Assignment No 7 - ML-1

October 8, 2024

1 #Predict the price of the Uber ride from a given pickup point to the agreed drop-off location. Perform following tasks:

1. Pre-process the dataset.
2. Identify outliers.
3. Check the correlation.
4. Implement linear regression and random forest regression models.
5. Evaluate the models and compare their respective scores like R2, RMSE, etc. Dataset link: https://www.kaggle.com/datasets/yasserh/uber-fares-dataset

[11]:
```
#Importing the required libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

[12]:
```
#importing the dataset
df  = pd.read_csv("/home/pc13/Downloads/uber.csv")
```

1.1 1. Pre-process the dataset.

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

[4]:
```
   Unnamed: 0                            key  fare_amount  \
0    24238194    2015-05-07 19:52:06.0000003          7.5
1    27835199    2009-07-17 20:04:56.0000002          7.7
2    44984355   2009-08-24 21:45:00.00000061         12.9
3    25894730    2009-06-26 08:22:21.0000001          5.3
4    17610152  2014-08-28 17:47:00.000000188         16.0

           pickup_datetime  pickup_longitude  pickup_latitude  \
0  2015-05-07 19:52:06 UTC        -73.999817        40.738354
1  2009-07-17 20:04:56 UTC        -73.994355        40.728225
2  2009-08-24 21:45:00 UTC        -74.005043        40.740770
3  2009-06-26 08:22:21 UTC        -73.976124        40.790844
4  2014-08-28 17:47:00 UTC        -73.925023        40.744085
```

```
   dropoff_longitude  dropoff_latitude  passenger_count
0         -73.999512         40.723217                1
1         -73.994710         40.750325                1
2         -73.962565         40.772647                1
3         -73.965316         40.803349                3
4         -73.973082         40.761247                5
```

[5]:
```
df.info() #To get the required information of the dataset
```

```
<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
```

[6]:
```
df.columns #TO get number of columns in the dataset
```

[6]:
```
Index(['Unnamed: 0', 'key', 'fare_amount', 'pickup_datetime',
       'pickup_longitude', 'pickup_latitude', 'dropoff_longitude',
       'dropoff_latitude', 'passenger_count'],
      dtype='object')
```

[7]:
```
df = df.drop(['Unnamed: 0', 'key'], axis= 1) #To drop unnamed column as it␣
 ↪isn't required
```

[8]:
```
df.head()
```

[8]:
```
   fare_amount          pickup_datetime  pickup_longitude  pickup_latitude  \
0          7.5  2015-05-07 19:52:06 UTC        -73.999817        40.738354   
1          7.7  2009-07-17 20:04:56 UTC        -73.994355        40.728225   
2         12.9  2009-08-24 21:45:00 UTC        -74.005043        40.740770   
3          5.3  2009-06-26 08:22:21 UTC        -73.976124        40.790844   
4         16.0  2014-08-28 17:47:00 UTC        -73.925023        40.744085   

   dropoff_longitude  dropoff_latitude  passenger_count
0         -73.999512         40.723217                1
```

```
1          -73.994710         40.750325                1
2          -73.962565         40.772647                1
3          -73.965316         40.803349                3
4          -73.973082         40.761247                5
```

[9]:
```
df.shape #To get the total (Rows,Columns)
```

[9]:
```
(200000, 7)
```

[10]:
```
df.dtypes #To get the type of each column
```

[10]:
```
fare_amount          float64
pickup_datetime       object
pickup_longitude     float64
pickup_latitude      float64
dropoff_longitude    float64
dropoff_latitude     float64
passenger_count        int64
dtype: object
```

[ ]:
```
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
```

[ ]:
```
df.describe() #To get statistics of each columns
```

[ ]:
```
          fare_amount  pickup_longitude  pickup_latitude  dropoff_longitude  \
count  200000.000000     200000.000000    200000.000000      199999.000000
mean       11.359955        -72.527638        39.935885         -72.525292
std         9.901776         11.437787         7.720539          13.117408
min       -52.000000      -1340.648410       -74.015515       -3356.666300
25%         6.000000        -73.992065        40.734796         -73.991407
50%         8.500000        -73.981823        40.752592         -73.980093
75%        12.500000        -73.967154        40.767158         -73.963658
```

```
max       499.000000         57.418457      1644.421482        1153.572603

       dropoff_latitude  passenger_count
count     199999.000000    200000.000000
mean          39.923890         1.684535
std            6.794829         1.385997
min         -881.985513         0.000000
25%           40.733823         1.000000
50%           40.753042         1.000000
75%           40.768001         2.000000
max          872.697628       208.000000
```

1.1.1 Filling Missing values

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

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

[ ]:
```
df['dropoff_latitude'].fillna(value=df['dropoff_latitude'].mean(),inplace =␣
 ↪True)
df['dropoff_longitude'].fillna(value=df['dropoff_longitude'].median(),inplace =␣
 ↪True)
```

[ ]:
```
df['dropoff_latitude']
```

[ ]:
```
0         40.723217
1         40.750325
2         40.772647
3         40.803349
4         40.761247
             …
199995    40.740297
199996    40.739620
199997    40.692588
199998    40.695415
199999    40.768793
Name: dropoff_latitude, Length: 200000, dtype: float64
```

[ ]:
```
df.head(10)
```

[ ]:

	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	\
0	7.5	2015-05-07 19:52:06 UTC	-73.999817	40.738354	
1	7.7	2009-07-17 20:04:56 UTC	-73.994355	40.728225	
2	12.9	2009-08-24 21:45:00 UTC	-74.005043	40.740770	
3	5.3	2009-06-26 08:22:21 UTC	-73.976124	40.790844	
4	16.0	2014-08-28 17:47:00 UTC	-73.925023	40.744085	
5	4.9	2011-02-12 02:27:09 UTC	-73.969019	40.755910	
6	24.5	2014-10-12 07:04:00 UTC	-73.961447	40.693965	
7	2.5	2012-12-11 13:52:00 UTC	0.000000	0.000000	
8	9.7	2012-02-17 09:32:00 UTC	-73.975187	40.745767	
9	12.5	2012-03-29 19:06:00 UTC	-74.001065	40.741787	

	dropoff_longitude	dropoff_latitude	passenger_count
0	-73.999512	40.723217	1
1	-73.994710	40.750325	1
2	-73.962565	40.772647	1
3	-73.965316	40.803349	3
4	-73.973082	40.761247	5
5	-73.969019	40.755910	1
6	-73.871195	40.774297	5
7	0.000000	0.000000	1
8	-74.002720	40.743537	1
9	-73.963040	40.775012	1

[ ]:
```
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
```

[ ]:
```
df.dtypes
```

[ ]:
```
fare_amount          float64
pickup_datetime       object
pickup_longitude     float64
pickup_latitude      float64
dropoff_longitude    float64
dropoff_latitude     float64
passenger_count        int64
dtype: object
```

1.1.2 Column pickup_datetime is in wrong format (Object). Convert it to DateTime Format

[ ]:
```
df.pickup_datetime = pd.to_datetime(df.pickup_datetime,)
```

[ ]:
```
df.dtypes
```

[ ]:
```
fare_amount                       float64
pickup_datetime      datetime64[ns, UTC]
pickup_longitude                  float64
pickup_latitude                   float64
dropoff_longitude                 float64
dropoff_latitude                  float64
passenger_count                     int64
dtype: object
```

1.1.3 To segregate each time of date and time

[ ]:
```
df= df.assign(hour = df.pickup_datetime.dt.hour,
              day= df.pickup_datetime.dt.day,
              month = df.pickup_datetime.dt.month,
              year = df.pickup_datetime.dt.year,
              dayofweek = df.pickup_datetime.dt.dayofweek)
```

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

[ ]:

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

	dropoff_longitude	dropoff_latitude	passenger_count	hour	day	month	\
0	-73.999512	40.723217	1	19	7	5	
1	-73.994710	40.750325	1	20	17	7	
2	-73.962565	40.772647	1	21	24	8	
3	-73.965316	40.803349	3	8	26	6	
4	-73.973082	40.761247	5	17	28	8	

	year	dayofweek
0	2015	3
1	2009	4
2	2009	0
3	2009	4
4	2014	3

[ ]:

[ ]:
```
# drop the column 'pickup_daetime' using drop()
# 'axis = 1' drops the specified column

