**Rafae Abdullah**

**A20137768**

**Badhan Deb**

**A20331147**

**Jarrett Hollie**

**A20357851**

**Weston Salmon**

**A20072077**

**BAN 5753-65925**

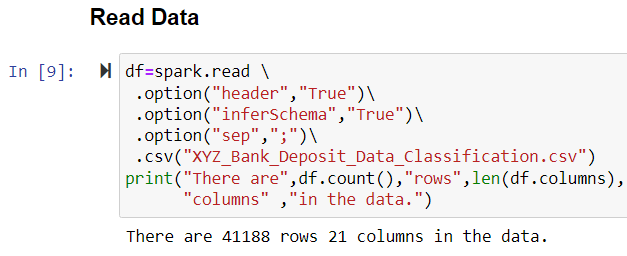
**November 16, 2022**

**Mini Project 2: Spark**

**Introduction**

The dataset provided for this analysis reviews XYZ bank’s direct marketing campaign. The bank utilized phone calls to collect consumer data. The attributes in the dataset include customer demographics, such as age, marital status, job, and education. Additionally, there are attributes regarding the bank's latest contact with a customer during the ongoing campaign, such as the communication method, the last day of the week and month in which the bank contacted the customer, and the duration of the call. Furthermore, there is a group of additional attributes that pertain to the number of contacts made during the campaign, the number of days that has passed since a client was contacted from the previous campaign, the outcome of the previous marketing campaign, etc. Lastly, the final group of attributes relates to social and economic context. These variables include employment variation rate, consumer price index, Consumer confidence index, Euribor 3-month rate, and the number of employees. The dataset contains information lasting between May 2008 and November 2010. Overall, these attributes are being utilized to develop predictive models in identifying clients who will subscribe for a term deposit.

**Exploratory Analysis**



**Figure 1: Code for Reading in Data File**

The first step of the exploratory data process was to read in the data file to access the attributes. As seen in Figure 1 above, the CSV file is read into the system, and we are provided with a total of 41,188 rows and 21 columns of data. To receive a glimpse of the data file and ensure it was read in properly, we wanted to look at a sample of the dataset.

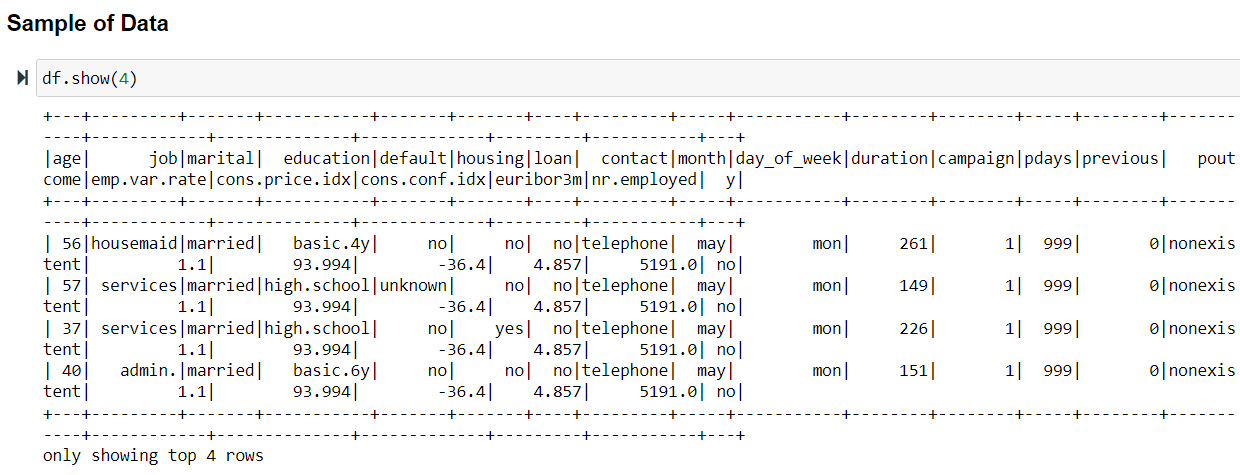
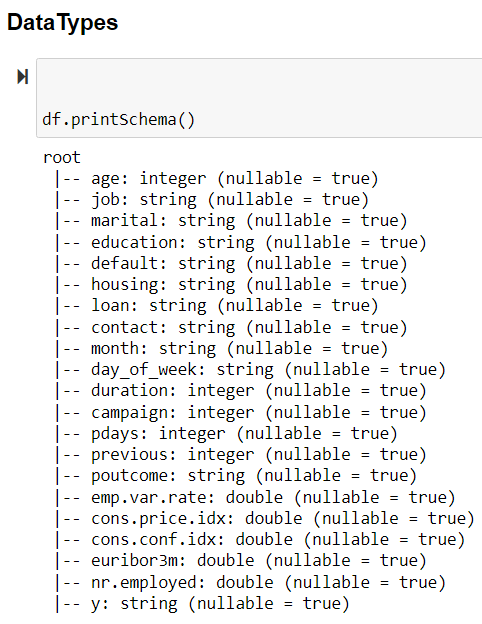
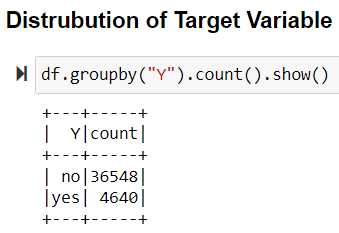
**Figure 2: Sample of the Dataset**

Figure 2 illustrates the top four rows from the dataset, along with all the variable names. For example, the client in the first row is 56 years old, works as a housemaid, is married, has a basic four-year education, etc. There is a plethora of other variables providing information about this specific client that helps determine whether they will subscribe for a term deposit. Although this sample is only looking at the first four rows, this information can be obtained for every client in the dataset. Next, we wanted to observe the data types of each variable for further use in our analysis.



**Figure 3: Data Types**

Figure 3 presents the data type for each of the 21 variables within the dataset. As it can be seen, there is a mixture of integer, string, and double variable types. Depending on the variable’s data type, we will explore the variables further by observing their cardinality if they are string (categorical) variables or distributions with they are numeric or double. Additionally, Figure 3 allows us to see that each variable has the ability to hold null values, which we will check later in our explanatory analysis.



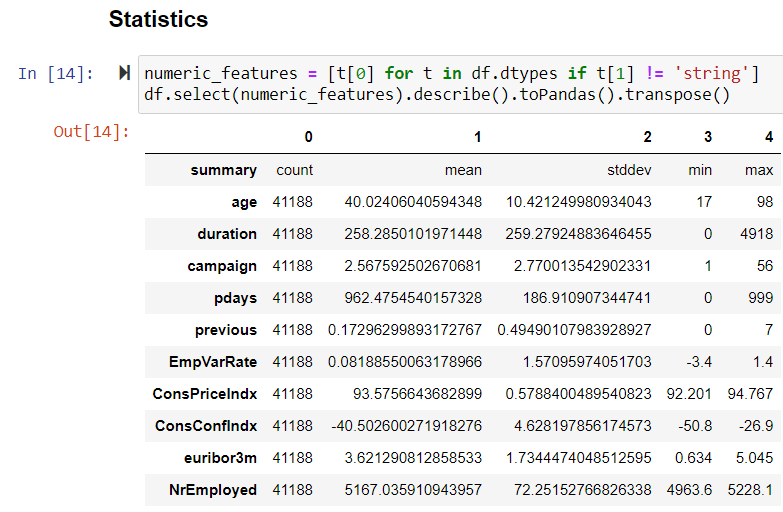
**Figure 4: Distribution of Target Variable**

Continuing our analysis, we wanted to understand the distribution of the target variable (Y) that we are trying to predict later through our models. Figure 4 illustrates this distribution, showing that 36,548 people said “no” to subscribing for a term deposit and 4,640 people said “yes” to this same scenario. Overall, we want to look at each client’s attributes to understand who is likely to say “yes” and who will say “no” to subscribing for term deposits.



