|  |
| --- |
| California Lutheran University |
| Predictive Modeling for Direct Marketing Optimization in the Financial Services |
| ECON 562: Analytics II  John Garcia  Spring 2025 |

|  |
| --- |
| Shaun Levenson  5-18-2025 |

**Predictive Modeling for Direct Marketing Optimization in the Banking Sector**

**Introduction**

In today’s competitive financial services environment, banks aim to enhance customer engagement and increase revenue through targeted direct marketing campaigns. Our client, a major financial institution, seeks to better understand customer behavior to drive the adoption of new products and services. Using a dataset of over one million customer records, this project develops three predictive models: a classification model to determine product adoption (B\_TGT), and two numeric models to predict total new sales (INT\_TGT) and the number of new products purchased (CNT\_TGT). Following the CRISP-DM methodology, we conduct thorough exploratory data analysis, feature engineering, and model selection using H2O’s automatic machine learning function to identify the most important predictors and most accurate predictive modeling techniques. The performance of these models on the test set will inform future marketing strategies and guide long-term consulting decisions for the client.

**Overview**

This project aims to build predictive models that help a major financial institution better understand and anticipate customer behavior in response to marketing efforts. Specifically, we focus on three target variables: B\_TGT, a binary indicator of whether a customer tried a new product; INT\_TGT, representing total new sales per customer; and CNT\_TGT, the count of new products purchased. To support both strategic planning and actionable targeting, our goal is to develop models that balance predictive accuracy with interpretability.

We use H2O’s AutoML function to efficiently train and evaluate multiple modeling techniques—including GLM, DRF, GBM, and Deep Learning—across both classification and regression tasks. By extracting the top models from each method, we ensure a comprehensive assessment of performance across different algorithm families. The resulting models are evaluated using appropriate metrics for each target variable, laying the foundation for data-driven marketing strategies and future consulting recommendations.

**Data Background**

The dataset used in this analysis comprises **over 1 million observations (1,060,038 rows)** and includes **24 variables** encompassing identifiers, target outcomes, and a wide range of customer-level predictors. It is structured to support predictive modeling across multiple outcomes related to customer purchasing behavior. Specifically, the dataset is partitioned into three subsets—training, validation, and testing—which facilitates robust model training and evaluation.

There are **three primary target variables** of interest. b\_tgt is a **binary indicator** representing whether a customer has tried a new product. int\_tgt is a **continuous numeric variable** that captures the **total new sales** made to each customer. cnt\_tgt is a **count variable** indicating the **number of new products and services** a customer has purchased. The predictors include both **categorical** and **numeric** inputs, such as customer demographics, behavioral metrics, and purchasing history.

The **numeric predictors** (18 variables) primarily consist of **RFM metrics** (Recency, Frequency, Monetary value), along with demographic features like demog\_age, demog\_homeval, demog\_inc, and demog\_pr. Categorical predictors (6 variables) include account activity level, customer value segment, gender, and homeownership status. A unique identifier (account) ensures traceability at the individual level, and a partition flag (dataset) indicates the data split used for modeling, however I ended up removing this in favor of an h2o partition. This rich and well-labeled dataset provides a comprehensive foundation for developing predictive models tailored to customer segmentation, targeting, and product uptake. Full descriptions of the variables can be found in **Appendix.a** in **Table 1** and **Table 2**.

**Data Quality Check**

As part of the initial data quality checks, I conducted a comprehensive assessment of missing values and structural integrity in the dataset. Four variables were identified with missing values: int\_tgt, cnt\_tgt, rfm3, and demog\_age. The percentage of missing values for each was relatively low and is documented in a summary table. For the variable int\_tgt, which represents total new sales, missing values were found only in cases where b\_tgt was 0—indicating the customer did not purchase a product. In this context, imputing missing int\_tgt values with 0 was a logical decision, as it clearly reflects a lack of purchase activity and avoids the unnecessary removal of valid records.

For the other variables—rfm3, cnt\_tgt, and demog\_age—I opted to remove observations with missing values. This decision was driven by the project goal of producing a highly interpretable model for the client. Introducing imputation methods for these predictors could obscure the interpretability of variable relationships, which is particularly important for stakeholders in a financial setting. Notably, only one observation had a missing value in cnt\_tgt, so its exclusion had negligible impact. Additionally, I removed any records where demog\_age was less than 16. This threshold was applied based on legal and practical considerations, as customers under 16 typically lack financial autonom. These steps ensured that the data passed structural and validity checks, laying a clean and reliable foundation for subsequent exploratory analysis. Shown in **Table 3** are the exact numbers and percentage shares of missing values.

**Table 3:** Missing Values

|  |  |  |
| --- | --- | --- |
| **Variable** | **# Missing** | **% Missing** |
| **int\_tgt** | 848529 | 80.0 |
| **demog\_age** | 266861 | 25.2 |
| **rfm3** | 225786 | 21.3 |
| **cnt\_tgt** | 1 | 0 |

As part of the data quality checks, all variables initially stored as character types were systematically converted to factors. This step ensures that categorical information is correctly treated in downstream analyses and modeling, preserving the integrity of level-based distinctions. Converting character variables to factors was essential for maintaining accurate representations of categorical data and enabling proper handling by modeling algorithms that rely on factor structures. All other important DQC factors seemed to be in order.

**Exploratory Data Analysis**

Summary Statistics

After addressing missing values, several numerical variables exhibited substantial changes in their distribution. Most notably, the range of int\_tgt decreased dramatically, with the maximum dropping from 500,000 to 200,000. This reflects the removal of extreme outliers and the imputation of 0s for non-purchasing customers, resulting in a large decrease in the mean (from 11,235.87 to 2,583.73) and skewness (from 13.30 to 4.77). Similarly, rfm3 saw a major reduction in skewness from 114.92 to 9.85, indicating a more balanced distribution after excluding rows with missing values. Changes in central tendency were also observed across several variables—for example, demog\_inc increased in mean (from 40,368.69 to 46,562.05) and median, suggesting that the removed cases tended to have lower incomes. Overall, these shifts highlight how missing data handling can significantly affect the interpretation of key variables and influence modeling outcomes.

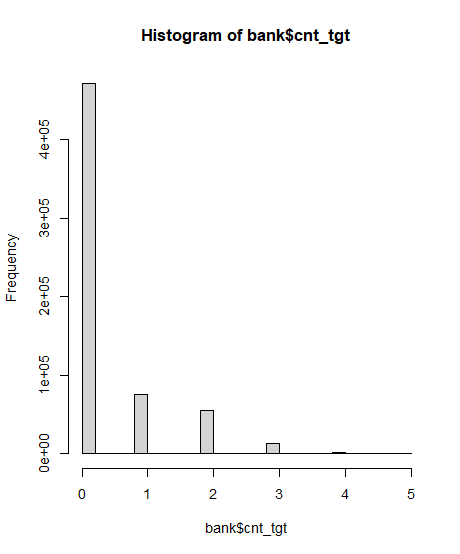
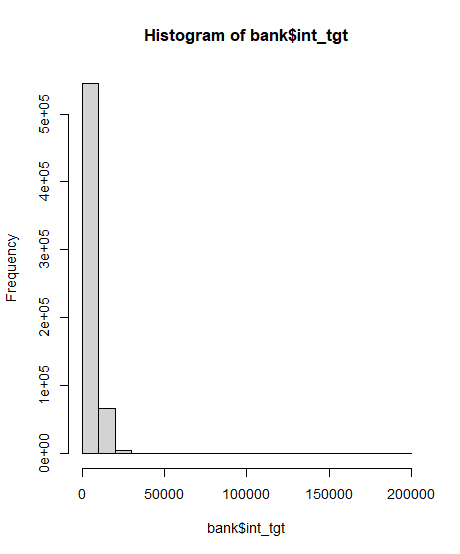
**Table 4 and 8:** Conjoined Summary Statistics – Numerical

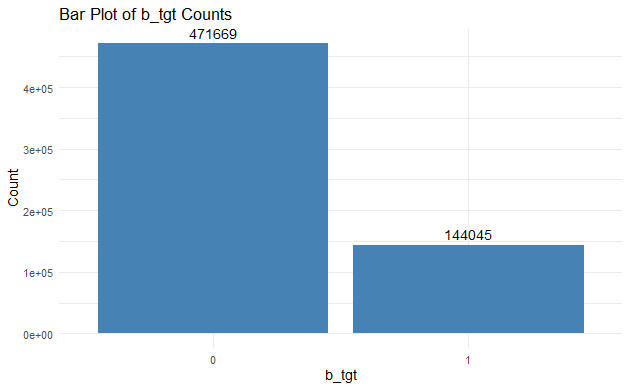
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Variable** | **Stage** | **Mean** | **Std Dev** | **Median** | **Min** | **Max** | **Skewness** |
| int\_tgt | Before | 11235.87 | 8491.80 | 10000.00 | 0.00 | 500000.00 | 13.30 |
| – | After | 2583.73 | 5931.40 | 0.00 | 0.00 | 200000.00 | 4.77 |
| cnt\_tgt | Before | 0.31 | 0.70 | 0.00 | 0.00 | 6.00 | 2.40 |
| – | After | 0.37 | 0.76 | 0.00 | 0.00 | 5.00 | 2.09 |
| rfm1 | Before | 16.09 | 19.30 | 15.00 | 0.00 | 3713.31 | 103.31 |
| – | After | 15.44 | 15.09 | 13.98 | 1.15 | 2507.50 | 84.12 |
| rfm2 | Before | 13.35 | 9.47 | 11.67 | 1.58 | 650.00 | 11.12 |
| – | After | 12.80 | 8.93 | 11.10 | 1.58 | 436.00 | 9.92 |
| rfm3 | Before | 15.31 | 18.97 | 14.00 | 0.00 | 3713.31 | 114.92 |
| – | After | 15.22 | 11.10 | 14.00 | 0.00 | 600.00 | 9.85 |
| rfm4 | Before | 17.47 | 37.55 | 15.00 | 0.00 | 10000.00 | 207.18 |
| – | After | 16.70 | 44.24 | 15.00 | 0.00 | 10000.00 | 205.74 |
| rfm5 | Before | 2.91 | 2.03 | 2.00 | 0.00 | 18.00 | 1.23 |
| – | After | 3.29 | 2.05 | 3.00 | 0.00 | 18.00 | 1.10 |
| rfm6 | Before | 9.54 | 8.47 | 7.00 | 0.00 | 127.00 | 1.91 |
| – | After | 10.34 | 8.78 | 8.00 | 0.00 | 127.00 | 1.77 |
| rfm7 | Before | 1.67 | 1.53 | 1.00 | 0.00 | 11.00 | 1.22 |
| – | After | 2.07 | 1.46 | 2.00 | 0.00 | 11.00 | 1.24 |
| rfm8 | Before | 5.03 | 4.51 | 4.00 | 0.00 | 46.00 | 1.42 |
| – | After | 5.67 | 4.61 | 4.00 | 0.00 | 36.00 | 1.31 |
| rfm9 | Before | 18.35 | 4.02 | 18.00 | 2.00 | 29.00 | -0.60 |
| – | After | 18.18 | 3.87 | 18.00 | 2.00 | 28.00 | -0.73 |
| rfm10 | Before | 12.89 | 4.61 | 12.00 | 0.00 | 77.00 | 2.86 |
| – | After | 13.07 | 4.51 | 13.00 | 0.00 | 60.00 | 2.91 |
| rfm11 | Before | 5.36 | 1.36 | 6.00 | 0.00 | 22.00 | 0.32 |
| – | After | 5.50 | 1.29 | 6.00 | 0.00 | 17.00 | 0.47 |
| rfm12 | Before | 68.13 | 37.35 | 64.00 | 0.00 | 571.00 | 0.30 |
| – | After | 69.32 | 37.36 | 65.00 | 13.00 | 571.00 | 0.30 |
| demog\_age | Before | 58.72 | 16.85 | 60.00 | -1.00 | 89.00 | -0.36 |
| – | After | 59.11 | 16.43 | 60.00 | 16.00 | 89.00 | -0.24 |
| demog\_homeval | Before | 106103.55 | 93289.97 | 73880.00 | 0.00 | 600067.00 | 2.46 |
| – | After | 106619.64 | 92456.21 | 74908.00 | 0.00 | 600067.00 | 2.50 |
| demog\_inc | Before | 40368.69 | 28029.02 | 43174.00 | 0.00 | 200007.00 | 0.23 |
| – | After | 46562.05 | 25211.91 | 46450.00 | 0.00 | 200007.00 | 0.24 |
| demog\_pr | Before | 30.57 | 11.53 | 31.00 | 0.00 | 101.00 | -0.15 |
| – | After | 31.16 | 11.24 | 31.00 | 0.00 | 100.00 | -0.09 |

Handling missing values did not significantly alter the distribution of categorical variables, with the exception of demog\_ho, which became noticeably more imbalanced following imputation.

Target Variable Analysis

This section explores the three target variables used in the analysis. B\_TGT is a binary variable indicating whether an individual tried a new product (Yes or No). INT\_TGT is a numeric variable representing the total value of new product sales made by the individual. CNT\_TGT is a count variable reflecting the number of new products purchased. Together, these variables offer complementary perspectives on customer behavior, capturing participation, quantity, and monetary value. Shown below are their respective distributions.





The three graphs illustrate the distribution of the target variables used in the analysis and provide important insights into the structure of the dataset.

The first histogram shows the distribution of **int\_tgt**, representing total new sales. The distribution is heavily right-skewed, with the vast majority of observations concentrated near zero and a small number of extreme values upwards of 200000 (shown by summary statistics). This pronounced skew suggests that a transformation, such as a log(x + 1) scaling, is appropriate to reduce the influence of outliers and improve normality for modeling purposes, which is further depicted by the skew of 4.77.

The second histogram displays **cnt\_tgt**, which counts the number of new products purchased. Similar to int\_tgt, this variable is also strongly right-skewed. Most customers purchased zero or one new product, and only a few purchased more than that. The discrete nature and skewness of this variable indicate that count-based models, such as Poisson or negative binomial regression, could be more suitable for prediction tasks involving cnt\_tgt.

The third bar chart presents the distribution of **b\_tgt**, a binary indicator of whether a customer tried a new product. The chart reveals a significant class imbalance, with roughly 80% of customers not trying a new product and only about 20% doing so. This imbalance should be taken into account when modeling, potentially using resampling techniques or model strategies that account for class weights to avoid bias toward the majority class. However, due to the clients requests of focusing on causality for marketing purposes, using strategies such as SMOTE may cause biased coefficients for regression based models.

Transformations

As part of the exploratory data analysis, I applied log transformations to several highly skewed numeric variables, specifically int\_tgt, rfm1, rfm2, rfm3, rfm4, and rfm10. The transformation used was log(x + 1), which helps to handle zero values while reducing skewness in the data distributions while maintaining a level of interpretability that transformations such as box-cox do not provide. This adjustment is critical because many machine learning algorithms perform better and produce more stable results when the input variables approximate a normal distribution. Additionally, the log transformation facilitates more meaningful interpretation of model coefficients, allowing them to be understood in terms of relative or percentage changes rather than absolute differences. Summary statistics for the transformed variables were computed and compared to their original counterparts to confirm the effect of the transformation on data distribution. Shown below in the chart space are before and after histograms of int\_tgt and rfm1 as examples, showing that performing transformations significantly improves normality.

**Chart 9:** Transformation Visualization

A graph of a log-transformed log

AI-generated content may be incorrect.

A graph of a log and a graph of a log and a graph of a log and a graph of a log and a graph of a log and a graph of a log and a graph of

AI-generated content may be incorrect.

I also put consideration towards transforming demog\_homeval, however using a log transformation for this variable flipped the skew to the other side and increased it’s magnitude, thus harming normality assumptions.

Variable Correlations

The correlation analysis reveals several strong relationships among the numerical predictors, particularly within the RFM variables. Notably, rfm1, rfm2, and rfm3 exhibit extremely high correlations, with values reaching as high as 0.96. This is intuitive given that these variables capture related measures of sales and promotional responses over different time frames, leading to considerable overlap in the information they provide. Similarly, rfm6 and rfm8 show a strong correlation of 0.89, reflecting the close connection between total lifetime purchases and those specifically driven by direct promotions. Additionally, rfm5 demonstrates a moderate to strong correlation with the target variable cnt\_tgt (0.79), underscoring its potential value as a predictor for modeling customer purchase behavior.

**Table 6:** Strong Correlations (>|0.6|)

|  |  |  |
| --- | --- | --- |
| Variable1 | Variable2 | Correlation |
| rfm3 | rfm1 | 0.96 |
| rfm2 | rfm1 | 0.91 |
| rfm8 | rfm6 | 0.89 |
| rfm3 | rfm2 | 0.88 |
| rfm7 | rfm5 | 0.80 |
| rfm5 | cnt\_tgt | 0.79 |
| rfm4 | rfm2 | 0.76 |
| rfm12 | rfm8 | 0.74 |
| rfm12 | rfm6 | 0.71 |
| rfm4 | rfm1 | 0.69 |
| rfm4 | rfm3 | 0.66 |
| rfm11 | rfm10 | 0.66 |
| rfm7 | cnt\_tgt | 0.62 |
|  |  |  |

While strong correlations among predictors such as **rfm1**, **rfm2**, and **rfm3** raised concerns about multicollinearity, I leveraged the **h2o AutoML** framework to address these challenges effectively. The AutoML function performs automatic variable selection, coefficient regularization, and dimensionality reduction during model training. This helps to mitigate multicollinearity by shrinking less important coefficients and selecting the most relevant features, thereby improving model stability and interpretability without requiring manual intervention. Although alternative approaches like Principal Component Analysis (PCA) or explicit removal of highly correlated variables are commonly used to handle multicollinearity, the AutoML approach provides a streamlined and data-driven solution that balances predictive performance with model simplicity. Additionally, leaving in these highly correlated variables may allow for more information to be extracted.

Interactions

To capture more complex, non-additive relationships between key demographic variables, I created three interaction terms. The interaction between income and home value (demog\_inc\_homeval) aims to reflect combined economic capacity, where high income and high home value together may signal greater financial potential than either alone. Similarly, the age and home value interaction (demog\_age\_homeval) may capture generational differences in wealth accumulation and property ownership—older individuals with higher-value homes might exhibit different spending behavior than younger individuals with similar assets. Finally, the interaction between age and income (demog\_age\_inc) considers life stage effects; for instance, younger individuals earning a high income may behave differently from older individuals with the same income. These interactions help enrich the feature set by introducing potentially meaningful nonlinear relationships that could enhance model performance.

**Model Fitting**

As previously mentioned several times, I took full advantage of h2o’s automatic machine learning algorithms to smooth modelling efforts and optimize work time. AutoML automates the supervised machine learning model training process by finding the best model, given a training dataframe and a response, which then returns a leaderboard of the best performing models that were trained in the process. The models were ranked by their corresponding performance metric (RMSE for int\_tgt and cnt\_tgt, AUC for b\_tgt). The AutoML function has access to the following algorithms: DRF (Distributed Random Forest, an ensemble learning algorithm that builds multiple decision trees in parallel for robust, scalable predictions), GLM (Generalized Linear Model, which allows for the modelling of continuous, binary, and count response variables), XGBoost (fast, scalable machine learning algorithm based on gradient-boosted decision trees), GBM (Gradient Boosting Machine, an ensemble method that builds trees sequentially, optimizing performance by correcting errors from previous trees), Deep Learning (A form of neural network), and Stacked Ensemble (which stacks the performances of many different models to account for each algorithms individual issues). The models that I considered specifically for each were DRF, GBM, Deep Learning, and GLM to get a wide range of predictability and interpretability.

For consistency and to maximize the information extracted from the data, I used the same set of variables across all models. The only exceptions were perfectly multicollinear variables—such as **demog\_genf** and **demog\_genm**, where I retained only **demog\_genf**—and identifiers like **account**, which do not contribute meaningful predictive value.

**Model Comparison and Final Evaluation**

Modelling – int\_tgt

The target metric used for **int\_tgt** was RMSE. As shown in **Table 10**, DRF and GBM both vastly outperformed compared to Deep Learning and GLM, with DRF in this case taking rank 1 with an RMSE of 1.4412 (lower RMSE is better) after completing validation. As shown in **Table 11**, running that model on the holdout test set returns highly comparable results, indicating that this model is very generalizable and runs well on unseen data.

**Table 10:** Int\_tgt Validation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model Rank** | **Type** | **RMSE** | **MSE** | **Unlogged\_RMSE** | **r2** |
| Model 1 | drf | 1.4412 | 2.0770 | 2.64 | 0.8599 |
| Model 2 | gbm | 1.7095 | 2.9224 | 4.19 | 0.8029 |
| Model 3 | deeplearning | 3.1637 | 10.0093 | 149.10 | 0.3251 |
| Model 4 | glm | 3.3184 | 11.0119 | 246.15 | 0.2575 |

**Table 11:** Best Model

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model Rank** | **Type** | **RMSE** | **MSE** | **Unlogged\_RMSE** | **r2** |
| Final Model (Model 1) | drf | 1.4457 | 2.0902 | 2.66 | 0.8585 |

Modelling – b\_tgt

The target metric used for **b\_tgt** was AUC (Area Under Curve). As shown in **Table 13**, DRF and GBM both vastly outperformed compared to Deep Learning and GLM once again, with GBM in this case taking rank 1 with an exceptional AUC of 0.9970 (higher AUC is better) after completing validation. As shown in **Table 14**, running that model on the holdout test set returns highly comparable results, indicating that this model is very generalizable and runs well on unseen data. A visualization of the model performance can be seen in **Appendix.a** in **Chart 6.**

**Table 13:** b\_tgt Validation

|  |  |  |
| --- | --- | --- |
| **Model Rank** | **Type** | **AUC** |
| Model 1 | gbm | 0.9970 |
| Model 2 | drf | 0.9961 |
| Model 3 | deeplearning | 0.8657 |
| Model 4 | glm | 0.8521 |

**Table 14:** Best Model b\_tgt

|  |  |  |
| --- | --- | --- |
| **Model Rank** | **Type** | **AUC** |
| Best Model (Model 1) | gbm | 0.9965 |

Modelling – cnt\_tgt

The target metric used for **cnt\_tgt** was RMSE. As shown in **Table 17**, DRF and GBM both vastly outperformed compared to Deep Learning and GLM, with DRF in this case taking rank 1 with an RMSE of 0.2574 (lower RMSE is better) after completing validation. As shown in **Table 18**, running that model on the holdout test set returns highly comparable results, indicating that this model is very generalizable and runs well on unseen data.

**Table 17:** cnt\_tgt Validation

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Type** | **RMSE** | **MSE** |
| Model 1 | drf | 0.2574 | 0.0663 |
| Model 2 | gbm | 0.2754 | 0.0758 |
| Model 3 | deeplearning | 0.5745 | 0.3301 |
| Model 4 | glm | 0.6165 | 0.3800 |

**Table 18:** Best Model

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Type** | **RMSE** | **MSE** |
| Best Model (Test Set) | drf | 0.2562 | 0.0656 |

**Causal Factors**

The drawback of using the models shown above is that picking out factor effects can be a little bit tricky. However, we can leverage Variable Importance Plots as well as the signs on the coefficients for the GLM model to incorporate a broad, robust interpretation of the variables.

Looking at **Charts 3, 5,** and **7** provides relative importance for each variable in the models shown above.

**Chart 3:** Variable Importance for Int\_tgt

A graph of a number of variable

AI-generated content may be incorrect.

**Chart 5:** Variable Importance for b\_tgt

A graph of a number of variable importance

AI-generated content may be incorrect.

**Chart 7:** Variable Importance cnt\_tgt

A graph of a graph

AI-generated content may be incorrect.

Notice that we consistently see variables such as rfm1, rfm2, and demog\_homeval as the variables with the most impact in the models. This is absolutely to be expected as rfm1 and 2 are variables regarding average sales (past 3 years and lifetime respectively) and demog\_homeval (home value) can be used as a proxy for wealth, where logically an individual with greater wealth is more likely to spend more on financial products/services. Interestingly, the coefficients for rfm1 across all response variables is negative, indicating that as average sales increase from previous years, total new sales, trying new products, and buying more of them decrease. This is likely due to market saturation, in which these customers have already spent a large amount on products in the past, thus decreasing the likelihood of purchasing more in the future. This can be useful for marketing purposes as marketing towards those who have not yet spent large quantities on products may be more willing to try them out if they were made aware of the new products.

This is in contrast to rfm5 (count purchased past 3 years), which had a consistent positive impact across responses. This may indicate that, unlike average sales, these customers are consistent and loyal, likely indicating high repeats, making them great candidates for marketing. These individuals would likely be more responsive to direct promotions and great deals.

When considering marketing campaigns, you should also take into consideration what sort of individuals you should avoid marketing towards, so as to not waste resources. Rfm9 (months since last purchase), is an incredibly important factor to consider for marketing. It has a large negative coefficient and is highly impactful across models indicating that as more months between product purchases go by, the less likely the individual is to spend, so marketing resources should not be wasted on individuals with a high rfm9.

**Conclusion**

The predictive modeling framework developed for B\_TGT, INT\_TGT, and CNT\_TGT has successfully enabled a comprehensive understanding of customer behavior across multiple dimensions. By leveraging H2O AutoML and rigorous evaluation metrics, we’ve identified high-probability product adopters, estimated potential sales value with meaningful accuracy, and predicted the number of products customers are likely to adopt. Additionally, we leveraged more interpretable models alongside variable importance charts to generate actionable strategies for more effective targeting, personalized marketing strategies, and revenue forecasting. As a result, this modeling initiative not only enhances decision-making but also empowers the business to allocate resources more efficiently and drive measurable growth.

**Appendix.a**

Part 1: EDA

**Table 1:** Data Understanding

|  |  |  |
| --- | --- | --- |
| **Data Type** | **Description** | **Columns** |
| Numeric | Continuous variables or discrete numeric inputs. | 18 Columns: int\_tgt, cnt\_tgt, rfm1-12, demog\_age, demog\_homeval, demog\_inc, demog\_pr |
| Factor | Categorical with fixed levels | 6 Columns: b\_tgt, cat\_input1, cat\_input2, demog\_ho, demog\_genf, demog\_genm |
| Identifier | Unique identifiers for each row | 1 Column: account |
| Partition | Subset of the dataset | 1 Column: dataset |

**Table 2:** Variable Descriptions

|  |  |  |
| --- | --- | --- |
| **Variable** | **Type** | **Description** |
| **Target Variables** |  |  |
| B\_TGT | Binary | Tried a New Product (Yes/No) |
| INT\_TGT | Numeric | Total New Sales |
| CNT\_TGT | Count | Count of New Products Purchased |
| **Categorical Predictors** |  |  |
| CAT\_INPUT1 | Categorical | Account Activity |
| CAT\_INPUT2 | Categorical | Customer Value Level |
| **RFM Interval Inputs** |  |  |
| RFM1 | Numeric | Average Sales Past 3 Years |
| RFM2 | Numeric | Average Sales Lifetime |
| RFM3 | Numeric | Avg Sales Past 3 Years - Direct Promo Response |
| RFM4 | Numeric | Last Product Purchase Amount |
| RFM5 | Numeric | Count Purchased Past 3 Years |
| RFM6 | Numeric | Count Purchased Lifetime |
| RFM7 | Numeric | Count Purchased Past 3 Years - Direct Promo Response |
| RFM8 | Numeric | Count Purchased Lifetime - Direct Promo Response |
| RFM9 | Numeric | Months Since Last Purchase |
| RFM10 | Numeric | Count Total Promos Past Year |
| RFM11 | Numeric | Count Direct Promos Past Year |
| RFM12 | Numeric | Customer Tenure |
| **Demographic Inputs** |  |  |
| DEMOG\_AGE | Numeric | Customer Age |
| DEMOG\_GENF | Binary | Female (Yes/No) |
| DEMOG\_GENM | Binary | Male (Yes/No) |
| DEMOG\_HO | Binary | Homeowner (Yes/No) |
| DEMOG\_HOMEVAL | Numeric | Home Value |
| DEMOG\_INC | Numeric | Income |
| DEMOG\_PR | Numeric | Geographical Retirement Percentage |

**Table 3:** Missing Values

|  |  |  |
| --- | --- | --- |
| **Variable** | **# Missing** | **% Missing** |
| **int\_tgt** | 848529 | 80.0 |
| **demog\_age** | 266861 | 25.2 |
| **rfm3** | 225786 | 21.3 |
| **cnt\_tgt** | 1 | 0 |

**Table 4:** Summary Statistics – Numerical

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Variable** | **Mean** | **Standard Deviation** | **Median** | **Min** | **Max** | **Skewness** |
| int\_tgt (>0) | 11235.87 | 8491.80 | 10000.00 | 0.00 | 500000.00 | 13.30 |
| cnt\_tgt | 0.31 | 0.70 | 0.00 | 0.00 | 6.00 | 2.40 |
| rfm1 | 16.09 | 19.30 | 15.00 | 0.00 | 3713.31 | 103.31 |
| rfm2 | 13.35 | 9.47 | 11.67 | 1.58 | 650.00 | 11.12 |
| rfm3 | 15.31 | 18.97 | 14.00 | 0.00 | 3713.31 | 114.92 |
| rfm4 | 17.47 | 37.55 | 15.00 | 0.00 | 10000.00 | 207.18 |
| rfm5 | 2.91 | 2.03 | 2.00 | 0.00 | 18.00 | 1.23 |
| rfm6 | 9.54 | 8.47 | 7.00 | 0.00 | 127.00 | 1.91 |
| rfm7 | 1.67 | 1.53 | 1.00 | 0.00 | 11.00 | 1.22 |
| rfm8 | 5.03 | 4.51 | 4.00 | 0.00 | 46.00 | 1.42 |
| rfm9 | 18.35 | 4.02 | 18.00 | 2.00 | 29.00 | -0.60 |
| rfm10 | 12.89 | 4.61 | 12.00 | 0.00 | 77.00 | 2.86 |
| rfm11 | 5.36 | 1.36 | 6.00 | 0.00 | 22.00 | 0.32 |
| rfm12 | 68.13 | 37.35 | 64.00 | 0.00 | 571.00 | 0.30 |
| demog\_age | 58.72 | 16.85 | 60.00 | -1.00 | 89.00 | -0.36 |
| demog\_homeval | 106103.55 | 93289.97 | 73880.00 | 0.00 | 600067.00 | 2.46 |
| demog\_inc | 40368.69 | 28029.02 | 43174.00 | 0.00 | 200007.00 | 0.23 |
| demog\_pr | 30.57 | 11.53 | 31.00 | 0.00 | 101.00 | -0.15 |

**Table 5:** Summary Statistics – Categorical/Binary

|  |  |  |  |
| --- | --- | --- | --- |
| **Variable** | **Level** | **Count** | **Percent** |
| b\_tgt | 0 | 848529 | 80.05 |
|  | 1 | 211509 | 19.95 |
| cat\_input1 | X | 831371 | 78.43 |
|  | Y | 77847 | 7.34 |
|  | Z | 150820 | 14.23 |
| cat\_input2 | A | 188398 | 17.77 |
|  | B | 192382 | 18.15 |
|  | C | 169550 | 15.99 |
|  | D | 122282 | 11.54 |
|  | E | 387426 | 36.55 |
| demog\_ho | 0 | 476741 | 44.97 |
|  | 1 | 583297 | 55.03 |
| demog\_genf | 0 | 464288 | 43.80 |
|  | 1 | 595750 | 56.20 |

**Charts:** Distributions and Counts

A group of graphs showing different distribution of int tgt

AI-generated content may be incorrect.

A group of green and white graphs

AI-generated content may be incorrect.

A group of graphs showing distribution of frequency

AI-generated content may be incorrect.

A graph of distribution of frequency

AI-generated content may be incorrect.

A comparison of a distribution of the demog

AI-generated content may be incorrect.

A group of gray bars with numbers

AI-generated content may be incorrect.

A comparison of a number of bars

AI-generated content may be incorrect.

**Chart 2:** Correlation Heatmap

A graph of heatmap of numerical variable

AI-generated content may be incorrect.

**Table 6:** Strong Correlations (>|0.6|)

|  |  |  |
| --- | --- | --- |
| Variable1 | Variable2 | Correlation |
| rfm3 | rfm1 | 0.96 |
| rfm2 | rfm1 | 0.91 |
| rfm8 | rfm6 | 0.89 |
| rfm3 | rfm2 | 0.88 |
| rfm7 | rfm5 | 0.80 |
| rfm5 | cnt\_tgt | 0.79 |
| rfm4 | rfm2 | 0.76 |
| rfm12 | rfm8 | 0.74 |
| rfm12 | rfm6 | 0.71 |
| rfm4 | rfm1 | 0.69 |
| rfm4 | rfm3 | 0.66 |
| rfm11 | rfm10 | 0.66 |
| rfm7 | cnt\_tgt | 0.62 |

**Table 7:** Correlation Analysis

|  |  |  |
| --- | --- | --- |
| Correlation (Abs. Value) | Strength | Suggestion |
| 0.0–0.3 | Weak | Usually not worth flagging |
| 0.3–0.5 | Moderate | Context-dependent |
| 0.5–0.7 | Strong-ish | Often worth a closer look |
| 0.7–0.9+ | Very strong | Good candidates for multicollinearity check |

Part 2: Modelling

**Table 8:** Post Data Processing

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Variable** | **mean** | **sd** | **median** | **min** | **max** | **skewness** |
| int\_tgt | 2583.73 | 5931.40 | 0.00 | 0.00 | 200000.0 | 4.77 |
| cnt\_tgt | 0.37 | 0.76 | 0.00 | 0.00 | 5.0 | 2.09 |
| rfm1 | 15.44 | 15.09 | 13.98 | 1.15 | 2507.5 | 84.12 |
| rfm2 | 12.80 | 8.93 | 11.10 | 1.58 | 436.0 | 9.92 |
| rfm3 | 15.22 | 11.10 | 14.00 | 0.00 | 600.0 | 9.85 |
| rfm4 | 16.70 | 44.24 | 15.00 | 0.00 | 10000.0 | 205.74 |
| rfm5 | 3.29 | 2.05 | 3.00 | 0.00 | 18.0 | 1.10 |
| rfm6 | 10.34 | 8.78 | 8.00 | 0.00 | 127.0 | 1.77 |
| rfm7 | 2.07 | 1.46 | 2.00 | 0.00 | 11.0 | 1.24 |
| rfm8 | 5.67 | 4.61 | 4.00 | 0.00 | 36.0 | 1.31 |
| rfm9 | 18.18 | 3.87 | 18.00 | 2.00 | 28.0 | -0.73 |
| rfm10 | 13.07 | 4.51 | 13.00 | 0.00 | 60.0 | 2.91 |
| rfm11 | 5.50 | 1.29 | 6.00 | 0.00 | 17.0 | 0.47 |
| rfm12 | 69.32 | 37.36 | 65.00 | 13.00 | 571.0 | 0.30 |
| demog\_age | 59.11 | 16.43 | 60.00 | 16.00 | 89.0 | -0.24 |
| demog\_homeval | 106619.64 | 92456.21 | 74908.00 | 0.00 | 600067.0 | 2.50 |
| demog\_inc | 46562.05 | 25211.91 | 46450.00 | 0.00 | 200007.0 | 0.24 |
| demog\_pr | 31.16 | 11.24 | 31.00 | 0.00 | 100.0 | -0.09 |

**Table 9:** Logged Variable Statistics

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Variable** | **mean** | **sd** | **median** | **min** | **max** | **skewness** |
| log\_int\_tgt | 2.09 | 3.84 | 0.00 | 0.00 | 12.21 | 1.31 |
| log\_rfm1 | 2.67 | 0.48 | 2.71 | 0.77 | 7.83 | 0.25 |
| log\_rfm2 | 2.51 | 0.47 | 2.49 | 0.95 | 6.08 | 0.36 |
| log\_rfm3 | 2.65 | 0.50 | 2.71 | 0.00 | 6.40 | 0.16 |
| log\_rfm4 | 2.71 | 0.54 | 2.77 | 0.00 | 9.21 | -0.30 |
| log\_rfm10 | 2.61 | 0.26 | 2.64 | 0.00 | 4.11 | 0.97 |

**Table 10:** Int\_tgt Validation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model Rank** | **Type** | **RMSE** | **MSE** | **Unlogged\_RMSE** | **r2** |
| Model 1 | drf | 1.4412 | 2.0770 | 2.64 | 0.8599 |
| Model 2 | gbm | 1.7095 | 2.9224 | 4.19 | 0.8029 |
| Model 3 | deeplearning | 3.1637 | 10.0093 | 149.10 | 0.3251 |
| Model 4 | glm | 3.3184 | 11.0119 | 246.15 | 0.2575 |

**Table 11:** Best Model

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model Rank** | **Type** | **RMSE** | **MSE** | **Unlogged\_RMSE** | **r2** |
| Final Model (Model 1) | drf | 1.4457 | 2.0902 | 2.66 | 0.8585 |

**Chart 3:** Variable Importance for Int\_tgt

A graph of a number of variable

AI-generated content may be incorrect.

**Chart 4:** Predicted vs Actual Plot

A graph with a red line and blue dots

AI-generated content may be incorrect.

**Table 12:** int\_tgt GLM Coeffs

|  |  |  |
| --- | --- | --- |
| Predictor | Coefficient | Odds Ratio (%) |
| Intercept | 9.1656809 | 956223.08 |
| cat\_input2.A | 0.1102875 | 11.66 |
| cat\_input2.B | 0.0886604 | 9.27 |
| cat\_input2.C | -0.0050504 | -0.50 |
| cat\_input2.D | -0.0575689 | -5.59 |
| cat\_input2.E | -0.1377800 | -12.87 |
| cat\_input1.X | 0.0404378 | 4.13 |
| cat\_input1.Y | -0.0142278 | -1.41 |
| cat\_input1.Z | -0.0243132 | -2.40 |
| demog\_genf.0 | -0.0134046 | -1.33 |
| demog\_genf.1 | 0.0153435 | 1.55 |
| demog\_age | -0.0042269 | -0.42 |
| demog\_ho | -0.0318255 | -3.13 |
| demog\_homeval | 0.0000085 | 0.00 |
| demog\_inc | 0.0000022 | 0.00 |
| demog\_pr | -0.0011127 | -0.11 |
| rfm5 | 0.4166238 | 51.68 |
| rfm6 | -0.0062728 | -0.63 |
| rfm7 | -0.1152239 | -10.88 |
| rfm8 | 0.0249737 | 2.53 |
| rfm9 | -0.1990148 | -18.05 |
| rfm11 | 0.0126836 | 1.28 |
| rfm12 | 0.0026433 | 0.26 |
| log\_rfm1 | -0.9859614 | -62.69 |
| log\_rfm2 | -1.1087890 | -67.00 |
| log\_rfm3 | 0.1029872 | 10.85 |
| log\_rfm4 | -0.0548908 | -5.34 |
| log\_rfm10 | -0.1889794 | -17.22 |
| demog\_inc\_homeval | 0.0000000 | 0.00 |
| demog\_age\_homeval | 0.0000000 | 0.00 |
| demog\_age\_inc | 0.0000000 | 0.00 |

**Table 13:** b\_tgt Validation

|  |  |  |
| --- | --- | --- |
| **Model Rank** | **Type** | **AUC** |
| Model 1 | gbm | 0.9970 |
| Model 2 | drf | 0.9961 |
| Model 3 | deeplearning | 0.8657 |
| Model 4 | glm | 0.8521 |

**Table 14:** Best Model b\_tgt

|  |  |  |
| --- | --- | --- |
| **Model Rank** | **Type** | **AUC** |
| Best Model (Model 1) | gbm | 0.9965 |

**Chart 5:** Variable Importance for b\_tgt

A graph of a number of variable importance

AI-generated content may be incorrect.

**Chart 6:** Visualization of the AUC

A graph with a line

AI-generated content may be incorrect.

**Table 15:** GLM Coefficients for b\_tgt

|  |  |  |
| --- | --- | --- |
| Predictor | Coefficient | Odds Ratio (%) |
| Intercept | 5.0998005 | 16298.92 |
| cat\_input2.A | 0.1119309 | 11.84 |
| cat\_input2.B | 0.0942768 | 9.89 |
| cat\_input2.C | -0.0037163 | -0.37 |
| cat\_input2.D | -0.0437802 | -4.28 |
| cat\_input2.E | -0.1597183 | -14.76 |
| cat\_input1.X | 0.1709740 | 18.65 |
| cat\_input1.Y | 0.1377629 | 14.77 |
| cat\_input1.Z | -0.3239737 | -27.67 |
| demog\_genf.0 | -0.0148939 | -1.48 |
| demog\_genf.1 | 0.0032531 | 0.33 |
| demog\_age | -0.0015614 | -0.16 |
| demog\_ho | -0.0298982 | -2.95 |
| demog\_homeval | 0.0000078 | 0.00 |
| demog\_inc | 0.0000041 | 0.00 |
| demog\_pr | 0.0006671 | 0.07 |
| rfm5 | 0.2830672 | 32.72 |
| rfm6 | -0.0106399 | -1.06 |
| rfm7 | -0.0864276 | -8.28 |
| rfm8 | 0.0154354 | 1.56 |
| rfm9 | -0.1787102 | -16.37 |
| rfm11 | 0.0672027 | 6.95 |
| rfm12 | 0.0027677 | 0.28 |
| log\_rfm1 | -0.7637720 | -53.41 |
| log\_rfm2 | -1.3508529 | -74.10 |
| log\_rfm3 | 0.0610994 | 6.30 |
| log\_rfm4 | 0.0424615 | 4.34 |
| log\_rfm10 | -0.2590137 | -22.82 |
| demog\_inc\_homeval | 0.0000000 | 0.00 |
| demog\_age\_homeval | 0.0000000 | 0.00 |
| demog\_age\_inc | 0.0000000 | 0.00 |

**Table 16:** Best Model Confusion Matrix

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **0** | **1** | **Error** | **Rate** |
| **0** | 93280 | 875 | 0.0092932 | =875/94155 |
| **1** | 1213 | 27490 | 0.0422604 | =1213/28703 |
| **Totals** | 94493 | 28365 | 0.0169952 | =2088/122858 |

**Table 17:** cnt\_tgt Validation

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Type** | **RMSE** | **MSE** |
| Model 1 | drf | 0.2574 | 0.0663 |
| Model 2 | gbm | 0.2754 | 0.0758 |
| Model 3 | deeplearning | 0.5745 | 0.3301 |
| Model 4 | glm | 0.6165 | 0.3800 |

**Table 18:** Best Model

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Type** | **RMSE** | **MSE** |
| Best Model (Test Set) | drf | 0.2562 | 0.0656 |

**Chart 7:** Variable Importance cnt\_tgt

A graph of a graph

AI-generated content may be incorrect.

**Chart 8:** Actual vs Predicted for cnt\_tgt

A graph with blue line and orange lines

AI-generated content may be incorrect.

**Table 19:**

|  |  |  |
| --- | --- | --- |
| Predictor | Coefficient | Odds Ratio (%) |
| Intercept | 1.5087728 | 352.12 |
| cat\_input2.A | 0.0181896 | 1.84 |
| cat\_input2.B | 0.0232440 | 2.35 |
| cat\_input2.C | -0.0044892 | -0.45 |
| cat\_input2.D | -0.0100478 | -1.00 |
| cat\_input2.E | -0.0272705 | -2.69 |
| cat\_input1.X | -0.0171031 | -1.70 |
| cat\_input1.Y | -0.0281818 | -2.78 |
| cat\_input1.Z | 0.0487102 | 4.99 |
| demog\_genf.0 | 0.0003223 | 0.03 |
| demog\_genf.1 | 0.0023244 | 0.23 |
| demog\_age | -0.0009596 | -0.10 |
| demog\_ho | -0.0055315 | -0.55 |
| demog\_homeval | 0.0000013 | 0.00 |
| demog\_inc | 0.0000004 | 0.00 |
| demog\_pr | -0.0003986 | -0.04 |
| rfm5 | 0.1519339 | 16.41 |
| rfm6 | 0.0021239 | 0.21 |
| rfm7 | -0.0164522 | -1.63 |
| rfm8 | 0.0029292 | 0.29 |
| rfm9 | -0.0364782 | -3.58 |
| rfm11 | -0.0255904 | -2.53 |
| rfm12 | -0.0001056 | -0.01 |
| log\_rfm1 | -0.1606913 | -14.84 |
| log\_rfm2 | -0.1890883 | -17.23 |
| log\_rfm3 | 0.0418248 | 4.27 |
| log\_rfm4 | -0.0147741 | -1.47 |
| log\_rfm10 | -0.0331659 | -3.26 |
| demog\_inc\_homeval | 0.0000000 | 0.00 |
| demog\_age\_homeval | 0.0000000 | 0.00 |
| demog\_age\_inc | 0.0000000 | 0.00 |

**Chart 9:** Transformation Visualization

A graph of a log-transformed log

AI-generated content may be incorrect.

A graph of a log-graph

AI-generated content may be incorrect.

A graph of a log and a log

AI-generated content may be incorrect.

**Table 20:** Categorical/Binary Counts after Missing Removal

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Level | Count | Percent |
| b\_tgt | 0 | 471669 | 76.61 |
| b\_tgt | 1 | 144045 | 23.39 |
| cat\_input1 | X | 498539 | 80.97 |
| cat\_input1 | Y | 37624 | 6.11 |
| cat\_input1 | Z | 79551 | 12.92 |
| cat\_input2 | A | 110689 | 17.98 |
| cat\_input2 | B | 118279 | 19.21 |
| cat\_input2 | C | 101607 | 16.50 |
| cat\_input2 | D | 71510 | 11.61 |
| cat\_input2 | E | 213629 | 34.70 |
| demog\_ho | 0 | 206092 | 33.47 |
| demog\_ho | 1 | 409622 | 66.53 |
| demog\_genf | 0 | 272906 | 44.32 |
| demog\_genf | 1 | 342808 | 55.68 |

**Appendix.b.1**

**---**

**title: "Final Project"**

**author: "Shaun Levenson"**

**date: "`r Sys.Date()`"**

**output:**

**html\_document:**

**toc: true**

**toc\_float: true**

**self\_contained: true**

**code\_folding: hide**

**---**

**```{r packages, echo=TRUE}**

**# Install the required packages only if missing and suppress redundant output**

**packages <- c("tidyverse", "caret", "knitr", "kableExtra", "skimr", "ggplot2","dplyr","ISLR","tree","rpart","rpart.plot","randomForest","gbm","doParallel","naniar","patchwork","MASS","scales","plotly","reshape2","e1071","ggcorrplot","gt","nortest","gridExtra")**

**new\_packages <- setdiff(packages, rownames(installed.packages()))**

**if (length(new\_packages) > 0) install.packages(new\_packages, quietly = TRUE)**

**# Load the required packages quietly**

**invisible(lapply(packages, function(pkg) suppressPackageStartupMessages(library(pkg, character.only = TRUE))))**

**```**

**# 1. Data Loading and Inspection**

**```{r load data, echo=TRUE}**

**#Import the dataset. Change the file path to where you saved your bank.txt file**

**bank <- read\_csv("F:/ECON/562\_Analytics\_2/Final Project/bank.txt",**

**col\_types = cols(b\_tgt = col\_character(),**

**int\_tgt = col\_number(), cnt\_tgt = col\_double(),**

**demog\_homeval = col\_number(), demog\_inc = col\_number(),**

**rfm1 = col\_number(), rfm2 = col\_number(),**

**rfm3 = col\_number(), rfm4 = col\_number(),**

**demog\_genf = col\_character(), demog\_genm = col\_character(),**

**dataset = col\_character()))**

**# Display the first 10 rows of the dataset**

**bank %>%**

**slice(1:10) %>%**

**kable("html", caption = "First 10 Rows of the Bank Dataset") %>%**

**kable\_styling(bootstrap\_options = c("striped", "hover", "condensed", "responsive"),**

**full\_width = FALSE,**

**position = "center")**

**str(bank)**

**summary(bank)**

**```**

**## Data Understanding**

**|Data Type | Description | Columns |**

**|---------|------------|-------|**

**|\*\*Numeric\*\* | Continuous variables or discrete numeric inputs. | 18 Columns: `int\_tgt`, `cnt\_tgt`, `rfm1-12`, `demog\_age`, `demog\_homeval`, `demog\_inc`, `demog\_pr`|**