df = df.drop('pickup_datetime',axis=1)
```

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

[ ]:
```
   fare_amount  pickup_longitude  pickup_latitude  dropoff_longitude  \
0          7.5        -73.999817        40.738354         -73.999512
1          7.7        -73.994355        40.728225         -73.994710
2         12.9        -74.005043        40.740770         -73.962565
3          5.3        -73.976124        40.790844         -73.965316
4         16.0        -73.925023        40.744085         -73.973082

   dropoff_latitude  passenger_count  hour  day  month  year  dayofweek
0         40.723217                1    19    7      5  2015          3
1         40.750325                1    20   17      7  2009          4
2         40.772647                1    21   24      8  2009          0
3         40.803349                3     8   26      6  2009          4
4         40.761247                5    17   28      8  2014          3
```

[ ]:
```
df.dtypes
```

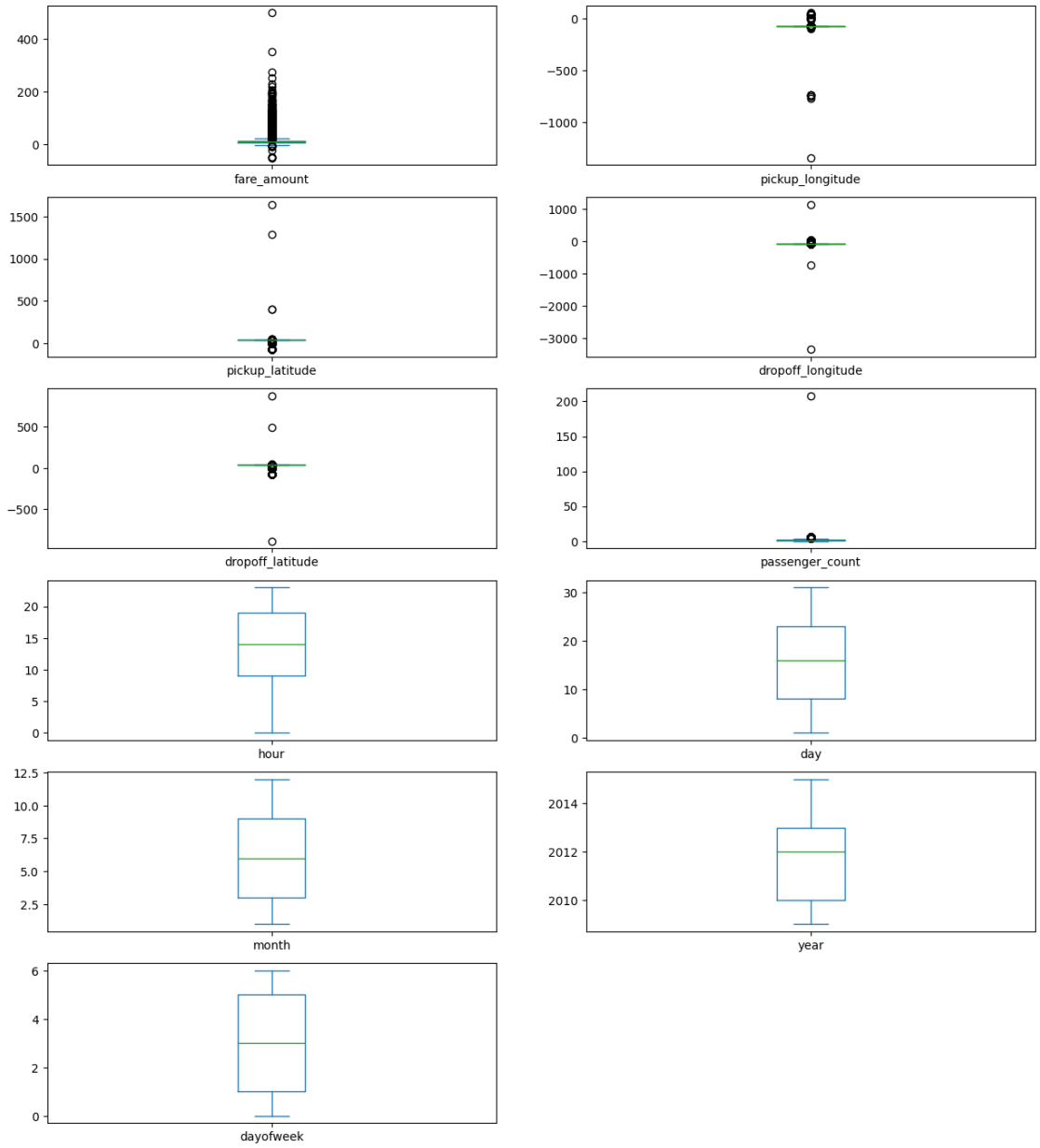
[ ]:
```
fare_amount          float64
pickup_longitude     float64
pickup_latitude      float64
dropoff_longitude    float64
dropoff_latitude     float64
passenger_count        int64
hour                   int64
day                    int64
month                  int64
year                   int64
dayofweek              int64
dtype: object
```

1.2 Checking outliers and filling them

[ ]:
```
df.plot(kind = "box",subplots = True,layout = (7,2),figsize=(15,20)) #Boxplot␣
↪to check the outliers
```

[ ]:
```
fare_amount            Axes(0.125,0.786098;0.352273x0.0939024)
pickup_longitude     Axes(0.547727,0.786098;0.352273x0.0939024)
pickup_latitude        Axes(0.125,0.673415;0.352273x0.0939024)
dropoff_longitude    Axes(0.547727,0.673415;0.352273x0.0939024)
```

```
dropoff_latitude        Axes(0.125,0.560732;0.352273x0.0939024)
passenger_count      Axes(0.547727,0.560732;0.352273x0.0939024)
hour                    Axes(0.125,0.448049;0.352273x0.0939024)
day                  Axes(0.547727,0.448049;0.352273x0.0939024)
month                   Axes(0.125,0.335366;0.352273x0.0939024)
year                 Axes(0.547727,0.335366;0.352273x0.0939024)
dayofweek               Axes(0.125,0.222683;0.352273x0.0939024)
dtype: object
```

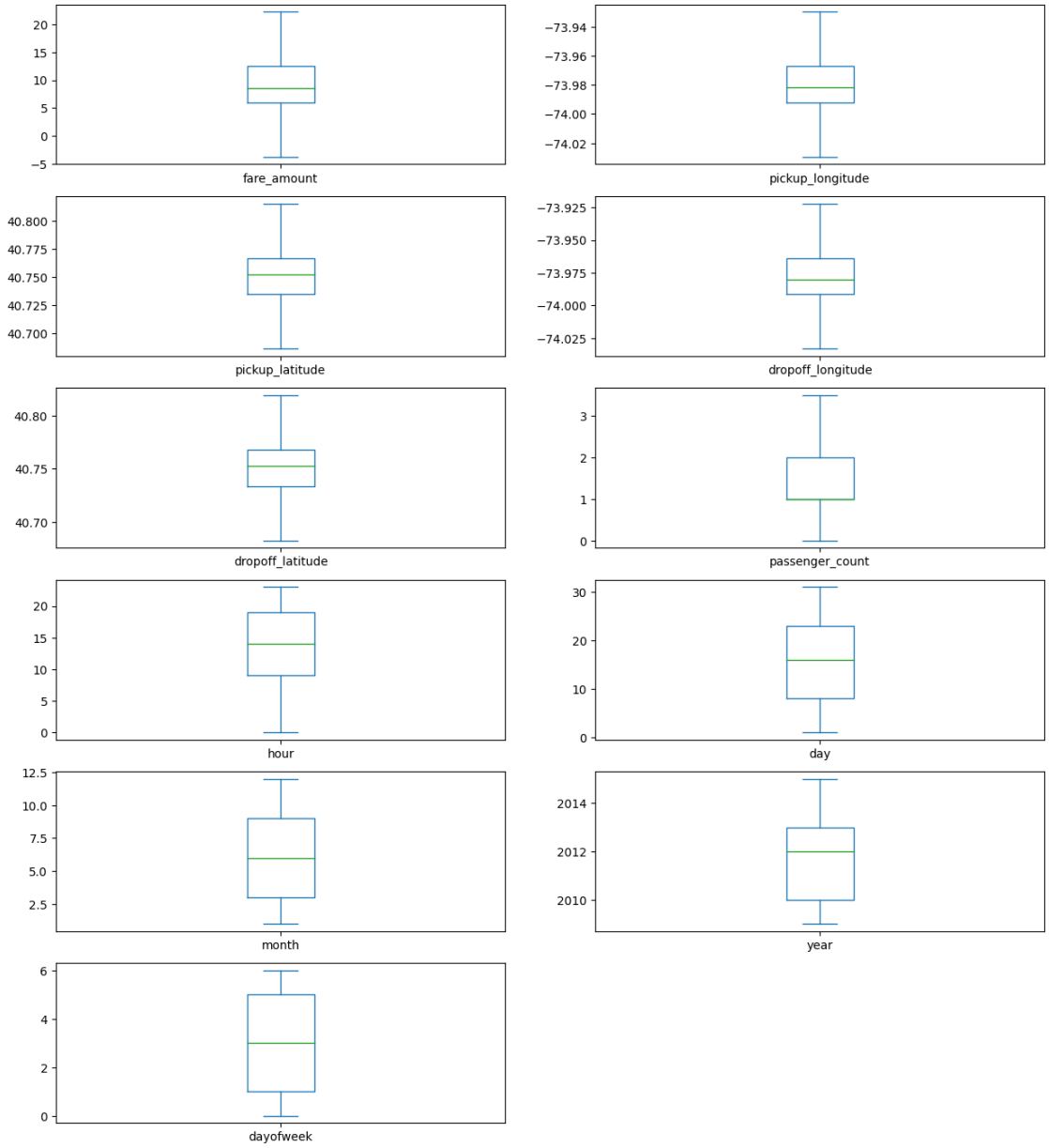


[ ]:
```
#Using the InterQuartile Range to fill the values
def remove_outlier(df1 , col):
    Q1 = df1[col].quantile(0.25)
    Q3 = df1[col].quantile(0.75)
    IQR = Q3 - Q1
    lower_whisker = Q1-1.5*IQR
    upper_whisker = Q3+1.5*IQR
    df[col] = np.clip(df1[col] , lower_whisker , upper_whisker)
    return df1

