Capstone2 Final Report:

The Introduction

The Problem:

Can I use a data set from the fictional telecommunications company Telco to predict customer churn? If so, can I identify which features have the greatest predictive power in identifying those customers who will leave?

The Why:

As the author and speaker, Jeffery Gitomer, states in the title of his book, “Customer Satisfaction Is Worthless, Customer Loyalty Is Priceless”, having customers that stick with your company is extremely important. That is why it is of great value to a business to be able to identify those customers who will potentially leave the business and why.

For Telco specifically, here is why it matters. The customers who left spent a total of $139,130.85/month with an average of $74.44/month per customer. The customers who are still with Telco spend $316,985.75/month with an average of $61.27/month per customer. This is a 30.5% per month loss due to customer churn! Our best model was able to predict 79% of the customers who would leave. If just half of the 79% the model predicted to leave Telco were convinced to stay, then that would be a savings of $54,956.68/month. This would reduce the per month loss to 18.5%!

The Solution:

After cleaning the data set and conducting extensive exploratory data analysis (EDA) I was able to train a logistic regression model in conjunction with the SMOTE oversampling technique that had an overall accuracy of 0.75 and a recall score on the target feature of 0.79.

If you are interested understand more the discovery process that led to this model, and the rejection of many others, please peruse these notebooks. I suggest the order given, but each notebook should be able to stand on its own as well.

1.0_Capstone2_cleaningdata
2.0_Capstone2_EDA
2.1_Capstone2_featureselection
3.0_Capstone2_LogisticRegression_initialassessment
3.1_Capstone2_LogisticRegression_imbalanceddata
3.2_Capstone2_randomforest
3.3_Capstone2_LogReg_and_RFC_usingselectedfeatures
4.0_Capstone2_furtherexploration_year1model
5.0_Capstone2_BusinessAnalysis_and_modelcomparison

This introduction is part of a larger report telling the story of my search to identify the customers who left Telco. This larger report is in the file “Capstone2_Final_Report”.

The Process

The Data:

The data set that I will be using was obtained from Kaggle:

https://www.kaggle.com/blastchar/telco-customer-churn

It was originally obtained from an IBM Congos Analytics sample data set. The IBM data set has numerous other features that were not included in the data set used in this project obtained from the Kaggle source.

https://www.ibm.com/support/knowledgecenter/SSEP7J_11.1.0/com.ibm.swg.ba.cognos.ig_smples.doc/c_telco_dm_sam.html

I downloaded the data set as a csv file from Kaggle. I then loaded the raw csv data file to my GitHub. I then loaded the data as a pandas DataFrame into the Capstone2_cleaningdata notebook. With the data as pandas DataFrame, I was able to scour it for missing values, anomalous entries, and to make sure the data types of the features were correct.

I first noticed an issue when I observed that the ‘TotalCharges’ feature was listed as an object dtype instead of a numeric dtype (I was expecting a float64 dtype as it’s sister column, ‘MonthlyCharges’ was, or even an int64 dtype). This indicated to me that there were some non-number entries in this feature. The search for missing or NaN entries returned fruitless, so I knew they were not the culprit. I decided to force the situation by using .to_numeric on the troubling column with errors=’coerce’. This would cause any non-numeric entry to become a NaN. This worked! By using .isna().any(axis=1) on the DataFrame, I was able to isolate the troublesome rows. There were only 11, so this allowed me to inspect them all visually. Upon close inspection, I noticed that they all had a ‘tenure’ value of 0. It was at that moment that the story of the incorrect column dtype came into focus.

