wqzwo0dec

July 31, 2023

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

[ ]:
```
df=pd.read_csv("/content/7_uber.csv")[0:500]
df
```

[ ]:
```
     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
..          …                             …          …
495     1204312    2012-06-03 12:18:02.0000001         25.7
496     2511529    2014-12-24 05:54:45.0000001          8.0
497    24116460    2010-01-18 02:18:16.0000001         10.5
498    42607669    2015-03-30 10:58:37.0000001          5.5
499    36533403    2015-03-09 16:16:21.0000006         10.0

              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
..                       …                 …                …
495  2012-06-03 12:18:02 UTC        -73.862765        40.770908
496  2014-12-24 05:54:45 UTC        -73.918530        40.743330
497  2010-01-18 02:18:16 UTC        -74.005734        40.743641
498  2015-03-30 10:58:37 UTC        -74.001648        40.740940
499  2015-03-09 16:16:21 UTC        -73.960037        40.780624

     dropoff_longitude  dropoff_latitude  passenger_count
0           -73.999512         40.723217              1.0
1           -73.994710         40.750325              1.0
```

```
2           -73.962565         40.772647              1.0
3           -73.965316         40.803349              3.0
4           -73.973082         40.761247              5.0
..                 …                 …              …
495         -73.989013         40.688776              1.0
496         -73.946696         40.749438              1.0
497         -74.006287         40.708330              2.0
498         -74.005730         40.750175              1.0
499         -73.971756         40.765934              1.0

[500 rows x 9 columns]
```

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

[ ]:
```
   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              1.0
1         -73.994710         40.750325              1.0
2         -73.962565         40.772647              1.0
3         -73.965316         40.803349              3.0
4         -73.973082         40.761247              5.0
```

1 DATA CLEANING AND DATA PREPROCESSING

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 9 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Unnamed: 0         500 non-null    int64
 1   key                500 non-null    object
 2   fare_amount        500 non-null    float64
```

```
 3   pickup_datetime    500 non-null    object
 4   pickup_longitude   500 non-null    float64
 5   pickup_latitude    500 non-null    float64
 6   dropoff_longitude  500 non-null    float64
 7   dropoff_latitude   500 non-null    float64
 8   passenger_count    500 non-null    float64
dtypes: float64(6), int64(1), object(2)
memory usage: 35.3+ KB
```

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

[ ]:
```
         Unnamed: 0  fare_amount  pickup_longitude  pickup_latitude  \
count  5.000000e+02   500.000000        500.000000       500.000000
mean   2.737940e+07    10.708720        -72.053865        39.692497
std    1.607155e+07     8.334145         11.784239         6.491541
min    1.862090e+05     2.500000        -74.030417         0.000000
25%    1.250293e+07     6.000000        -73.992804        40.735994
50%    2.749836e+07     8.100000        -73.982352        40.752445
75%    4.157492e+07    12.500000        -73.968724        40.765865
max    5.519870e+07    57.330000          0.001782        40.850558

       dropoff_longitude  dropoff_latitude  passenger_count
count         500.000000        500.000000       500.000000
mean          -72.201155         39.772818         1.664000
std            11.333432          6.243123         1.267405
min           -74.027813          0.000000         0.000000
25%           -73.991571         40.730869         1.000000
50%           -73.980784         40.750428         1.000000
75%           -73.965878         40.767497         2.000000
max             0.000875         40.901391         6.000000
```

[ ]:
```
df.columns
```

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

[ ]:
```
df1=df.dropna(axis=1)
df1
```

[ ]:
```
     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
```

```
..          …                              …          …
495     1204312    2012-06-03 12:18:02.0000001         25.7
496     2511529    2014-12-24 05:54:45.0000001          8.0
497    24116460    2010-01-18 02:18:16.0000001         10.5
498    42607669    2015-03-30 10:58:37.0000001          5.5
499    36533403    2015-03-09 16:16:21.0000006         10.0

