSE: ",np.sqrt(mean_squared_error(y_test,y_pred_rf))

Linear Regression vs Random Forest Regression: Linear Regression R2 Score: 0.0010594227072876494

Linear Regression RMSE: 9.45007180929204

Random Forest Regression R2 Score: 0.8032075026481964
Random Forest Regression RMSE: 4.194397152263582