Unit-I

1. Always keep test data separate from training data. Re-using training data will result in over-match or over-fit and will result in wrong accuracy of the system. As a best practice, reserve 10% of total data as test data
2. Naïve Bayes’ rule assumptions:
   1. Classification labels need not to be known, only features are important
   2. Probability of each feature is known. If probabilities of features for different labels are different, classification is easy.
3. Execution of Naïve Bayes’ : Multiply probabilities of all features (known prior for each label) observed in test data to obtain probability of label; so as to classify the test data
4. Drawback of Naïve Bayes’: It doesn’t consider the order, it only considers the frequency or probabilities.

Unit-II

1. Support Vector Machines (SVM) first take into account the correct classification of data points; and then tries to maximize the margin on all sides while creating the separating line (decision boundary).
2. Decision boundary is always something linear; a line in case of 2D data, a plane may be in case of 3D data. To use SVM to create a non-linear decision boundary in given plane of visualization, the training data must be converted or **projected** to some other plane where SVM will work. This step is important in order to leverage capabilities of SVM. This method is usually called adding features to SVM
3. Always avoid **overfitting** while creating a decision boundary in machine learning. Overfitting just takes every feature literally and results in erratic non-linear decision boundary where we could have achieved a linear (close to linear, simple in a way) decision boundary instead.
4. Advantages of SVM:
   1. Work well in complex situations where there is clear separation between data set
5. Disadvantages of SVM:
   1. Don’t work well in large data sets as the computational time is of the cubic order
   2. Don’t work well when there is lot of noise so that classification becomes difficult. Here, naïve Bayes’ works better
6. SVM is more accurate than Naïve Bayes’. But it is much slower.

For a linear kernel,

* 1. NB took 15 seconds for training while SVM took almost 2 minutes. To reduce the time, we need to compromise on training data.
  2. When training data was down to 1% of what used in previous example, the accuracy dropped to 88% from 97% while the training time dropped to 0.07 seconds. That’s a lot of accuracy at lot less time.

For rbf kernel,

1. The accuracy has now dropped to 61% with almost no change in training time.
2. This accuracy can be improved by increasing ‘C’ value. Trying with 10, 100, 1000 and 10000, 10000 gives an accuracy of about 89% with 0.1 second increase in training time.
3. But beyond 10000, accuracy dropped. Hence ‘C’ value should be optimal