Part-I Data Science Concepts & Methods

Task 1 Data Science/Machine Learning methods.

Decision Trees

Decision Trees are a supervised learning method employed for both classification and regression tasks. They are known for their intuitive structure, which mimics human decision-making processes (Loh, 2011).

Main Components:

- **Root Node:** This is where data splitting begins, representing the entire dataset.
- **Internal Nodes:** These nodes test specific attributes, dividing the data into branches based on outcomes. Each split aims to increase homogeneity within the resultant branches.
- **Branches:** Represent the outcomes of tests at internal nodes, partitioning the dataset into subgroups.
- **Leaf Nodes:** These terminal nodes provide the decision or classification outcome, associated with a class label or continuous value (Quinlan, 1986).

How It Works: Decision Trees recursively partition the data into subsets based on feature values, forming a tree-like model. This process continues until a stopping criterion is met, such as a minimum number of samples in a node or maximum tree depth.

Advantages:

- **Interpretability:** Easy to understand and interpret, even for non-experts.
- **Flexibility:** Can handle both categorical and numerical data.
- **Handling Missing Values:** Capable of dealing with missing data points without requiring imputation.

Limitations:

- **Overfitting:** Prone to overfitting, particularly with noisy data or complex trees.
- **High-Dimensional Data:** Performance may degrade with high-dimensional data.
- **Instability:** Small variations in the data can result in different tree structures.

Application in Finance: In finance, Decision Trees are widely used for credit scoring, risk assessment, and fraud detection. For instance, a financial institution might use a Decision Tree to evaluate loan applicants' creditworthiness. The root node could assess the applicant’s credit score, with internal nodes evaluating employment status and debt-to-income ratio. This hierarchical structure helps in categorizing applicants into different risk levels, making the decision process transparent and systematic (Hand & Henley, 1997).

Despite their utility, Decision Trees' tendency to overfit can be problematic. Techniques like pruning, setting a maximum depth, or using ensemble methods like Random Forests can mitigate overfitting and improve model generalization (Breiman, 2001).

Critique of Explanation: The explanation provided by ChatGPT is clear and covers the basic structure and function of Decision Trees. However, it could be improved by including more on practical implementation aspects, such as pruning techniques and the use of ensemble methods to counteract overfitting. Additionally, discussing real-world applications in finance provides context, but referencing specific studies or recent examples would strengthen the argument.

Recent Examples: A recent example is the use of PCA in factor investing, where it helps identify the most significant factors driving stock returns. Researchers analyzed the US stock market and extracted common factors such as size and value, which were then used to construct diversified portfolios that outperformed the market (Fan, J., & Yao, Q., 2017). This demonstrates PCA's ability to distill complex market data into actionable investment strategies.

Popularity: Despite their limitations, Decision Trees remain popular due to their simplicity and ease of interpretation. Their popularity has been bolstered by the development of ensemble methods like Random Forests and Gradient Boosting Machines, which mitigate some of the inherent weaknesses of single Decision Trees.

Principal Component Analysis

Principal Component Analysis (PCA) is an unsupervised dimension reduction technique used to simplify complex datasets. It transforms a set of possibly correlated variables into a smaller set of uncorrelated variables called principal components.

Main Components:

- **Eigenvectors:** Directions of the new coordinate system.
- **Eigenvalues:** Amount of variance explained by each eigenvector.

How It Works: PCA projects the data onto a new coordinate system, retaining the most important features and reducing dimensionality. The process involves several key steps:

1. **Standardization:** Standardizing the range of the initial variables so each contributes equally to the analysis.
2. **Covariance Matrix Computation:** Expressing the correlation between pairs of variables in the dataset.
3. **Eigenvalue and Eigenvector Calculation:** Identifying the directions (principal components) that maximize the variance in the data by calculating the eigenvalues and eigenvectors of the covariance matrix.
4. **Component Selection:** Ordering principal components by the proportion of the total variance they explain and selecting a subset for further analysis.

Advantages:

- **Noise Reduction:** PCA reduces noise in the data by focusing on the most significant components.
- **Improved Visualization:** It simplifies complex datasets, making them easier to visualize.
- **Computational Efficiency:** Reduces the number of variables, speeding up computations.

Limitations:

- **Sensitivity to Scale:** PCA is sensitive to the scale of the variables, requiring standardization.
- **Linearity Assumption:** It assumes linear relationships between variables, which may not hold in all datasets.
- **Interpretability:** The principal components can be difficult to interpret as they are linear combinations of the original variables.

Application in Finance: In finance, PCA is used to reduce the complexity of financial data while retaining influential variables. A common application is in stock returns analysis. Financial markets generate vast amounts of data, with intricate relationships between different asset classes or stocks. PCA distills these relationships into principal components that might represent common factors influencing returns, such as market, sectoral, or macroeconomic influences (Connor & Korajczyk, 1993).

For instance, PCA can be employed in portfolio construction. By identifying principal components, analysts understand which asset combinations move together and how they contribute to the portfolio's risk and return. This is useful in risk management, where PCA highlights the principal sources of variance in a portfolio, aiding in the creation of more resilient investment strategies (Jolliffe, 2002).

Recent Examples: A recent example is the use of PCA in factor investing, where it helps identify the most significant factors driving stock returns. Analysts use PCA to extract factors like value, momentum, and size, which are then used to construct and manage investment portfolios. This approach enhances portfolio diversification and risk management.

Popularity: PCA remains popular in finance due to its effectiveness in reducing data complexity and identifying underlying patterns. Its ability to preprocess data for other machine learning models and simplify data visualization makes it a valuable tool for financial analysts and portfolio managers.

Critique of Explanation: The explanation provided by ChatGPT is clear and covers the basic structure and function of PCA. However, it could be improved by including more on practical implementation aspects, such as the need for data standardization and how to interpret the principal components. Additionally, discussing real-world applications in finance provides context, but referencing specific studies or recent examples would strengthen the argument.

Text Mining

Text mining, also known as text analytics, transforms unstructured text into actionable information using NLP, statistics, and machine learning. This method is valuable in finance, where volumes of text like news articles, reports, and social media can impact markets and strategies (Feldman & Sanger, 2007).

Primary components:

- **Tokenization:** Breaking text into individual words or phrases.
- **Stopword Removal:** Removing common words like "the", "and", etc., to focus on more meaningful words.
- **Stemming or Lemmatization:** Reducing words to their base form to unify different forms of the same word.

How It Works: Text Mining uses various techniques to analyze and extract information from text data:

- **Sentiment Analysis:** Determines the sentiment expressed in the text, whether positive, negative, or neutral. This can be used to gauge public sentiment towards a company or financial market.
- **Topic Modeling:** Identifies topics within a set of documents, helping to categorize and understand large volumes of text data.
- **Named Entity Recognition (NER):** Identifies and classifies named entities (such as names of people, organizations, locations) within the text, providing structured information from unstructured data.

Advantages:

- **Extracts Valuable Information:** Efficiently processes unstructured data to find useful information, converting text data into actionable insights.
- **Handles Large Volumes of Text:** Capable of managing and analyzing vast amounts of textual data, making it useful for real-time analysis.

Limitations:

- **Domain-Specific Knowledge Required:** Often needs specialized understanding of the domain to accurately interpret the results.

- **Sensitivity to Language and Syntax:** Can be affected by nuances in language and syntax, such as sarcasm, slang, and context.

Application in Finance: Text mining, also known as text analytics, transforms unstructured text into actionable information using Natural Language Processing (NLP), statistics, and machine learning. This method is valuable in finance, where volumes of text like news articles, reports, and social media can impact markets and strategies.

