Question 3: Theoretical Questions

1. We have a dataset with k features and N samples and N>> k then:
   (a) In the case of discrete features (2 Category only), the highest no of leaves would be 2^k as each feature can be used only one time from each path when we are taking root to the leaf thus making the highest no of features to be 2^k.
   (b) In the event of continuous features, we can use the sample values many times. So, the maximum no of leaf nodes could be equal to number of samples which in this case would be N samples.
2. Email classification
   (a) This plan may not work because the probability is only of the total emails containing the keywords. The major problem is of curse of dimensionality. The no. of spam emails with (w1, w2…wn) as keywords will be 0 for a reasonably sized training set as an email will very unlikely contain n keywords for large n. So, for new emails probability will be undefined with the given equation in question. This is because the dataset considered is only for total emails containing specific set of keywords. The probability formula should look something like this:

$$P(Spam|word) = \frac{P(word|Spam) \times P(Spam)}{P(Word)}$$

$$P(Word) = P(word|spam) \times P(spam) + (P(word|Not\ Spam) \times P(Not\ spam)$$

   Here the features are the words in the email and so the probability should be calculated of each distinct word in the emails. After training the data set by counting number of times each word appears in the email, we can build the classifier and then test it on the email. Finally, for classification into Spam and No spam it would be needed to compare the computed probability with a threshold value. This threshold value would determine the efficiency of the classification which depends on "false positives" which would be spams wrongly classified as "non-spam" while true-positives would be spam emails correctly classified as Spam.
   (b) The dataset should consider all the emails while training the dataset and not only with those specific keywords. There should be a text column that contains the entire message in the email and another label column that contains the outcomes of emails as spam or legitimate while training the dataset. It is needed to include the word count in the dataset as well to check if any new word is not there in the training set. So, a large dataset is needed. Maybe 2^n emails in training set, n here is no of keywords.
   (c) Derivation:
   By Bayes Theorem we know that:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

$$where\ P(A) \neq 0\ and\ P(B) \neq 0$$

   Alternative form of Bayes Theorem says that:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B|A)P(A) + P(B|A')P(A')}$$

Extended form of Bayes Theorem says that:

$$P(A|B) = \frac{P(B|A)P(A)}{\Sigma P(B|A_i)P(A_i)}$$

Let the keywords ($w1$, $w2$, ..., wn) be denoted by vector $\vec{X}$. So $\vec{X}$ =< $w1$, $w2$, ..., wn >. Let every email be represented by vector $\vec{x} =< x1, x2 \ldots . xn >$ where each xi represents value of keyword w_i. In this it has been assumed that all words are non-independent of each other and no word's presence or absence effects other words i.e. they are conditionally independent. Using the extended version of Bayes Theorem, the extended version of Naïve Bayes can be written as below:

$$P\left(Spam\middle|\vec{X} = \vec{x}\right) = P(spam|e)$$

$$= \frac{P\left(\vec{X} = \vec{x}\middle|Spam\right)P(Spam)}{P\left(\vec{X} = \vec{x}\middle|Spam\right)P(Spam) + P\left(\vec{X} = \vec{x}\middle|Spam\right)P(Spam)}$$

where $P\left(\vec{X} = \vec{x}\middle|Spam\right) = \prod_{i=1}^{n} P\left(\vec{X}i = \overrightarrow{xi}\middle|Spam\right)$

The formula used to compute the probability of an email being spam thus will:

$$P\left(Spam\middle|\vec{X} = \vec{x}\right) = P(spam|e)$$

$$= \frac{\prod_{i=1}^{n} P\left(\vec{X} = \vec{x}\middle|Spam\right) \times P(Spam)}{\prod_{i=1}^{n} P\left(\vec{X} = \vec{x}\middle|Spam\right) \times P(Spam) + \prod_{i=1}^{n} P\left(\vec{X} = \vec{x}\middle|Spam'\right) \times P(Spam')}$$

or in other words:

$$P(Spam|word) = \frac{P(word|Spam) \times P(Spam)}{P(Word)}$$

$$P(Word) = P(word|spam) \times P(spam) + (P(word|Not\ Spam) \times P(Not\ spam)$$

