Support Vector Machines Notes

In [1]:
```
import pandas as pd
import numpy as np
```

In [2]:
```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [3]:
```
from sklearn.datasets import load_breast_cancer
```

In [4]:
```
cancer = load_breast_cancer()
```

In [5]:
```
cancer.keys()
```

Out[5]:
```
dict_keys(['data', 'target', 'frame', 'target_names', 'DESCR', 'feature_names', 'filename', 'data_module'])
```

In [6]:
```
print(cancer['DESCR'])
```

```
.. _breast_cancer_dataset:

Breast cancer wisconsin (diagnostic) dataset
--------------------------------------------

**Data Set Characteristics:**

    :Number of Instances: 569

    :Number of Attributes: 30 numeric, predictive attributes and the class

    :Attribute Information:
        - radius (mean of distances from center to points on the perimeter)
        - texture (standard deviation of gray-scale values)
        - perimeter
        - area
        - smoothness (local variation in radius lengths)
        - compactness (perimeter^2 / area - 1.0)
        - concavity (severity of concave portions of the contour)
        - concave points (number of concave portions of the contour)
        - symmetry
        - fractal dimension ("coastline approximation" - 1)

        The mean, standard error, and "worst" or largest (mean of the three
        worst/largest values) of these features were computed for each image,
        resulting in 30 features.  For instance, field 0 is Mean Radius, field
        10 is Radius SE, field 20 is Worst Radius.

        - class:
                - WDBC-Malignant
                - WDBC-Benign

    :Summary Statistics:

    ===================================== ====== ======
                                           Min    Max
    ===================================== ====== ======
    radius (mean):                        6.981  28.11
    texture (mean):                       9.71   39.28
    perimeter (mean):                     43.79  188.5
    area (mean):                          143.5  2501.0
    smoothness (mean):                    0.053  0.163
    compactness (mean):                   0.019  0.345
    concavity (mean):                     0.0    0.427
    concave points (mean):                0.0    0.201
    symmetry (mean):                      0.106  0.304
    fractal dimension (mean):             0.05   0.097
    radius (standard error):              0.112  2.873
    texture (standard error):             0.36   4.885
    perimeter (standard error):           0.757  21.98
    area (standard error):                6.802  542.2
    smoothness (standard error):          0.002  0.031
    compactness (standard error):         0.002  0.135
    concavity (standard error):           0.0    0.396
    concave points (standard error):      0.0    0.053
    symmetry (standard error):            0.008  0.079
    fractal dimension (standard error):   0.001  0.03
    radius (worst):                       7.93   36.04
    texture (worst):                      12.02  49.54
    perimeter (worst):                    50.41  251.2
    area (worst):                         185.2  4254.0
    smoothness (worst):                   0.071  0.223
    compactness (worst):                  0.027  1.058
    concavity (worst):                    0.0    1.252
```

```
    concave points (worst):               0.0    0.291
    symmetry (worst):                     0.156  0.664
    fractal dimension (worst):            0.055  0.208
    ====================================  ======  ======

    :Missing Attribute Values: None

    :Class Distribution: 212 - Malignant, 357 - Benign

    :Creator:  Dr. William H. Wolberg, W. Nick Street, Olvi L. Mangasarian

    :Donor: Nick Street

    :Date: November, 1995

This is a copy of UCI ML Breast Cancer Wisconsin (Diagnostic) datasets.
https://goo.gl/U2Uwz2

Features are computed from a digitized image of a fine needle
aspirate (FNA) of a breast mass.  They describe
characteristics of the cell nuclei present in the image.

Separating plane described above was obtained using
Multisurface Method-Tree (MSM-T) [K. P. Bennett, "Decision Tree
Construction Via Linear Programming." Proceedings of the 4th
Midwest Artificial Intelligence and Cognitive Science Society,
pp. 97-101, 1992], a classification method which uses linear
programming to construct a decision tree.  Relevant features
were selected using an exhaustive search in the space of 1-4
features and 1-3 separating planes.

The actual linear program used to obtain the separating plane
in the 3-dimensional space is that described in:
[K. P. Bennett and O. L. Mangasarian: "Robust Linear
Programming Discrimination of Two Linearly Inseparable Sets",
Optimization Methods and Software 1, 1992, 23-34].

