Phase 1

Kel Gruber

November 26, 2023

1 Phase Goal

In this phase, I cleaned and prepared the "Bank Marketing Data Set" from Kaggle Data Science. to be used in a Neural Network model. In addition, after the data has been cleaned the data was visualized and preliminary data analysis will be performed.

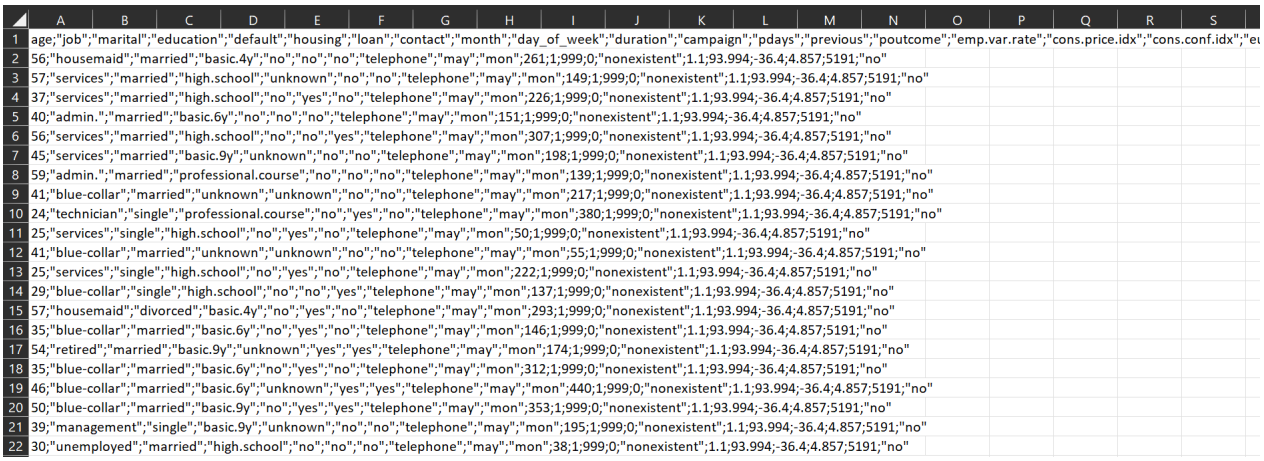
2 Dataset

This project uses the "Bank Marketing Data Set" from Kaggle Data Science. This data was obtained from the direct marketing campaigns of a Portuguese banking institution to its clients. These contacts were made by phone calls to either the customer's home telephone or cellphone.

This data set initially consists of 41,188 instances and 20 feature variables. These variables are a mix of continuous data and non-numeric classification data. The original output variable is labeled 'y' and has 36548 'no' values and 4640 'yes' values. For clarity, this column was relabeled at the beginning of the project to 'output'.
Feature Information:

1. age : (numeric) - age of the client
2. job : (categorical) - type of job
3. marital : (categorical) - marital status
4. education : (categorical) - education level
5. default: (categorical) - If the client has credit in default
6. housing: (categorical) - If client has a housing loan
7. loan: (categorical) - If client has personal loan
8. contact: (categorical) - Contact communication type
9. month: (categorical) - Last contact month of year
10. day.of.week: (categorical) - Last contact day of the week
11. duration: (numeric) - Last contact duration, in seconds
12. campaign: (numeric) - number of contacts performed during this campaign and for this client
13. pdays: (numeric) Number of days that passed by after the client was last contacted from a previous campaign; 999 means client was not previously contacted
14. previous: (numeric) number of contacts performed before this campaign and for this client
15. poutcome: (categorical) - Outcome of the previous marketing campaign
16. emp.var.rate: (numeric) - Employment variation rate - quarterly indicator
17. cons.price.idx: (numeric) - Consumer price index - monthly indicator

18. cons.conf.idx: (numeric) - Consumer confidence index - monthly indicator
19. euribor3m: (numeric) - The Euribor 3 month rate - daily indicator
20. nr.employed: (numeric) - Number of bank employees - quarterly indicator
21. y: (categorical) - 'yes' or 'no' to indicated if a client has opened a term deposit account with the bank or not, I renamed this column to 'output'



	A
1	age;"job";"marital";"education";"default";"housing";"loan";"contact";"month";"day_of_week";"duration";"campaign";"pdays";"previous";"poutcome";"emp.var.rate";"cons.price.idx";"cons.conf.idx";"e
2	56;"housemaid";"married";"basic.4y";"no";"no";"no";"telephone";"may";"mon";261;1;999;0;"nonexistent";1.1;93.994;-36.4;4.857;5191;"no"
3	57;"services";"married";"high.school";"unknown";"no";"no";"telephone";"may";"mon";149;1;999;0;"nonexistent";1.1;93.994;-36.4;4.857;5191;"no"
4	37;"services";"married";"high.school";"no";"yes";"no";"telephone";"may";"mon";226;1;999;0;"nonexistent";1.1;93.994;-36.4;4.857;5191;"no"
5	40;"admin.";"married";"basic.6y";"no";"no";"no";"telephone";"may";"mon";151;1;999;0;"nonexistent";1.1;93.994;-36.4;4.857;5191;"no"
6	56;"services";"married";"high.school";"no";"no";"yes";"telephone";"may";"mon";307;1;999;0;"nonexistent";1.1;93.994;-36.4;4.857;5191;"no"
7	45;"services";"married";"basic.9y";"unknown";"no";"no";"telephone";"may";"mon";198;1;999;0;"nonexistent";1.1;93.994;-36.4;4.857;5191;"no"
8	59;"admin.";"married";"professional.course";"no";"no";"no";"telephone";"may";"mon";139;1;999;0;"nonexistent";1.1;93.994;-36.4;4.857;5191;"no"
9	41;"blue-collar";"married";"unknown";"unknown";"no";"no";"telephone";"may";"mon";217;1;999;0;"nonexistent";1.1;93.994;-36.4;4.857;5191;"no"
10	24;"technician";"single";"professional.course";"no";"yes";"no";"telephone";"may";"mon";380;1;999;0;"nonexistent";1.1;93.994;-36.4;4.857;5191;"no"
11	25;"services";"single";"high.school";"no";"yes";"no";"telephone";"may";"mon";50;1;999;0;"nonexistent";1.1;93.994;-36.4;4.857;5191;"no"
12	41;"blue-collar";"married";"unknown";"unknown";"no";"no";"telephone";"may";"mon";55;1;999;0;"nonexistent";1.1;93.994;-36.4;4.857;5191;"no"
13	25;"services";"single";"high.school";"no";"yes";"no";"telephone";"may";"mon";222;1;999;0;"nonexistent";1.1;93.994;-36.4;4.857;5191;"no"
14	29;"blue-collar";"single";"high.school";"no";"no";"yes";"telephone";"may";"mon";137;1;999;0;"nonexistent";1.1;93.994;-36.4;4.857;5191;"no"
15	57;"housemaid";"divorced";"basic.4y";"no";"yes";"no";"telephone";"may";"mon";293;1;999;0;"nonexistent";1.1;93.994;-36.4;4.857;5191;"no"
16	35;"blue-collar";"married";"basic.6y";"no";"yes";"no";"telephone";"may";"mon";146;1;999;0;"nonexistent";1.1;93.994;-36.4;4.857;5191;"no"
17	54;"retired";"married";"basic.9y";"unknown";"yes";"yes";"telephone";"may";"mon";174;1;999;0;"nonexistent";1.1;93.994;-36.4;4.857;5191;"no"
18	35;"blue-collar";"married";"basic.6y";"no";"yes";"no";"telephone";"may";"mon";312;1;999;0;"nonexistent";1.1;93.994;-36.4;4.857;5191;"no"
19	46;"blue-collar";"married";"basic.6y";"unknown";"yes";"yes";"telephone";"may";"mon";440;1;999;0;"nonexistent";1.1;93.994;-36.4;4.857;5191;"no"
20	50;"blue-collar";"married";"basic.9y";"no";"yes";"yes";"telephone";"may";"mon";353;1;999;0;"nonexistent";1.1;93.994;-36.4;4.857;5191;"no"
21	39;"management";"single";"basic.9y";"unknown";"no";"no";"telephone";"may";"mon";195;1;999;0;"nonexistent";1.1;93.994;-36.4;4.857;5191;"no"
22	30;"unemployed";"married";"high.school";"no";"no";"no";"telephone";"may";"mon";38;1;999;0;"nonexistent";1.1;93.994;-36.4;4.857;5191;"no"

Figure 1: Initial Banking Dataset.

3 Data Processing

The initial downloaded dataset was stored in a CSV (comma-separated values) file with the data delimited by semicolons. To prepare this data to work with in the Jupyter Notebook I loaded it into Excel and used the Text-to-Columns tool to separate the data into individual columns. I then did a cursory review of the data and its features in Excel and this is when I changed the name of the 'y' column to 'output' to prevent confusion later in the project. While exploring the initial dataset in Excel it became clear that quite a bit of cleaning and processing was required before it was ready for building a model.

I then loaded this file into my Jupyter Notebook and reviewed the shape of the data. Initially, there were 41,188 instances, 20 input variables, and 1 output variable. This is plenty of data for building a neural network model, and it is not square-shaped so we do not have to worry about having too many features. At this time I also reviewed the features themselves and decided to remove some that were not relevant to whether someone opens a term deposit account or not. For example, the contact column only contains two categories 'telephone' and 'cellphone'. These two categories are essentially the same form of communication so it will not have any impact on whether a client opened a term account or not. In addition, the number of employees employed at the bank, and the month and day of the week a client was contacted are not directly related to the client's decision to make a deposit so I removed these columns as well. Columns that are all one value or do not directly relate to the problem do not affect the output variable but can create noise in the model and its predictions. I also decided to drop the duration column which stored the total seconds the bank spoke with a customer on the phone, per the author's suggestion that can lead to bias in the model and poor predictions.

I then looked for missing values. There were no null values in this dataset however, further exploration showed that there were plenty of rows with 'unknown' values stored in several different features in the dataset. Looking at the default column I saw that it had 8,597 'unknown' values, only 3 'yes' values, and the remainder were 'no' values. Since this column is almost entirely the same value 'no' it was also dropped to prevent noise in our model.

I decided for the remainder of the columns with 'unknown' values that columns since they had only a small number of instances with 'unknown', approximately only 2-3% of the total data that I

could safely remove these rows from the dataset. In addition, since most of this data was categorical, attempting to replace the 'unknown' values with a mean value did not make sense. So all the rows that had 'unknown' values in the education, job, marital status, housing, and loan columns were then removed.

Now that I had handled any missing values it was time to convert any categorical data from non-numerical data to numeric data. For the housing, loan, and output columns this process was simple. I just replaced the 'no' values with 0 and the 'yes' values with 1. For the education column, there were 7 categories present: illiterate, basic 4y, basic 6y, basic 9y, high school, professional course, and university degree. For these categories, there is a basic hierarchical structure to them, for example, being illiterate is the same as having no education while a university degree is the same as the highest education an individual will receive. So I replaced these categories with numbers that correspond to the height of the completed education so 'illiterate' was replaced with 0, 'university degree' was replaced with 6 and the other values were ranked in order of education completion accordingly from 1 to 5.

The marital and job columns were trickier to convert to numeric data. These columns are categorical data with no hierarchical structure to them. For example, single, married, and divorced have no innate numerical structure so categorizing these variables with 1,2, and 3 could introduce bias and error to our model as it tries to understand why divorced might be worth more than single or is equal to single plus married. To solve this problem I used One-Hot Encoding to convert the marital column into 3 columns and the job column into 11 columns that use binary values instead of categorical numbers to group the data. I also considered and tested several versions of the model without the marital status and job categories as well as with to see their effect on the outcome and ultimately ended up leaving this data in because it appears to be relevant to the client's current situation.

At the end of this process, I was left with a total of 26 input feature variables.

4 Data Analysis

I then began to try to visualize the features included in the dataset. Below are some of the graphs of the features that I thought would have the greatest impact on the models I would build in the later phases. My initial hypothesis is that the best predictors to whether a person would open a term deposit account are going to be related to their own financial status such as if they have a loan or housing mortgage payment along with their economic class status such as their job and education level. I am curious to see though how the market indicators such as the Employment variation rate, Consumer Price index, Consumer Confidence index and Euribor the Euro Interbank Offered Rate might have an impact on the clients investment decisions.

To view all feature graphs please see the Phase 1 Jupyter Notebook. For clarity and simplicity I included the Job and Marital features in pie charts of their original values before the One-Hot encoding, that it because it is easier to see how they are distributed as one pie chart than in it is 7 pie charts.

