In [25]:

```python
import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings("ignore")
```

In [2]:

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

In [3]:

```python
df = data.copy()
```

In [4]:

```python
df.head()
```

Out[4]:

	Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_lat
0	24238194	2015-05-07 19:52:06.0000003	7.5	2015-05-07 19:52:06 UTC	-73.999817	40.73
1	27835199	2009-07-17 20:04:56.0000002	7.7	2009-07-17 20:04:56 UTC	-73.994355	40.72
2	44984355	2009-08-24 21:45:00.00000061	12.9	2009-08-24 21:45:00 UTC	-74.005043	40.74
3	25894730	2009-06-26 08:22:21.0000001	5.3	2009-06-26 08:22:21 UTC	-73.976124	40.79
4	17610152	2014-08-28 17:47:00.000000188	16.0	2014-08-28 17:47:00 UTC	-73.925023	40.74

In [9]:

```python
print (df.shape)
print (df.columns)
```

```
(200000, 7)
Index(['fare_amount', 'pickup_datetime', 'pickup_longitude', 'picku
p_latitude',
       'dropoff_longitude', 'dropoff_latitude', 'passenger_count'],
      dtype='object')
```

In [10]:

```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200000 entries, 0 to 199999
Data columns (total 7 columns):
 #   Column             Non-Null Count   Dtype  
---  ------             --------------   -----  
 0   fare_amount        200000 non-null  float64
 1   pickup_datetime    200000 non-null  object 
 2   pickup_longitude   200000 non-null  float64
 3   pickup_latitude    200000 non-null  float64
 4   dropoff_longitude  199999 non-null  float64
 5   dropoff_latitude   199999 non-null  float64
 6   passenger_count    200000 non-null  int64  
dtypes: float64(5), int64(1), object(1)
memory usage: 10.7+ MB
```

In [12]:

```python
df["pickup_datetime"]=pd.to_datetime(df['pickup_datetime'])
```

In [13]:
```
df.head()
```

Out[13]:

	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropo
0	7.5	2015-05-07 19:52:06+00:00	-73.999817	40.738354	-73.999512	
1	7.7	2009-07-17 20:04:56+00:00	-73.994355	40.728225	-73.994710	
2	12.9	2009-08-24 21:45:00+00:00	-74.005043	40.740770	-73.962565	
3	5.3	2009-06-26 08:22:21+00:00	-73.976124	40.790844	-73.965316	
4	16.0	2014-08-28 17:47:00+00:00	-73.925023	40.744085	-73.973082	

In [14]:
```
df.describe()
```

Out[14]:

	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude
count	200000.000000	200000.000000	200000.000000	199999.000000	199999.000000
mean	11.359955	-72.527638	39.935885	-72.525292	39.923890
std	9.901776	11.437787	7.720539	13.117408	6.794829
min	-52.000000	-1340.648410	-74.015515	-3356.666300	-881.985513
25%	6.000000	-73.992065	40.734796	-73.991407	40.733823
50%	8.500000	-73.981823	40.752592	-73.980093	40.753042
75%	12.500000	-73.967154	40.767158	-73.963658	40.768001
max	499.000000	57.418457	1644.421482	1153.572603	872.697628

In [15]:
```
df.isnull().sum()
```

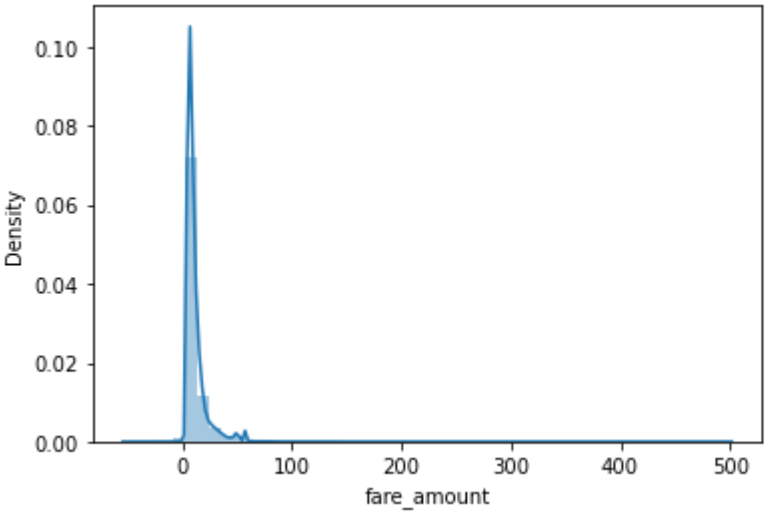
Out[15]:
```
fare_amount          0
pickup_datetime      0
pickup_longitude     0
pickup_latitude      0
dropoff_longitude    1
dropoff_latitude     1
passenger_count      0
dtype: int64
```

