Name : Ijaz Ullah

Pipeline

It is used to execute the process sequentially and execute the steps, transformers, or estimators are named manually

make_pipeline

make_pipleine is an advanced method in scikit learn, in which the naming of the estimators or transformers are done automatically.

Assigment Day 2 : KKN Pipelines Date 12/12/2022

https://github.com/ijazkhan101/Machine-Learning-with-Python

```
import matplotlib.pylab as plt
from sklearn.metrics import mean_squared_error
from sklearn.datasets import load_boston
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
from sklearn.neighbors import KNeighborsRegressor
```

```
X,y = load_boston(return_X_y=True)
```

```
pipe =Pipeline([("scaler:",StandardScaler()),
("Algo",KNeighborsRegressor())])
```

```
pipe
```

```
Pipeline(steps=[('scaler:', StandardScaler()), ('Algo',
KNeighborsRegressor())])
```

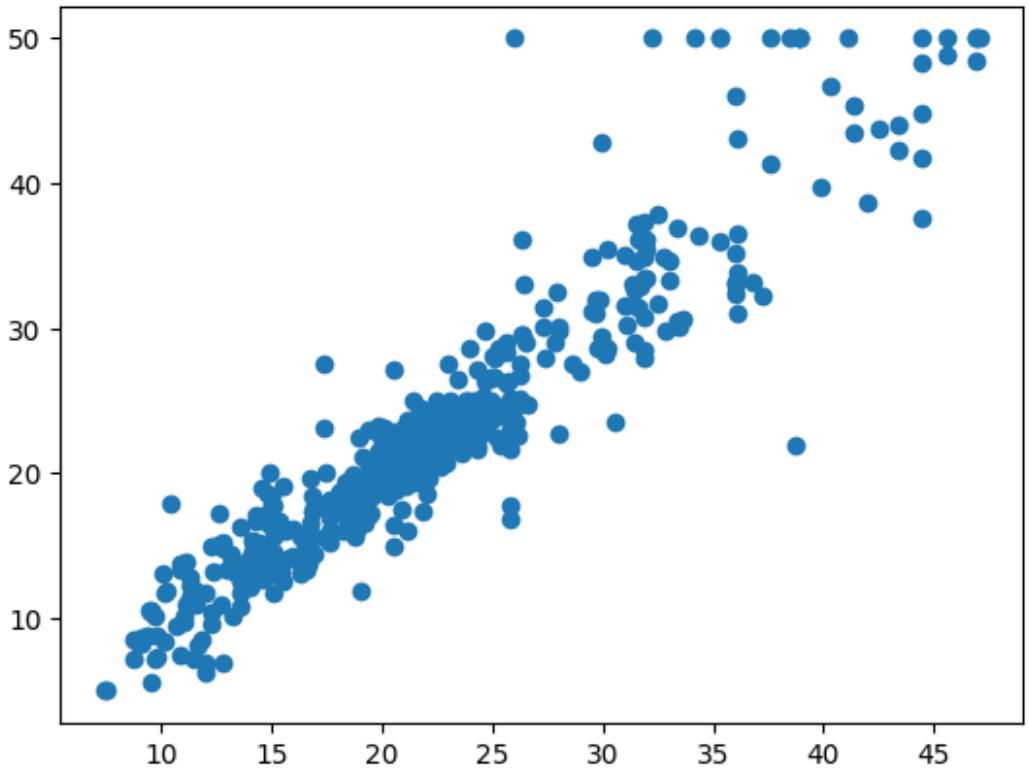
```
pipe.fit(X,y)
```

```
Pipeline(steps=[('scaler:', StandardScaler()), ('Algo',
KNeighborsRegressor())])
```

```
predicted_y=pipe.predict(X)
```

```
plt.scatter(predicted_y,y)
```

```
<matplotlib.collections.PathCollection at 0x212bfde8160>
```



2nd Assigment RandomizedSearchCV

RandomizedSearchCV is very useful when we have many parameters to try and the training time is very long.

