1. Names:
   1. Art Grichine – ArtGrichine@csu.fullerton.edu – 50%
   2. Adam Beck – AdamJBeck@csu.fullerton.edu – 50%
2. A brief description about the software tool you used or program you implemented.
   1. Our software consists of a Python script which was created by Chris Strelioff and is found at: <http://chrisstrelioff.ws/sandbox/2015/06/08/decision_trees_in_python_with_scikit_learn_and_pandas.html>

This link leads to a tutorial of how to create decision trees in python with the ‘*scikit-learn’* and ‘*pandas’* libraries. The tutorial uses a data set for flowers. We modified the script to fit our data set and project goal. Our script follows a different pre-processing for the data as well as different settings to the decision tree (Entropy vs. Gini).

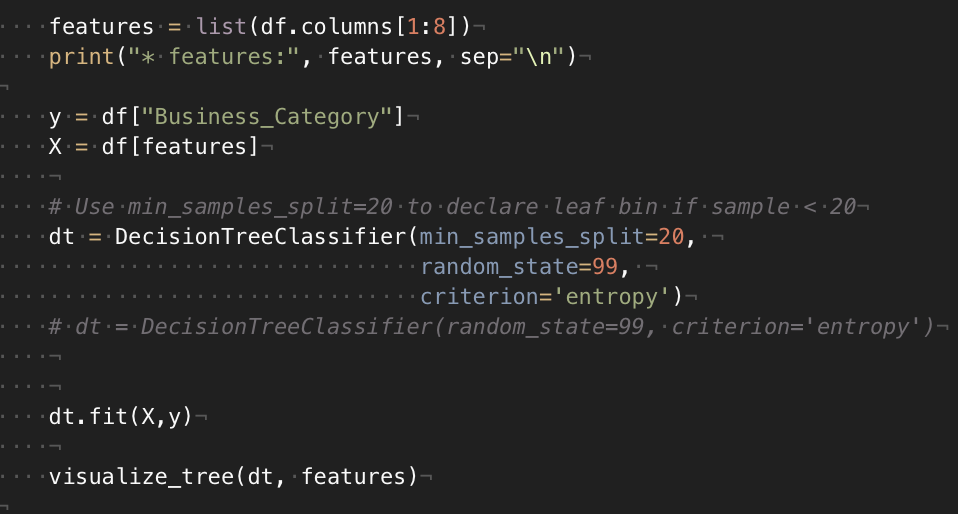
The heart of the program uses the ‘pandas’ library to extract the information from the csv dataset and place it into a pandas DataFrame data structure. The class column is defined as the ‘BusinessCategory’ column. Then, the DecisionTreeClassifier module from the SciKit-Learn library is used to train the model based on the defined parameters. Once the model has been trained, it is exported to the graphviz module (which is contained in the SciKit-Learn library) for graphical output. Also, a get\_code() function was written to generate the rules for the given tree. This code is output to the screen as well as a file called ‘code.txt’.

Note: Implementation of our tree uses a bin (node) size of 20. This means that a leaf node will not split if it contains <= 20 samples. This greatly reduces the complexity and size of the tree. Without this, the tree would split many more times for certain nodes just to accommodate a single case. This grows the tree size many more times and reduces the readability of the tree.

Understanding the graph:

A leaf bin (node) contains 3 parameters: Entropy, Samples, and Value. Entropy is the amount of entropy that is found within the sample size of the current bin. An example of this would be that if all samples belong to a single class, the entropy would equal 0. This can be seen in the far left bin of Figure 1 where all 200 samples belong to class 1 (Restaurant) therefore the entropy is equal to 0. Conversely, the bin that shares the same parent bin as the one to the far left has an entropy value of 0.65 with 6 samples and a value of [5,1]. As mentioned above, bins with <= 20 samples do not split by design to reduce the size of the tree therefore this bin contains 6 samples but even though it’s a leaf node it does not contain all samples of a single class. Here, value = [x,y] represents the distribution of the sample size where x is the amount of samples that belong to the Restaurant class and y is the amount of samples that belong to the Retail class.