Key Areas in Finance:

- **Market Sentiment Analysis:** Analysts use text mining to gauge public sentiment towards stocks or the market by analyzing news articles and social media posts. For example, positive or negative sentiments expressed in financial news can affect stock prices, and text mining helps predict these movements (Tetlock, 2007).
- **Regulatory Compliance Monitoring:** Financial institutions apply text mining to ensure compliance with regulations by analyzing regulatory documents and internal communications. This helps identify necessary adjustments in practices quickly and efficiently (Goltz & Mayo, 2017).
- **Fraud Detection:** By analyzing textual communication like emails or transaction documents, text mining helps detect potential fraud. It identifies unusual patterns or anomalies that may indicate fraudulent activities, thereby enhancing the institution's ability to prevent fraud (Ngai et al., 2011).

Recent Examples: A recent example is the use of text mining in analyzing social media platforms like Twitter to monitor real-time market sentiment. For instance, text mining techniques were applied to Twitter feeds to predict stock market movements based on public sentiment towards particular companies. This approach led to more informed trading decisions, demonstrating text mining's ability to provide actionable insights from real-time data (Bollen, Mao & Zeng, 2011).

Popularity: Text mining has gained significant popularity in the financial sector due to its ability to process and analyze vast amounts of unstructured data. Its applications in market sentiment analysis, compliance monitoring, and fraud detection have made it an indispensable tool for financial analysts and institutions. The increasing availability of large datasets and advancements in NLP technologies further contribute to its widespread adoption. As the financial industry continues to recognize the value of extracting insights from textual data, the popularity of text mining is expected to grow.

Critique of Explanation: The explanation provided by ChatGPT is thorough and covers the basic structure and function of text mining. However, it could be enhanced by including more practical examples and discussing the challenges associated with processing different types of textual data. For instance, while sentiment analysis can be powerful, it may struggle with understanding context-specific nuances such as sarcasm or idiomatic expressions. Additionally, the explanation can benefit from discussing the computational challenges involved in processing large volumes of text data and the sophisticated algorithms required to achieve accurate results.

Task 2 Big Data

Critique of the ChatGPT Response

Strengths

- **Clear Definition and Characteristics**: The response succinctly defines Big Data and explains its five key characteristics: Volume, Velocity, Variety, Veracity, and Value. These are well-accepted aspects in both academic and industry contexts.
- **Relevant Use Cases**: The examples of Big Data applications in finance, such as risk management and fraud detection, are pertinent and illustrate practical uses.

Weaknesses

- **Superficial Analysis**: The critical evaluation section lacks depth. It mentions challenges but does not provide detailed analysis or specific examples.

- **Lack of Specificity**: The described use cases are general. Including specific case studies or real-world examples would enhance the discussion.
- **Limited Ethical and Privacy Discussion**: The response briefly mentions data privacy and ethical considerations but lacks comprehensive analysis. More detailed discussions with examples would strengthen the evaluation.

Critical Evaluation of Data Privacy Issues

Data privacy is a significant concern in Big Data, especially in finance, where sensitive financial data is handled. The ChatGPT response identifies data privacy as an issue but lacks depth.

Australian Government's Approach to Data Privacy

The Australian Government addresses data privacy through several key measures:

- **Privacy Act 1988**: This act regulates the handling of personal information by government agencies and private sector organizations. The Australian Privacy Principles (APPs) within the act set standards for data collection, use, disclosure, and security (Office of the Australian Information Commissioner, n.d.).
- **Notifiable Data Breaches (NDB) Scheme**: Introduced in February 2018, the NDB scheme requires organizations to notify affected individuals and the Office of the Australian Information Commissioner (OAIC) about data breaches likely to result in serious harm (Office of the Australian Information Commissioner, n.d.).
- **Consumer Data Right (CDR)**: CDR gives consumers greater control over their data, allowing them to access and transfer their data between service providers, enhancing transparency and competition while maintaining strict privacy protections (Australian Government, n.d.).

Despite these measures, ongoing challenges include ensuring compliance and protecting against sophisticated cyber threats, given the increasing volume and complexity of data.

Critical Evaluation of Ethical Considerations

Ethical considerations are crucial in Big Data analytics to ensure fairness, transparency, and accountability. The ChatGPT response mentions avoiding biases and ensuring transparency but lacks detailed evaluation.

Ethical Considerations

- **Algorithmic Bias**: Algorithms can unintentionally perpetuate biases present in data, leading to unfair outcomes. Regular audits and refinements of algorithms are essential to mitigate these biases (Australian Human Rights Commission, 2019).
- **Transparency and Accountability**: Financial institutions must ensure their use of Big Data analytics is transparent and that they are accountable for decisions based on these analyses (Australian Human Rights Commission, 2019).
- **Informed Consent**: Obtaining informed consent from individuals whose data is used is critical. This includes clearly communicating data use and ensuring individuals understand and agree to these uses (Australian Human Rights Commission, 2019).

Australian Government's Approach to Ethical Considerations

The Australian Government addresses ethical considerations through various initiatives:

- **Ethical Frameworks and Guidelines**: The "AI Ethics Framework" provides principles for the ethical development and use of AI, closely related to Big Data analytics (Australian Government, 2021).
- **Regulatory Oversight**: Bodies like the OAIC ensure organizations adhere to ethical standards in data practices (Office of the Australian Information Commissioner, n.d.).
- **Public Consultations and Reports**: The government engages in public consultations and publishes reports on ethical considerations in technology and data use, aiming to gather diverse perspectives and promote best practices across industries (Australian Government, 2021).

Task 3 Classifying Credit Risk

The box plot

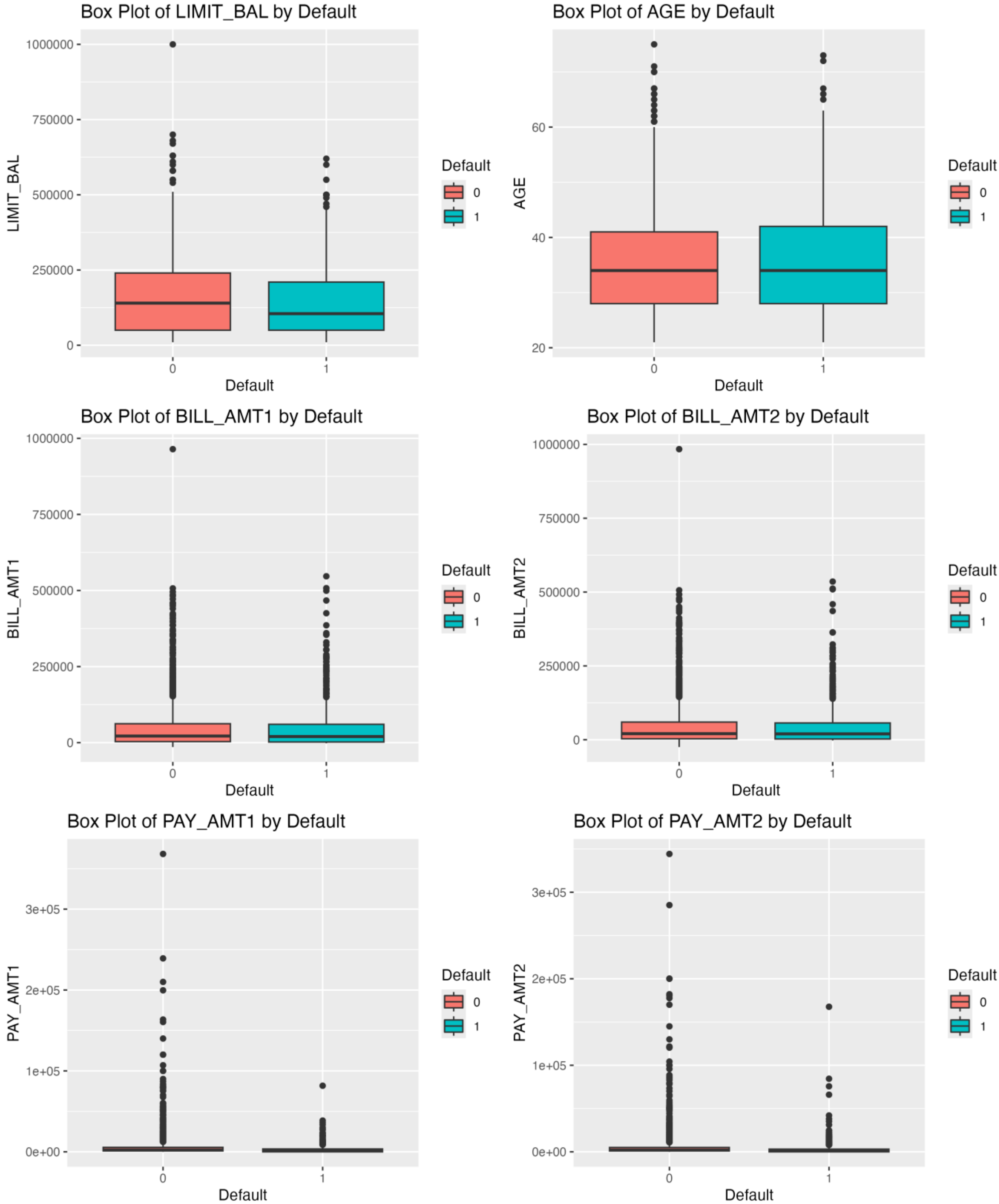

Figure 1: Box Plots for Continuous Variables by Default Status

Purpose

Box plots are used to visualize the distribution of continuous variables. They provide a summary of key statistics such as the median, quartiles, and potential outliers. In this context, the box plots help compare the distributions of various financial metrics between those who defaulted on their credit card payments (default = 1) and those who did not (default = 0).

Techniques Used

1. **Data Cleaning**: We used **na.omit(data)** to remove any rows with **NA** values from the dataset. This ensures that our analysis is not skewed by missing data.

2. **Box Plot Creation with ggplot2**:

   - We used the **ggplot2** package to create box plots for continuous variables.

   - The **geom_boxplot()** function is used to generate the box plot.

   - The **aes()** function specifies the aesthetics, such as the x-axis (default status), y-axis (continuous variable), and fill color (default status).

   - The **labs()** function is used to label the axes and title.

   - The **scale_fill_discrete()** function is used to ensure the fill color corresponds to the default values.

3. **Plot Arrangement**:

   - We used the **gridExtra** package to arrange multiple plots in a grid layout.

   - The **grid.arrange()** function is used to arrange the plots in a 2x3 grid.

   - The **ggsave()** function saves the combined plot as an image file.

Assumptions

1. **Continuity of Data**: The continuous variables (e.g., **LIMIT_BAL**, **AGE**, **BILL_AMT1**, etc.) are assumed to be measured on a continuous scale. This allows for the computation of quartiles and the identification of outliers.

2. **Independence of Observations**: Each observation (i.e., each credit card client's data) is assumed to be independent of others. This is important for the validity of the statistical summaries represented in the box plots.

3. **Handling of Outliers**: Box plots inherently show outliers as points beyond the whiskers. It is assumed that these outliers are genuine data points and not errors. Outliers are included to give a complete picture of the data distribution.

4. **Categorical Representation of Default Status**: The default status (**default**) is treated as a categorical variable with two levels (0 and 1). The box plots are filled and separated based on these levels to facilitate comparison.

The bar chart

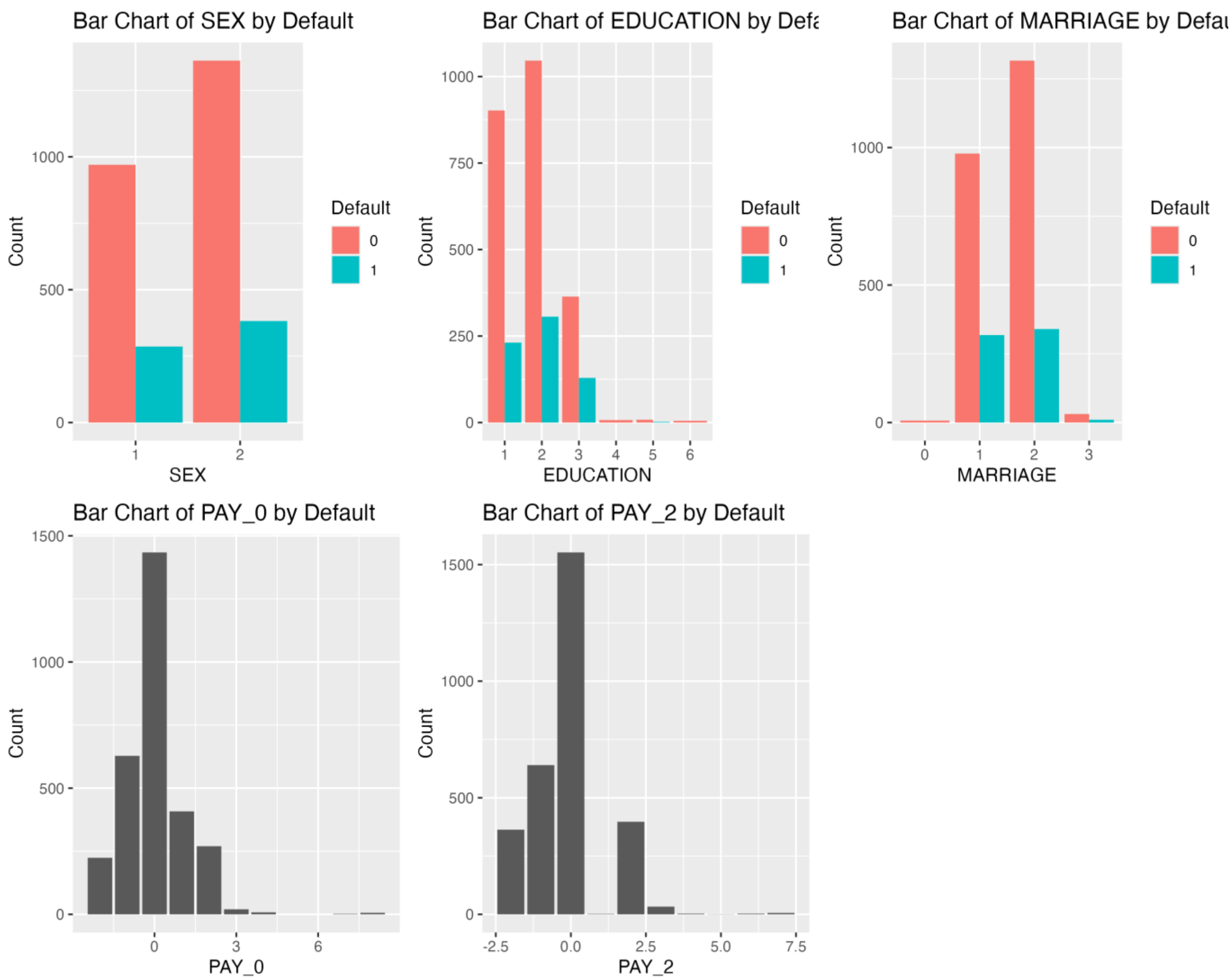

Figure 2: Bar Charts for Categorical Variables by Default Status

Purpose

Bar charts are used to visualize the distribution of categorical variables. They provide a clear comparison of counts across different categories. In this context, the bar charts help compare the distribution of demographic factors (SEX, EDUCATION, MARRIAGE) and repayment status (PAY_0, PAY_2) between those who defaulted on their credit card payments (default = 1) and those who did not (default = 0)..

Techniques Used

1. **Data Cleaning**: Similar to the box plots, **na.omit(data)** is used to remove any rows with **NA** values from the dataset to ensure the analysis is not affected by missing data.

2. **Bar Chart Creation with ggplot2**:

   - We used the **ggplot2** package to create bar charts for categorical variables.
   - The **geom_bar(position = "dodge")** function is used to generate the bar charts with bars for different default statuses placed side-by-side (dodge position).
   - The **aes()** function specifies the aesthetics, such as the x-axis (categorical variable), fill color (default status), and y-axis (count).
   - The **labs()** function is used to label the axes and title.
   - The **scale_fill_discrete()** function is used to ensure the fill color corresponds to the default values.

3. **Plot Arrangement**:
   - We used the **gridExtra** package to arrange multiple plots in a grid layout.
   - The **grid.arrange()** function is used to arrange the plots in a 2x2 grid.
   - The **ggsave()** function saves the combined plot as an image file.

Assumptions

1. **Categorical Data**: The variables **SEX, EDUCATION, MARRIAGE, PAY_0,** and **PAY_2** are treated as categorical variables with distinct levels. This allows for the counting and comparison of occurrences within each category.
2. **Independence of Observations**: Each observation (i.e., each credit card client's data) is assumed to be independent of others. This is important for the validity of the counts represented in the bar charts.
3. **Removed Missing Values**: By using **na.omit(data)**, we assume that the remaining data without **NA** values is representative of the overall dataset. This helps ensure that the analysis is not biased by missing data.

Discussion on the Variables

Descriptive Statistics

Continuous Variables:

Variable	Min	Mean	Median	Std Dev	Max
LIMIT_BAL	10000	164036.7	140000	128263.9	1,000,000
AGE	21	35.34	34	9.36	75
BILL_AMT1	-14386	50199.78	21253.5	75769.4	964511
BILL_AMT2	-24704	48144.65	20384.5	73539.03	983931
PAY_AMT1	0	5387.124	2091	14354.51	368199
PAY_AMT2	0	5296.843	2000	15661.35	344261

Table 1: Descriptive Statistics for Continuous Variables

Categorical Variables:

Variable	Default = 0	Default = 1
SEX_1	970	286
SEX_2	1362	382
EDUCATION_1	902	231
EDUCATION_2	1046	306
EDUCATION_3	364	129

EDUCATION_4	7	0
EDUCATION_5	8	2
EDUCATION_6	5	0
MARRIAGE_0	7	0
MARRIAGE_1	978	318
MARRIAGE_2	1316	340
MARRIAGE_3	31	10
PAY_0_0	1245	189
PAY_0_1	269	139
PAY_0_2	92	178
PAY_0_3	9	11
PAY_0_4	1	7
PAY_0_5	0	0
PAY_0_6	0	0
PAY_0_7	0	2
PAY_0_8	5	1
PAY_0_9	0	0
PAY_2_0	1299	253
PAY_2_1	202	195
PAY_2_2	11	22
PAY_2_3	0	3
PAY_2_4	1	0
PAY_2_5	0	3
PAY_2_6	5	1
PAY_2_7	0	0

PAY_2_8	0	0
PAY_2_9	2	3

Table 2: Descriptive Statistics for Categorical Variables

Comments on the Variables

Continuous Variables

1. **LIMIT_BAL**:
    - Both groups (default and non-default) have similar distributions in terms of credit limit.
    - The mean and median credit limits are slightly higher for non-defaulters.
    - There are outliers in both groups with very high credit limits, indicating some individuals have access to large amounts of credit.
2. **AGE**:
    - Age distribution is similar for both groups, with a median age around 34 years.
    - The standard deviation indicates a moderate spread around the mean age.
    - No significant difference in age between defaulters and non-defaulters.
3. **BILL_AMT1 and BILL_AMT2**:
    - The distributions of bill amounts for the two months are similar across both groups.
    - There are negative values, indicating possible corrections or refunds.
    - The mean and median bill amounts are slightly higher for non-defaulters.
4. **PAY_AMT1 and PAY_AMT2**:
    - Both groups show wide ranges of payment amounts.
    - There are high outliers, especially among non-defaulters, indicating some individuals make large payments.
    - The median payment amounts are low, showing that most clients make smaller payments.

Categorical Variables:

1. **SEX**:
    - More females (SEX_2) in the dataset.
    - Higher count of defaults among females, but the proportion of defaults is similar for both males and females.
2. **EDUCATION**:
    - Most clients have a university (EDUCATION_2) or graduate school (EDUCATION_1) education.
    - Higher counts of defaults among these education levels, but this aligns with the higher number of individuals in these categories.
3. **MARRIAGE**:
    - Majority of clients are single (MARRIAGE_2) or married (MARRIAGE_1).
    - More defaults among single clients compared to married ones, suggesting marital status might influence default rates.
4. **PAY_0**:
    - Most clients have a PAY_0 status of 0 (pay duly) for both defaulters and non-defaulters.

- Higher counts of default are observed for higher PAY_0 values, indicating longer payment delays are associated with higher default rates.

5. **PAY_2**:
   - Similar to PAY_0, most clients have a PAY_2 status of 0 (pay duly).
   - Higher counts of default are observed for higher PAY_2 values, indicating that clients with longer payment delays in August 2005 are more likely to default

Discussion on Variables

From the graphs and summary statistics, several observations can be made:

- **Credit Limit (LIMIT_BAL)** and **Age (AGE)** distributions do not show significant differences between the default and non-default groups.
- **Bill Amounts (BILL_AMT1 and BILL_AMT2)** have similar distributions across both groups, though higher values are observed among non-defaulters.
- **Payment Amounts (PAY_AMT1 and PAY_AMT2)** indicate that non-defaulters tend to make larger payments, reflected in the presence of higher outliers.
- **Gender (SEX)** shows that defaults are more frequent among females, but the proportion of defaults is not significantly different between genders.
- **Education Level (EDUCATION)** reveals that higher education levels (university and graduate school) have higher counts of defaults, possibly due to the larger representation of these groups in the dataset.
- **Marital Status (MARRIAGE)** suggests single individuals are more likely to default compared to married ones.
- **Repayment Status (PAY_0 and PAY_2)** indicates that longer delays in payments are associated with higher default rates. Most clients who pay duly (status 0) have lower default rates, while those with higher delays (status 1 and above) show higher default rates. This suggests that recent repayment behavior is a strong indicator of default risk, highlighting the importance of timely payments to avoid default

Task 4 Forecasting Stock Returns using Technical Indicators

Outcome Variable:

4.1.1 Daily Logarithmic Returns

- **Description**: Logarithmic returns, or log returns, are calculated as the natural logarithm of the ratio of today's closing price to yesterday's closing price.
- **Formula**: $\log\_\text{ret} = \log\left(\frac{P_{t-1}}{P_t}\right)$
- **Reason for Use**: Logarithmic returns are preferred in financial time series analysis because they are time-additive, making it easier to aggregate returns over multiple periods. They also account for compounding effects.

Predictor Variables:

4.1.2 One Period Lag of the Log Returns

- **Description**: The log return of the previous day.
- **Reason for Use**: Lagged log returns can capture the momentum effect in stock prices, where past returns might influence future returns.

4.1.3 One Period Lag of Moving Average: 5 day

- **Description**: The simple moving average (SMA) over the past 5 days, lagged by one period.
- **Formula**: $SMA_5 = \frac{P_{t-1}+P_{t-2}+P_{t-3}+P_{t-4}+P_{t-5}}{5}$
- **Reason for Use**: Moving averages smooth out short-term price fluctuations and help identify trends. The lagged SMA indicates the trend direction from the previous day.

4.1.4 One Period Lag of Exponential Moving Average: 5 day

- **Description**: The exponentially weighted moving average (EMA) over the past 5 days, lagged by one period.
- **Formula**: $EMA_5 = \left(\frac{2}{n+1}\right)(P_t - EMA_{t-1}) + EMA_{t-1} \quad \text{where} \quad n = 5$
- **Reason for Use**: The EMA gives more weight to recent prices, making it more responsive to new information. The lagged EMA helps in identifying recent trends.

