

```

import pandas as pd
import numpy as np

from sklearn.datasets import load_iris
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import GridSearchCV

iris = load_iris()

df_iris = pd.DataFrame(data= np.c_[iris['data'], iris['target']], columns= iris['feature_name'] + ['target'], index= iris['data'].range)

df_iris.describe()

```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	Species
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333	1.000000
std	0.828066	0.435866	1.765298	0.762238	0.810000
min	4.300000	2.000000	1.000000	0.100000	0.000000
25%	5.100000	2.800000	1.600000	0.300000	0.000000
50%	5.800000	3.000000	4.350000	1.300000	1.000000
75%	6.400000	3.300000	5.100000	1.800000	2.000000
max	7.900000	4.400000	6.900000	2.500000	2.000000

```
df_iris.head()
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	Species
0	5.1	3.5	1.4	0.2	0.0
1	4.9	3.0	1.4	0.2	0.0
2	4.7	3.2	1.3	0.2	0.0
3	4.6	3.1	1.5	0.2	0.0
4	5.0	3.6	1.4	0.2	0.0

```

X = df_iris.iloc[:, 0:4]
print(X)

```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2
..
145	6.7	3.0	5.2	2.3
146	6.3	2.5	5.0	1.9
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

[150 rows x 4 columns]

```
y = df_iris.iloc[:, 4]
print(y)
```

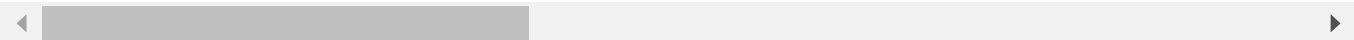
0	0.0
1	0.0
2	0.0
3	0.0
4	0.0

..	
145	2.0
146	2.0
147	2.0
148	2.0
149	2.0

Name: Species, Length: 150, dtype: float64

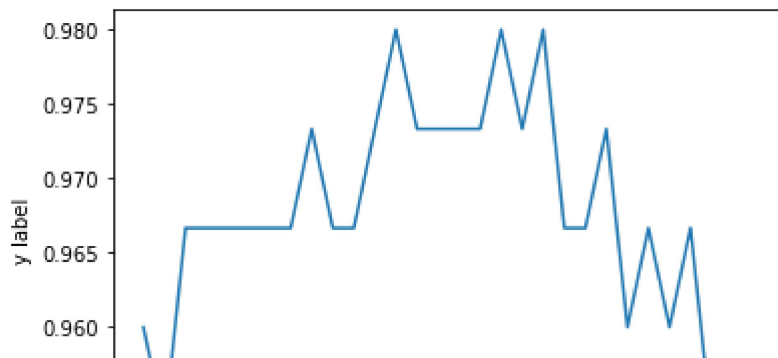
```
k = 1
k_range = 31
avg_score = pd.Series([])
for k in range(1, k_range):
    knn = KNeighborsClassifier(n_neighbors=k)
    scores = cross_val_score(knn, X, y, cv=10)
    avg_score[k] = scores.mean()
```

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:3: DeprecationWarning: The This is separate from the ipykernel package so we can avoid doing imports until



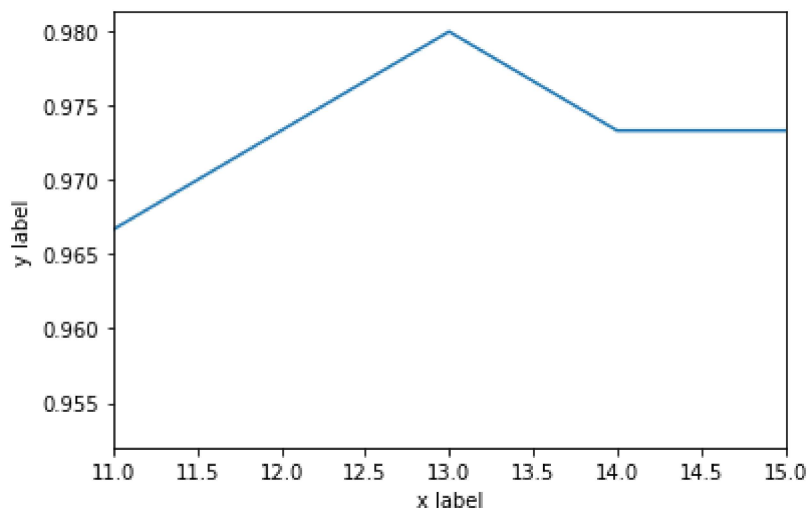
```
line = avg_score.plot.line()
line.set(xlabel="x label", ylabel="y label")
```

```
[Text(0, 0.5, 'y label'), Text(0.5, 0, 'x label')]
```



```
line = avg_score.plot.line()
line.set(xlabel="x label", ylabel="y label")
line.set_xlim([11, 15])
```

```
(11.0, 15.0)
```



```
k_range = [*range(1, 31)]
param_grid = dict(n_neighbors = k_range)
```

```
grid = GridSearchCV(knn, param_grid, cv = 10, scoring = 'accuracy')
```

[+ Code](#)
[+ Text](#)

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

```
GridSearchCV(cv=10, estimator=KNeighborsClassifier(n_neighbors=30),
             param_grid={'n_neighbors': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
                                           13, 14, 15, 16, 17, 18, 19, 20, 21, 22,
                                           23, 24, 25, 26, 27, 28, 29, 30]},
             scoring='accuracy')
```

```
print (grid.best_params_)
```

```
{'n_neighbors': 13}
```

[Colab paid products](#) - [Cancel contracts here](#)

✓ 0s completed at 3:46 PM