In [17]:
```
df.dropna(inplace=True)
print(df.isnull().sum())
```

```
fare_amount          0
pickup_datetime      0
pickup_longitude     0
pickup_latitude      0
dropoff_longitude    0
dropoff_latitude     0
passenger_count      0
dtype: int64
```

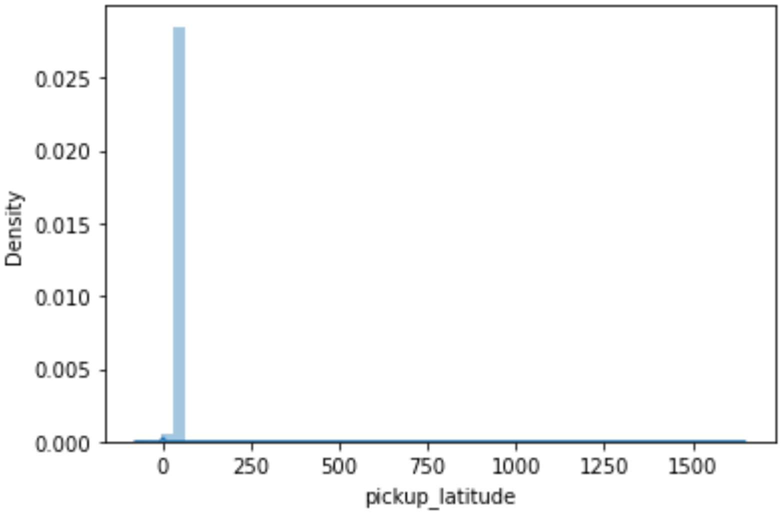
```
In [18]: import matplotlib.pyplot as plt
         import seaborn as sns
         %matplotlib inline
```

```
In [26]: sns.distplot(df['fare_amount'])

Out[26]: <AxesSubplot:xlabel='fare_amount', ylabel='Density'>
```

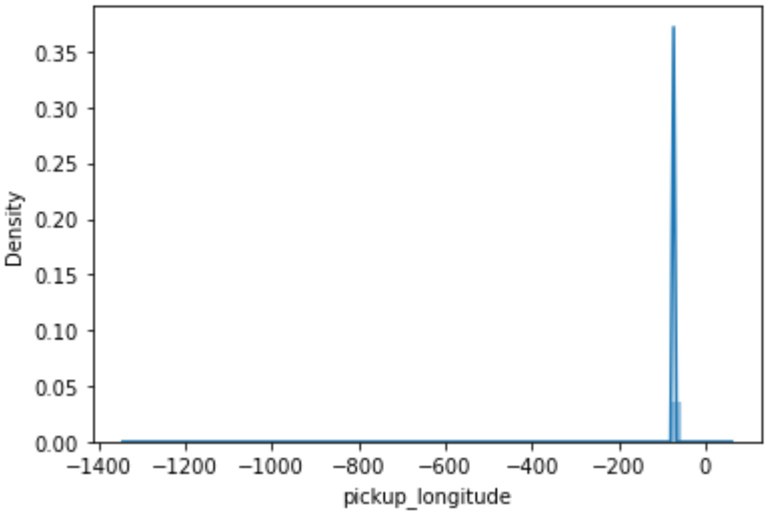


Above we can see some fares having negative values

```
In [27]: sns.distplot(df['pickup_latitude'])

Out[27]: <AxesSubplot:xlabel='pickup_latitude', ylabel='Density'>
```