This database is also available through the UW CS ftp server:

ftp ftp.cs.wisc.edu
cd math-prog/cpo-dataset/machine-learn/WDBC/

.. topic:: References

   - W.N. Street, W.H. Wolberg and O.L. Mangasarian. Nuclear feature extraction
     for breast tumor diagnosis. IS&T/SPIE 1993 International Symposium on
     Electronic Imaging: Science and Technology, volume 1905, pages 861-870,
     San Jose, CA, 1993.
   - O.L. Mangasarian, W.N. Street and W.H. Wolberg. Breast cancer diagnosis and
     prognosis via linear programming. Operations Research, 43(4), pages 570-577,
     July-August 1995.
   - W.H. Wolberg, W.N. Street, and O.L. Mangasarian. Machine learning techniques
     to diagnose breast cancer from fine-needle aspirates. Cancer Letters 77 (1994)
     163-171.
```

In [7]:
```
df_feat = pd.DataFrame(cancer['data'],columns=cancer['feature_names'])
```

In [8]:
```
df_feat.head(2)
```

Out[8]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst radius	worst texture	worst perimeter	v
0	17.99	10.38	122.8	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	...	25.38	17.33	184.6	2(
1	20.57	17.77	132.9	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	...	24.99	23.41	158.8	1(

2 rows × 30 columns

In [9]:
```
df_feat.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 30 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   mean radius              569 non-null    float64
 1   mean texture             569 non-null    float64
 2   mean perimeter           569 non-null    float64
 3   mean area                569 non-null    float64
 4   mean smoothness          569 non-null    float64
 5   mean compactness         569 non-null    float64
 6   mean concavity           569 non-null    float64
 7   mean concave points      569 non-null    float64
 8   mean symmetry            569 non-null    float64
 9   mean fractal dimension   569 non-null    float64
 10  radius error             569 non-null    float64
 11  texture error            569 non-null    float64
 12  perimeter error          569 non-null    float64
 13  area error               569 non-null    float64
 14  smoothness error         569 non-null    float64
 15  compactness error        569 non-null    float64
 16  concavity error          569 non-null    float64
 17  concave points error     569 non-null    float64
 18  symmetry error           569 non-null    float64
 19  fractal dimension error  569 non-null    float64
 20  worst radius             569 non-null    float64
 21  worst texture            569 non-null    float64
 22  worst perimeter          569 non-null    float64
 23  worst area               569 non-null    float64
 24  worst smoothness         569 non-null    float64
 25  worst compactness        569 non-null    float64
 26  worst concavity          569 non-null    float64
 27  worst concave points     569 non-null    float64
 28  worst symmetry           569 non-null    float64
 29  worst fractal dimension  569 non-null    float64
dtypes: float64(30)
memory usage: 133.5 KB
```

In [10]:
```
cancer['target_names']
```

Out[10]:
```
array(['malignant', 'benign'], dtype='<U9')
```

In [11]:
```
#This is a display of malignant and benign
cancer['target']
```

Out[11]:
```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0,
       1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0,
       1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1,
       1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0,
       0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1,
       1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0,
       0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0,
       1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1,
       1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0,
       0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0,
       0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0,
       1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1,
       1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1,
       1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0,
       1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1,
       1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1,
       1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1])
```

In [12]:
```
#We are going to skip the visualization part as it is mostly useful if
# you know what you are looking for. There are a lot of categories and we
#  are not pros in cancer research
```

In [13]:
```
#Train test split process

from sklearn.model_selection import train_test_split
```

In [14]:
```
X = df_feat
y = cancer['target']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=101)
```

In [21]:
```
from sklearn.svm import SVC
```

In [22]:
```
model = SVC()
```

In [23]:
```
model.fit(X_train,y_train)
```

Out[23]:
```
SVC()
```

In [24]:
```
predictions = model.predict(X_test)
```

In [25]:
```
from sklearn.metrics import classification_report,confusion_matrix
```

In [26]:
```
print(classification_report(y_test,predictions))
print ('\n')
print(confusion_matrix(y_test,predictions))
```

```
              precision    recall  f1-score   support

           0       0.95      0.85      0.90        66
           1       0.91      0.97      0.94       105

    accuracy                           0.92       171
   macro avg       0.93      0.91      0.92       171
weighted avg       0.93      0.92      0.92       171



[[ 56  10]
 [  3 102]]
```

In [28]:
```
from sklearn.model_selection import GridSearchCV
```

In [29]:
```
param_grid = {'C':[0.1,1,10,100,1000],'gamma':[1,0.1,0.01,0.001,0.0001]}
```

In [30]:
```
grid = GridSearchCV(SVC(),param_grid,verbose=3)
```

In [31]:
```
grid.fit(X_train,y_train) #It will run the same loop to find the best combination
```

