ML1

July 18, 2024

[1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
import warnings
warnings.filterwarnings('ignore')
```

[2]:

```
df=pd.read_csv('uber.csv')
```

[3]:

```
df.head()
```

[3]:

```
   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
```

[4]:

```
df.describe()
```

[4]:

```
         Unnamed: 0    fare_amount  pickup_longitude  pickup_latitude  \
count  2.000000e+05  200000.000000     200000.000000    200000.000000
mean   2.771250e+07      11.359955        -72.527638        39.935885
```

```
std    1.601382e+07       9.901776         11.437787         7.720539
min    1.000000e+00     -52.000000      -1340.648410       -74.015515
25%    1.382535e+07       6.000000        -73.992065        40.734796
50%    2.774550e+07       8.500000        -73.981823        40.752592
75%    4.155530e+07      12.500000        -73.967154        40.767158
max    5.542357e+07     499.000000         57.418457      1644.421482

       dropoff_longitude  dropoff_latitude  passenger_count
count      199999.000000     199999.000000    200000.000000
mean          -72.525292         39.923890         1.684535
std            13.117408          6.794829         1.385997
min         -3356.666300       -881.985513         0.000000
25%           -73.991407         40.733823         1.000000
50%           -73.980093         40.753042         1.000000
75%           -73.963658         40.768001         2.000000
max          1153.572603        872.697628       208.000000
```

[5]:

```
df.shape
```

[5]:

```
(200000, 9)
```

[6]:

```
df.dtypes
```

[6]:

```
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
```

[7]:

```
df=df.drop(['Unnamed: 0', 'key'], axis=1)
```

[8]:

```
df.columns
```

[8]:

```
Index(['fare_amount', 'pickup_datetime', 'pickup_longitude', 'pickup_latitude',
       'dropoff_longitude', 'dropoff_latitude', 'passenger_count'],
      dtype='object')
```

[9]:

```
df.pickup_datetime=pd.to_datetime(df.pickup_datetime)
```

[10]:

```
df.dtypes
```

[10]:

```
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
```

[11]:
```
df.isnull().sum()
```

[11]:
```
fare_amount          0
pickup_datetime      0
pickup_longitude     0
pickup_latitude      0
dropoff_longitude    1
dropoff_latitude     1
passenger_count      0
dtype: int64
```

[12]:
```
df['dropoff_longitude'].fillna(value=df['dropoff_longitude'].mean(),inplace=True)
df['dropoff_latitude'].fillna(value=df['dropoff_latitude'].mean(),inplace=True)
```

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

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

[14]:
```
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)
```

[15]:
```
df = df.drop(["pickup_datetime"], axis =1)
df
```

[15]:
```
        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
...             ...               ...              ...                ...
```

```
199995         3.0        -73.987042        40.739367        -73.986525
199996         7.5        -73.984722        40.736837        -74.006672
199997        30.9        -73.986017        40.756487        -73.858957
199998        14.5        -73.997124        40.725452        -73.983215
199999        14.1        -73.984395        40.720077        -73.985508

        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
...                  ...              ...   ...  ...    ...   ...        ...
199995         40.740297                1    10   28     10  2012          6
199996         40.739620                1     1   14      3  2014          4
199997         40.692588                2     0   29      6  2009          0
199998         40.695415                1    14   20      5  2015          2
199999         40.768793                1     4   15      5  2010          5

[200000 rows x 11 columns]
```

[16]:

```
from math import *

def distance_formula(longitude1, latitude1, longitude2, latitude2):
    travel_dist = []

    for pos in range (len(longitude1)):
        lon1, lan1, lon2, lan2 = map(radians, [longitude1[pos], latitude1[pos],␣
 ↪longitude2[pos], latitude2[pos]])
        dist_lon = lon2 - lon1
        dist_lan = lan2 - lan1

        a = sin(dist_lan/2)**2 + cos(lan1) * cos(lan2) * sin(dist_lon/2)**2

        c = 2 * asin(sqrt(a)) * 6371
        travel_dist.append(c)

    return  travel_dist
```

[17]:

