**Money Maker or Money Loser?**

Braden Heuglin, [bheuglin2@bellarmine.edu](mailto:bheuglin2@bellarmine.edu)

**ABSTRACT**

I am applying a confusion matrix ML Classification model to a dataset on customers at a Portuguese Bank. Some information about the customers include age, marital status, education, duration with the bank, and so on. The bank is using this information to decide which customers they will drop and which ones they will keep. I am going to use a confusion matrix and predictor values to see which customers I think will be dropped, versus which customers are actually dropped.

1. **INTRODUCTION**

As I mentioned in the abstract above, I chose to work with a dataset on the background information of customers at a Portuguese Bank. As a finance major, I am very interested in money and how it works, including what makes a customer one a company wishes to keep. The ML Classification model I chose to apply is the confusion matrix, classification report, and accuracy score. The reason for this is because the dataset is looking at which customers should be dropped based on their ability to generate revenue for the bank.

1. **BACKGROUND**
   1. *Data Set Description*

I got my dataset from Kaggle by simply searching for “Finance” datasets and selecting the one pertaining to a Portuguese Bank. The Bank collected this data because it noticed it was losing revenue and wanted to find out which customers were losing them revenue and which ones were generating them revenue. After determining which customers were losing them money, they decided to drop those customers and not allow them to continue with the bank.

* 1. *Machine Learning Model*

I used a confusion matrix, classification report, and accuracy score to predict which customers would be dropped, and compared my predictions to the actual results of which customers were dropped.

1. **EXPLORATORY ANALYSIS**

My dataset contains 16 columns, with almost 33,000 data entries. There were no missing values or extreme outliers which threw the data off. I didn’t have to put any work into cleaning the data up, I just had to encode objects into integers for the confusion matrix to work.

**Table 1: Data Types**

|  |  |
| --- | --- |
| *Variable Name* | *Data Type* |
| Age | Categorical Int |
| Job | Categorical Object |
| Marital | Categorical Object encoded to Int |
| Education | Categorical Object encoded to Int |
| Default | Categorical Object encoded to Int |
| Housing | Categorical Object encoded to Int |
| Loan | Categorical Object encoded to Int |
| Contact | Categorical Object |
| Month | Categorical Object |
| Day of Week | Categorical Object |
| Duration | Quantitative Int |
| Campaign | Quantitative Int |
| PDays | Quantitative Int |
| Previous | Categorical Int |
| POutcome | Categorical Object encoded to Int |
| Y | Categorical Object encoded to Int |

1. **METHODS**
   1. *Data Preparation*

After reading the dataset into python, I used the .info() function to find out what each columns data type was. Once I found that out, I was able to see which columns I needed to encode. Then, I used the pad.get\_dummies() function to predict what the outcomes could be versus what the actual outcomes were.

After encoding the necessary columns, I used the .isnull.sum() function to check for any missing values. My dataset had no missing values so I did not have impute anything and was able to move directly into splitting the data into a training and test set.

* 1. *Experimental Design*

I ran four different tests and changed the random state and test size in each case. The results are shown below:

Table X: Experiment Parameters

|  |  |
| --- | --- |
| **Experiment Number** | **Parameters** |
| 1 | 80/20 split for train and test set with a random state of 17 |
| 2 | 70/30 split for train and test set with a random state of 17 |
| 3 | 80/20 split for train and test set with a random state of 23 |
| 4 | 75/25 split for train and test set with a random state of 23 |

* 1. *Tools Used*

The following tools were used for this analysis: Windows computer in Bellarmine’s library, Python as the main coding database, numpy, matplotlib.pyplot (not used but imported anyway), pandas, seaborn, sklearn.model\_selection (train\_test\_split), sklearn.linear\_model (LogisticRegression), sklearn.metrics (confusion\_matrix, accuracy\_score, ConfusionMatrixDisplay, and classification\_report).

1. **RESULTS**
   1. *Classification Measures*

Experiment 1:

A picture containing text, screenshot, line, software

Description automatically generated

Classification Report:

precision recall f1-score support

0 0.92 0.97 0.95 5840

1 0.63 0.34 0.44 750

accuracy 0.90 6590

macro avg 0.78 0.66 0.69 6590

weighted avg 0.89 0.90 0.89 6590

Experiment 2:

A screen shot of a graph

Description automatically generated with low confidence

Classification Report:

precision recall f1-score support

0 0.92 0.97 0.95 8768

1 0.63 0.34 0.44 1117

accuracy 0.90 9885

macro avg 0.78 0.66 0.69 9885

weighted avg 0.89 0.90 0.89 9885

Experiment 3:

A screen shot of a graph

Description automatically generated with low confidence

Classification Report:

precision recall f1-score support

0 0.92 0.98 0.95 5853

1 0.66 0.34 0.45 737

accuracy 0.91 6590

macro avg 0.79 0.66 0.70 6590

weighted avg 0.89 0.91 0.89 6590

Experiment 4:

A screenshot of a computer screen

Description automatically generated with low confidence

Classification Report:

precision recall f1-score support

0 0.92 0.98 0.95 7314

1 0.65 0.34 0.44 924

accuracy 0.91 8238

macro avg 0.79 0.66 0.70 8238

weighted avg 0.89 0.91 0.89 8238

* 1. *Discussion of Results*

Experiment 1:

Confusion matrix and classification reports above

* True Negative: 5690
* True Positive: 256
* False Positive: 150
* False Negative: 494

Accuracy score: 90.23%

Based on my confusion matrix, classification report, and accuracy score, my predictor is pretty accurate with an accuracy of over 90%.

Experiment 2:

Confusion matrix and classification reports above

* True Negative: 8548
* True Positive: 380
* False Positive: 220
* False Negative: 737

Accuracy score: 90.32%

Based on my confusion matrix, classification report, and accuracy score, my predictor is pretty accurate with an accuracy of over 90%.

Experiment 3:

Confusion matrix and classification reports above

* True Negative: 5723
* True Positive: 249
* False Positive: 130
* False Negative: 488

Accuracy score: 90.62%

Based on my confusion matrix, classification report, and accuracy score, my predictor is pretty accurate with an accuracy of over 90%.

Experiment 4:

Confusion matrix and classification reports above

* True Negative: 7158
* True Positive: 310
* False Positive: 166
* False Negative: 614

Accuracy score: 90.53%

Based on my confusion matrix, classification report, and accuracy score, my predictor is pretty accurate with an accuracy of over 90%.

* 1. *Problems Encountered*

The most common problem I ran into was selecting the right dataset. I ended up trying to use four different datasets before I found one that would work with my classification model. However, once I found a dataset that worked, I only encountered one small problem. The small problem I encountered was which columns to encode to include in the training and test sets to use as predictors. I had to play around with a few of the columns to get the right ones that worked properly with my model.

* 1. *Limitations of Implementation*

My model is limited by the number of customers whose information was collected. It also only collected limited information on the customers, so outside factors which were looked at may not have been included in the dataset. This could result in my predictions being wrong, as it only looked at very specific information collected on each customer.

* 1. *Improvements/Future Work*

I could have included more plots to display any skew in the data. On top of that, I could have ran the experiment using different variables in my prediction model. Other than that, I think my model worked pretty well as it consistently produced an accuracy score of right around 90% which is decent.

1. **CONCLUSION**

In conclusion, as a finance major, I love working with money which is why I wanted to work with a dataset which also worked with money. This dataset was collected by a Portuguese bank because it realized it was losing revenue. They collected various background information on their customers to determine which customers were causing their revenue loss. After collecting the data and seeing which customers were causing the revenue loss, they decided to drop those customers to prevent further loss.

I ran a confusion matrix, classification report, and accuracy score on this dataset to predict which customers I thought would be dropped. I then compared what my model predicted to what actually happened and checked to see how accurate my model was. My model consistently had an accuracy score of around 90% which is decent, so overall, I am happy with my prediction model.

**REFERENCES**

“Search.” *Kaggle*, <https://www.kaggle.com/search?q=portuguese%2Bbank>..

“Sklearn.metrics.classification\_report.” *Scikit*, <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html>.

Various Individual Projects and In Class Assignments on Python from this past semester.

**Other directions:**

1. 10-pt, Times New Roman, 1” margins all around (if you use this template you are already set).
2. Ensure all tables and figures are numbered appropriately and referenced in the text. See examples above and below.

|  |  |
| --- | --- |
| **Figure 1: Comparison of X/Y from dataset (single plot) (8 pt.)** | **Figure 2: (a) Function Output (b) A against B (multiple plots) (8 pt.)** |