```
Fitting 5 folds for each of 25 candidates, totalling 125 fits
[CV 1/5] END .....................C=0.1, gamma=1;, score=0.637 total time=   0.0s
[CV 2/5] END .....................C=0.1, gamma=1;, score=0.637 total time=   0.0s
[CV 3/5] END .....................C=0.1, gamma=1;, score=0.625 total time=   0.0s
[CV 4/5] END .....................C=0.1, gamma=1;, score=0.633 total time=   0.0s
[CV 5/5] END .....................C=0.1, gamma=1;, score=0.633 total time=   0.0s
[CV 1/5] END ...................C=0.1, gamma=0.1;, score=0.637 total time=   0.0s
[CV 2/5] END ...................C=0.1, gamma=0.1;, score=0.637 total time=   0.0s
[CV 3/5] END ...................C=0.1, gamma=0.1;, score=0.625 total time=   0.0s
[CV 4/5] END ...................C=0.1, gamma=0.1;, score=0.633 total time=   0.0s
[CV 5/5] END ...................C=0.1, gamma=0.1;, score=0.633 total time=   0.0s
[CV 1/5] END ..................C=0.1, gamma=0.01;, score=0.637 total time=   0.0s
[CV 2/5] END ..................C=0.1, gamma=0.01;, score=0.637 total time=   0.0s
[CV 3/5] END ..................C=0.1, gamma=0.01;, score=0.625 total time=   0.0s
[CV 4/5] END ..................C=0.1, gamma=0.01;, score=0.633 total time=   0.0s
[CV 5/5] END ..................C=0.1, gamma=0.01;, score=0.633 total time=   0.0s
[CV 1/5] END .................C=0.1, gamma=0.001;, score=0.637 total time=   0.0s
[CV 2/5] END .................C=0.1, gamma=0.001;, score=0.637 total time=   0.0s
[CV 3/5] END .................C=0.1, gamma=0.001;, score=0.625 total time=   0.0s
[CV 4/5] END .................C=0.1, gamma=0.001;, score=0.633 total time=   0.0s
[CV 5/5] END .................C=0.1, gamma=0.001;, score=0.633 total time=   0.0s
[CV 1/5] END ................C=0.1, gamma=0.0001;, score=0.887 total time=   0.0s
[CV 2/5] END ................C=0.1, gamma=0.0001;, score=0.938 total time=   0.0s
[CV 3/5] END ................C=0.1, gamma=0.0001;, score=0.963 total time=   0.0s
[CV 4/5] END ................C=0.1, gamma=0.0001;, score=0.962 total time=   0.0s
[CV 5/5] END ................C=0.1, gamma=0.0001;, score=0.886 total time=   0.0s
[CV 1/5] END .......................C=1, gamma=1;, score=0.637 total time=   0.0s
[CV 2/5] END .......................C=1, gamma=1;, score=0.637 total time=   0.0s
[CV 3/5] END .......................C=1, gamma=1;, score=0.625 total time=   0.0s
[CV 4/5] END .......................C=1, gamma=1;, score=0.633 total time=   0.0s
[CV 5/5] END .......................C=1, gamma=1;, score=0.633 total time=   0.0s
[CV 1/5] END ...................C=1, gamma=0.1;, score=0.637 total time=   0.0s
[CV 2/5] END ...................C=1, gamma=0.1;, score=0.637 total time=   0.0s
[CV 3/5] END ...................C=1, gamma=0.1;, score=0.625 total time=   0.0s
[CV 4/5] END ...................C=1, gamma=0.1;, score=0.633 total time=   0.0s
[CV 5/5] END ...................C=1, gamma=0.1;, score=0.633 total time=   0.0s
[CV 1/5] END ..................C=1, gamma=0.01;, score=0.637 total time=   0.0s
[CV 2/5] END ..................C=1, gamma=0.01;, score=0.637 total time=   0.0s
[CV 3/5] END ..................C=1, gamma=0.01;, score=0.625 total time=   0.0s
[CV 4/5] END ..................C=1, gamma=0.01;, score=0.633 total time=   0.0s
[CV 5/5] END ..................C=1, gamma=0.01;, score=0.633 total time=   0.0s
[CV 1/5] END .................C=1, gamma=0.001;, score=0.900 total time=   0.0s
[CV 2/5] END .................C=1, gamma=0.001;, score=0.912 total time=   0.0s
[CV 3/5] END .................C=1, gamma=0.001;, score=0.925 total time=   0.0s
[CV 4/5] END .................C=1, gamma=0.001;, score=0.962 total time=   0.0s
[CV 5/5] END .................C=1, gamma=0.001;, score=0.937 total time=   0.0s
[CV 1/5] END ................C=1, gamma=0.0001;, score=0.912 total time=   0.0s
[CV 2/5] END ................C=1, gamma=0.0001;, score=0.950 total time=   0.0s
[CV 3/5] END ................C=1, gamma=0.0001;, score=0.975 total time=   0.0s
[CV 4/5] END ................C=1, gamma=0.0001;, score=0.962 total time=   0.0s
```