```
df['dist_travel_km'] = distance_formula(df.pickup_longitude.to_numpy(), df.
 ↪pickup_latitude.to_numpy(), df.dropoff_longitude.to_numpy(), df.
 ↪dropoff_latitude.to_numpy())
```

[18]:

```
def remove_outlier(df1 , col):
    Q1 = df1[col].quantile(0.25)
    Q3 = df1[col].quantile(0.75)
    IQR = Q3 - Q1
    lower = Q1-1.5*IQR
    upper= Q3+1.5*IQR
    df[col] = np.clip(df1[col] , lower , upper)
    return df1

def remove_all(df1 , col_list):
    for i in col_list:
        df1 = remove_outlier(df , i)
    return df1
```

[19]:

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

[20]:

```
df_x =␣
 ↪df[['pickup_longitude','pickup_latitude','dropoff_longitude','dropoff_latitude','passenger_co
df_y = df['fare_amount']
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
scaler.fit_transform(df_x)
```

[20]:

```
array([[-1.04868902, -0.46151906, -1.01920818, ...,  1.75477984,
        -0.02487235, -0.52650782],
       [-0.78236166, -0.85626623, -0.80934206, ..., -1.4772954 ,
         0.48875385, -0.17833351],
       [-1.30351906, -0.3673497 ,  0.59560201, ..., -1.4772954 ,
        -1.56575094,  0.98130178],
       ...,
       [-0.37579257,  0.24518925,  2.36704659, ..., -1.4772954 ,
        -1.56575094,  2.25341213],
       [-0.91736709, -0.96432192, -0.30695085, ...,  1.75477984,
        -0.53849854,  0.30827932],
       [-0.29670226, -1.17381839, -0.40715524, ..., -0.9386162 ,
         1.00238004,  1.15281348]])
```

[21]:

```
x_train, x_test, y_train, y_test = train_test_split(df_x, df_y, test_size=0.2,␣
 ↪random_state=1)
```

[22]:

```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score, mean_squared_error
```

[23]:

```
reg = LinearRegression()
reg.fit(x_train, y_train)
```

[23]:

```
LinearRegression()
```

[24]:

```
y_pred_lin = reg.predict(x_test)
print(y_pred_lin)
```

```
[ 6.2761843   5.09988121  9.43640959 ... 11.07661434 12.15390374
 11.41498106]
```

[25]:

```
rmse_lr = np.sqrt(mean_squared_error(y_test, y_pred_lin))
r2_lr = r2_score(y_test, y_pred_lin)

print("Linear Regression - RMSE:", rmse_lr)
print("Linear Regression - R2 score:", r2_lr)
```

```
Linear Regression - RMSE: 2.7039561758284734
Linear Regression - R2 score: 0.75390636301046
```

[26]:

```
from sklearn.ensemble import RandomForestRegressor
```

[27]:

```
rf = RandomForestRegressor(n_estimators=100)
rf.fit(x_train,y_train)
```

[27]:

```
RandomForestRegressor()
```

[28]:

```
RandomForestRegressor()
```

[28]:

```
RandomForestRegressor()
```

[29]:

```
y_pred_rf = rf.predict(x_test)
print(y_pred_rf)
```

```
[ 4.999   6.674   9.2175 ... 11.975  11.012  13.429 ]
```

[30]:

```
rmse_rf= np.sqrt(mean_squared_error(y_test, y_pred_rf))
r2_rf = r2_score(y_test, y_pred_rf)

print("Random Forest - RMSE:", rmse_rf)
print("Random Forest - R2 score:", r2_rf)
```

```
Random Forest - RMSE: 2.364547091246532
Random Forest - R2 score: 0.8118097917070926
```

[ ]:

```

```