Neil Shah - 301356829

Assignment 1 - CMPT 459

Assignment 1.1 [20 marks for a), 20 marks for b)]

In class, we have covered the most common type of decision trees, namely decision trees that predict a categorical variable. These decision trees are also called classification trees. In this assignment, we will work on another type of decision trees, called regression trees. We consider the special case of a regression tree for numerical data (both the features, x, and the variable to be predicted, y, are numerical).

a) The metrics for choosing a split attribute for a classification tree (information gain or gini-index) are not meaningful if y is a numerical variable. Propose a metric for the quality of a split attribute for a regression tree.

Ans:

1. **Reduction in Variance**
   Variance is used as a measure for deciding the feature on which node is split into child nodes. We want minimum variation in the nodes after the split.

$$Variance = \frac{\sum (X - \mu)^2}{N}$$

   Method:
   a. Calculate variance for each child node of each split
   b. Calculate the variance of each split as the weighted average variance of child nodes
   c. The lower variance split will be selected
   d. Repeat a-c until all homogeneous nodes are achieved

2. Classification and Regression Trees (CART)
   Splitting Criterion: Sum of Squared Error (SSE)
   Select the variable/value **($X=t_1$)** that produces the greatest "separation" in the target.
   a. **($X=t_1$)** is called a "split".
   b. For Regression Trees: we use SSE (sum of squared errors)
   c. Find split that produces greatest separation in **$\sum[y - E(y)]^2$**
   d. The split is determined by testing every value of all variables, the one which minimizes the sum of squares error (SSE) best is chosen.
   $$SSE = \sum_{i \in S1}(yi - \bar{y}1)^2 + \sum_{i \in S2}(yi - \bar{y}2)^2$$
   e. yi -> predictors value
   $\bar{y}1$ & $\bar{y}2$ -> mean value of left and right hand side of the possible split
   f. The one minimizing SSE best, would be chosen for split

b) In the simplest approach, the regression tree labels a leaf node (and all test examples that fall into this leaf node) by the mean of the y-values of all training examples whose x-valued correspond to this leaf node. For the case that the features (attributes x) are all numerical, propose a more accurate approach to predict the y-value for a test example. What is the drawback of this approach?

Ans:

Recursive Feature Elimination with cross validation is more accurate approach. Builds a model on the entire set of predictors and then for each predictor, calculates an importance score. The predictor with the lowest importance score is removed and then steps 1-2 are repeated. The optimal subset which is used to train the final model is chosen by finding the size that optimizes the performance criteria. One drawback of this technique is that

calculation of importance is related to multicollinearity. If predictors are highly correlated, choosing predictor for partitioning samples becomes a random selection. This results in all highly redundant and useful predictors being used in splits. Hence, the importance scores are diluted as a result of redundant features.

Assignment 1.2 [25 marks for a), 35 marks for b)]

a) Present the pseudo-code for a simple Random Forest for categorical data (no numerical attributes). The method for training and testing a Random Forest should have the following input parameters

TrainAndTestRandomForest(trainingdata, numberOfTrees, percentageOfAttributes, testdata)

and return a file with predictions for the test data.

The method should do the following:

- Train numberOfTrees different trees. [5 marks]
- The trees are grown deep, i.e. they are grown until all training examples corresponding to a leaf node belong to the same class. [5 marks]
- Each tree is trained on the entire training dataset. [5 marks]
- For each tree, a different randomly selected percentageOfAttributes is considered as potential split attributes. [5 marks]
- The information gain (or entropy) is used as a split criterion. [5 marks]

Ans:

```
TrainAndTestRandomForest(trainingdata, numberOfTrees, percentageOfAttributes, testdata){
    forest <- empty dictionary/list.
    FOR i from 1 to numberOfTrees do
        randomly select percentageOfAttributes from trainingdata
        IF informationGain != 0
            input this randomly selected subset into decision_tree
            calculate the information gain using informationGain
            find best split and generate children nodes
            forest[i] <- decisionTree
            repeat IF condition until condition not met
        ENDIF
    ENFOR
    FOR each tree in forest:
        outputFileCSV <- predict(data, decisionTree)
    outputFileCSV <- numberOfTrees and percentageOfAttributes
    ENDFOR
}
```