Assignment No.2

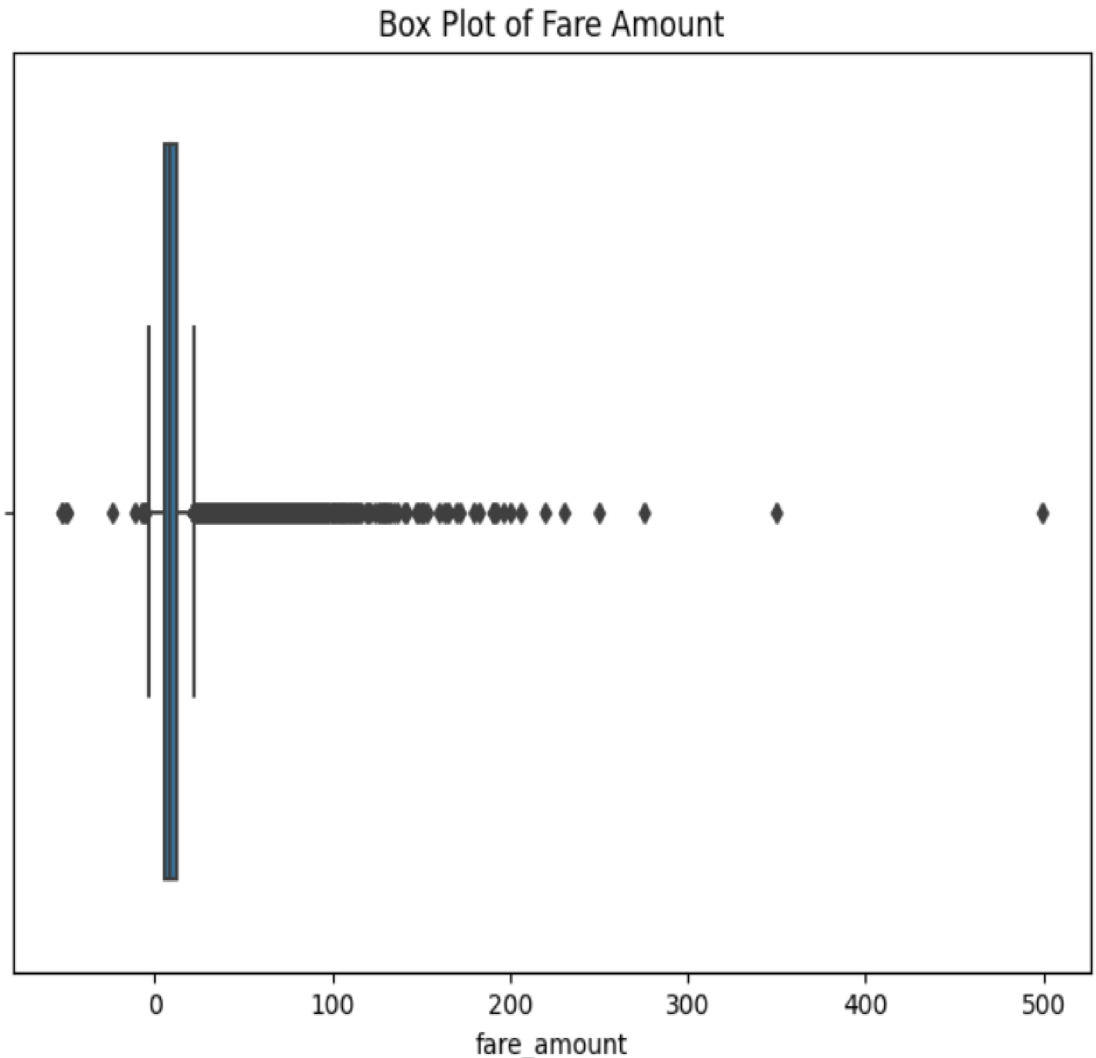
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression, Ridge, Lasso
from sklearn.metrics import r2_score, mean_squared_error
from sklearn.impute import SimpleImputer  # Import SimpleImputer
import seaborn as sns
import matplotlib.pyplot as plt

data = pd.read_csv('uber.csv')

data['pickup_datetime'] = pd.to_datetime(data['pickup_datetime'])

data['hour'] = data['pickup_datetime'].dt.hour
data['day_of_week'] = data['pickup_datetime'].dt.dayofweek

data = data.drop(columns=['Unnamed: 0', 'key', 'pickup_datetime'])

imputer = SimpleImputer(strategy='mean')
data_imputed = pd.DataFrame(imputer.fit_transform(data), columns=data.columns)

