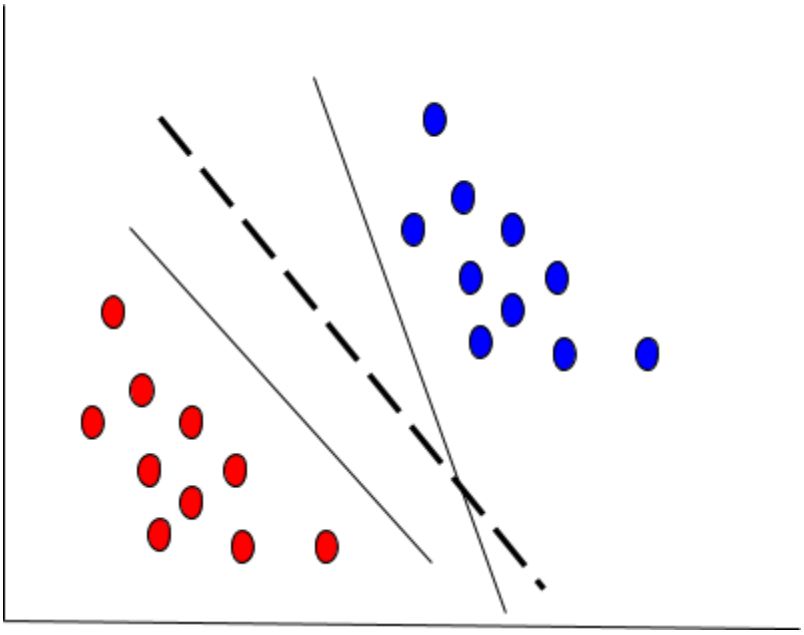
Support Vector Machines

In the last post we understood about Decision Tree classifiers, how it is build and about choosing the best split for building the classifier. Please visit the link below to catch up with the happening.

Link: https://codingmachinelearning.wordpress.com/2016/07/05/decision-trees-derivation-demystified/

In this post, we deal with one another awesome classifier, called the Support vector machines. They are called large margin classifiers - you will soon understand why. Here is a problem



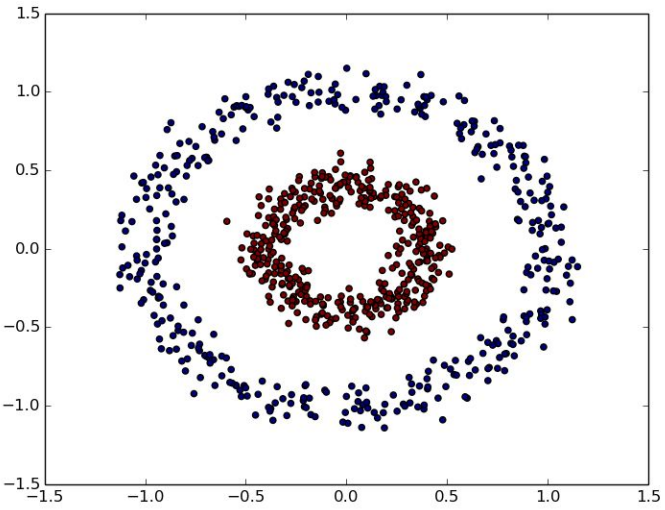
You see the red points and blue points in the x-y axis of the plot. Our aim is to come up with a margin or a line, which can help us separate the classes (dots)

You have 3 choices in front of you. Which one will you choose ? Obviously the dashed line correct? The reason, intuitively your brain has checked the distance between the two classes and fit a line that has the largest margin of divide between the two classes.

This is a very basic understanding of the large margin classifier called the SVM. What it does, is to get a hyperplane separating the two classes such that they have a wide margin between them to allow for generalization.

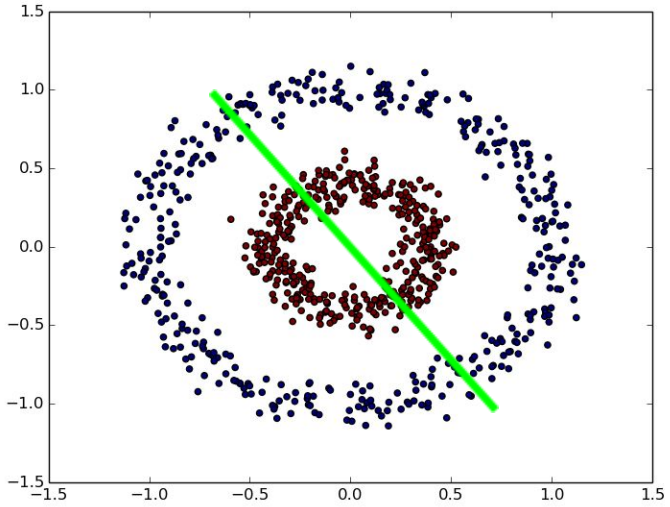
SVM derivation and proof is quite mathematically taxing, I will deal with it separately later. In this post we will focus on knowing how to use SVM.

First we will create some data points to understand the way svm works. If the points are linearly separable, then it is just a simple cakewalk. The above diagram explains it all. What if the points are not linearly separable. For example we have data in the form of doughnut - 2 circles, one inside another. Here a simple linear classifier will not work here.



We again use the scikit-learn package to generate the data set for demonstration purposes.

Just by using a linear decision boundary we cannot achieve a good classification. Please see the below image for illustration.



What to do in such situation ? how do we classify the points? How to deal with non linearly separable points?

All these questions will be answered in the later posts, but for now it is enough we understand what actually an SVM does.

Now we will move on to how to use SVM. It is quite simple by using scikit. We need to instantiate an object of SVC class and call fit and method and voila we are done.

We will now see how to fit an SVM for data. Say data is in variable train and labels are stored in variable target. We will have the following command to fit an SVM

```
from sklearn.svm import SVC

clf=SVC()

clf.fit(train,target)
```

Simple enough ? Scikit learn package takes care of how to get the best decision boundary to split the points.

To obtain the prediction on the test set invoke the predict method of the classifier on the test set. The prediction will be returned as output.

There you go, you now know how to fit and SVM to the data. It is quite simple when you are using default parameters.

In the next post, we will see about what is the meaning of a kernel, and how to get an decision boundary that can separate non linearly distributed points and so on.

Also we need to look up on the parameters of SVC() constructor for tuning the parameters etc, which will be explained further in the next set of posts. Till then bye!!
Note: Code for generating the doughnut-shaped dataset is posted on github

References:

- http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC
- https://en.wikipedia.org/wiki/Support_vector_machine