              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
..                       …               …               …
495  2012-06-03 12:18:02 UTC        -73.862765        40.770908
496  2014-12-24 05:54:45 UTC        -73.918530        40.743330
497  2010-01-18 02:18:16 UTC        -74.005734        40.743641
498  2015-03-30 10:58:37 UTC        -74.001648        40.740940
499  2015-03-09 16:16:21 UTC        -73.960037        40.780624

     dropoff_longitude  dropoff_latitude  passenger_count
0           -73.999512         40.723217              1.0
1           -73.994710         40.750325              1.0
2           -73.962565         40.772647              1.0
3           -73.965316         40.803349              3.0
4           -73.973082         40.761247              5.0
..                 …                …              …
495         -73.989013         40.688776              1.0
496         -73.946696         40.749438              1.0
497         -74.006287         40.708330              2.0
498         -74.005730         40.750175              1.0
499         -73.971756         40.765934              1.0

[500 rows x 9 columns]
```

[ ]:
```
df1.columns
```

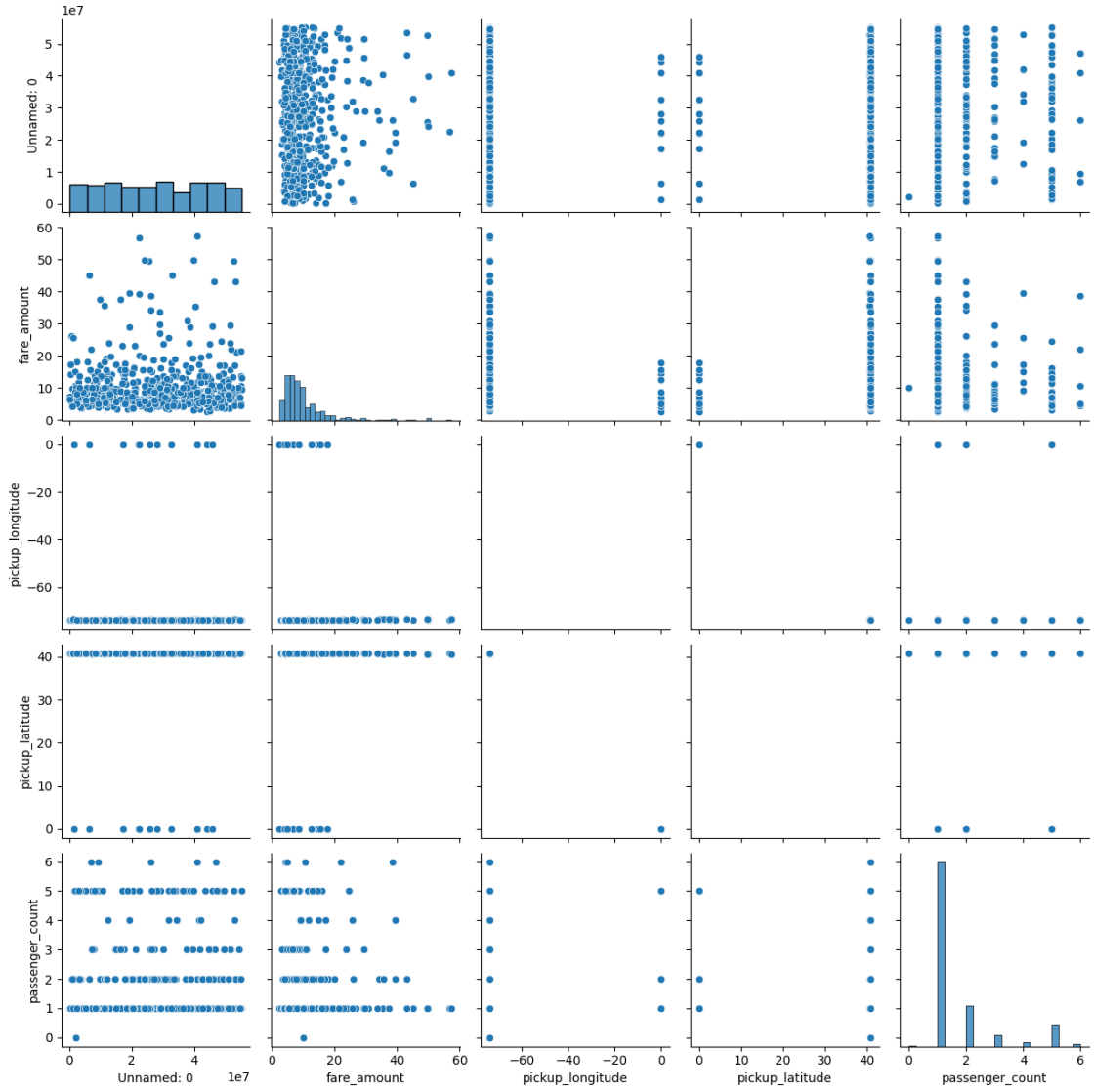
[ ]:
```
Index(['Unnamed: 0', 'key', 'fare_amount', 'pickup_datetime',
       'pickup_longitude', 'pickup_latitude', 'dropoff_longitude',
       'dropoff_latitude', 'passenger_count'],
      dtype='object')
```

[ ]:
```
df1=df1[['Unnamed: 0', 'fare_amount',
       'pickup_longitude', 'pickup_latitude', 'passenger_count']]
```

2 EDA AND VISUALIZATION

[ ]:
```
sns.pairplot(df1)
```

[ ]:
```
<seaborn.axisgrid.PairGrid at 0x7f8e89bb4df0>
```

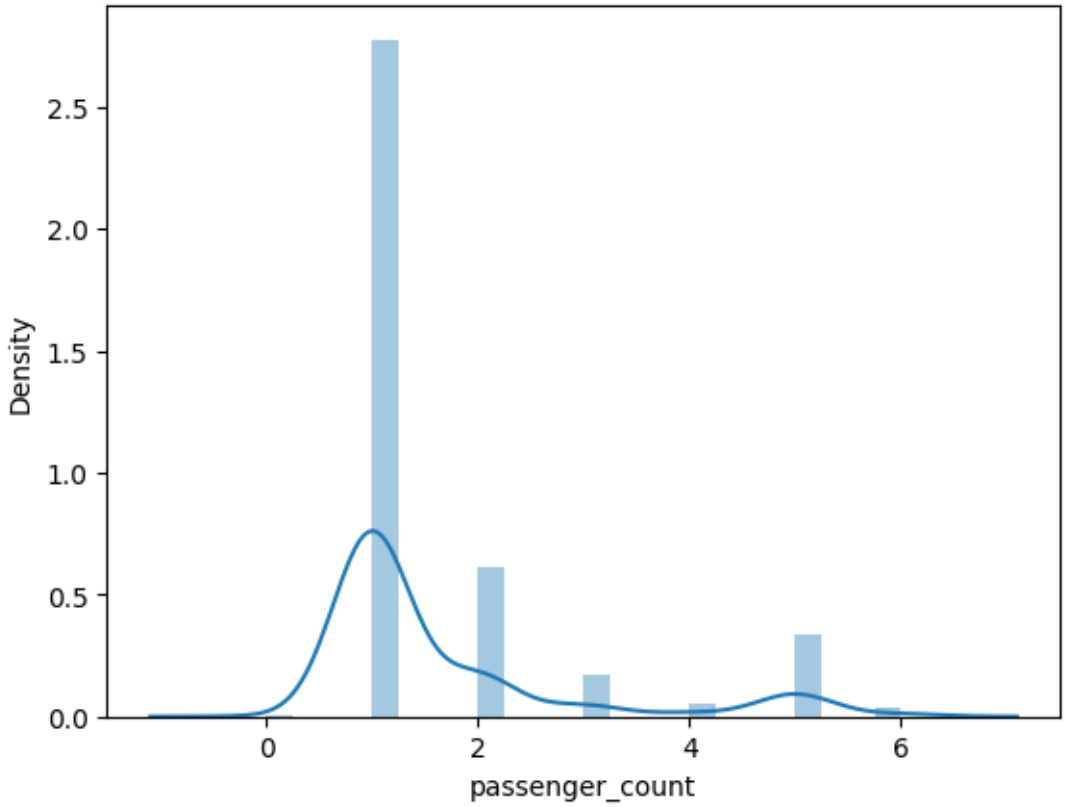


[ ]:
```
sns.distplot(df1['passenger_count'])
```

```
<ipython-input-11-dd1ad478bc93>:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
```

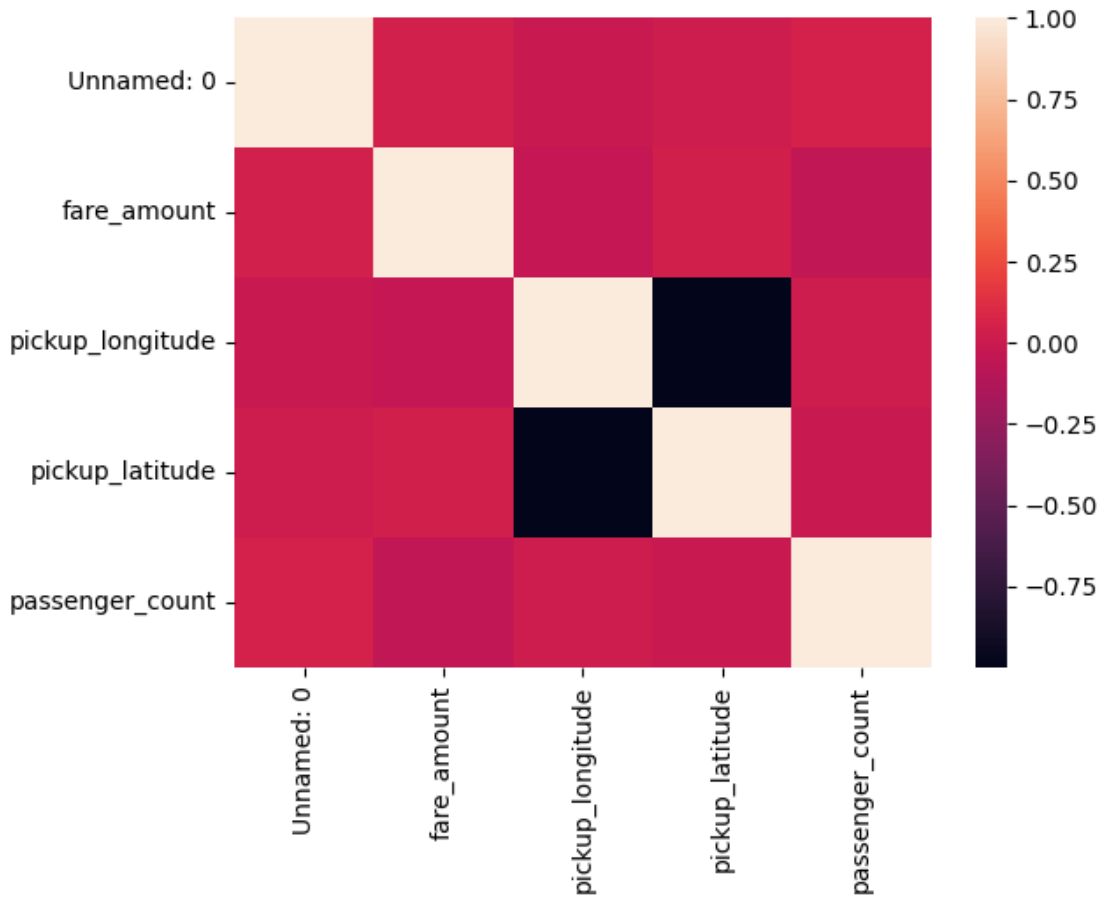
```
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df1['passenger_count'])
```

[ ]:
```
<Axes: xlabel='passenger_count', ylabel='Density'>
```



[ ]:
```
sns.heatmap(df1.corr())
```

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



3 TO TRAIN THE MODEL AND MODEL BULDING

[ ]:
```
x=df1[['Unnamed: 0', 'fare_amount',
       'pickup_longitude', 'pickup_latitude']]
y=df1['passenger_count']
```

[ ]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

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

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

[ ]:
```
lr.intercept_
```

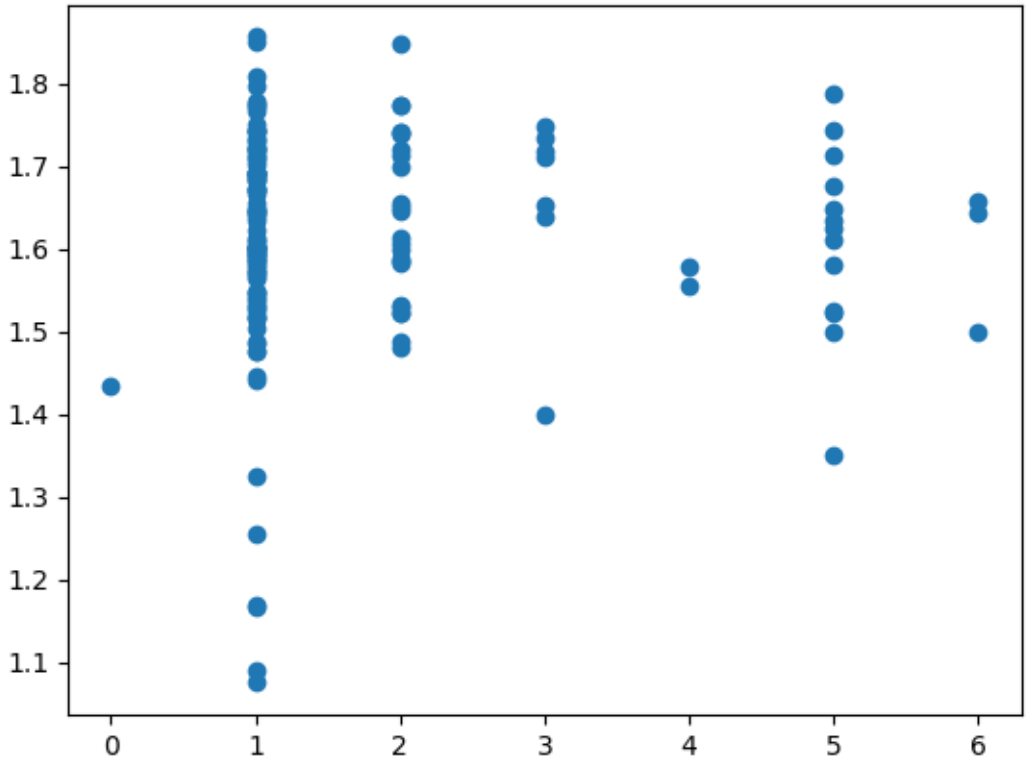
[ ]:
```
1.9026592548679888
```

[ ]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

[ ]:
```
                  Co-efficient
Unnamed: 0        1.060758e-09
fare_amount      -1.201690e-02
pickup_longitude  1.459743e+00
pickup_latitude   2.645480e+00
```

[ ]:
```
prediction =lr.predict(x_test)
plt.scatter(y_test,prediction)
```

[ ]:
```
<matplotlib.collections.PathCollection at 0x7f8e7843d840>
```



4 ACCURACY

[ ]:
```
lr.score(x_test,y_test)
```

[ ]:
```
-0.013825230029084645
```

[ ]:
```
lr.score(x_train,y_train)
```

[ ]: 0.009651415235651495

[ ]:
```
from sklearn.linear_model import Ridge,Lasso
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_ridge.py:216:
LinAlgWarning: Ill-conditioned matrix (rcond=9.75176e-17): result may not be
accurate.
  return linalg.solve(A, Xy, assume_a="pos", overwrite_a=True).T
```

[ ]: Ridge(alpha=10)

[ ]:
```
rr.score(x_train,y_train)
```

[ ]: 0.0052371388878637015

[ ]:
```
rr.score(x_test,y_test)
```

[ ]: -0.004467586358262388

[ ]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

[ ]: Lasso(alpha=10)

[ ]:
```
la.score(x_train,y_train)
```

[ ]: 0.00048490761672093097

[ ]:
```
la.score(x_test,y_test)
```

[ ]: -0.0015402410727491933

[ ]:
```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