Let w_i=0, if the email contains word with index I otherwise w_i=1 then:

$$P(wi = 1|spam) = \frac{Number\ of\ spams\ containing\ wi}{Number\ of\ spams}$$

$$P(wi = 0|spam) = 1 - P(wi = 0|spam)$$

$$P(wi = 1|Spam') = \frac{Number\ of\ not\ spams\ containing\ wi}{Number\ of\ not\ spams}$$

$$P(wi = 0|Spam') = 1 - P(wi = 0|Spam')$$

$$P(Spam) = \frac{Number\ of\ spam\ emails}{Total\ number\ of\ emails}$$

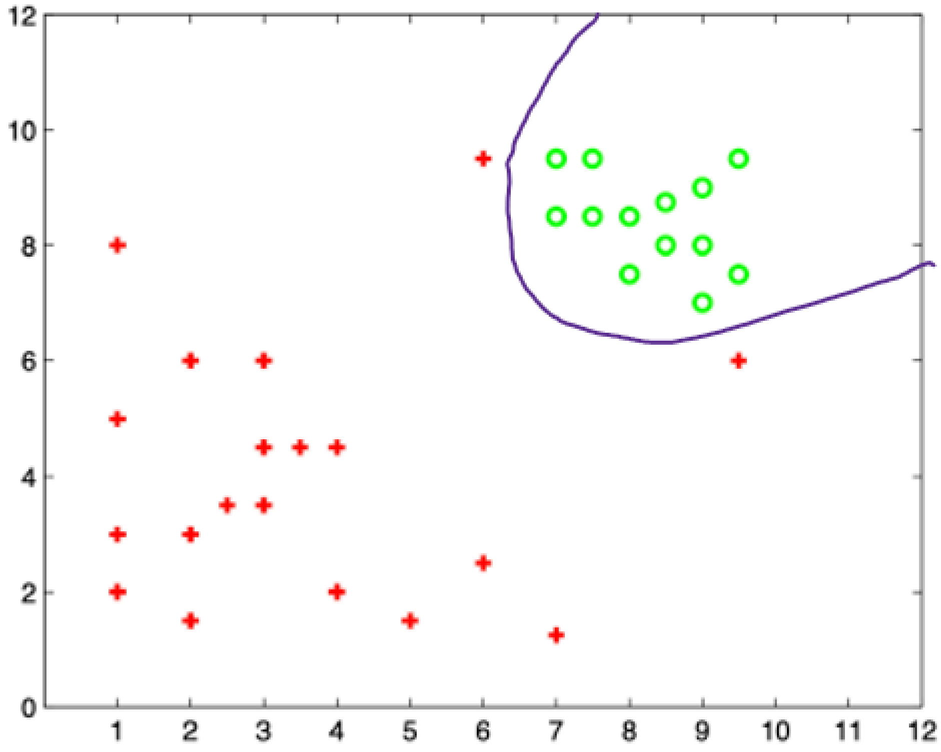
$$P(Spam') = 1 - P(Spam)$$

(d) It caters to our plan as it would correctly train and then classify the dataset by finding the correct no of occurrences of the spam words in the email.
As
$P(wi|spam) = \frac{Number\ of\ spams\ containing\ wi}{Number\ of\ spams}$ while we are deriving so the count will not be 0 often as we will just need 1 keyword to appear in email we are using to train instead of multiple keyword combination.