def treat_outliers_all(df1 , col_list):
    for c in col_list:
        df1 = remove_outlier(df , c)
    return df1
```

[ ]:
```
df = treat_outliers_all(df , df.iloc[: , 0::])
```

[ ]:
```
df.plot(kind = "box",subplots = True,layout = (7,2),figsize=(15,20)) #Boxplot␣
↪shows that dataset is free from outliers
```

[ ]:
```
fare_amount            Axes(0.125,0.786098;0.352273x0.0939024)
pickup_longitude     Axes(0.547727,0.786098;0.352273x0.0939024)
pickup_latitude        Axes(0.125,0.673415;0.352273x0.0939024)
dropoff_longitude    Axes(0.547727,0.673415;0.352273x0.0939024)
dropoff_latitude       Axes(0.125,0.560732;0.352273x0.0939024)
passenger_count      Axes(0.547727,0.560732;0.352273x0.0939024)
hour                   Axes(0.125,0.448049;0.352273x0.0939024)
day                  Axes(0.547727,0.448049;0.352273x0.0939024)
month                  Axes(0.125,0.335366;0.352273x0.0939024)
year                 Axes(0.547727,0.335366;0.352273x0.0939024)
dayofweek              Axes(0.125,0.222683;0.352273x0.0939024)
dtype: object
```



[ ]:
```
#!pip install haversine
```

[ ]:
```
!pip install haversine
import haversine as hs  #Calculate the distance using Haversine to calculate␣
 ↪the distance between to points. Can't use Eucladian as it is for flat␣
 ↪surface.
travel_dist = []
for pos in range(len(df['pickup_longitude'])):
        long1,lati1,long2,lati2 =␣
 ↪[df['pickup_longitude'][pos],df['pickup_latitude'][pos],df['dropoff_longitude'][pos],df['dr
```

```
        loc1=(lati1,long1)
        loc2=(lati2,long2)
        c = hs.haversine(loc1,loc2)
        travel_dist.append(c)

print(travel_dist)
df['dist_travel_km'] = travel_dist
df.head()
```

```
Collecting haversine
  Downloading haversine-2.8.0-py2.py3-none-any.whl (7.7 kB)
Installing collected packages: haversine
Successfully installed haversine-2.8.0
```

```
IOPub data rate exceeded.
The notebook server will temporarily stop sending output
to the client in order to avoid crashing it.
To change this limit, set the config variable
`--NotebookApp.iopub_data_rate_limit`.

Current values:
NotebookApp.iopub_data_rate_limit=1000000.0 (bytes/sec)
NotebookApp.rate_limit_window=3.0 (secs)
```

[ ]:

	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count	hour	day	month	year	dayofweek	dist_travel_km
0	7.5	-73.999817	40.738354	-73.999512	40.723217	1.0	19	7	5	2015	3	1.683325
1	7.7	-73.994355	40.728225	-73.994710	40.750325	1.0	20	17	7	2009	4	2.457593
2	12.9	-74.005043	40.740770	-73.962565	40.772647	1.0	21	24	8	2009	0	5.036384
3	5.3	-73.976124	40.790844	-73.965316	40.803349	3.0	8	26	6	2009	4	1.661686
4	16.0	-73.929786	40.744085	-73.973082	40.761247	3.5	17	28	8	2014	3	4.116088

[ ]:
```
travel_dist
```