X = data_imputed.drop(columns=['fare_amount'])
y = data_imputed['fare_amount']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

```
data
```

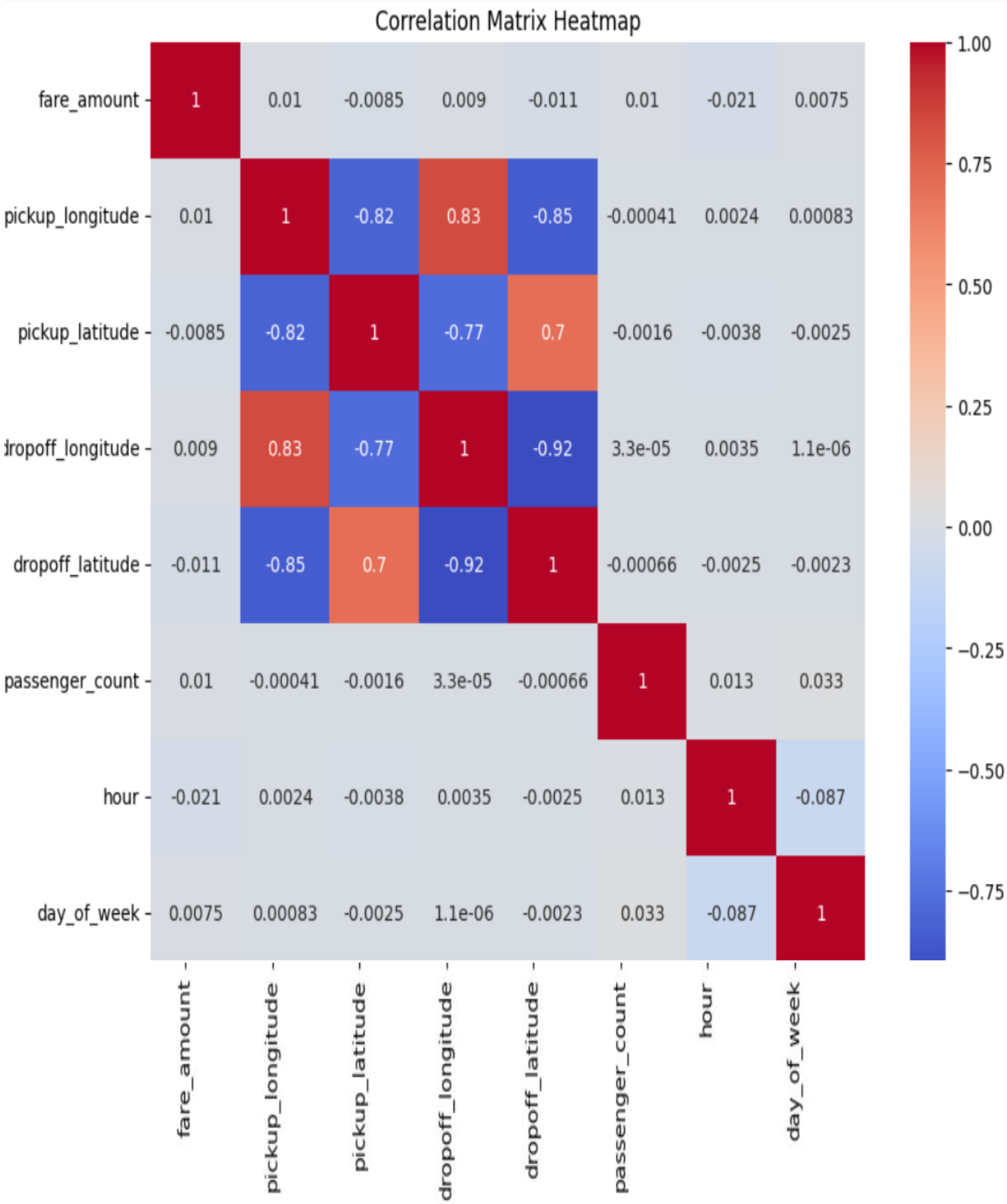
	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count	hour	day_of_week
0	7.5	-73.999817	40.738354	-73.999512	40.723217	1	19	3
1	7.7	-73.994355	40.728225	-73.994710	40.750325	1	20	4
2	12.9	-74.005043	40.740770	-73.962565	40.772647	1	21	0
3	5.3	-73.976124	40.790844	-73.965316	40.803349	3	8	4
4	16.0	-73.925023	40.744085	-73.973082	40.761247	5	17	3
...	...	...	...	...	...	...	...	...
199995	3.0	-73.987042	40.739367	-73.986525	40.740297	1	10	6
199996	7.5	-73.984722	40.736837	-74.006672	40.739620	1	1	4
199997	30.9	-73.986017	40.756487	-73.858957	40.692588	2	0	0
199998	14.5	-73.997124	40.725452	-73.983215	40.695415	1	14	2
199999	14.1	-73.984395	40.720077	-73.985508	40.768793	1	4	5

200000 rows × 8 columns

```
plt.figure(figsize=(8, 6))
sns.boxplot(data=data, x='fare_amount')
plt.title('Box Plot of Fare Amount')
plt.show()
```




```
correlation_matrix = data.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap="coolwarm")
plt.title('Correlation Matrix Heatmap')
plt.show()
```



Correlation Matrix Heatmap
fare_amount
pickup_longitude
pickup_latitude
dropoff_longitude
dropoff_latitude
passenger_count
hour
day_of_week
1 0.01 -0.0085 0.009 -0.011 0.01 -0.021 0.0075
0.01 1 -0.82 0.83 -0.85 -0.00041 0.0024 0.00083
-0.0085 -0.82 1 -0.77 0.7 -0.0016 -0.0038 -0.0025
0.009 0.83 -0.77 1 -0.92 3.3e-05 0.0035 1.1e-06
-0.011 -0.85 0.7 -0.92 1 -0.00066 -0.0025 -0.0023
0.01 -0.00041 -0.0016 3.3e-05 -0.00066 1 0.013 0.033
-0.021 0.0024 -0.0038 0.0035 -0.0025 0.013 1 -0.087
0.0075 0.00083 -0.0025 1.1e-06 -0.0023 0.033 -0.087 1
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75

```python
# Linear Regression
lr = LinearRegression()
lr.fit(X_train, y_train)
y_pred_lr = lr.predict(X_test)
r2_lr = r2_score(y_test, y_pred_lr)
rmse_lr = np.sqrt(mean_squared_error(y_test, y_pred_lr))

# Ridge Regression
ridge = Ridge(alpha=1.0)  # You can adjust the alpha parameter
ridge.fit(X_train, y_train)
y_pred_ridge = ridge.predict(X_test)
r2_ridge = r2_score(y_test, y_pred_ridge)
rmse_ridge = np.sqrt(mean_squared_error(y_test, y_pred_ridge))

# Lasso Regression
lasso = Lasso(alpha=1.0)  # You can adjust the alpha parameter
lasso.fit(X_train, y_train)
y_pred_lasso = lasso.predict(X_test)
r2_lasso = r2_score(y_test, y_pred_lasso)
rmse_lasso = np.sqrt(mean_squared_error(y_test, y_pred_lasso))

# Print results
print("Linear Regression - R2:", r2_lr, "RMSE:", rmse_lr)
print("Ridge Regression - R2:", r2_ridge, "RMSE:", rmse_ridge)
print("Lasso Regression - R2:", r2_lasso, "RMSE:", rmse_lasso)
```

```
Linear Regression - R2: 0.0007463283549304922 RMSE: 10.307359776695948
Ridge Regression - R2: 0.0007463398803480015 RMSE: 10.307359717253272
Lasso Regression - R2: -1.683808098018602e-05 RMSE: 10.311295078556256
```