SVM in Binary Calssification

By using SVM to predict whether a tumor is malignant or benign. From the built in breast cancer dataset from Scikit Learn. Challenge: all the data are trained into one class (due to using deafult parameters in svc), using grid search to test combination of parameters

In [2]:
```
# data library
import pandas as pd
import numpy as np
# visualization libaray
import matplotlib.pyplot as plt
import seaborn as sns
# since using jupyter notebook
%matplotlib inline
```

Load the Data

In [4]:
```
from sklearn.datasets import load_breast_cancer
```

In [6]:
```
cancer = load_breast_cancer()
```

In [7]:
```
print(type(cancer))
cancer.keys()
```

```
<class 'sklearn.utils.Bunch'>
```

Out[7]:
```
dict_keys(['data', 'target', 'target_names', 'DESCR', 'feature_names', 'filen
ame'])
```

In [11]:
```
print('nb_features=',len(cancer['feature_names']))
```

```
nb_features= 30
```

In [16]:
```
print('nb_samples =',len(cancer.data))
```

```
nb_samples = 569
```

Create a data frame of X, y

In [19]:
```
X = pd.DataFrame(cancer['data'], columns=cancer['feature_names'])
X.head(5)
```

Out[19]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809

5 rows × 30 columns

In [27]:
```
y =  cancer['target']
len(y)
```

Out[27]: 569

In [29]:
```
## statistic of X
X.describe()
```

Out[29]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity
count	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000
mean	14.127292	19.289649	91.969033	654.889104	0.096360	0.104341	0.088799
std	3.524049	4.301036	24.298981	351.914129	0.014064	0.052813	0.079720
min	6.981000	9.710000	43.790000	143.500000	0.052630	0.019380	0.000000
25%	11.700000	16.170000	75.170000	420.300000	0.086370	0.064920	0.029560
50%	13.370000	18.840000	86.240000	551.100000	0.095870	0.092630	0.061540
75%	15.780000	21.800000	104.100000	782.700000	0.105300	0.130400	0.130700
max	28.110000	39.280000	188.500000	2501.000000	0.163400	0.345400	0.426800

8 rows × 30 columns

In [31]:
```
# is there any object type column
(X.dtypes == 'object').any()
```

Out[31]: False

Train Test Split

In [33]:
```
from sklearn.model_selection import train_test_split
```

In [34]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.30, rando
m_state=101)
```

build the model

Using gridsearch for parameters

In [48]:
```
from sklearn.svm import SVC
from sklearn.model_selection import GridSearchCV
```

In [50]:
```
param_grid = {'C': [0.1,1, 10, 100, 1000], 'gamma': [1,0.1,0.01,0.001,0.0001],
'kernel': ['rbf']}
```

In [56]:
```
grid = GridSearchCV(SVC(), param_grid, refit=True, verbose=3, cv =3)
```

In [57]:
```
grid.fit(X_train,y_train)
```

```
Fitting 3 folds for each of 25 candidates, totalling 75 fits
[CV] C=0.1, gamma=1, kernel=rbf .....................................
[CV]  C=0.1, gamma=1, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=0.1, gamma=1, kernel=rbf .....................................
[CV]  C=0.1, gamma=1, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=0.1, gamma=1, kernel=rbf .....................................
[CV]  C=0.1, gamma=1, kernel=rbf, score=0.6363636363636364, total=   0.0s
[CV] C=0.1, gamma=0.1, kernel=rbf ...................................
[CV]  C=0.1, gamma=0.1, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=0.1, gamma=0.1, kernel=rbf ...................................
[CV]  C=0.1, gamma=0.1, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=0.1, gamma=0.1, kernel=rbf ...................................
[CV]  C=0.1, gamma=0.1, kernel=rbf, score=0.6363636363636364, total=   0.0s
[CV] C=0.1, gamma=0.01, kernel=rbf ..................................
[CV]  C=0.1, gamma=0.01, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=0.1, gamma=0.01, kernel=rbf ..................................
[CV]  C=0.1, gamma=0.01, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=0.1, gamma=0.01, kernel=rbf ..................................
[CV]  C=0.1, gamma=0.01, kernel=rbf, score=0.6363636363636364, total=   0.0s

