CSCE 5216 Pattern Recognition

Final Project

Decision Tree Implementation

Pruthvi Teja Anumandla (11194349)

Anoop Sai Vengala (11058492)

Table of Contents

[**1.** **Summary** 1](#_Toc532234744)

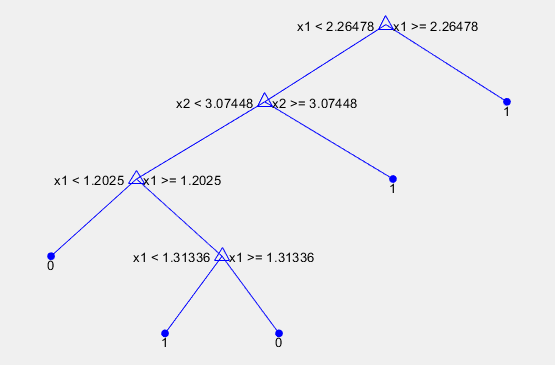
[**2.** **Decision Tree Generation** 2](#_Toc532234745)

[**3.** **Implementation Plan** 3](#_Toc532234746)

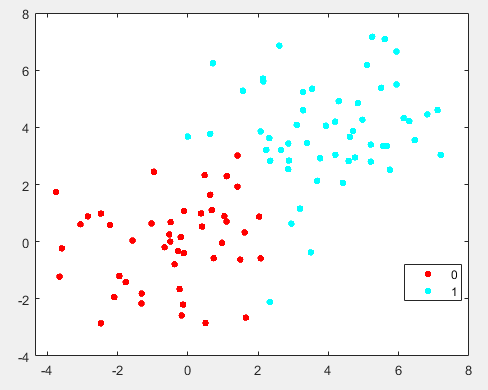
[**4.** **Implementation Report** 4](#_Toc532234747)

# **Summary**

A decision tree is drawn upside down with its root at the top. In the image shown below, the text in black represents a **condition/internal node**, based on which the tree splits into **branches/ edges** as shown in blue lines. The end of the branch that doesn’t split anymore is the **decision/leaf** which are represented by blue dots.



The data points spread in space shown below are split by decision boundaries based on above conditions so any test data which falls in one of boundary is classified as (1, 0) with help of the decision tree.



# **Decision Tree Generation**

* 1. **Tree Construction**

In this procedure of tree construction all the features are considered, and different split points are tried and tested using a cost function, the goal of the decision tree is to result in a set that minimizes impurity. The data set is then split on different variables as shown in above decision tree figure until we arrive at a subset where everyone in that subset either belongs to 1 or 0. Ideally, after traversing our decision tree to the leaves, we should arrive at pure subset. We have chosen entropy as criteria to calculate the information gain and decide the split.

Entropy is the measure of information in an attribute and it is given as below

**Entropy (X) =**

Information gain is the amount of information that's gained by knowing the value of the attribute, which is the entropy of the distribution before the split minus the entropy of the distribution after it.

**Gain (X) = Entropy(X) -**

We begin calculating information gain for each attribute of the features present one by one, select the attributes with highest gain among within each feature and compare it with highest information among other features to decide the root node.

We build the left and right nodes of the tree from root node by recursively splitting the nodes based on information gain criteria till the subset of nodes split are homogenous.

* 1. **Tree Pruning**

By visualizing the performance of decision tree using cross validation we can analyse if tree is over fitting. To make the decision tree able to generalize on the new unseen data we prune the extra grown branches, we prefer post pruning at this instance.

We are also planning to implement Random data generation to test the decision tree on various dimensions and number of classes. We calculate the performance of the tree based on its performance on cross validation.

# **Implementation Plan**

October 28 - Entropy Calculation (***Pruthvi***)

November 4 – Information Gain, Comparison of Info gain among features (***Anoop***)

November 11 – Splitting nodes (***Pruthvi***)