```
import matplotlib.pylab as plt
from sklearn.linear_model import LinearRegression
from sklearn.datasets import load_boston
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
from sklearn.model_selection import RandomizedSearchCV
from sklearn.neighbors import KNeighborsRegressor
from scipy.stats import uniform
import warnings
warnings.filterwarnings('ignore')

X,y = load_boston(return_X_y=True)

pipe =Pipeline([
    ('scaler',StandardScaler()),
    ("algo",KNeighborsRegressor())]
)

pipe.get_params()

{'memory': None,
 'steps': [('scaler', StandardScaler()), ('algo',
```

```
KNeighborsRegressor())],
 'verbose': False,
 'scaler': StandardScaler(),
 'algo': KNeighborsRegressor(),
 'scaler__copy': True,
 'scaler__with_mean': True,
 'scaler__with_std': True,
 'algo__algorithm': 'auto',
 'algo__leaf_size': 30,
 'algo__metric': 'minkowski',
 'algo__metric_params': None,
 'algo__n_jobs': None,
 'algo__n_neighbors': 5,
 'algo__p': 2,
 'algo__weights': 'uniform'}
```

```
# using RandomizedSearchCV
model =RandomizedSearchCV(
    estimator=pipe,
    param_distributions ={'algo__n_neighbors':
[1,2,3,4,5,6,7,8,9,10]},
    cv=5
)
```

```
model
```

```
RandomizedSearchCV(cv=5,
                   estimator=Pipeline(steps=[('scaler',
StandardScaler()),
                                             ('algo',
KNeighborsRegressor())]),
                   param_distributions={'algo__n_neighbors': [1, 2, 3,
4, 5, 6,
                                                              7, 8, 9,
10]})
```

```
model.fit(X, y)
```

```
RandomizedSearchCV(cv=5,
                   estimator=Pipeline(steps=[('scaler',
StandardScaler()),
                                             ('algo',
KNeighborsRegressor())]),
                   param_distributions={'algo__n_neighbors': [1, 2, 3,
4, 5, 6,
                                                              7, 8, 9,
10]})
```

```
model.predict(X)
```

```
array([25.96666667, 23.02222222, 32.57777778, 28.26666667,
30.42222222,
```

```
       26.51111111, 21.63333333, 19.31111111, 18.63333333,
20.16666667,
       19.31111111, 20.6       , 20.85555556, 19.2       ,
18.26666667,
       19.74444444, 21.86666667, 17.08888889, 20.24444444,
18.76666667,
       14.96666667, 17.21111111, 15.47777778, 15.37777778,
15.57777778,
       15.26666667, 16.08888889, 14.96666667, 17.4       ,
20.33333333,
       14.44444444, 16.52222222, 13.83333333, 14.97777778,
13.83333333,
       20.01111111, 20.95555556, 21.41111111, 22.65555556,
30.47777778,
       32.75555556, 24.64444444, 24.27777778, 24.27777778,
21.71111111,
       21.81111111, 21.44444444, 20.24444444, 16.44444444,
19.96666667,
       20.11111111, 21.64444444, 24.71111111, 22.26666667,
20.05555556,
       31.47777778, 26.34444444, 31.66666667, 23.66666667,
22.35555556,
       19.44444444, 18.68888889, 22.4       , 23.6       ,
27.14444444,
       25.34444444, 22.2       , 22.46666667, 19.8       ,
21.17777778,
       24.15555556, 22.86666667, 23.75555556, 23.75555556,
23.53333333,
       22.77777778, 21.48888889, 22.64444444, 21.81111111,
22.8       ,
       25.95555556, 25.92222222, 24.1       , 23.27777778,
23.18888889,
       25.33333333, 21.88888889, 23.81111111, 29.94444444,
29.43333333,
       25.62222222, 25.62222222, 24.54444444, 24.4       ,
22.34444444,
       25.88888889, 24.58888889, 40.64444444, 40.14444444,
35.71111111,
       21.82222222, 22.25555556, 15.6       , 19.45555556,
20.56666667,
       18.96666667, 17.88888889, 19.37777778, 20.6       ,
19.2       ,
       21.5       , 22.56666667, 19.03333333, 19.03333333,
19.62222222,
       19.02222222, 19.96666667, 19.08888889, 18.86666667,
19.52222222,
       19.08888889, 19.08888889, 19.02222222, 18.14444444,
18.84444444,
       19.08888889, 18.14444444, 16.73333333, 18.73333333,
16.38888889,
```

```
       18.73333333, 18.73333333, 18.73333333, 17.43333333,
16.56666667,
       18.73333333, 16.73333333, 18.73333333, 16.38888889,
16.73333333,
       16.24444444, 13.36666667, 15.84444444, 16.61111111,
16.61111111,
       16.86666667, 17.77777778, 16.61111111, 16.61111111,
16.81111111,
       19.13333333, 18.58888889, 19.21111111, 17.47777778,
22.9       ,
       16.68888889, 18.12222222, 31.35555556, 24.41111111,
23.43333333,
       27.22222222, 32.02222222, 31.08888889, 32.61111111,
24.41111111,
       24.41111111, 35.86666667, 22.08888889, 24.41111111,
24.41111111,
       21.85555556, 24.41111111, 22.66666667, 25.8       ,
24.91111111,
       28.02222222, 25.45555556, 24.85555556, 26.3       ,
28.02222222,
       35.68888889, 24.45555556, 32.54444444, 27.42222222,
24.72222222,
       25.25555556, 39.37777778, 31.34444444, 31.34444444,
32.04444444,
       32.03333333, 29.88888889, 32.96666667, 27.72222222,
27.07777778,
       40.57777778, 34.23333333, 30.34444444, 34.23333333,
28.91111111,
       33.14444444, 24.38888889, 40.43333333, 40.57777778,
40.57777778,
       22.52222222, 23.45555556, 19.08888889, 23.97777778,
21.66666667,
       21.63333333, 21.66666667, 22.23333333, 24.22222222,
21.47777778,
       23.44444444, 23.45555556, 23.42222222, 21.92222222,
24.97777778,
       24.51111111, 22.62222222, 24.42222222, 25.55555556,
40.45555556,
       42.51111111, 40.45555556, 29.75555556, 37.88888889,
26.01111111,
       21.16666667, 30.87777778, 41.58888889, 41.58888889,
26.1       ,
       22.41111111, 24.42222222, 31.38888889, 24.44444444,
24.44444444,
       23.22222222, 21.66666667, 21.84444444, 24.87777778,
18.9       ,
       18.32222222, 23.35555556, 19.78888889, 22.56666667,
25.23333333,
       24.81111111, 24.81111111, 25.4       , 29.5       ,
22.28888889,
```

```
       22.28888889, 40.76666667, 41.41111111, 37.77777778,
37.05555556,
       37.18888889, 37.77777778, 41.41111111, 37.77777778,
37.05555556,
       25.91111111, 37.18888889, 41.41111111, 38.6       ,
23.97777778,
       21.65555556, 25.86666667, 24.35555556, 32.02222222,
31.35555556,
       28.48888889, 31.35555556, 31.35555556, 25.96666667,
29.61111111,
       40.06666667, 30.63333333, 32.02222222, 42.83333333,
32.76666667,
       25.58888889, 21.92222222, 23.5       , 23.8       ,
23.84444444,
       31.24444444, 33.55555556, 29.07777778, 24.08888889,
22.67777778,
       25.21111111, 23.31111111, 21.17777778, 24.6       ,
30.16666667,
       27.23333333, 25.81111111, 25.15555556, 29.13333333,
29.27777778,
       28.71111111, 32.15555556, 30.25555556, 24.41111111,
20.38888889,
       20.95555556, 21.93333333, 19.87777778, 20.54444444,
23.62222222,
       19.33333333, 18.94444444, 19.31111111, 21.82222222,
21.53333333,
       23.91111111, 23.43333333, 21.23333333, 18.3       ,
23.73333333,
       23.91111111, 22.37777778, 20.64444444, 20.92222222,