4.1.5 One Period Lag of RSI with 5 Day Period

- **Description**: The Relative Strength Index (RSI) over the past 5 days, lagged by one period.
- **Formula**: RSI is calculated as$100 - \left(\frac{100}{1+RS}\right)$, where RS is the average of 'n' days' up closes divided by the average of 'n' days' down closes.
- **Reason for Use**: RSI measures the speed and change of price movements, helping to identify overbought or oversold conditions. The lagged RSI provides insight into past market momentum.

Sentiment Indicator

4.2 Sentiment Indicator

- **Description**: The daily aggregated sentiment score from Alexandria (ACTA) data, converted to an **xts** time series object.
- **Reason for Use**: Sentiment indicators can capture market emotions and investor behavior, which can influence stock prices. Lagging the sentiment indicator allows us to use past sentiment as a predictor for future stock movements.

Data Splitting

4.3 Training and Testing Sample:

- **Description**: The dataset is split into training and testing sets, with the last 100 days used for testing and the remaining data for training.
- **Reason for Use**: This split ensures that the model is trained on historical data and tested on recent, unseen data, allowing for an evaluation of its predictive performance.

Date	**log_returns**	**ma_5**	**ema_5**	**rsi_5**	**sentiment**
2021-01-12	-0.004280781	90.114	90.24467	57.04324	0.04269946
2021-01-13	-0.018428585	89.826	89.91311	44.24013	0.05108035
2021-01-14	0.005809423	90.180	89.86541	48.74472	0.20270494

2021-01-15	0.002447709	90.244	89.90694	50.84478	0.21189820
2021-01-19	0.007748544	90.122	90.16796	57.73264	0.06954699
2021-01-20	0.026764000	90.570	91.16197	73.83713	0.21663397

Table 3: Sample Trainning set

Neural Network Regression Forecasting

4.4 Neural Network Regression Forecasting:

- **4.4.1 Training on Training Sample**: Using 'timeslice' sampling with a 100-day horizon and the remaining training days as the initial window.
- **4.4.2 Data Pre-processing**: Standardizing the data to ensure all predictors have a mean of zero and a standard deviation of one.
- **4.4.3 Model Fitting**: Using the **nnet** method with RMSE metric and a tuneLength of 5 for model tuning.
- **4.4.4 Prediction on Test Set**:

Metric	Value
RMSE	0.009827654
R-squared	0.001399213
MAE	0.007019770

Table 4: Neural Network results

- **4.4.5 Plot of Predicted and Actual Values**: Visualizing the predictions against actual values to assess model performance.

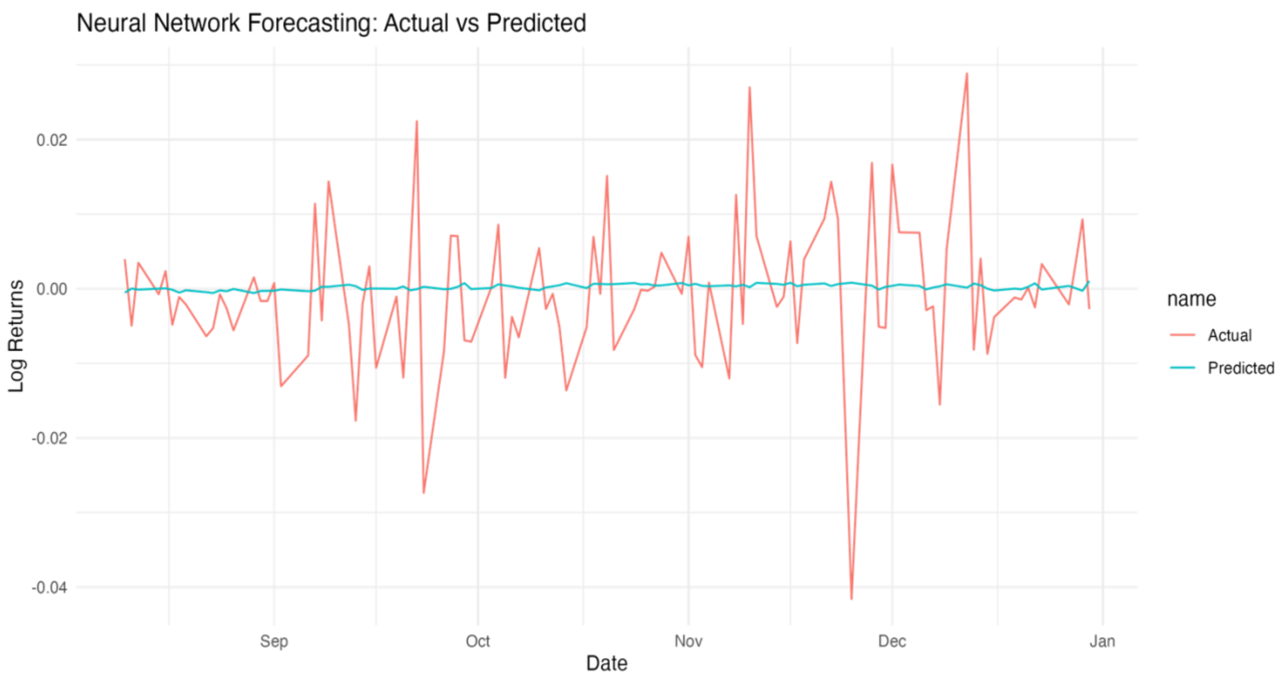

Figure 3: Neural Network Forecasting: Actual vs Predicted

- **4.4.6 Discussion on Accuracy**: The performance of the Neural Network Regression model for forecasting stock returns is evaluated using three key metrics: RMSE, R-squared, and MAE. The RMSE of 0.0098 indicates that the average deviation of the predicted values from the actual values is quite low, suggesting the model's predictions are relatively accurate. The MAE of 0.0070 further supports this by showing that the average absolute error between the predicted and actual values is minimal.

  However, the R-squared value is only 0.0014, indicating that the model explains just 0.14% of the variance in the log returns. This low R-squared suggests that while the model's predictions are close to the actual values in terms of magnitude, it fails to capture the underlying variability in the stock returns. This could be due to the inherent volatility of stock prices, the limitations of the selected technical indicators and sentiment analysis, or the need for further tuning and additional features.

SVM Regression Forecasting

4.5 SVM Regression Forecasting:

- **4.5.1 Radial Kernel**: Implementing SVM regression with a radial basis function (RBF) kernel for capturing non-linear relationships.

- **4.5.2 Prediction on Test Set**:

Metric	Value
RMSE	0.010307150
R-squared	0.005913989
MAE	0.007485382

Table 5: SVM Regression results

- **4.5.3 Plot of Predicted and Actual Values**:

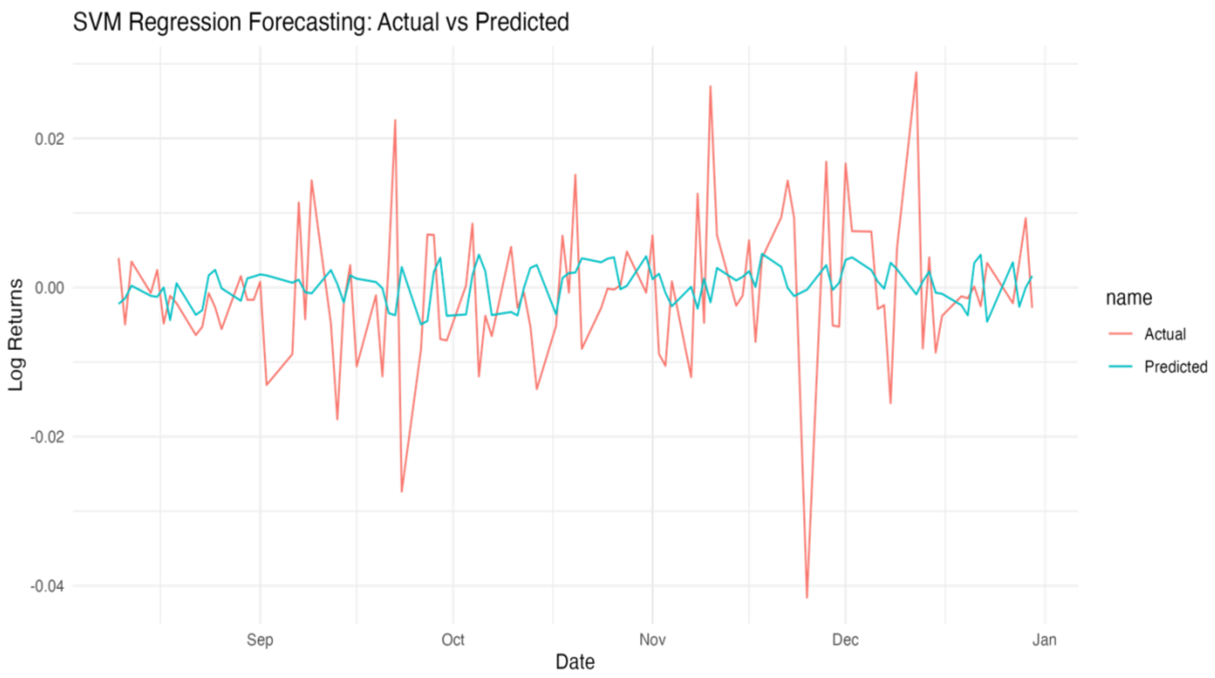

Figure 4: SVM Regression Forecasting: Actual vs Predicted

- **4.5.4 Discussion on Accuracy**: The performance of the SVM Regression model for forecasting stock returns is evaluated using RMSE, R-squared, and MAE. The RMSE of 0.0103 indicates that the average deviation of the predicted values from the actual values is low, suggesting reasonable accuracy. The MAE of 0.0075 further supports this by showing that the average absolute error between the predicted and actual values is minimal. The R-squared value of 0.0059 suggests that the model explains approximately 0.59% of the variance in the log returns, which is still quite low but slightly better than the Neural Network model.

Comparing the SVM model with the Neural Network model, the Neural Network has a slightly lower RMSE (0.0098) and MAE (0.0070), indicating better predictive accuracy in terms of average error magnitude. However, the R-squared value for the Neural Network is only 0.0014, much lower than the SVM's R-squared of 0.0059, suggesting that the SVM explains more variance in the log returns than the Neural Network.

Based on the predictive performance, both models show similar predictive accuracy, but the Neural Network has a slight edge in terms of RMSE and MAE. However, considering the explanatory power, the SVM is better as it explains more variance. If the goal is purely to minimize prediction error, the Neural Network is recommended. If understanding and explaining the variance in stock returns is also important, the SVM might be more suitable. Overall, given the marginal difference in RMSE and MAE, the Neural Network might be preferred for its slightly better predictive accuracy.

Reference list

Loh, W.-Y., 2011. Classification and regression trees. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 1(1), pp. 14-23.Available at: https://onlinelibrary.wiley.com/doi/abs/10.1002/widm.8 [Accessed 18 May, 2024].

Hand, D.J. and Henley, W.E., 1997. Statistical classification methods in consumer credit scoring: a review. Journal of the Royal Statistical Society: Series A (Statistics in Society), 160(3), pp. 523-541.Available at: https://www.jstor.org/stable/2983089 [Accessed 18 May, 2024].

Breiman, L., 2001. Random forests. Machine Learning, 45(1), pp. 5-32. Available at: https://link.springer.com/article/10.1023/A:1010933404324 [Accessed 18 May, 2024].

Quinlan, J.R., 1986. Induction of decision trees. *Machine Learning*, 1(1), pp. 81-106. Available at: https://link.springer.com/article/10.1007/BF00116251 [Accessed 18 May 2024].

Jolliffe, I.T., 2002. Principal component analysis. Springer, New York. Available at: https://link.springer.com/book/10.1007/b98835 [Accessed 18 May, 2024].

Fan, J., & Yao, Q., 2017. The Elements of Financial Econometrics. Cambridge University Press. ISBN: 978-1107161318. Available at: https://www.cambridge.org/core/books/elements-of-financial-econometrics/3B20C70EB6F913A2E7C6F8AC1A74D058 [Accessed 18 May 2024].

Connor, G. and Korajczyk, R.A., 1993. A Test for the Number of Factors in an Approximate Factor Model. The Journal of Finance, 48(4), pp.1263-1291. Available at: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=959023 [Accessed 18 May, 2024].

Feldman, R. and Sanger, J., 2007. The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data. Cambridge: Cambridge University Press. Online publication date: August 2009. DOI: 10.1017/CBO9780511546914 [Accessed 18 May, 2024].

Tetlock, P.C., 2007. Giving content to investor sentiment: The role of media in the stock market. The Journal of Finance, 62(3), pp.1139-1168. First published: 08 May 2007. DOI: 10.1111/j.1540-6261.2007.01232.x [Accessed 18 May, 2024].

Ngai, E.W.T., Hu, Y., Wong, Y.H., Chen, Y. and Sun, X., 2011. The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature. Decision Support Systems, 50(3), pp.559-569. Available at: https://www.sciencedirect.com/science/article/pii/S0167923610001302 [Accessed 18 May, 2024].

Goltz, N. and Mayo, M., 2017. Enhancing regulatory compliance by using artificial intelligence text mining to identify penalty clauses in legislation. MIREL 2017 - Workshop on MIning and REasoning with Legal texts, [online] June 16th, 2017, London (UK). Available at: https://ssrn.com/abstract=2977570 [Accessed 18 May, 2024].

Bollen, J., Mao, H., & Zeng, X., 2011. Twitter mood predicts the stock market. Journal of Computational Science, 2(1), pp. 1-8. Available at: https://doi.org/10.1016/j.jocs.2010.12.007 [Accessed 18 May 2024].

Office of the Australian Information Commissioner (OAIC). (n.d.). Privacy Act 1988. Available at: https://www.oaic.gov.au/privacy/privacy-legislation/the-privacy-act#:~:text=The%20Privacy%20Act%201988%20was,other%20organisations%2C%20handle%20personal%20information. [Accessed 18 May 2024].

Office of the Australian Information Commissioner (OAIC). (n.d.). Notifiable Data Breaches Scheme. Available at: https://www.oaic.gov.au/privacy/notifiable-data-breaches [Accessed 18 May 2024].

Australian Government. (n.d.). Consumer Data Right. Available at: https://www.cdr.gov.au/ [Accessed 18 May 2024].

Australian Human Rights Commission. (2019). Human Rights and Technology. Available at:

https://www.qld.gov.au/law/your-rights/human-rights#:~:text=The%20Human%20Rights%20Act%202019,protect%20and%20promote%20human%20rights [Accessed 18 May 2024].

Australian Government. (2021). Artificial Intelligence Ethics Framework. Available at: https://www.industry.gov.au/publications/australias-artificial-intelligence-ethics-framework [Accessed 18 May 2024]. Date published: 7 November 2019.

Office of the Australian Information Commissioner (OAIC). (n.d.). Privacy resources. Available at: https://www.oaic.gov.au/privacy [Accessed 18 May 2024].

Australian Government. (2021). AI Ethics Principles. Available at: https://www.industry.gov.au/publications/australias-artificial-intelligence-ethics-framework/australias-ai-ethics-principles [Accessed 18 May 2024].

Appendix

```
# ================ Task 3 =====================

# Load the necessary libraries
library(ggplot2)
library(readr)
library(gridExtra)

# ========== 1. Load and Preprocess Data ==========

# Load the dataset with suppressed column type message
data <- read_csv("UCI_credit_sample.csv", show_col_types = FALSE)

# Remove rows with any NA values
data <- na.omit(data)

# ========== 2. Create Individual Plots ==========

# Box plot for LIMIT_BAL
plot1 <- ggplot(data, aes(x = as.factor(default), y = LIMIT_BAL, fill = as.factor(default))) +
  geom_boxplot() +
  labs(title = "Box Plot of LIMIT_BAL by Default", x = "Default", y = "LIMIT_BAL") +
  scale_fill_discrete(name = "Default")

# Box plot for AGE
plot2 <- ggplot(data, aes(x = as.factor(default), y = AGE, fill = as.factor(default))) +
  geom_boxplot() +
  labs(title = "Box Plot of AGE by Default", x = "Default", y = "AGE") +
  scale_fill_discrete(name = "Default")

# Box plot for BILL_AMT1
plot3 <- ggplot(data, aes(x = as.factor(default), y = BILL_AMT1, fill = as.factor(default))) +
  geom_boxplot() +
  labs(title = "Box Plot of BILL_AMT1 by Default", x = "Default", y = "BILL_AMT1") +
  scale_fill_discrete(name = "Default")

# Box plot for BILL_AMT2
plot4 <- ggplot(data, aes(x = as.factor(default), y = BILL_AMT2, fill = as.factor(default))) +
  geom_boxplot() +
  labs(title = "Box Plot of BILL_AMT2 by Default", x = "Default", y = "BILL_AMT2") +
  scale_fill_discrete(name = "Default")

# Box plot for PAY_AMT1
plot5 <- ggplot(data, aes(x = as.factor(default), y = PAY_AMT1, fill = as.factor(default))) +
  geom_boxplot() +
  labs(title = "Box Plot of PAY_AMT1 by Default", x = "Default", y = "PAY_AMT1") +
  scale_fill_discrete(name = "Default")

# Box plot for PAY_AMT2
plot6 <- ggplot(data, aes(x = as.factor(default), y = PAY_AMT2, fill = as.factor(default))) +
  geom_boxplot() +
  labs(title = "Box Plot of PAY_AMT2 by Default", x = "Default", y = "PAY_AMT2") +
  scale_fill_discrete(name = "Default")
```

```
# ========== 3. Arrange and Save Combined Box Plots ==========

# Arrange plots in a 2x3 grid with adjusted height
combined_plot <- grid.arrange(plot1, plot2, plot3, plot4, plot5, plot6, ncol = 2, heights = c(1, 1, 1))

# Save the combined plot with increased height
ggsave("combined_plot.png", plot = combined_plot, width = 10, height = 12)

# ========== 4. Create Individual Bar Charts for Categorical Variables ==========

# Bar chart for SEX
bar_chart_sex <- ggplot(data, aes(x = as.factor(SEX), fill = as.factor(default))) +
  geom_bar(position = "dodge") +
  labs(title = "Bar Chart of SEX by Default", x = "SEX", y = "Count") +
  scale_fill_discrete(name = "Default")

# Bar chart for EDUCATION
bar_chart_education <- ggplot(data, aes(x = as.factor(EDUCATION), fill = as.factor(default))) +
  geom_bar(position = "dodge") +
  labs(title = "Bar Chart of EDUCATION by Default", x = "EDUCATION", y = "Count") +
  scale_fill_discrete(name = "Default")

# Bar chart for MARRIAGE
bar_chart_marriage <- ggplot(data, aes(x = as.factor(MARRIAGE), fill = as.factor(default))) +
  geom_bar(position = "dodge") +
  labs(title = "Bar Chart of MARRIAGE by Default", x = "MARRIAGE", y = "Count") +
  scale_fill_discrete(name = "Default")

# Bar chart for PAY_0
bar_chart_pay_0 <- ggplot(data, aes(x = PAY_0, fill = default)) +
  geom_bar(position = "dodge") +
  labs(title = "Bar Chart of PAY_0 by Default", x = "PAY_0", y = "Count") +
  scale_fill_discrete(name = "Default")

# Bar chart for PAY_2
bar_chart_pay_2 <- ggplot(data, aes(x = PAY_2, fill = default)) +
  geom_bar(position = "dodge") +
  labs(title = "Bar Chart of PAY_2 by Default", x = "PAY_2", y = "Count") +
  scale_fill_discrete(name = "Default")

# ========== 5. Arrange and Save Combined Bar Charts ==========

# Arrange bar charts in a 2x3 grid
combined_bar_charts <- grid.arrange(
  bar_chart_sex, bar_chart_education, bar_chart_marriage, bar_chart_pay_0, bar_chart_pay_2,
  ncol = 3, nrow = 2
)

# Save the combined bar charts with increased height
ggsave("combined_bar_charts.png", plot = combined_bar_charts, width = 10, height = 8)
# ========== 6. Generate Summary Statistics for Continuous Variables ==========

summary_continuous <- data %>%
  summarise(
```

```
    LIMIT_BAL_min = min(LIMIT_BAL),
    LIMIT_BAL_mean = mean(LIMIT_BAL),
    LIMIT_BAL_median = median(LIMIT_BAL),
    LIMIT_BAL_sd = sd(LIMIT_BAL),
    LIMIT_BAL_max = max(LIMIT_BAL),

    AGE_min = min(AGE),
    AGE_mean = mean(AGE),
    AGE_median = median(AGE),
    AGE_sd = sd(AGE),
    AGE_max = max(AGE),

    BILL_AMT1_min = min(BILL_AMT1),
    BILL_AMT1_mean = mean(BILL_AMT1),
    BILL_AMT1_median = median(BILL_AMT1),
    BILL_AMT1_sd = sd(BILL_AMT1),
    BILL_AMT1_max = max(BILL_AMT1),

    BILL_AMT2_min = min(BILL_AMT2),
    BILL_AMT2_mean = mean(BILL_AMT2),
    BILL_AMT2_median = median(BILL_AMT2),
    BILL_AMT2_sd = sd(BILL_AMT2),
    BILL_AMT2_max = max(BILL_AMT2),

    PAY_AMT1_min = min(PAY_AMT1),
    PAY_AMT1_mean = mean(PAY_AMT1),
    PAY_AMT1_median = median(PAY_AMT1),
    PAY_AMT1_sd = sd(PAY_AMT1),
    PAY_AMT1_max = max(PAY_AMT1),

    PAY_AMT2_min = min(PAY_AMT2),
    PAY_AMT2_mean = mean(PAY_AMT2),
    PAY_AMT2_median = median(PAY_AMT2),
    PAY_AMT2_sd = sd(PAY_AMT2),
    PAY_AMT2_max = max(PAY_AMT2)
  )

# ========== 7. Generate Summary Statistics for Categorical Variables ==========

summary_categorical <- data %>%
  group_by(default) %>%
  summarise(
    SEX_1 = sum(SEX == 1),
    SEX_2 = sum(SEX == 2),

    EDUCATION_1 = sum(EDUCATION == 1),
    EDUCATION_2 = sum(EDUCATION == 2),
    EDUCATION_3 = sum(EDUCATION == 3),
    EDUCATION_4 = sum(EDUCATION == 4),
    EDUCATION_5 = sum(EDUCATION == 5),
    EDUCATION_6 = sum(EDUCATION == 6),

    MARRIAGE_0 = sum(MARRIAGE == 0),
    MARRIAGE_1 = sum(MARRIAGE == 1),
    MARRIAGE_2 = sum(MARRIAGE == 2),
```

```
    MARRIAGE_3 = sum(MARRIAGE == 3),

    PAY_0_0 = sum(PAY_0 == 0),
    PAY_0_1 = sum(PAY_0 == 1),
    PAY_0_2 = sum(PAY_0 == 2),
    PAY_0_3 = sum(PAY_0 == 3),
    PAY_0_4 = sum(PAY_0 == 4),
    PAY_0_5 = sum(PAY_0 == 5),
    PAY_0_6 = sum(PAY_0 == 6),
    PAY_0_7 = sum(PAY_0 == 7),
    PAY_0_8 = sum(PAY_0 == 8),
    PAY_0_9 = sum(PAY_0 == 9),

    PAY_2_0 = sum(PAY_2 == 0),
    PAY_2_1 = sum(PAY_2 == 1),
    PAY_2_2 = sum(PAY_2 == 2),
    PAY_2_3 = sum(PAY_2 == 3),
    PAY_2_4 = sum(PAY_2 == 4),
    PAY_2_5 = sum(PAY_2 == 5),
    PAY_2_6 = sum(PAY_2 == 6),
    PAY_2_7 = sum(PAY_2 == 7),
    PAY_2_8 = sum(PAY_2 == 8),
    PAY_2_9 = sum(PAY_2 == 9)
  )


# =============== Task 4 ====================

# ============================================
# 1. Load Necessary Libraries
# ============================================
library(tidyverse)
