Support Vector Machines

Import Packages

In [16]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

%matplotlib inline
```

Investigate Dataset

In [17]:
```
from sklearn.datasets import load_breast_cancer
```

In [18]:
```
cancer_data = load_breast_cancer()
```

Every Dataset included in `scikit-learn` comes with a description field.

Print the description field. Notice that :

- *569* observations in the Dataset.
- *30* numeric attributes for each observation.

In [19]:
```
display(cancer_data['DESCR'])
```

```
'.. _breast_cancer_dataset:\n\nBreast cancer wisconsin (diagnostic) dataset\n-----
--------------------------------------\n\n**Data Set Characteristics:**\n\n    :N
umber of Instances: 569\n\n    :Number of Attributes: 30 numeric, predictive attri
butes and the class\n\n    :Attribute Information:\n        - radius (mean of dist
ances from center to points on the perimeter)\n        - texture (standard deviati
on of gray-scale values)\n        - perimeter\n        - area\n        - smoothnes
s (local variation in radius lengths)\n        - compactness (perimeter^2 / area -
1.0)\n        - concavity (severity of concave portions of the contour)\n        -
concave points (number of concave portions of the contour)\n        - symmetry\n
      - fractal dimension ("coastline approximation" - 1)\n\n        The mean, sta
ndard error, and "worst" or largest (mean of the three\n        worst/largest valu
es) of these features were computed for each image,\n        resulting in 30 featu
res.  For instance, field 0 is Mean Radius, field\n        10 is Radius SE, field
20 is Worst Radius.\n\n        - class:\n                - WDBC-Malignant\n
         - WDBC-Benign\n\n    :Summary Statistics:\n\n    =======================
============= ====== ======\n                                           Min    Max
\n    ===================================== ====== ======\n    radius (mean):
                  6.981  28.11\n    texture (mean):                        9.71
39.28\n    perimeter (mean):                     43.79  188.5\n    area (mean):
                      143.5  2501.0\n    smoothness (mean):                     0.
053  0.163\n    compactness (mean):                   0.019  0.345\n    concavity
(mean):                     0.0    0.427\n    concave points (mean):
  0.0    0.201\n    symmetry (mean):                      0.106  0.304\n    fracta
l dimension (mean):             0.05   0.097\n    radius (standard error):
      0.112  2.873\n    texture (standard error):            0.36   4.885\n    pe
rimeter (standard error):          0.757  21.98\n    area (standard error):
          6.802  542.2\n    smoothness (standard error):         0.002  0.031\n
  compactness (standard error):         0.002  0.135\n    concavity (standard erro
r):           0.0    0.396\n    concave points (standard error):      0.0    0.053
\n    symmetry (standard error):           0.008  0.079\n    fractal dimension (s
tandard error):   0.001  0.03\n    radius (worst):                       7.93   3
6.04\n    texture (worst):                     12.02  49.54\n    perimeter (wors
t):                   50.41  251.2\n    area (worst):                        18
5.2  4254.0\n    smoothness (worst):                  0.071  0.223\n    compactne
ss (worst):                  0.027  1.058\n    concavity (worst):
   0.0    1.252\n    concave points (worst):              0.0    0.291\n    symme
try (worst):                     0.156  0.664\n    fractal dimension (worst):
       0.055  0.208\n    ===================================== ====== ======\n\n
  :Missing Attribute Values: None\n\n    :Class Distribution: 212 - Malignant, 357
- Benign\n\n    :Creator:  Dr. William H. Wolberg, W. Nick Street, Olvi L. Mangasa
rian\n\n    :Donor: Nick Street\n\n    :Date: November, 1995\n\nThis is a copy of
UCI ML Breast Cancer Wisconsin (Diagnostic) datasets.\nhttps://goo.gl/U2Uwz2\n\nFe
atures are computed from a digitized image of a fine needle\naspirate (FNA) of a b
reast mass.  They describe\ncharacteristics of the cell nuclei present in the imag
e.\n\nSeparating plane described above was obtained using\nMultisurface Method-Tre
e (MSM-T) [K. P. Bennett, "Decision Tree\nConstruction Via Linear Programming." Pr
oceedings of the 4th\nMidwest Artificial Intelligence and Cognitive Science Societ
y,\npp. 97-101, 1992], a classification method which uses linear\nprogramming to c
onstruct a decision tree.  Relevant features\nwere selected using an exhaustive se
arch in the space of 1-4\nfeatures and 1-3 separating planes.\n\nThe actual linear
program used to obtain the separating plane\nin the 3-dimensional space is that de
scribed in:\n[K. P. Bennett and O. L. Mangasarian: "Robust Linear\nProgramming Dis
crimination of Two Linearly Inseparable Sets",\nOptimization Methods and Software
1, 1992, 23-34].\n\nThis database is also available through the UW CS ftp serve
r:\n\nftp ftp.cs.wisc.edu\ncd math-prog/cpo-dataset/machine-learn/WDBC/\n\n.. topi
c:: References\n\n   - W.N. Street, W.H. Wolberg and O.L. Mangasarian. Nuclear fea
```

```
ture extraction \n     for breast tumor diagnosis. IS&T/SPIE 1993 International Sy
mposium on \n     Electronic Imaging: Science and Technology, volume 1905, pages 8
61-870,\n     San Jose, CA, 1993.\n    - O.L. Mangasarian, W.N. Street and W.H. Wol
berg. Breast cancer diagnosis and \n     prognosis via linear programming. Operati
ons Research, 43(4), pages 570-577, \n     July-August 1995.\n    - W.H. Wolberg,
W.N. Street, and O.L. Mangasarian. Machine learning techniques\n     to diagnose b
reast cancer from fine-needle aspirates. Cancer Letters 77 (1994) \n     163-171.'
```

Get Variables

In [20]:
```
raw_df = pd.DataFrame(cancer_data['data'], columns = cancer_data['feature_names'])
```

In [21]:
```
# Specify x variables as raw_df pandas DataFrame.
x = raw_df
# Specify y variable as what's stored under the key target, parsed from the origina
y = cancer_data['target']
```

Split Training and Test Data

Let's split the data into training and test sets. We will use *70%* of the data for training and *30%* for testing.

In [22]:
```
from sklearn.model_selection import train_test_split
```

In [23]:
```
x_training_data, x_test_data, y_training_data, y_test_data = train_test_split(x, y,
```

SVC Model

In [24]:
```
from sklearn.svm import SVC
```

In [25]:
```
model = SVC().fit(x_training_data, y_training_data)
```

Make Predictions and Evaluate Model

In [26]:
```
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
```

In [27]:
```
predictions = model.predict(x_test_data)
```

In [28]:
```
perforamance_report = classification_report(y_test_data, predictions)
print(perforamance_report)
```

```
              precision    recall  f1-score   support

           0       0.96      0.71      0.81        62
           1       0.86      0.98      0.91       109

    accuracy                           0.88       171
   macro avg       0.91      0.85      0.86       171
weighted avg       0.89      0.88      0.88       171
```

In [29]:
```
performance_matrix = confusion_matrix(y_test_data, predictions)
print(performance_matrix)
```

```
[[ 44  18]
 [  2 107]]
```