1. The results of your data analysis including the following elements:
   1. The name of the class/decision column for your analysis.
      1. BusinessCategory
   2. A brief explanation of the process of your analysis listing a series of steps taken, specifying input and output if there is any, to produce the final decision tree from this data set.
      1. To train the decision tree with SciKit-Learn’s Decision Tree tools, we needed to identify the class column (‘BusinessCategory’ – Column 1) and the rest of the data (Columns 2-7) as the training data. Also, we must initialize SciKit-Learn’s DecisionTreeClassifier module. We indicate minimum sample split size of 20 (bins don’t split if they have less than 20 samples so as to not create a very large tree), and set the criterion to entropy (vs Gini). Then, our decision tree is trained using the fit() module which takes in two parameters: class, attributes. We then visualize our decision tree using GraphViz. Implementation of our decision tree can be seen below. Also, entire code is found at the end of this document.



* 1. A snapshot of the final decision learned. If the tree is too big, show only the important part of the tree based on your judgment (to meet the managers’ expectations).
     1. Attached to document (below) on next page (Figure 1)/(Figure 2).
  2. Give two rules you can obtain from the decision tree that can be used to explain the nature of the data. If a rule is too big, you can show only the important part again based on your judgment.
     1. Rule 1:
        1. IF (Detergent <= 1746.5 Grocery <= 4480.5 AND Delicatessan <= 3403.5) OR (Detergent <= 1746.5 AND Grocery > 4480.5 Detergent <= 507.5) OR (Detergent <= 1746.5 AND Grocery >)
     2. Rule 2:
     3. Full rule representation is generated by the get\_code() function and printed to a file named code.txt. Also, this output is attached (below) at the end of this document.
  3. A brief explanation of knowledge learned from the data analysis in layman terms so that two managers can understand.