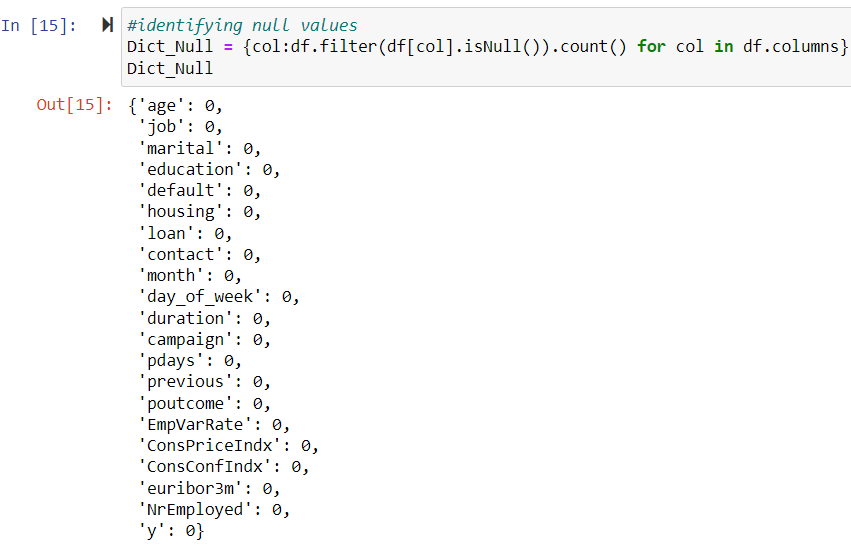
**Figure 5: Renaming Variables**

However, the next step in the process was to rename a few variables in the dataset that contained periods (.) in their name. For example, the original variable “emp.var.rate” was renamed to “EmpVarRate.” This same process was completed to rename three other variables “cons.price.idx,” “nr.employed,” and “cons.conf.idx” to “ConsPriceIndx,” “NrEmployed,” and “ConsConfIndx” respectively. Once printing the schema, its seen that the new renamed variables have replaced the original variables.



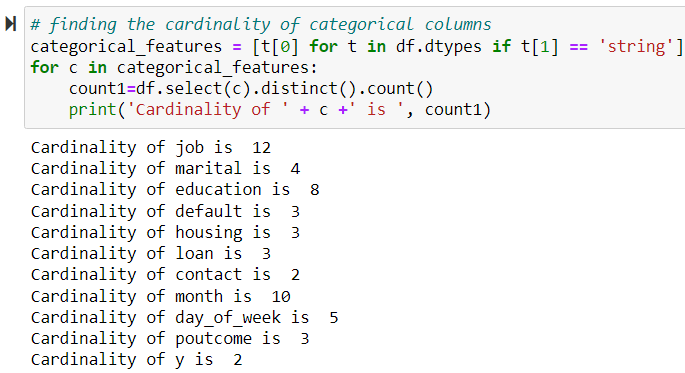
**Figure 6: Variable Statistics**

Additionally, we wanted to view the variable statistics for each of the numeric attributes. In total, there are ten numeric variables in which statistics were collected. The written code in Figure 6 provides the count, mean, standard deviation, minimum, and maximum values for each numeric variable. For example, looking at the summary statistics for the variable “age,” the count is 41,188, the mean is 40.02, the standard deviation is 10.42, the minimum value is 17, and the maximum value is 98.



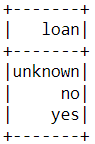
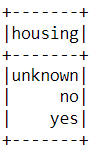
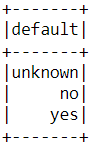
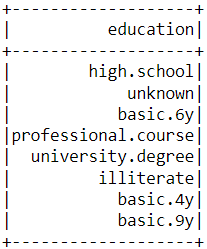
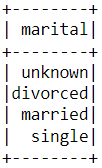
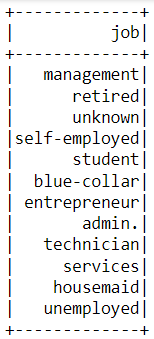
**Figure 7: Identifying Null Values**

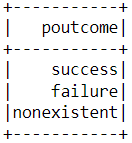
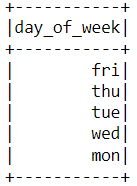
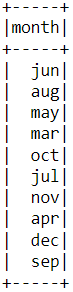
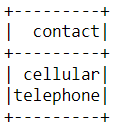
The next step was to identify whether any of the variables contained null values. The code in Figure 7 provided us with a count of null values for each variable in the dataset. Observing the output in Figure 7, there were zero null values for each variable. We deemed it reasonable for the variables that did contain a value of zero because it would not be out of the ordinary for a variable, such as ‘pdays,’ to possess a value of zero.



**Figure 8: Cardinality of Categorical Variables**

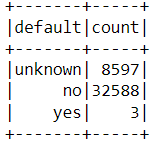
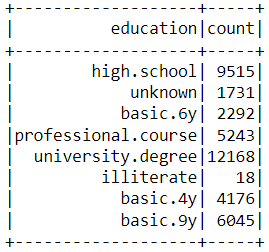
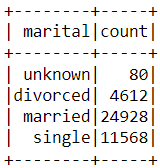
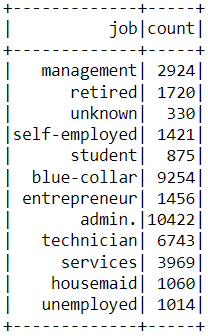
Furthermore, we decided to view the cardinality of the categorical variables to understand the number of unique values that each variable could take on. From Figure 8 above, looking at the first few variables, it is seen that the variable ‘job’ has a cardinality of 12, ‘marital’ has a cardinality of 4, and ‘education’ has a cardinality of 8. This means that the variable ‘job’ can take on one of twelve different options. Likewise, ‘marital’ has four different options that it could take, and ‘education’ has eight unique values. The cardinality is present for each of the categorical variables in the dataset.

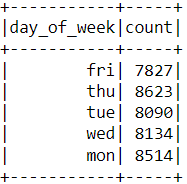
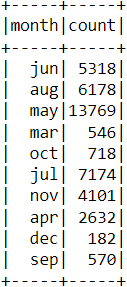
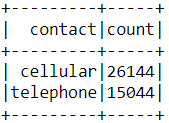
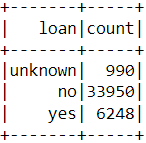
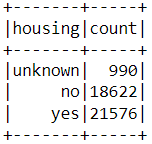


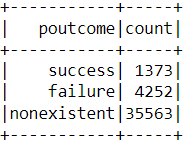


**Figure 9: Distinct Values of Categorical Variables**

Continuing with the idea of cardinality, each of the eleven charts in Figure 9 represents the unique options that each categorical variable can possess. This assists in identifying each individual option present in the categorical variables that a client could use to represent themselves. In particular, knowing these values helps with prediction purposes in the future.

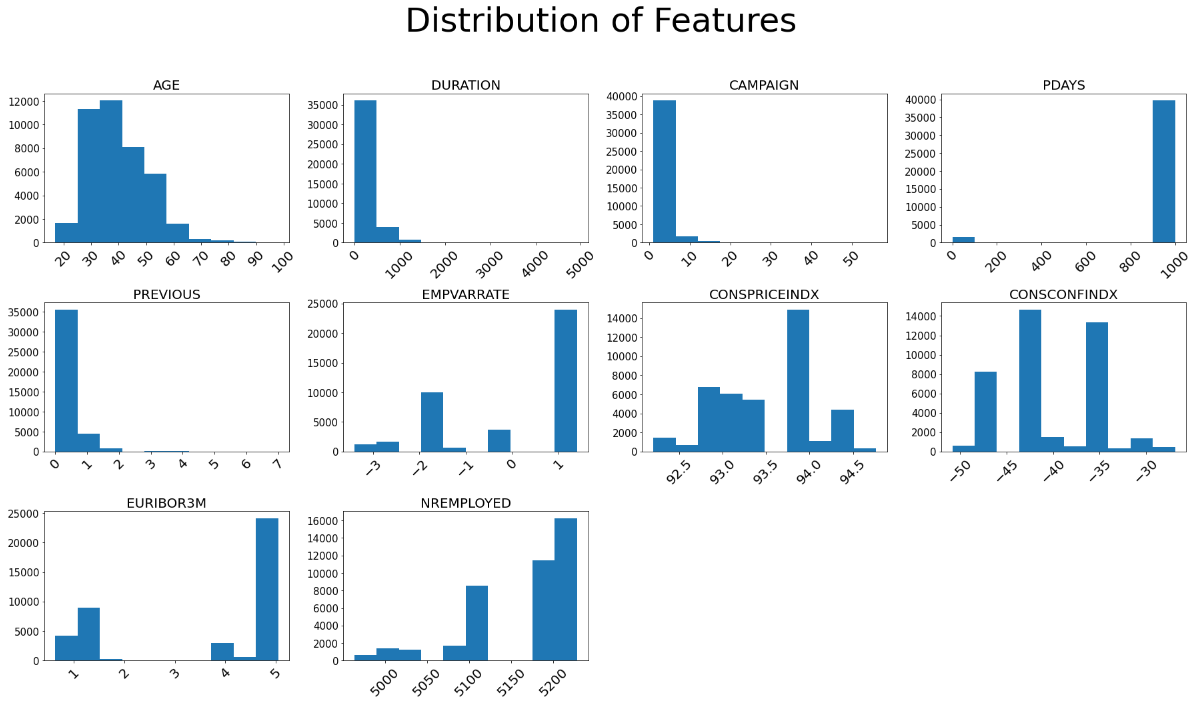






**Figure 10: Count of Distinct Values for Categorical Variables**

Figure 10 illustrates that similar to what was seen in Figure 9; however, these charts are presenting the total count of each unique value for the categorical variables. This allows us to understand which unique value appears most frequently in each variable.

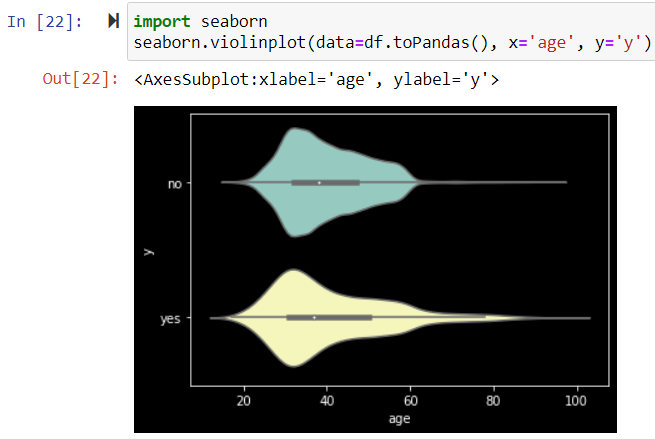


**Figure 11: Distribution of Numeric Feature Variables**

Furthermore, now that we have analyzed the categorical variables, we will look at the distribution of the numeric feature variables. Figure 11 presents histograms for each of the ten numeric variables in the dataset. Similar to what we saw with cardinality, the distributions allow us to see the number of times each value occurred for the numeric variables. For example, when observing the distributions, the majority of clients in the dataset span between the ages of 30 and 40 years old. These distributions are available for all numeric variables. Next, we decided to conduct a bivariate analysis to further enhance our exploratory data analysis.

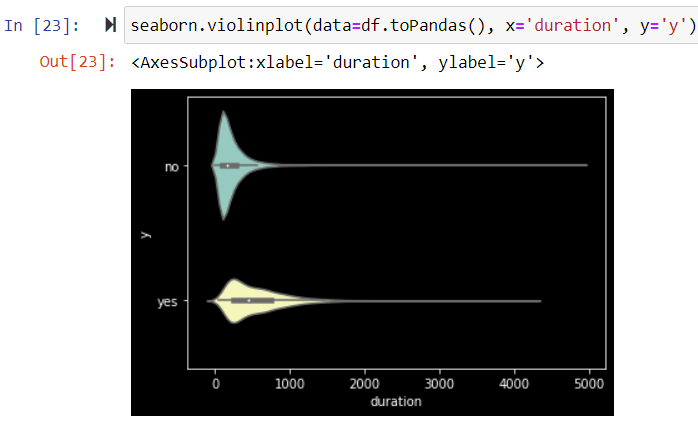
**Bivariate Analysis**

A bivariate analysis enables us to determine the relationship between two variables, a feature variable and the target variable. Below, we will utilize bivariate analyses to explore the relationship between each of the numeric feature variables and the target variable of whether someone will subscribe for a term deposit or not.



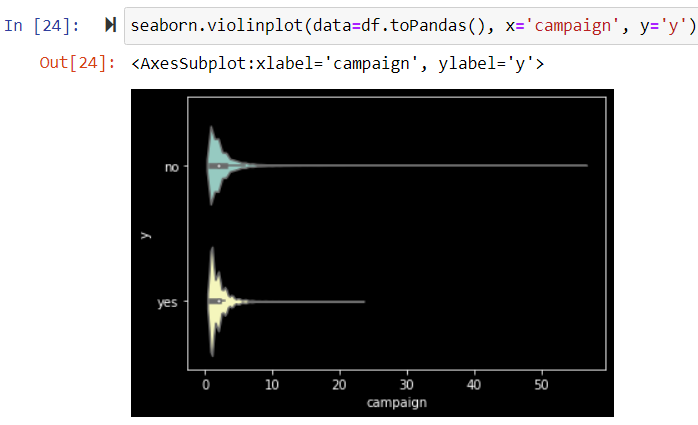
**Figure 12: Bivariate Analysis Age vs. Target**

From Figure 12, when comparing the ‘age’ variable with the target variable (Y), it appears that the greatest number of ‘yes’ and ‘no’ responses occur between the ages of 20 to 40. However, we can say that older people, ages 60 and up, tend to respond ‘yes’ to the campaigns more frequently than not.



