Machine Learning Laboratory – Assignment 1

- NAME :- ANURAG AVINASH SHEVALE
- CLASS :- BE COMP I
- ROLL NO :- 20

In [ ]:

```
#Name :- Anurag Avinash
Shevale#Class :- BE Comp I
#Roll No :- 20
```

In [1]:

```
#import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import warnings
#We do not want to see warnings
warnings.filterwarnings("ignore")
```

In [3]:

```
data = pd.read_csv(r"C:\Users\HP\Desktop\Machine Learning\Uber.csv")
```

In [4]:

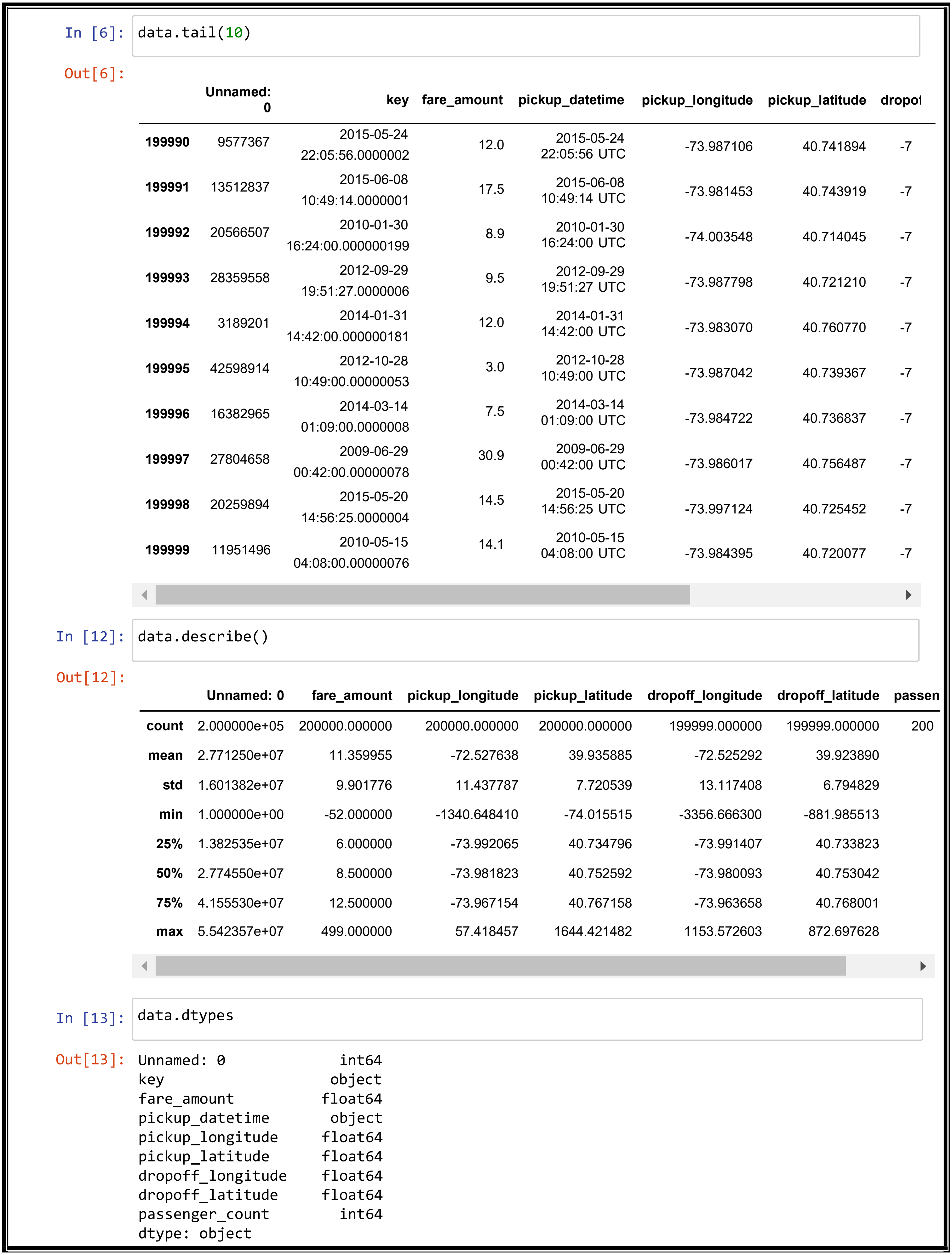
```
data1 = data.copy()
```

In [5]:

```
data.head(10)
```

Out[5]:

	Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_long
0	24238194	2015-05-07 19:52:06.0000003	7.5	2015-05-07 19:52:06 UTC	-73.999817	40.738354	-73.9
1	27835199	2009-07-17 20:04:56.0000002	7.7	2009-07-17 20:04:56 UTC	-73.994355	40.728225	-73.9
2	44984355	2009-08-24 21:45:00.00000061	12.9	2009-08-24 21:45:00 UTC	-74.005043	40.740770	-73.9
3	25894730	2009-06-26 08:22:21.0000001	5.3	2009-06-26 08:22:21 UTC	-73.976124	40.790844	-73.9
4	17610152	2014-08-28 17:47:00.000000188	16.0	2014-08-28 17:47:00 UTC	-73.925023	40.744085	-73.9
5	44470845	2011-02-12 02:27:09.0000006	4.9	2011-02-12 02:27:09 UTC	-73.969019	40.755910	-73.9
6	48725865	2014-10-12 07:04:00.0000002	24.5	2014-10-12 07:04:00 UTC	-73.961447	40.693965	-73.8
7	44195482	2012-12-11 13:52:00.00000029	2.5	2012-12-11 13:52:00 UTC	0.000000	0.000000	0.0
8	15822268	2012-02-17 09:32:00.00000043	9.7	2012-02-17 09:32:00 UTC	-73.975187	40.745767	-74.0
9	50611056	2012-03-29 19:06:00.000000273	12.5	2012-03-29 19:06:00 UTC	-74.001065	40.741787	-73.9



In [6]:
```
data.tail(10)
```

Out[6]:

	Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropo
199990	9577367	2015-05-24 22:05:56.0000002	12.0	2015-05-24 22:05:56 UTC	-73.987106	40.741894	-7
199991	13512837	2015-06-08 10:49:14.0000001	17.5	2015-06-08 10:49:14 UTC	-73.981453	40.743919	-7
199992	20566507	2010-01-30 16:24:00.000000199	8.9	2010-01-30 16:24:00 UTC	-74.003548	40.714045	-7
199993	28359558	2012-09-29 19:51:27.0000006	9.5	2012-09-29 19:51:27 UTC	-73.987798	40.721210	-7
199994	3189201	2014-01-31 14:42:00.000000181	12.0	2014-01-31 14:42:00 UTC	-73.983070	40.760770	-7
199995	42598914	2012-10-28 10:49:00.00000053	3.0	2012-10-28 10:49:00 UTC	-73.987042	40.739367	-7
199996	16382965	2014-03-14 01:09:00.0000008	7.5	2014-03-14 01:09:00 UTC	-73.984722	40.736837	-7
199997	27804658	2009-06-29 00:42:00.00000078	30.9	2009-06-29 00:42:00 UTC	-73.986017	40.756487	-7
199998	20259894	2015-05-20 14:56:25.0000004	14.5	2015-05-20 14:56:25 UTC	-73.997124	40.725452	-7
199999	11951496	2010-05-15 04:08:00.00000076	14.1	2010-05-15 04:08:00 UTC	-73.984395	40.720077	-7

In [12]:
```
data.describe()
```

Out[12]:

	Unnamed: 0	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passen
count	2.000000e+05	200000.000000	200000.000000	200000.000000	199999.000000	199999.000000	200
mean	2.771250e+07	11.359955	-72.527638	39.935885	-72.525292	39.923890	
std	1.601382e+07	9.901776	11.437787	7.720539	13.117408	6.794829	
min	1.000000e+00	-52.000000	-1340.648410	-74.015515	-3356.666300	-881.985513	
25%	1.382535e+07	6.000000	-73.992065	40.734796	-73.991407	40.733823	
50%	2.774550e+07	8.500000	-73.981823	40.752592	-73.980093	40.753042	
75%	4.155530e+07	12.500000	-73.967154	40.767158	-73.963658	40.768001	
max	5.542357e+07	499.000000	57.418457	1644.421482	1153.572603	872.697628	

In [13]:
```
data.dtypes
```

Out[13]:
```
Unnamed: 0             int64
key                   object
fare_amount          float64
pickup_datetime       object
pickup_longitude     float64
pickup_latitude      float64
dropoff_longitude    float64
dropoff_latitude     float64
passenger_count        int64
dtype: object
```