```
[CV 5/5] END .................C=1, gamma=0.0001;, score=0.937 total time=   0.0s
[CV 1/5] END ..................C=10, gamma=1;, score=0.637 total time=   0.0s
[CV 2/5] END ..................C=10, gamma=1;, score=0.637 total time=   0.0s
[CV 3/5] END ..................C=10, gamma=1;, score=0.625 total time=   0.0s
[CV 4/5] END ..................C=10, gamma=1;, score=0.633 total time=   0.0s
[CV 5/5] END ..................C=10, gamma=1;, score=0.633 total time=   0.0s
[CV 1/5] END ................C=10, gamma=0.1;, score=0.637 total time=   0.0s
[CV 2/5] END ................C=10, gamma=0.1;, score=0.637 total time=   0.0s
[CV 3/5] END ................C=10, gamma=0.1;, score=0.625 total time=   0.0s
[CV 4/5] END ................C=10, gamma=0.1;, score=0.633 total time=   0.0s
[CV 5/5] END ................C=10, gamma=0.1;, score=0.633 total time=   0.0s
[CV 1/5] END ...............C=10, gamma=0.01;, score=0.637 total time=   0.0s
[CV 2/5] END ...............C=10, gamma=0.01;, score=0.637 total time=   0.0s
[CV 3/5] END ...............C=10, gamma=0.01;, score=0.613 total time=   0.0s
[CV 4/5] END ...............C=10, gamma=0.01;, score=0.633 total time=   0.0s
[CV 5/5] END ...............C=10, gamma=0.01;, score=0.633 total time=   0.0s
[CV 1/5] END ..............C=10, gamma=0.001;, score=0.887 total time=   0.0s
[CV 2/5] END ..............C=10, gamma=0.001;, score=0.912 total time=   0.0s
[CV 3/5] END ..............C=10, gamma=0.001;, score=0.900 total time=   0.0s
[CV 4/5] END ..............C=10, gamma=0.001;, score=0.937 total time=   0.0s
[CV 5/5] END ..............C=10, gamma=0.001;, score=0.924 total time=   0.0s
[CV 1/5] END .............C=10, gamma=0.0001;, score=0.950 total time=   0.0s
[CV 2/5] END .............C=10, gamma=0.0001;, score=0.912 total time=   0.0s
[CV 3/5] END .............C=10, gamma=0.0001;, score=0.975 total time=   0.0s
[CV 4/5] END .............C=10, gamma=0.0001;, score=0.949 total time=   0.0s
[CV 5/5] END .............C=10, gamma=0.0001;, score=0.949 total time=   0.0s
[CV 1/5] END .................C=100, gamma=1;, score=0.637 total time=   0.0s
[CV 2/5] END .................C=100, gamma=1;, score=0.637 total time=   0.0s
[CV 3/5] END .................C=100, gamma=1;, score=0.625 total time=   0.0s
[CV 4/5] END .................C=100, gamma=1;, score=0.633 total time=   0.0s
[CV 5/5] END .................C=100, gamma=1;, score=0.633 total time=   0.0s
[CV 1/5] END ................C=100, gamma=0.1;, score=0.637 total time=   0.0s
[CV 2/5] END ................C=100, gamma=0.1;, score=0.637 total time=   0.0s
[CV 3/5] END ................C=100, gamma=0.1;, score=0.625 total time=   0.0s
[CV 4/5] END ................C=100, gamma=0.1;, score=0.633 total time=   0.0s
[CV 5/5] END ................C=100, gamma=0.1;, score=0.633 total time=   0.0s
[CV 1/5] END ...............C=100, gamma=0.01;, score=0.637 total time=   0.0s
[CV 2/5] END ...............C=100, gamma=0.01;, score=0.637 total time=   0.0s
[CV 3/5] END ...............C=100, gamma=0.01;, score=0.613 total time=   0.0s
[CV 4/5] END ...............C=100, gamma=0.01;, score=0.633 total time=   0.0s
[CV 5/5] END ...............C=100, gamma=0.01;, score=0.633 total time=   0.0s
[CV 1/5] END ..............C=100, gamma=0.001;, score=0.887 total time=   0.0s
[CV 2/5] END ..............C=100, gamma=0.001;, score=0.912 total time=   0.0s
[CV 3/5] END ..............C=100, gamma=0.001;, score=0.900 total time=   0.0s