24.57777778,
       20.33333333, 20.37777778, 22.8       , 21.74444444,
21.06666667,
       20.75555556, 20.11111111, 19.75555556, 21.11111111,
20.11111111,
       20.01111111, 30.46666667, 21.48888889, 26.25555556,
30.1       ,
       20.51111111, 19.72222222, 26.86666667, 27.97777778,
26.25555556,
       24.34444444, 22.94444444, 21.61111111, 26.16666667,
21.24444444,
       21.24444444, 28.74444444, 29.76666667, 30.38888889,
19.93333333,
       19.67777778, 18.14444444, 19.76666667, 29.54444444,
37.12222222,
       23.15555556, 17.16666667, 16.93333333, 27.53333333,
30.87777778,
       33.43333333, 21.32222222, 34.        , 10.3       ,
10.36666667,
       14.36666667, 12.57777778, 12.83333333, 10.2       ,
10.68888889,
```

```
        8.46666667, 12.57777778, 11.92222222, 12.26666667,
10.36666667,
       10.48888889,  9.75555556,  9.66666667, 10.21111111,
13.01111111,
       15.25555556, 16.76666667, 11.2       , 15.58888889,
13.33333333,
       14.08888889, 14.88888889, 13.5       ,  8.45555556,
11.06666667,
        9.65555556, 12.66666667, 13.04444444, 10.28888889,
9.46666667,
        8.17777778,  9.55555556, 21.07777778, 12.64444444,
15.03333333,
       10.76666667, 12.12222222, 11.04444444, 11.97777778,
10.35555556,
        9.66666667, 10.95555556, 10.75555556, 10.25555556,
11.13333333,
       14.44444444, 15.12222222, 17.71111111, 11.95555556,
13.56666667,
       10.35555556, 13.56666667, 11.84444444, 12.18888889,
11.81111111,
       14.85555556, 14.47777778, 13.88888889, 13.78888889,
13.        ,
       11.06666667, 11.86666667,  9.37777778,  9.53333333,
13.16666667,
       10.14444444, 14.6       , 15.04444444, 13.61111111,
13.47777778,
       10.87777778, 15.77777778, 15.62222222, 14.74444444,
14.94444444,
       12.52222222, 15.        , 17.25555556, 14.61111111,
11.32222222,
       13.6       , 14.03333333, 13.58888889, 15.7       ,
18.21111111,
       15.7       , 18.68888889, 17.87777778, 19.42222222,
20.28888889,
       20.87777778, 13.8       , 17.43333333, 18.11111111,
20.57777778,
       19.62222222, 20.13333333, 21.54444444, 23.        ,
15.92222222,
       15.88888889, 15.93333333, 11.68888889, 14.55555556,
21.55555556,
       21.3       , 22.72222222, 22.75555556, 20.85555556,
20.85555556,
       21.62222222, 19.78888889, 20.85555556, 14.47777778,
13.53333333,
       13.33333333, 14.98888889, 16.72222222, 20.41111111,
20.93333333,
       20.48888889, 18.78888889, 19.57777778, 20.1       ,
19.42222222,
       18.98888889, 21.94444444, 19.23333333, 21.93333333,
```

```
21.73333333,
       19.14444444])
```

```
import pandas as pd
pd.DataFrame(model.cv_results_)
```

```
   mean_fit_time  std_fit_time  mean_score_time  std_score_time  \
0       0.004998      0.002531         0.004200        0.001166
1       0.007800      0.004400         0.011601        0.010052
2       0.007598      0.005315         0.011001        0.010899
3       0.004597      0.001622         0.010401        0.008593
4       0.023999      0.022564         0.007799        0.002786
5       0.009199      0.003430         0.008600        0.007735
6       0.007000      0.004427         0.012601        0.009457
7       0.007597      0.002155         0.016400        0.005678
8       0.006799      0.002317         0.007404        0.002503
9       0.004399      0.001198         0.004599        0.001498

  param_algo__n_neighbors                    params
split0_test_score  \
0                       1   {'algo__n_neighbors': 1}
0.339313
1                       2   {'algo__n_neighbors': 2}
0.441649
2                       3   {'algo__n_neighbors': 3}
0.520304
3                       4   {'algo__n_neighbors': 4}
0.547088
4                       5   {'algo__n_neighbors': 5}
0.560895
5                       6   {'algo__n_neighbors': 6}
0.582450
6                       7   {'algo__n_neighbors': 7}
0.602434
7                       8   {'algo__n_neighbors': 8}
0.615090
8                       9   {'algo__n_neighbors': 9}
0.625314
9                      10  {'algo__n_neighbors': 10}
0.614446

   split1_test_score  split2_test_score  split3_test_score
split4_test_score  \
0           0.423779           0.534566           0.486373          -
1.623928
1           0.547962           0.474980           0.496794          -
0.548699
2           0.593339           0.547746           0.513891
0.002980
3           0.606925           0.509770           0.490452
```

```
0.211278
4           0.619174           0.486619           0.469869
0.231330
5           0.621194           0.509111           0.446859
0.250417
6           0.636185           0.516102           0.442088
0.245749
7           0.631185           0.551340           0.440117
0.239072
8           0.630621           0.564464           0.429107
0.279376
9           0.652489           0.555555           0.420648
0.261128
```

	mean_test_score	std_test_score	rank_test_score
0	0.032020	0.830549	10
1	0.282537	0.417052	9
2	0.435652	0.218139	8
3	0.473103	0.136807	7
4	0.473577	0.132431	6
5	0.482006	0.130434	5
6	0.488512	0.139022	4
7	0.495361	0.144674	3
8	0.505776	0.134503	1
9	0.500853	0.143381	2

Model Grid Search CV : Model grid search CV in scikit learn (Part-2)

Grid Search is an effective method for adjusting the parameters in supervised learning and improve the generalization performance of a model. With Grid Search, we try all possible combinations of the parameters of interest and find the best ones.

```
import matplotlib.pylab as plt
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
from sklearn.datasets import load_boston
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
from sklearn.model_selection import GridSearchCV
from sklearn.neighbors import KNeighborsRegressor
import warnings
warnings.filterwarnings('ignore')

X,y =load_boston(return_X_y=True)

pipe =Pipeline([
    ('scaler',StandardScaler()),
    ("algo",KNeighborsRegressor())]
)

pipe.get_params()
```

```
{'memory': None,
 'steps': [('scaler', StandardScaler()), ('algo',
KNeighborsRegressor())],
 'verbose': False,
 'scaler': StandardScaler(),
 'algo': KNeighborsRegressor(),
 'scaler__copy': True,
 'scaler__with_mean': True,
 'scaler__with_std': True,
 'algo__algorithm': 'auto',
 'algo__leaf_size': 30,
 'algo__metric': 'minkowski',
 'algo__metric_params': None,
 'algo__n_jobs': None,
 'algo__n_neighbors': 5,
 'algo__p': 2,
 'algo__weights': 'uniform'}
```

```
model =GridSearchCV(
    estimator=pipe,
    param_grid= {'algo__n_neighbors': [1,2,3,4,5,6,7,8,9,10]},
    cv=5
)

model.fit(X,y)
```

```
GridSearchCV(cv=5,
             estimator=Pipeline(steps=[('scaler', StandardScaler()),
                                       ('algo',
KNeighborsRegressor())]),
