Taylor Imhof  
Bellevue University | DSC 550

Final Project Submission

3/3/2022

**INTRODUCTION**

For my final DSC 550 project, I decided to work with a marketing dataset that I found on the website Kaggle.com. After pouring over quite a few different datasets, I opted for the marketing one as I felt it had a good mix of different features that I could put my newfound data mining skills to the test! While the required data cleaning efforts were not too extensive, I was still able to implement practical data munging techniques. For visualizations, I relied almost solely on the Seaborn library. The syntax for creating the different charts with Seaborn is much more straightforward than other techniques that I have been exposed to during this program. I wound up creating two models: a simple linear regression model as well as a logistic regression model.

**SUMMARY OF MILESTONE EFFORTS**

**Milestone One: Data Selection and Exploratory Data Analysis (EDA)**

As stated in the introduction, I worked with a marketing dataset that I downloaded from Kaggle. A link to the data set can be found [here](https://www.kaggle.com/jackdaoud/marketing-data). There was not too much information surrounding the specifics about the data itself. I got the impression that a Spanish professor created this dataset for one of their data science courses. Reading in the data was straightforward. Since the data was provided in Excel format, I was able to call on the Pandas method <read\_csv> to get the data loaded into a Pandas dataframe. After checking some basic information about the dataframe itself (e.g., dimensions and data types), I found that the dataset contained 2,240 observations, each with 28 columns. One of the main columns that I was interested in exploring was the column that measured each customer’s yearly income. However, I noticed that the column had been encoded as a string. So, although the data preparation was a part of the work needed for milestone two, I had to do some minor data cleaning. All I had to do was extract the dollar signs and comms from each income row’s values. I wound up creating a utility function that I could apply to each row using the <apply> function built into the Pandas dataframe. After cleaning the values, I cast the “cleaned” strings to float data types using the <astype> function. Also, for whatever reason, the income column header contained unnecessary whitespace, so I also went ahead and renamed the column.

Chart, box and whisker chart

Description automatically generatedChart, histogram

Description automatically generatedAfter plotting a histogram of this newly cleaned income column, it was obvious that there were some outliers in the dataset. So, I created a boxplot to better picture where the outliers were located.

Chart, histogram

Description automatically generated

After identifying the outliers (really, there was just one record with an income value of >600000), I filtered the income column for customers with yearly incomes below $200,000. The last of the three graphs above is a pretty good visualization of how the income values were distributed in the dataset. Even though there are still a few records on the higher end (160000), I felt that they were close enough to the “normal” values to keep in the final dataset without having too much impact.

Chart, histogram

Description automatically generatedChart, box and whisker chart

Description automatically generatedThe next column that I was interested in exploring was the column that contained each customer’s birth year. Again, I felt that I could better understand how old each customer was by using Seaborn visualizations. First, I decided to create a boxplot to identify any potential outliers. There were a few instances of customers whose birth year was well below the 25th quartile whisker of the boxplot. Therefore, I decided to filter the dataframe by customers born after 1940. Then, after reviewing the histogram of the birth year column, it did seem like the distribution was normal, with the average birth year of the customer being around the mid-1970s.

The third column that I explored visually was the education column, measuring the highest education of each customer. I felt that a simple bar chart that tracked the membership of each education label would provide enough insight. A great majority of the customers in this dataset had graduated from college. Interestingly, the number of Ph.D. holders was greater than the number of Master’s holders. The last column that I explored was the country column that measured which country each customer was from. I decided to view this information using a pie-chart. Nearly fifty percent of the customers are from Spain, followed by Saudi Arabia (15%), Chart, bar chart

Description automatically generatedCanada (12%), Australia (7%), and the US (5%).

**Chart, pie chart

Description automatically generated**

This concluded my work for milestone one. It was great working with Seaborn again to create my visualizations. I felt that my work in creating them helped solidify the visualization concepts we have been learning during this program.

**Milestone Two: Data Preparation**

Whenever I have to perform some data cleaning, the first step that I have been doing is to check for missing or duplicate values. The Pandas package comes packed with useful methods for this goal, included the <isnull()> and <duplicated()> functions. Fortunately (or unfortunately, as I did not get to practice handling missing data), my dataset was clean in that it did not have any missing or duplicated values.

There were a few columns that I felt could benefit from transformation. As I stated in the previous section, I did some minor data cleaning to the income column (remove whitespace, cast to float datatype). There was also a column denoting when each member first became a customer. This column’s values were encoded as strings, so I converted them to DateTime objects by using the <to\_datetime()> function. I did not ultimately wind up using this column in my models. Still, it was beneficial working through the steps to create DateTime objects with Python.