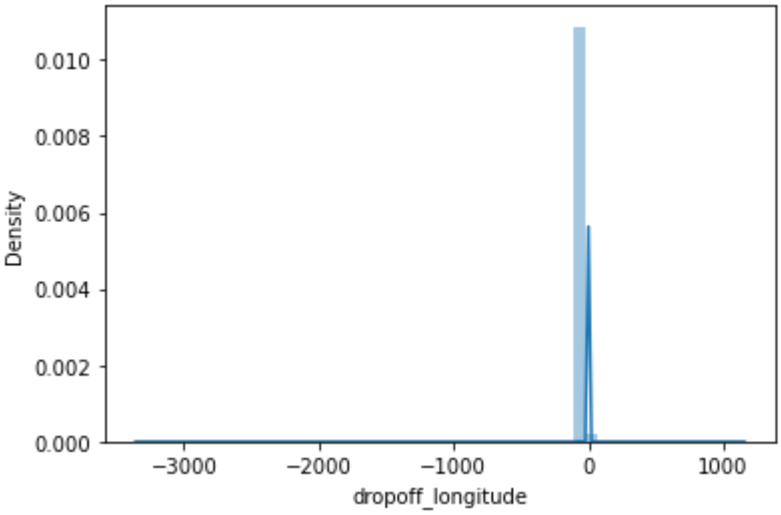


```
In [28]: sns.distplot(df['pickup_longitude'])

Out[28]: <AxesSubplot:xlabel='pickup_longitude', ylabel='Density'>
```



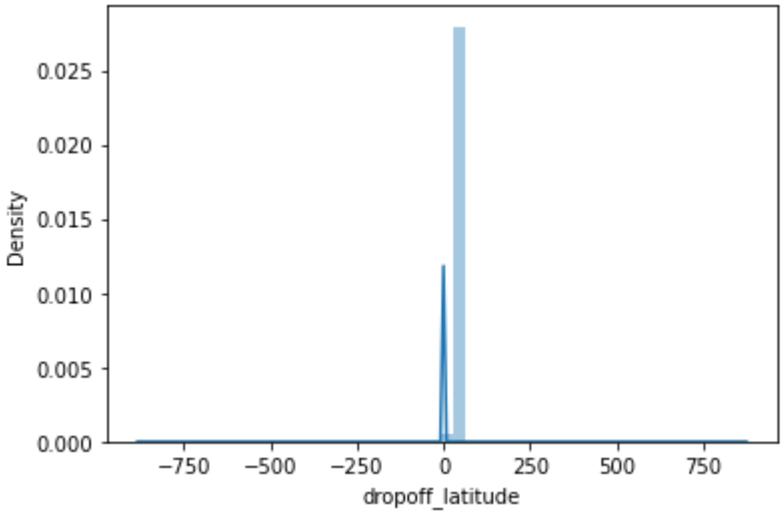
Negative and Positive values are excedding far behond the real limit.

```
In [29]: sns.distplot(df['dropoff_longitude'])

Out[29]: <AxesSubplot:xlabel='dropoff_longitude', ylabel='Density'>
```



In [30]:
```
sns.distplot(df['dropoff_latitude'])
```

Out[30]:
```
<AxesSubplot:xlabel='dropoff_latitude', ylabel='Density'>
```



In [32]:
```
import calendar
df['day']=df['pickup_datetime'].apply(lambda x:x.day)
df['hour']=df['pickup_datetime'].apply(lambda x:x.hour)
df['weekday']=df['pickup_datetime'].apply(lambda x:calendar.day_name[
df['month']=df['pickup_datetime'].apply(lambda x:x.month)
df['year']=df['pickup_datetime'].apply(lambda x:x.year)
```

In [33]:
```
df.head()
```

Out[33]:

	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropo
0	7.5	2015-05-07 19:52:06+00:00	-73.999817	40.738354	-73.999512	
1	7.7	2009-07-17 20:04:56+00:00	-73.994355	40.728225	-73.994710	
2	12.9	2009-08-24 21:45:00+00:00	-74.005043	40.740770	-73.962565	
3	5.3	2009-06-26 08:22:21+00:00	-73.976124	40.790844	-73.965316	
4	16.0	2014-08-28 17:47:00+00:00	-73.925023	40.744085	-73.973082	

In [34]:
```
df.weekday = df.weekday.map({'Sunday':0,'Monday':1,'Tuesday':2,'Wedne
```

