Matt Burns
DSC 550
11/16/19

For this project I wanted to analyze something less serious than I've analyzed in other courses in this program. I chose 538's candy data that details the ingredients of candy and how often it won in a head to head challenge. It is not a huge dataset as it only analyzes 85 candies and 13 attributes, but I still run a confusion matrix and use the ingredients to forecast weather or not a candy is a winner.

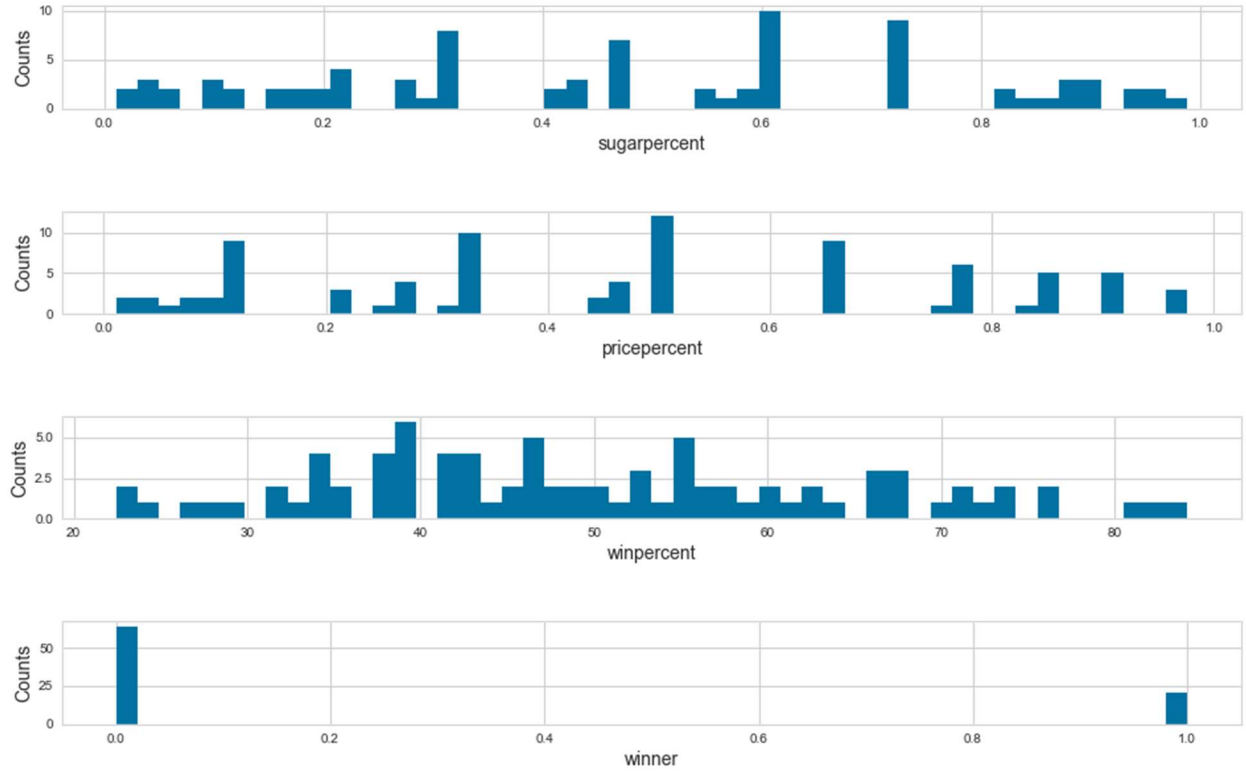
The end goal of this project is to identify which features would make a successful candy. For this analysis I defined a winner as a candy that won at least 60% of the time.

Graphics Analysis

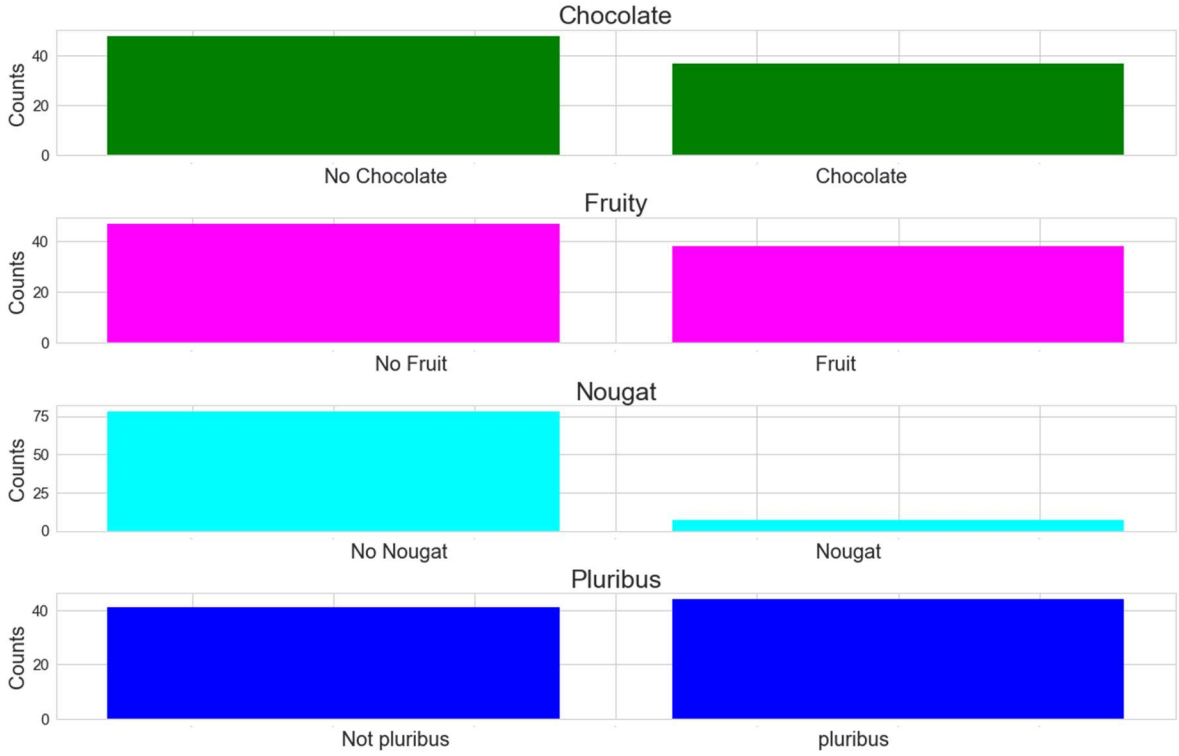
The first step was to get a CSV from 538 and then pull in the file into a pandas dataframe. After checking the quality of the data, I itemized the features of the file. They consisted of a name, 9 dummy variables and 3 ordinals:

- Competitorname
- Chocolate
- Fruity
- Caramel
- Peanutyalmondy
- Nougat
- crispedricewafer
- hard
- bar
- pluribus
- sugarpercent
- pricepercent
- winpercent

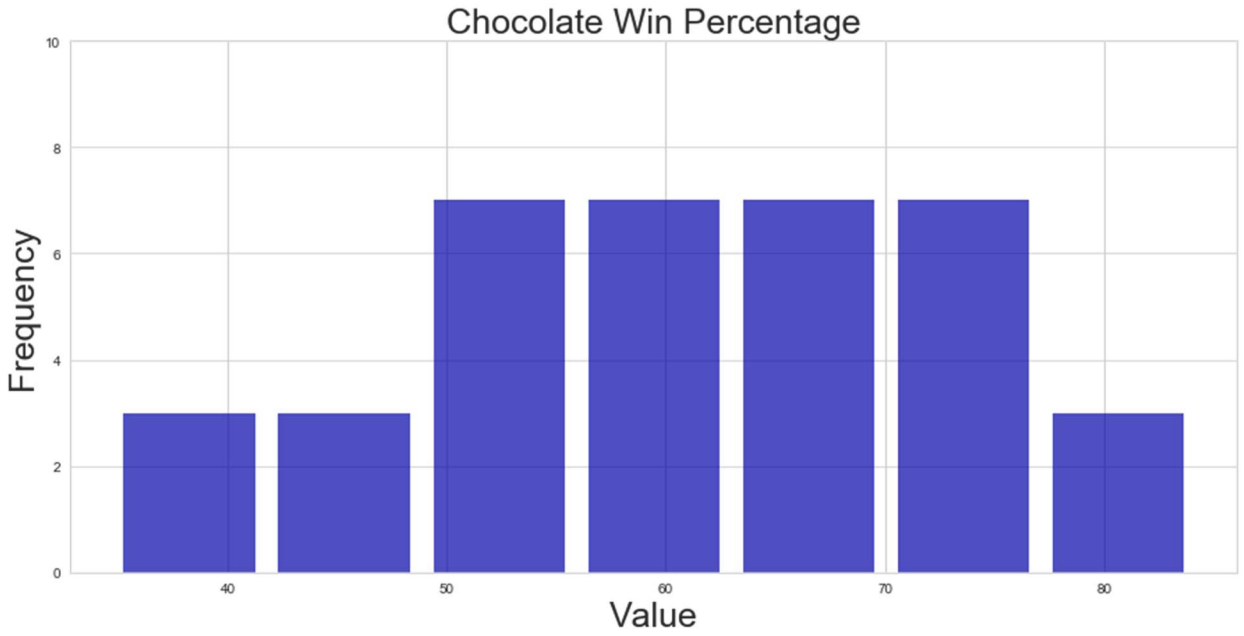
Then I created histograms for the 3 parametric attributes as well as split the winners from the non-winners. You can see that a minority of the candies are winners.

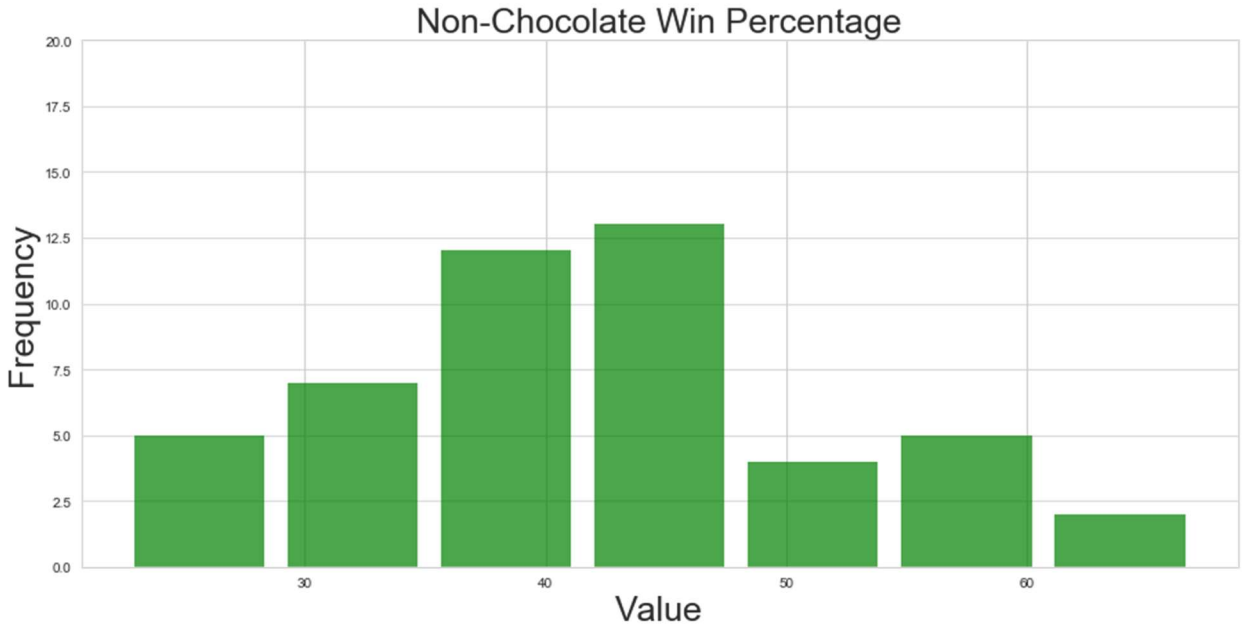


The next step in my exploratory data analysis was to see the win percentage of the candies based one three attributes. Not surprisingly, chocolate candies had more wins than candy with nougat. ☹



This made me curious what it would like if I blew out win percentages of chocolate and non-chocolate candies.





Not surprisingly, chocolate candies had a higher win percentage.

Dimensionality and Feature Reduction

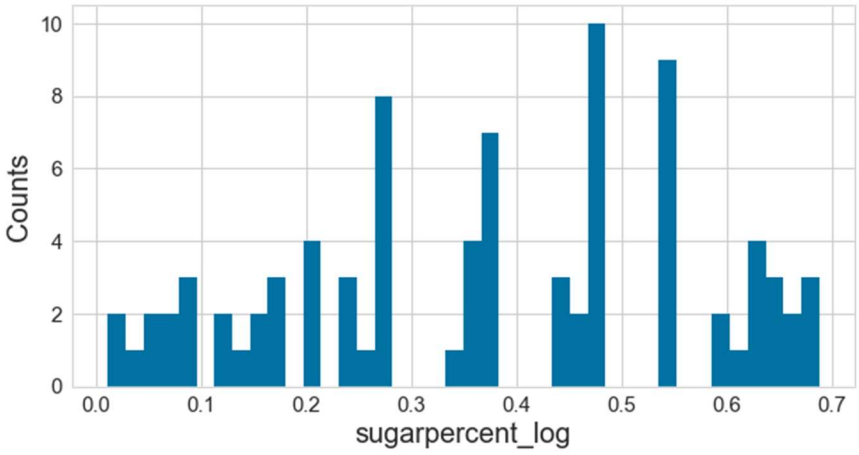
There wasn't much to do with this project regarding dimensionality and feature reduction. The data came in pretty clean from 538. I did have one extra row that needed to be cleaned up.

```
[16]:  1  # Check for NAs
       2  data.isnull().sum()
```

```
t[16]: competitorname      0
       chocolate           1
       fruity              1
       caramel             1
       peanutyalmondy      1
       nougat              1
       crispedricewafer    1
       hard                1
       bar                 1
       pluribus            1
       sugarpercent        1
       pricepercent        1
       winpercent          1
       winner              1
       dtype: int64
```

I ran this line of code (*data.isnull().sum(axis = 1)*) to learn that only row 85 had to be knocked out to get a clean dataset.

Then I plotted the log of the sugar percentage to see how that looked in a histogram.



Finally, I updated my dummy variables with one hot encoding. These are my results.

```
5
6  # One Hot Encoding
7  data_cat_dummies = pd.get_dummies(data_cat)
8
9  # check the data
10 print(data_cat_dummies.head(8))
11
```