             param_grid={'algo__n_neighbors': [1, 2, 3, 4, 5, 6, 7, 8,
9, 10]})

model.predict(X)
```

```
array([25.96666667, 23.02222222, 32.57777778, 28.26666667,
30.42222222,
       26.51111111, 21.63333333, 19.31111111, 18.63333333,
20.16666667,
       19.31111111, 20.6       , 20.85555556, 19.2       ,
18.26666667,
       19.74444444, 21.86666667, 17.08888889, 20.24444444,
18.76666667,
       14.96666667, 17.21111111, 15.47777778, 15.37777778,
15.57777778,
       15.26666667, 16.08888889, 14.96666667, 17.4       ,
20.33333333,
       14.44444444, 16.52222222, 13.83333333, 14.97777778,
13.83333333,
       20.01111111, 20.95555556, 21.41111111, 22.65555556,
30.47777778,
```

```
       32.75555556, 24.64444444, 24.27777778, 24.27777778,
21.71111111,
       21.81111111, 21.44444444, 20.24444444, 16.44444444,
19.96666667,
       20.11111111, 21.64444444, 24.71111111, 22.26666667,
20.05555556,
       31.47777778, 26.34444444, 31.66666667, 23.66666667,
22.35555556,
       19.44444444, 18.68888889, 22.4       , 23.6       ,
27.14444444,
       25.34444444, 22.2       , 22.46666667, 19.8       ,
21.17777778,
       24.15555556, 22.86666667, 23.75555556, 23.75555556,
23.53333333,
       22.77777778, 21.48888889, 22.64444444, 21.81111111,
22.8       ,
       25.95555556, 25.92222222, 24.1       , 23.27777778,
23.18888889,
       25.33333333, 21.88888889, 23.81111111, 29.94444444,
29.43333333,
       25.62222222, 25.62222222, 24.54444444, 24.4       ,
22.34444444,
       25.88888889, 24.58888889, 40.64444444, 40.14444444,
35.71111111,
       21.82222222, 22.25555556, 15.6       , 19.45555556,
20.56666667,
       18.96666667, 17.88888889, 19.37777778, 20.6       ,
19.2       ,
       21.5       , 22.56666667, 19.03333333, 19.03333333,
19.62222222,
       19.02222222, 19.96666667, 19.08888889, 18.86666667,
19.52222222,
       19.08888889, 19.08888889, 19.02222222, 18.14444444,
18.84444444,
       19.08888889, 18.14444444, 16.73333333, 18.73333333,
16.38888889,
       18.73333333, 18.73333333, 18.73333333, 17.43333333,
16.56666667,
       18.73333333, 16.73333333, 18.73333333, 16.38888889,
16.73333333,
       16.24444444, 13.36666667, 15.84444444, 16.61111111,
16.61111111,
       16.86666667, 17.77777778, 16.61111111, 16.61111111,
16.81111111,
       19.13333333, 18.58888889, 19.21111111, 17.47777778,
22.9       ,
       16.68888889, 18.12222222, 31.35555556, 24.41111111,
23.43333333,
       27.22222222, 32.02222222, 31.08888889, 32.61111111,
24.41111111,
```

```
       24.41111111, 35.86666667, 22.08888889, 24.41111111,
24.41111111,
       21.85555556, 24.41111111, 22.66666667, 25.8       ,
24.91111111,
       28.02222222, 25.45555556, 24.85555556, 26.3       ,
28.02222222,
       35.68888889, 24.45555556, 32.54444444, 27.42222222,
24.72222222,
       25.25555556, 39.37777778, 31.34444444, 31.34444444,
32.04444444,
       32.03333333, 29.88888889, 32.96666667, 27.72222222,
27.07777778,
       40.57777778, 34.23333333, 30.34444444, 34.23333333,
28.91111111,
       33.14444444, 24.38888889, 40.43333333, 40.57777778,
40.57777778,
       22.52222222, 23.45555556, 19.08888889, 23.97777778,
21.66666667,
       21.63333333, 21.66666667, 22.23333333, 24.22222222,
21.47777778,
       23.44444444, 23.45555556, 23.42222222, 21.92222222,
24.97777778,
       24.51111111, 22.62222222, 24.42222222, 25.55555556,