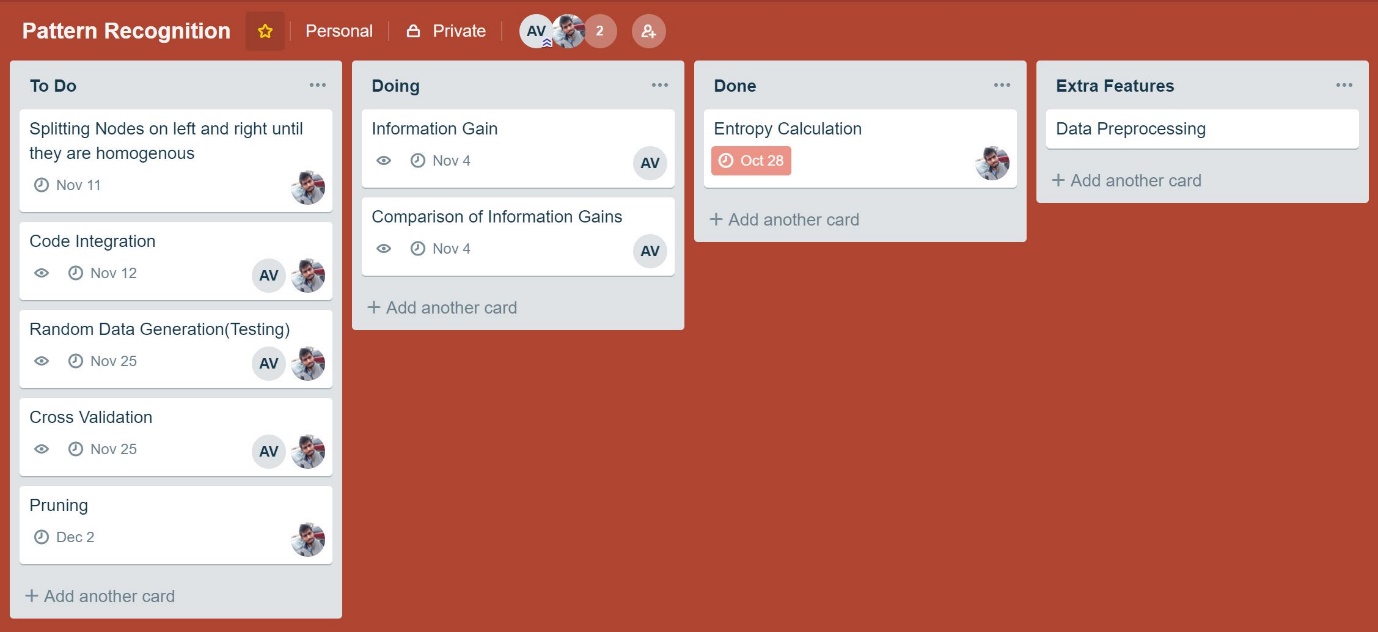
November 12 – Code Integration and testing (***Both***)

November 25 – Random data generation for testing (***Anoop***) and Cross validation (***Pruthvi***)

December 2 – Pruning and code clean-up (***Both***)

December 6 – Project Presentation

December 10 – Final Project Submission



# **Implementation Report**

* 1. **main.m**

This is the main file where our program execution starts. We read our dataset and continue with the following steps.

* + 1. Different methods like xlsread and readtable are employed to read data.
    2. If the data contains categorical data, then we convert data from categorical to numerical.
    3. Spilt data based for cross validation randomly each time.
    4. Send the training data to recursive function decisionNodeSplit to build the tree, splitconditions is the decision tree obtained.
    5. Sending test data for prediction the labels and calculation performance.
    6. Plot decision tree.
  1. **categoricalToNumerical.m**

This file takes our dataset as input and convert all categorical data to numerical and stores them in the form [features, label].

* 1. **decisionNodeSplit.m**

This function takes our dataset, split condition and counter as input and build the decision tree recursively and populating the split conditions and using counter for denoting the depth of the tree. We iteratively split the dataset into left and right nodes by splitting the dataset on data point of feature with highest information gain across the dataset and so on till we reach the leaf nodes which are pure.

* 1. **InfoGainAcrossFeatures.m**

This function takes our features list and labels as input and output as best feature and particular data point on which we get highest gain as output. In this function we calculate Information gain on each feature and then return feature with highest gain among all these features.

* 1. **InfoGainOnFeature.m**

This function takes our single feature and label as input and calculates on which point we get highest entropy among the data points and returns the gain and point at which we attained the highest gain.

* 1. **calculate\_Entropy.m**

This function takes a set of count of different input labels and calculates the entropy using the equation and return the value.

* 1. **splitDataSet.m**

This function takes the dataset, split data point and the split column and divides the dataset based on the split point and split column and returns children datasets left and right.

* 1. **purityCheck.m**

This function checks if the given datasets are unique or they are still impure, if the labels are unique we return 1 and else 0.

* 1. **evaluation.m**

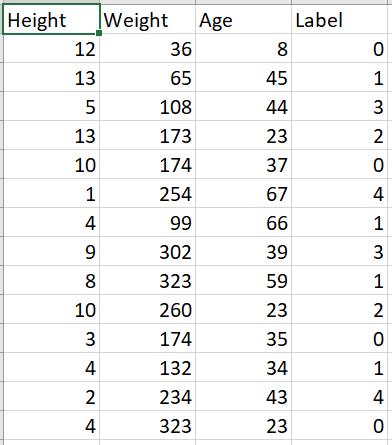
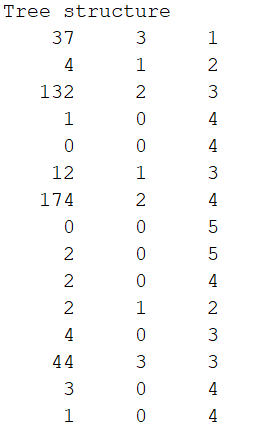
This function takes test set and tree structure as input and predicts the decisions and returns all the results.