[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent worke
rs.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.0s remaining:    0.
0s
[Parallel(n_jobs=1)]: Done   2 out of   2 | elapsed:    0.0s remaining:    0.
0s
```

```
[CV] C=0.1, gamma=0.001, kernel=rbf ..............................
[CV]  C=0.1, gamma=0.001, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=0.1, gamma=0.001, kernel=rbf ..............................
[CV]  C=0.1, gamma=0.001, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=0.1, gamma=0.001, kernel=rbf ..............................
[CV]  C=0.1, gamma=0.001, kernel=rbf, score=0.6363636363636364, total=   0.0s
[CV] C=0.1, gamma=0.0001, kernel=rbf .............................
[CV]  C=0.1, gamma=0.0001, kernel=rbf, score=0.9022556390977443, total=   0.0
s
[CV] C=0.1, gamma=0.0001, kernel=rbf .............................
[CV]  C=0.1, gamma=0.0001, kernel=rbf, score=0.9624060150375939, total=   0.0
s
[CV] C=0.1, gamma=0.0001, kernel=rbf .............................
[CV]  C=0.1, gamma=0.0001, kernel=rbf, score=0.9166666666666666, total=   0.0
s
[CV] C=1, gamma=1, kernel=rbf ....................................
[CV]  C=1, gamma=1, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=1, gamma=1, kernel=rbf ....................................
[CV]  C=1, gamma=1, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=1, gamma=1, kernel=rbf ....................................
[CV]  C=1, gamma=1, kernel=rbf, score=0.6363636363636364, total=   0.0s
[CV] C=1, gamma=0.1, kernel=rbf ..................................
[CV]  C=1, gamma=0.1, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=1, gamma=0.1, kernel=rbf ..................................
[CV]  C=1, gamma=0.1, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=1, gamma=0.1, kernel=rbf ..................................
[CV]  C=1, gamma=0.1, kernel=rbf, score=0.6363636363636364, total=   0.0s
[CV] C=1, gamma=0.01, kernel=rbf .................................
[CV]  C=1, gamma=0.01, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=1, gamma=0.01, kernel=rbf .................................
[CV]  C=1, gamma=0.01, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=1, gamma=0.01, kernel=rbf .................................
[CV]  C=1, gamma=0.01, kernel=rbf, score=0.6363636363636364, total=   0.0s
[CV] C=1, gamma=0.001, kernel=rbf ................................
[CV]  C=1, gamma=0.001, kernel=rbf, score=0.9022556390977443, total=   0.0s
[CV] C=1, gamma=0.001, kernel=rbf ................................
[CV]  C=1, gamma=0.001, kernel=rbf, score=0.9398496240601504, total=   0.0s
[CV] C=1, gamma=0.001, kernel=rbf ................................
[CV]  C=1, gamma=0.001, kernel=rbf, score=0.9545454545454546, total=   0.0s
[CV] C=1, gamma=0.0001, kernel=rbf ...............................
[CV]  C=1, gamma=0.0001, kernel=rbf, score=0.9398496240601504, total=   0.0s
[CV] C=1, gamma=0.0001, kernel=rbf ...............................
[CV]  C=1, gamma=0.0001, kernel=rbf, score=0.9699248120300752, total=   0.0s
[CV] C=1, gamma=0.0001, kernel=rbf ...............................
[CV]  C=1, gamma=0.0001, kernel=rbf, score=0.946969696969697, total=   0.0s
[CV] C=10, gamma=1, kernel=rbf ...................................
[CV]  C=10, gamma=1, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=10, gamma=1, kernel=rbf ...................................
[CV]  C=10, gamma=1, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=10, gamma=1, kernel=rbf ...................................
[CV]  C=10, gamma=1, kernel=rbf, score=0.6363636363636364, total=   0.0s
[CV] C=10, gamma=0.1, kernel=rbf .................................
[CV]  C=10, gamma=0.1, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=10, gamma=0.1, kernel=rbf .................................
[CV]  C=10, gamma=0.1, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=10, gamma=0.1, kernel=rbf .................................
[CV]  C=10, gamma=0.1, kernel=rbf, score=0.6363636363636364, total=   0.0s
```