[ ]: ElasticNet()

[ ]:
```
print(en.coef_)
print(en.intercept_)
```

```
[ 1.68782784e-09 -1.28851135e-03  1.20657980e-05 -0.00000000e+00]
1.6026312665254503
```

[ ]:
```
prediction = en.predict(x_test)
prediction
```

[ ]:
```
array([1.67132417, 1.64889407, 1.61128848, 1.68360392, 1.60309542,
       1.6155592 , 1.62766005, 1.63463464, 1.65843818, 1.66470457,
       1.66490561, 1.65573492, 1.66099988, 1.64647574, 1.6546951 ,
       1.60119881, 1.6650644 , 1.68002378, 1.61661018, 1.60965753,
       1.65746556, 1.6403476 , 1.6183411 , 1.6751887 , 1.60206243,
       1.59152021, 1.58402852, 1.630265  , 1.59271351, 1.59684599,
       1.62104816, 1.6579752 , 1.66501653, 1.6500686 , 1.65047254,
       1.65477321, 1.5808827 , 1.61563238, 1.61643749, 1.62304988,
       1.6097478 , 1.61623722, 1.60670835, 1.6607571 , 1.61457532,
       1.63798865, 1.61104973, 1.59112598, 1.61425396, 1.68437406,
       1.59774226, 1.68488096, 1.62610653, 1.64903287, 1.5974419 ,
       1.66927057, 1.60874312, 1.66069775, 1.65933505, 1.63559332,
       1.67392219, 1.62526864, 1.60920159, 1.66217706, 1.67945896,
       1.59536572, 1.58989034, 1.65776389, 1.59927478, 1.65241121,
       1.58262326, 1.65051586, 1.66294435, 1.66613817, 1.61235921,
       1.6468597 , 1.64572261, 1.63233992, 1.59567032, 1.59798862,
       1.68135154, 1.66352433, 1.60303177, 1.65764517, 1.65760777,
       1.64089348, 1.62674564, 1.68816794, 1.61613686, 1.63574627,
       1.64904654, 1.67681475, 1.64036159, 1.59612876, 1.60630912,
       1.66583981, 1.65119341, 1.66610041, 1.66927314, 1.63298446,
       1.61206582, 1.6568117 , 1.62635277, 1.60661891, 1.60739268,
       1.60159104, 1.59517394, 1.60024284, 1.61546901, 1.60895238,
       1.64482255, 1.60256069, 1.64005551, 1.59098249, 1.60793943,
       1.63410221, 1.61147639, 1.62430467, 1.64858599, 1.56636769,
       1.64485958, 1.59233274, 1.6462163 , 1.59696764, 1.63646101,
       1.62721023, 1.64280627, 1.67281723, 1.66009496, 1.60651139,
       1.68777873, 1.67371939, 1.66947454, 1.60479501, 1.6150129 ,
       1.66753882, 1.64983837, 1.62580118, 1.63148588, 1.67154142,
       1.65056539, 1.60683557, 1.66942127, 1.62104354, 1.61371533,
       1.65720616, 1.60738067, 1.63294324, 1.63518034, 1.67105474])
```

[ ]:
```
en.score(x_test,y_test)
```

[ ]:
```
-0.001145473979576872
```

[ ]:
```
from sklearn import metrics
print("Mean Absolute Error: ", metrics.mean_absolute_error(y_test,prediction))
print("Mean Squared Error: ", metrics.mean_squared_error(y_test,prediction))
print("Root Mean Squared Error: ", np.sqrt(metrics.
  ↪mean_squared_error(y_test,prediction)))
```

```
Mean Absolute Error:  0.9637195454231516
Mean Squared Error:  1.8243095303627845
Root Mean Squared Error:  1.3506700301564347
```