**Data and Execution procedure**

The dataset considered for this experiment is taken from a file ‘Letter2Class.data’, which is present in the same folder where all other .m files are located and this data was accessed through these Matlab codes. In this data file the row header represents labels. The data consists of 16 types of attributes which represents a character ‘A’ or ‘X’. All the Matlab codes along with the data file are stored in a single folder. For all the questions same dataset is being used.

Using ‘importdata’ function the data is being read from the file. For all the 16 types of attributes representing ‘A’ or ‘X’ a total of 1576 instances are collected of which 789 instances belong to ‘A’ and remaining 787 instances belong to ‘B’.

**Question 1**

**Implementing the perceptron method using MATLAB**

The perceptron method, perceptron algorithm, or the perceptron of Rosenblatt (1962), is a linear discriminant model algorithm, which is used for pattern recognition. We consider the input vector with a 1 in the beginning to account for the bias. The model is a linear combination , of the inputs and a constant value which is then evaluated by an activation function.

Function is the nonlinear activation function:

And the classes are encoded as -1 and 1.

The objective of the implementation is to find vector so that it minimizes the error.

The weights and bias are first selected all equal to 0 then updated as

Where is the learning rate is the value of the kth element of the dataset, and is the prediction. The algorithm is repeated until the values of the weights are considered to converge or max iterations are achieved.

The implementation of the algorithm is in the file perceptron.m. The function is called perceptron. It takes a dataSet (struct) and learningRate (double) as inputs and returns hyperplane (array of double, coefficients of the weights) and counters, a struct that contains integers (TP, TN, FP, and FN).

function [hyperplane, counters] = perceptron(dataSet,learningRate)

Since there is no randomness in the procedure, the obtained values are always the same.

To try the function’s performance, a test script perceptrontest.m was implemented. This script creates two clusters of 2D data and labels them and the function perceptron is called. With this, a graphical representation of the algorithm is obtained.

**Question 2**

**Implementing a single-node decision tree using MATLAB.**

Decision trees are a decision support tool that uses a tree-like model of decisions. They are used to classify data according to its different components. They are called trees because of their graph representation. They are based in nodes that are arranged in a hierarchical manner where there is a starting node or root which branches to sub nodes and eventually end up in the final nodes which are called leaves.

When a single node is used, the root, it is also called a decision stump. This is the kind of structure that is implemented in this section. Since the training data has 16 variables, one of them has to be chosen to represent the whole sample. To decide which attribute will give the best split, information gain will be calculated for each attribute.

So to implement the algorithm, entropy of the whole dataset is calculated as:

The sum is done over all the classes (2 in this case) and is the probability of that class.

For each possible split, the sum of the entropy of each subset is calculated and the information gain for that split is then:

In this case K is 2 classes.

So to implement this, each variable is considered, and equally distributed thresholds within its variation range are tested with the data set. The entropy of each subset is calculated and the one with the smallest entropy (or equivalently greatest information gain) is selected as the best.

The implementation of the algorithm is in the file singlenodedecisiontree.m. The function is called singlenodedecisiontree. It takes a dataSet (struct) as input and returns counters, a struct that contains integers (TP, TN, FP, and FN), kFieldBest (integer) that is the index in the data set of the field that obtained the best information gain, thresholdBest (double) that is the decision threshold for that vaiable, and inforationGainBest (double) which is the information gain for the best split

function [counters, kFieldBest,thresholdBest,InformationGainBest]= singlenodedecisiontree (dataSet)

To try the function’s performance, a test script singlenodedecisiontreetest.m was implemented. This script creates two clusters of 2D data and labels them and the function singlenodedecisiontree is called. With this, a graphical representation of the algorithm is obtained.

**Question 3**

**Write functions to evaluate the perceptron and decision tree methods using MATLAB.**

The testing scripts both create two cluster of 2D data points, label them and call the implemented functions and graph the decision plane when useDataSet is false, the results can be seen in the following Figures.

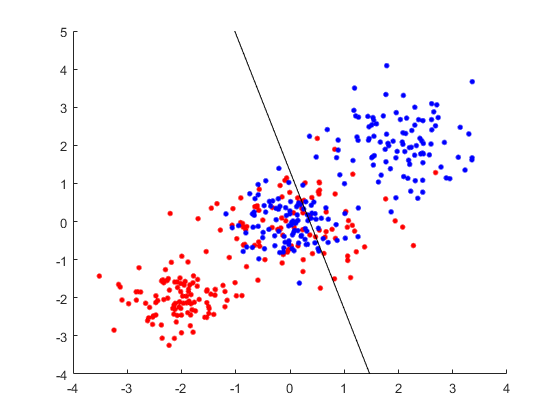


Figure 1: clusters and decision plane for perceptron

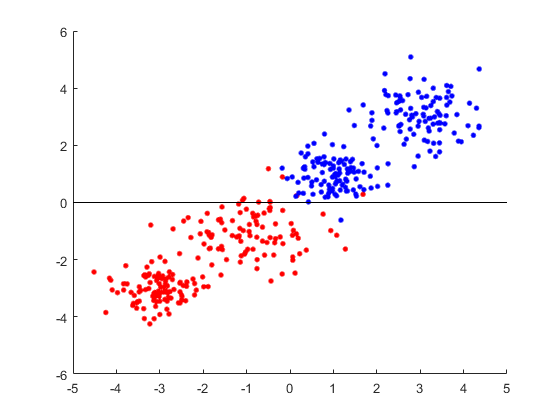


Figure 2: clusters and decision plane for single node decision tree

When useDataSet is set to true, both test srcipts (perceptrontest.m and singlenodedecisiontreetest.m) perform cross-validation and calculate accuracy, sensitivity, and specificity.

Standard deviation and average are computed in a separate worksheet to be able to see the numbers.

**Cross-validation**

To perform cross-validation, 5 runs for each algorithm are considered, each one using 70 % of the data selected randomly with reposition



Table 1. Cross validation for perceptron, including average and standard deviation



Table 2. Cross validation for single node decision tree, including average and standard deviation

**Accuracy**

To calculate accuracy, the following formula is used:

Where:

is true positive = X classified as X  
 is true negative = A classified as A  
 is false positive = A classified as X  
 is false negative = X classified as A

For the perceptron the accuracy is 0.9994

For the single-node decision tree the accuracy is 0.9270

**Sensitivity**

To calculate sensitivity, the following formula is used:

For the perceptron the sensitivity is 0.9987

For the single-node decision tree the sensitivity is 0.9936

**Specificity**

To calculate specificity, the following formula is used:

For the perceptron the specificity is 1

For the single-node decision tree the specificity is 0.8606

**Auxiliary functions**

An auxiliary function labelsXAtoTarget1minus1 was implemented in the labelsXAtoTarget1minus1.m file which takes the dataset and returns an array of 1’s and -1’s for the labels “X” and “A” respectively.

function targetValues = labelsXAtoTarget1minus1(dataSet)