ML 2015  
Section B

2. No, the data isn’t linearly separable in some places, there will be misclassifications, this could be resolved by using MLP.
3. Testing point (X’):   
   for each training data point X,  
    measure distance(X’, X)   
   end   
   Sort distance  
   Select K smallest distances (k-nearest)  
   Assign most common class.  
     
   Measure the distance using Euclidean equation: Sqrt(for i = 1: i =N)(Xi-X’i)^2)
4. The learning rule is the rule that optimise weights to get least amount of error,   
   “new weight = old weight + learning\_rate \*(TrueLabel - Output) \* Input”  
     
   user can alter the learning\_rate, having smaller learning rate is more computationally expensive, but can get optimal results, larger learning rate means it is faster to get the results but could end up doing a lot of corrections as it keep missing the sweet spot.  
     
   The weight is how important a certain feature is   
   learning rate: is user defined parameter that affects how fast the output converge to a stable solution.  
   TrueLabel: known output  
   output: the output label from the perceptron   
   input: feature of training data  
     
   The whole learning algorithm for a perceptron is   
     
   initialise weight with random numbers between -1 and 1  
   for n = 1 until Number of iteration  
    for each training example (x,y)  
    calculate activation  
    for each weight  
    update weight by learning rule  
    end  
    end  
   end
5. function BuildTree(subsample, depth)  
     
   // base case  
   if(depth == 0) or (all examples have same label)   
    return the most common label in the subsample  
   end if  
     
   // recursive case  
   for each feature do   
    try splitting the data   
    calculate gain for this stump  
   end for  
   pick feature with maximum information gain  
   find left/right subsamples  
   add left branch ← BuildTree(leftSubSample, depth -1)  
   add right branch ← BuildTree(rightSubSample, depth - 1)  
     
   return tree  
   end function   
     
   The largest information gain is done by calculating entropy before and after the split, and choosing the feature with lowest average entropy after the split.   
   Entropy = -1 \* (the sum of probabilities of all features, multiplied by the log of that probability)   
     
   The entropy is the amount of information contained in a variable.   
   We choose the feature with the maximum value of H(x) - H(x|w)   
   where H(x|w) is the weighted average of entropy on left and right of the data after we split.

----------

Section C

1. Description: Each data object its own cluster initially, then they are merged together progressively into larger clusters.   
   Steps;  
   1- Convert the object attribute into a distance attribute, making each object a cluster of size 1  
   2- Merge two closest cluster together  
   3- Update the distance matrix to take into account the new cluster  
   4- repeat until no more clusters to merge or reach a threshold   
     
   This algorithm is very sensitive to noise and outliers, also very slow computationally O(n^2) where n is number of objects
2. Requirements:   
   A large set of video clips :- a big n meaning Agglomerative is out of picture since it has O(n^2) run time. While k-means run in O(tKn) where t iterations and K is clusters will perform way better, consider upgrading to CLARA when you hire a more experienced data engineer.   
   Features to look for:   
   Tags (most relevant): User defined tags on each video clip, tags of similar nature should be shown when searched for or to be used by the recommendation system that you will hire a more experienced data engineer to build.  
   Title: Keywords in video clip title extracted with filler words omitted (the, and, or, etc…) groups of similar words should be grouped together.  
   Genre: Similar to tags, a user searching for comedy should get comedic results.  
   Authour: or uploader, all video clips by one user should be preferably grouped together even more so if all other features also align.   
   Upload date: some tags and keywords means different things in the old days of the internet, upload date should have influence on how clips will be grouped.  
   Views: low rated/low viewed clips shouldn’t be in same set as higher rated to make recommendation easier, and search result more relevant.  
   Rating : same as views  
   Duration: Longer\shorter videos should be grouped together as that what the user is expecting when searching for certain results, but I am not a UX designer so I don’t know.  
     
   those features transformed into a distance matrix using minkowskai distance, choosing P based on various factor such as what google tell me to choose, and how much I get paid in £££, and if I can be bothered to trial and error optimal one, otherwise I will just go with p = 2  
   d(x,y) = (|x1-y1|^p + |x2-y2|^p + …+ |xn-yn|^p)^1/p   
     
   For k-means, selecting a K is important, starting with k = sqrt(n) and going up and down based on results and trial and error.   
     
     
   Based on that applying the k-mean algorithm which is the following:  
   1- initialise seed points for each K cluster  
   2- Assign each data point to the cluster of nearest seed point measured with the distance matrix  
   3- compute new seed point as centroid based on the average distances between each partition   
   4- go back to step 2 until no more new assignments are made  
     
   repeat this for number of iterations t as the seed point change everytime   
     
   add cluster validation to get optimal value of K, internal indexes is easy to see by simply searching and seeing what results gets grouped together