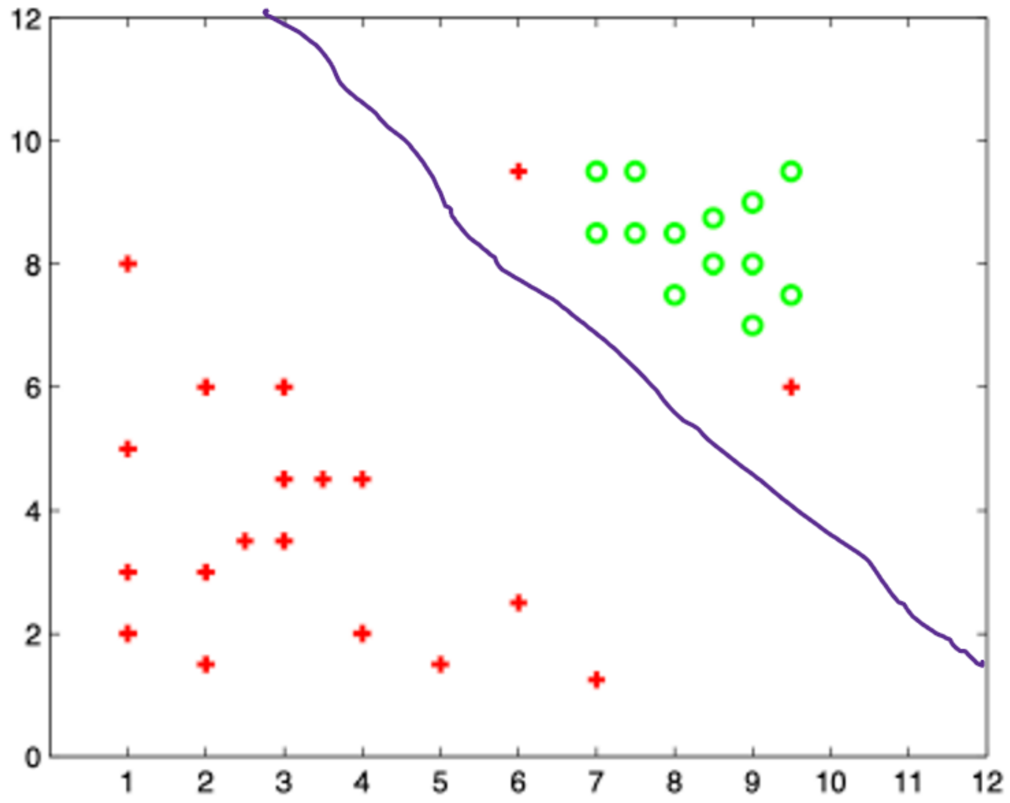
Further to minimize "false positives" we will need to compare probability of email being spam and not spam and set a threshold value (say c) as:

$$\frac{P(spam|\vec{X} = \vec{x})}{P\left(not\ spam|\vec{X} = \vec{x}\right)} > c$$

3. SVM with Quadratic kernel: As the Kernel is Quadratic lines will not be straight
   (a) High C that is approaching infinity means that the cost function is very high that means that the decision boundary will try to reduce the misclassification as much as possible that is it will try to perfectly separate the data as shown as below if is possible to do so.