[CV 4/5] END ..............C=100, gamma=0.001;, score=0.937 total time=   0.0s
[CV 5/5] END ..............C=100, gamma=0.001;, score=0.924 total time=   0.0s
[CV 1/5] END .............C=100, gamma=0.0001;, score=0.925 total time=   0.0s
[CV 2/5] END .............C=100, gamma=0.0001;, score=0.912 total time=   0.0s
[CV 3/5] END .............C=100, gamma=0.0001;, score=0.975 total time=   0.0s
[CV 4/5] END .............C=100, gamma=0.0001;, score=0.937 total time=   0.0s
[CV 5/5] END .............C=100, gamma=0.0001;, score=0.949 total time=   0.0s
[CV 1/5] END ................C=1000, gamma=1;, score=0.637 total time=   0.0s
[CV 2/5] END ................C=1000, gamma=1;, score=0.637 total time=   0.0s
[CV 3/5] END ................C=1000, gamma=1;, score=0.625 total time=   0.0s
[CV 4/5] END ................C=1000, gamma=1;, score=0.633 total time=   0.0s
[CV 5/5] END ................C=1000, gamma=1;, score=0.633 total time=   0.0s
[CV 1/5] END ..............C=1000, gamma=0.1;, score=0.637 total time=   0.0s
[CV 2/5] END ..............C=1000, gamma=0.1;, score=0.637 total time=   0.0s
[CV 3/5] END ..............C=1000, gamma=0.1;, score=0.625 total time=   0.0s
[CV 4/5] END ..............C=1000, gamma=0.1;, score=0.633 total time=   0.0s
[CV 5/5] END ..............C=1000, gamma=0.1;, score=0.633 total time=   0.0s
[CV 1/5] END ............C=1000, gamma=0.01;, score=0.637 total time=   0.0s
[CV 2/5] END ............C=1000, gamma=0.01;, score=0.637 total time=   0.0s
[CV 3/5] END ............C=1000, gamma=0.01;, score=0.613 total time=   0.0s
[CV 4/5] END ............C=1000, gamma=0.01;, score=0.633 total time=   0.0s
[CV 5/5] END ............C=1000, gamma=0.01;, score=0.633 total time=   0.0s
[CV 1/5] END ...........C=1000, gamma=0.001;, score=0.887 total time=   0.0s
[CV 2/5] END ...........C=1000, gamma=0.001;, score=0.912 total time=   0.0s
[CV 3/5] END ...........C=1000, gamma=0.001;, score=0.900 total time=   0.0s
[CV 4/5] END ...........C=1000, gamma=0.001;, score=0.937 total time=   0.0s
[CV 5/5] END ...........C=1000, gamma=0.001;, score=0.924 total time=   0.0s
[CV 1/5] END ..........C=1000, gamma=0.0001;, score=0.938 total time=   0.0s
[CV 2/5] END ..........C=1000, gamma=0.0001;, score=0.912 total time=   0.0s
[CV 3/5] END ..........C=1000, gamma=0.0001;, score=0.963 total time=   0.0s
[CV 4/5] END ..........C=1000, gamma=0.0001;, score=0.924 total time=   0.0s
[CV 5/5] END ..........C=1000, gamma=0.0001;, score=0.962 total time=   0.0s
```

Out[31]:
```
GridSearchCV(estimator=SVC(),
             param_grid={'C': [0.1, 1, 10, 100, 1000],
                         'gamma': [1, 0.1, 0.01, 0.001, 0.0001]},
             verbose=3)
```

In [32]:
```
grid.best_params_
```

Out[32]:
```
{'C': 1, 'gamma': 0.0001}
```

In [33]:
```
grid.best_estimator_
```

Out[33]:
```
SVC(C=1, gamma=0.0001)
```

In [36]:
```
grid_predictions= grid.predict(X_test)
```

In [35]:
```
print(classification_report(y_test,grid_predictions))
print ('\n')
print(confusion_matrix(y_test,grid_predictions))
```

```
              precision    recall  f1-score   support

           0       0.94      0.89      0.91        66
           1       0.94      0.96      0.95       105

    accuracy                           0.94       171
   macro avg       0.94      0.93      0.93       171
weighted avg       0.94      0.94      0.94       171



[[ 59   7]
 [  4 101]]
```

In [ ]: