In [1]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
```

In [2]:
```
data = pd.read_csv("uber.csv")
df = pd.DataFrame(data)
df
```

Out[2]:

	Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_
0	24238194	2015-05-07 19:52:06.0000003	7.5	2015-05-07 19:52:06 UTC	-73.999817	40
1	27835199	2009-07-17 20:04:56.0000002	7.7	2009-07-17 20:04:56 UTC	-73.994355	40
2	44984355	2009-08-24 21:45:00.00000061	12.9	2009-08-24 21:45:00 UTC	-74.005043	40
3	25894730	2009-06-26 08:22:21.0000001	5.3	2009-06-26 08:22:21 UTC	-73.976124	40
4	17610152	2014-08-28 17:47:00.000000188	16.0	2014-08-28 17:47:00 UTC	-73.925023	40
...	...	...	...	...	...	
199995	42598914	2012-10-28 10:49:00.00000053	3.0	2012-10-28 10:49:00 UTC	-73.987042	40
199996	16382965	2014-03-14 01:09:00.0000008	7.5	2014-03-14 01:09:00 UTC	-73.984722	40
199997	27804658	2009-06-29 00:42:00.00000078	30.9	2009-06-29 00:42:00 UTC	-73.986017	40
199998	20259894	2015-05-20 14:56:25.0000004	14.5	2015-05-20 14:56:25 UTC	-73.997124	40
199999	11951496	2010-05-15 04:08:00.00000076	14.1	2010-05-15 04:08:00 UTC	-73.984395	40

200000 rows × 9 columns

Create a dataset copy

In [3]:
```
df.copy()
```

Out[3]:

	Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_
0	24238194	2015-05-07 19:52:06.0000003	7.5	2015-05-07 19:52:06 UTC	-73.999817	40
1	27835199	2009-07-17 20:04:56.0000002	7.7	2009-07-17 20:04:56 UTC	-73.994355	40
2	44984355	2009-08-24 21:45:00.00000061	12.9	2009-08-24 21:45:00 UTC	-74.005043	40
3	25894730	2009-06-26 08:22:21.0000001	5.3	2009-06-26 08:22:21 UTC	-73.976124	40
4	17610152	2014-08-28 17:47:00.000000188	16.0	2014-08-28 17:47:00 UTC	-73.925023	40
...	...	...	...	...	...	
199995	42598914	2012-10-28 10:49:00.00000053	3.0	2012-10-28 10:49:00 UTC	-73.987042	40
199996	16382965	2014-03-14 01:09:00.0000008	7.5	2014-03-14 01:09:00 UTC	-73.984722	40
199997	27804658	2009-06-29 00:42:00.00000078	30.9	2009-06-29 00:42:00 UTC	-73.986017	40
199998	20259894	2015-05-20 14:56:25.0000004	14.5	2015-05-20 14:56:25 UTC	-73.997124	40
199999	11951496	2010-05-15 04:08:00.00000076	14.1	2010-05-15 04:08:00 UTC	-73.984395	40

200000 rows × 9 columns

Print Data

In [4]:
```
df.head()
```

Out[4]:

	Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_latitud
0	24238194	2015-05-07 19:52:06.0000003	7.5	2015-05-07 19:52:06 UTC	-73.999817	40.73835
1	27835199	2009-07-17 20:04:56.0000002	7.7	2009-07-17 20:04:56 UTC	-73.994355	40.72822
2	44984355	2009-08-24 21:45:00.00000061	12.9	2009-08-24 21:45:00 UTC	-74.005043	40.74077
3	25894730	2009-06-26 08:22:21.0000001	5.3	2009-06-26 08:22:21 UTC	-73.976124	40.79084
4	17610152	2014-08-28 17:47:00.000000188	16.0	2014-08-28 17:47:00 UTC	-73.925023	40.74408

Get Info

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200000 entries, 0 to 199999
Data columns (total 9 columns):
 #   Column             Non-Null Count   Dtype  
---  ------             --------------   -----  
 0   Unnamed: 0         200000 non-null  int64  
 1   key                200000 non-null  object 
 2   fare_amount        200000 non-null  float64
 3   pickup_datetime    200000 non-null  object 
 4   pickup_longitude   200000 non-null  float64
 5   pickup_latitude    200000 non-null  float64
 6   dropoff_longitude  199999 non-null  float64
 7   dropoff_latitude   199999 non-null  float64
 8   passenger_count    200000 non-null  int64  
dtypes: float64(5), int64(2), object(2)
memory usage: 13.7+ MB
```