40.45555556,
       42.51111111, 40.45555556, 29.75555556, 37.88888889,
26.01111111,
       21.16666667, 30.87777778, 41.58888889, 41.58888889,
26.1       ,
       22.41111111, 24.42222222, 31.38888889, 24.44444444,
24.44444444,
       23.22222222, 21.66666667, 21.84444444, 24.87777778,
18.9       ,
       18.32222222, 23.35555556, 19.78888889, 22.56666667,
25.23333333,
       24.81111111, 24.81111111, 25.4       , 29.5       ,
22.28888889,
       22.28888889, 40.76666667, 41.41111111, 37.77777778,
37.05555556,
       37.18888889, 37.77777778, 41.41111111, 37.77777778,
37.05555556,
       25.91111111, 37.18888889, 41.41111111, 38.6       ,
23.97777778,
       21.65555556, 25.86666667, 24.35555556, 32.02222222,
31.35555556,
       28.48888889, 31.35555556, 31.35555556, 25.96666667,
29.61111111,
       40.06666667, 30.63333333, 32.02222222, 42.83333333,
32.76666667,
       25.58888889, 21.92222222, 23.5       , 23.8       ,
23.84444444,
```

```
       31.24444444, 33.55555556, 29.07777778, 24.08888889,
22.67777778,
       25.21111111, 23.31111111, 21.17777778, 24.6       ,
30.16666667,
       27.23333333, 25.81111111, 25.15555556, 29.13333333,
29.27777778,
       28.71111111, 32.15555556, 30.25555556, 24.41111111,
20.38888889,
       20.95555556, 21.93333333, 19.87777778, 20.54444444,
23.62222222,
       19.33333333, 18.94444444, 19.31111111, 21.82222222,
21.53333333,
       23.91111111, 23.43333333, 21.23333333, 18.3       ,
23.73333333,
       23.91111111, 22.37777778, 20.64444444, 20.92222222,
24.57777778,
       20.33333333, 20.37777778, 22.8       , 21.74444444,
21.06666667,
       20.75555556, 20.11111111, 19.75555556, 21.11111111,
20.11111111,
       20.01111111, 30.46666667, 21.48888889, 26.25555556,
30.1       ,
       20.51111111, 19.72222222, 26.86666667, 27.97777778,
26.25555556,
       24.34444444, 22.94444444, 21.61111111, 26.16666667,
21.24444444,
       21.24444444, 28.74444444, 29.76666667, 30.38888889,
19.93333333,
       19.67777778, 18.14444444, 19.76666667, 29.54444444,
37.12222222,
       23.15555556, 17.16666667, 16.93333333, 27.53333333,
30.87777778,
       33.43333333, 21.32222222, 34.        , 10.3       ,
10.36666667,
       14.36666667, 12.57777778, 12.83333333, 10.2       ,
10.68888889,
        8.46666667, 12.57777778, 11.92222222, 12.26666667,
10.36666667,
       10.48888889,  9.75555556,  9.66666667, 10.21111111,
13.01111111,
       15.25555556, 16.76666667, 11.2       , 15.58888889,
13.33333333,
       14.08888889, 14.88888889, 13.5       ,  8.45555556,
11.06666667,
        9.65555556, 12.66666667, 13.04444444, 10.28888889,
9.46666667,
        8.17777778,  9.55555556, 21.07777778, 12.64444444,
15.03333333,
       10.76666667, 12.12222222, 11.04444444, 11.97777778,
10.35555556,
```

```
        9.66666667, 10.95555556, 10.75555556, 10.25555556,
11.13333333,
       14.44444444, 15.12222222, 17.71111111, 11.95555556,
13.56666667,
       10.35555556, 13.56666667, 11.84444444, 12.18888889,
11.81111111,
       14.85555556, 14.47777778, 13.88888889, 13.78888889,
13.        ,
       11.06666667, 11.86666667,  9.37777778,  9.53333333,
13.16666667,
       10.14444444, 14.6       , 15.04444444, 13.61111111,
13.47777778,
       10.87777778, 15.77777778, 15.62222222, 14.74444444,
14.94444444,
       12.52222222, 15.        , 17.25555556, 14.61111111,