**|\*\*Factor\*\* | Categorical with fixed levels | 6 Columns: `b\_tgt`, `cat\_input1`, `cat\_input2`, `demog\_ho`, `demog\_genf`, `demog\_genm` |**

**|\*\*Identifier\*\* | Unique identifiers for each row | 1 Column: `account` |**

**|\*\*Partition\*\* | Subset of the dataset | 1 Column: `dataset` |**

**\*\*Variable Descriptions\*\***

**| Variable | Type | Description |**

**|------------------|--------------|-----------------------------------------------------------|**

**| \*\*Target Variables\*\* |||**

**| `B\_TGT` | Binary | Tried a New Product (Yes/No — may be recoded as 1/0)|**

**| `INT\_TGT` | Numeric | Total New Sales |**

**|`CNT\_TGT`| Count| Count of New Products Purchased |**

**| \*\*Categorical Predictors\*\* |||**

**| `CAT\_INPUT1`| Categorical | Account Activity|**

**| `CAT\_INPUT2`| Categorical | Customer Value Level|**

**| \*\*RFM Interval Inputs\*\* |||**

**| `RFM1`| Numeric| Average Sales Past 3 Years|**

**| `RFM2`| Numeric| Average Sales Lifetime|**

**| `RFM3`| Numeric| Avg Sales Past 3 Years - Direct Promo Response|**

**| `RFM4`| Numeric | Last Product Purchase Amount|**

**| `RFM5`| Numeric| Count Purchased Past 3 Years|**

**| `RFM6`| Numeric| Count Purchased Lifetime |**

**| `RFM7`| Numeric| Count Purchased Past 3 Years - Direct Promo Response|**

**| `RFM8`| Numeric| Count Purchased Lifetime - Direct Promo Response|**

**| `RFM9`| Numeric| Months Since Last Purchase|**

**| `RFM10`| Numeric| Count Total Promos Past Year |**

**| `RFM11`| Numeric| Count Direct Promos Past Year|**

**| `RFM12`| Numeric| Customer Tenure|**

**| \*\*Demographic Inputs\*\* |||**

**| `DEMOG\_AGE`| Numeric | Customer Age |**

**| `DEMOG\_GENF`| Binary | Female (Yes/No)|**

**| `DEMOG\_GENM`| Binary | Male (Yes/No) |**

**| `DEMOG\_HO`| Binary | Homeowner (Yes/No)|**

**| `DEMOG\_HOMEVAL`| Numeric | Home Value|**

**| `DEMOG\_INC`| Numeric | Income |**

**| `DEMOG\_PR`| Numeric | Geographical Retirement Percentage**

**\*\*General Structure\*\***

**- The dataset contains \*\*1,060,038 rows\*\* and \*\*26 columns.\*\***

**- The dataset is divided into three subsets: training (1), validation (2), and testing (3).**

**- 3 target variables**

**- \*\*b\_tgt\*\*: Binary target variable indicating whether the customer tried a new product (yes/no).**

**- \*\*int\_tgt\*\*: Numeric target variable indicating the total number of new sales.**

**- \*\*cnt\_tgt\*\*: Count target variable indicating the total number of new products and services purchased by customers.**

**\*\*Potential Issues\*\***

**- Many missing values**

**- Redundant Gender columns: do not need both male and female. Will only keep the female column.**

**- Zero values in `demog\_homeval`, `demog\_inc`, and `demog\_pr` columns. These values are suspicious and may be treated as missing values.**

**# 2. Missing Values**

**```{r missing values, echo=TRUE}**

**# Missing Value Table**

**bank %>%**

**miss\_var\_summary() %>%**

**mutate(pct\_miss = round(pct\_miss, 2)) %>%**

**kable("html", caption = "Missing Data Summary for 'bank' Dataset") %>%**

**kable\_styling(bootstrap\_options = c("striped", "hover", "condensed", "responsive"),**

**full\_width = F,**

**position = "center")**

**```**

**\*\*Important Note:\*\***

**- The dataset contains a significant amount of missing values, particularly in the `int\_tgt` variable, which has \*\*80%\*\* missing values. This is not a major concern due to the nature of the modelling focus. The `int\_tgt` variable is set to N/A for any `b\_tgt` that is "no" and since we will be modelling for how much a customer will spend given they have purchased a new product, we can effectively ignore all missing values for this variable.**

**- The other two highly missing variables are `demog\_age` and `rfm3`, which have \*\*25.2%\*\* and \*\*21.3%\*\* missing values, respectively. These variables will be imputed using the median or mean values depending on the normality due to the importance of these variables.**

**# 3. Univariate Analysis**

**```{r summary statistics, echo=TRUE}**

**# Numerical variable names**

**numerical\_vars <- c("int\_tgt", "cnt\_tgt", "rfm1", "rfm2", "rfm3", "rfm4",**

**"rfm5", "rfm6", "rfm7", "rfm8", "rfm9", "rfm10",**

**"rfm11", "rfm12", "demog\_age", "demog\_homeval",**

**"demog\_inc", "demog\_pr")**

**# Function to compute summary statistics**

**compute\_summary\_statistics <- function(bank, numerical\_vars) {**

**summary\_stats <- bank %>%**

**dplyr::select(dplyr::all\_of(numerical\_vars)) %>%**

**dplyr::summarise(dplyr::across(**

**everything(),**

**list(**

**mean = ~mean(., na.rm = TRUE),**

**sd = ~sd(., na.rm = TRUE),**

**median = ~median(., na.rm = TRUE),**

**min = ~min(., na.rm = TRUE),**

**max = ~max(., na.rm = TRUE),**

**skewness = ~e1071::skewness(., na.rm = TRUE)**

**),**

**.names = "{col}\_{fn}"**

**)) %>%**

**tidyr::pivot\_longer(cols = everything(),**

**names\_to = c("Variable", "Statistic"),**

**names\_pattern = "(.\*)\_(.\*)",**

**values\_to = "Value") %>%**

**tidyr::pivot\_wider(names\_from = Statistic, values\_from = Value) %>%**

**dplyr::mutate(dplyr::across(where(is.numeric), ~round(.x, 2)))**

**return(summary\_stats)**

**}**

**# Run the function**

**summary\_statistics <- compute\_summary\_statistics(bank, numerical\_vars)**

**# Display nicely formatted HTML table**

**summary\_statistics %>%**

**kable("html", caption = "Summary Statistics for Selected Numerical Variables") %>%**

**kable\_styling(full\_width = TRUE, bootstrap\_options = c("striped", "hover")) %>%**

**scroll\_box(width = "100%", height = "500px")**

**```**

**```{r categorical variables\_stats, echo=TRUE}**

**# List of categorical variable names**

**categorical\_vars <- c("b\_tgt", "cat\_input1", "cat\_input2", "demog\_ho", "demog\_genf")**

**# Function to compute frequency tables**

**compute\_frequency\_tables <- function(bank, categorical\_vars) {**

**freq\_tables <- lapply(categorical\_vars, function(var) {**

**bank %>%**

**dplyr::select(dplyr::all\_of(var)) %>%**

**dplyr::filter(!is.na(.data[[var]])) %>%**

**dplyr::group\_by(.data[[var]]) %>%**

**dplyr::summarise(Count = n(), .groups = "drop") %>%**

**dplyr::mutate(**

**Level = as.character(.data[[var]]), # Force Level to be character**

**Percent = round(100 \* Count / sum(Count), 2),**

**Variable = var**

**) %>%**

**dplyr::select(Variable, Level, Count, Percent)**

**})**

**# Bind all tables into one**

**freq\_tables\_df <- dplyr::bind\_rows(freq\_tables)**

**return(freq\_tables\_df)**

**}**

**# Run the function**

**frequency\_tables <- compute\_frequency\_tables(bank, categorical\_vars)**

**# Display nicely formatted HTML table**

**frequency\_tables %>%**

**kable("html", caption = "Frequency Tables for Selected Categorical Variables") %>%**

**kable\_styling(full\_width = TRUE, bootstrap\_options = c("striped", "hover")) %>%**

**scroll\_box(width = "100%", height = "500px")**

**```**

**```{r univariate analysis visualization, echo=TRUE, warning=FALSE, message=FALSE}**

**# Define all numerical variables**

**numerical\_vars <- c("int\_tgt", "cnt\_tgt", paste0("rfm", 1:12),**

**"demog\_age", "demog\_homeval", "demog\_inc", "demog\_pr")**

**# Generate a vector of distinct colors**

**colors <- hue\_pal()(length(numerical\_vars))**

**# Create a list to store plots**

**num\_plots <- list()**

**for (i in seq\_along(numerical\_vars)) {**

**var <- numerical\_vars[i]**

**color <- colors[i]**

**plot\_data <- bank %>% filter(!is.na(.data[[var]]))**

**p <- ggplot(plot\_data, aes(x = .data[[var]])) +**

**geom\_histogram(bins = 30, fill = color, color = "black") +**

**labs(title = paste("Distribution of", var), x = var, y = "Frequency") +**

**theme\_minimal()**

**num\_plots[[i]] <- p**

**}**

**# Display in 2x2 grids**

**num\_chunks <- ceiling(length(num\_plots) / 4)**

**for (i in 1:num\_chunks) {**

**idx <- ((i - 1) \* 4 + 1):min(i \* 4, length(num\_plots))**

**grid.arrange(grobs = num\_plots[idx], ncol = 2)**

**}**

**# Now do the same for categorical variables**

**categorical\_vars <- c("b\_tgt", "cat\_input1", "cat\_input2", "demog\_ho", "demog\_genf", "demog\_genm")**

**cat\_plots <- list()**

**for (var in categorical\_vars) {**

**p <- ggplot(bank, aes(x = .data[[var]])) +**

**geom\_bar(fill = "#868686FF") +**

**labs(title = paste("Distribution of", var), x = var, y = "Count") +**

**theme\_minimal()**

**cat\_plots[[length(cat\_plots) + 1]] <- p**

**}**

**# Display in 2x2 grids**

**cat\_chunks <- ceiling(length(cat\_plots) / 4)**

**for (i in 1:cat\_chunks) {**

**idx <- ((i - 1) \* 4 + 1):min(i \* 4, length(cat\_plots))**

**grid.arrange(grobs = cat\_plots[idx], ncol = 2)**

**}**

**```**

**## Analysis**

**The notable observations from these distributations for consideration:**

**\*\*`b\_tgt`\*\*:**

**- This variable exhibits a significant imbalance, with a large number of "no" responses compared to "yes". This is common in marketing datasets where the target event (e.g., trying a new product) is rare.**

**- Class imbalance could cause models to favor the majority class, reducing predictive accuracy for the minority class. Addressing this is vital for effective model performance, especially in classification tasks.**

**\*\*`int\_tgt`\*\*:**

**- This variable shows high skewness, indicating a right-skewed distribution. This suggests that most customers have low total new sales, with a few customers having very high sales. This chart may also indicate the presence of outliers by the long right tail. Previously, the summary statistic showed a max value of 500000, which is likely a significant outlier. This will be properly addressed in the data cleaning section to return to a normal distribution as many regression models assume normality.**

**- This histogram does not include the missing values, which are set to NA. This is important to note as the missing values are not included in the analysis.**

**\*\*`cnt\_tgt`\*\*:**

**- This variable also shows a right-skewed distribution, similar to `int\_tgt`. This is due to most customers having 0 new products purchased, with a few customers having purchased multiple products. However, the summary statistics show that the skew of this variable is much lower than `int\_tgt`, indicating that the distribution is more normal, yet still may benefit from transformations.**

**# 4. Bivariate Analysis**

**```{r correlation matrix, echo=TRUE}**

**# Compute the correlation matrix for numeric columns in the bank dataset**

**correlation\_matrix <- cor(bank[, sapply(bank, is.numeric)], use = "complete.obs")**

**# Melt the correlation matrix to long format**

**melted\_correlation <- melt(correlation\_matrix)**

**# Truncate the correlation values to the nearest hundredth**

**melted\_correlation$value <- trunc(melted\_correlation$value \* 100) / 100**

**# Create a heatmap using ggplot2**

**heatmap\_plot <- ggplot(data = melted\_correlation, aes(Var1, Var2, fill = value)) +**

**geom\_tile(color = "white") +**

**scale\_fill\_gradient2(low = "blue", mid = "white", high = "red", midpoint = 0) +**

**labs(title = "Correlation Heatmap of Numerical Variables",**

**x = "Variable",**

**y = "Variable") +**

**theme\_minimal() +**

**theme(**

**plot.title = element\_text(hjust = 0.5, size = 16, face = "bold"),**

**axis.title.x = element\_text(size = 12, face = "bold"),**

**axis.title.y = element\_text(size = 12, face = "bold"),**

**axis.text.x = element\_text(angle = 45, hjust = 1, vjust = 1, size = 10),**

**axis.text.y = element\_text(size = 10)**

**)**

**# Convert to interactive plot**

**interactive\_heatmap <- ggplotly(heatmap\_plot, tooltip = c("Var1", "Var2", "value"))**

**# Show the heatmap**

**interactive\_heatmap**

**```**

**```{r correlations, echo=TRUE}**

**# Compute correlation matrix**

**cor\_matrix <- cor(bank %>% select(where(is.numeric)), use = "complete.obs")**

**# Melt to long format**

**cor\_long <- melt(cor\_matrix)**

**# Remove self-correlations and duplicate pairs**

**cor\_filtered <- cor\_long %>%**

**filter(Var1 != Var2) %>%**

**rowwise() %>%**

**mutate(pair = paste(sort(c(Var1, Var2)), collapse = "\_")) %>%**

**distinct(pair, .keep\_all = TRUE) %>%**

**filter(abs(value) > 0.6) %>%**

**arrange(desc(abs(value)))**

**# Create a nice table**

**cor\_table <- cor\_filtered %>%**

**select(Variable1 = Var1, Variable2 = Var2, Correlation = value) %>%**

**mutate(Correlation = round(Correlation, 2)) %>%**

**kable("html", caption = "Strong Correlations (>|0.6|)", escape = FALSE) %>%**

**kable\_styling(bootstrap\_options = c("striped", "hover", "condensed"), full\_width = FALSE)**

**# Display**

**cor\_table**

**```**

**## Analysis**

**| Correlation (Abs. Value) | Strength | Suggestion |**

**|-------------------|------------------|------------------------------------|**

**| 0.0–0.3 | Weak | Usually not worth flagging |**

**| 0.3–0.5 | Moderate | Context-dependent |**

**| 0.5–0.7 | Strong-ish | Often worth a closer look |**

**| 0.7–0.9+ | Very strong | Good candidates for multicollinearity check |**

**- There is extremely high correlation across the RFM variables, especially `rfm1`, `rfm2`, and `rfm3`.**

**- `rfm1` (Avg Sales past 3 yrs) is highly correlated with:**

**- `rfm2` (Avg Sales Lifetime): 0.91**

**- `rfm3` (Avg Sales Past 3 yrs Dir Promo Resp): 0.96**

**This is expected as these variables are all related to average sales over different time periods.**

**- `rfm6` (Count Purchased Lifetime) and `rfm8` (Count Purchased Lifetime Dir Promo Resp): 0.89**

**- `rfm5`(Count Purchased Past 3 Yrs) and `cnt\_tgt` (Target variable): 0.79**

**\*\*Modelling Approach\*\***

**### 1. Multicollinearity Warning**

**- Extremely high correlations (e.g., > 0.9), like between `rfm1` and `rfm3`, can cause multicollinearity, which may:**

**- Distort the interpretation of coefficients in linear models**

**- Inflate standard errors**

**- \*\*Solution\*\*: Consider removing one of each highly correlated pair (e.g., keep either `rfm1` or `rfm3`, not both), or use dimension reduction techniques such as:**

**- Principal Component Analysis (PCA)**

**- Regularization (e.g., Ridge, Lasso)**

**### 2. Feature Selection**

**- Strong correlation between some predictors (e.g., `rfm5`) and the target variable (`cnt\_tgt`) suggests these may be strong predictors:**

**- These should be prioritized in model training or feature selection algorithms.**

**### 3. Model Choice Consideration**

**- Linear models are sensitive to multicollinearity, but tree-based models (like Random Forest or Gradient Boosting) are more robust in this context.**

**- For models where interpretability matters, handling multicollinearity is crucial.**

**# 5. Data Quality and Outliers**

**```{r log transformation, echo=TRUE}**

**# Log transformation for skewed variables**

**skewed\_vars <- c("int\_tgt", "cnt\_tgt", "rfm1", "rfm2", "rfm3", "rfm4", "rfm10", "demog\_homeval")**

**bank[paste("log\_", skewed\_vars, sep = "")] <- lapply(bank[skewed\_vars], function(x) {**

**if (any(x <= 0, na.rm = TRUE)) {**

**x[x <= 0] <- NA**

**}**

**log(x)**

**})**

**# Check the distribution of the transformed variables**

**skewed\_vars\_transformed <- bank %>%**

**select(starts\_with("log\_")) %>%**

**gather(key = "Variable", value = "Value") %>%**

**filter(!is.na(Value))**

**ggplot(skewed\_vars\_transformed, aes(x = Value)) +**

**geom\_histogram(bins = 30, fill = "#868686FF", color = "black") +**

**facet\_wrap(~ Variable, scales = "free") +**

**labs(title = "Distribution of Log-Transformed Variables", x = "Log Value", y = "Count") +**

**theme\_minimal()**

**```**

**```{r outlier checks, echo=TRUE,warning=FALSE}**

**# Function to generate boxplot and return both plot and outlier count**

**outlier\_check\_plot <- function(data, var) {**

**# Generate boxplot**

**p <- ggplot(data, aes(y = .data[[var]])) +**

**geom\_boxplot(fill = "#00AFBB", color = "black", outlier.color = "red", outlier.shape = 16) +**

**labs(title = paste("Boxplot of", var), y = var) +**

**theme\_minimal()**

**# Identify outliers using 1.5\*IQR rule**

**qnt <- quantile(data[[var]], probs = c(0.25, 0.75), na.rm = TRUE)**

**iqr <- qnt[2] - qnt[1]**

**lower <- qnt[1] - 1.5 \* iqr**

**upper <- qnt[2] + 1.5 \* iqr**

**outliers <- which(data[[var]] < lower | data[[var]] > upper)**

**return(list(plot = p, count = length(outliers)))**

**}**

**# Define variables to check**

**variables\_to\_check <- c(**

**"log\_cnt\_tgt", "log\_demog\_homeval", "log\_int\_tgt", "log\_rfm1", "log\_rfm10",**

**"log\_rfm2", "log\_rfm3", "log\_rfm4", "rfm5", "rfm6", "rfm7", "rfm8",**

**"rfm9", "rfm11", "rfm12", "demog\_age", "demog\_inc", "demog\_pr"**

**)**

**# Store results**

**results <- lapply(variables\_to\_check, function(v) outlier\_check\_plot(bank, v))**

**names(results) <- variables\_to\_check**

**# Extract outlier counts and plots**

**outlier\_counts <- sapply(results, function(x) x$count)**

**outlier\_plots <- lapply(results, function(x) x$plot)**

**# Print summary sorted by outlier count**

**outlier\_summary <- data.frame(Variable = names(outlier\_counts), Outlier\_Count = outlier\_counts)**

**outlier\_summary <- outlier\_summary %>% arrange(desc(Outlier\_Count))**

**print(outlier\_summary)**

**# Display boxplots in 2x2 grid layout**

**plot\_chunks <- ceiling(length(outlier\_plots) / 4)**

**for (i in 1:plot\_chunks) {**

**idx <- ((i - 1) \* 4 + 1):min(i \* 4, length(outlier\_plots))**

**grid.arrange(grobs = outlier\_plots[idx], ncol = 2)**

**}**

**```**

**### Impact of Outliers and Strategies for Handling Them**

**Outliers can significantly affect the performance of predictive models. They can distort model parameters, increase variance, and reduce the ability of models to generalize well to unseen data. The strategies outlined below can help manage the influence of outliers:**

**#### \*\*Impact of Outliers\*\*:**

**- \*\*Linear Models\*\*: Can lead to biased coefficients and inflated standard errors.**

**- \*\*Distance-based Algorithms\*\*: Outliers influence distance calculations, affecting models like KNN and SVM.**

**- \*\*Tree-based Models\*\*: More robust but can still be affected by extreme values.**

**#### \*\*Strategies to Handle Outliers\*\*:**

**1. \*\*Removing Outliers\*\*:**

**- Use \*\*IQR\*\* or \*\*Z-score\*\* methods to remove extreme values.**

**2. \*\*Capping/Winsorization\*\*:**

**- Limit extreme values to a certain percentile (e.g., 95th percentile).**

**3. \*\*Log Transformation\*\*:**

**- Apply log or Box-Cox transformations to reduce the spread of data and make models less sensitive to outliers (already applied to `cnt\_tgt`, `int\_tgt`, etc.).**

**4. \*\*Robust Models\*\*:**

**- Use \*\*Ridge\*\*, \*\*Lasso\*\*, or \*\*Huber Regression\*\* to reduce the impact of outliers on model training.**

**5. \*\*Quantile Regression\*\*:**

**- Focus on specific quantiles (e.g., 50th percentile) to minimize the impact of outliers.**

**6. \*\*Outlier Detection\*\*:**

**- Use algorithms like \*\*Isolation Forest\*\* to automatically identify and handle outliers.**

**#### \*\*Strategy for This Dataset\*\*:**

**- \*\*Log Transform\*\*: Continue using the log-transformed variables (e.g., `cnt\_tgt`, `rfm1`, `rfm2`).**

**- \*\*Boxplots and IQR\*\*: Identify extreme outliers in both transformed and non-transformed variables.**

**- \*\*Capping or Removal\*\*: Cap or remove extreme outliers where appropriate (e.g., `cnt\_tgt`).**

**- \*\*Robust Models\*\*: Consider tree-based or regularized regression models to reduce the impact of outliers.**

**# 6. Target Variable Analysis**

**```{r target variable analysis, echo=TRUE, warning=FALSE}**

**# indicate the target variables int\_tgt, cnt\_tgt, and b\_tgt**

**# Create a bar plot for b\_tgt with count labels and plain y-axis**

**b\_tgt\_plot <- ggplot(bank, aes(x = as.factor(b\_tgt))) +**

**geom\_bar(fill = "#00AFBB", color = "black") +**

**geom\_text(stat = "count", aes(label = ..count..), vjust = -0.5, size = 4) +**

**scale\_y\_continuous(labels = scales::comma) +**

**labs(title = "Distribution of b\_tgt", x = "b\_tgt", y = "Count") +**

**theme\_minimal()**

**# Create histogram for int\_tgt with colors and plain y-axis**

**int\_tgt\_hist <- ggplot(bank, aes(x = int\_tgt)) +**

**geom\_histogram(bins = 30, fill = "#E7B800", color = "black") +**

**scale\_y\_continuous(labels = scales::comma) +**

**labs(title = "Histogram of int\_tgt", x = "int\_tgt", y = "Frequency") +**

**theme\_minimal()**

**# Create histogram for cnt\_tgt with colors and plain y-axis**

**cnt\_tgt\_hist <- ggplot(bank, aes(x = cnt\_tgt)) +**

**geom\_histogram(bins = 30, fill = "#FC4E07", color = "black") +**

**scale\_y\_continuous(labels = scales::comma) +**

**labs(title = "Histogram of cnt\_tgt", x = "cnt\_tgt", y = "Frequency") +**

**theme\_minimal()**

**# Create a list to store plots**

**plots <- list()**

**plots$b\_tgt\_plot <- b\_tgt\_plot**

**plots$int\_tgt\_hist <- int\_tgt\_hist**

**plots$cnt\_tgt\_hist <- cnt\_tgt\_hist**

**# Display the plots**

**plots$b\_tgt\_plot**

**plots$int\_tgt\_hist**

**plots$cnt\_tgt\_hist**

**```**

**```{r tgt\_stats, echo=TRUE}**

**# Load gt**

**library(gt)**

**# Define target variables**

**tgt\_vars <- c("int\_tgt", "cnt\_tgt")**

**# Function to compute summary statistics**

**compute\_summary\_statistics <- function(bank, target\_vars) {**

**summary\_stats <- bank %>%**

**dplyr::select(dplyr::all\_of(target\_vars)) %>%**

**dplyr::summarise(dplyr::across(**

**everything(),**

**list(**

**mean = ~mean(., na.rm = TRUE),**

**sd = ~sd(., na.rm = TRUE),**

**median = ~median(., na.rm = TRUE),**

**min = ~min(., na.rm = TRUE),**

**max = ~max(., na.rm = TRUE),**

**skewness = ~e1071::skewness(., na.rm = TRUE)**

**),**

**.names = "{col}\_{fn}"**

**)) %>%**

**tidyr::pivot\_longer(cols = everything(),**

**names\_to = c("Variable", "Statistic"),**

**names\_pattern = "(.\*)\_(.\*)",**

**values\_to = "Value") %>%**

**tidyr::pivot\_wider(names\_from = Statistic, values\_from = Value) %>%**

**dplyr::mutate(dplyr::across(where(is.numeric), ~round(.x, 2)))**

**return(summary\_stats)**

**}**

**# Run the function for target variables**

**summary\_statistics\_tgt <- compute\_summary\_statistics(bank, tgt\_vars)**

**# Create a beautiful gt table**

**summary\_statistics\_tgt %>%**

**gt() %>%**

**fmt\_number(**

**columns = where(is.numeric),**

**decimals = 2, # Keep 2 decimal places**

**use\_seps = TRUE # Add commas (thousand separators)**

**) %>%**

**tab\_header(**

**title = "Summary Statistics for Target Variables"**

**) %>%**

**tab\_options(**

**table.width = pct(100),**

**table.font.size = 14,**

**heading.align = "center",**

**data\_row.padding = px(6)**

**)**

**```**

**```{r b\_tgt\_stats, echo=TRUE}**

**# Function to compute summary statistics for a binary variable**

**compute\_binary\_summary <- function(data, binary\_var) {**

**binary\_data <- data[[binary\_var]]**

**# Remove missing values**

**binary\_data <- binary\_data[!is.na(binary\_data)]**

**total\_n <- length(binary\_data)**

**n\_ones <- sum(binary\_data == 1)**

**n\_zeros <- sum(binary\_data == 0)**

**prop\_ones <- round(n\_ones / total\_n, 2)**

**prop\_zeros <- round(n\_zeros / total\_n, 2)**

**mode\_val <- ifelse(n\_ones >= n\_zeros, 1, 0)**

**summary\_df <- data.frame(**

**Variable = binary\_var,**

**Count = total\_n,**

**Count\_1s = n\_ones,**

**Count\_0s = n\_zeros,**

**Proportion\_1s = prop\_ones,**

**Proportion\_0s = prop\_zeros,**

**Mode = mode\_val**

**)**

**return(summary\_df)**

**}**

**# Run for binary variable b\_tgt**

**binary\_summary <- compute\_binary\_summary(bank, "b\_tgt")**

**# Display nicely formatted HTML table**

**binary\_summary %>%**

**kable("html", caption = "Summary Statistics for Binary Target Variable (b\_tgt)") %>%**

**kable\_styling(full\_width = FALSE, bootstrap\_options = c("striped", "hover"))**

**```**

**```{r normality tests, echo=TRUE}**

**# Load necessary library**

**library(nortest)**

**# Formal Normality Testing (Large Sample Safe)**

**for (var in tgt\_vars) {**

**cat("\nAnderson-Darling Test for", var, "\n")**

**print(ad.test(bank[[var]]))**

**cat("\nKolmogorov-Smirnov Test for", var, "\n")**

**# Standardize the variable first**

**var\_std <- scale(bank[[var]])**

**print(ks.test(var\_std, "pnorm"))**

**}**

**```**

**## Normality Tests for Target Variables**

**### Anderson-Darling Test Results**

**- `int\_tgt`:**

**- Test Statistic: A = 4250.3**

**- p-value < 2.2e-16**

**- `cnt\_tgt`:**

**- Test Statistic: A = 242695**

**- p-value < 2.2e-16**

**Both tests strongly reject the null hypothesis of normality (p-value << 0.05), indicating that neither `int\_tgt` nor `cnt\_tgt` follows a normal distribution.**

**### Kolmogorov-Smirnov Test Results**

**- `int\_tgt`:**

**- D = 0.11227**

**- p-value < 2.2e-16**

**- `cnt\_tgt`:**

**- D = 0.47252**

**- p-value < 2.2e-16**

**Similar to the Anderson-Darling test, the Kolmogorov-Smirnov test results indicate significant deviation from a normal distribution.**

**Note: The KS test assumes no ties in the data, and warnings were issued, but given the extremely small p-values, the evidence against normality remains very strong.**

**## Interpretation and Impact on Modeling Choices**

**Given the strong evidence of non-normality for both `int\_tgt` and `cnt\_tgt`:**

**- Linear regression models may perform poorly unless transformations are applied.**

**- Transformations such as log(x + 1) or square root could be considered to reduce skewness if linear models are still desired.**

**- Alternatively, non-parametric or tree-based models (e.g., Random Forest, Gradient Boosting) can be applied, as they do not assume normality of the target.**

**- Performance metrics such as Mean Absolute Error (MAE) may be preferred over Mean Squared Error (MSE) to reduce the influence of extreme values (outliers).**

**Additionally:**

**- `cnt\_tgt`, being a count variable, may be better modeled with Poisson regression or Negative Binomial regression depending on dispersion.**

**- If `cnt\_tgt` has a high proportion of zeros, Zero-Inflated models could also be appropriate.**

**# 7. Predictor Analysis**

**```{r categorical predictor analysis, echo=TRUE}**

**# List of categorical variable names excluding b\_tgt**

**categorical\_vars <- c("cat\_input1", "cat\_input2", "demog\_ho", "demog\_genf")**

**# Function to compute frequency tables and create bar charts**

**compute\_frequency\_and\_plot <- function(bank, categorical\_vars) {**

**# Loop through each variable**

**lapply(categorical\_vars, function(var) {**

**# Frequency table for the variable**

**freq\_table <- bank %>%**

**dplyr::select(dplyr::all\_of(var)) %>%**

**dplyr::filter(!is.na(.data[[var]])) %>%**

**dplyr::group\_by(.data[[var]]) %>%**

**dplyr::summarise(Count = n(), .groups = "drop") %>%**

**dplyr::mutate(**

**Level = as.character(.data[[var]]), # Ensure Level is character**

**Percent = round(100 \* Count / sum(Count), 2),**

**Variable = var**

**) %>%**

**dplyr::select(Variable, Level, Count, Percent)**

**# Display frequency table**

**print(freq\_table)**

**# Plot bar chart for the variable**

**ggplot(freq\_table, aes(x = Level, y = Count, fill = Level)) +**

**geom\_bar(stat = "identity", show.legend = FALSE) +**

**theme\_minimal() +**

**labs(title = paste("Frequency of", var), x = "Category", y = "Count") +**

**theme(axis.text.x = element\_text(angle = 45, hjust = 1))**

**})**

**}**

**# Run the function for categorical variables**

**compute\_frequency\_and\_plot(bank, categorical\_vars)**

**```**

**### Categorical Predictors Analysis**

**We analyzed the categorical predictors in the dataset to understand the number of categories each predictor has and how these categories are distributed. Below are the findings for each of the predictors:**

**- \*\*cat\_input1 (Account Activity)\*\***

**- This variable has three categories:**

**- \*\*X\*\*: 831,371 observations (78.43%)**

**- \*\*Y\*\*: 77,847 observations (7.34%)**

**- \*\*Z\*\*: 150,820 observations (14.23%)**

**- The majority of the observations (78.43%) are in the "X" category, indicating that most customers fall under this category. The "Y" category has a significantly smaller proportion (7.34%), while the "Z" category has 14.23% of the observations.**

**- \*\*cat\_input2 (Customer Value Level)\*\***

**- This variable has five categories:**

**- \*\*A\*\*: 188,398 observations (17.77%)**

**- \*\*B\*\*: 192,382 observations (18.15%)**

**- \*\*C\*\*: 169,550 observations (15.99%)**

**- \*\*D\*\*: 122,282 observations (11.54%)**

**- \*\*E\*\*: 387,426 observations (36.55%)**

**- The distribution of the "cat\_input2" variable shows that the largest proportion of customers (36.55%) fall under the "E" category, followed by "B" (18.15%) and "A" (17.77%). Categories "C" and "D" have relatively smaller proportions, with "D" having the lowest proportion at 11.54%.**

**- \*\*demog\_ho (Homeownership)\*\***

**- This binary variable indicates whether the customer is a homeowner:**

**- \*\*0 (Not Homeowner)\*\*: 476,741 observations (44.97%)**

**- \*\*1 (Homeowner)\*\*: 583,297 observations (55.03%)**

**- The dataset is fairly evenly distributed between homeowners and non-homeowners, with a slightly higher percentage of homeowners (55.03%).**

**- \*\*demog\_genf (Gender - Female)\*\***

**- This binary variable indicates the gender of the customer:**

**- \*\*0 (Not Female)\*\*: 464,288 observations (43.8%)**

**- \*\*1 (Female)\*\*: 595,750 observations (56.2%)**

**- A majority of the customers (56.2%) are categorized as female, with 43.8% being categorized as not female.**

**### Numerical Predictors Analysis**

**## Analysis of Numerical Predictors and Correlation Insights**

**### 1. Range of Values for Numerical Predictors**

**The following table provides a summary of the range of values for the key numerical predictors in the dataset from prior code:**

**| Variable | Range |**

**|----------------|-------------------|**

**| \*\*int\_tgt\*\* | 0.00 to 500,000 |**

**| \*\*cnt\_tgt\*\* | 0.00 to 6.00 |**

**| \*\*rfm1\*\* | 0.00 to 3,713.31 |**

**| \*\*rfm2\*\* | 1.58 to 650.00 |**

**| \*\*rfm3\*\* | 0.00 to 3,713.31 |**

**| \*\*rfm4\*\* | 0.00 to 10,000.00 |**

**| \*\*rfm5\*\* | 0.00 to 18.00 |**

**| \*\*rfm6\*\* | 0.00 to 127.00 |**

**| \*\*rfm7\*\* | 0.00 to 11.00 |**

**| \*\*rfm8\*\* | 0.00 to 46.00 |**

**| \*\*rfm9\*\* | 2.00 to 29.00 |**

**| \*\*rfm10\*\* | 0.00 to 77.00 |**

**| \*\*rfm11\*\* | 0.00 to 22.00 |**

**| \*\*rfm12\*\* | 0.00 to 571.00 |**

**| \*\*demog\_age\*\* | -1.00 to 89.00 |**

**| \*\*demog\_homeval\*\* | 0.00 to 600,067.00 |**

**| \*\*demog\_inc\*\* | 0.00 to 200,007.00|**

**### 2. Observations from Range Analysis**

**- \*\*Wide Range of Values:\*\* Variables such as \*\*int\_tgt\*\*, \*\*rfm1\*\*, \*\*rfm3\*\*, \*\*rfm4\*\*, \*\*demog\_homeval\*\*, and \*\*demog\_inc\*\* exhibit very large ranges, suggesting a high degree of variability. This might indicate the presence of extreme outliers that could skew analyses and should be checked further.**

**- \*\*Narrower Range:\*\* Variables like \*\*cnt\_tgt\*\* and \*\*rfm5\*\* have narrower ranges, indicating less variation and potential stability across customers. These could serve as stable predictors in models.**

**- \*\*Data Quality Issues:\*\* The variable \*\*demog\_age\*\* includes a negative value (-1.00), which likely represents missing or invalid data. This should be addressed during data cleaning.**

**---**

**### 3. Correlation Insights**

**The following table highlights some key correlation values between numerical predictors:**

**| Variable1 | Variable2 | Correlation |**

**|------------|------------|-------------|**

**| \*\*rfm1\*\* | \*\*rfm2\*\* | 0.91 |**

**| \*\*rfm1\*\* | \*\*rfm3\*\* | 0.96 |**

**| \*\*rfm2\*\* | \*\*rfm3\*\* | 0.88 |**

**| \*\*rfm5\*\* | \*\*cnt\_tgt\*\*| 0.79 |**

**| \*\*rfm6\*\* | \*\*rfm8\*\* | 0.89 |**

**#### Correlation Observations:**

**- There is an \*\*extremely high correlation\*\* between \*\*rfm1\*\*, \*\*rfm2\*\*, and \*\*rfm3\*\*, with values reaching up to 0.96. This is expected because these variables represent different aspects of sales and promotional responses over time, leading to strong relationships between them.**

**- \*\*rfm6\*\* (Count Purchased Lifetime) and \*\*rfm8\*\* (Count Purchased Lifetime Dir Promo Resp) are also highly correlated (0.89), indicating that the total purchases and purchases via direct promotions are closely linked.**

**- \*\*rfm5\*\* (Count Purchased Past 3 Years) shows a moderate correlation with the target variable \*\*cnt\_tgt\*\* (0.79), making it a valuable predictor for modeling.**

**---**

**### 4. Multicollinearity Concerns**

**Given the \*\*very strong correlations\*\* observed between several predictors (e.g., \*\*rfm1\*\* and \*\*rfm3\*\*), we must be cautious about potential \*\*multicollinearity\*\*. Multicollinearity can distort model interpretations by inflating standard errors and making coefficient estimates unreliable.**

**#### Recommendations to Address Multicollinearity:**

**- \*\*Variable Removal or Combination:\*\* One approach is to remove one variable from each highly correlated pair (e.g., keep \*\*rfm1\*\* or \*\*rfm3\*\*, but not both). This reduces redundancy and helps stabilize the model.**

**- \*\*Principal Component Analysis (PCA):\*\* PCA can be used to reduce dimensionality while maintaining the variance across the correlated predictors.**

**- \*\*Regularization Techniques:\*\* Using techniques like \*\*Ridge\*\* or \*\*Lasso regression\*\* can help mitigate multicollinearity by shrinking the coefficients of less important variables.**

**---**

**### 5. Feature Selection**

**The strong correlation between some of the predictors and the target variable (\*\*cnt\_tgt\*\*) suggests that these features should be prioritized for model training. Notably, \*\*rfm5\*\* (Count Purchased Past 3 Years) shows a correlation of 0.79 with \*\*cnt\_tgt\*\*, making it a key predictor.**

**#### Prioritizing Strong Predictors:**

**- \*\*rfm5\*\* and \*\*cnt\_tgt\*\* should be retained in the feature set, as they have a strong relationship.**

**- Other variables with moderate to high correlations with the target, such as \*\*rfm7\*\* and \*\*rfm12\*\*, could also be considered for inclusion.**

**---**

**### 6. Modelling Considerations**

**When selecting models, it's important to consider how multicollinearity might impact performance, especially for linear models. \*\*Tree-based models\*\* like \*\*Random Forest\*\* or \*\*Gradient Boosting Machines (GBM)\*\* are robust to multicollinearity and can handle correlated features without significant issues.**

**However, if model interpretability is a priority, addressing multicollinearity should be part of the preprocessing steps to ensure clearer insights into predictor relationships and model coefficients.**

**## Impact of Predictor Characteristics on Feature Selection and Engineering**

**### 1. \*\*High Correlation Among RFM Variables\*\***

**A major observation in the data is the \*\*extremely high correlation\*\* among several of the RFM variables, particularly \*\*rfm1\*\*, \*\*rfm2\*\*, and \*\*rfm3\*\*, with correlations ranging from 0.88 to 0.96. These variables are related to different aspects of sales and promotional response over time, leading to high multicollinearity.**

**#### \*\*Impact on Feature Selection:\*\***

**- \*\*Multicollinearity Issues:\*\* High multicollinearity between these predictors can cause instability in linear models, where the model may struggle to differentiate between correlated variables, leading to inflated standard errors and less reliable coefficient estimates.**

**- \*\*Feature Engineering Approach:\*\* To mitigate this issue, we can:**

**- \*\*Remove Redundant Features:\*\* Consider removing one of the highly correlated variables from each pair (e.g., choosing between \*\*rfm1\*\* or \*\*rfm3\*\*) to reduce redundancy.**

**- \*\*Dimensionality Reduction:\*\* Use \*\*Principal Component Analysis (PCA)\*\* to transform these correlated variables into uncorrelated principal components, capturing most of the variance while eliminating multicollinearity.**

**- \*\*Regularization Techniques:\*\* If retaining multiple correlated variables is necessary for predictive power, using regularization techniques such as \*\*Ridge Regression\*\* or \*\*Lasso\*\* can help by penalizing large coefficients, thus reducing multicollinearity.**

**### 2. \*\*Skewed Distributions in Certain Variables\*\***

**Several predictors, such as \*\*int\_tgt\*\*, \*\*rfm1\*\*, \*\*rfm3\*\*, \*\*rfm4\*\*, and \*\*demog\_homeval\*\*, exhibit skewed distributions with heavy tails (e.g., \*\*int\_tgt\*\* has a skewness of 13.30). This could indicate the presence of outliers or extreme values that may disproportionately influence model training.**

**#### \*\*Impact on Feature Engineering:\*\***

**- \*\*Handling Skewed Distributions:\*\* For skewed variables, we can apply \*\*log transformation\*\* or \*\*Box-Cox transformation\*\* to stabilize variance and reduce the impact of outliers. This transformation can make these variables more normally distributed, improving the performance of certain models, particularly linear models and models that assume normality.**

**- \*\*Outlier Detection and Treatment:\*\* Identifying and addressing outliers in these variables (e.g., capping or winsorizing extreme values) can also prevent them from dominating the model’s learning process.**

**### 3. \*\*Numerical Predictors with Large Range of Values\*\***

**Some variables like \*\*demog\_homeval\*\* and \*\*demog\_inc\*\* have large ranges, spanning from zero to very large values. These wide ranges may lead to challenges in models that rely on scale-sensitive techniques (e.g., linear models, SVMs).**

**#### \*\*Impact on Feature Engineering:\*\***

**- \*\*Feature Scaling:\*\* To address this, \*\*standardization\*\* (zero mean, unit variance) or \*\*min-max scaling\*\* (rescaling the range to [0, 1]) should be applied to these predictors. This will ensure that all variables contribute equally to the model and prevent variables with large ranges from dominating the learning process.**

**- \*\*Log Transformation:\*\* For financial variables like \*\*demog\_homeval\*\* and \*\*demog\_inc\*\*, log transformation can be particularly effective in compressing large values and reducing skewness.**

**### 4. \*\*Target Variable Characteristics\*\***

**The target variable \*\*cnt\_tgt\*\* (Count of New Product Purchased) has a wide range but a relatively low mean and a moderate skewness (2.40). This suggests that most customers purchase a small number of products, but there are a few customers with very high counts.**

**#### \*\*Impact on Feature Engineering:\*\***

**- \*\*Target Transformation:\*\* To deal with the skewness, we may consider applying a \*\*log transformation\*\* to \*\*cnt\_tgt\*\*, turning the skewed distribution into a more normal-like distribution. This transformation will help certain models, like linear regression or tree-based models, to better predict the target.**

**- \*\*Handling Zero-Inflation:\*\* Given that \*\*cnt\_tgt\*\* has many zero values, it might benefit from being treated as a \*\*zero-inflated count variable\*\*. Techniques like \*\*Poisson regression\*\* or \*\*Negative Binomial regression\*\* could be considered for modeling this type of target variable.**

**### 5. \*\*Demographic Variables\*\***

**Demographic variables like \*\*demog\_age\*\*, \*\*demog\_homeval\*\*, and \*\*demog\_inc\*\* have important implications for feature engineering. For example, \*\*demog\_age\*\* exhibits a moderate level of skewness and includes negative values (likely indicating missing or erroneous data), which suggests that this variable needs to be cleaned and transformed.**

**#### \*\*Impact on Feature Selection and Engineering:\*\***

**- \*\*Data Cleaning:\*\* Any invalid or missing values in \*\*demog\_age\*\* (e.g., the negative value) should be addressed through imputation or removal, depending on the context and the percentage of missing data.**

**- \*\*Categorization:\*\* Age-related variables like \*\*demog\_age\*\* may also be grouped into \*\*age bands\*\* (e.g., 18-25, 26-35) to capture non-linear relationships between age and target behavior.**

**- \*\*Feature Interaction:\*\* Interaction terms between demographic variables and other predictors (e.g., \*\*demog\_inc\*\* with \*\*rfm5\*\*) could uncover valuable insights, especially in tree-based models, where interactions are automatically handled.**

**### 6. \*\*Targeted Feature Selection\*\***

**Given the strong correlation between some of the RFM variables and the target variable \*\*cnt\_tgt\*\* (e.g., \*\*rfm5\*\* and \*\*cnt\_tgt\*\* have a correlation of 0.79), it is essential to focus on features that exhibit strong predictive power.**

**#### \*\*Impact on Feature Selection:\*\***

**- \*\*Prioritize Strong Correlates:\*\* Variables like \*\*rfm5\*\* and other features that show strong relationships with the target (e.g., \*\*rfm12\*\*, \*\*rfm7\*\*) should be given priority during feature selection.**

**- \*\*Redundant Feature Removal:\*\* Features that are highly correlated with each other and with the target should be reviewed and potentially removed to avoid overfitting.**

**Appendix.b.2**

**---**

**title: "Final Project"**

**author: "Shaun Levenson"**

**date: "`r Sys.Date()`"**

**output:**

**html\_document:**

**toc: true**

**toc\_float: true**

**self\_contained: true**

**code\_folding: hide**

**---**

**\*\*Packages\*\***

**```{r packages, echo=TRUE, message=FALSE, warning=FALSE}**

**options(timeout = 300)**

**options(java.parameters = "-Xmx8g")**

**# Install the required packages only if missing and suppress redundant output**

**packages <- c("h2o","readr","kableExtra","dplyr","ggplot2","knitr","tidyverse","scales","gridExtra","patchwork","e1071","mlbench","tidymodels")**

**new\_packages <- setdiff(packages, rownames(installed.packages()))**

**if (length(new\_packages) > 0) install.packages(new\_packages, quietly = TRUE)**

**# Load the required packages quietly**

**invisible(lapply(packages, function(pkg) suppressPackageStartupMessages(library(pkg, character.only = TRUE))))**

**invisible(capture.output(h2o.init()))**

**cat("Packages used:", packages, "\n")**

**```**

**```{r data, echo=TRUE}**

**load("F:/ECON/562\_Analytics\_2/Final Project/bank.Rda")**

**# Remove the columns that are not needed for the analysis**

**bank$account <- NULL**

**bank$dataset <- NULL**

**bank$demog\_genm <- NULL**

**# Set cat\_input1, cat\_input2, and demog\_genf as factors**

**bank[] <- lapply(bank, function(x) if(is.character(x)) as.factor(x) else x)**

**```**

**# Missing Values, Outliers, and Data Cleaning**

**```{r missing, echo=TRUE}**

**# Check for missing values in the dataset**

**missing\_values <- sapply(bank, function(x) sum(is.na(x)))**

**missing\_percentage <- sapply(bank, function(x) sum(is.na(x)) / length(x) \* 100)**

**missing\_summary <- data.frame(**

**Variable = names(missing\_values),**

**Missing\_Values = missing\_values,**

**Missing\_Percentage = round(missing\_percentage, 2)**

**)**

**# remove rows with 0 missing values**

**missing\_summary <- missing\_summary[missing\_summary$Missing\_Values > 0, ]**

**# Sort the summary by missing values**

**missing\_summary <- missing\_summary[order(-missing\_summary$Missing\_Values), ]**

**# remove row names**

**rownames(missing\_summary) <- NULL**

**# Display the missing values summary in kable**

**kable(missing\_summary, caption = "Missing Values Summary") %>%**

**kable\_styling(full\_width = FALSE, position = "left") %>%**

**column\_spec(1, bold = TRUE) %>%**

**column\_spec(2, color = "blue") %>%**

**column\_spec(3, color = "red")**

**```**

**```{r missing\_removal, echo=TRUE}**

**# Impute int\_tgt missing values with 0**

**bank$int\_tgt[is.na(bank$int\_tgt)] <- 0**

**# Remove rows with missing values for demog\_age and rfm3**

**bank <- bank[!is.na(bank$demog\_age) & !is.na(bank$rfm3), ]**

**# Remove rows with missing values for cnt\_tgt**

**bank <- bank[!is.na(bank$cnt\_tgt), ]**

**```**

**There are 4 variables with missing values in the dataset. The variables are `int\_tgt`, `demog\_age`, `rfm3`, and `cnt\_tgt`. The percentage of missing values is also shown in the table.**

**\*\*Missing Removal Reasoning\*\***

**- `int\_tgt`: This variable is only set to `NA` when the customer is did not purchase a product (aka when b\_tgt is set = 0). We can impute this with `0` to indicate that they did not spend any money on purchasing a product.**

**- Additionally, only a small percentage of the data for `int\_tgt` is set to 0, so we can impute the missing values with `0` without losing too much information.**

**- For `demog\_age` and `rfm3`, I decided to exclude any observation with missing values. This is because the client is looking for a highly interpretable model, and I believe that using imputation methods will harm the model's interpretability.**

**- same for `cnt\_tgt` as well because there was only a single observation with a missing value.**

**As for `demog\_age`, I decided to remove any observations with an age of less than 16. This is because the client is a financial institution and including customers under the age of 16 will not be useful for the analysis due to legal reasons. Additionally, individuals under 16 years old typically do not have financial authority or independence, making them less relevant for the analysis.**

**```{r skewness, echo=TRUE}**

**# Numerical variable names**

**numerical\_vars <- c("int\_tgt", "cnt\_tgt", "rfm1", "rfm2", "rfm3", "rfm4",**

**"rfm5", "rfm6", "rfm7", "rfm8", "rfm9", "rfm10",**

**"rfm11", "rfm12", "demog\_age", "demog\_homeval",**

**"demog\_inc", "demog\_pr")**

**# Remove rows with age less than 16**

**bank <- bank[bank$demog\_age >= 16, ]**

**# Function to compute summary statistics**

**compute\_summary\_statistics <- function(bank, numerical\_vars) {**

**summary\_stats <- bank %>%**

**dplyr::select(dplyr::all\_of(numerical\_vars)) %>%**

**dplyr::summarise(dplyr::across(**

**everything(),**

**list(**

**mean = ~mean(., na.rm = TRUE),**

**sd = ~sd(., na.rm = TRUE),**

**median = ~median(., na.rm = TRUE),**

**min = ~min(., na.rm = TRUE),**

**max = ~max(., na.rm = TRUE),**

**skewness = ~e1071::skewness(., na.rm = TRUE)**

**),**

**.names = "{col}\_{fn}"**

**)) %>%**

**tidyr::pivot\_longer(cols = everything(),**

**names\_to = c("Variable", "Statistic"),**

**names\_pattern = "(.\*)\_(.\*)",**

**values\_to = "Value") %>%**

**tidyr::pivot\_wider(names\_from = Statistic, values\_from = Value) %>%**

**dplyr::mutate(dplyr::across(where(is.numeric), ~round(.x, 2)))**

**return(summary\_stats)**

**}**

**# Run the function**

**summary\_statistics <- compute\_summary\_statistics(bank, numerical\_vars)**

**# Display nicely formatted HTML table**

**summary\_statistics %>%**

**kable("html", caption = "Summary Statistics for Selected Numerical Variables") %>%**

**kable\_styling(full\_width = TRUE, bootstrap\_options = c("striped", "hover")) %>%**

**scroll\_box(width = "100%", height = "500px")**

**```**

**```{r categorical, echo=TRUE}**

**# List of categorical variable names**

**categorical\_vars <- c("b\_tgt", "cat\_input1", "cat\_input2", "demog\_ho", "demog\_genf")**

**# Function to compute frequency tables**

**compute\_frequency\_tables <- function(bank, categorical\_vars) {**

**freq\_tables <- lapply(categorical\_vars, function(var) {**

**bank %>%**

**dplyr::select(dplyr::all\_of(var)) %>%**

**dplyr::filter(!is.na(.data[[var]])) %>%**

**dplyr::group\_by(.data[[var]]) %>%**

**dplyr::summarise(Count = n(), .groups = "drop") %>%**

**dplyr::mutate(**

**Level = as.character(.data[[var]]), # Force Level to be character**

**Percent = round(100 \* Count / sum(Count), 2),**

**Variable = var**

**) %>%**

**dplyr::select(Variable, Level, Count, Percent)**

**})**

**# Bind all tables into one**

**freq\_tables\_df <- dplyr::bind\_rows(freq\_tables)**

**return(freq\_tables\_df)**

**}**

**# Run the function**

**frequency\_tables <- compute\_frequency\_tables(bank, categorical\_vars)**

**# Display nicely formatted HTML table**

**frequency\_tables %>%**

**kable("html", caption = "Frequency Tables for Selected Categorical Variables") %>%**

**kable\_styling(full\_width = TRUE, bootstrap\_options = c("striped", "hover")) %>%**

**scroll\_box(width = "100%", height = "500px")**

**```**

**```{r transformations, echo=TRUE}**

**# transform int\_tgt, rfm1, rfm2, rfm3, rfm4, rfm10 using log(x+1) transformation, have the new variables named log\_(x)**

**bank$log\_int\_tgt <- log(bank$int\_tgt + 1)**

**bank$log\_rfm1 <- log(bank$rfm1 + 1)**

**bank$log\_rfm2 <- log(bank$rfm2 + 1)**

**bank$log\_rfm3 <- log(bank$rfm3 + 1)**

**bank$log\_rfm4 <- log(bank$rfm4 + 1)**

**bank$log\_rfm10 <- log(bank$rfm10 + 1)**

**# Create a new summary statistic data frame with the transformed variables**

**transformed\_vars <- c("log\_int\_tgt", "log\_rfm1", "log\_rfm2", "log\_rfm3", "log\_rfm4", "log\_rfm10")**

**transformed\_summary\_statistics <- compute\_summary\_statistics(bank, transformed\_vars)**

**# Display the summary statistics for transformed variables**

**transformed\_summary\_statistics %>%**

**kable("html", caption = "Summary Statistics for Transformed Variables") %>%**

**kable\_styling(full\_width = TRUE, bootstrap\_options = c("striped", "hover")) %>%**

**scroll\_box(width = "100%", height = "500px")**

**```**

**\*\*Transformation Reasoning\*\***

**- For each variable that was highly skewed, I applied a log transformation to reduce the skewness and make the distribution more normal. This is important for many machine learning algorithms that assume normally distributed data.**

**- Additionally, using a log transformation allows for more interpretable coefficients in the models, as they can be interpreted as percentage changes.**

