IPA □□ □□□□□□□ □□(fundamental) □□

- GitHub link: here
- E-Mail: windkyle7@gmail.com

In [1]:

```
import seaborn as sns
iris = sns.load_dataset('iris')
```

□□ □□□□□ evaluation□ □□ □□ □□ □□(Learning curve)□ □□□□□□ □□.

In [2]:

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import learning_curve

knn = KNeighborsClassifier()
train_sizes, train_scores, test_scores =\
    learning_curve(knn, iris.iloc[:, :-1], iris.iloc[:, -1], cv=10, n_jobs=-1)
```

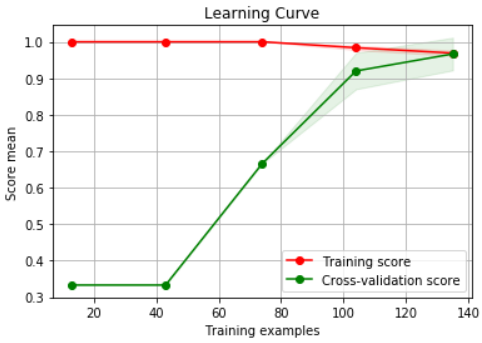
`sklearn_evaluation` □□□□ □□ □□□□ □□□□.

In [3]:

```
import sklearn_evaluation
sklearn_evaluation.plot.learning_curve(train_scores, test_scores, train_sizes)
```

Out[3]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fbd5da64080>
```



`make_classification`□□ □□ □□□□ □□□□ □□ □□ □□ □□□ □□□□.

In [4]:

```
import pandas as pd
from sklearn.datasets import make_classification

X, y = make_classification(1000, 5)
X = pd.DataFrame(X, columns=['x' + str(i + 1) for i in range(X.shape[1])])
y = pd.DataFrame(y, columns=['target'])
dataset = pd.concat([X, y], axis=1)
dataset.info()
```

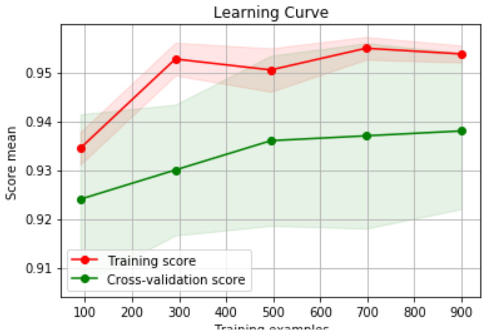
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 6 columns):
x1        1000 non-null float64
x2        1000 non-null float64
x3        1000 non-null float64
x4        1000 non-null float64
x5        1000 non-null float64
target    1000 non-null int64
dtypes: float64(5), int64(1)
memory usage: 47.0 KB
```

In [5]:

```
knn = KNeighborsClassifier()
train_sizes, train_scores, test_scores =\
    learning_curve(knn, dataset.iloc[:, :-1], dataset.iloc[:, -1], cv=10)
sklearn_evaluation.plot.learning_curve(train_scores, test_scores, train_sizes)
```

Out[5]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fbd5d9c09b0>
```



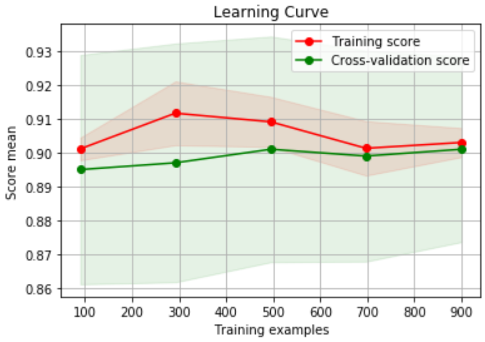
Training examples

`LogisticRegression` □□□ □□□□ □ □□□ □□.

```
In [6]: from sklearn.linear_model import LogisticRegression
        lr = LogisticRegression(solver='lbfgs')
```

```
In [7]: train_sizes, train_scores, test_scores =\
            learning_curve(lr, dataset.iloc[:, :-1], dataset.iloc[:, -1], cv=10)
        sklearn_evaluation.plot.learning_curve(train_scores, test_scores, train_sizes)
```

```
Out[7]: <matplotlib.axes._subplots.AxesSubplot at 0x7fbd5cfd1e80>
```



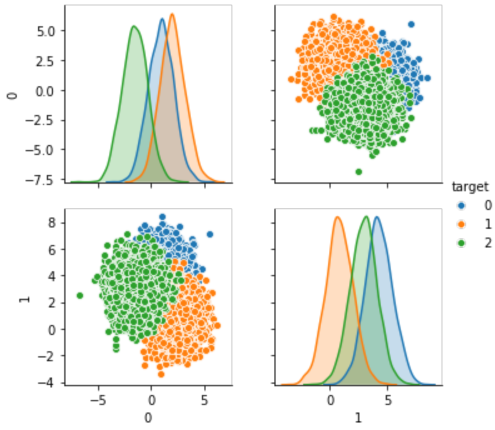
10000□□ □□ □□□□ □□□□□ □□ □□ □□□ □□□□□.

```
In [8]: from sklearn.datasets import make_blobs
        from sklearn.model_selection import train_test_split

        X, y = make_blobs(n_samples=10000, random_state=0, cluster_std=1.2)
        X, y = pd.DataFrame(X), pd.DataFrame(y, columns=['target'])
        blobs = pd.concat([X, y], axis=1)
```

```
In [9]: sns.pairplot(blobs, vars=blobs.columns[:-1], hue='target')
```

```
Out[9]: <seaborn.axisgrid.PairGrid at 0x7fbd5cf7bac8>
```

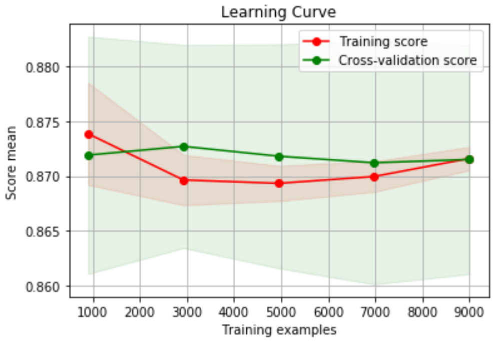


`n_jobs`□ □□□□□□ -1□ □□□□□ □□ □□□□□ CPU □□□ □□ □□□□□.

```
In [10]: lr = LogisticRegression(solver='lbfgs', max_iter=1000, multi_class='auto')
         train_sizes, train_scores, test_scores =\
             learning_curve(lr, blobs.iloc[:, :-1], blobs.iloc[:, -1], cv=10, n_jobs=-1)
```

```
In [11]: sklearn_evaluation.plot.learning_curve(train_scores, test_scores, train_sizes)
```

```
Out[11]: <matplotlib.axes._subplots.AxesSubplot at 0x7fbd5c53e8d0>
```



□□ □□□ □□□□ □□ CPU □□ □□ □□□ □□.

In [12]:
```
import multiprocessing
multiprocessing.cpu_count()
```

Out[12]:
```
8
```