**Support Vector Machines**

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples. In two dimentional space this hyperplane is a line dividing a plane in two parts where in each class lay in either side.

**Implementation in Python**

Anaconda Python 3 Package

**Dataset**

This data has two types of cancer classes: malignant (harmful) and benign (not harmful). Here, we can build a model to classify the type of cancer. The dataset is available in the scikit-learn library or we can also download it from the UCI Machine Learning Library.

**Loading Data**

Let's first load the required dataset you will use.

#Import scikit-learn dataset library

from sklearn import datasets

#Load dataset

cancer = datasets.load\_breast\_cancer()

**Exploring Data**

After you have loaded the dataset, you might want to know a little bit more about it. You can check feature and target names.

# print the names of the 13 features

print("Features: ", cancer.feature\_names)

# print the label type of cancer('malignant' 'benign')

print("Labels: ", cancer.target\_names)

Let's explore it for a bit more. We can also check the shape of the dataset using shape.

# print data(feature)shape

cancer.data.shape

Let's check top 5 records of the feature set.

# print the cancer data features (top 5 records)

print (cancer.data[0:5])

Let's take a look at the target set.

# print the cancer labels (0:malignant, 1:benign)

print(cancer.target)

**Splitting Data**

To understand model performance, dividing the dataset into a training set and a test set is a good strategy. Split the dataset by using the function train\_test\_split(). We need to pass 3 parameters features, target, and test\_set size. Additionally, we can use random\_state to select records randomly.

# Import train\_test\_split function

from sklearn.model\_selection import train\_test\_split

# Split dataset into training set and test set

X\_train, X\_test, y\_train, y\_test = train\_test\_split(cancer.data, cancer.target, test\_size=0.3,random\_state=109) # 70% training and 30% test

**Generating Model**

Let's build support vector machine model. First, import the SVM module and create support vector classifier object by passing argument kernel as the linear kernel in SVC() function.

Then, fit your model on train set using fit() and perform prediction on the test set using predict().

#Import svm model

from sklearn import svm

#Create a svm Classifier

clf = svm.SVC(kernel='linear') # Linear Kernel

#Train the model using the training sets

clf.fit(X\_train, y\_train)

#Predict the response for test dataset

y\_pred = clf.predict(X\_test)

**Evaluating the Model**

Let's estimate how accurately the classifier or model can predict the breast cancer of patients. Accuracy can be computed by comparing actual test set values and predicted values.

#Import scikit-learn metrics module for accuracy calculation

from sklearn import metrics

# Model Accuracy: how often is the classifier correct?

print("Accuracy:",metrics.accuracy\_score(y\_test, y\_pred))