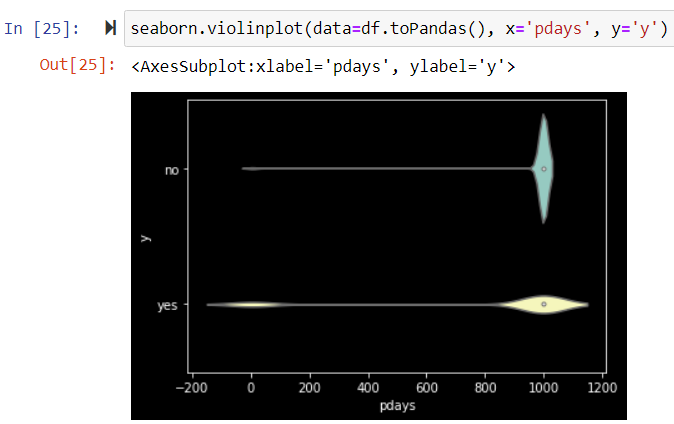
**Figure 13: Bivariate Analysis Duration vs. Target**

When comparing the relationship between the variable ‘duration’ and the target variable, Figure 12 illustrates that the higher the duration, the better chance to get a ‘yes’ response to the marketing campaigns. When the duration time stays around zero in the graph above, there are many more ‘no’ responses; however, as duration extends out to a total of 1,500, people are much likelier to say ‘yes.’



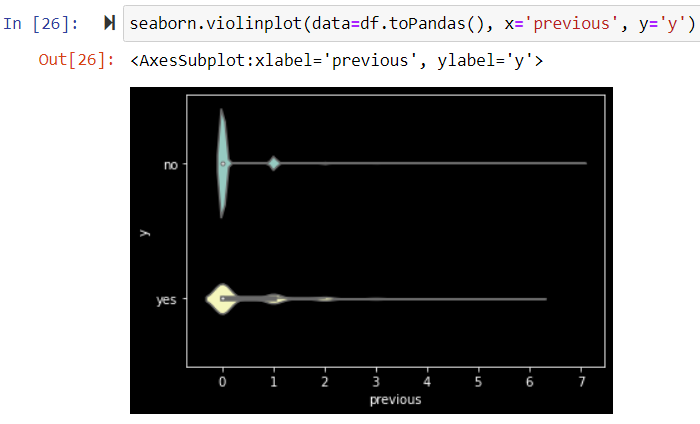
**Figure 14: Bivariate Analysis Campaign vs. Target**

From Figure 14 above, the campaign variable does not show a very high difference in terms of response to campaign. The ‘yes’ and ‘no’ responses follow a very similar pattern across both graphs, meaning ‘campaign’ may not play a large role in predicting the outcome of the target variable.



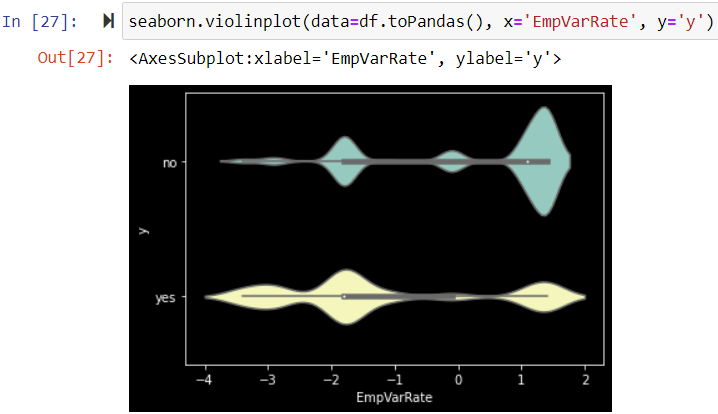
**Figure 15: Bivariate Analysis Pdays vs. Target**

According to Figure 15, the number of days that passed by after the client was last contacted from a previous campaign should be minimized if the bank wants to receive more ‘yes’ responses to from their campaign. There are ‘yes’ responses close to zero in the graph above, meaning people are likelier to say ‘yes’ if they are contacted more frequently. However, as more days pass by, the chances of clients responding ‘yes’ decline tremendously.



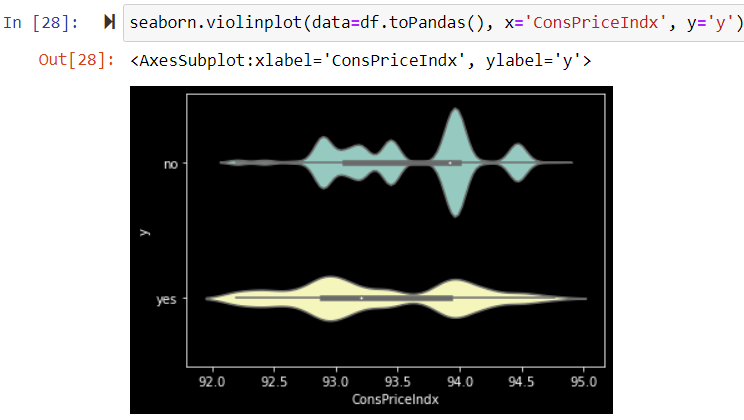
**Figure 16: Bivariate Analysis Previous vs. Target**

As seen in Figure 16 above, the ‘previous’ variable does not show very high difference in terms of response to campaign. Nonetheless, people are still more likely to respond ‘no’ to the campaign if number of contacts performed before this campaign is zero.



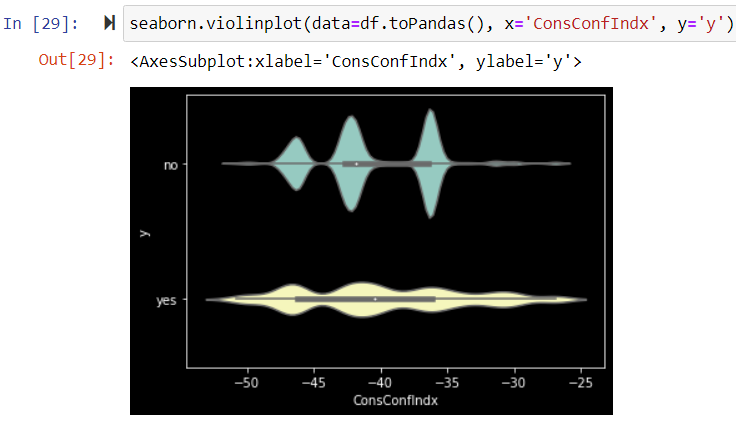
**Figure 17: Bivariate Analysis EmpVarRate vs. Target**

Figure 17 represents the impact that the employment variation rate has on whether a response is ‘yes’ or ‘no.’ The ‘yes’ and ‘no’ responses follow very similar patterns; however, it is likelier that the bank receives a ‘yes’ from their marketing campaign when the employment variation rate numbers are negative. On the other hand, the bank will see more frequent ‘no’ responses when this variable witnesses positive values, especially between 1 and 2.



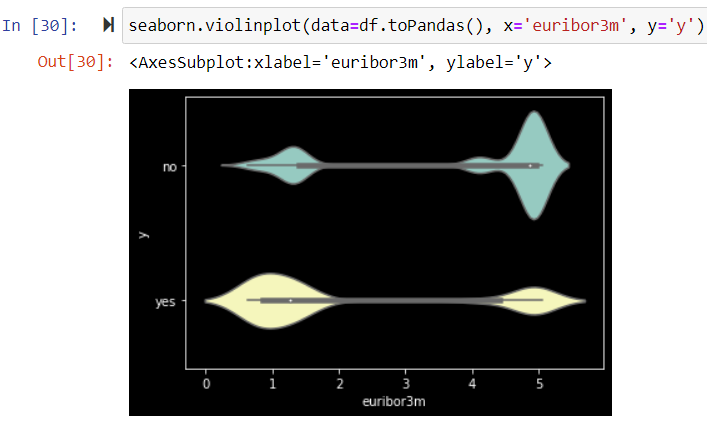
**Figure 18: Bivariate Analysis ConsPriceIndx vs. Target**

Observing Figure 18, simply, the lower the consumer price index, the better chance the bank has in receiving ‘yes’ responses to their marketing campaign. The greater the numbers, the higher chance of clients saying ‘no.’



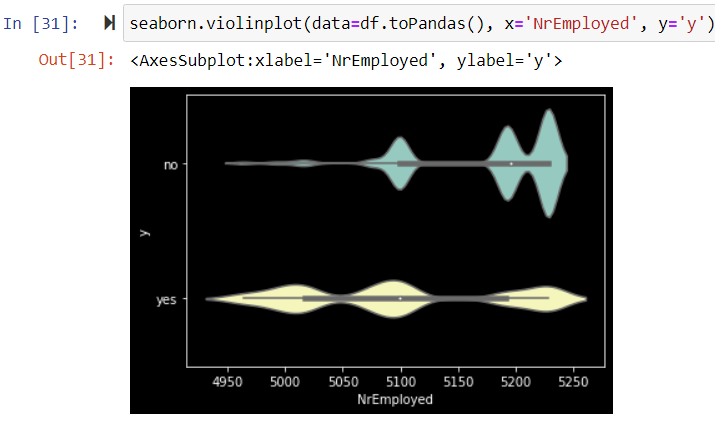
**Figure 19: Bivariate Analysis ConsConfIndx vs. Target**

The impact of ConsConfIndx on the target variable can be observed from Figure 19. From the chart above, other than three areas where the value of ConsConfIndx ranges around -46, -43, and -36, people are more likely to respond ‘yes’ to the bank’s marketing campaign. However, in those three ranges, clients are significantly likelier to respond ‘no.’



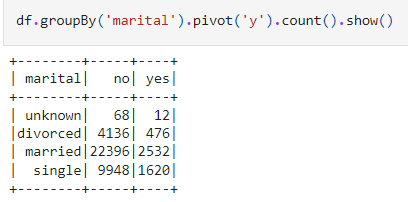
**Figure 20: Bivariate Analysis Euribor3m vs. Target**

The relationship between the variable ‘euribor3m’ and whether a client responds ’yes’ or ‘no’ is opposite from one another. In Figure 20, the chart shows that when the value of euribor3m ranges between 1 and 3, clients are far likelier to respond ‘yes.’ However, when these numbers range higher from 4-5, the bank is going to witness more ‘no’ responses.



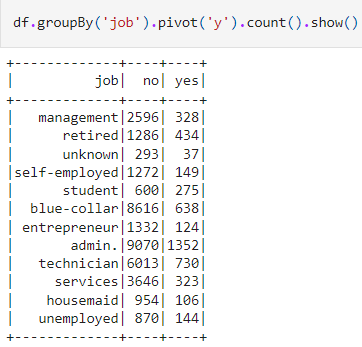
**Figure 21: Bivariate Analysis NrEmployed vs. Target**

Lastly, for the bivariate analysis of numeric variables, Figure 21 illustrates the relationship between the variable ‘NrEmployed’ and the target. From Figure 21, it is observed that when the number of employees is lowest, there is a greater number of ‘yes’ responses as opposed to when the number of employees is higher. In this situation, we realize far more ‘no’ responses. Now that we have examined the bivariate analyses for the dataset’s numeric variables, it is time to dive into the analysis of categorical variables. It should be noted that the category with the greatest number of ‘no’ responses also contains the most ‘yes’ responses because this specific category has the largest number of responses overall.



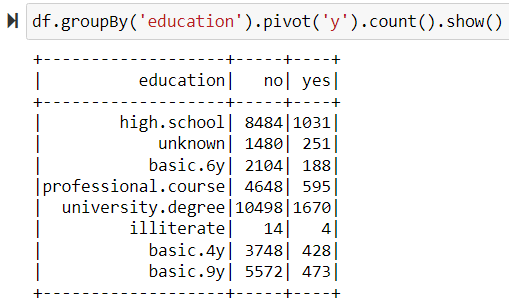
**Figure 22: Bivariate Analysis Marital vs. Target**

Figure 22 above shows the bivariate analysis of the variable ‘martial’ in relation to the target variable. It appears that the marital status of married provided the greatest number of ‘no’ responses and ‘yes’ responses. Thus, the bank should target its campaign towards clients who are married.



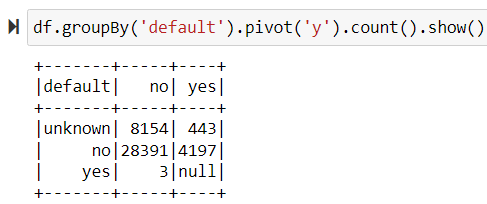
**Figure 23: Bivariate Analysis Job vs. Target**

When observing Figure 23, analyzing the variable ‘job’ in relation to the target variable, it is apparent that the job category of administrator carries the greatest number of ‘no’ and ‘yes’ responses.



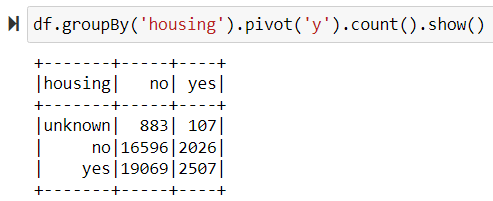
**Figure 24: Bivariate Analysis Education vs. Target**

Next, Figure 24 illustrates the bivariate analysis of the variable ‘education’ and the target variable. The category that provided the greatest number of ‘no’ and ‘yes’ responses are from clients who obtained a university degree. Therefore, the bank can look at people who completed a university degree to subscribe for a term deposit.



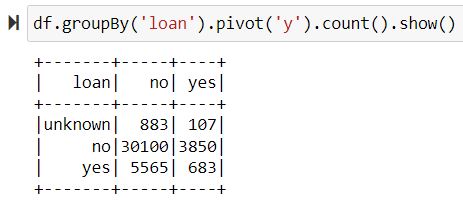
**Figure 25: Bivariate Analysis Default vs. Target**

Continuing, we will observe the relation between the variable ‘default’ and the target variable from Figure 25 above. Clients who do not have any credit defaults are most likely to respond ‘no’ and ‘yes’ to the bank’s marketing campaign. The bank should then focus on people without credit defaults.



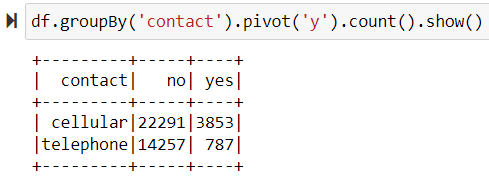
**Figure 26: Bivariate Analysis Housing vs. Target**

On the other hand, when observing the relation between ‘housing’ and the target variable in Figure 26, clients who have taken housing loans are more likely to respond ‘no’ and ‘yes’ to the marketing campaign. Thus, the bank may want to focus the campaign on those who have housing loans.



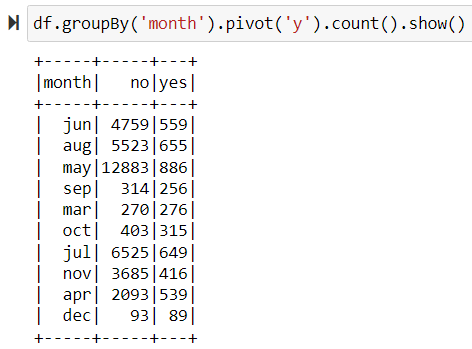
**Figure 27: Bivariate Analysis Loan vs. Target**

Furthermore, when analyzing the results from Figure 27 above, it is shown that clients who have a personal loan responded both ‘no’ and ‘yes’ the greatest number of times in comparison to the other categories. Therefore, it could be beneficial to focus on people without personal loans.



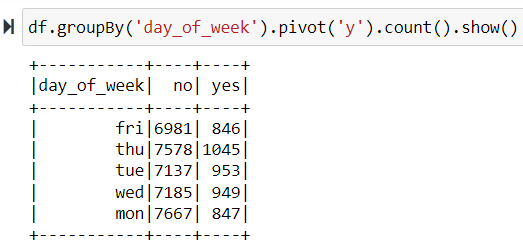
**Figure 28: Bivariate Analysis Contact vs. Target**

When observing Figure 28, analyzing the variable ‘contact’ in relation to the target variable, it is apparent that the cellular category provides the greatest number of ‘no’ and ‘yes’ responses. Thus, the bank should focus on contacting people via their cellphones to have the greatest chance of having the clients respond ‘yes.’



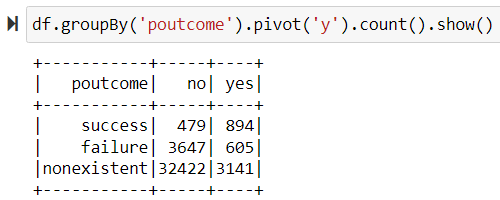
**Figure 29: Bivariate Analysis Month vs. Target**

Next, Figure 29 above shows the bivariate analysis of the variable ‘month’ in relation to the target variable. It appears that the month of May offers the greatest number of ‘no’ responses and ‘yes’ responses.



**Figure 30: Bivariate Analysis Day\_of\_Week vs. Target**

Figure 30 is illustrating the bivariate analysis of the variable ‘day\_of\_week’ in comparison to the target variable. The days of the week provided similar numbers; however, Monday provided the greatest number of ‘no’ responses and Thursday offered the highest number of ‘yes’ responses.



**Figure 31: Bivariate Analysis Poutcome vs. Target**

Lastly, Figure 31 illustrates the bivariate analysis of the variable ‘poutcome’ in comparison to the target variable. When the outcome of the previous campaign is nonexistent, there are a vast majority of ‘no’ responses and ‘yes’ responses in comparison to the other categories. Therefore, perhaps, campaigns for the bank are more effective when clients have not been reached out to before.

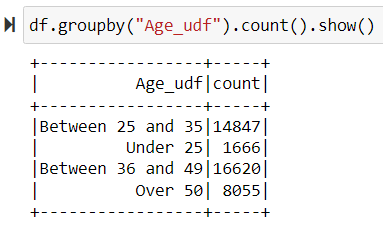
**Correlation**



**Figure 32: Correlation of Variables**

The chart above seen in Figure 32 shows the Pearson correlation to understand which variables are highly correlated with one another. There are a few variables that have a strong correlation, such as ‘euribor3m’ and ‘EmpVarRate’ which has a correlation of 0.972. This is the strongest correlation between two variables in the dataset. Additionally, ‘euribor3m’ and ‘NrEmployed’ have a strong correlation at 0.945. Lastly, two variables that have a correlation greater than 0.9 are ‘NrEmployed’ and ‘EmpVarRate’ which has a correlation of 0.907. Therefore, there are a couple variables that have a strong correlation and have an impact on each other.

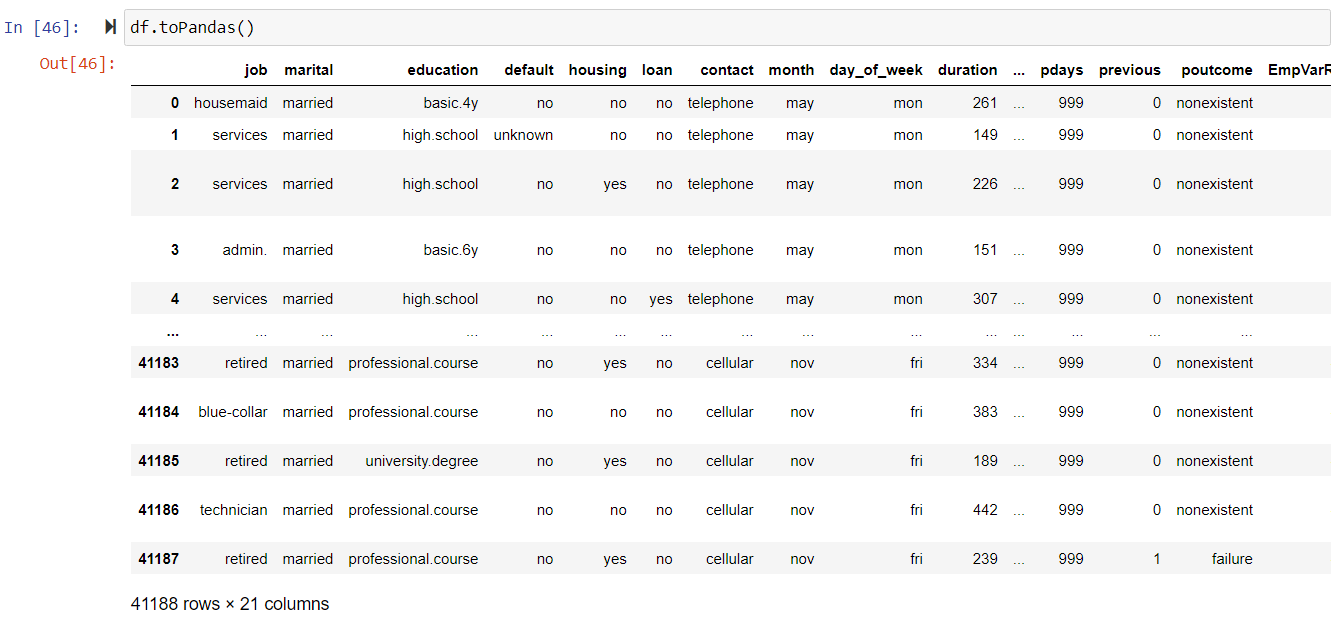
**Creating Age Bin**



**Figure 33: Creating an Age Bin Using UDF**

Our final step in transforming the data, as seen in Figure 33, was to create a new variable called ‘Age\_udf’ that binned the ages together into separate groups. When creating this new variable, we grouped everyone under the age of 25 together, everyone between the ages of 25 and 30, everyone between the ages of 36 and 49, and everyone over the age of 50 together. This new variable ‘Age\_udf’ was then utilized to replace the existing ‘age’ variable in the dataset. This provided us with an additional string/categorical variable.

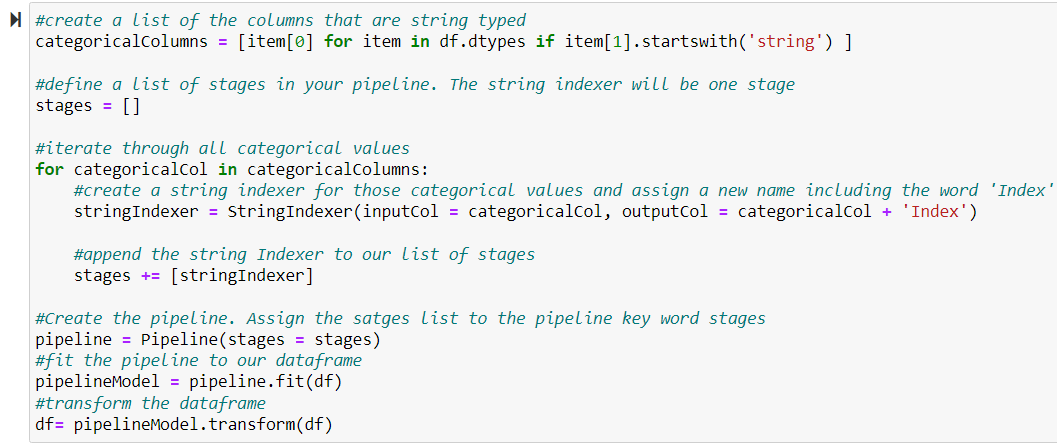
**Creating Pandas Table**



**Figure 34: Creating a Table Using Pandas**

Once the exploratory analysis was complete and we were finished adjusting the variables, we were ready to utilize the Pandas function to create our data frame. Figure 34 above illustrates a few of the variables present in the data frame along with some of the rows of data. Overall, the count of rows and columns did not change from our original dataset. We still have a total of 41,188 rows and 21 columns.

**String Indexer**

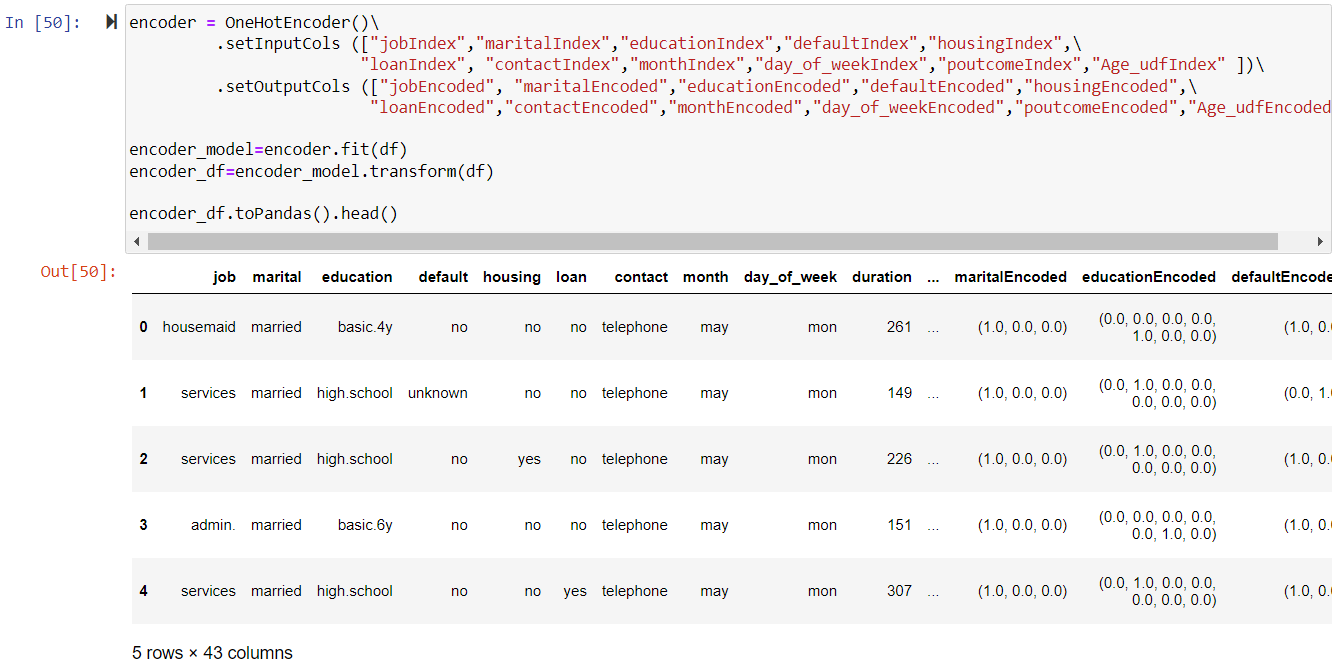
Next, we needed to prepare the variables for modeling. The categorical string variables needed to be converted to numerical values. We created a loop that made a list of all string variables, then with another for loop created index values for those string variables. Next using pipeline.fit and pipeline.transform transformed the data frame to include the index variables we created. The new index variables followed a naming convention of “original categorical name” + “Index”. 

**Figure 35: String Indexer**

Because the target variable was also a string an Index variable was created for it. We will deal with target variable later with a label index, so for now we dropped ’yIndex’ variable.

**One Hot Encoding**

While the categorical string variables had been converted to numerical values, they still cannot be treated as a true continuous variable. Because they are categorical a “2” value doesn't represent a value twice as much as a “1”, it is simply a different category. To use these indexes for modeling, they must be transformed using one hot encoding. One hot encoding creates a new column or variable for each different value in the categorical index variable. As we can see in the figure below after being converted “maritalEncoded” contains three separate variables separated by commas. There are four values of marital status, unknown, married, single, and divorced, but a zero in the first three columns would indicate the original value is the fourth.



**Figure 36: One Hot Encoding**

**Vector Assembler**

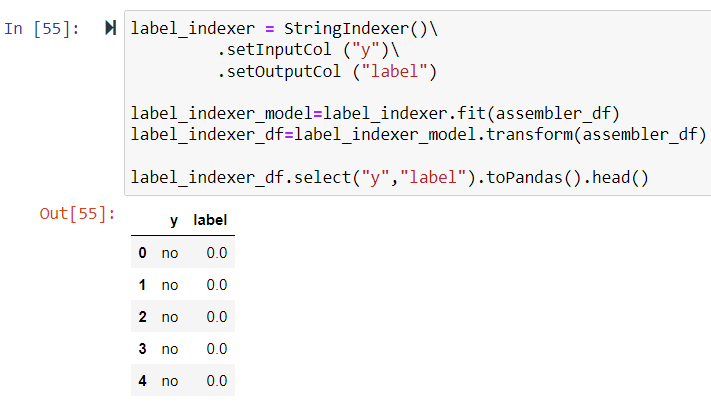
Now that the categorical variables have been properly converted for use in the model, the next step is to assemble all the input variables into one vector. This results in a single vector variable called “vectorized\_features”.



**Figure 37: Vector Assembler**

**Label Indexer**

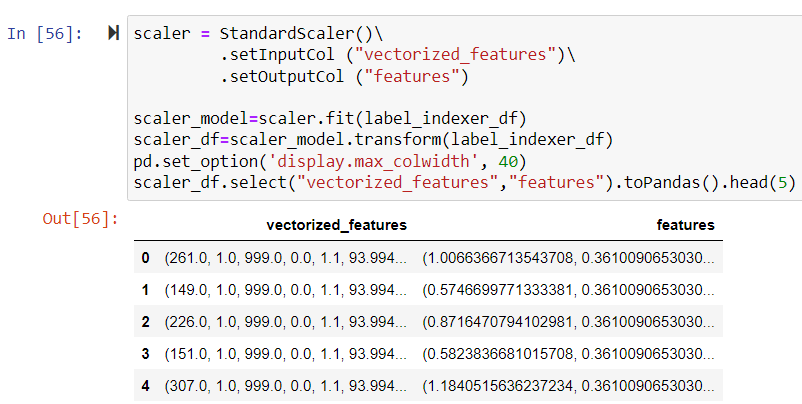
As mentioned before, we used the lable\_indexer function to transform the target to an index value. Original values of “no” were transformed to 0.0 and values of “yes” were transformed to 1.0.