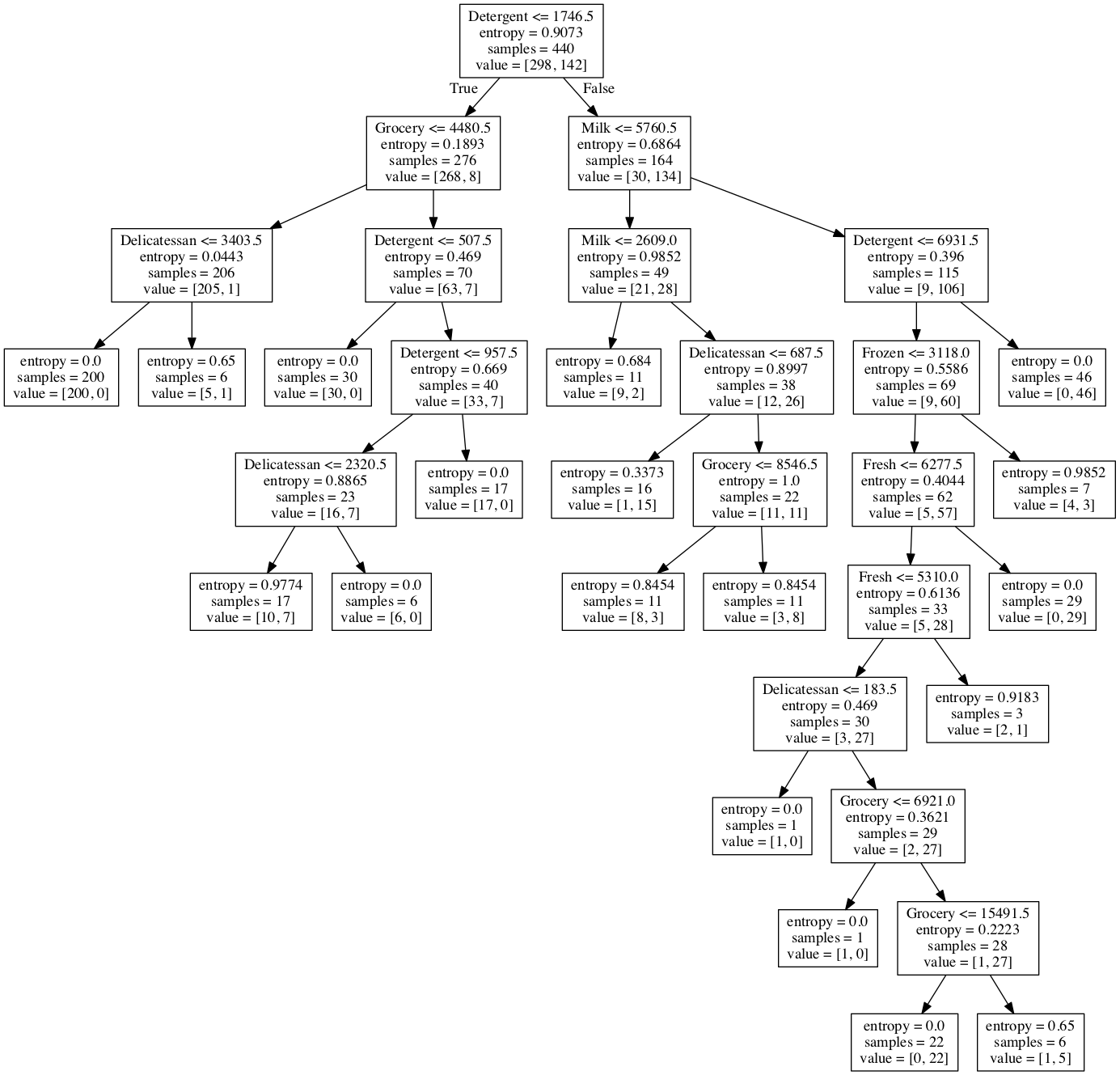


Figure 1 – Decision Tree

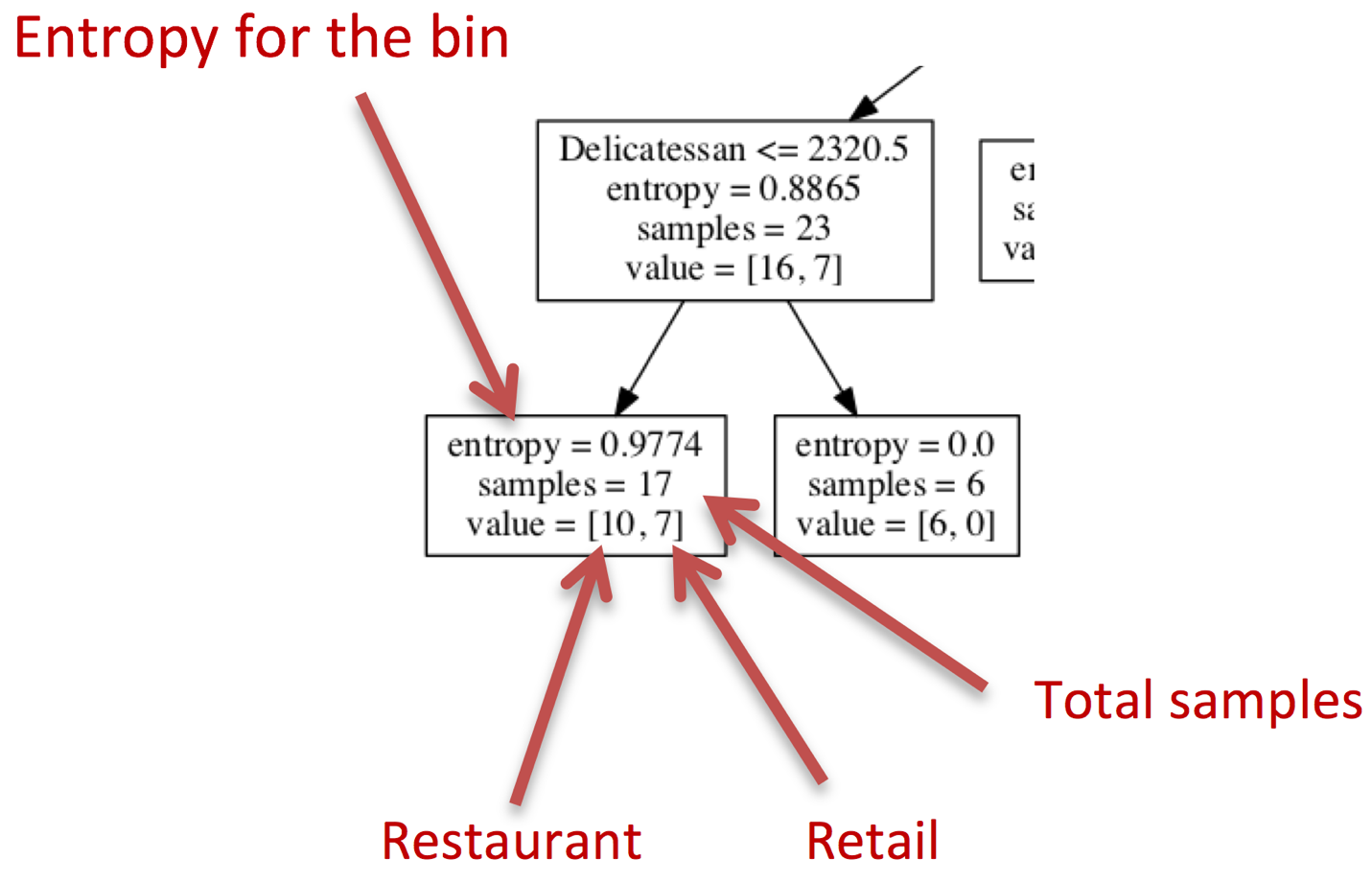


Figure 2 - Detailed view of leaf bins

**Rules generated from the decision tree via get\_code() function: code.txt**

if ( Detergent <= 1746.5 ) {

if ( Grocery <= 4480.5 ) {

if ( Delicatessan <= 3403.5 ) {

return Restaurant ( 200 examples )

}

else {

return Restaurant ( 5 examples )

return Retail ( 1 examples )

}

}

else {

if ( Detergent <= 507.5 ) {

return Restaurant ( 30 examples )

}

else {

if ( Detergent <= 957.5 ) {

if ( Delicatessan <= 2320.5 ) {

return Restaurant ( 10 examples )

return Retail ( 7 examples )

}

else {

return Restaurant ( 6 examples )

}

}

else {

return Restaurant ( 17 examples )

}

}

}

}

else {

if ( Milk <= 5760.5 ) {

if ( Milk <= 2609.0 ) {

return Restaurant ( 9 examples )

return Retail ( 2 examples )

}

else {

if ( Delicatessan <= 687.5 ) {

return Restaurant ( 1 examples )

return Retail ( 15 examples )

}

else {

if ( Grocery <= 8546.5 ) {

return Restaurant ( 8 examples )

return Retail ( 3 examples )

}

else {

return Restaurant ( 3 examples )

return Retail ( 8 examples )

}

}

}

}

else {

if ( Detergent <= 6931.5 ) {

if ( Frozen <= 3118.0 ) {

if ( Fresh <= 6277.5 ) {

if ( Fresh <= 5310.0 ) {

if ( Delicatessan <= 183.5 ) {

return Restaurant ( 1 examples )

}

else {

if ( Grocery <= 6921.0 ) {

return Restaurant ( 1 examples )

}

else {

if ( Grocery <= 15491.5 ) {

return Retail ( 22 examples )

}

else {

return Restaurant ( 1 examples )

return Retail ( 5 examples )

}

}

}

}

else {

return Restaurant ( 2 examples )

return Retail ( 1 examples )

}

}

else {

return Retail ( 29 examples )

}

}

else {

return Restaurant ( 4 examples )

return Retail ( 3 examples )

}

}

else {

return Retail ( 46 examples )

}

}

}

CODE:

