Elisabeth Webb

CA03 Decision Tree Algorithm

BSAN6070

Professor Brahma

Answered Questions

Q.1.1 Why does it make sense to discretize columns for this problem?

The continuous data columns have been transformed into bins. This is called discretization which is a data reduction method. It takes all the numeric values in the column and groups it into discrete categories. The decision tree algorithm can handle both continuous and categorical variables, but discretizing the continuous columns makes the data more uniform by reducing the fluctuations in values. For the model, the smoother data allows for better predictions.

Q.1.2 What might be the issues (if any) if we DID NOT discretize the columns?

If we did not discretize the columns there would be more fluctuations in the results. Decision trees are sensitive. Small variations can cause large changes in the structure of the tree. If we did not discretize the continuous columns it could lead to potential instability and variations in the results. Discretizing allows us to be more confident in the results.

Q.7.1 Decision Tree Hyper-parameter variation vs. performance

I first looked at how the decision tree preformed with only specifying the random\_state=100 hyperparameter and using the defaults of criterion = “gini”, min\_samples\_split = 2, min\_samples\_leaf = 1, max\_depth = None. Then I experimented with adding values to each of these hyperparameters to see how they would influence the performance metrics. I noticed that at a certain point, the metrics plateaued and stopped improving.

Performance without specifying hyperparameters (except random\_state=100).

Accuracy: 0.84

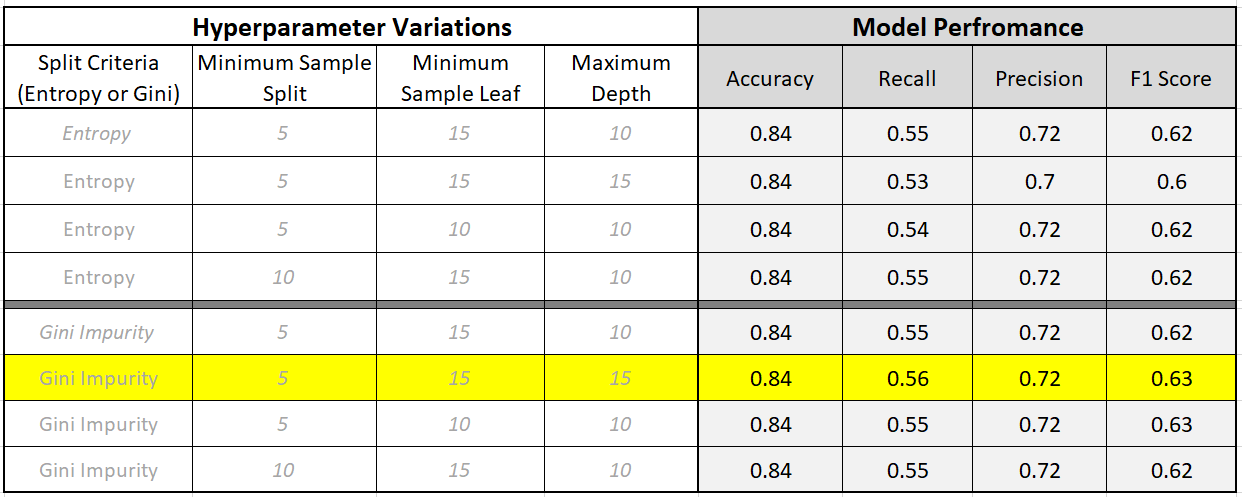
Precision: 0.71

Recall: 0.54

F1 Score: 0.62

AUC Value: 0.74

The table below shows how changing the hyperparameters affects the model’s performance. The yellow highlighted row is the best preforming tree in the table.



Q.8.1 How long was your total run time to train the model?

To run the best tree that will be used as the trained model for predicting, it took 101 ms.

time: 101 ms

Q.8.2 Did you find the BEST TREE?

I found the tree that produced the highest metrics using the hyperparameters I experimented with. The best tree I found is shown below.

dtree\_best = tree.DecisionTreeClassifier(criterion='gini', min\_samples\_split=5,min\_samples\_leaf=15, max\_depth=15)

precision recall f1-score support

0 0.87 0.93 0.90 12435

1 0.72 0.56 0.63 3846

accuracy 0.84 16281

macro avg 0.80 0.75 0.77 16281

weighted avg 0.84 0.84 0.84 16281

Q.8.3 Draw the Graph of the BEST TREE Using Graph Viz



Images of the original tree (census.png) and best tree (census\_best.png) are in my elisabethgwebb / BSAN6070-CA-Webb / CA03 Git Hub folder.

Q.8.4 What makes it the best tree?

This tree has the highest accuracy score, precision, recall, and f1 score. When evaluating the models, I looked for the model with the best f1 score. F1 score indicates that precision and recall are balanced, and the higher the f1 score, the better the model will be. Comparing the recall (0.56) and precision (0.72) scores showed that they are somewhat balanced in their distance to one another. I also looked at the accuracy and found that this tree had the highest number of correctly predicted classifications in terms of the score.

Q.10.1 What is the probability that your prediction for this person is accurate?

First, I added the new individual’s data into the original data frame, and then transformed it into dummy variables and split the data into train and test. Using the trained predictor, the model predicted that the y value (income level) of the new individual (which is in the last row of the data frame) will be 1 (>50K). The array produced by the model\_best.predict\_proba(x\_test), shows the probabilities for the y variable in each row being either [0, 1]. The results show that there is a 6.6% chance of the prediction being 0 (<=50K), and a 93.3% chance of the prediction being 1 (>50K). These probability results indicate that the prediction was pretty accurate in classifying correctly.

[0. 0. 0. ... 0. 1. 1.] #The model’s prediction

[[1. 0. ] #Array showing the probability of y being [0,1]

[0.60855263 0.39144737] #The new individual was entered in the last row

[0.80058651 0.19941349] #So we look at the last list or item

...

[0.66666667 0.33333333]

[0.26666667 0.73333333]

[0.06666667 0.93333333]]