Big Data Analytics

Nihal Surendra Parchand

np9603@rit.edu

Homework 5 Building a One Rule Classifier

The goal of this assignment was to write a program that writes another program which gives a set of classifications and then generates a csv file that predicts the value for the target variable. The csv file will contain 0 for all rows where target variable is False and 1 for all rows where the target variable is True.

Data cleaning:
The data was collected from the Obtuse Quiz and a csv file was generated. The initial csv file contained a lot of duplicate answers, misinterpreted answers, bad data.

For example:
Duplicate answers: cookies n creme and cookies and cream
Misinterpreted answers: 28 for median sleep which could mean that the answer should have been 28/7=4 hrs
Bad data: Values like ‘8-Jun’ which got converted because of the excel. The actual value was 6-8.
dont Êknow was converted to dont know. Answer for median sleep in one row was 0 which is not possible. So I converted the answer to N/A.

Some columns were removed because all values were same and such columns are useless for classification(Polydactyl? (6 fingers?)','Who invented the first compiler?','Youngest Nobel Laureate', 'Log2(1024)').

Also, some columns were removed as mentioned in the email (UID, FavColor, Musical instrument).

Replacing values using .loc
clean_data_df.loc[clean_data_df['Median Sleep'] == '7:30', 'Median Sleep'] = '7.5'

Finding location of median sleep where median sleep is ‘7:30’ and then replacing it with ‘7.5’.

Replacing values using regex
clean_data_df['Airspeed of Swallow'] = clean_data_df['Airspeed of Swallow'].replace(to_replace =['.*african or european.*'], value ='african or european swallow?', regex = True)

Replacing different answers containing the regex pattern ‘african or european’ where it can start from anything and end with anything and then replacing it with ‘african or european swallow?’

Using str.strip() to remove leading and trailing white spaces.

Using str.lower() to convert all answers to lowercase.

There was some confusion as to whether I should delete one of the two columns ‘Left or right foot going up stairs?’ and ‘Right or left Footed? Going up Stairs?’ as they both ask the same question but the responses were different. Hence, I did not delete these columns. Same with Watch Hand?

Finally the clean data was stored in clean_data.csv.

Finding the best attribute:
For finding the best attribute, we iterate through each column in the column list except the target attribute column. First, we store the unique values of each column in a list and then initialize two dictionaries (true and false) with keys as unique values of each column and its value initialized to 0. Now, for each row in the dataframe, we check the value for the target attribute at that same index. If the value for target attribute == peppers, then increment the value of key in true dictionary. Else increment the value of key in false dictionary. This results in dictionary with key value pairs having key as unique values of each column and value as count of matches or misses for the target variable.

Now, for finding the misclassification rate for each column, we iterate through the two dictionaries (true and false) and for each key, we check the minimum of true values and false values and add it to the missed values. The resulting missed values is divided by the total number of observations to get the misclassification rate. We find the minimum misclassification rate and the best attribute. For breaking ties for same misclassification rates, I chose the < sign which means I chose the attribute which comes first in the dataframe column list. For my assignment, School Club? and Favorite flavor of ice cream? both had the same misclassification rate = 0.0375.

```
Best attribute is School Club? with Misclassification rate = 0.0375
```

Building the One-Rule:
I created a dictionary which stores the key value pairs where key is the unique value of each column and value as 1 if the true values for that unique value is greater than the false values and 0 if vice versa. I also appended the keys to two lists for storing which unique values gives true values and false values to generate One-Rule accordingly.

Program which creates another program:
In this step, we write actual python code in f.write() as text. This code includes code for importing libraries like pandas. Then we start with the main() function which includes reading the clean csv file and storing it in a dataframe, initializing a result list which will be used to display the final output (0 or 1). We iterate through the values of each record for the best attribute and then write a one-rule for each unique value of that attribute.

```
if value == 'environment':
    result.append(0)
```

The value to be appended is 1 if there are more true values, else 0 if there are more false values. These values are retrieved from the dictionary created above. The final result is then stored in the csv file.

Accuracy:

```
One rule :
if (School Club?==socialservice):
    Most Favorite Pizza Topping? = true
else:
    Most Favorite Pizza Topping? = false
```

According to the one-rule generated, if school club == socialservice, then peppers is true.
There is only 1 social service value in the School Club? Column for given dataframe. Hence, the accuracy is 1 - 0.0375 = 0.9625 (96.2%).
The false alarms were 0.
The misses were 0.

Conclusion:

In this assignment, the first thing we learned was how to clean the data that is provided and it takes 70% time in the data analysis task. I also learned how to use different methods like strip, replace,strip. I learnt how to calculate the misclassification rate for the given data and how to use it as a measure of impurity to find the best attribute by minimizing it. Another thing that I learned was how to write and debug a program that generates another program. Although writing this program and debugging it was time consuming, it is very useful for future as it can work for multiple datasets.