library(TTR)
library(quantmod)
library(xts)
library(caret)
library(e1071)
library(nnet)
library(ggplot2)

# ============================================
# 2. Load and Preprocess the Data
# ============================================
# Load the data
data_stock_24 <- read.csv("data_stock_24.csv")
data_stock_24$Date <- as.Date(data_stock_24$Date, format="%Y-%m-%d")

# Inspect the column names to ensure correctness
colnames(data_stock_24)

# Subset the data to include only Date, ATVI-US, and sentiment columns
data_subset <- data_stock_24[, c("Date", "ATVI.US", "sentiment")]

# Convert data to xts format
data_xts <- xts(data_subset$`ATVI.US`, order.by=data_subset$Date)
```

```
# ================================================
# 3. Calculate Technical Indicators
# ================================================
# Calculate Daily Logarithmic Returns
log_returns <- ROC(data_xts, type="continuous", na.pad=FALSE)
colnames(log_returns) <- "log_returns"

# Calculate Moving Average (5-day)
ma_5 <- SMA(data_xts, n=5)
colnames(ma_5) <- "ma_5"

# Calculate Exponential Moving Average (5-day)
ema_5 <- EMA(data_xts, n=5)
colnames(ema_5) <- "ema_5"

# Calculate RSI (5-day)
rsi_5 <- RSI(data_xts, n=5)
colnames(rsi_5) <- "rsi_5"

# Combine all indicators into a single xts object
indicators <- merge(log_returns, ma_5, ema_5, rsi_5, all=FALSE)

# Lag the predictors by one period using xts::lag.xts, excluding log_returns
predictors <- indicators[, -1]  # Exclude the first column (log_returns)
lagged_predictors <- xts::lag.xts(predictors, k=1)

# Combine the non-lagged log_returns with lagged predictors
final_indicators <- merge(indicators$log_returns, lagged_predictors, all=FALSE)

# Remove NA values resulting from the lag operation
final_indicators <- na.omit(final_indicators)

# ================================================
# 4. Process Sentiment Indicator
# ================================================
# Convert sentiment data to xts format
sentiment_xts <- xts(data_subset$sentiment, order.by=data_subset$Date)
colnames(sentiment_xts) <- "sentiment"

# Lag the sentiment indicator by one period
lagged_sentiment <- xts::lag.xts(sentiment_xts, k=1)

# Remove NA values resulting from the lag operation
lagged_sentiment <- na.omit(lagged_sentiment)

# ================================================
# 5. Merge All Indicators and Prepare Final Data
# ================================================
# Merge the lagged sentiment indicator with final indicators
final_data <- merge(final_indicators, lagged_sentiment, all=FALSE)

# Convert to data frame for easier handling
final_data_df <- data.frame(Date=index(final_data), coredata(final_data))
```

```
# ================================================
# 6. Set Seed and Split Data into Training and Testing Sets
# ================================================
# Set seed for reproducibility
set.seed(2345)

# Define the prediction window size
pred.window <- 100 # Last 100 days for the testing set

# Determine the indices for training and testing sets
idx <- c(1:(nrow(final_data_df) - pred.window))
d_train <- final_data_df[idx, ]
d_test <- final_data_df[(max(idx) + 1):nrow(final_data_df), ]

# Inspect the training and testing data
head(d_train)
head(d_test)

# ================================================
# 7. Neural Network Regression Forecasting
# ================================================
# Define the control parameters for model training
cntrl2 <- trainControl(method = "timeslice",
                       initialWindow = nrow(d_train) - pred.window,
                       horizon = pred.window,
                       fixedWindow = TRUE,
                       savePredictions = TRUE,
                       allowParallel = TRUE)

# Define preprocessing steps
prep1 <- c("center", "scale", "zv") # Include "zv" which removes variables with close to zero variance

# Train the neural network model
set.seed(2345)
nnet_model <- train(log_returns ~ . - Date,
                    data = d_train,
                    method = "nnet",
                    trControl = cntrl2,
                    preProcess = prep1,
                    linout = TRUE,
                    trace = FALSE,
                    metric = "RMSE",
                    tuneLength = 5)

# Print the trained model details
print(nnet_model)

# Prediction on the test set
nnet_predictions <- predict(nnet_model, newdata = d_test)

# Calculate performance measures
nnet_performance <- postResample(nnet_predictions, d_test$log_returns)

# Print performance measures
```

```
print(nnet_performance)

# Plot actual vs predicted values for Neural Network
data_pred_nnet <- data.frame(Date = d_test$Date, Actual = d_test$log_returns, Predicted = nnet_predictions)
data_pred_long_nnet <- pivot_longer(data_pred_nnet, cols = -Date)

plot_nn <- ggplot(data_pred_long_nnet, aes(x = Date, y = value, color = name)) +
  geom_line() +
  labs(title = "Neural Network Forecasting: Actual vs Predicted", x = "Date", y = "Log Returns") +
  theme_minimal()

# Save the plot using ggsave
ggsave("NN.png", plot = plot_nn, width = 10, height = 5)


# ============================================
# 8. SVM Regression Forecasting
# ============================================
# Train the SVM model with radial kernel
set.seed(2345)
svm_model <- train(log_returns ~ . - Date,
                   data = d_train,
                   method = "svmRadial",
                   trControl = cntrl2,
                   preProcess = prep1,
                   metric = "RMSE",
                   tuneLength = 5)

# Print the trained model details
print(svm_model)

# Prediction on the test set
svm_predictions <- predict(svm_model, newdata = d_test)

# Calculate performance measures
svm_performance <- postResample(svm_predictions, d_test$log_returns)

# Print performance measures
print(svm_performance)

# Plot actual vs predicted values for SVM
data_pred_svm <- data.frame(Date = d_test$Date, Actual = d_test$log_returns, Predicted = svm_predictions)
data_pred_long_svm <- pivot_longer(data_pred_svm, cols = -Date)

# Create the plot for SVM
plot_svm <- ggplot(data_pred_long_svm, aes(x = Date, y = value, color = name)) +
  geom_line() +
  labs(title = "SVM Regression Forecasting: Actual vs Predicted", x = "Date", y = "Log Returns") +
  theme_minimal()

# Save the plot using ggsave
ggsave("SVM.png", plot = plot_svm, width = 10, height = 5)
```

```
# Print performance measures for SVM
cat("SVM Performance Measures:\n")
print(svm_performance)

cat("Neural Network Performance Measures:\n")
print(nnet_performance)
```