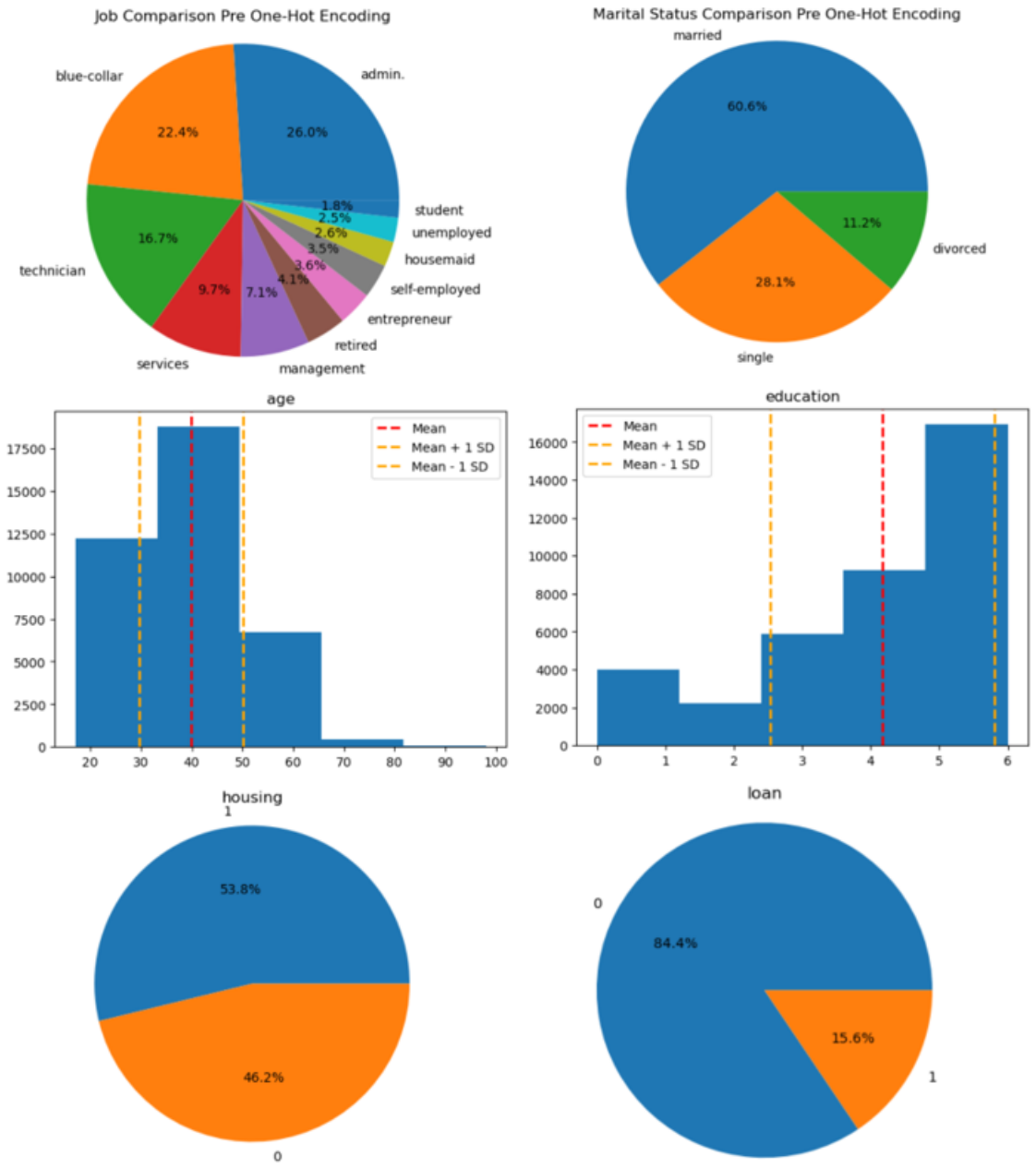


Figure 2: Collected Visualization of Feature Distributions.

5 Balancing Data

At this point, before I could begin building models I needed to see if our data was balanced, and if not it needed to be balanced. This is important because an unbalanced dataset can create bias in the model in favor of the majority class. Looking at our data, please see the graph below. There is a significant imbalance between the '0' output values and the '1' output values, this means that the clients that have not opened a term deposit account outnumber the clients that have roughly 10 to 1. If I were to train my model as is, this could cause the model to have a bias towards predicting more frequently that a client will not open a term deposit account.

Instead, to prevent this bias I re-sampled the data by down-sampling the majority class. This means that I reduced the total instances of the '0' output values to be equal to the total '1' output values. After resampling the dataset had a total of 8,516 instances and 26 features and I was ready to begin Phase 2.

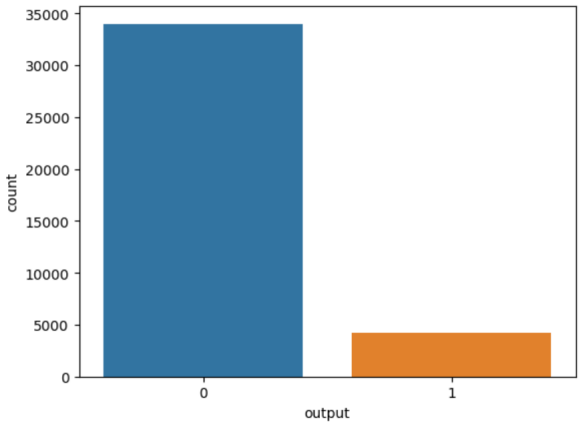


Figure 3: View the unbalanced data after cleaning.

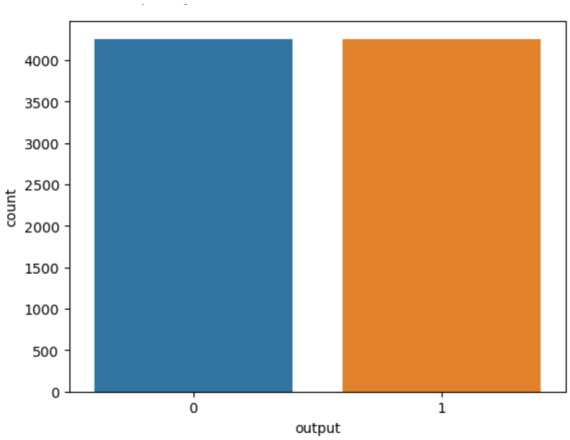


Figure 4: Resampled and Balanced Data.

6 Challenges

It was difficult to determine which features were important in this dataset and which could cause noise or bias in the final model. Looking at the heat map most of these features do not have strong correlations with the output variable so I did not want to remove too many variables. At the same time I questioned if some of the prior campaign variables like the pdays or poutcome input variables were going to improve the model or if they were going to cause noise and bias in the model. Ultimately I decided to leave them in because in Phase 4 I was going to look closer at the importance of each feature so it made sense to wait and see if these features would be features that could be removed later.

7 Final Dataset Features

Final Feature Information:

1. age - age of the client
2. job - admin.
3. job - blue-collar
4. job - entrepreneur

5. job - housemaid
6. job - management
7. job - retired
8. job - self-employed
9. job - services
10. job - student
11. job - technician
12. job - unemployed
13. marital - single
14. marital - married
15. marital - divorced
16. education Client education level
17. housing - If client has a housing loan
18. loan - If client has personal loan
19. campaign - number of contacts performed during this campaign and for this client
20. pdays - Number of days that passed by after the client was last contacted from a previous campaign; 999 means client was not previously contacted
21. previous - number of contacts performed before this campaign and for this client
22. poutcome - Outcome of the previous marketing campaign
23. emp.var.rate - Employment variation rate - quarterly indicator
24. cons.price.idx - Consumer price index - monthly indicator
25. cons.conf.idx - Consumer confidence index - monthly indicator
26. euribor3m - The Euribor 3 month rate - daily indicator
27. output

Please see the Phase 1 Jupyter Notebook for more details on the Phase 1 execution and results.

References

[You, 2018] (2018). *Machine Learning Tutorial Python - 6: Dummy Variables & One Hot Encoding.* YouTube.

[Kumar, 2021] Kumar, S. (2021). 7 ways to handle missing values in machine learning.

[Online,] Online, S. Handling categorical variables with one-hot encoding - shiksha online.

[Singh, 2023] Singh, S. (2023). Title: Label encoding and one-hot encoding for data preprocessing.

[Wohlwend, 2023] Wohlwend, B. (2023). Converting categorical data into numerical form: A practical guide for data science.