```
[ ]: [1.6833250775073447,
      2.4575932783467835,
      5.036384146783453,
      1.661685753650294,
      4.1160881895779955,
      0.0,
      9.529776771411873,
      0.7719995370784762,
      2.3327142314177545,
      4.889423641655177,
      2.2508607308770285,
      0.7719995370784762,
      0.3022521108558365,
      3.5812557740132496,
      1.3099517093917648,
      1.716279792276335,
      0.7299665570466272,
      2.515953547298386,
      1.790321726187665,
      1.0347050399795192,
      2.4902472008677727,
      0.9594701844599927,
      1.2613897673437817,
      1.7517650017211177,
      6.1932445014761095,
      2.736192584061414,
      0.7232537124105735,
      3.229443537425455,
      1.4295172964395384,
      2.233699311547041,
      11.081439278313375,
      1.8950491608266506,
      1.9049353402365328,
      3.1821178242889583,
      4.467427177611096,
      2.9230236888626995,
      1.200213842045202,
      2.635790807404098,
      2.253613903554444,
      9.186987668633932,
      4.826533532872274,
      1.2502926870845612,
      0.7984742276755328,
      0.840396152683202,
      0.38519924001701444,
      2.434346176233359,
      3.127909533264099,
```

3.7346507724526368,
0.0,
4.504360509008704,
1.5579008497303448,
4.923160610490316,
7.683158068167262,
0.6855715417446852,
4.523573083259081,
4.071896214031055,
1.1746737081325782,
1.2923819784535335,
1.21232151965934,
0.8729716749950138,
2.1074756058990856,
5.901643616945056,
0.6855640840334924,
5.634300221883286,
0.7550126930090186,
0.7719995370784762,
1.0321749030315608,
5.847123145760689,
3.0962647003859574,
2.068044688771067,
1.5086978898301036,
3.310688313214333,
0.5099461974827887,
1.9072963928940811,
6.152304270973421,
6.100540899228414,
1.9571596722303939,
4.580228282398262,
4.247622321534987,
0.9043278463938618,
2.499149034842446,
0.41957484389859023,
9.507952873857636,
4.741483270405726,
0.0,
2.944307722325589,
1.2980360742853914,
6.903605106108037,
1.80509590056403,
1.3540037466619592,
1.3985793924855354,
1.0855348055513168,
0.7719995370784762,
1.2560044656304787,

1.2322854358550752,
2.177184869699798,
0.5913097887913589,
0.7059208750601758,
6.346979417679082,
5.5182615502859935,
2.0175432998561025,
0.5730744516191166,
0.6404690741121153,
2.800572945460828,
6.571086756415957,
2.151452815480442,
0.9853760531912669,
1.705646050191744,
0.6325957503446071,
2.5205599224510924,
5.5555911761089485,
1.2902717916405408,
2.2039006742188842,
6.669364569080815,
1.3620086545357155,
0.8339963144694047,
2.339481564210149,
1.7643888489494255,
3.2125588562896175,
1.9504898596578084,
0.7719995370784762,
2.3992175140695715,
1.4694773253693545,
1.9741348182430898,
3.740438452483117,
2.176311362468996,
1.2195234142647462,
1.1528728732020785,
2.1235004521220673,
3.5739610800397585,
1.5679009708942961,
1.0121554184708625,
3.513305615404955,
0.7719995370784762,
1.7841984734351823,
3.3520418164013033,
3.7726640147517996,
5.060744411441399,
2.7231858480669056,
4.715419442881193,
2.1793115241552314,

1.70866368385628,
1.19476321862848,
1.6390222354302528,
3.0685608081368967,
4.479393523696071,
1.37341302240955,
3.481380394599612,
1.0186481929664464,
3.6116963717578914,
5.578421201063091,
0.919827160362576,
3.5885450229857203,
3.4468771664826057,
0.6774823849168342,
1.166554730338164,
0.8456297568751322,
1.481729691783599,
0.5808106243688994,
1.4819618793333265,
0.9054249595313008,
2.6227549008377546,
4.431784656145714,
1.5240431957846732,
1.8947043339560112,
3.0793847352288597,
0.0,
1.066524360439298,
3.907631850108434,
0.7719995370784762,
1.3238717252730092,
1.4225749614603436,
2.31597774099886,
1.6598673834539075,
2.3359208961641458,
3.6193655097463826,
2.9547192169738765,
1.0581972284125865,
5.259926514450871,
0.9517585737436477,
2.622033614711671,
3.6544207914251037,
5.744520049956267,
1.4099935576123503,
2.3731618371467786,
0.7459983249024945,
2.037028759274383,
4.447412149306085,

6.266021875809639,
6.102571795022024,
1.136059938462684,
0.4445695655507225,
8.305590458344271,
2.9969312113645437,
1.6880697796691957,
3.4192543921764096,
6.379442517616808,
3.239160591551795,
1.3160927544902608,
6.3239664101151,
1.403449368792598,
1.824875242257348,
4.806348751494792,
1.1540979512543523,
9.355053828737084,
8.963162127856634,
1.8044575499927282,
0.2630230396164467,
1.0843264058702755,
1.1339639614584733,
3.066968595549588,
0.8447879467847005,
2.4567303881601017,
1.9441282991504945,
4.424827920856918,
0.7549463978280366,
6.340800401734032,
1.4864084038744412,
4.615772843385764,
1.7514377249484463,
1.9070604598703718,
5.0955040732855235,
6.88898020535702,
10.074405138041094,
0.8717589098431497,
4.946085236853659,
9.422584168938348,
0.8805574307190026,
1.4136419949939485,
2.1171056720053976,
2.396713554128903,
1.428890513842054,
1.2696816522951435,
3.0761080538691283,
0.777776683309271,

4.524011276386306,
1.8316384585992311,
0.7719995370784762,
0.5878310700387969,
2.2671016634316494,
6.109066282333016,
1.8301186747611875,
1.0320970913885732,
3.287281125183714,
2.5309548788864404,
2.820714319336463,
5.153358154745971,
1.0514557394368378,
0.9370347202687204,
4.0076157324919,
3.407794584431871,
0.9069650672102905,
3.1957201482771658,
1.855751233786371,
1.7816896808136653,
3.2893642153446696,
1.1102900259853528,
1.5305966750547555,
5.669289630045369,
2.0619516487367537,
2.964787207618063,
1.37439119863637,
1.4867873316830493,
2.615256304607202,
5.307339210348947,
2.0551270047297665,
2.913775111717478,
2.2854336229377803,
6.862146431500417,
1.1321098421075944,
3.8980643064935903,
1.3483004540015666,
1.8778521168502242,
2.7086127461385794,
2.849660131367168,
0.7502259308508177,
7.816854975491772,
0.8197169165531962,
0.637783187568734,
6.019992146736847,
1.0740364899145673,
1.2700700374569274,

2.9734034401985547,
3.0116147920135834,
6.815085550014233,
2.3117681592280888,
2.6167108650408415,
3.8605586262495906,
0.8376659220622726,
2.2318566446333223,
6.870033039426992,
2.1701328664295407,
2.260245533570669,
2.0675428885425866,
0.45343004068156095,
5.197545351800834,
2.050534429744812,
0.7906358101792909,
3.7898192571585114,
1.7345080962077137,
2.9310646591022587,
4.996994197565253,
1.69676275592188,
0.8182452803165237,
1.558254498123731,
1.9977641638140224,
1.8739592430616279,
5.648431080943058,
0.41208797613463466,
2.2216634205992745,
5.188630768374825,
1.2482518665688118,
2.7076963816300155,
2.15681425811478,
2.0354111961964683,
9.097765691338271,
0.7719995370784762,
1.6792879719299658,
0.7726437078664028,
6.601001273588086,
2.794748182640361,
0.11662473456151841,
7.817767956401142,
1.4989314914213383,
1.1376001974132999,
1.768631576406727,
3.7201449113142577,
6.106378475888705,
4.253132968700945,

4.672706670619601,
1.59605229666174,
2.4309190094703865,
6.173318526640938,
4.960576514428606,
3.152673004602956,
0.8871533865617451,
2.1021187803224657,
0.7016708392612302,
3.6039242909497586,
7.419214144493825,
1.7752639784781883,
2.892455314938483,
0.7123582388742966,
0.6329587547560238,
5.464886666552504,
1.7724901954681087,
11.868204958924887,
0.848050391473813,
5.387823809894834,
1.3814764234177208,
2.003784831804579,
2.8051963320067212,
1.2784706662883207,
1.6347419897288369,
1.1709104013792118,
4.172434360901016,
2.165961078746152,
0.7345049093557242,
1.676533378261942,
1.8541170093620831,
3.424185644245665,
0.09107703597241526,
2.6136025292117417,
0.19864745506307777,
0.3303543804390589,
1.8324676049519892,
7.97055524642739,
3.770414619529034,
1.3649227997441662,
2.282314718096799,
1.995323305096004,
1.8249615157515897,
1.0888024691132883,
2.831060645922549,
1.6703799542221531,
10.819659783124164,

1.9584552291998643,
3.418008322105889,
2.463244768079452,
4.8558516146941155,
1.650257559898209,
1.537444614708705,
0.2825599378027148,
0.5373337309360071,
3.3508800232590463,
1.301423904539013,
3.1399660251977752,
3.8763297477390313,
4.4109768947999335,
0.7719995370784762,
1.8870437474802155,
1.8754267410108403,
8.981676918718176,
1.795792427591927,
0.5878927920839351,
1.3369185040260436,
2.2472584882969744,
1.2986515625210666,
6.935330988463934,
3.1837289930871018,
2.557923445563547,
6.568760646608057,
2.0593281719808725,
1.7191102533721345,
4.83780843371231,
0.689300644079872,
0.6487597203396355,
2.4077790198451168,
3.4855635064245534,
4.286426008464891,
2.3513078928385456,
1.532139389138959,
0.7719995370784762,
1.1877918931639615,
4.322649423125438,
3.565825156790359,
0.5891569005623903,
1.714143172025093,
8.415111778225066,
1.6024941617278432,
2.27087121603153,
1.0936375681686001,
5.016570592882959,

5.017390315715445,
5.2168277536888015,
3.422279491799945,
3.876951546882687,
0.7719995370784762,
0.7240951406553017,
0.5204338393564141,
3.7533170584463833,
4.13885049561834,
0.6952572918176088,
2.3045712719846834,
1.3135581791415394,
1.5779749196009698,
6.144441005188224,
4.6888655751009995,
2.112291107300044,
8.238326728107612,
0.885665457677127,
3.8341329919278313,
10.845399257617478,
1.7309400715206562,
1.0125791809859863,
3.1809399052764062,
1.9477408522623545,
0.8459764927908993,
1.2687659135334046,
1.3414709430292162,
0.7719995370784762,
1.4588232891326043,
2.2895923323056895,
7.609385030250415,
0.92604591608269,
2.86880046589127,
2.0441538459888444,
0.16712845649335173,
3.692351849835081,
2.0193965043999618,
1.0099614279388553,
1.4917185101746704,
1.51978635774001,
3.6186879383790576,
4.915393538619769,
1.1679764088591194,
3.9509978615022536,
0.0,
1.0237567701848143,
1.3942824554541593,

0.8182888341956682,
1.2266922540603835,
0.6486966346700421,
2.0517326775865308,
4.465562222562157,
0.36798407759072044,
2.257552311699234,
1.2724626621587156,
1.5426496032860373,
2.7696182273659207,
3.0030659342015555,
0.5137198062988275,
0.47050099976415233,
1.152215578608276,
2.3595576810789183,
1.8238927179091515,
0.7301496430602897,
11.794648205867638,
1.1702807628426788,
4.278186898403274,
2.139020834850892,
1.9419779675245836,
3.2200924471559387,
3.716347437140229,
12.129538821325154,
10.40730839860501,
1.5781383782896836,
3.926686002760724,
1.082968680897182,
1.908434319748239,
0.9767591601490874,
6.312720052945961,
6.671440855060088,
2.1674998278671826,
5.358575939695272,
6.7979794503671735,
1.4984470715648295,
2.047379865183603,
2.014341745641009,
1.6805957719951525,
1.0114910223076548,
2.4469897836708903,
0.6288854558640983,
3.816324786242919,
3.860193932056304,
1.8677046164956665,
7.282039813063789,

4.7299648223310875,
1.1449848475485134,
4.94341894373907,
2.3853207884604952,
3.59905488314125,
0.8581643580961487,
0.9219224946595967,
4.859360909188374,
1.777950048066609,
1.6167585902738237,
4.373199866470946,
4.350333025395098,
7.627247373919188,
1.3296967160736641,
3.2128052234443634,
0.7369009671532577,
1.3935106503520962,
0.9556730444087008,
1.1105039204554548,
4.9497404432382055,
4.991862892928841,
0.6466676309960426,
4.657554246822394,
2.3135943540068773,
3.058593312489884,
2.8096621236304116,
3.4066170747753244,
2.613176167892551,
1.852589505893817,
1.841827555746606,
1.5784920027165998,
4.080073278378141,
1.3788036841230704,
1.7718559472541635,
4.324522024442271,
1.2998178822150275,
6.8878940388607175,
2.5144596357259785,
6.5560441387239505,
0.7719995370784762,
3.214164805099608,
0.7117751106482453,
2.4111707644656954,
0.7719995370784762,
2.6403859631565156,
3.0065595446660582,
1.2032987178516288,

3.794471155550668,
4.682163085196376,
1.0972841005012344,
1.428788072292393,
3.8393914713678123,
5.620734537791365,
9.913905565886196,
2.328867189977751,
3.1788468764468965,
1.864766514057273,
2.963557269614453,
3.414396911213883,
2.525972999216851,
3.396135340864232,
0.8089603221053214,
1.26263752157333,
5.295749734128347,
0.5333099584649932,
3.759930274437996,
6.311572517380114,
6.073379502958095,
6.773948621639868,
0.5598729272408635,
0.8575046049062067,
2.445397820328373,
2.819299882067107,
0.6631050632140973,
1.529175531905398,
2.8065552316422444,
2.415435767100913,
9.759335295039696,
1.6709866693739857,
2.3089477018006606,
5.069408912952638,
2.279424174203932,
2.175872183824103,
1.7822549639345722,
2.027691556820331,
4.935236863266085,
1.769930606439239,
3.5464445522995054,
1.930228556402269,
2.8967472892238733,
8.178663857407265,
1.7356454118294677,
3.179668880507512,
10.975501731564078,

3.3392491828727238,
2.802434054464456,
0.7780875857644323,
1.0555368055749053,
7.284584344684599,
1.1033684919098945,
1.330409545845683,
1.0945881283615069,
1.1278515221295327,
0.07312531440803731,
7.139796982642585,
1.5250910735289303,
2.132863748846976,
1.217169659775424,
4.8891917255868105,
2.0582005327690176,
1.5933552497708745,
1.5271127136282383,
1.4425099381741628,
2.9763896465671817,
1.9064433193421977,
2.435203045963132,
1.5220758462984687,
2.1145179355258907,
0.4245331575392342,
1.284338033040748,
6.547689760188495,
3.9565476409007556,
0.45435419424602635,
3.182334385545619,
4.121121331188664,
1.182727946057354,
0.904183465628801,
7.065031578957336,
3.4830842837589238,
2.207251298003133,
4.7724404291016675,
11.345043930440726,
10.996544776212774,
2.666080402801578,
9.53187703033514,
5.55314298116919,
1.0131147993453922,
1.2246396471765753,
1.4105469835413413,
1.782242992232637,
1.887976791816379,

