**Project 1 :**

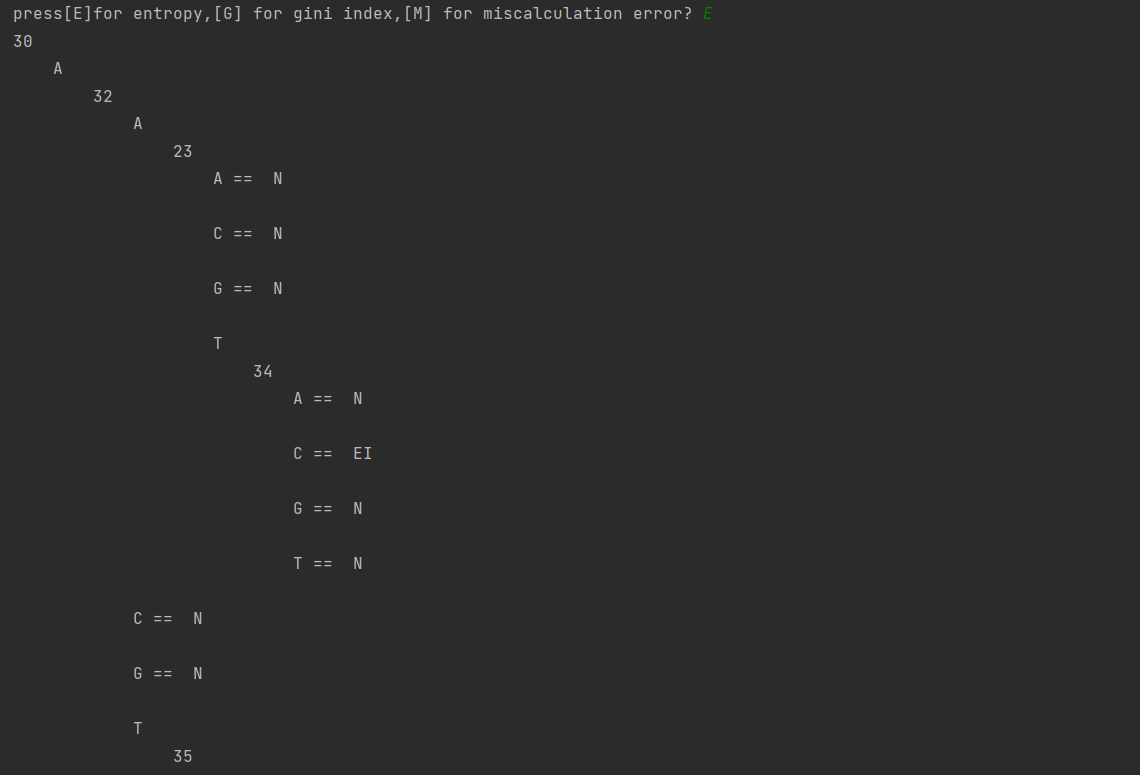
**Decision Tree**

**Karan Aryal**

Description of How the code works:

The decision tree is built upon using ID3 implementation. First of all, the data from the training set is recorded and stored on a data frame. When running the program, the user is prompted with this following message “press[E]for entropy,[G] for gini index,[M] for miscalculation error?” on the command line terminal. When the user enters a desired key for the calculation of error, the decision tree is built accordingly. One thing to notice is that is the user inputs wrong key then the tree is built in the default mode that is with entropy.

During the formation of each node, Chi square test is carried out checking the randomness of the data. Also, errors calculations like gini index, entropy and miss-classification error are performed. Each error test can be used to create different trees respectively. Split stopping accuracies of 99, 95 and 0 confidence are also employed.

Image of the Decision Tree with Entropy:

As the decision tree is very large, partial tree is shown in the above figure.

Working of the program:

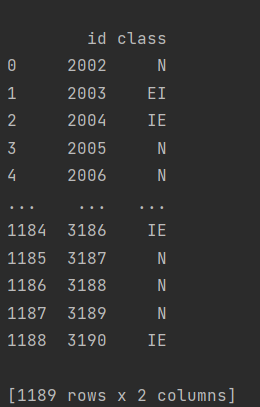
First and foremost, the input is read from the user. Depending upon the input error are calculated and tree is constructed. After that csv files are read for training and testing of the data. The file is split into 1 – 60 attributes while dropping column ‘1’ and ‘N’ assigning the new column as ‘result’. Same this is done for the test data as well.

After those different methods are used for calculation of entropy, Gini index and miss-classification error. Max information gain is calculated using the above error technique and later implemented on the tree. Chi square test is also used to determine the randomness of each attribute on the node, so that we don’t have to calculate error every time if the split is completely random.

The tree is created with the implementation of ID3 algorithm. Using recursion and backtracking each node is filled with the certain attributes whilst following the rule. After the formation of the tree, the tree is later printed and showed in the command line terminal.

The detailed usage of each method is explained in the README.md file.

The program after successful implementation of decision tree and predicting new values.



Chi square Test:

This test performs to check the randomness of the data. The degree of freedom is calculated 0.99, 0.95, 0.001 confidence level. If the value of chi square test is less than the degree of freedom, then the node will not split. It will be a leaf node. If the value if greater than degree of freedom, then we do the usual split.

Options which work best:

Chi square test with degree of freedom, gini index, entropy and the algorithm for the decision tree are the features which works the best. As the algorithm doesnot take a lot of time while implementing these methods and are very fast.

Low accuracy reasoning:

The reason our accuracy might be low is because of ineffective use of miss- classification error as I feel that implementation is not so accurate in our code which leads to low accuracy. We suspect that miss-classification error handling is very ineffective, and this also slows down the formation of the tree. Whilst with gini index and entropy the tree is formed in less than 30 seconds, with miss-classification error the tree takes nearly 2 minutes to be created. We think this is the drawback of our project.