11.32222222,
       13.6       , 14.03333333, 13.58888889, 15.7       ,
18.21111111,
       15.7       , 18.68888889, 17.87777778, 19.42222222,
20.28888889,
       20.87777778, 13.8       , 17.43333333, 18.11111111,
20.57777778,
       19.62222222, 20.13333333, 21.54444444, 23.        ,
15.92222222,
       15.88888889, 15.93333333, 11.68888889, 14.55555556,
21.55555556,
       21.3       , 22.72222222, 22.75555556, 20.85555556,
20.85555556,
       21.62222222, 19.78888889, 20.85555556, 14.47777778,
13.53333333,
       13.33333333, 14.98888889, 16.72222222, 20.41111111,
20.93333333,
       20.48888889, 18.78888889, 19.57777778, 20.1       ,
19.42222222,
       18.98888889, 21.94444444, 19.23333333, 21.93333333,
21.73333333,
       19.14444444])
```

```
import pandas as pd
pd.DataFrame(model.cv_results_)
```

```
   mean_fit_time  std_fit_time  mean_score_time  std_score_time  \
0       0.004600      0.000799         0.004401        0.000491
1       0.009199      0.007414         0.006400        0.002059
2       0.005200      0.001469         0.009400        0.003384
3       0.014603      0.007914         0.007998        0.004561
4       0.017800      0.014275         0.008800        0.005114
5       0.007199      0.004168         0.009602        0.003007
6       0.006200      0.001942         0.005401        0.002246
7       0.008201      0.003920         0.007199        0.002040
8       0.004799      0.002135         0.005400        0.002801
```

```
9       0.002998      0.002002         0.002198        0.000401

  param_algo__n_neighbors                    params
split0_test_score  \
0                        1   {'algo__n_neighbors': 1}
0.339313
1                        2   {'algo__n_neighbors': 2}
0.441649
2                        3   {'algo__n_neighbors': 3}
0.520304
3                        4   {'algo__n_neighbors': 4}
0.547088
4                        5   {'algo__n_neighbors': 5}
0.560895
5                        6   {'algo__n_neighbors': 6}
0.582450
6                        7   {'algo__n_neighbors': 7}
0.602434
7                        8   {'algo__n_neighbors': 8}
0.615090
8                        9   {'algo__n_neighbors': 9}
0.625314
9                       10  {'algo__n_neighbors': 10}
0.614446

   split1_test_score  split2_test_score  split3_test_score
split4_test_score  \
0           0.423779           0.534566           0.486373          -
1.623928
1           0.547962           0.474980           0.496794          -
0.548699
2           0.593339           0.547746           0.513891
0.002980
3           0.606925           0.509770           0.490452
0.211278
4           0.619174           0.486619           0.469869
0.231330
5           0.621194           0.509111           0.446859
0.250417
6           0.636185           0.516102           0.442088
0.245749
7           0.631185           0.551340           0.440117
0.239072
8           0.630621           0.564464           0.429107
0.279376
9           0.652489           0.555555           0.420648
0.261128

   mean_test_score  std_test_score  rank_test_score
0         0.032020        0.830549               10
```

1	0.282537	0.417052	9
2	0.435652	0.218139	8
3	0.473103	0.136807	7
4	0.473577	0.132431	6
5	0.482006	0.130434	5
6	0.488512	0.139022	4
7	0.495361	0.144674	3
8	0.505776	0.134503	1
9	0.500853	0.143381	2