```
2.677739679886167,
5.532827433154519,
1.2364921515287135,
0.7549119616813444,
0.2017058942707934,
7.9865945024153575,
3.376204629461158,
6.5406716356446735,
2.426261251065555,
4.94516562845086,
2.494627957066291,
1.4965143701203412,
2.311715660284443,
1.501226980843212,
2.866730457175025,
0.7719995370784762,
1.303413803403511,
0.6665943697135664,
2.699298171593384,
4.6038445164375705,
2.0045432916833734,
2.123431317796762,
2.0471557363265602,
0.6356044232156017,
6.481609505674108,
3.3328148795753636,
2.053888426902192,
7.8876717616702585,
1.9971729869288573,
0.3971168333138258,
1.8780531712593953,
0.7254565419178712,
2.0819031925372244,
3.550899328485873,
1.8263187880684726,
2.296210737760639,
4.874023545192515,
7.688550309957377,
0.5108359239272235,
4.588853209670556,
3.5783986884586465,
3.436540345098955,
0.9312765398652313,
0.6826715332999583,
1.70332048812373,
4.8486555695047695,
1.5513312701575195,
```

```
0.6974908980076775,
1.8224928357945687,
2.8553297299716744,
0.4931170059278314,
11.048627280431589,
1.2713293208229537,
3.331561741870138,
3.9625167343368672,
1.5497326059200096,
3.321773821369834,
2.2364969655721434,
3.809148114825199,
3.0454453670685764,
3.4492974348802976,
2.0277384242189176,
2.9290610134344797,
2.3443416255896694,
1.8585713607209204,
0.6093896212130627,
0.8047490492658376,
3.304256051582717,
0.9185079416078408,
3.5363177950907168,
3.0186548581069483,
1.2879034431578669,
0.31016035126772595,
9.589430316816125,
2.839134060354311,
0.0,
4.839891625106934,
6.884989589419282,
1.6212933234712446,
1.3404013731030586,
3.075714506482144,
4.067888171759627,
5.429864131246641,
2.479076237927193,
9.859757843462774,
1.8827128326093123,
8.563789267552464,
2.757322651253192,
2.770005031649036,
7.326873726213767,
1.0242935350137097,
1.0322114367941204,
1.3859234785861263,
0.6896975406083913,
```

1.125851875869708,
1.9641104557749733,
1.1436945655908968,
0.7719995370784762,
1.350391816480814,
6.3601580530711574,
1.033316530682402,
5.578543275221338,
5.774863362904877,
2.090677676851702,
5.602092869892499,
4.426859137077277,
2.832460756471197,
2.0247851919397766,
4.0752079081654164,
1.6933716363352969,
1.7685964798353426,
3.2966683584597067,
0.6330598672139752,
0.6128430173625699,
0.9372042554319255,
5.417997173496873,
6.406606504026187,
2.335206712502547,
6.689555776387006,
5.413129768788556,
2.2314265324856306,
1.3649818172237111,
2.813700460878114,
2.262515664750233,
1.5414797447137452,
5.6046793110415285,
1.5229000197903686,
6.097240264921754,
0.7473281639199638,
5.014757908337086,
4.564305581115082,
2.8945506181928304,
2.131555536480811,
0.9856030297032413,
0.40503073577719884,
1.8108124245764523,
2.987086952763552,
3.32984254393489,
0.772834881151759,
0.5724932652355229,
2.731224317059544,

5.086062626835162,
1.7182794768417162,
7.80644373034564,
6.354719284049434,
1.4245655213651893,
1.6023858287615176,
2.2352804497202703,
2.9886067281847546,
2.1394393129097398,
3.530043089623396,
1.7618994146890958,
0.9090491377711108,
1.3853815603152442,
4.586366881261418,
2.1023665139757393,
4.534256429097138,
1.3904209472479547,
0.7571467452313818,
5.136243742344894,
4.938556865065128,
3.6287690778314197,
0.29093105264477015,
1.8403084964090806,
1.824288691023579,
4.459290972461384,
2.0840566460800054,
1.5748103638724262,
2.2668017020507274,
2.125461484269435,
2.4428989788380493,
2.13954990258659,
2.712606747689459,
1.7376771124581445,
2.4503482806172467,
1.9358712394820639,
2.053116325168115,
0.7719995370784762,
0.6816597044927067,
3.265221205768907,
1.433951899799339,
1.2418717726238544,
2.2232635821474296,
2.74312403285971,
1.396495653062777,
0.7533563429767167,
1.918366338563976,
3.276090734545401,

1.0220981275353787,
0.8028731813128359,
4.10441547118317,
1.1815959751383016,
1.75364153968604,
0.7402785252804821,
2.4152364904704666,
0.958111361314147,
1.3539310414657006,
3.2411526012417027,
4.66093964473386,
0.6784269362697066,
7.066827837218263,
1.2323472113721292,
1.5729504646537178,
0.00068810967834488 81,
1.0194462591538884,
0.2673044885838518,
1.4923319965749529,
1.7413080398998093,
1.4677377563732839,
2.114199690018914,
6.300863261137619,
5.398460357329391,
4.857843833938755,
4.410081739428757,
1.984787243886089,
1.1980500351934138,
7.397915631399255,
5.046197411642443,
0.9298059679606306,
4.708010402865085,
0.36597878731840056,
3.8098111050268852,
8.715214331333861,
4.755259017839335,
2.101033210408425,
1.7796155439711778,
1.2416116683275618,
1.9655005971330346,
0.044975441822248746,
0.68184564424943,
1.7351825105424725,
0.7426908936122262,
4.94680694085774,
1.2615669163699652,
2.3497105174371966,

7.26529266859538,
5.460809036483912,
6.818246737854603,
3.6356623107458077,
2.3763138233998538,
1.2398187111762753,
7.507305415911398,
3.8766178679922474,
0.38987470455768436,
1.5771435499248874,
1.4710299431608118,
2.2043613706362843,
4.014265396642929,
2.444316628523308,
0.0011469893917261562,
2.162838141820863,
4.234554310158971,
1.6649876952639204,
3.209033024155447,
6.1923121478466925,
4.736899083648394,
1.7046039223136809,
2.3395153521883296,
1.153296505750822,
10.225242339322417,
1.9894121213893767,
3.512904476068075,
3.792825880046498,
2.956198730866217,
2.229959454641979,
2.487535423486973,
8.469000313782255,
1.7229164226352438,
0.3499243531594926,
5.0716600856411285,
4.088464938992787,
5.908508133967225,
2.9518125984276957,
1.9424496727134426,
1.1476506080296858,
1.4214939464842302,
3.24626798590751,
1.4854335767232132,
0.7914244945415305,
8.823221045719995,
0.7748849127532341,
0.574224053339419,

1.5572587259249049,
5.417481197925349,
1.9910491217540927,
0.8163556242359274,
2.735692293927604,
2.1836766349334704,
1.6379171357573095,
6.610539990722355,
0.84176399432555,
3.8236706667296887,
7.131694041260707,
3.0733435903123745,
0.8890510578085677,
4.271729805094748,
4.721744721434179,
2.663716247771879,
1.0980991988116056,
3.8046520760400804,
2.4475734097926733,
3.689529829970618,
1.341043427245331,
0.6398617286820958,
1.5215388206847715,
2.5993959286001367,
2.3379833439380504,
2.27566569572684,
9.03067787232202,
0.6661168931067618,
2.2536868002308568,
1.02463767282862,
3.1742910627896994,
1.344639742223829,
4.493648352494523,
3.8806438419065334,
2.7676433882404505,
1.7570889810476737,
2.9112630931231327,
1.1944836627488047,
1.9954416277497475,
1.3414797805465732,
3.0115765585384358,
5.119835556459173,
1.3890144546952996,
1.971155434455477,
7.360919160254177,
8.168851795768553,
1.8998668113230914,

```
 2.7330975056713473,
 0.3046672682119567,
 2.3681857721120796,
 7.736758887641531,
 5.5952430670051205,
 2.3364752879742516,
 0.7200963018558747,
 4.356159432256988,
 1.7001217778094833,
 2.368860127883199,
 2.568767296328695,
 1.1259514789242318,
 3.2279354198279515,
 …]
```

[ ]:
```
#Uber doesn't travel over 130 kms so minimize the distance
df= df.loc[(df.dist_travel_km >= 1) | (df.dist_travel_km <= 130)]
print("Remaining observastions in the dataset:", df.shape)
```

```
Remaining observastions in the dataset: (200000, 12)
```

[ ]:
```
#Finding inccorect latitude (Less than or greater than 90) and longitude␣
 ↪(greater than or less than 180)
incorrect_coordinates = df.loc[(df.pickup_latitude > 90) |(df.pickup_latitude <␣
 ↪-90) |
                               (df.dropoff_latitude > 90) |(df.
 ↪dropoff_latitude < -90) |
                               (df.pickup_longitude > 180) |(df.
 ↪pickup_longitude < -180) |
                               (df.dropoff_longitude > 180) |(df.
 ↪dropoff_longitude < -180)
                                ]
```

[ ]:
```
incorrect_coordinates
```

[ ]:
```
Empty DataFrame
Columns: [fare_amount, pickup_longitude, pickup_latitude, dropoff_longitude,
dropoff_latitude, passenger_count, hour, day, month, year, dayofweek,
dist_travel_km]
Index: []
```

[ ]:
```
df.drop(incorrect_coordinates, inplace = True, errors = 'ignore')
```

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

[ ]:
```
   fare_amount  pickup_longitude  pickup_latitude  dropoff_longitude  \
0          7.5        -73.999817        40.738354         -73.999512
```

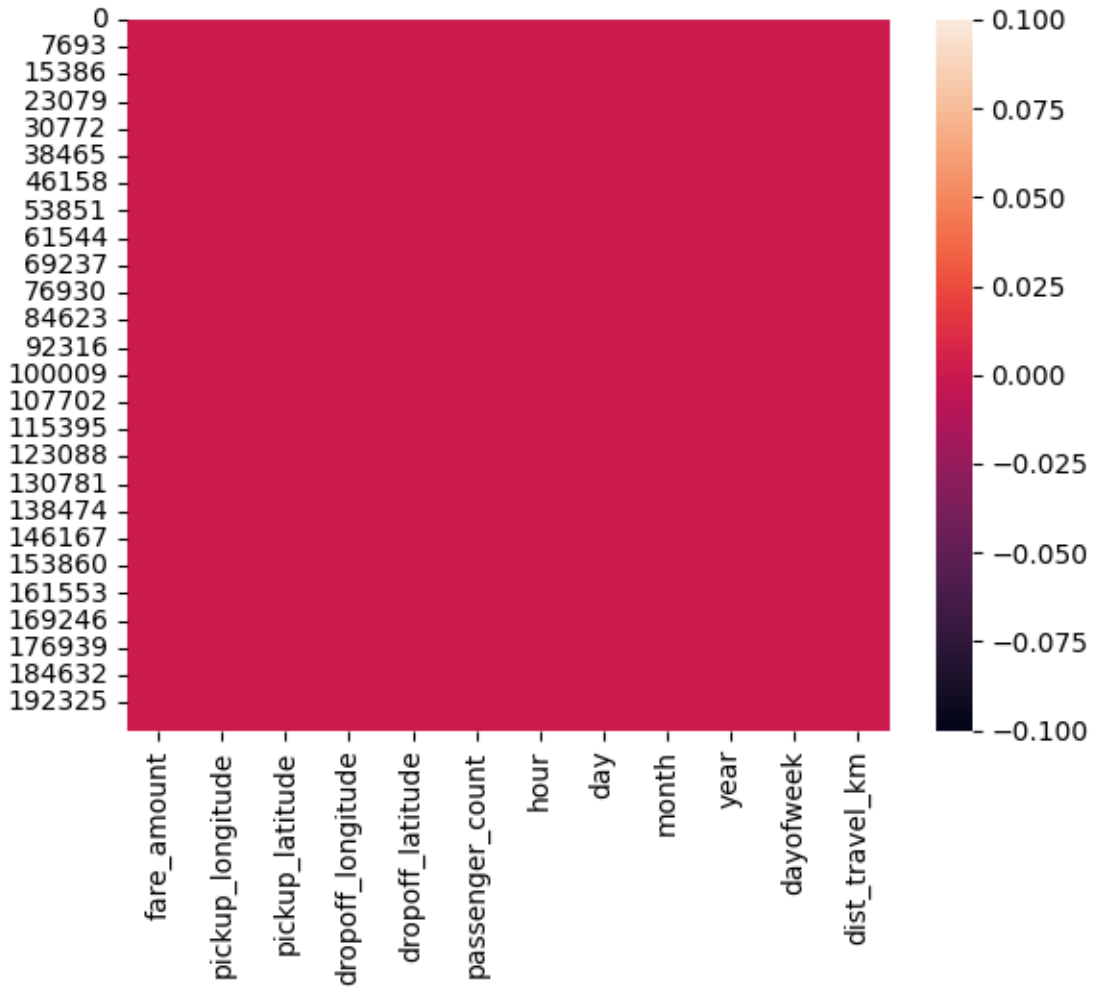
```
1          7.7        -73.994355         40.728225         -73.994710
2         12.9        -74.005043         40.740770         -73.962565
3          5.3        -73.976124         40.790844         -73.965316
4         16.0        -73.929786         40.744085         -73.973082