Statistics of data

In [6]:
```
df.describe()
```

Out[6]:

	Unnamed: 0	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropof
count	2.000000e+05	200000.000000	200000.000000	200000.000000	199999.000000	1999
mean	2.771250e+07	11.359955	-72.527638	39.935885	-72.525292	
std	1.601382e+07	9.901776	11.437787	7.720539	13.117408	
min	1.000000e+00	-52.000000	-1340.648410	-74.015515	-3356.666300	-8
25%	1.382535e+07	6.000000	-73.992065	40.734796	-73.991407	
50%	2.774550e+07	8.500000	-73.981823	40.752592	-73.980093	
75%	4.155530e+07	12.500000	-73.967154	40.767158	-73.963658	
max	5.542357e+07	499.000000	57.418457	1644.421482	1153.572603	8

Missing Values

In [7]:
```
df.isnull().sum()
```

Out[7]:
```
Unnamed: 0           0
key                  0
fare_amount          0
pickup_datetime      0
pickup_longitude     0
pickup_latitude      0
dropoff_longitude    1
dropoff_latitude     1
passenger_count      0
dtype: int64
```

In [9]:
```
df.dropna(inplace=True)
```

Correlation

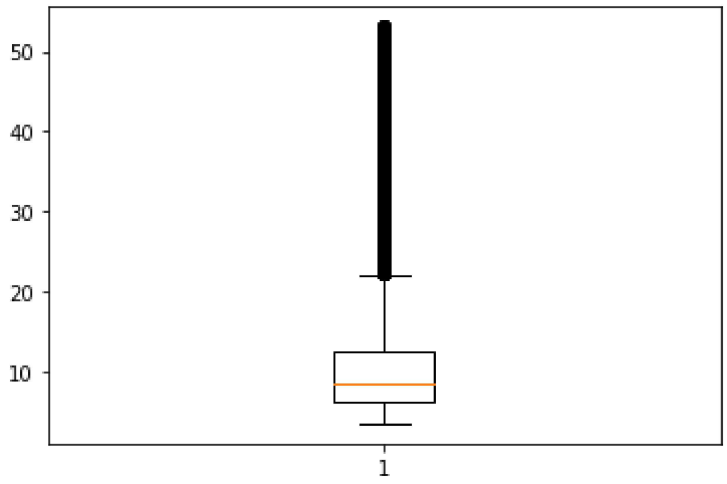
In [8]:
```
df.corr()
```

Out[8]:

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

In [35]:
```
plt.boxplot(x = df['fare_amount'])
```

Out[35]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x276b924a9a0>,
  <matplotlib.lines.Line2D at 0x276b924ad30>],
 'caps': [<matplotlib.lines.Line2D at 0x276b9268100>,
  <matplotlib.lines.Line2D at 0x276b9268490>],
 'boxes': [<matplotlib.lines.Line2D at 0x276b924a610>],
 'medians': [<matplotlib.lines.Line2D at 0x276b9268820>],
 'fliers': [<matplotlib.lines.Line2D at 0x276b9268bb0>],
 'means': []}
```



In [12]:
```
q_low = df['fare_amount'].quantile(0.01)
q_high = df['fare_amount'].quantile(0.99)

df = df[(df['fare_amount'] < q_high) & (df['fare_amount'] > q_low)]
```

In [13]:
```
df.isnull().sum()
```

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

Learning Model

In [14]:
```
from sklearn.model_selection import train_test_split
x = df.drop('fare_amount', axis = 1)
y = df['fare_amount']
```

In [15]:
```
x['pickup_datetime'] = pd.to_numeric(pd.to_datetime(x['pickup_datetime']))
x = x.loc[:, x.columns.str.contains('^Unnamed')]
```

In [16]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, rar
```

#Linear Regression

In [17]:
```
from sklearn.linear_model import LinearRegression
lrmodel = LinearRegression()
lrmodel.fit(x_train, y_train)
```