In [35]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 199999 entries, 0 to 199999
Data columns (total 12 columns):
 #   Column             Non-Null Count   Dtype
---  ------             --------------   -----
 0   fare_amount        199999 non-null  float64
 1   pickup_datetime    199999 non-null  datetime64[ns, UTC]
 2   pickup_longitude   199999 non-null  float64
 3   pickup_latitude    199999 non-null  float64
 4   dropoff_longitude  199999 non-null  float64
 5   dropoff_latitude   199999 non-null  float64
 6   passenger_count    199999 non-null  int64
 7   day                199999 non-null  int64
 8   hour               199999 non-null  int64
 9   weekday            199999 non-null  int64
 10  month              199999 non-null  int64
 11  year               199999 non-null  int64
dtypes: datetime64[ns, UTC](1), float64(5), int64(6)
memory usage: 19.8 MB
```

In [37]:
```
df=df[df['passenger_count']<=8]
```

In [38]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 199998 entries, 0 to 199999
Data columns (total 12 columns):
 #   Column             Non-Null Count   Dtype
---  ------             --------------   -----
 0   fare_amount        199998 non-null  float64
 1   pickup_datetime    199998 non-null  datetime64[ns, UTC]
 2   pickup_longitude   199998 non-null  float64
 3   pickup_latitude    199998 non-null  float64
 4   dropoff_longitude  199998 non-null  float64
 5   dropoff_latitude   199998 non-null  float64
 6   passenger_count    199998 non-null  int64
 7   day                199998 non-null  int64
 8   hour               199998 non-null  int64
 9   weekday            199998 non-null  int64
 10  month              199998 non-null  int64
 11  year               199998 non-null  int64
dtypes: datetime64[ns, UTC](1), float64(5), int64(6)
memory usage: 19.8 MB
```

In [40]:
```
from sklearn.model_selection import train_test_split
```

In [41]:
```
x=df.drop("fare_amount", axis=1)
```

In [42]:
```
y=df['fare_amount']
```

In [43]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,ra
```

In [44]:
```
x_train.head()
```

Out[44]:

	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitud
80768	2009-02-22 01:12:00+00:00	-73.983703	40.725752	-73.972000	40.79388
111783	2009-03-07 14:49:00+00:00	-73.961175	40.760667	-73.976507	40.74757
24615	2011-03-17 11:51:08+00:00	-73.947784	40.783111	-73.955408	40.77940
46932	2010-01-15 07:01:38+00:00	-73.980596	40.733797	-73.972092	40.74729
86655	2014-06-28 19:25:00+00:00	-73.963035	40.758380	-73.987877	40.74547

In [45]:
```
x_test.head()
```

Out[45]:

	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitud
13588	2013-06-25 22:32:00+00:00	-73.982810	40.771687	-73.977065	40.76320
29803	2011-02-20 20:16:00+00:00	-73.991985	40.725763	-73.995762	40.75979
138266	2011-08-09 20:15:00+00:00	-73.989458	40.741665	-73.983463	40.75884
82856	2011-11-17 18:49:03+00:00	-73.973200	40.748100	-73.973500	40.74820
162748	2012-08-26 04:37:52+00:00	-73.856011	40.824512	-73.981732	40.76175

In [50]:
```
x_train['pickup_datetime'] = pd.to_numeric(pd.to_datetime(x_train['pi
x_test['pickup_datetime'] = pd.to_numeric(pd.to_datetime(x_test['pick
```

In [51]:
```
x_train.shape
```

Out[51]:
```
(159998, 11)
```

In [52]:
```
x_test.shape
```

Out[52]:
```
(40000, 11)
```

In [53]:
```
from sklearn.linear_model import LinearRegression
```

In [54]:
```
lrmodel=LinearRegression()
lrmodel.fit(x_train, y_train)
```

Out[54]:
```
LinearRegression()
```

In [55]:
```
predictedvalues = lrmodel.predict(x_test)
```

In [56]:

```
from sklearn.metrics import mean_squared_error
lrmodelrmse = np.sqrt(mean_squared_error(predictedvalues, y_test))
print("RMSE value for Linear regression is", lrmodelrmse)
```

```
RMSE value for Linear regression is 9.922284743566864
```

In [57]:

```
from sklearn.ensemble import RandomForestRegressor
rfrmodel = RandomForestRegressor(n_estimators=100, random_state=101)
```

In [58]:

```
rfrmodel.fit(x_train,y_train)
rfrmodel_pred= rfrmodel.predict(x_test)
```

In [59]:

```
rfrmodel_rmse=np.sqrt(mean_squared_error(rfrmodel_pred, y_test))
print("RMSE value for Random forest regression is ",rfrmodel_rmse)
```

```
RMSE value for Random forest regression is  4.79773791418236
```

In [ ]:

```

```