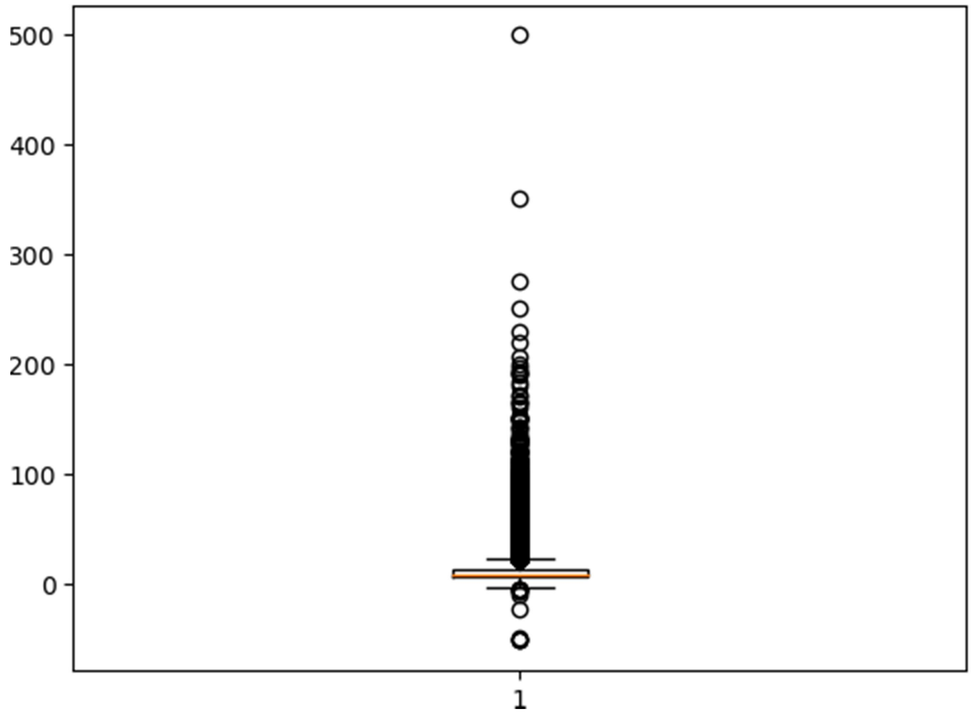
In [15]:
```
data.corr()
```

Out[15]:

	Unnamed: 0	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude
Unnamed: 0	1.000000	0.000589	0.000230	-0.000341	0.000270	0.000271
fare_amount	0.000589	1.000000	0.010457	-0.008481	0.008986	-0.011014
pickup_longitude	0.000230	0.010457	1.000000	-0.816461	0.833026	-0.846324
pickup_latitude	-0.000341	-0.008481	-0.816461	1.000000	-0.774787	0.702367
dropoff_longitude	0.000270	0.008986	0.833026	-0.774787	1.000000	-0.917010
dropoff_latitude	0.000271	-0.011014	-0.846324	0.702367	-0.917010	1.000000
passenger_count	0.002257	0.010150	-0.000414	-0.001560	0.000033	-0.000659

In [16]:
```
#Drop the rows with missing values
data1.dropna(inplace=True)
```

In [26]:
```
plt.boxplot(data1['fare_amount'])
figure = plt.figure(figsize = (3,1))
```



```
<Figure size 300x100 with 0 Axes>
```

In [27]:
```
#Remove Outliers
q_low =
data1["fare_amount"].quantile(0.01)q_hi
= data1["fare_amount"].quantile(0.99)

data1 = data1[(data1["fare_amount"] < q_hi) & (data1["fare_amount"] > q_low)]
```

In [29]:
```
#Time to apply learning models
from sklearn.model_selection import train_test_split
```

In [30]:
```
#Take x as predictor variable
x = data1.drop("fare_amount", axis = 1)
#And y as target variable
y = data1['fare_amount']
```

In [31]:
```
#Necessary to apply model
x['pickup_datetime'] = pd.to_numeric(pd.to_datetime(x['pickup_datetime']))x
= x.loc[:, x.columns.str.contains('^Unnamed')]
```

In [32]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 1
```

In [33]:
```
from sklearn.linear_model import LinearRegression
```

In [34]:
```
lrmodel = LinearRegression()
lrmodel.fit(x_train,y_train)
```

Out[34]:
```
LinearRegression()
```

In [35]:
```
#Prediction
predict = lrmodel.predict(x_test)
```

In [36]:
```
#Check Error
from sklearn.metrics import mean_squared_error
lrmodelrmse = np.sqrt(mean_squared_error(predict,
y_test))print("RMSE error for the model is ",
lrmodelrmse)
```

```
RMSE error for the model is  8.063863046328835
```

In [37]:
```
#Let's Apply Random Forest Regressor
from sklearn.ensemble import RandomForestRegressor
rfrmodel = RandomForestRegressor(n_estimators = 100, random_state = 101)
```

In [38]:
```
#Fit the Forest
rfrmodel.fit(x_train, y_train)
rfrmodel_pred = rfrmodel.predict(x_test)
```

In [39]:
```
#Errors for the forest
rfrmodel_rmse = np.sqrt(mean_squared_error(rfrmodel_pred,
y_test))print("RMSE value for Random Forest is:",rfrmodel_rmse)
```

```
RMSE value for Random Forest is: 9.757713738069647
```