There were several columns that I felt could be consolidated into single columns. These included the two columns tracking the number of kids and the number of teens in each customer’s household. I opted for creating a new <children\_at\_home> column that added the values of the two columns together. Also, several columns measured the quantity of the different types of products (gold, fish, sweets, etc.) and the different purchasing methods (catalog, store, web, etc.). I created two columns that consolidated these related values together for simplicity.

While the dataset came with a column for each customer’s year of birth, I felt it would be more intuitive to have a column that held each customer’s age. For this task, I had to subtract the year of the customer’s birth from the current year using the <datetime> module. It should be noted that there might be some error as 2022 is still in its early months. However, I did not feel that this would be significant enough to impact my results.

After reviewing the unique labels for the education column, I felt that I could rename them so North American readers would better understand them. For instance, there was an education label of “2n cycle,” which seems to be mostly used in Europe for denoting post-graduate education. So I wound up creating a dictionary of the current labels as keys and the desired label changes as the values. Then, by using the <map()> function, I was able to apply the label changes across all of the dataframe’s education values.

A few records’ labels did not make much sense within the marital status column. I believe these were inserted into the dataset as a joke and were meant to be discovered. Therefore, I used basic conditional logic to filter the marital status column to include only relevant values such as “married,” “single,” “divorced,” and “widow.”

After creating and modifying existing columns, I decided to drop the ones that were now redundant. Since I combined the contents of the kids and teens at home columns, they could be safely dropped. Also, I had created a new education column and age column, so I dropped the original two columns. I went ahead a dropped the Dt\_Customer column as I was unable to think of a use-case where I could need such data. There was also an ID column to identify each customer uniquely. Since I did not need such values, I dropped this column from the final dataframe.

The last bit of data cleaning that I performed was simply renaming all of the column headers to align with the convention that I am used to. Since the beginning of this program, I have also liked to name my variables using lowercase letters. Also, if there are multiple words in a variable or identifier, I have separated them using underscores. To accomplish this task, I simply passed a dictionary of the original column headers and my desired column headers to the <rename()> function.

**Milestone Three: Model Building and Evaluation**

Now that I had an explored and clean dataset, I started building my models. Prior to buildling models in the past, I have found it beneficial to create a correlation matrix. I feel that this provides a good look into the future of what the model will prove statistically. I found a slick way of displaying the correlation matrix so that only certain strong relationship indicators are displayed at each variable intersection. Most of the stronger relationships (both positive and negative) were intuitive. For instance, the income column was strongly related to the total expenditures column. One coefficient that I found interesting was that the number of wine purchases was very stronger (0.9) correlated with total expenditures. I felt that this might be due to the larger portion of the customers being from Spain and wine being a large part of Spanish culture. Also, perhaps the customers included in this dataset enjoyed higher quality (and thus, more expensive) wines.

For one of my models, I decided it would be interesting to predict new customers’ income based on the values measured in this dataset. I opted for a simple linear regression and used both the <LinearRegression> object from the Sklearn package and the <OLS> object from the Statsmodels package. I wanted to get practice using both implementations, and I found that their approaches are nearly identical. However, I did appreciate the ability to create a robust summary of metics from the <summary()> function provided by the <OLS> implementation.

The R2 value of the linear model was significant (0.816), meaning the model fits the observed data quite well. The linear model also scored approximately 0.8. After sorting the p-values produced from the OLS model, I was able to identify the features that had the most predictive power of the customer’s annual income. These include the level of education (Master’s and Ph.D.), the number of web visits, age, and the number of children at home.

I also wanted to see if I could create a model that predicted whether a customer would accept an offer during a marketing campaign. One of the columns (response) measured whether the customer had accepted the most recent campaign offer. Since this column had binary values, I trained a logistic model with the response column as the target. The steps were pretty similar to how I created and trained my linear model. I first had to create a feature and target subsets of the original dataframe. I then used the <train\_test\_split()> method from the Sklearn package. Then, I trained the model by passing the training values to the <fit()> method. Then, I created a set of predictions using the test data and the <predict()> function. To understand how the model performed, I created a confusion matrix. My trained model was quite good at accurately predicting whether the customer would not accept the offer (0). However, only a total of 18 customers had accepted the most recent campaign offer. Thus, I did not feel that there were sufficient data points to assess the model’s ability to predict whether the customer would accept the offer (1).

**CONCLUSION**

By working through each milestone of this final project, I feel that I have been able to use all of the skills and knowledge that I have gained up to this point in the program. From initially identifying a data problem to creating a fully trained and predictive model, I feel that I have a much better grasp of the skills and processes involved with real-world data science applications. I am so thankful for all of the extremely useful Python data science packages that abstract the complicated statistical computations and allow me to focus on generating useful insights. While I do not believe this will be the last project I work on during this program, working through it has provided the most learning opportunities that I have encountered thus far. I am now confident that given a certain data science or business problem, I can gather the data and create models to gain useful insights to solve it.