**```{r before and after, echo=TRUE}**

**vars <- c("int\_tgt", "rfm1", "rfm2", "rfm3", "rfm4", "rfm10")**

**plot\_list <- list()**

**for (v in vars) {**

**p1 <- ggplot(bank, aes\_string(x = v)) +**

**geom\_histogram(bins = 30, fill = "skyblue", color = "black") +**

**labs(title = paste("Original", v), x = v, y = "Count") +**

**theme\_minimal()**

**log\_var <- paste0("log\_", v)**

**p2 <- ggplot(bank, aes\_string(x = log\_var)) +**

**geom\_histogram(bins = 30, fill = "salmon", color = "black") +**

**labs(title = paste("Log-transformed", log\_var), x = log\_var, y = "Count") +**

**theme\_minimal()**

**# Combine plots side by side**

**plot\_list[[v]] <- gridExtra::grid.arrange(p1, p2, ncol = 2)**

**}**

**# To display all plots, just run this:**

**# This will print each pair of plots one after another**

**for (p in plot\_list) {**

**print(p)**

**}**

**```**

**# Interactions**

**```{r interactions, echo=TRUE}**

**# Create interaction terms**

**# Income and home value**

**bank$demog\_inc\_homeval <- bank$demog\_inc \* bank$demog\_homeval**

**# Age and home value**

**bank$demog\_age\_homeval <- bank$demog\_age \* bank$demog\_homeval**

**# Age and income**

**bank$demog\_age\_inc <- bank$demog\_age \* bank$demog\_inc**

**```**

**# Modelling by Target Variable**

**- `int\_tgt`**

**```{r data\_split, echo=TRUE}**

**options(h2o.max\_mem\_size = "8G")**

**#Import the dataset**

**bank\_h2o <- as.h2o(bank)**

**options(timeout = 300)**

**# Split the data into training, validation, and test sets**

**splits <- h2o.splitFrame(data = bank\_h2o, ratios = c(0.6, 0.2), seed = 123)**

**train <- splits[[1]]**

**valid <- splits[[2]]**

**test <- splits[[3]]**

**```**

**```{r int\_tgt, echo=TRUE}**

**# Set the response variable**

**response <- "log\_int\_tgt"**

**# Set the predictor variables**

**predictors\_int <- c(**

**"cat\_input1", "cat\_input2",**

**"log\_rfm1", "log\_rfm2", "log\_rfm3", "log\_rfm4", "rfm5", "rfm6",**

**"rfm7", "rfm8", "rfm9", "log\_rfm10", "rfm11", "rfm12",**

**"demog\_age", "demog\_genf", "demog\_ho",**

**"demog\_homeval", "demog\_inc", "demog\_pr","demog\_inc\_homeval",**

**"demog\_age\_homeval", "demog\_age\_inc"**

**)**

**```**

**```{r h2o\_auto\_int, echo=TRUE}**

**# Run AutoML with leaderboard on validation frame**

**automl <- h2o.automl(**

**x = predictors\_int,**

**y = "log\_int\_tgt",**

**training\_frame = train,**

**validation\_frame = valid,**

**leaderboard\_frame = valid,**

**max\_runtime\_secs = 300,**

**seed = 123,**

**stopping\_rounds = 5,**

**stopping\_metric = "RMSE",**

**nfolds = 0**

**)**

**# Convert leaderboard to data frame and get model types**

**leaderboard\_df <- as.data.frame(automl@leaderboard)**

**leaderboard\_df$model\_id <- as.character(leaderboard\_df$model\_id)**

**leaderboard\_df$model\_type <- sapply(leaderboard\_df$model\_id, function(id) {**

**model <- h2o.getModel(id)**

**model@algorithm**

**})**

**# Keep only the top 4 unique model types**

**leaderboard\_unique <- leaderboard\_df %>%**

**distinct(model\_type, .keep\_all = TRUE) %>%**

**slice(1:4) # just in case fewer than 4**

**# Get model IDs and types for unique models**

**top\_models <- leaderboard\_unique$model\_id**

**top\_types <- leaderboard\_unique$model\_type**

**# Extract models**

**models <- lapply(top\_models, h2o.getModel)**

**# --- VALIDATION METRICS (Leaderboard frame) ---**

**valid\_perfs <- lapply(models, function(model) h2o.performance(model, newdata = valid))**

**valid\_rmses <- sapply(valid\_perfs, h2o.rmse)**

**valid\_mses <- sapply(valid\_perfs, h2o.mse)**

**valid\_r2s <- sapply(valid\_perfs, h2o.r2)**

**# Convert logged RMSE to unlogged scale**

**valid\_unlogged\_rmses <- sqrt(exp(valid\_rmses^2) - 1)**

**validation\_summary <- data.frame(**

**Model = paste("Model", 1:length(top\_models)),**

**Type = top\_types,**

**RMSE = round(valid\_rmses, 4),**

**MSE = round(valid\_mses, 4),**

**Unlogged\_RMSE = round(valid\_unlogged\_rmses, 2),**

**R2 = round(valid\_r2s, 4)**

**)**

**# Display validation summary table**

**kable(validation\_summary, caption = "Validation Metrics: Top 4 Unique Model Types") %>%**

**kable\_styling(full\_width = FALSE, position = "left") %>%**

**column\_spec(1, bold = TRUE) %>%**

**column\_spec(2, color = "purple") %>%**

**column\_spec(3, color = "blue") %>%**

**column\_spec(4, color = "red") %>%**

**column\_spec(5, color = "darkgreen")**

**# --- TEST METRICS (Only for the top model on leaderboard) ---**

**final\_model <- models[[1]] # best model based on validation leaderboard**

**test\_perf <- h2o.performance(final\_model, newdata = test)**

**test\_rmse <- h2o.rmse(test\_perf)**

**test\_mse <- h2o.mse(test\_perf)**

**test\_r2 <- h2o.r2(test\_perf)**

**test\_unlogged\_rmse <- sqrt(exp(test\_rmse^2) - 1)**

**test\_summary <- data.frame(**

**Model = "Final Model (Model 1)",**

**Type = final\_model@algorithm,**

**RMSE = round(test\_rmse, 4),**

**MSE = round(test\_mse, 4),**

**Unlogged\_RMSE = round(test\_unlogged\_rmse, 2),**

**R2 = round(test\_r2, 4)**

**)**

**# Display test summary table**

**kable(test\_summary, caption = "Test Metrics: Final Selected Model") %>%**

**kable\_styling(full\_width = FALSE, position = "left") %>%**

**column\_spec(1, bold = TRUE) %>%**

**column\_spec(2, color = "purple") %>%**

**column\_spec(3, color = "blue") %>%**

**column\_spec(4, color = "red") %>%**

**column\_spec(5, color = "darkgreen")**

**```**

**```{r int\_tgt\_plot, echo=TRUE}**

**# Variable importance for the final best model (models[[1]])**

**varimp\_df <- as.data.frame(h2o.varimp(models[[1]]))**

**ggplot(varimp\_df[1:20, ], aes(x = reorder(variable, -relative\_importance), y = relative\_importance)) +**

**geom\_bar(stat = "identity", fill = "steelblue") +**

**coord\_flip() +**

**labs(title = "Top 20 Variable Importances (Final Best Model)",**

**x = "Variable",**

**y = "Relative Importance") +**

**theme\_minimal()**

**# Predictions on test set with final best model**

**best\_model <- models[[1]]**

**predictions <- h2o.predict(best\_model, newdata = test)**

**predictions\_df <- as.data.frame(predictions)**

**# Actual values (log-transformed) from test set**

**actuals\_log <- as.data.frame(test[["log\_int\_tgt"]])**

**actuals <- exp(actuals\_log[[1]]) # back-transform**

**# Predicted values back-transformed**

**predicted <- exp(predictions\_df[[1]])**

**# Actual vs Predicted dataframe**

**pred\_vs\_actual\_df <- data.frame(**

**Actual = actuals,**

**Predicted = predicted**

**)**

**ggplot(pred\_vs\_actual\_df, aes(x = Actual, y = Predicted)) +**

**geom\_point(color = "blue") +**

**geom\_abline(slope = 1, intercept = 0, color = "red") + # perfect prediction line**

**labs(title = "Predicted vs Actual Values (Final Best Model)",**

**x = "Actual Values",**

**y = "Predicted Values") +**

**theme\_minimal() +**

**theme(text = element\_text(size = 12))**

**# GLM coefficient table only if the 4th model is a GLM**

**model\_4 <- models[[4]] # 4th model from the unique top 4 list**

**if (model\_4@algorithm == "glm") {**

**coef\_table <- model\_4@model$coefficients\_table**

**coef\_df <- coef\_table[, c("names", "coefficients")]**

**coef\_df$Odds\_Ratio\_Percent <- round((exp(coef\_df$coefficients) - 1) \* 100, 2)**

**colnames(coef\_df) <- c("Predictor", "Coefficient", "Odds Ratio (%)")**

**kable(coef\_df, caption = "GLM Coefficients and Odds Ratios (%) for 4th Model") %>%**

**kable\_styling(full\_width = FALSE, position = "left") %>%**

**column\_spec(1, bold = TRUE) %>%**

**column\_spec(2, color = "blue") %>%**

**column\_spec(3, color = "red")**

**} else {**

**cat("The 4th model is not a GLM. It is a", model\_4@algorithm, "model.\n")**

**}**

**```**

**- `b\_tgt`**

**```{r b\_tgt, echo=TRUE}**

**# Set the response variable**

**response\_b <- "b\_tgt"**

**# Set the predictor variables**

**predictors\_b <- c(**

**"cat\_input1", "cat\_input2",**

**"log\_rfm1", "log\_rfm2", "log\_rfm3", "log\_rfm4", "rfm5", "rfm6",**

**"rfm7", "rfm8", "rfm9", "log\_rfm10", "rfm11", "rfm12",**

**"demog\_age", "demog\_genf", "demog\_ho",**

**"demog\_homeval", "demog\_inc", "demog\_pr","demog\_inc\_homeval",**

**"demog\_age\_homeval", "demog\_age\_inc"**

**)**

**```**

**```{r h2o\_auto\_b, echo=TRUE}**

**# Run AutoML with validation leaderboard frame**

**automl\_b <- h2o.automl(**

**x = predictors\_b,**

**y = response\_b,**

**training\_frame = train,**

**validation\_frame = valid,**

**leaderboard\_frame = valid, # validation used for leaderboard**

**max\_runtime\_secs = 300,**

**seed = 123,**

**stopping\_rounds = 5,**

**stopping\_metric = "AUC",**

**nfolds = 0**

**)**

**# Convert leaderboard to data frame and get model types**

**leaderboard\_df\_b <- as.data.frame(automl\_b@leaderboard)**

**leaderboard\_df\_b$model\_id <- as.character(leaderboard\_df\_b$model\_id)**

**leaderboard\_df\_b$model\_type <- sapply(leaderboard\_df\_b$model\_id, function(id) {**

**model <- h2o.getModel(id)**

**model@algorithm**

**})**

**# Keep only the top 4 unique model types (from validation leaderboard)**

**leaderboard\_unique\_b <- leaderboard\_df\_b %>%**

**distinct(model\_type, .keep\_all = TRUE) %>%**

**slice(1:4)**

**top\_models\_b <- leaderboard\_unique\_b$model\_id**

**top\_types\_b <- leaderboard\_unique\_b$model\_type**

**# Extract the top 4 models**

**models\_b <- lapply(top\_models\_b, h2o.getModel)**

**# Validation AUC for top 4 models (from leaderboard)**

**validation\_performances\_b <- lapply(models\_b, function(model) h2o.performance(model, newdata = valid))**

**validation\_auc\_b <- sapply(validation\_performances\_b, h2o.auc)**

**# Build validation summary table**

**validation\_summary\_b <- data.frame(**

**Model = paste("Model", 1:length(top\_models\_b)),**

**Type = top\_types\_b,**

**Validation\_AUC = round(validation\_auc\_b, 4)**

**)**

**# Display validation summary table**

**kable(validation\_summary\_b, caption = "Top 4 Unique Model Types: Validation Performance Summary for b\_tgt") %>%**

**kable\_styling(full\_width = FALSE, position = "left") %>%**

**column\_spec(1, bold = TRUE) %>%**

**column\_spec(2, color = "purple") %>%**

**column\_spec(3, color = "blue")**

**# Now evaluate the best model (models\_b[[1]]) on the test set for final performance**

**test\_performance\_b <- h2o.performance(models\_b[[1]], newdata = test)**

**test\_auc\_b <- h2o.auc(test\_performance\_b)**

**# Build test summary table for final chosen model**

**test\_summary\_b <- data.frame(**

**Model = "Best Model (Model 1)",**

**Type = top\_types\_b[1],**

**Test\_AUC = round(test\_auc\_b, 4)**

**)**

**# Display test performance summary**

**kable(test\_summary\_b, caption = "Final Best Model Test Performance for b\_tgt") %>%**

**kable\_styling(full\_width = FALSE, position = "left") %>%**

**column\_spec(1, bold = TRUE) %>%**

**column\_spec(2, color = "purple") %>%**

**column\_spec(3, color = "blue")**

**# Save the final best model**

**h2o.saveModel(models\_b[[1]], path = "F:/ECON/562\_Analytics\_2/Final Project/models/model\_1\_b\_tgt", force = TRUE)**

**```**

**```{r b\_tgt\_plot, echo=TRUE}**

**# === Variable Importance for Best Model ===**

**varimp\_df\_b <- as.data.frame(h2o.varimp(models\_b[[1]]))**

**# Plot Top 20 Variable Importances**

**ggplot(varimp\_df\_b[1:20, ], aes(x = reorder(variable, -relative\_importance), y = relative\_importance)) +**

**geom\_bar(stat = "identity", fill = "steelblue") +**

**coord\_flip() +**

**labs(title = paste("Top 20 Variable Importances (", top\_types\_b[1], ")", sep = ""),**

**x = "Variable",**

**y = "Relative Importance") +**

**theme\_minimal()**

**# === ROC / AUC Curves for Best Model ===**

**perf\_train <- h2o.performance(models\_b[[1]], newdata = train)**

**perf\_valid <- h2o.performance(models\_b[[1]], newdata = valid)**

**perf\_test <- h2o.performance(models\_b[[1]], newdata = test)**

**# Extract TPR and FPR**

**roc\_train <- as.data.frame(h2o.tpr(perf\_train))**

**roc\_train$fpr <- as.data.frame(h2o.fpr(perf\_train))$fpr**

**roc\_valid <- as.data.frame(h2o.tpr(perf\_valid))**

**roc\_valid$fpr <- as.data.frame(h2o.fpr(perf\_valid))$fpr**

**roc\_test <- as.data.frame(h2o.tpr(perf\_test))**

**roc\_test$fpr <- as.data.frame(h2o.fpr(perf\_test))$fpr**

**# Plot**

**plot(roc\_train$fpr, roc\_train$tpr, type = "l", col = "green", lwd = 2,**

**xlab = "False Positive Rate", ylab = "True Positive Rate",**

**main = "ROC Curves for Best Model")**

**lines(roc\_valid$fpr, roc\_valid$tpr, col = "blue", lwd = 2)**

**lines(roc\_test$fpr, roc\_test$tpr, col = "red", lwd = 2)**

**abline(a = 0, b = 1, lty = 2, col = "gray") # Diagonal reference**

**# Add legend**

**legend("bottomright", legend = c("Train", "Validation", "Test"),**

**col = c("green", "blue", "red"), lwd = 2, bty = "n")**

**# === GLM Coefficient Table (4th Model) ===**

**model\_4 <- h2o.getModel(top\_models\_b[4])**

**if (model\_4@algorithm == "glm") {**

**coef\_table <- model\_4@model$coefficients\_table**

**coef\_df <- coef\_table[, c("names", "coefficients")]**

**coef\_df$`Odds Ratio (%)` <- round((exp(coef\_df$coefficients) - 1) \* 100, 2)**

**colnames(coef\_df) <- c("Predictor", "Coefficient", "Odds Ratio (%)")**

**kable(coef\_df, caption = "GLM Coefficients and Odds Ratios (%)") %>%**

**kable\_styling(full\_width = FALSE, position = "left") %>%**

**column\_spec(1, bold = TRUE) %>%**

**column\_spec(2, color = "blue") %>%**

**column\_spec(3, color = "red")**

**} else {**

**cat("The 4th model is not a GLM. It is a", model\_4@algorithm, "model.\n")**

**}**

**```**

**```{r confusion\_matrix, echo=TRUE}**

**# Get the best model (first in the list)**

**best\_model <- models\_b[[1]]**

**best\_model\_type <- top\_types\_b[1]**

**# Generate confusion matrix on test set**

**best\_model\_test\_cm <- as.data.frame(h2o.confusionMatrix(best\_model, newdata = test))**

**# Display as a styled table**

**library(knitr)**

**library(kableExtra)**

**kable(best\_model\_test\_cm, caption = paste("Confusion Matrix on Test Set -", best\_model\_type, "(Best Model)")) %>%**

**kable\_styling(full\_width = FALSE, position = "left") %>%**

**column\_spec(1, bold = TRUE) %>%**

**column\_spec(2, color = "red") %>%**

**column\_spec(3, color = "green") %>%**

**column\_spec(4, color = "blue") %>%**

**row\_spec(0, bold = TRUE, background = "#f2f2f2")**

**```**

**- `cnt\_tgt`**

**```{r cnt\_tgt, echo=TRUE}**

**# Set the response variable**

**response\_cnt <- "cnt\_tgt"**

**# Set the predictor variables**

**predictors\_cnt <- c(**

**"cat\_input1", "cat\_input2",**

**"log\_rfm1", "log\_rfm2", "log\_rfm3", "log\_rfm4", "rfm5", "rfm6",**

**"rfm7", "rfm8", "rfm9", "log\_rfm10", "rfm11", "rfm12",**

**"demog\_age", "demog\_genf", "demog\_ho",**

**"demog\_homeval", "demog\_inc", "demog\_pr","demog\_inc\_homeval",**

**"demog\_age\_homeval", "demog\_age\_inc"**

**)**

**```**

**```{r h2o\_auto\_cnt, echo=TRUE}**

**# Run AutoML for count target (Poisson regression setup)**

**automl\_cnt <- h2o.automl(**

**x = predictors\_cnt,**

**y = response\_cnt,**

**training\_frame = train,**

**validation\_frame = valid,**

**leaderboard\_frame = valid, # validation used for leaderboard**

**max\_runtime\_secs = 300,**

**seed = 123,**

**stopping\_rounds = 5,**

**stopping\_metric = "RMSE",**

**nfolds = 0**

**)**

**# Convert leaderboard to data frame and get model types**

**leaderboard\_df\_cnt <- as.data.frame(automl\_cnt@leaderboard)**

**leaderboard\_df\_cnt$model\_id <- as.character(leaderboard\_df\_cnt$model\_id)**

**leaderboard\_df\_cnt$model\_type <- sapply(leaderboard\_df\_cnt$model\_id, function(id) {**

**model <- h2o.getModel(id)**

**model@algorithm**

**})**

**# Keep only the top 4 unique model types**

**leaderboard\_unique\_cnt <- leaderboard\_df\_cnt %>%**

**distinct(model\_type, .keep\_all = TRUE) %>%**

**slice(1:4)**

**# Get model IDs and types**

**top\_models\_cnt <- leaderboard\_unique\_cnt$model\_id**

**top\_types\_cnt <- leaderboard\_unique\_cnt$model\_type**

**# Extract top models**

**models\_cnt <- lapply(top\_models\_cnt, h2o.getModel)**

**# Evaluate on validation (already used in leaderboard)**

**performances\_cnt\_valid <- lapply(models\_cnt, function(model) h2o.performance(model, newdata = valid))**

**rmses\_cnt\_valid <- sapply(performances\_cnt\_valid, h2o.rmse)**

**mses\_cnt\_valid <- sapply(performances\_cnt\_valid, h2o.mse)**

**# Build validation summary table**

**summary\_table\_cnt <- data.frame(**

**Model = paste("Model", 1:length(top\_models\_cnt)),**

**Type = top\_types\_cnt,**

**RMSE = round(rmses\_cnt\_valid, 4),**

**MSE = round(mses\_cnt\_valid, 4)**

**)**

**# Display summary table for validation set**

**kable(summary\_table\_cnt, caption = "Top 4 Unique Model Types: Validation Performance Summary for cnt\_tgt") %>%**

**kable\_styling(full\_width = FALSE, position = "left") %>%**

**column\_spec(1, bold = TRUE) %>%**

**column\_spec(2, color = "purple") %>%**

**column\_spec(3, color = "blue") %>%**

**column\_spec(4, color = "red")**

**# Evaluate only the best model on the test set**

**best\_model\_cnt <- models\_cnt[[1]]**

**perf\_test\_cnt <- h2o.performance(best\_model\_cnt, newdata = test)**

**rmse\_test\_cnt <- h2o.rmse(perf\_test\_cnt)**

**mse\_test\_cnt <- h2o.mse(perf\_test\_cnt)**

**# Display separate table for best model test performance**

**data.frame(**

**Model = "Best Model (Test Set)",**

**Type = top\_types\_cnt[1],**

**RMSE = round(rmse\_test\_cnt, 4),**

**MSE = round(mse\_test\_cnt, 4)**

**) %>%**

**kable(caption = "Best Model Performance on Test Set (cnt\_tgt)") %>%**

**kable\_styling(full\_width = FALSE, position = "left") %>%**

**column\_spec(1, bold = TRUE, color = "darkgreen") %>%**

**column\_spec(2, color = "purple") %>%**

**column\_spec(3, color = "blue") %>%**

**column\_spec(4, color = "red")**

**# Save the best model**

**h2o.saveModel(best\_model\_cnt, path = "F:/ECON/562\_Analytics\_2/Final Project/models/model\_1\_cnt\_tgt", force = TRUE)**

**```**

**```{r cnt\_tgt\_plot, echo=TRUE}**

**# Get variable importance for the best model**

**varimp\_df\_cnt <- as.data.frame(h2o.varimp(best\_model\_cnt))**

**# Plot top 20 variable importances using ggplot2**

**ggplot(varimp\_df\_cnt[1:20, ], aes(x = reorder(variable, -relative\_importance), y = relative\_importance)) +**

**geom\_bar(stat = "identity", fill = "steelblue") +**

**coord\_flip() +**

**labs(title = "Top 20 Variable Importances (Best Model - cnt\_tgt)",**

**x = "Variable",**

**y = "Relative Importance") +**

**theme\_minimal()**

**# Predicted vs Actual plot for the best model**

**predicted\_cnt <- h2o.predict(best\_model\_cnt, newdata = test)**

**predicted\_cnt\_df <- as.data.frame(predicted\_cnt)**

**predicted\_cnt\_df$actual <- as.data.frame(test)$cnt\_tgt**

**predicted\_cnt\_df$actual <- as.numeric(predicted\_cnt\_df$actual)**

**predicted\_cnt\_df$predicted <- as.numeric(predicted\_cnt\_df$predict)**

**ggplot(predicted\_cnt\_df, aes(x = actual, y = predicted)) +**

**geom\_point(alpha = 0.5, color = "darkorange") +**

**geom\_smooth(method = "lm", color = "blue") +**

**labs(title = "Predicted vs Actual for cnt\_tgt (Best Model)",**

**x = "Actual cnt\_tgt",**

**y = "Predicted cnt\_tgt") +**

**theme\_minimal()**

**# Inspect GLM coefficients if the 4th model is GLM**

**model\_4\_cnt <- h2o.getModel(top\_models\_cnt[4])**

**if (model\_4\_cnt@algorithm == "glm") {**

**# Extract the full coefficient table**

**coef\_table\_cnt <- model\_4\_cnt@model$coefficients\_table**

**# Keep relevant columns and calculate odds ratio percentage**

**coef\_df\_cnt <- coef\_table\_cnt[, c("names", "coefficients")]**

**coef\_df\_cnt$Odds\_Ratio\_Percent <- round((exp(coef\_df\_cnt$coefficients) - 1) \* 100, 2)**

**# Rename columns for clarity**

**colnames(coef\_df\_cnt) <- c("Predictor", "Coefficient", "Odds Ratio (%)")**

**# Display table**

**kable(coef\_df\_cnt, caption = "GLM Coefficients and Odds Ratios (%) for cnt\_tgt") %>%**

**kable\_styling(full\_width = FALSE, position = "left") %>%**

**column\_spec(1, bold = TRUE) %>%**

**column\_spec(2, color = "blue") %>%**

**column\_spec(3, color = "red")**

**} else {**

**cat("The 4th model is not a GLM. It is a", model\_4\_cnt@algorithm, "model.\n")**

**}**

**```**