   dropoff_latitude  passenger_count  hour  day  month  year  dayofweek  \
0         40.723217              1.0    19    7      5  2015          3
1         40.750325              1.0    20   17      7  2009          4
2         40.772647              1.0    21   24      8  2009          0
3         40.803349              3.0     8   26      6  2009          4
4         40.761247              3.5    17   28      8  2014          3

   dist_travel_km
0        1.683325
1        2.457593
2        5.036384
3        1.661686
4        4.116088
```

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

[ ]:
```
fare_amount          0
pickup_longitude     0
pickup_latitude      0
dropoff_longitude    0
dropoff_latitude     0
passenger_count      0
hour                 0
day                  0
month                0
year                 0
dayofweek            0
dist_travel_km       0
dtype: int64
```

[ ]:
```
sns.heatmap(df.isnull()) #Free for null values
```

[ ]:
```
<Axes: >
```



[ ]:
```
corr = df.corr() #Function to find the correlation
```

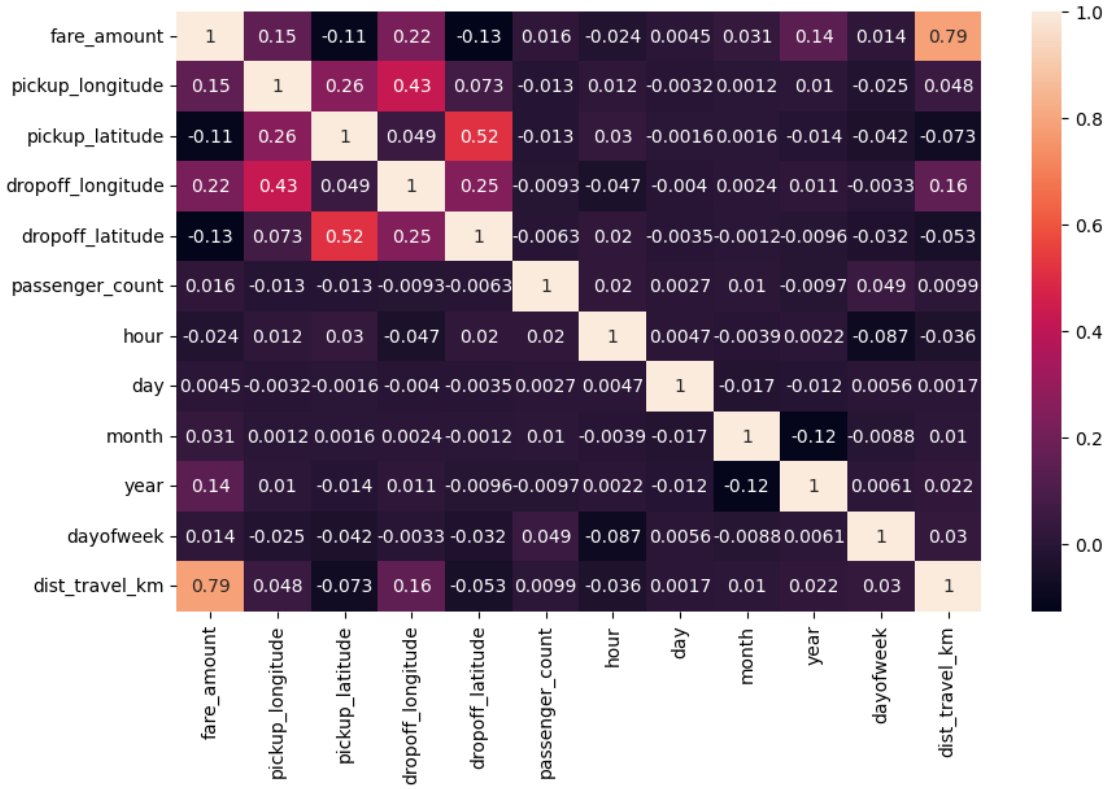
[ ]:
```
corr
```

[ ]:
```
                   fare_amount  pickup_longitude  pickup_latitude  \
fare_amount           1.000000          0.154069        -0.110842
pickup_longitude      0.154069          1.000000         0.259497
pickup_latitude      -0.110842          0.259497         1.000000
dropoff_longitude     0.218675          0.425619         0.048889
dropoff_latitude     -0.125898          0.073290         0.515714
passenger_count       0.015778         -0.013213        -0.012889
hour                 -0.023623          0.011579         0.029681
day                   0.004534         -0.003204        -0.001553
month                 0.030817          0.001169         0.001562
year                  0.141277          0.010198        -0.014243
dayofweek             0.013652         -0.024652        -0.042310
```

```
dist_travel_km        0.786385          0.048446        -0.073362

                   dropoff_longitude  dropoff_latitude  passenger_count  \
fare_amount                 0.218675         -0.125898         0.015778
pickup_longitude            0.425619          0.073290        -0.013213
pickup_latitude             0.048889          0.515714        -0.012889
dropoff_longitude           1.000000          0.245667        -0.009303
dropoff_latitude            0.245667          1.000000        -0.006308
passenger_count            -0.009303         -0.006308         1.000000
hour                       -0.046558          0.019783         0.020274
day                        -0.004007         -0.003479         0.002712
month                       0.002391         -0.001193         0.010351
year                        0.011346         -0.009603        -0.009749
dayofweek                  -0.003336         -0.031919         0.048550
dist_travel_km              0.155191         -0.052701         0.009884