**Figure 38: Label Indexer**

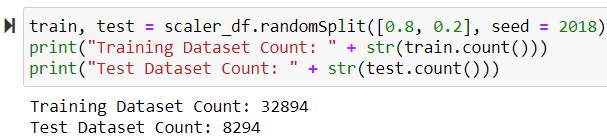
**Standard Scaler**

The last step to prepare the variables for modeling was to scale the vectorized features. This process entailed using the StandardScaler function. In the figure below we can see the original vectorized features and the result after the scaling. After scaling we had a data frame, we called scaler\_df that contained the transformed variables in a single vectorized and scaled variable called “features” and a target variable called “label”.



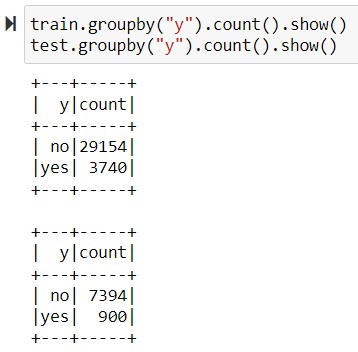
**Figure 39: Standard Scalar**

**Train/Test Data**



**Figure 40: Count of Training and Test Data**

To begin the modeling process, we first split our data into a training dataset and test dataset, as seen in Figure 40. We decided to put 80% of the data in the training dataset and 20% in the test dataset. Overall, this gave us 32,894 rows in the training dataset and 8,294 rows in the test dataset. The training dataset allows our model to learn the data and then use the test data to make predictions based on what it learned.



**Figure 41: Count of Training and Test Data by Target Variable Outcomes**

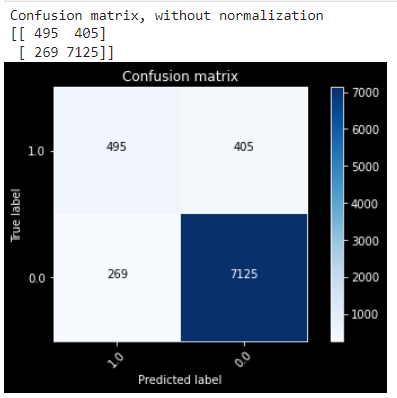
Next, we also wanted to see the split of data based on the clients who responded ‘yes’ and those who responded ‘no.’ Therefore, looking at the training data, its seen that 29,154 clients responded ‘no’ and 3,740 responded ‘yes.’ Likewise, in the test dataset, 7,394 clients responded ‘no’ while 900 responded ‘yes.’

**Predictive Models**

|  |  |  |
| --- | --- | --- |
| **Model Name** | **Accuracy** | **Test ROC** |
| Logistic Regression | 0.915 | 0.937 |
| Decision Tree | 0.918 | 0.802 |
| Random Forest | 0.907 | 0.926 |
| Gradient Boosting | 0.919 | 0.946 |
|  |  |  |

**Figure 42: Accuracy and ROC Values of Models**

Overall, we created four different models, including a logistic regression, a decision tree, a random forest, and a gradient boosting model. Looking at the models’ performances in Figure 42, the Gradient Boosting model performed the best in both Accuracy and ROC value. The Gradient Boosting model had a 91.9% accuracy rating with an ROC value of 0.946, making it our champion model. Overall, Gradient Boosting is a decision tree ensemble learning algorithm for classification and regression. Ensemble learning techniques combine multiple machine learning algorithms to obtain a better model. The idea of gradient boosting is improving a single weak model by combining it with several other weak models in order to generate a collectively strong model. Gradient boosting iteratively trains an ensemble of decision trees, with each iteration using the error residuals of the previous model to fit the next model. The final prediction is a weighted sum of all the decision tree predictions. Gradient boosting helps to minimize bias and underfitting.

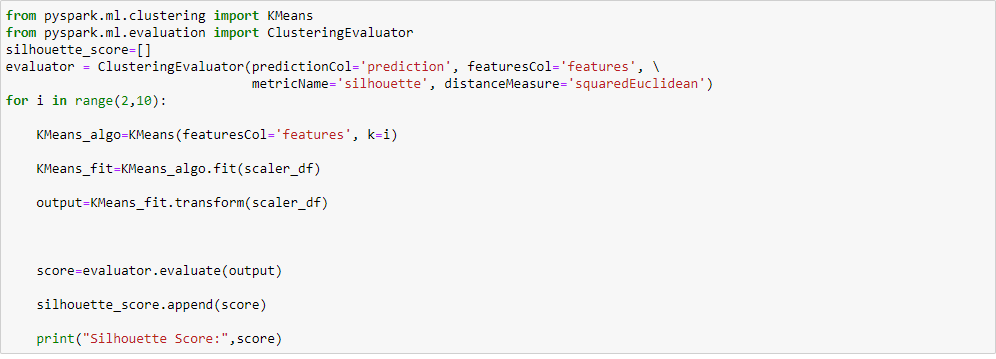


**Figure 43: Gradient Boosting Confusion Matrix**

Above, in Figure 43, is the Confusion Matrix for our Gradient Boosting model. Overall, the model correctly predicted 7,620 clients, 495 of those being true positives and 7,125 being true negatives. On the other hand, the model predicted 674 clients incorrectly. The model predicted that 269 clients would respond ‘yes’ to the term deposit when, in fact, they responded ‘no.’ Likewise, the model predicted that 405 clients would respond ‘no’ to subscribing for a term deposit when they actually responded ‘yes.’ Nonetheless, this model had a great accuracy rating as mentioned previously at 91.9%.

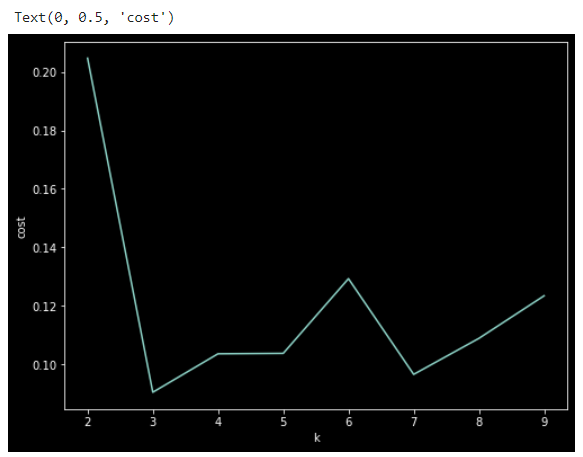
**K-means Clustering**

K-means clustering is an unsupervised machine learning technique to create clusters that group similar records of a dataset together. To determine the proper number of clusters we created a for-loop that created clusters of 2 to 9 groups and reported the Silhouette Score distance. Below is a snippet of the code.



**Figure 44: K-means Clustering Code**

After running the above code and adding the Silhouette Score, we created the visualization below. As you can see the K-means model with two clusters has the best Silhouette score. This made sense to us because the target variable had only two possible values. Clustering the data into two groups seemed logical.

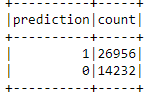


**Figure 45: Selecting the Ideal Number of Clusters**

We then applied the two cluster K-means model to the data set with code below. The result was a predictions variable that grouped the variables into one of two groups. The result was a group 0 with 14,232 members and a group 1 with 26,956 members.



**Figure 46: Creating Two Clusters Using K-means**



**Figure 47: Prediction and Count of K-means Clusters**