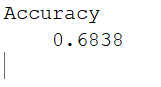
* 1. **Predict.m**

This function takes our data instance to predict decision, tree structure and index. We traverse decision tree reclusively and arrive at the leaf to return the decision.

**Results**

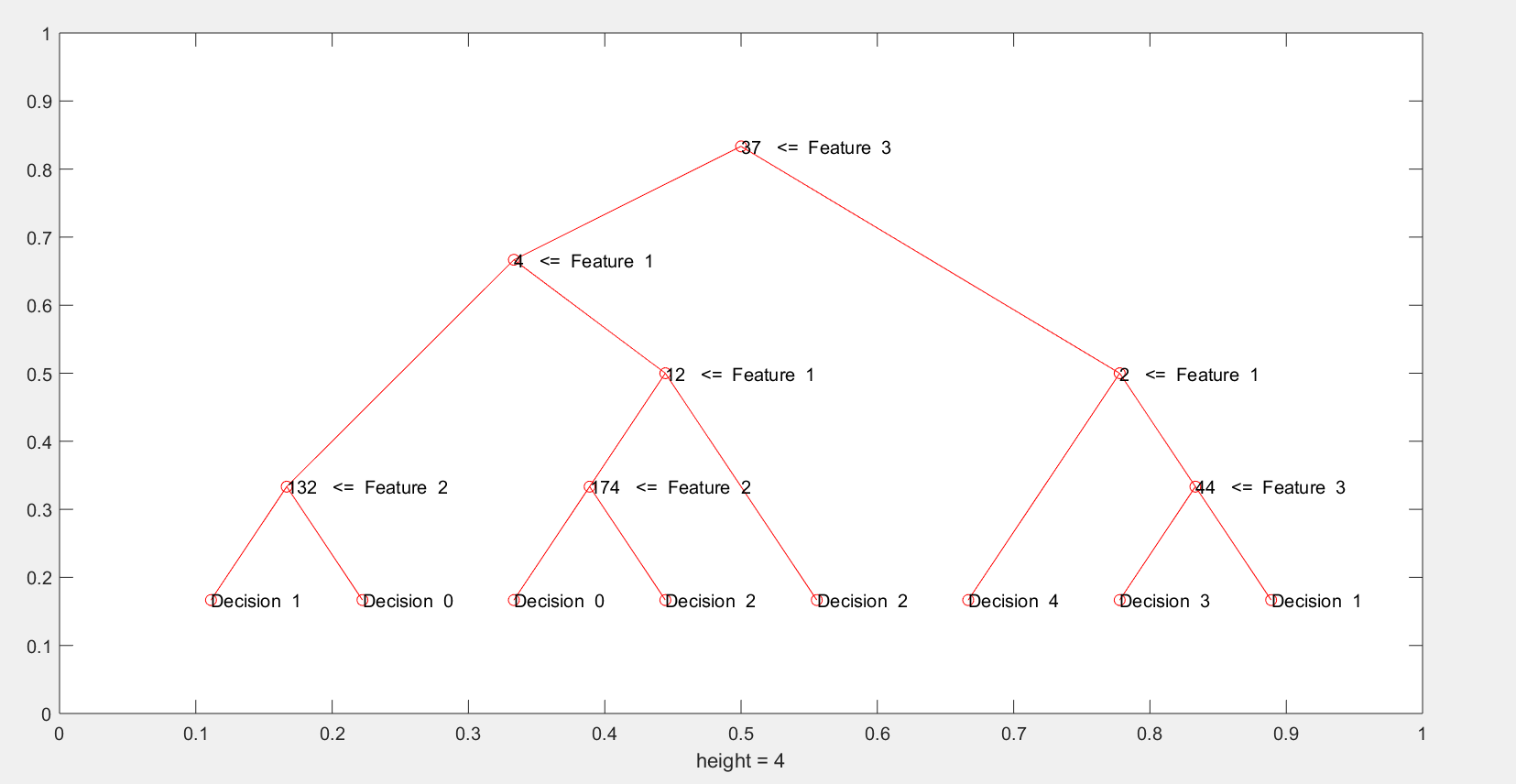
Implemented decision tree initially using the features and data given in slides and then made it compatible with breast cancer dataset given.



Accuracy on the above dataset.

***Decision tree for above data***



***Decision tree for breast cancer data***