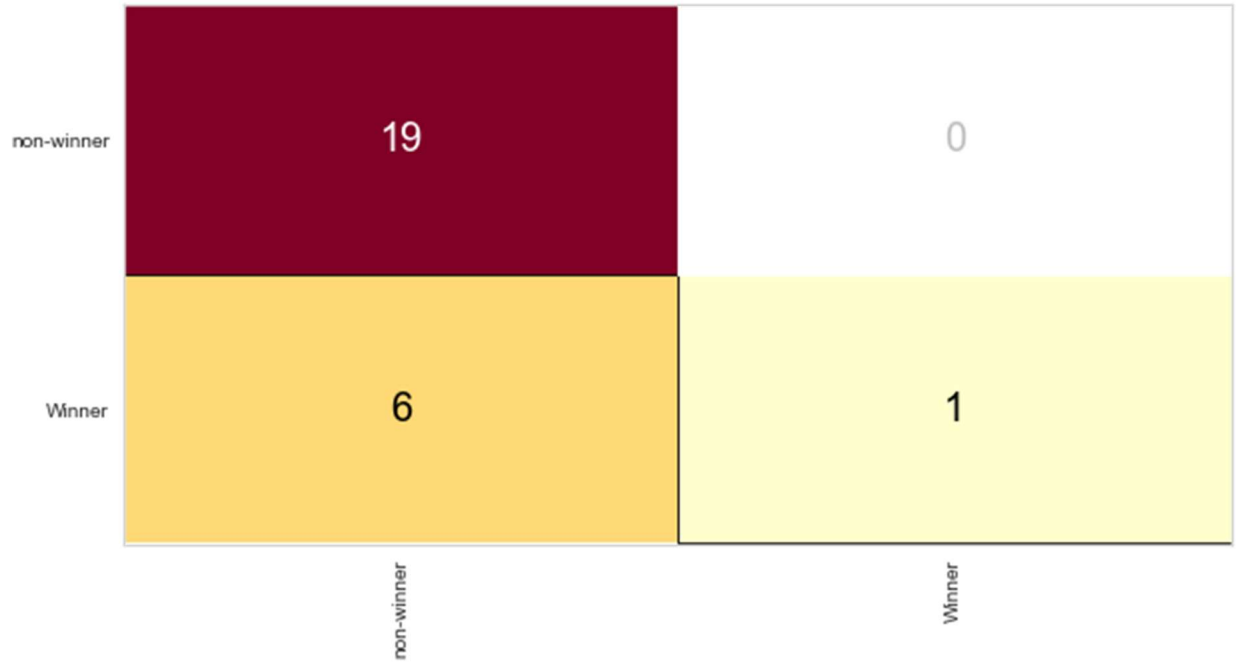
```
   chocolate  fruity  caramel  peanutyalmondy  pluribus
0        1.0     0.0      1.0             0.0       0.0
1        1.0     0.0      0.0             0.0       0.0
2        0.0     0.0      0.0             0.0       0.0
3        0.0     0.0      0.0             0.0       0.0
4        0.0     1.0      0.0             0.0       0.0
5        1.0     0.0      0.0             1.0       0.0
6        1.0     0.0      1.0             1.0       0.0
7        0.0     0.0      0.0             1.0       1.0
```