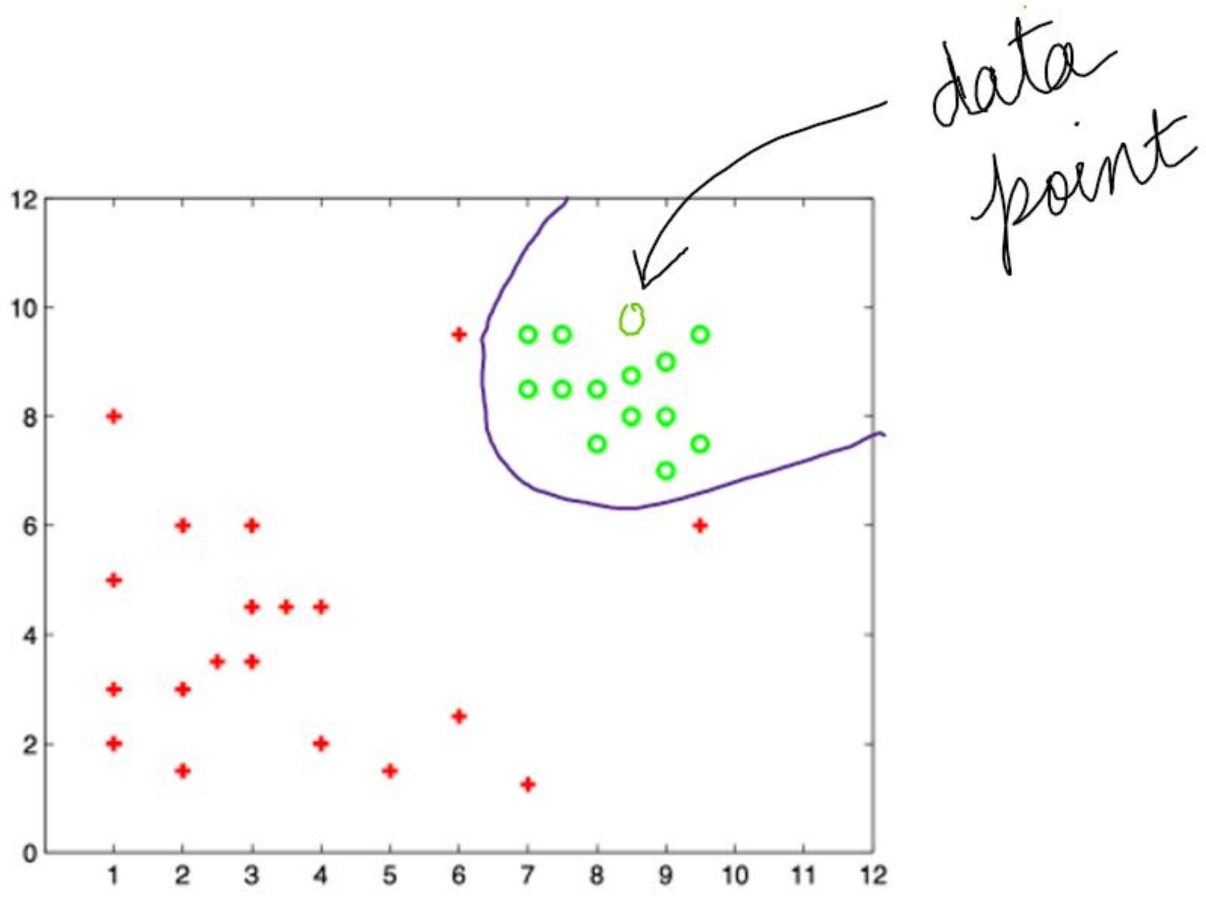


   (b) C is close to 0 means that the cost function is very low so cost for misclassifying (or in other words penalty) is low that means that decision boundary will misclassify few points as shown below:

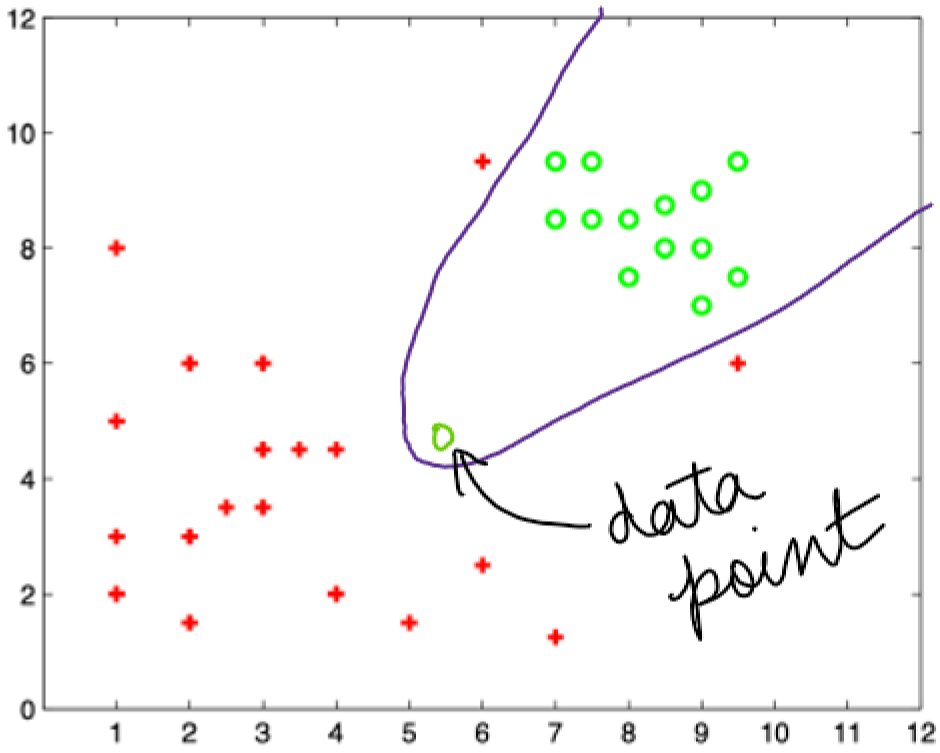


(c) As C close to 0 sort of marginalize the error between the major region of points, we think that it would work best to classify, that is assuming if we do not rely on one specific data point a lot.

(d) For very large values of C the datapoint made below will certainly not change the decision boundary made. This is because it is within the original correctly classified region.



(e) Now as question demands to change the decision boundary then, a point should be drawn as shown below as it would certainly force the decision boundary to change for C approaching infinity. Reason: As this point is added in the region, which was originally classified as red, so adding an opposite point that is green in that region will force the boundary to shift in order to correctly classify this new data point.



data point