Out[17]:
```
LinearRegression()
```

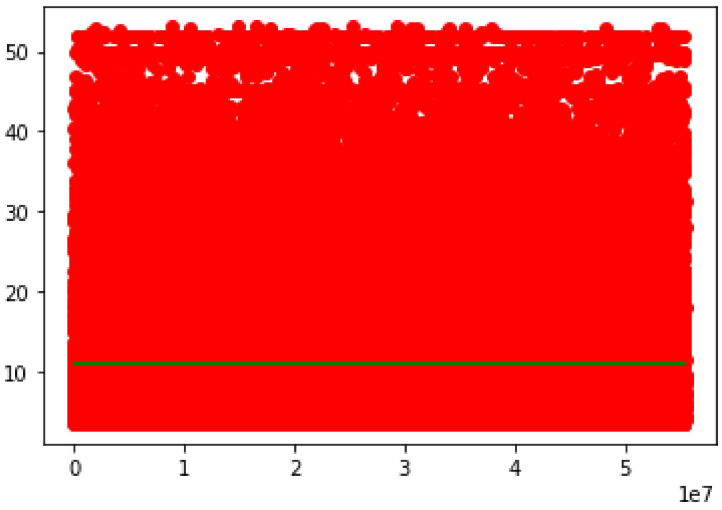
Prediction

In [19]:
```
pre = lrmodel.predict(x_test)
pre
```

Out[19]:
```
array([10.9555249 , 10.95231122, 10.95241127, ..., 10.95330031,
       10.95102493, 10.95402869])
```

Visualization of LinearRegression

In [32]:
```
plt.scatter(x_train, y_train, color = "red")
plt.plot(x_test, lrmodel.predict(x_test), color = "green")
plt.show()
```



Check Error

In [21]:
```
from sklearn.metrics import mean_squared_error
modelrmse = np.sqrt(mean_squared_error(pre, y_test))
print("RMSE error for the model is ", modelrmse)
```

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

Random Forest Regression

In [22]:
```
from sklearn.ensemble import RandomForestRegressor
rfmodel = RandomForestRegressor(n_estimators = 100, random_state = 101)
```

In [23]:
```
rfmodel.fit(x_train, y_train)
```

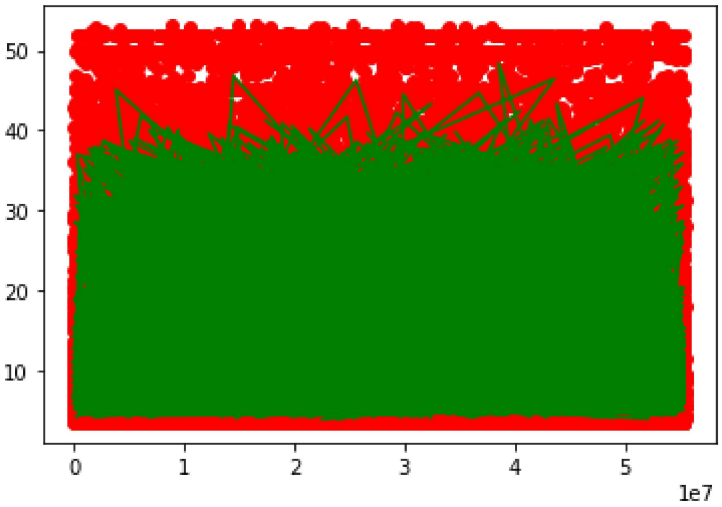
Out[23]:
```
RandomForestRegressor(random_state=101)
```

In [24]:
```
rfmodel_pred = rfmodel.predict(x_test)
rfmodel_pred
```

Out[24]:
```
array([11.125 ,  7.77  ,  6.927 , ...,  7.676 , 12.124 ,  6.7083])
```

Visulzation of Forest Regression

In [33]:
```
plt.scatter(x_train, y_train, color = "red")
plt.plot(x_test, rfmodel.predict(x_test), color = "green")
plt.show()
```



Check Error

In [25]:
```
rfrmodel_rmse = np.sqrt(mean_squared_error(rfmodel_pred, y_test))
print("RMSE value for Random Forest is:",rfrmodel_rmse)
```

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