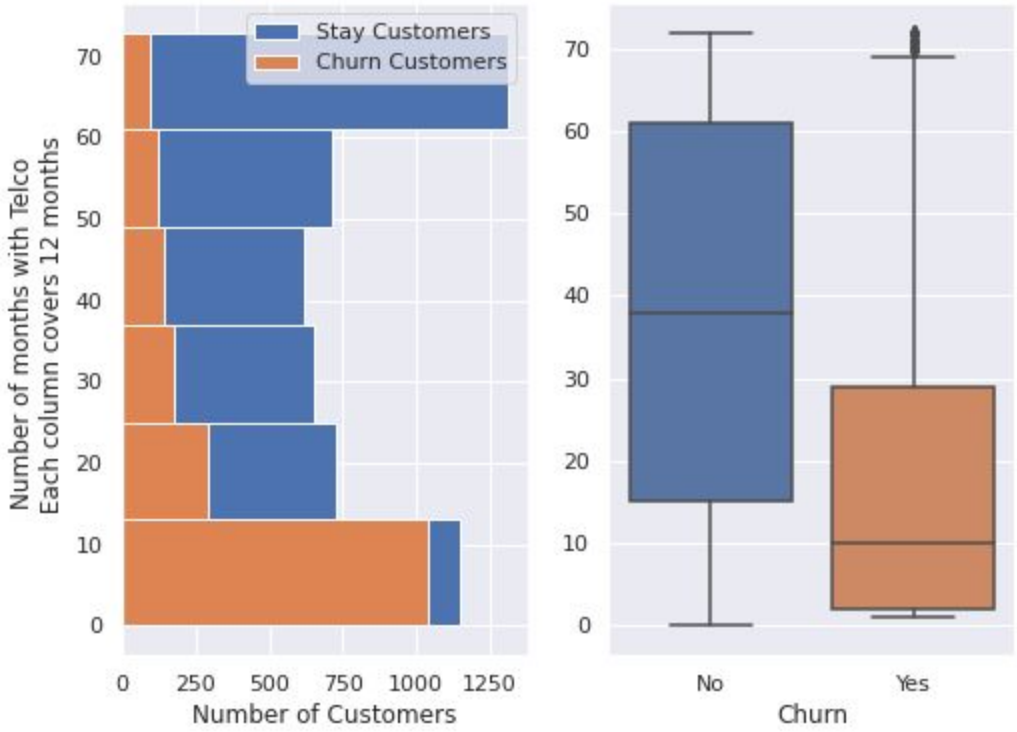
These rows did not have any ‘TotalCharges’ because they were all brand new customers who had not been charged anything yet. With this new knowledge in hand, I quickly changed their NaN’s in the ‘TotalCharges’ feature to 0, as that is exactly what they had been charged so far.

The rest of the data looked clean and ready for some EDA!

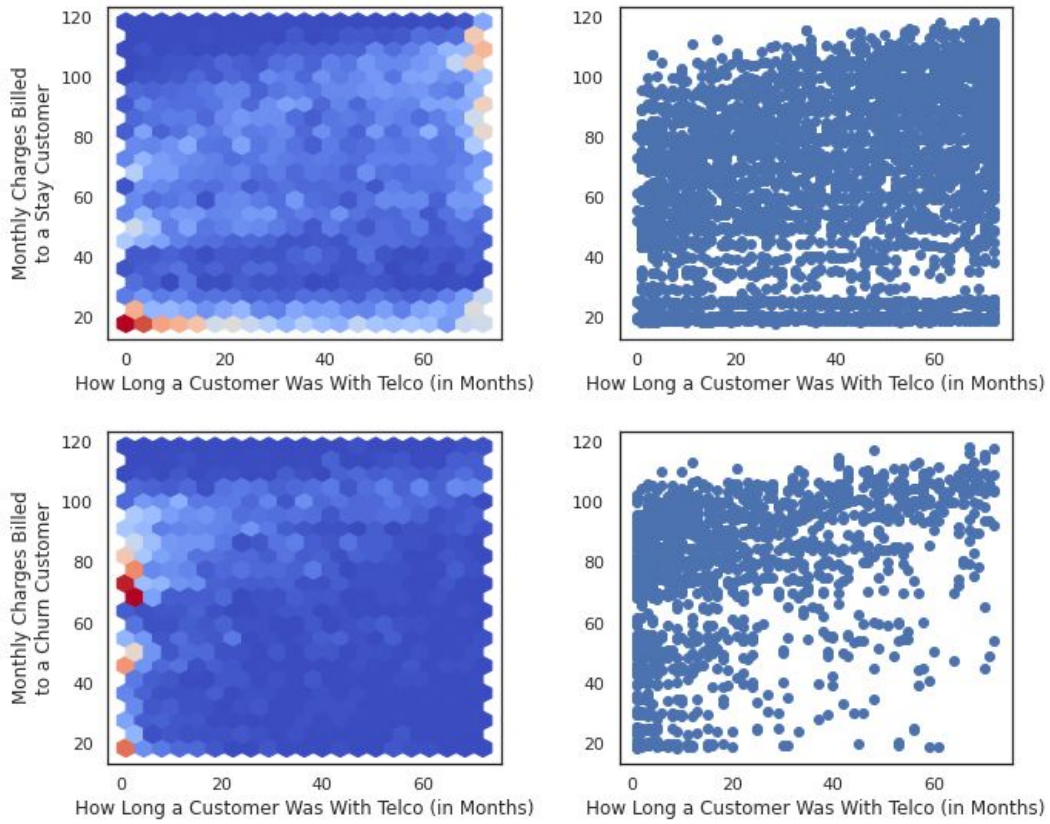
EDA:

This data set, although not as large and as intricate as others, still held delightful surprises to unearth upon exploration.

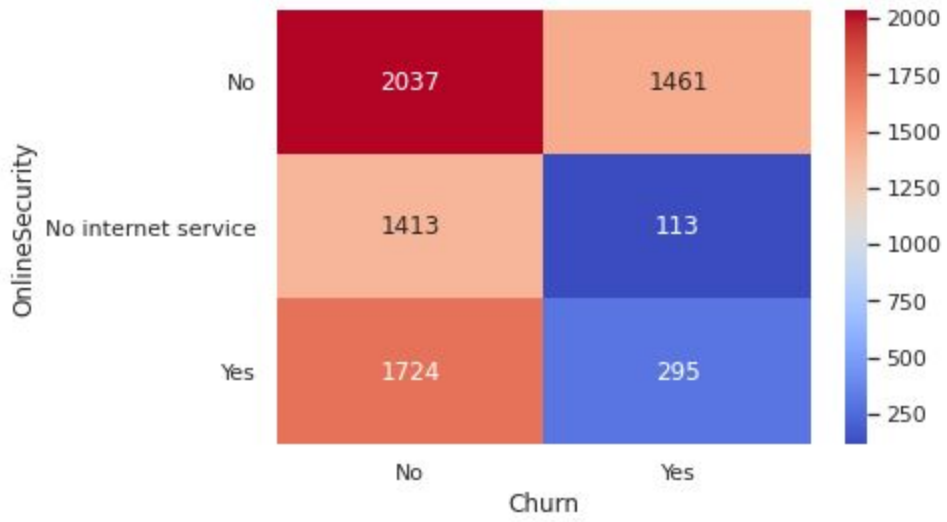
One of the first themes to be encountered when exploring the Telco churn data is how many customers left in less than one year of being with Telco. This is easily seen in the chart below. I go on to study this issue more in Capstone2_furtherexploration_year1model notebook.



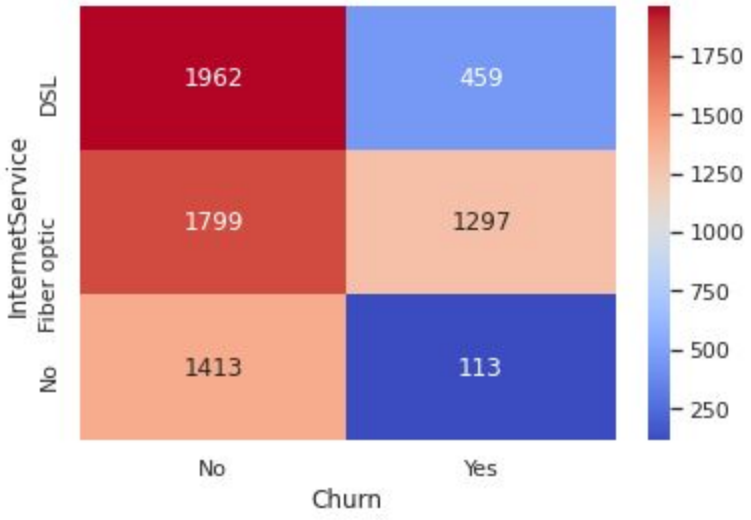
Another theme that stood out was the concentration disparity related to the monthly charges of a customer and the tenure of that customer. This can be seen in this figure.

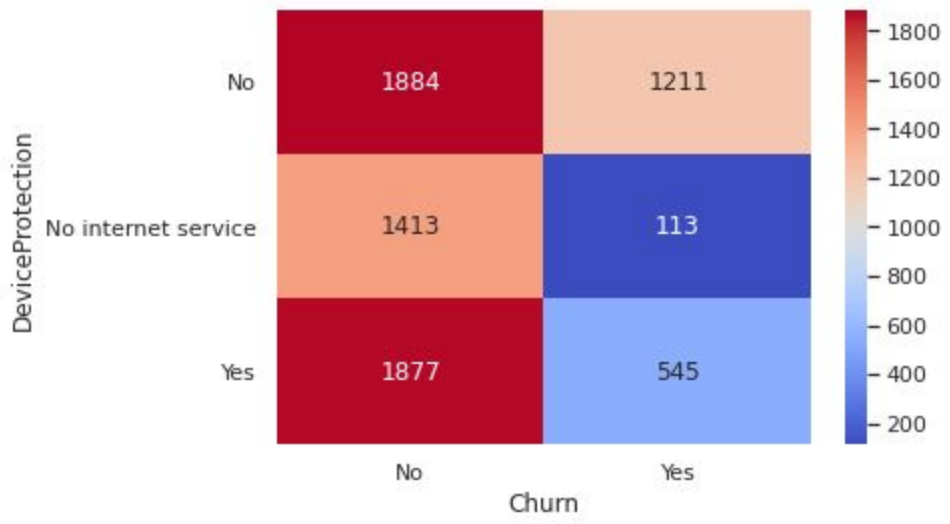


When I looked at the heatmap distributions of the Churn customers and the stay customers across the different categorical variables there were 2 major issues that stood out. The first, as demonstrated in the heat map image below, shows that there was a large proportion of Yes churn customers who were ‘No’ in these categories. This indicates that they were paying for internet, but they were not utilizing all of the services that Telco offered. This furthers explain the concentration disparity discussed earlier.



The other issue that stood out from studying the heatmaps was the specter of collinearity between certain categories. This is demonstrated when looking at these two heatmaps. The ‘No internet service’ is the exact same in both. This held true for the following features: 'InternetService', 'OnlineSecurity', 'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingTV', and 'StreamingMovies'. This was something I ended up exploring more in both the Capstone2_featureselection & Capstone2_LogReg_and_RFC_usingselectedfeatures notebooks.





Models:

Baseline Model - Because this is a classification problem I chose to use scikit-learn's LogisticRegression model as my baseline. I chose to initially use OneHotEncoder to transform the categorical data, because it fits well in a pipeline and, if the model is put into use, can handle new data with ease. I also chose to use StandardizedScaler to scale the numeric values as well. This initial model obtained an Yes recall score of .55 and an overall accuracy of .82. This model can be found in the 3.0_Capstone2_LogisticRegression_initialassessment notebook.

Correcting for Imbalanced Data Models - The information that I obtained from my baseline model indicated that it had not overfit. But I realized that I was dealing with an imbalance data issue. So to obtain a higher Yes recall score, I would need to address this issue. The next 3 models can be found in the 3.1_Capstone2_LogisticRegression_imbalanceddata notebook.

Still using a logistic regression model, I used the stratification process on the train test split to see if it produced a noticeable difference in the results of the model. Then, I implemented an oversampling technique, SMOTE. Finally, I use an undersampling technique, Nearmiss. The SMOTE and Nearmiss techniques both produced increased Yes recall score (both were .79), but the SMOTE showed a slight drop in overall accuracy (to .75) while the Nearmiss showed a large reduction in overall accuracy (.66).

RandomForest Models- I wanted to use an ensemble approach to this classification problem, so I decided to use scikit-learn's RandomForestClassifier. I first try using the model with no changes to it. I continue to stratify my data on the train test split because of the imbalance issue. This model has a Yes recall of .51 and an overall accuracy of .79.

To be able to use the model's feature_importance, I was forced to stop using the OneHotEncoder. Instead I switch to using get_dummies. This allowed me to see which features the random forest model was finding important.

Again, as with the logistic regression models, I trained random forest models using the SMOTE oversampling technique and the Nearmiss undersampling technique to see if they can help improve both accuracy of the model and the Yes recall score. The RandomForest with SMOTE technique produced a Yes recall score of .52 and an accuracy score of .79 while the RandomForest with Nearmiss technique produced a Yes recall score of .69 and an accuracy score of .58.

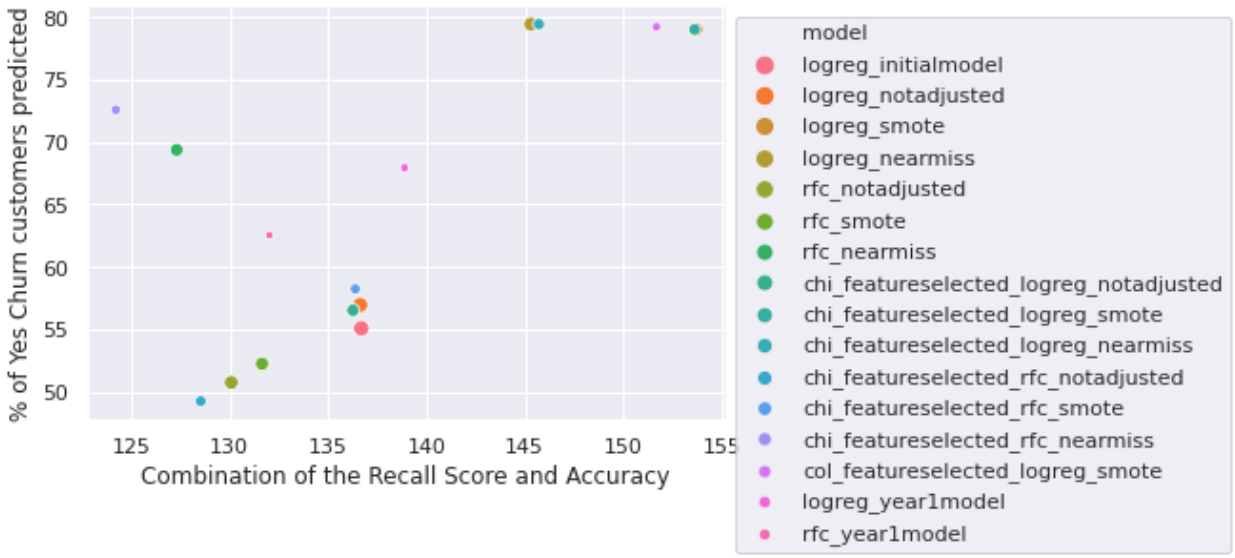
Feature Selection Models- In the EDA notebook and the feature selection notebook, I identify certain features that might benefit a model by having them dropped or changed. In the Capstone2_LogReg_and_RFC_usingselectedfeatures notebook, I do just that. In this notebook, I use the baseline scikit-learn LogisticRegression model and the RandomForestClassifier. I used the same two models and the same the attempts to address the imbalance data (standard, with SMOTE, and with Nearmiss) so I would be able to accurately compare the feature selection model with the models that I had trained and tested earlier with the complete Telco data set. The results were mixed, with Yes recall scores ranging from .49 to .79 and accuracy scores ranging from .52 to .80.

In this notebook, I also trained a 7th model using a LogisticRegression model with the SMOTE oversampling technique to see how the model would perform if I dropped the features that seemed to have collinearity. The model scored a Yes recall of .79 and an accuracy of .72.

Year 1 models- In my final model notebook I trained two models a scikit-learn LogisticRegression and a scikit-learn RandomForestClassifier. As data, I used just those customers who had been with Telco for 12 months or less. Using this data set solved the imbalanced data issue because the churn Yes to No ratio was 47% to 53%. It also allowed me to explore more the reasons as to why such a large number of customers left Telco in the first year. The models scored .68 and .63 on the Yes recall and .71 and .69 on the overall accuracy.

Findings:

The Results- Below is a scatterplot with all of the points plotted out for a visual representation of the different models and how wel they scored on Yes recall and on accuracy. There is also a table that contains the results of all of the models I trained and tested for this project.



Model	Recall Score	Accuracy Score	Combination Score
LogisticRegression w/SMOTE	79.01	74.67	153.69
LogisticRegression w/SMOTE using features selected based on Chi2 tests	79.01	74.62	153.63
LogisticRegression w/SMOTE using columns modified for collinearity	79.23	72.46	151.69
LogisticRegression w/Nearmiss using features selected based on Chi2 tests	79.44	66.27	145.71
LogisticRegression w/Nearmiss	79.44	65.87	145.31
LogisticRegression using first year customer data only	67.95	70.93	138.89
LogisticRegression baseline model	55.08	81.60	136.69
LogisticRegression using just the stratification process on train test split	56.96	79.67	136.63
RandomForestClassifier w/SMOTE using features selected based on Chi2 tests	58.24	78.13	136.38
LogisticRegression using just the stratification process on train test split while using the features selected based on Chi2 tests	56.53	79.73	136.26
RandomForestClassifier using first year customer data only	62.55	69.47	132.02

RandomForestClassifier w/SMOTE	52.25	79.39	131.64
RandomForestClassifier using just the stratification process on train test split	50.75	79.33	130.08
RandomForestClassifier using just the stratification process on the train test split while using the features selected based on Chi2 tests	49.25	79.27	128.52
RandomForestClassifier w/Nearmiss	69.38	57.92	127.30
RandomForestClassifier w/Nearmiss while using the features selected based on Chi2 tests	72.59	51.62	124.21

The 'Best'- Since there were five models that each scored .79 on the Yes recall, I formed a combination measure that added a model's accuracy score to the Yes recall score. This new value was named the 'combination score'. Using this metric, the top 3 were all models that utilized the SMOTE technique to address the imbalanced data issue. It is also worth noting that the top 50% of models were all LogisticRegression models.

Any of the top 3 models would serve the Telco well if it went into production. In terms of simplicity of implementation, the LogisticRegression model with the SMOTE technique tested and trained on the original data would be easiest (it also, just barely, had the highest combination score). Thus, I would consider it the best model, but any of the top three had a high enough Yes recall score with a corresponding high accuracy score that it would benefit Telco in identifying customers who will leave.

Conclusion:

Future Work- I would have liked to have tried other models. I would have especially liked to have tried an unsupervised approach. Along the same line of thought, I would have liked to have tried to ensemble the LogisticRegression and RandomForest models with at least one more model.

Something that was outside the scope of this project, but that I would have liked to have explored more were the details of certain groups of customers, such as the high-end paying group of customers that have a long tenure with Telco. Along with this, I would have liked to have explored more the characteristics of the customers who left Telco within the first year.

It would have been interesting to explore which Yes churn customers the models were having a tough time with and explore this data set. It would be interesting to compare the False negatives between the two models to see if they were having a difficult time with the same customers or if they were each being able to identify different customers correctly and incorrectly.

Business Takeaways- As stated in the intro, if just 50% of customers that the best models identified as customers who will leave Telco ended up staying with Telco, then that would be a retention of $54,956.68/month, or close to $660,000 a year.

Per the information gleaned from the EDA, there needs to be further research done by Telco as to why 42% of their fiber optic customers are leaving. Along the same lines, why is there a significant portion of the customers who purchase internet through Telco, but do not sign up for Telco's higher-end services end up leaving?

Telco should also look into eliminating the month-to-month payment plan. Again, according to the EDA, 43% of the customers who had a month-to-month payment plan ended up leaving Telco.

Resources Used or Helpful:
This project was done using Google Colabs.

Packages- pandas, numpy, matplotlib, seaborn, scipy, sklearn, imblearn

Idea and base code for type and shape checking of train test split; the recommendation to look at the feature coef when using the LogisticRegression model; suggestion to use SMOTE and Nearmiss to address imbalanced data - A J Sanchez - Thankful for the private correspondence and the copious help!

Idea and base code for using the Chi-squared tests - Cornellius Yudha Wijaya
https://towardsdatascience.com/categorical-feature-selection-via-chi-square-fc558b09de43

Idea and base code for sklearn pipeline with OneHotEncoder and StandardScaler - Kevin Markham
https://youtu.be/irHhDMbw3xo

Base code for ROC curve and AUC score - Ajay Ramesh - found in answer to this stack exchange question: https://stackoverflow.com/questions/25009284/how-to-plot-roc-curve-in-python

Base code for the SMOTE and Nearmiss was from the imbalanced-learn documentation - https://imbalanced-learn.readthedocs.io/en/stable/auto_examples/index.html#model-selection

Idea and base code for RandomForest feature importances - Coder Unknown - found on Springboard's Data Science Career Track Jupyter Notebook: RandomForest_casestudy_covid19

Help with my Seaborn plots and the side by side plots came from - Jovian Lin - https://jovianlin.io/data-visualization-seaborn-part-2/