                        hour       day     month      year  dayofweek  \
fare_amount       -0.023623  0.004534  0.030817  0.141277   0.013652
pickup_longitude   0.011579 -0.003204  0.001169  0.010198  -0.024652
pickup_latitude    0.029681 -0.001553  0.001562 -0.014243  -0.042310
dropoff_longitude -0.046558 -0.004007  0.002391  0.011346  -0.003336
dropoff_latitude   0.019783 -0.003479 -0.001193 -0.009603  -0.031919
passenger_count    0.020274  0.002712  0.010351 -0.009749   0.048550
hour               1.000000  0.004677 -0.003926  0.002156  -0.086947
day                0.004677  1.000000 -0.017360 -0.012170   0.005617
month             -0.003926 -0.017360  1.000000 -0.115859  -0.008786
year               0.002156 -0.012170 -0.115859  1.000000   0.006113
dayofweek         -0.086947  0.005617 -0.008786  0.006113   1.000000
dist_travel_km    -0.035708  0.001709  0.010050  0.022294   0.030382

                   dist_travel_km
fare_amount              0.786385
pickup_longitude         0.048446
pickup_latitude         -0.073362
dropoff_longitude        0.155191
dropoff_latitude        -0.052701
passenger_count          0.009884
hour                    -0.035708
day                      0.001709
month                    0.010050
year                     0.022294
dayofweek                0.030382
dist_travel_km           1.000000
```

[ ]:
```
fig,axis = plt.subplots(figsize = (10,6))
sns.heatmap(df.corr(),annot = True) #Correlation Heatmap (Light values means␣
 ↪highly correlated)
```

[ ]: <Axes: >



link text### Dividing the dataset into feature and target values

[ ]:
```
x =␣
↪df[['pickup_longitude','pickup_latitude','dropoff_longitude','dropoff_latitude','passenger_
```

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

1.2.1 Dividing the dataset into training and testing dataset

[ ]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(x,y,test_size = 0.20)
```

[ ]:
```
X_train
```

[ ]:

	pickup_longitude	pickup_latitude	dropoff_longitude	\
157695	-73.989162	40.774402	-73.963910	
138060	-73.976410	40.763250	-73.974850	
154745	-73.979125	40.767502	-73.975640	
46559	-73.949280	40.781178	-73.995903	
46288	-73.968557	40.761720	-73.980340	

```
…                …              …                …
86479        -73.988541       40.737064        -74.001640
175709       -73.961227       40.770820        -73.965755
90215        -73.977798       40.789196        -73.994744
177725       -74.006920       40.729010        -73.952205
171840       -74.002592       40.747009        -73.986591

        dropoff_latitude  passenger_count  hour  day  month  year  dayofweek  \
157695          40.806314              1.0    20    1      4  2010          3
138060          40.753368              1.0    15   25      6  2011          5
154745          40.781337              1.0    12    8     12  2012          5
46559           40.723438              3.5    15   14      3  2015          5
46288           40.740283              1.0    20   23      7  2012          0
…                     …                …    …  …    …    …         …
86479           40.726208              3.5    12   10      1  2015          5
175709          40.764065              3.0    12    4      1  2009          6
90215           40.750306              1.0     8   16      3  2013          5
177725          40.784377              1.0     0   20     11  2013          2
171840          40.740498              1.0    13   30     12  2013          0

        dist_travel_km
157695        4.136531
138060        1.106658
154745        1.566124
46559         7.526228
46288         2.582080
…                …
86479         1.635753
175709        0.842375
90215         4.553762
177725        7.690388
171840        1.530130

[160000 rows x 11 columns]
```

[ ]:
```
X_test
```

[ ]:
```
        pickup_longitude  pickup_latitude  dropoff_longitude  \
27952         -73.978792        40.736315         -73.975704
193032        -74.004529        40.753718         -74.004529
171813        -73.969090        40.797722         -73.963737
102534        -73.963525        40.711047         -73.975850
164693        -73.981447        40.741292         -73.998290
…                    …                …                …
100480        -73.995567        40.726507         -73.981867
179674        -73.986132        40.760803         -73.955890
179836        -73.985319        40.727517         -73.972175
```

```
16945         -73.929786        40.768318         -73.931804
143499        -74.002419        40.733700         -73.999237

        dropoff_latitude  passenger_count  hour  day  month  year  dayofweek  \
27952           40.753726              1.0    13    2      4  2014          2
193032          40.753718              1.0    18   24     11  2010          2
171813          40.776917              1.0    18    7     10  2014          1
102534          40.781695              2.0     4   10      3  2013          6
164693          40.754300              2.0    12    2      1  2014          3
…                    …                …    …  …     …    …        …
100480          40.729350              3.5    12   29      5  2013          2
179674          40.764217              1.0    17   23      3  2010          1
179836          40.747370              1.0    16    4      1  2014          5
16945           40.766520              1.0     3   15     12  2012          5
143499          40.744091              3.5     0    1      1  2015          3

        dist_travel_km
27952         1.953417
193032        0.000000
171813        2.356902
102534        7.924028
164693        2.026156
…                …
100480        1.196938
179674        2.575166
179836        2.469756
16945         0.262386
143499        1.186137

[40000 rows x 11 columns]
```

[ ]:
```
y_train
```

[ ]:
```
157695     9.70
138060     5.30
154745     8.00
46559     22.25
46288      6.50
          …
86479     10.00
175709     4.10
90215     12.50
177725    21.50
171840     6.00
Name: fare_amount, Length: 160000, dtype: float64
```

[ ]:
```
y_test
```

[ ]:
```
27952      9.50
193032     5.70
171813    10.50
102534    22.25
164693     8.00
          …
100480     6.50
179674     9.30
179836     7.00
16945      5.50
143499     5.50
Name: fare_amount, Length: 40000, dtype: float64
```

1.2.2 Linear Regression

[ ]:
```
from sklearn.linear_model import LinearRegression
regression = LinearRegression()
```

[ ]:
```
regression.fit(X_train,y_train)
```

[ ]:
```
LinearRegression()
```

[ ]:
```
regression.intercept_ #To find the linear intercept
```

[ ]:
```
3683.630887451561
```

[ ]:
```
regression.coef_ #To find the linear coeeficient
```

[ ]:
```
array([ 2.54340686e+01, -7.57923589e+00,  2.02667021e+01, -1.80328583e+01,
        7.19258624e-02,  6.40579214e-03,  3.69636670e-03,  5.89888832e-02,
        3.70474390e-01, -3.44173390e-02,  1.84433879e+00])
```

[ ]:
```
prediction = regression.predict(X_test) #To predict the target values
```

[ ]:
```
print(prediction)
```

```
[ 9.21725408  3.28832288 10.00595244 … 10.00276185  7.37297707
  7.16977415]
```

[ ]:
```
y_test
```

[ ]:
```
27952      9.50
193032     5.70
171813    10.50
102534    22.25
164693     8.00
          …
```

```
100480     6.50
179674     9.30
179836     7.00
16945      5.50
143499     5.50
Name: fare_amount, Length: 40000, dtype: float64
```

1.2.3 Metrics Evaluation using R2, Mean Squared Error, Root Mean Sqared Error

[ ]:
```
from sklearn.metrics import r2_score
```

[ ]:
```
r2_score(y_test,prediction)
```

[ ]:
```
0.664995440543817
```

[ ]:
```
from sklearn.metrics import mean_squared_error
```

[ ]:
```
MSE = mean_squared_error(y_test,prediction)
```

[ ]:
```
MSE
```

[ ]:
```
9.830495516036072
```

[ ]:
```
RMSE = np.sqrt(MSE)
```

[ ]:
```
RMSE
```

[ ]:
```
3.1631357043785315
```

1.2.4 Random Forest Regression

[ ]:
```
from sklearn.ensemble import RandomForestRegressor
```

[ ]:
```
rf = RandomForestRegressor(n_estimators=50) #Here n_estimators means number of␣
↪trees you want to build before making the prediction
```

[ ]:
```
rf.fit(X_train,y_train)
```

[ ]:
```
RandomForestRegressor(n_estimators=50)
```

[ ]:
```
y_pred = rf.predict(X_test)
```

[ ]:
```
y_pred
```

[ ]:
```
array([10.155,  8.863, 10.03 , …,  9.75 ,  6.516,  6.485])
```

1.2.5 Metrics evaluatin for Random Forest

[ ]:
```
R2_Random = r2_score(y_test,y_pred)
```

[ ]:
```
R2_Random
```

[ ]:
```
0.8000142586249277
```

[ ]:
```
MSE_Random = mean_squared_error(y_test,y_pred)
```

[ ]:
```
MSE_Random
```

[ ]:
```
5.868454259399225
```

[ ]:
```
RMSE_Random = np.sqrt(MSE_Random)
```

[ ]:
```
RMSE_Random
```

[ ]:
```
2.460259594138213
```