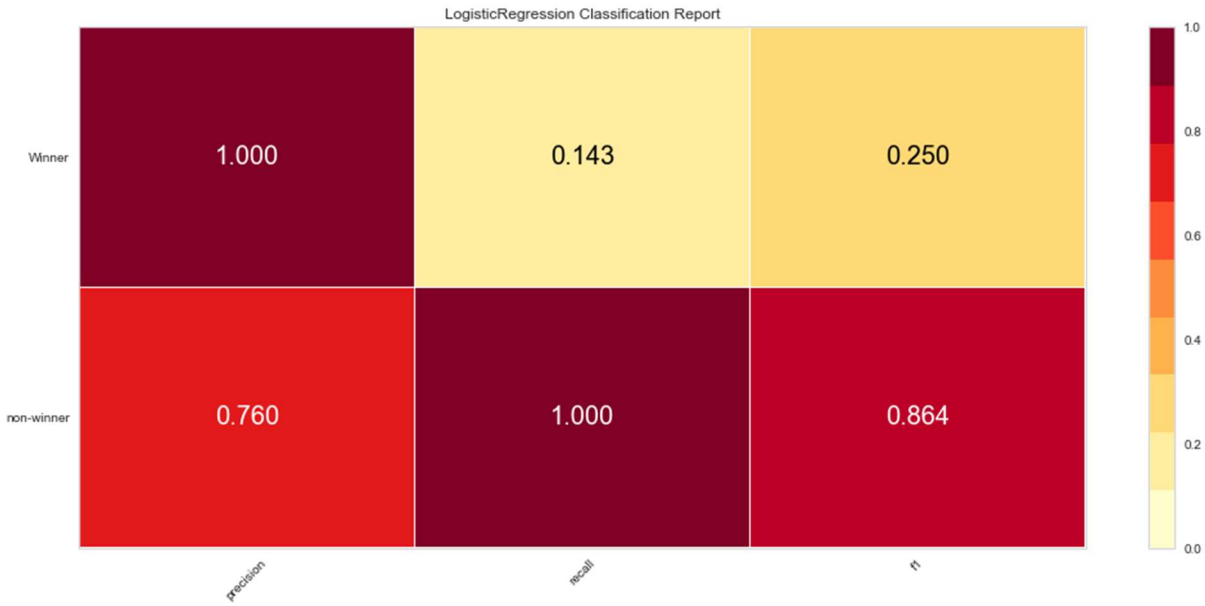
Model Evaluation and Selection

I wanted to see if I could build a model that could I identify winning candies. For that I created a new feature called winner. It is a dummy variable that identifies a candy that won at least 60% of the time.

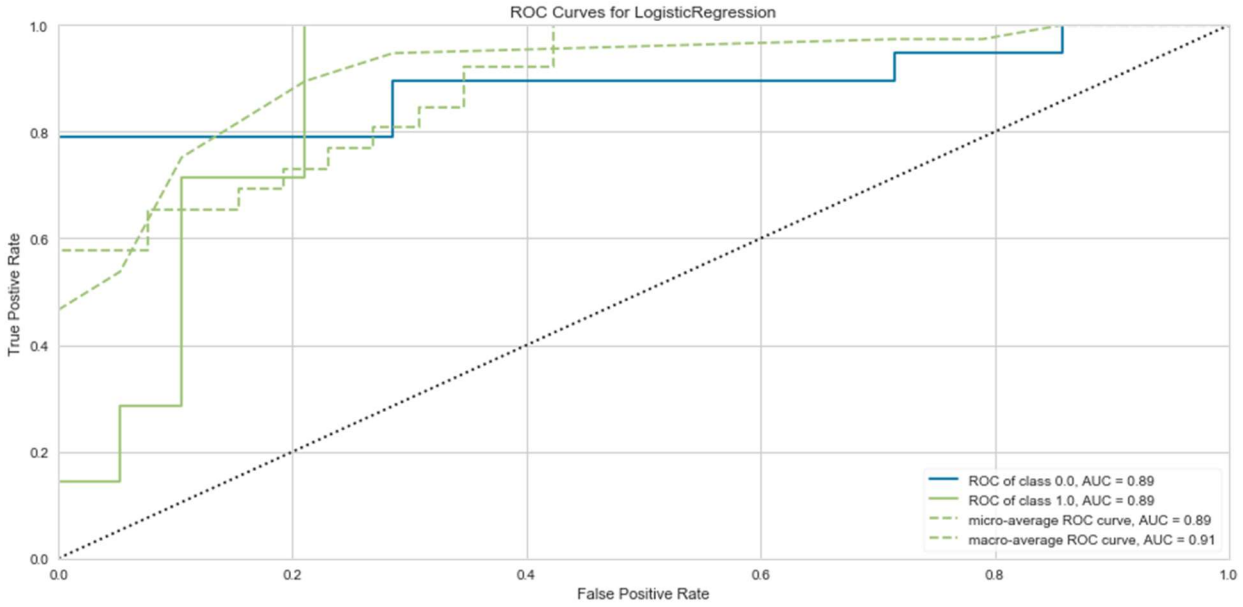
I built a model with pricepercent, sugarpercent_log , chocolate, fruity, caramel, peanutyalmondy and pluribus. 59 candies in the training set and 26 in the validation.

When I put it in the confusion matrix, I got 20 out of the 26 correct.





My ROC curves didn't look bad, but the model didn't do that much better as I added the one hot variables to supplement the prices and log sugar.



The area under the curve for both winners and non-winners is 0.89 which is closer to 1 than 0.5. Therefore, the classification system is decent, but can be improved. What I am uncertain of is if it should be improved with a different model or different usage / manipulation of the parameters.