


```
'feature_names': array(['mean radius', 'mean texture', 'mean perimeter', 'mean area',  
                        'mean smoothness', 'mean compactness', 'mean concavity',  
                        'mean concave points', 'mean symmetry', 'mean fractal dimension',  
                        'radius error', 'texture error', 'perimeter error', 'area error',  
                        'smoothness error', 'compactness error', 'concavity error',  
                        'concave points error', 'symmetry error',  
                        'fractal dimension error', 'worst radius', 'worst texture',  
                        'worst perimeter', 'worst area', 'worst smoothness',  
                        'worst compactness', 'worst concavity', 'worst concave points',  
                        'worst symmetry', 'worst fractal dimension'], dtype='<U23'),  
'filename': 'breast_cancer.csv',  
'data_module': 'sklearn.datasets.data'}
```

```
In [5]: ### input variable
        x
```

```
In [6]: ### output variable
y
```

```
In [9]: X_train, X_test, Y_train, Y_test = train_test_split(x, y, test_size=0.2, random_state=42)
```

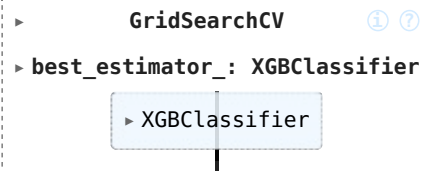
```
In [10]: ### apply model
param_grid = {
    'max_depth': [3, 4, 5],
    'learning_rate': [0.1, 0.01, 0.05],
    'n_estimators': [50, 100, 200],
    'subsample': [0.8, 1.0],
    'colsample_bytree': [0.8, 1.0]
}
```

```
In [11]: model =XGBClassifier(objective='binary:logistic' ,random_state =42)
```

```
In [12]: gc=GridSearchCV(estimator=model ,param_grid=param_grid ,cv =4 ,n_jobs=1)
```

```
In [13]: gc.fit(X_train ,Y_train)
```

```
Out[13]:
```



```
In [14]: pred =gc.predict(X_test)
```

```
In [15]: pred
```

```
Out[15]: array([[1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1,
    0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1,
    1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1,
    0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0,
    1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1,
    0, 1, 1, 0])
```

```
In [16]: ### check accuracy of model
print("best score :",gc.best_score_)
print("best estimator :",gc.best_estimator_)
print("best para_gramid :",gc.best_params_)
```

```
best score : 0.9714136003726129
```

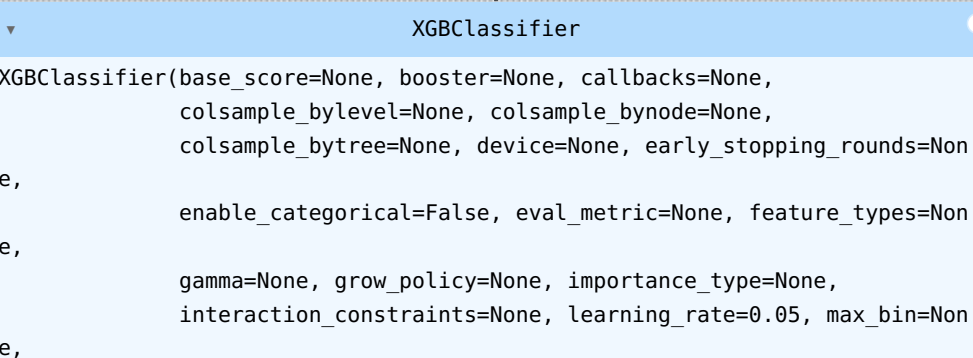
```
best estimator : XGBClassifier(base_score=None, booster=None, callbacks=None,
    colsample_bylevel=None, colsample_bynode=None,
    colsample_bytree=1.0, device=None, early_stopping_rounds=None,
    enable_categorical=False, eval_metric=None, feature_types=None,
    gamma=None, grow_policy=None, importance_type=None,
    interaction_constraints=None, learning_rate=0.05, max_bin=None,
    max_cat_threshold=None, max_cat_to_onehot=None,
    max_delta_step=None, max_depth=3, max_leaves=None,
    min_child_weight=None, missing=nan, monotone_constraints=None,
    multi_strategy=None, n_estimators=100, n_jobs=None,
    num_parallel_tree=None, random_state=42, ...)
```

```
best para_gramid : {'colsample_bytree': 1.0, 'learning_rate': 0.05, 'max_depth': 3, 'n_estimators': 100, 'subsample': 0.8}
```

```
In [18]: model2=XGBClassifier(n_estimators= 100 ,max_depth =3 ,learning_rate =0.05)
```

```
In [19]: model2.fit(X_train ,Y_train)
```

```
Out[19]:
```



```
In [20]: ### prediction
prediction =model2.predict(X_test)
```

```
In [21]: #### find out the accuracy
from sklearn.metrics import accuracy_score ,classification_report ,confusion_matrix

accuracy_score =accuracy_score(Y_test ,prediction)
```

In [22]: accuracy_score

Out[22]: 0.956140350877193

In [23]: *### classification report*
classification_report = classification_report(Y_test ,prediction)
print(classification_report)

	precision	recall	f1-score	support
0	0.95	0.93	0.94	43
1	0.96	0.97	0.97	71
accuracy			0.96	114
macro avg	0.96	0.95	0.95	114
weighted avg	0.96	0.96	0.96	114

In [24]: *## confusion matrix*
confusion_matrix = confusion_matrix(Y_test ,prediction)
print(confusion_matrix)

```
[[40  3]
 [ 2 69]]
```

In [25]: x.shape

Out[25]: (569, 30)

In [26]: X_train.shape

Out[26]: (455, 30)

In []: