

Machine Learning Meets Term Life Insurance: Targeting High-Value Customers

CONTENTS

| | |
|--|-----------|
| CONTENTS | 2 |
| INTRODUCTION | 3 |
| CHAPTER ONE: DATA EXPLORATION AND PREPARATION | 4 |
| 1.1 DATA EXPLORATION | 4 |
| 1.2 HANDLING MISSING VALUES AND OUTLIERS..... | 10 |
| 1.3 DATA VISUALIZATION | 14 |
| CHAPTER TWO: MODEL SELECTION AND TRAINING..... | 20 |
| 2.1 MODEL SELECTION..... | 20 |
| 2.2 DATA SPLITTING | 23 |
| CHAPTER THREE: MODEL INTERPRETATION AND EVALUATION..... | 25 |
| 3.1 MODEL INTERPRETATION | 25 |
| 3.2 MODEL EVALUATION..... | 27 |
| CONCLUDING REMARKS..... | 29 |
| BIBLIOGRAPHY..... | 30 |
| LIST OF FIGURES..... | 31 |
| LIST OF TABLES | 33 |

INTRODUCTION

In today's competitive insurance industry, understanding customer behavior and optimizing marketing strategies are essential. The purpose of this report is to analyze HashSysTech Insurance data to determine the main features that affect customer conversions and consider our customer decision strategy.

Data visualization and machine learning modeling methods will help us to this purpose. In particular, it was aimed to determine important variables to optimize campaign strategies and use resources more efficiently. In light of these analyses, the report aims to reveal the necessary steps to improve future marketing campaigns.

Random Forest and Logistic Regression will be used in this analysis. Random Forest is well-suited for handling both categorical and numerical data, making it ideal for complex datasets where relationships between variables may not be linear (Breiman, 2001). Logistic Regression, on the other hand, is a simpler model that provides an interpretable way to understand how different factors impact customer decisions (Hosmer & Lemeshow, 2000). To achieve better results in model tuning methods will be applied.

In addition to the predictive power of these models, effective data preprocessing and feature engineering are critical to ensure that the models perform optimally. Data cleaning and transformation, including handling missing values, encoding categorical variables, and standardizing numerical features, will be part of the analysis.

Another essential aspect of this analysis will be model evaluation. By using performance metrics such as accuracy, precision, recall, and the F1 score, we will assess the effectiveness of the models in predicting customer conversions. This process of model refinement is crucial for translating the model's predictive power into actionable business insights.

In conclusion, the learning at the end of this analysis will definitely give enough scope to the company for targeting high-value customers, reducing costs, and enhancing conversion rates in successive campaigns.

CHAPTER ONE: Data Exploration and Preparation

1.1 Data Exploration

First of all, I started to analyze the insurance.csv file by uploading it in Python (Figure 1). Table 1 shows the first five rows of the data. When we look at it, there have been 11 columns, mostly categorical values.

Features:

- **Customer Demographics:**
 - age (numeric): Customer's age.
 - job (categorical): Customer's type of job.
 - marital (categorical): Customer's marital status.
 - educational_qual (categorical): Customer's educational qualifications.
- **Contact Details:**
 - call_type (categorical): Type of communication used for contacting the customer (e.g., phone call, email).
 - day (numeric): Day of the month (numeric) when the last contact was made.
 - mon (numeric): Month of the year (numeric) when the last contact was made.
 - dur (numeric): Duration of the last contact, in seconds.
 - num_calls (numeric): Total number of contacts made with this customer during this campaign.
- **Campaign History:**
 - prev_outcome (categorical): Outcome of the previous marketing campaign for this customer
- **Target Variable**
 - y (categorical): Indicates whether the customer subscribed to the insurance product ("yes" or "no").

```
[2] import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

df=pd.read_csv('/content/drive/MyDrive/assignments/insurance.csv')
df.head()
```

Figure 1: Uploading the insurance data

| | age | job | marital | education_qual | call_type | day | mon | dur | num_calls | prev_outcome | y |
|---|-----|--------------|---------|----------------|-----------|-----|-----|-----|-----------|--------------|----|
| 0 | 58 | management | married | tertiary | unknown | 5 | may | 261 | 1 | unknown | no |
| 1 | 44 | technician | single | secondary | unknown | 5 | may | 151 | 1 | unknown | no |
| 2 | 33 | entrepreneur | married | secondary | unknown | 5 | may | 76 | 1 | unknown | no |
| 3 | 47 | blue-collar | married | unknown | unknown | 5 | may | 92 | 1 | unknown | no |
| 4 | 33 | unknown | single | unknown | unknown | 5 | may | 198 | 1 | unknown | no |

Table 1: First 5 rows of the data

Table 2 shows that the dataset has 45211 rows and no null values.



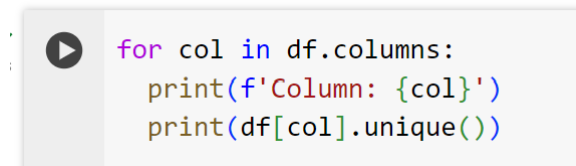
| | |
|---|------------|
|  | df.count() |
|  | 0 |
| age | 45211 |
| job | 45211 |
| marital | 45211 |
| education_qual | 45211 |
| call_type | 45211 |
| day | 45211 |
| mon | 45211 |
| dur | 45211 |
| num_calls | 45211 |
| prev_outcome | 45211 |
| y | 45211 |
| dtype: int64 | |

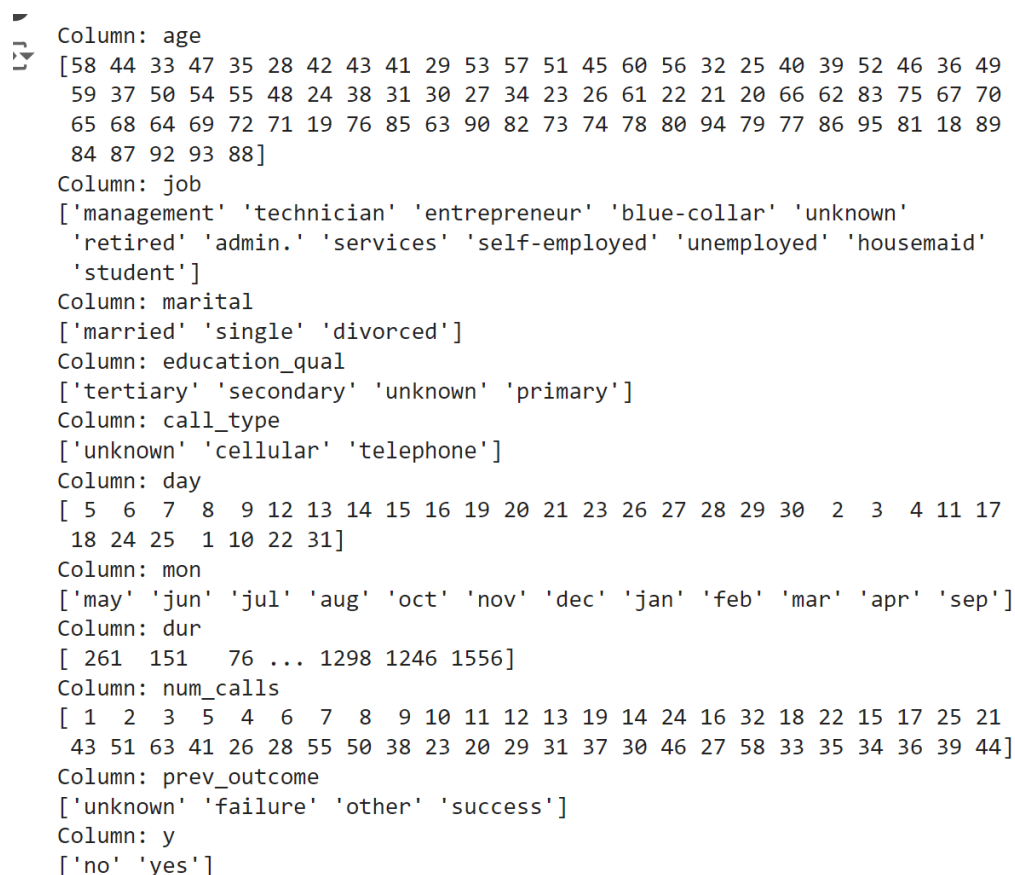
Table 2: Count of the columns

I first looked at the unique values in each column to understand the dataset better. Figure 2 shows the code part for the unique values and Figure 3 shows the values in it. The "age" column has a wide range of values, we could group it in the next step. Similarly, "duration" and "num_calls" have a broad spread. Job, "education_qual", "call_type", and "prev_outcome" include unknown values. We need to check count of the values. We can get valuable insight with the "prev_outcome" column. Besides "y" is a valuable column for prediction.



```
for col in df.columns:
    print(f'Column: {col}')
    print(df[col].unique())
```

Figure 2: Code Part For The Unique Values



```
Column: age
[58 44 33 47 35 28 42 43 41 29 53 57 51 45 60 56 32 25 40 39 52 46 36 49
 59 37 50 54 55 48 24 38 31 30 27 34 23 26 61 22 21 20 66 62 83 75 67 70
 65 68 64 69 72 71 19 76 85 63 90 82 73 74 78 80 94 79 77 86 95 81 18 89
 84 87 92 93 88]
Column: job
['management' 'technician' 'entrepreneur' 'blue-collar' 'unknown'
 'retired' 'admin.' 'services' 'self-employed' 'unemployed' 'housemaid'
 'student']
Column: marital
['married' 'single' 'divorced']
Column: education_qual
['tertiary' 'secondary' 'unknown' 'primary']
Column: call_type
['unknown' 'cellular' 'telephone']
Column: day
[ 5  6  7  8  9 12 13 14 15 16 19 20 21 23 26 27 28 29 30  2  3  4 11 17
 18 24 25  1 10 22 31]
Column: mon
['may' 'jun' 'jul' 'aug' 'oct' 'nov' 'dec' 'jan' 'feb' 'mar' 'apr' 'sep']
Column: dur
[ 261 151  76 ... 1298 1246 1556]
Column: num_calls
[ 1  2  3  5  4  6  7  8  9 10 11 12 13 19 14 24 16 32 18 22 15 17 25 21
 43 51 63 41 26 28 55 50 38 23 20 29 31 37 30 46 27 58 33 35 34 36 39 44]
Column: prev_outcome
['unknown' 'failure' 'other' 'success']
Column: y
['no' 'yes']
```

Figure 3: Distinct Values For The Dataset

Figure 4 includes code part for checking duplicate values. Six rows have been founded dropped so as not to impact the analysis in the following stages.

```
[7] #is there any duplicate values?  
df.duplicated().sum()
```

6

```
#drop duplicates  
insurance=df.copy()  
insurance.drop_duplicates(inplace=True)  
insurance.duplicated().sum()
```

0

Figure 4: Checking Duplicates

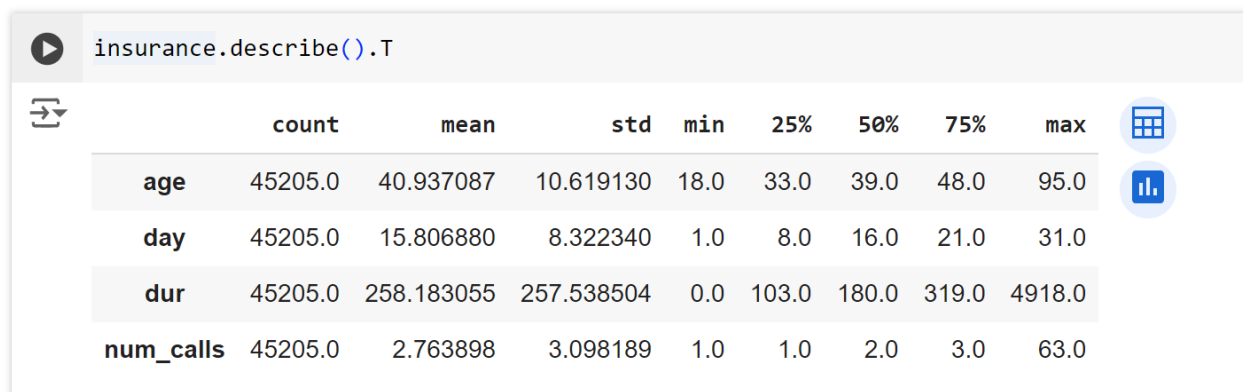
The insurance dataframe has 45205 rows after duplicates are removed, as shown in Figure 5. There are seven columns of object types and four integers. The numerical ones are "age," "day," "dur," "num_calls," and the categorical ones are "job," "marital," "education_qual," "call_type," "mon," "prev_outcome," and "y."

```
insurance.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
Index: 45205 entries, 0 to 45210  
Data columns (total 11 columns):  
#   Column             Non-Null Count  Dtype  
---  ---  
0   age                 45205 non-null  int64  
1   job                 45205 non-null  object  
2   marital             45205 non-null  object  
3   education_qual      45205 non-null  object  
4   call_type           45205 non-null  object  
5   day                 45205 non-null  int64  
6   mon                 45205 non-null  object  
7   dur                 45205 non-null  int64  
8   num_calls           45205 non-null  int64  
9   prev_outcome        45205 non-null  object  
10  y                   45205 non-null  object  
dtypes: int64(4), object(7)  
memory usage: 4.1+ MB
```

Figure 5: Information Of The Dataset

Table 3 shows basic statistics for the numerical features. The ages of the customers range from 18 to 95, with an average of around 41 and a standard deviation of 10. This suggests that dividing the ages into 10-year intervals makes sense. The day column ranges from 1 to 31, which makes sense since it represents days of the month. The values range from 0 to 4918, with an average of 258 and a standard deviation of 25. It seems we have outliers in the data because max value is far away from mean, std, 25%, 50%, 75% values. Num_calls column ranges from 1 to 63, with an average of 2.7 and a standard deviation of 3. Same as duration column, it appears that there are outliers.



```
insurance.describe().T
```

| | count | mean | std | min | 25% | 50% | 75% | max |
|-----------|---------|------------|------------|------|-------|-------|-------|--------|
| age | 45205.0 | 40.937087 | 10.619130 | 18.0 | 33.0 | 39.0 | 48.0 | 95.0 |
| day | 45205.0 | 15.806880 | 8.322340 | 1.0 | 8.0 | 16.0 | 21.0 | 31.0 |
| dur | 45205.0 | 258.183055 | 257.538504 | 0.0 | 103.0 | 180.0 | 319.0 | 4918.0 |
| num_calls | 45205.0 | 2.763898 | 3.098189 | 1.0 | 1.0 | 2.0 | 3.0 | 63.0 |

Table 3: Statistics For Numerical Features

Figure 6, visualized the frequencies of the categorical columns using bar graphs.

- **Marital Status:** More than half of the customers are married.
- **Job Distribution:** "Blue-collar" and "management" are the most common job roles in the dataset. The number of unknown values is a small percentage of the total.
- **Education Qualification:** The majority of customers have a secondary education. Assumed that the campaign is reaching individuals across various educational backgrounds.
- **Call Type:** The cellular method is used much more frequently than the telephone, likely due to the increased use of mobile phones. However, the number of unknown values are significant in the data.
- **Previous Outcome:** In the previous outcome column, most of the values are unknown, therefore we can get an insight into fewer amount of values.
- **Month Distribution:** May is the top month for calling, with a noticeable increase during the summer months (May to August). However, there is not a clear trend on it.

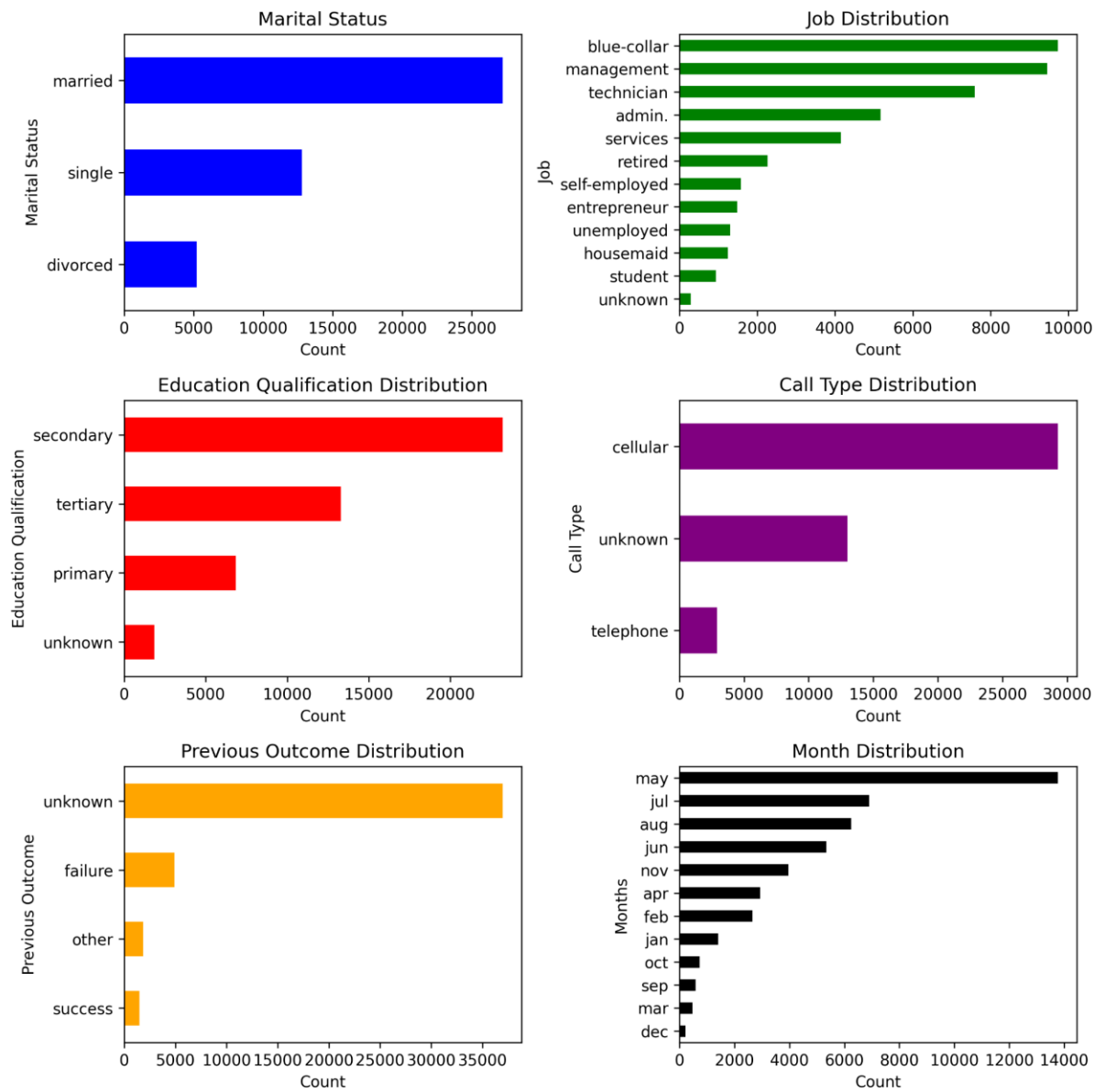


Figure 6: Frequencies For Categorical Features

1.2 Handling Missing Values and Outliers

As I mentioned, there are no missing values in the insurance data. However outliers possibility have shown in the statistics(Table 3). Figure 7 code parts were written to examine outliers by IQR method. The “age” column has 487 outliers, “dur” column has 3235 and “num_calls” has 3064 rows of outlier values (Table 4). It turns out we have a pretty significant number of outliers.

```
[27] # Checking for missing values in the dataset and not have
      #missing_values = insurance.isnull().sum()

      # Detecting outliers using the IQR method for numerical features
      Q1 = insurance[numerical_features].quantile(0.25)
      Q3 = insurance[numerical_features].quantile(0.75)
      IQR = Q3 - Q1

      # Identifying outliers
      outliers = ((insurance[numerical_features] < (Q1 - 1.5 * IQR)) | (insurance[numerical_features] > (Q3 + 1.5 * IQR))).sum()

      outliers
```

Figure 7: IQR Method Code Part

| | 0 |
|--------------|------|
| age | 487 |
| day | 0 |
| dur | 3235 |
| num_calls | 3064 |
| dtype: int64 | |

Table 4: Count Of Outliers

Boxplot can be useful for visualizing the outliers. Figure 8 is a clear visualization for identifying outliers for the age column.

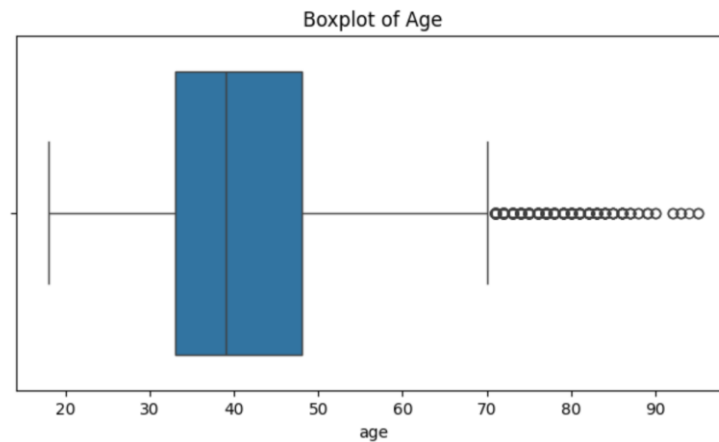


Figure 8: Boxplot Of Age

Duration ("dur" column) has many outlier values. Deleting all these values may affect the results.

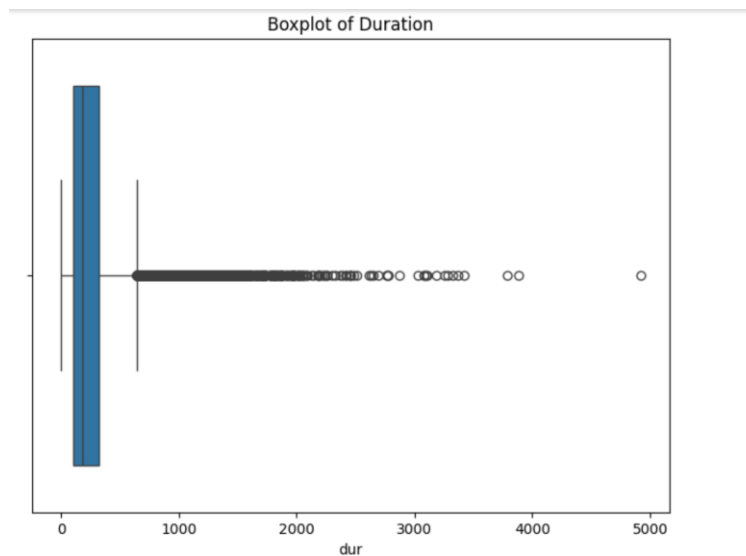


Figure 9: Boxplot Of Duration

IQR (Interquartile Range) method is chosen for handling outliers. Figure 10 includes the code part. If the outliers are handled improperly, outliers can have an exaggerated effect on statistical

conclusions and result in incorrect conclusions. Because it reduces the impact of extreme values that could otherwise shift the results, the IQR method is especially effective because it concentrates on the central distribution of the data and offers a reliable method for identifying outliers (Aguinis et al., 2013).

It may be an ideal choice when we aim to maintain most of the data while eliminating values that could affect the analysis. As a result, the IQR method is a practical and reliable choice for data cleaning in our analysis.

```
# applying the IQR method
|
Q1 = insurance[numerical_features].quantile(0.25)
Q3 = insurance[numerical_features].quantile(0.75)
IQR = Q3 - Q1

# Define outlier bounds for capping
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

insurance['age'] = insurance['age'].clip(lower=lower_bound['age'], upper=upper_bound['age'])
insurance['dur'] = insurance['dur'].clip(lower=lower_bound['dur'], upper=upper_bound['dur'])
insurance['num_calls'] = insurance['num_calls'].clip(lower=lower_bound['num_calls'], upper=upper_bound['num_calls'])

insurance.describe().T
```

Figure 10: Handling Outliers

Statistics after the IQR method are shown in Figure 11. The maximum values were modified. The maximum value for the “dur” column changed from 4918 to 643, the maximum value for the age column changed from 95 to 70.5, and the maximum value for num calls changed from 63 to 6. For the “dur” column, the mean and standard deviation were slightly different.

| | count | mean | std | min | 25% | 50% | 75% | max |
|-----------|---------|------------|------------|------|-------|-------|-------|-------|
| age | 45205.0 | 40.869052 | 10.395247 | 18.0 | 33.0 | 39.0 | 48.0 | 70.5 |
| day | 45205.0 | 15.806880 | 8.322340 | 1.0 | 8.0 | 16.0 | 21.0 | 31.0 |
| dur | 45205.0 | 234.956200 | 176.754760 | 0.0 | 103.0 | 180.0 | 319.0 | 643.0 |
| num_calls | 45205.0 | 2.392235 | 1.600152 | 1.0 | 1.0 | 2.0 | 3.0 | 6.0 |

Figure 11: Statistics After The IQR Method

Figure 12 shows the distribution of duration after trimming values by IQR method. The outlier values appear to be mostly cumulatively clustered around 600.

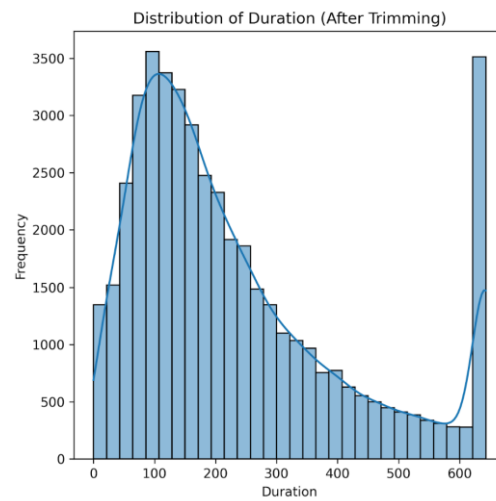


Figure 12: Distribution of Duration (After Trimming)

1.3 Data Visualization

Firstly, the “dur” column is renamed to duration. Analyzing correlation plays an important role in better understanding and visualizing data. The correlation coefficient varies between 1 and -1. Positive values indicate a positive correlation and negative values indicate negative correlation. When the matrix is examined (Figure 13):

- There are a weak relationships between age and other variables. For example, the correlation between num_calls and age is very low at 0.03.
- There is a positive correlation (0.15) between day and num_calls, but this relationship is still not considered strong. But it is the highest one.
- There is also no significant relationship between duration and other variables.

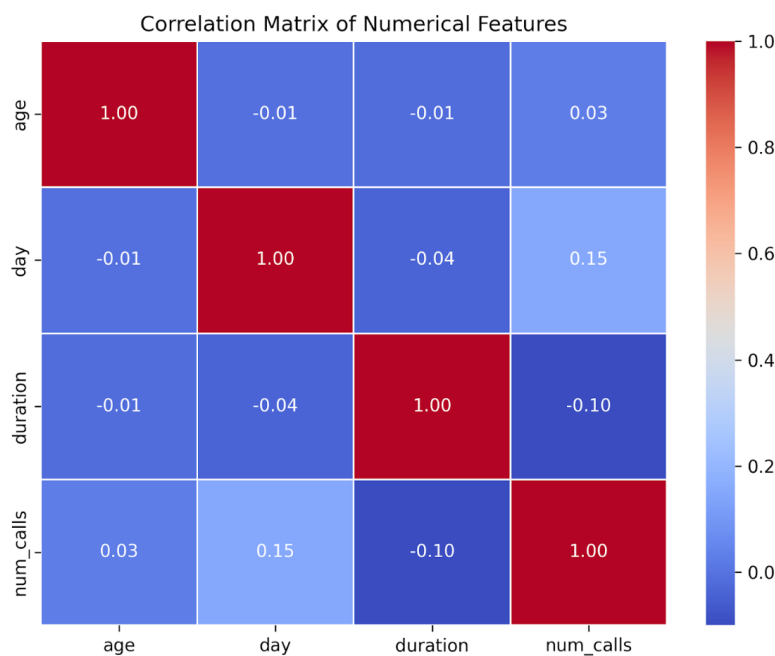


Figure 13: Heatmap Of The Correlation Matrix

For the numerical features, Figure 14 shows both density plots and scatterplots. The clients who have previously had insurance are represented by the y column. If there is a correlation between the age, duration, or number of calls and the y column, the graph helps in understanding the pattern.

Remarkably, only a small portion of clients are insured, and the majority are under 40. Regarding duration, there is a definite pattern which is clients who were insured were those who stayed on the call longer. Furthermore, the majority of customers received one to three calls; in this lower call range, there was a slight increase in positive responses.

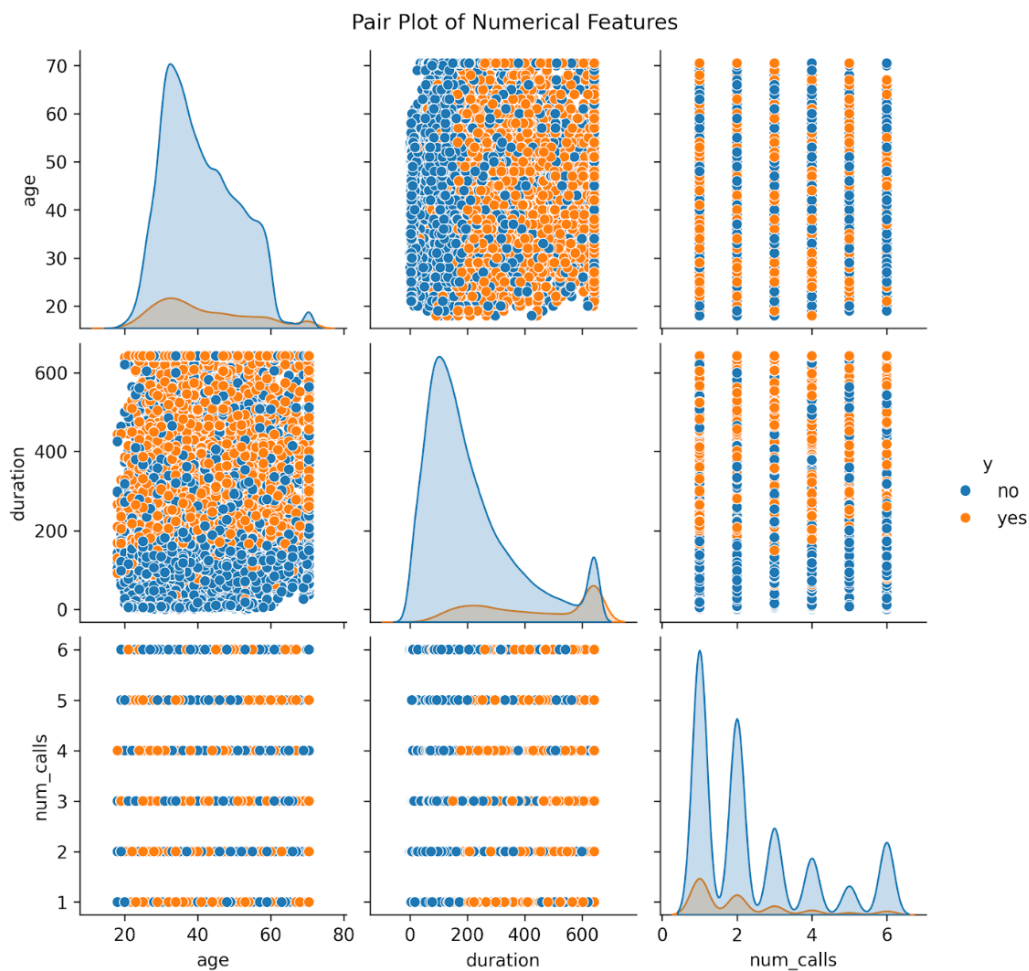


Figure 14: Pairplot Of The Numerical Values

Figure 15 presents the distribution of duration by conversion status which represents the “y” column. We analyzed the outliers cumulated after 600 seconds. Moreover, the frequency is high between 75 and 150 seconds. Customers who have insured percentage is low and is increasing a bit while the duration of seconds increasing. However, it is changing after 300 seconds.

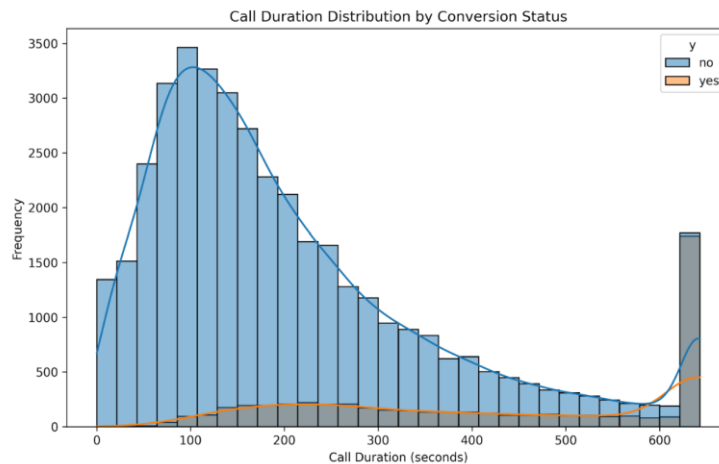


Figure 15: Distribution Of Duration By Conversion Status

Customers’s age is cumulated between 30 and 50 as shown in Figure 16. When we analyzed the age column by the “y” column, customers who have insurance products are younger than those who do not because of the mean value. However, the overall range of ages is quite similar between the two groups.

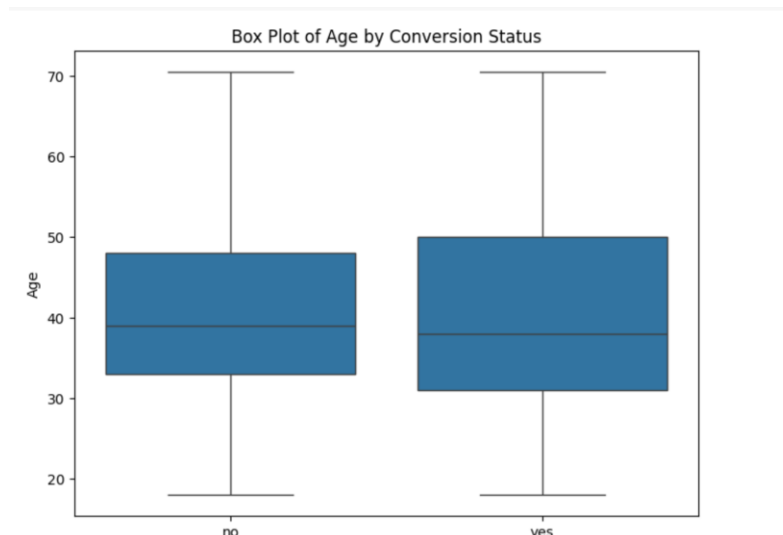


Figure 16: Box Plot Of Age By Conversion Status

The heatmap in Figure 17 shows the conversion rate by call type and previous outcome. Customers who had a successful outcome in the previous campaign ($\text{prev_outcome} = \text{success}$) have significantly higher conversion rates in the data. The conversion rate is around 65% for both cellular and telephone calls. This suggests that if a customer was successfully insured before, there is a strong likelihood they will get insured again if contacted. The high conversion rate for customers with previous success indicates a strong positive influence of past successful experiences. Conversion Rate is calculated by the “y” column count of yes divided by the total rows.

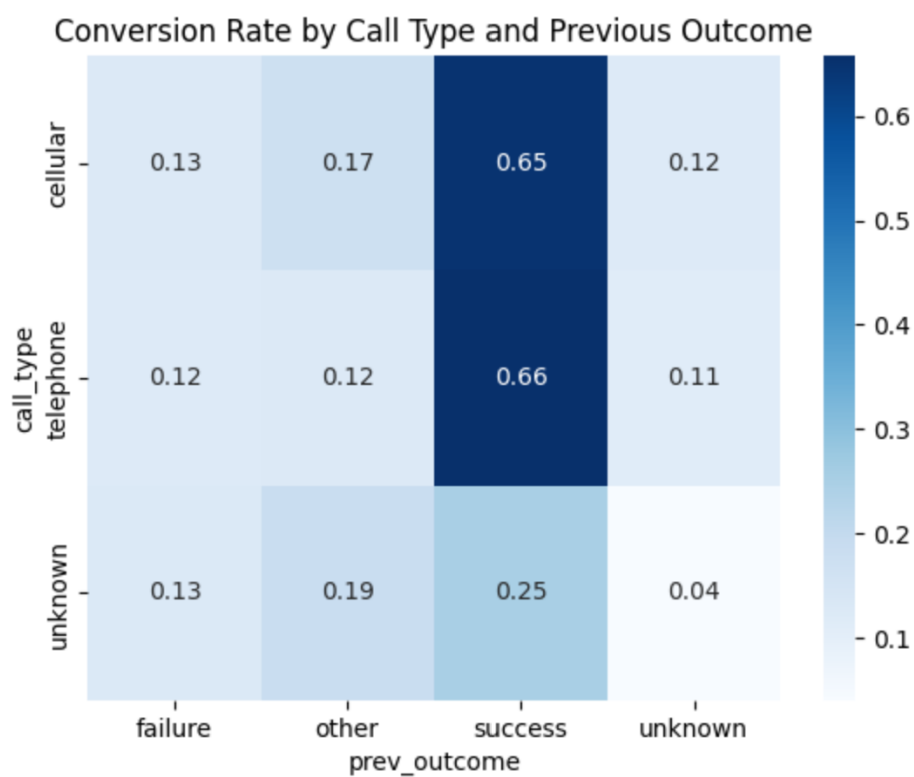


Figure 17: Conversion Rate By Call Type And Previous Outcome

The bar chart in Figure 18 shows that blue-collar and management roles are the most common job types in the dataset. However, when we analyze conversion rates in Figure 19, it becomes clear that the students and retired categories have the highest conversion rates, indicating that a larger proportion of customers from these groups successfully purchased insurance products. This shows that although management and blue-collar occupations are the most common in terms of counts, student and retired customers are converting insurance products most successfully.

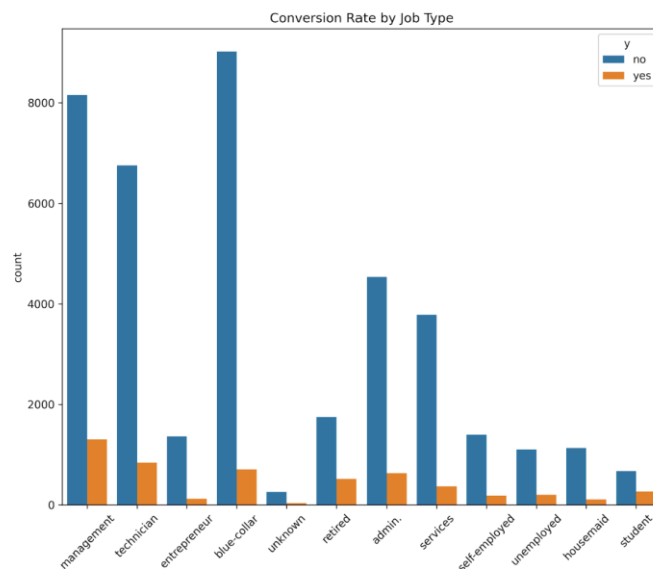


Figure 18: Counts Of Job Type

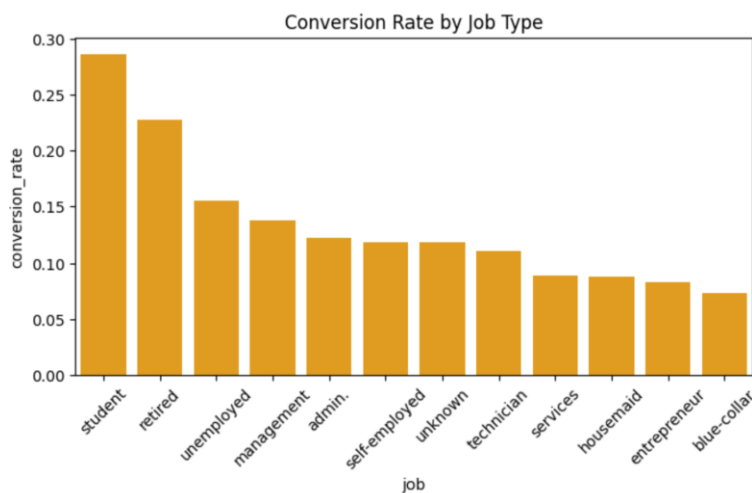


Figure 19: Conversion Rate By Job Type

When I analyzed job titles where the previous campaign outcome was successful (prev_outcome = success), I found that management roles had the highest number of conversions. However, students continue to be a key customer segment, as they also show significant conversion rates. So Figure 20 shows that both management professionals and students are important groups for future targeting.

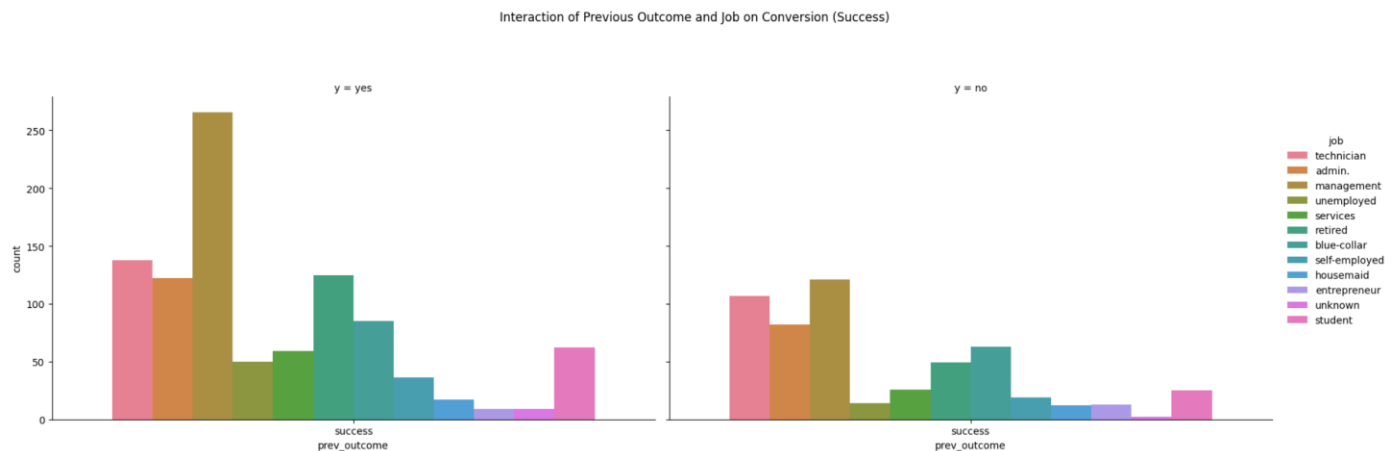


Figure 20: Interaction Of Previous Outcome And Job On Conversion

A detailed analysis was done on the columns job, call_type, age, duration, call_count and previous_output. Similarly, other features like marital_status, education_quality, month and day can also be analyzed in the same way. Throughout the analysis, various visualizations such as scatterplots, histograms, boxplots, heatmaps, and bar charts were used to explore different data features.

In conclusion, the visualizations provided valuable insights. It was clear that different types of data were best represented by certain visualizations. Therefore, it is important to choose the most appropriate chart type to correctly interpret the data and reveal underlying patterns.

CHAPTER TWO: Model Selection and Training

2.1 Model Selection

The insurance dataset has different types of data such as categorical and numerical, so choosing the right model is important. We need a model that can handle different kinds of features and capture the potentially complex relationships between them. The target variable is a type of yes no situation. So the dataset is suitable for a classic case of binary classification.

Random Forest is a strong choice for this because it works well with both categorical and numerical data. It's also great at figuring out non-linear connections, which are common in more complex datasets where things aren't as straightforward as they seem. Random Forest can automatically choose the most important features and adapt to different data types, therefore it is an excellent fit for mixed datasets. Moreover, it can give a high accuracy results while reducing the risk of overfitting (Breiman, 2001).

On the other hand, Logistic Regression is a simpler and more interpretable model for cases with two possible outcomes. In this situation, we have a possible customer or not. As LaValley (2008) points out, Logistic Regression does a good job of handling both continuous and categorical data, and it's particularly useful when you want to adjust for other influencing factors. This helps reduce bias and ensures a clearer understanding of the relationship between the predictors and the outcome (Hosmer & Lemeshow, 2000). While Logistic Regression assumes a linear relationship between the predictors and the outcome, it's still a good option for models that need to be easy to explain.

We can use Random Forest because of its flexibility in handling complex, non-linear relationships. Logistic Regression can be used because of its simple interpretation. And then we can compare the two models and gain deeper insights into the insurance dataset.

Data Preprocessing for Modeling

To improve the performance of the model, several preprocessing techniques were applied to the key columns. One-Hot Encoding was used for categorical variables such as 'job', 'marital', 'education_qual', 'call_type', and 'prev_outcome' (Figure 21). This method allows that categorical data can be processed by machine learning models, converting categorical variables into binary vectors (Al-Shehari & Alsowail, 2021).

The 'drop_first=True' was added, where only one category is eliminated per one-hot encoded variable, hence eliminating the disadvantage of the existing modes of that variable. This parameter ensures that the model avoids the dummy variables trap and maintains the model's statistical validity (Mohan, 2023)

The target variable 'y' was converted to binary values (0 for 'no' and 1 for 'yes') to prepare it for binary classification tasks, following standard practices in machine learning.

```
) from sklearn.preprocessing import OneHotEncoder

# data_iqr_trimmed is the DataFrame
# List of categorical variables to One-Hot Encode
categorical_vars = ['job', 'marital', 'education_qual', 'call_type', 'prev_outcome']

encoded_data = data_iqr_trimmed.iloc[:, :-2]

# Apply One-Hot Encoding using pandas get_dummies. for dummy variable trap used drop_first=true
encoded_data = pd.get_dummies(encoded_data, columns=categorical_vars, drop_first=True)

encoded_data['y']=encoded_data['y'].map({'no': 0, 'yes': 1})

# Display the first few rows of the encoded DataFrame
encoded_data.head()
```

Figure 21: Onehot Encoder

Additionally, Cyclical Encoding method was applied in Figure 22. It is used to time-related variables such as 'month' and 'day' by sin and cos transformations. This technique allows the model to capture the cyclical nature of time, which is critical for properly handling time-based data (El Moghrabi et al., 2021).

```
# Mapping month names to numbers
month_mapping = {
    'jan': 1, 'feb': 2, 'mar': 3, 'apr': 4, 'may': 5, 'jun': 6,
    'jul': 7, 'aug': 8, 'sep': 9, 'oct': 10, 'nov': 11, 'dec': 12
}

# Apply the mapping
encoded_data['mon_num'] = encoded_data['mon'].map(month_mapping)

# Cyclical encoding for 'month'
encoded_data['month_sin'] = np.sin(2 * np.pi * encoded_data['mon_num'] / 12)
encoded_data['month_cos'] = np.cos(2 * np.pi * encoded_data['mon_num'] / 12)

# Cyclical encoding for 'day'
encoded_data['day_sin'] = np.sin(2 * np.pi * encoded_data['day'] / 31)
encoded_data['day_cos'] = np.cos(2 * np.pi * encoded_data['day'] / 31)

# Optionally, drop the original 'mon' and 'mon_num' columns if no longer needed
encoded_data = encoded_data.drop(columns=['mon', 'mon_num', 'day'])

# Display the first few rows to check the results
encoded_data.head()
```

Figure 22: Cyclical Encoding

Table 5 shows data frame after applying encoding methods. We have 29 columns in the dataset and they are categorized with true-false values.

| | age | duration | num_calls | y | job_blue-collar | job_entrepreneur | job_housemaid | job_management | job_retired | job_self-employed | ... | education_qual_unknown | call_type_telephone | call_type_unknown | prev_out |
|---|------|----------|-----------|---|-----------------|------------------|---------------|----------------|-------------|-------------------|-----|------------------------|---------------------|-------------------|----------|
| 0 | 58.0 | 261 | 1 | 0 | False | False | False | True | False | False | ... | False | False | True | |
| 1 | 44.0 | 151 | 1 | 0 | False | False | False | False | False | False | ... | False | False | True | |
| 2 | 33.0 | 76 | 1 | 0 | False | True | False | False | False | False | ... | False | False | True | |
| 3 | 47.0 | 92 | 1 | 0 | True | False | False | False | False | False | ... | True | False | True | |
| 4 | 33.0 | 198 | 1 | 0 | False | False | False | False | False | False | ... | True | False | True | |

5 rows × 29 columns

Table 5: Dataframe After Encoding

2.2 Data Splitting

Splitting the data into training and testing sets is crucial for evaluating and fine-tuning the model's performance. The training set is used to fit the model, while the testing set evaluates the model's ability to generalize to unseen data (Hastie, Tibshirani, & Friedman, 2009).

The dataset will be divided into 70% training and 30% testing data, as well as 80% training and 20% testing data. These different proportions will be compared to determine which split works best for our model. Figure 23 shows the code part for 70-30 splitting. The training set helps the model learn patterns from the data, while the testing set ensures reliable performance on unseen data, avoiding overfitting (Hastie, Tibshirani & Friedman, 2009).

```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
from sklearn.preprocessing import StandardScaler

rf_data=encoded_data.copy()

# Step 1: Prepare the Data
X = rf_data.drop(columns=['y']) # Features (drop target 'y')
y = rf_data['y'] # Target variable

# Split into training and testing sets (70% training, 30% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, stratify=y, random_state=42)

# Check the counts of target variable in the full dataset, training, and test sets
y_counts_full = y.value_counts()
y_counts_train = y_train.value_counts()
y_counts_test = y_test.value_counts()

# Display the counts before and after the split
y_counts_full, y_counts_train, y_counts_test
```

Figure 23: Data Splitting Code Part

Stratified=y parameter is added to the code part to ensure the dataset split evenly. We can calculate the counts of classes 0 and 1 and their percentages in the training and test sets, to confirm whether the dataset is stratified. As shown in Table 6, the proportion of values is evenly distributed between

the training and test sets, maintaining the original class distribution. Stratification helps maintain the original class balance in both sets, leading to more accurate and fair model evaluations.

```
(y
0    39916
1     5289
Name: count, dtype: int64,
y
0    27941
1     3702
Name: count, dtype: int64,
y
0    11975
1     1587
Name: count, dtype: int64)
```

Table 6: Proportion of Values

Choosing appropriate dataset proportions and ensuring that the data is evenly distributed across both training and test sets is key to improving model accuracy. We can proceed to the modeling phase, knowing that our data preparation steps have been set. And we can reach reliable and accurate model performance.

CHAPTER THREE: Model Interpretation and Evaluation

3.1 Model Interpretation

The goal of this section is to explain how the Random Forest and Logistic Regression models make predictions, analyze which features most influence the outcome, and summarize key findings.

The prediction in the Random Forest model is an accumulation of predictions from many decision trees. Each of the trees makes a prediction based on random subsets of data input. The overall prediction then comes based on a majority vote from all the trees involved (Palczewska et al., 2013).

The feature importance method was used to analyze the importance of different features. This method reveals how each feature contributes to individual predictions, which is particularly useful for models like Random Forest, which can be hard to interpret (Palczewska et al., 2013).

The analysis showed that call duration was the most important feature, followed by age and time-related variables (Figure 24). The high value for the conversion rate of longer call duration can be elaborated from the analysis of the result in the data visualization section and the feature importance section. Also, previous campaign outcomes have presented a high propensity to predict customers who had successful past interactions converting in further campaigns.

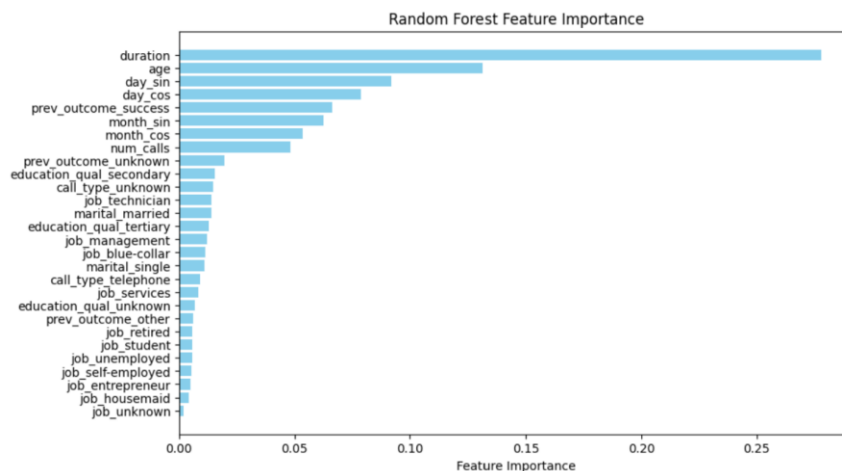


Figure 24: Feature Importance

In Logistic Regression, the model predicts the probability of a certain outcome based on the values of the features. Feature importance in Logistic Regression is evaluated by looking at the magnitude and sign of the coefficients. A positive coefficient means that higher values of the feature increase the chance of conversion. A negative coefficient means that higher values of the feature decrease the chance of conversion (James et al., 2013).

The most effective feature is prev_outcome_success with a large positive coefficient. On the other hand, call_type_unknown has a large negative impact. Job-related features such as job_student have a positive impact, while certain jobs like job_entrepreneur and job_blue-collar have a negative impact on conversion (Table 7).

The Logistic Regression model provides clear insights into which features most influence conversion outcomes. Previous campaign success stands out as the most important factor driving conversion, while features like call type and occupation also play significant roles. Understanding the effects of these features can help in targeting the right customers more effectively and improving overall campaign performance.

| Logistic Regression Coefficients: | | |
|-----------------------------------|--------------------------|-------------|
| | Feature | Coefficient |
| 22 | prev_outcome_success | 2.553305 |
| 20 | call_type_unknown | -1.122800 |
| 10 | job_student | 0.895095 |
| 17 | education_qual_tertiary | 0.532900 |
| 4 | job_entrepreneur | -0.481970 |
| 3 | job_blue-collar | -0.461368 |
| 8 | job_self-employed | -0.412679 |
| 7 | job_retired | 0.381511 |
| 18 | education_qual_unknown | 0.340023 |
| 9 | job_services | -0.336371 |
| 5 | job_housemaid | -0.326107 |
| 15 | marital_single | 0.276373 |
| 11 | job_technician | -0.239866 |
| 23 | prev_outcome_unknown | -0.218710 |
| 16 | education_qual_secondary | 0.207068 |
| 6 | job_management | -0.199481 |
| 21 | prev_outcome_other | 0.154716 |
| 12 | job_unemployed | -0.134518 |
| 13 | job_unknown | -0.126932 |
| 2 | num_calls | -0.114900 |
| 14 | marital_married | -0.111478 |
| 25 | month_cos | 0.095477 |
| 27 | day_cos | 0.094819 |
| 26 | day_sin | 0.067508 |
| 24 | month_sin | 0.013961 |
| 0 | age | 0.008608 |
| 1 | duration | 0.006704 |
| 19 | call_type_telephone | 0.002964 |

Table 7: Logistic Regression Coefficients

3.2 Model Evaluation

In this part I will try to evaluate the model and explain the results. When comparing the accuracy of random forest and logistic regression with a test size of 30% (Tables 8 and 9) the accuracy of random forest (%90) is slightly greater than that of logistic regression (%89). In the random forest model, there are also slightly better values for precision, recall, and f1 score.

```
Confusion Matrix:
[[11595  355]
 [ 993  619]]

Classification Report:
              precision    recall  f1-score   support

     0       0.92      0.97      0.95    11950
     1       0.64      0.38      0.48     1612

 accuracy      0.90      0.90      0.90    13562
 macro avg     0.78      0.68      0.71    13562
weighted avg     0.89      0.90      0.89    13562

Accuracy Score:
0.9006046305854594
```

Table 8: Random Forest Results (%30 test)

```
Confusion Matrix:
[[11610  340]
 [ 1082  530]]

Classification Report:
              precision    recall  f1-score   support

     0       0.91      0.97      0.94    11950
     1       0.61      0.33      0.43     1612

 accuracy      0.90      0.90      0.90    13562
 macro avg     0.76      0.65      0.68    13562
weighted avg     0.88      0.90      0.88    13562

Accuracy Score:
0.8951482082288748
```

Table 9: Logistic Regression Results (%30 test)

For Logistic Regression, hyperparameters such as C (regularization strength), penalty (type of regularization), and solver (optimization algorithm) were fine-tuned, while for Random Forest, hyperparameters like n_estimators (number of trees), max_depth (tree depth), max_features (features considered at each split), and criterion (splitting criterion) were adjusted. Both Grid Search and Randomized Search methods were applied to optimize the hyperparameters. I used this tuning approach to optimize the model's performance and applied recommended practices for hyperparameter tuning and model evaluation.

Based on Table 10, The Logistic Regression model performed better in terms of recall. Model identified more true positives correctly. However, F1-score is low. It means that the model had lower precision and more false positives.

In contrast, the Random Forest model performed better because of accuracy, recall, F1-score. It is a more balanced model overall. Therefore, the Random Forest model seems to be a better choice.

```
Evaluating Optimized Logistic Regression on test data...
Test Set Accuracy: 0.8131
Test Set Recall: 0.7810
Test Set F1-Score: 0.4983
```

```
Evaluating Optimized Random Forest on test data...
Test Set Accuracy: 0.8359
Test Set Recall: 0.8400
Test Set F1-Score: 0.5488
```

Table 10: Evaluated Model Results (%30 test)

A 20% test size was performed, and no significant changes were observed in the results. The Random Forest model with a 30% test size remains the best-performing model for our data. It performed well in terms of accuracy, recall, and F1-score compared to Logistic Regression.

```
Evaluating Optimized Logistic Regression on test data...
Test Set Accuracy: 0.8114
Test Set Recall: 0.7799
Test Set F1-Score: 0.4984
```

```
Evaluating Optimized Random Forest on test data...
Test Set Accuracy: 0.8332
Test Set Recall: 0.8545
Test Set F1-Score: 0.5517
```

Table 11: Evaluated Model Results (%20 test)

To sum up, when we split the data %30 for test and %70 for training and then applied tuning methods like Grid Search and Randomized Search methods for hyperparameters, Random Forest model is the best choice for the dataset.

CONCLUDING REMARKS

In this report, we thoroughly analyzed the HashSysTech Insurance dataset to identify important features that can influence customer conversions. We aimed to be able to make decisions about the future possible insurance campaign. Understanding which features are most important is the critical question.

From both the data visualization and modeling sections, it became clear that certain factors, like how long we spend on calls with potential customers, play a big role. The longer the conversation, the more likely they are to buy an insurance product. We also found that students and customers who've had positive experiences with us before are key target groups.

Other important factors include age, job type, and the number of calls, with too many calls potentially leading to customer loss. By focusing on these features, we can refine our telemarketing strategies and better allocate resources to target individuals more likely to convert. This approach will help optimize campaign success, reduce costs, and improve overall business outcomes.

In the model training process, the Random Forest model performed well. And it increased after optimizing parameters using Grid Search and Random Search methods. With a 30% test size, the model reached strong accuracy. The Random Forest model's ability to handle both categorical and numerical data proved invaluable and is helpful for our dataset.

To sum up, HashSysTech can refine its telemarketing strategies, reduce unnecessary costs, and improve campaign conversion rates. The insights derived from this analysis not only provide a clear path forward for targeted customer outreach but also allow the company to maximize the efficiency of its marketing efforts. Furthermore, the integration of these data-driven strategies will help HashSysTech stay competitive in an increasingly dynamic insurance market.

BIBLIOGRAPHY

- Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5-32.
- Hosmer, D.W. & Lemeshow, S., 2000. *Applied Logistic Regression*. 2nd ed. New York: John Wiley & Sons.
- Aguinis, H., Gottfredson, R.K., and Joo, H. (2013) 'Best-practice recommendations for defining, identifying, and handling outliers', *Organizational Research Methods*, 16(2), pp. 270-301
- LaValley, M.P., 2008. Logistic Regression. *Circulation*, 117, pp.2395-2399.
- Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed. New York: Springer, pp. 193-222.
- Muraina, I.O. (2021). Ideal Dataset Splitting Ratios in Machine Learning Algorithms: General Concerns for Data Scientists and Data Analysts. *7th International Mardin Artuklu Scientific Researches Conference*, pp. 496-504.
- Al-Shehari, T. & Alsowail, R.A. (2021). An Insider Data Leakage Detection Using One-Hot Encoding, Synthetic Minority Oversampling and Machine Learning Techniques. *Entropy*, 23(10), 1258.
- Mohanan, M. (2023). Understanding the Dummy Variable Trap in Regression Models. Preprint.
- El Moghrabi, R., Tian, R., Capretz, M., & Liboni, L. (2021). Data Preprocessing for Machine Learning Models. Presented at Western University, London, Canada.
- Palczewska, A., Palczewski, J., Marchese Robinson, R., & Neagu, D. (2013). Interpreting random forest models using a feature contribution method. *Proceedings of the IEEE 2013 International Conference on Information Reuse and Integration (IRI)*, San Francisco, USA.
- James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An Introduction to Statistical Learning: with Applications in R*. New York: Springer, pp. 138-150.

LIST OF FIGURES

| | |
|---|-----------|
| <i>Figure 1: Uploading the insurance data</i> | <i>5</i> |
| <i>Figure 2: Code Part For The Unique Values</i> | <i>6</i> |
| <i>Figure 3: Distinct Values For The Dataset</i> | <i>6</i> |
| <i>Figure 4: Checking Duplicates</i> | <i>7</i> |
| <i>Figure 5: Information Of The Dataset.....</i> | <i>7</i> |
| <i>Figure 6: Frequencies For Categorical Features</i> | <i>9</i> |
| <i>Figure 7: IQR Method Code Part.....</i> | <i>10</i> |
| <i>Figure 8: Boxplot Of Age.....</i> | <i>11</i> |
| <i>Figure 9: Boxplot Of Duration</i> | <i>11</i> |
| <i>Figure 10: Handling Outliers</i> | <i>12</i> |
| <i>Figure 11: Statistics After The IQR Method</i> | <i>12</i> |
| <i>Figure 12: Distribution of Duration (After Trimming).....</i> | <i>13</i> |
| <i>Figure 13: Heatmap Of The Correlation Matrix.....</i> | <i>14</i> |
| <i>Figure 14: Pairplot Of The Numerical Values</i> | <i>15</i> |
| <i>Figure 15: Distribution Of Duration By Conversion Status.....</i> | <i>16</i> |
| <i>Figure 16: Box Plot Of Age By Conversion Status.....</i> | <i>16</i> |
| <i>Figure 17: Conversion Rate By Call Type And Previous Outcome.....</i> | <i>17</i> |
| <i>Figure 18: Counts Of Job Type</i> | <i>18</i> |
| <i>Figure 19: Conversion Rate By Job Type.....</i> | <i>18</i> |
| <i>Figure 20: Interaction Of Previous Outcome And Job On Conversion</i> | <i>19</i> |
| <i>Figure 21: Onehot Encoder</i> | <i>21</i> |
| <i>Figure 22: Cyclical Encoding.....</i> | <i>22</i> |
| <i>Figure 23: Data Splitting Code Part</i> | <i>23</i> |

| | |
|--|----|
| <i>Figure 24: Feature Importance</i> | 25 |
|--|----|

LIST OF TABLES

| | |
|--|-----------|
| <i>Table 1: First 5 rows of the data</i> | <i>5</i> |
| <i>Table 2: Count of the columns</i> | <i>5</i> |
| <i>Table 3: Statistics For Numerical Features.....</i> | <i>8</i> |
| <i>Table 4: Count Of Outliers</i> | <i>10</i> |
| <i>Table 5: Dataframe After Encoding</i> | <i>22</i> |
| <i>Table 6: Logistic Regression Coefficients</i> | <i>26</i> |
| <i>Table 7: Random Forest Results (%30 test)</i> | <i>27</i> |
| <i>Table 8: Logistic Regression Results (%30 test)</i> | <i>27</i> |
| <i>Table 9: Evaluated Model Results (%30 test).....</i> | <i>28</i> |
| <i>Table 10: Evaluated Model Results (%20 test).....</i> | <i>28</i> |

