hyperparameters_optimization

February 10, 2019

```
In [5]: from sklearn.datasets import load_digits
        digits = load_digits()
        X, Y = digits.data, digits.target

        from sklearn import svm
        h = svm.SVC()
        hp = svm.SVC(probability=True, random_state=1)
```

```
In [7]: from sklearn.model_selection import GridSearchCV
        search_grid = [
            {'C': [1, 10, 100, 1000], 'kernel':['linear']},
            {'C': [1, 10, 100, 1000], 'gamma': [0.001, 0.0001], 'kernel':['rbf'] },
        ]
        socrer = 'accuracy'
```

```
In [8]: search_func = GridSearchCV(estimator=h, param_grid=search_grid, scoring=socrer, n_jobs
        %timeit search_func.fit(X, Y)
        print(search_func.best_estimator_)
        print(search_func.best_params_)
        print(search_func.best_score_)
```

```
5.67 s ś 103 ms per loop (mean ś std. dev. of 7 runs, 1 loop each)
SVC(C=10, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma=0.001, kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
{'C': 10, 'gamma': 0.001, 'kernel': 'rbf'}
0.981081122784369
```

```
In [9]: search_func = GridSearchCV(estimator=hp, param_grid=search_grid, scoring=socrer, n_job
        %timeit search_func.fit(X, Y)
        print(search_func.best_estimator_)
        print(search_func.best_params_)
        print(search_func.best_score_)
```

```
16.7 s ś 178 ms per loop (mean ś std. dev. of 7 runs, 1 loop each)
SVC(C=10, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma=0.001, kernel='rbf',
  max_iter=-1, probability=True, random_state=1, shrinking=True, tol=0.001,
  verbose=False)
{'C': 10, 'gamma': 0.001, 'kernel': 'rbf'}
0.981081122784369
```

```
In [11]: # scorrer f1_weighted
         search_func = GridSearchCV(estimator=h, param_grid=search_grid, scoring="f1_weighted"
         %timeit search_func.fit(X, Y)
         print(search_func.best_estimator_)
         print(search_func.best_params_)
         print(search_func.best_score_)
```

```
5.84 s ś 270 ms per loop (mean ś std. dev. of 7 runs, 1 loop each)
SVC(C=10, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma=0.001, kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
{'C': 10, 'gamma': 0.001, 'kernel': 'rbf'}
0.9808918310846065
```

```
In [ ]:
```