```
[CV] C=10, gamma=0.01, kernel=rbf ....................................
[CV]  C=10, gamma=0.01, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=10, gamma=0.01, kernel=rbf ....................................
[CV]  C=10, gamma=0.01, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=10, gamma=0.01, kernel=rbf ....................................
[CV]  C=10, gamma=0.01, kernel=rbf, score=0.6363636363636364, total=   0.0s
[CV] C=10, gamma=0.001, kernel=rbf ...................................
[CV]  C=10, gamma=0.001, kernel=rbf, score=0.8947368421052632, total=   0.0s
[CV] C=10, gamma=0.001, kernel=rbf ...................................
[CV]  C=10, gamma=0.001, kernel=rbf, score=0.9323308270676691, total=   0.0s
[CV] C=10, gamma=0.001, kernel=rbf ...................................
[CV]  C=10, gamma=0.001, kernel=rbf, score=0.9166666666666666, total=   0.0s
[CV] C=10, gamma=0.0001, kernel=rbf ..................................
[CV]  C=10, gamma=0.0001, kernel=rbf, score=0.9323308270676691, total=   0.0s
[CV] C=10, gamma=0.0001, kernel=rbf ..................................
[CV]  C=10, gamma=0.0001, kernel=rbf, score=0.9699248120300752, total=   0.0s
[CV] C=10, gamma=0.0001, kernel=rbf ..................................
[CV]  C=10, gamma=0.0001, kernel=rbf, score=0.9621212121212122, total=   0.0s
[CV] C=100, gamma=1, kernel=rbf ......................................
[CV]  C=100, gamma=1, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=100, gamma=1, kernel=rbf ......................................
[CV]  C=100, gamma=1, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=100, gamma=1, kernel=rbf ......................................
[CV]  C=100, gamma=1, kernel=rbf, score=0.6363636363636364, total=   0.0s
[CV] C=100, gamma=0.1, kernel=rbf ....................................
[CV]  C=100, gamma=0.1, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=100, gamma=0.1, kernel=rbf ....................................
[CV]  C=100, gamma=0.1, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=100, gamma=0.1, kernel=rbf ....................................
[CV]  C=100, gamma=0.1, kernel=rbf, score=0.6363636363636364, total=   0.0s
[CV] C=100, gamma=0.01, kernel=rbf ...................................
[CV]  C=100, gamma=0.01, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=100, gamma=0.01, kernel=rbf ...................................
[CV]  C=100, gamma=0.01, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=100, gamma=0.01, kernel=rbf ...................................
[CV]  C=100, gamma=0.01, kernel=rbf, score=0.6363636363636364, total=   0.0s
[CV] C=100, gamma=0.001, kernel=rbf ..................................
[CV]  C=100, gamma=0.001, kernel=rbf, score=0.8947368421052632, total=   0.0s
[CV] C=100, gamma=0.001, kernel=rbf ..................................
[CV]  C=100, gamma=0.001, kernel=rbf, score=0.9323308270676691, total=   0.0s
[CV] C=100, gamma=0.001, kernel=rbf ..................................
[CV]  C=100, gamma=0.001, kernel=rbf, score=0.9166666666666666, total=   0.0s
[CV] C=100, gamma=0.0001, kernel=rbf .................................
[CV]  C=100, gamma=0.0001, kernel=rbf, score=0.9172932330827067, total=   0.0
s
[CV] C=100, gamma=0.0001, kernel=rbf .................................
[CV]  C=100, gamma=0.0001, kernel=rbf, score=0.9774436090225563, total=   0.0
s
[CV] C=100, gamma=0.0001, kernel=rbf .................................
[CV]  C=100, gamma=0.0001, kernel=rbf, score=0.9393939393939394, total=   0.0
s
[CV] C=1000, gamma=1, kernel=rbf .....................................
[CV]  C=1000, gamma=1, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=1000, gamma=1, kernel=rbf .....................................
[CV]  C=1000, gamma=1, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=1000, gamma=1, kernel=rbf .....................................
[CV]  C=1000, gamma=1, kernel=rbf, score=0.6363636363636364, total=   0.0s
```

```
[CV] C=1000, gamma=0.1, kernel=rbf ..................................
[CV]  C=1000, gamma=0.1, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=1000, gamma=0.1, kernel=rbf ..................................
[CV]  C=1000, gamma=0.1, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=1000, gamma=0.1, kernel=rbf ..................................
[CV]  C=1000, gamma=0.1, kernel=rbf, score=0.6363636363636364, total=   0.0s
[CV] C=1000, gamma=0.01, kernel=rbf .................................
[CV]  C=1000, gamma=0.01, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=1000, gamma=0.01, kernel=rbf .................................
[CV]  C=1000, gamma=0.01, kernel=rbf, score=0.631578947368421, total=   0.0s
[CV] C=1000, gamma=0.01, kernel=rbf .................................
[CV]  C=1000, gamma=0.01, kernel=rbf, score=0.6363636363636364, total=   0.0s
[CV] C=1000, gamma=0.001, kernel=rbf ................................
[CV]  C=1000, gamma=0.001, kernel=rbf, score=0.8947368421052632, total=   0.0
s
[CV] C=1000, gamma=0.001, kernel=rbf ................................
[CV]  C=1000, gamma=0.001, kernel=rbf, score=0.9323308270676691, total=   0.0
s
[CV] C=1000, gamma=0.001, kernel=rbf ................................
[CV]  C=1000, gamma=0.001, kernel=rbf, score=0.9166666666666666, total=   0.0
s
[CV] C=1000, gamma=0.0001, kernel=rbf ...............................
[CV]  C=1000, gamma=0.0001, kernel=rbf, score=0.9097744360902256, total=   0.
0s
[CV] C=1000, gamma=0.0001, kernel=rbf ...............................
[CV]  C=1000, gamma=0.0001, kernel=rbf, score=0.9699248120300752, total=   0.
0s
[CV] C=1000, gamma=0.0001, kernel=rbf ...............................
[CV]  C=1000, gamma=0.0001, kernel=rbf, score=0.9318181818181818, total=   0.
0s

[Parallel(n_jobs=1)]: Done  75 out of  75 | elapsed:    1.1s finished
```

Out[57]:
```
GridSearchCV(cv=3, error_score='raise-deprecating',
       estimator=SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto_deprecated',
  kernel='rbf', max_iter=-1, probability=False, random_state=None,
  shrinking=True, tol=0.001, verbose=False),
       fit_params=None, iid='warn', n_jobs=None,
       param_grid={'C': [0.1, 1, 10, 100, 1000], 'gamma': [1, 0.1, 0.01, 0.00
1, 0.0001], 'kernel': ['rbf']},
       pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
       scoring=None, verbose=3)
```

In [60]:
```
grid.best_params_
```

Out[60]:
```
{'C': 10, 'gamma': 0.0001, 'kernel': 'rbf'}
```

In [67]:
```
grid.best_estimator_
```

Out[67]:
```
SVC(C=10, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma=0.0001, kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
```

prediction

In [78]:
```
pred = grid.predict(X_test)
pred_train = grid.predict(X_train)
```

In [ ]:
```

```

Evaluation

In [71]:
```
from sklearn.metrics import confusion_matrix, classification_report, accuracy_
score
```

In [77]:
```
print('test data:\n')
print('accuracy = %.2f' %(accuracy_score(y_test, pred)))
print(confusion_matrix(y_test, pred))
print(classification_report(y_test, pred))
print(grid.score(X_test, y_test))
```

```
test data:

accuracy = 0.95
[[ 60   6]
 [  3 102]]
              precision    recall  f1-score   support

           0       0.95      0.91      0.93        66
           1       0.94      0.97      0.96       105

   micro avg       0.95      0.95      0.95       171
   macro avg       0.95      0.94      0.94       171
weighted avg       0.95      0.95      0.95       171

0.9473684210526315
```

In [79]:
```
print('train data:\n')
print('accuracy = %.2f' %(accuracy_score(y_train, pred_train)))
print(confusion_matrix(y_train, pred_train))
print(classification_report(y_train, pred_train))
print(grid.score(X_train, y_train))
```

```
train data:

accuracy = 0.97
[[138   8]
 [  4 248]]
              precision    recall  f1-score   support

           0       0.97      0.95      0.96       146
           1       0.97      0.98      0.98       252

   micro avg       0.97      0.97      0.97       398
   macro avg       0.97      0.96      0.97       398
weighted avg       0.97      0.97      0.97       398

0.9698492462311558
```

In [ ]: