# AI-POWERED NETFLIX FORECASTING SYSTEM

AI-Powered Netflix Stock Prediction Using LSTM & CNN, and Subscriber Growth Forecasting Using RFR, with Power BI Integration

Submitted By: Maryam Khattak
Student ID: 22083966

SEPTEMBER 12, 2024

LONDON METROPOLITAN UNIVERSITY
MAK2723@my.londonmet.ac.uk

# Abstract

**Context:** The context of this thesis is to predict stock prices and subscriber growth rate, which are critical for informed decision-making in the media streaming industry. As a global leader, Netflix requires accurate prediction models to remain competitive and make strategic decisions based on market trends and user growth patterns.

**Objectives:** This project aims to develop predictive models for Netflix stock price and regional subscriber growth using deep learning and machine learning techniques, including LSTM, CNN, and RFR, integrated with Power BI for visualization.

**Methods:** Netflix stock price data and subscriber growth data were processed, and the models were trained to forecast stock prices and subscriber growth rates. Evaluation metrics like MSE, RMSE and $R^2$ were used to assess model performance.

**Results:** The LSTM model outperformed CNN in predicting stock prices, and the Random Forest Regressor showed strong predictive accuracy in predicting regional subscriber growth rate.

**Conclusion:** The project successfully built models for stock price and subscriber growth rate forecasting, providing valuable insights into Netflix's stock price trends and subscriber growth dynamics. Power BI dashboards further enhanced decision making with interactive visualizations.

**Keywords:** Stock Price Prediction, LSTM, CNN, Random Forest Regressor, Power BI, Netflix, Subscriber Growth Rate, Deep Learning, Machine Learning

# Acknowledgements

I would like to express my sincere gratitude to my supervisor, **Professor Preeti Patel**, for her invaluable guidance, support, and encouragement throughout this project. Her expertise and insights have been instrumental in shaping the direction of my work, and her mentorship has greatly enhanced my learning experience.

I also wish to thank my family and friends for their continuous support, motivation, and encouragement during the course of this project. Without their understanding and assistance, this endeavor would not have been possible.

Finally, I would like to acknowledge the resources and facilities provided by London Metropolitan University, which enabled me to successfully complete this project.

# List of Figures

Figure 2.1: LSTM Network Architecture....21
Figure 2.2: CNN Network Architecture....22
Figure 2.3: Random Forest Regressor Architecture....23

Figure 3.1: CRISP-DM Life Cycle....30
Figure 3.2: Project Schedule for Netflix Price & Subscriber Growth Prediction....32
Figure 3.3: Mean Squared Error (MSE) Formula....37
Figure 3.4: Root Mean Squared Error (RMSE) Formula....37
Figure 3.5: R-squared ($R^2$) Formula....38
Figure 3.6: CRISP-DM Phases and Tasks....39

Figure 4.1: System Overview....44
Figure 4.2: System Architecture for Netflix Stock Price Prediction....46
Figure 4.3: System Architecture for Subscriber Growth Rate Prediction....48
Figure 4.4: Code for Loading Netflix Stock Data....50
Figure 4.5: Output of Loaded Netflix Stock Data....51
Figure 4.6: Code for Loading Netflix Subscriber Growth Data....51
Figure 4.7: Output of Loaded Netflix Subscriber Growth Data....52
Figure 4.8: Overview of Netflix Stock Price Data Structure....53
Figure 4.9: Overview of Subscriber Growth Data Structure....54
Figure 4.10: Time Series Plot of Netflix Stock Closing Prices....55
Figure 4.11: Candlestick Plot with 20-Day and 30-Day Moving Averages....56
Figure 4.12: Daily Returns of Netflix Stock....56
Figure 4.13: Dual-Axis Plot of Stock Prices and Trading Volume....57
Figure 4.14: Correlation Matrix Heatmap of Netflix Stock Data....57
Figure 4.15: Number of Netflix Users by Country....58
Figure 4.16: Distribution of Netflix Subscribers by Age....59
Figure 4.17: Distribution of Netflix Subscribers by Subscription Type....59
Figure 4.18: Distribution of Netflix Subscription Plans by Country....60
Figure 4.19: Monthly Revenue Distribution by Subscription Type (Pie Chart)....61
Figure 4.20: Number of Users Joining Over Time....61
Figure 4.21: Netflix Stock Historical Data Power BI Dashboard....64

Figure 4.22: Netflix Subscriber Growth Rate Power BI Dashboard....65
Figure 4.23: Code for Data Scaling and Normalization....66
Figure 4.24: Code for Creating Datasets for LSTM/CNN....66
Figure 4.25: Code snippet for Preparing Data for Modeling....67
Figure 4.26: Prepared Data for LSTM/CNN....67
Figure 4.27: Code for Date Standardization and Cleaning....68
Figure 4.28: Cleaned Subscriber Data After Date Standardization....68
Figure 4.29: Code for Feature Engineering in Subscriber Growth Data....68
Figure 4.30: Output of Feature Engineering for Subscriber Growth Data....69
Figure 4.31: Code for Encoding Categorical Variables....70
Figure 4.32: Output of Encoded Subscriber Growth Data....70
Figure 4.33: Correlation Matrix....71
Figure 4.34: Feature Selection Code Snippet....72
Figure 4.35: Code for Data Splitting....73

Figure 5.1: LSTM Model Building & Training....75
Figure 5.2: CNN Model Building & Training....77
Figure 5.3: Random Forest Regressor Model Building....79
Figure 5.4: Random Forest Regressor Model Training....79
Figure 5.5: LSTM & CNN Models Evaluation....81
Figure 5.6: Evaluation Metrics Output....82
Figure 5.7: Actual Prices vs LSTM Model Predictions....82
Figure 5.8: Actual Prices vs CNN Model Predictions....82
Figure 5.9: Actual Prices vs Model Predictions....83
Figure 5.10: Code Snippet for Model Predictions....84
Figure 5.11: Output Result of Actual vs Predicted Growth Rate %....85
Figure 5.12: Evaluation Metrics for Random Forest Regressor Model....85
Figure 5.13: Actual vs Predicted Growth Rates by Country....86
Figure 5.14: Feature Importance in Random Forest Model....86
Figure 5.15: Monthly Subscriber Count by Quarter in Each Region....87
Figure 5.16: Regional Trends: Monthly Subscriber Count and Growth Rate....87

Figure 6.1: Real-Time Stock Prediction using LSTM & CNN Model....90
Figure 6.2: Power BI Dashboard for Netflix Stock Price Predictions....93
Figure 6.3: Power BI Dashboard for Netflix Subscriber Growth Rate Predictions....95

# List of Tables

Table 4.1: Attributes & Descriptions of Netflix Stock Prices Dataset........................................49
Table 4.2: Attributes & Descriptions of Subscriber Growth Dataset..........................................50
Table 4.3: Categories for Encoding..........................................................................................................69
Table 4.4: Selected Features & Target Variable for Subscriber Growth Data .........................72

# List of Abbreviations

**NFLX**: Netflix, Inc.

**AI**: Artificial Intelligence

**ML**: Machine Learning

**LSTM**: Long Short-Term Memory

**CNN**: Convolutional Neural Network

**RFR**: Random Forest Regressor

**ARIMA**: Autoregressive Integrated Moving Average

**SVM:** Support Vector Machine

**GBM:** Gradient Boosting Machines

**RNN:** Recurrent Neural Network

**GRU:** Gated Recurrent Units

**PCA:** Principal Component Analysis

**CRISP-DM:** Cross-Industry Standard Process for Data Mining

**MSE**: Mean Squared Error

**RMSE**: Root Mean Squared Error

**$R^2$**: R-squared (Coefficient of Determination)

**EDA:** Exploratory Data Analysis

**NaT:** Not a Time

**BI**: Business Intelligence

# Contents

**Abstract** .......... i

**Acknowledgements** .......... ii

**List of Figures** .......... iii

**List of Tables** .......... v

**List of Abbreviations** .......... vi

**Contents** .......... vii

**1 Introduction** .......... 10

1.1 Business Understanding .......... 11
1.2 Aim .......... 11
1.3 Objectives .......... 11
1.4 Research Questions .......... 12
1.5 Structure of the Thesis .......... 13

**2 Background & Related Work** .......... 14

2.1 Historical Context & Evolution .......... 15
2.2 Key Challenges in Forecasting Trends .......... 15
2.3 Advancements & Model Comparisons .......... 16
2.4 Previous Work in Stock Price Prediction .......... 17
2.5 Previous Work in Subscriber Growth Prediction .......... 19
2.6 Advanced Techniques & Models .......... 20
2.6.1 Deep Learning for Stock Price Prediction .......... 20
2.6.2 Machine Learning for Subscriber Growth Prediction .......... 23
2.7 Challenges in Dataset Gathering .......... 24

2.8 Model Integration with Power BI ........ 24
2.9 Advancements & Current Era Work ........ 24

**3 Methodology ........ 26**

3.1 Software Environment ........ 26
3.1.1 Google Colab ........ 26
3.1.2 Python ........ 27
3.2 Libraries & Tools ........ 27
3.2.1 TensorFlow and Keras ........ 27
3.2.2 scikit-learn ........ 28
3.2.3 yFinance ........ 28
3.2.4 Power BI ........ 28
3.3 Models for Prediction ........ 29
3.3.1 Long Short-Term Memory (LSTM) ........ 29
3.3.2 Convolutional Neural Network (CNN) ........ 29
3.3.3 Random Forest Regressor ........ 29
3.4 Data Sources ........ 29
3.5 CRISP-DM Methodology ........ 29
3.5.1 Business Understanding ........ 31
3.5.2 Data Understanding ........ 32
3.5.3 Data Preparation ........ 33
3.5.4 Modeling ........ 34
3.5.5 Evaluation ........ 36
3.5.6 Deployment ........ 39

**4 Analysis & Design ........ 41**

4.1 Problem Analysis ........ 42
4.1.1 Predictive Objectives ........ 42
4.1.2 Challenges ........ 42
4.2 System / Model Architecture ........ 43
4.2.1 System Overview ........ 43
4.2.2 Flow Diagrams ........ 45
4.3 Data Understanding & Exploration ........ 49
4.3.1 Data Description ........ 49
4.3.2 Data Loading ........ 50
4.3.3 Exploratory Analysis ........ 52

4.3.4 Data Preparation ........ 66

**5 Implementation & Evaluation ........ 74**

5.1 Model Building & Training ........ 75

5.1.1 Long Short-Term Memory (LSTM) ........ 75

5.1.2 Convolutional Neural Network (CNN) ........ 77

5.1.3 Random Forest Regressor ........ 79

5.2 Model Evaluation ........ 80

5.2.1 Netflix Stock Price Prediction ........ 80

5.2.2 Netflix Subscriber Growth Rate Prediction ........ 83

**6 Deployment ........ 89**

6.1 Real-Time Netflix Stock Price Prediction ........ 90

6.2 Integration with Power BI ........ 91

6.2.1 Exporting & Importing Data Files ........ 91

6.2.2 Results Visualization in Power BI ........ 92

6.3 Monitoring and Maintenance Strategies ........ 95

6.4 Regular Performance Reviews and Adjustments ........ 95

**7 Conclusion & Recommendations ........ 97**

7.1 Conclusion ........ 97

7.2 Recommendations for Future Work ........ 98

7.3 Practical Implications ........ 99

**8 References ........ 100**

**9 Appendix ........ 102**

# Chapter 1

# Introduction

In today's data-driven world, the use of advanced analytical techniques to make informed business decisions is essential. Rapid advances in technology have increased our ability to analyze complex data and gain actionable insights. This project focuses on using state-of-the-art machine learning and deep learning models to forecast stock prices and subscriber growth rates for Netflix. The process begins with a comprehensive literature review, which lays the groundwork by examining existing methodologies and identifying gaps in the field. Building on this foundation, we will proceed to data collection and preprocessing, ensuring that the information is clean, relevant, and ready for analysis.

Once data preparation is complete, the focus will shift to developing predictive models. While there are many methods for forecasting stock prices and forecasting subscriber growth, including traditional statistical methods such as ARIMA for time series forecasting or linear regression models, we learn deep learning methods, especially Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNNs) for neural networks. These models were selected due to their superior ability to capture complex temporal patterns and non-linear relationships in stock price data, which traditional methods typically struggle with due to their linear assumptions and limited capacity to model complex dependencies over time [1] [2]. Recent research has shown that LSTM networks, in particular, are more effective in predicting stock prices by accounting for both short-term fluctuations and long-term trends [3].

Furthermore, while other machine learning algorithms such as Support Vector Machines (SVM) or Gradient Boosting Machines (GBM) can be used to predict the subscriber growth rates, we have chosen the Random Forest Regressor. This choice is motivated by its robustness in handling high-dimensional data, its ability to handle overfitting, and its effectiveness in capturing nonlinear interactions between features. The Random Forest Regressor provides a balanced trade-off between accuracy and interpretability, making it the best choice for our project, which aims to understand regional and temporal factors affecting subscriber growth. Recent research highlights the ability of Random Forest to handle complexity and provide reliable predictions in different scenarios [4] [5].

Data visualization will play an important role in determining the outcome of our predictive models. We will create interactive dashboards using Power BI to visualize historical stock prices, forecasts as subscriber growth rates, and other key metrics. These dashboards will provide a clear and detailed view of the data, facilitating effective analysis and decision-making.

The final phase will involve moving these graphs and dashboards to a production facility. We will discuss ways to monitor and monitor the performance of our models to ensure that they remain consistent and reliable. By combining real-time data with consistent use of performance analytics, we will ensure that our forecasting tools continue to perform well in an ever-evolving market environment.

## 1.1 Business Understanding

Understanding the business aspects of Netflix is important for this project. Netflix operates in a competitive and growth environment where accurate forecasts of stock prices and subscriber growth are important for strategic management and investment decisions. Stock price forecasts provide insight into the future performance of the company, helping investors make informed decisions and optimize their portfolios. On the other hand, forecasting subscriber growth allows Netflix's executives to better plan, better allocate content, and plan for market expansion.

The project addresses this need by using advanced predictive modeling techniques to provide actionable insights into Netflix's revenue and subscriber metrics. By combining deep learning and machine learning techniques, we aim to provide a comprehensive understanding of Netflix's business dynamics and provide valuable insights through advanced predictive models.

## 1.2 Aim

The main aim of this project is to develop a robust predictive model for Netflix stock prices and subscriber growth rate using advanced deep learning and machine learning models. By building these models the project seeks to provide accurate forecasts and deliver actionable insights that can inform strategic decisions. The ultimate goal is to provide stakeholders with reliable tools that can help navigate market trends and make informed decisions through sophisticated predictive analytics.

## 1.3 Objectives

To achieve the primary aim of the project, the following objectives will be followed:

- **Develop LSTM and CNN Models:** Build and train Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNNs) to forecast Netflix Inc. (NFLX) stock prices using historical data sourced from Yahoo Finance. These models will capture both temporal dependencies and patterns complexity in the stock data.

- **Compare Model Performance:** Evaluate and compare the performance of LSTM and CNN models to determine which is more accurate and reliable for stock price forecasting. This type of research requires rigorous testing and validation to identify optimal models.
- **Implement a Random Forest Regressor:** Use a random forest regressor to predict customer growth rates in each region. This machine learning model will analyze various factors affecting customer growth and generate predictions that help in strategic decision-making.
- **Analyze Regional Subscriber Trends:** Examine region-based factors that affect customer dynamics to gain insight into how different regions contribute to overall growth. This study will support Netflix's strategic planning and market expansion efforts.
- **Integrate Power BI for Visualization:** Create an interactive dashboard in Power BI to view historical stock prices, forecasted customer growth rates, and comparative analysis. These dashboards will provide a comprehensive view of the data, facilitating effective analysis and insight interpretation.
- **Provide Interactive Dashboards:** Create interactive visualizations that allow stakeholders to explore data insights and make appropriate decisions based on the forecasts and analytics provided.

## 1.4 Research Questions

The project will address the following research questions:

**RQ 1:** How well can LSTM and CNN models predict Netflix’s stock price with historical data, and what unique insights do these models provide?
**Motivation:** Traditional statistical methods generally fail to capture the nonlinearity and time complexity of stock price data. The LSTM and CNN models, which are known for their ability to handle sequential and complex systems, were chosen to evaluate how well they can handle these challenges in Netflix stock price prediction.

**RQ 2:** What are the key factors influencing Netflix’s subscriber growth rates in different regions?
**Motivation:** Understanding regional differences in client development is important for targeted marketing and strategic decisions. This question aims to identify the most important drivers of subscriber growth, which can vary substantially depending on geographic and cultural contexts.

**RQ 3:** How can a Random Forest Regressor be effectively used to predict subscriber growth, and what advantages does it offer in dealing with diverse and complex datasets?

**Motivation:** Random Forest Regressor was chosen because of its robustness and ability to handle large and diverse datasets with multiple variables. This question seeks to examine the model's effectiveness in predicting subscriber growth and potential profitability over other simple or less complex models.

**RQ 4:** What insights can be gained from incorporating these predictive models into interactive dashboards for stakeholder decision making?
**Motivation:** Visualizing forecasts in an accessible and interactive format is key to making informed decisions. This question addresses how adding predictive models to Power BI dashboards can empower stakeholders to interpret complex data and make strategic decisions.

## 1.5 Structure of the Thesis

The thesis is structured as follows:

- **Chapter 1:** Discusses the introduction, business understanding, aim, objectives, and research questions.
- **Chapter 2:** Reviews existing research on stock price forecasting and subscriber growth prediction, identifying gaps the project aims to address.
- **Chapter 3:** Details the methodology, including the software environment, tools and libraries, selected models, data sources, and the CRISP-DM methodology.
- **Chapter 4:** Focuses on problem analysis, system architecture, data understanding, and exploration as part of the analysis and design phase.
- **Chapter 5:** Covers model building, training, and evaluation, including the assessment of model performance.
- **Chapter 6:** Outlines the deployment of predictive models, integration with Power BI, and strategies for monitoring, maintenance, and regular performance review.
- **Chapter 7:** Summarizes key findings, discusses objective achievement, and offers recommendations for future work.

# Chapter 2

# Background & Related Work

Predicting stock prices and subscriber growth using advanced machine learning techniques represents a critical research area with substantial implications for both the financial and media industries. Over the years, the field has seen significant advancements with the development and implementation of sophisticated algorithms. Techniques such as Long Short-Term Memory (LSTM) networks, Convolutional Neural Networks (CNNs), and ensemble models like Random Forest have greatly enhanced the accuracy and reliability of predictive analytics. These models have become essential tools for understanding market dynamics and consumer behavior.

This chapter provides a comprehensive review of the historical context, challenges, technological developments, and major contributions in the field of predictive analytics. It begins with a discussion of the historical evolution of stock prices and subscriber growth prediction models, highlighting key milestones and breakthroughs. The review then addresses various challenges faced in the field, such as data quality, feature engineering, and computational resource management, emphasizing their impact on model performance and accuracy.

Technological developments are explored in detail, focusing on the evolution of machine learning techniques from traditional statistical methods to advanced deep learning models. Key contributions from recent research are examined, including the application of LSTM networks, CNNs, and Random Forest algorithms, each bringing unique strengths to predictive modeling. The integration of these techniques with external data sources, such as social media sentiment and macroeconomic indicators, is also discussed, showcasing how these additional layers of information enhance prediction capabilities.

The chapter further explores the integration of predictive models with visualization tools like Power BI. This integration allows for real-time visualization and interactive exploration of predictive insights, facilitating better decision-making and strategic planning. By leveraging these advanced models and visualization techniques, stakeholders can gain a more nuanced understanding of market trends and subscriber dynamics.

In summary, the chapter outlines the significant progress made in predictive analytics using machine learning and deep learning techniques, while also highlighting the ongoing need for innovation and refinement in this dynamic field.

## 2.1 Historical Context & Evolution

The evolution of stock price forecasting has seen significant advancements from its inception in the early 20th century. Initially, stock price prediction was based on statistical techniques such as linear regression and time-series forecasting. These early methods primarily utilized historical data to identify trends and make predictions, relying on statistical and economic models that could describe observed patterns in stock prices.

The mid-20th century marked a pivotal shift with the advent of computer technology, which enabled the development of more complex forecasting models. Among these, the Autoregressive Integrated Moving Average (ARIMA) model gained prominence for its ability to handle time-series data and provide forecasts based on past values and trends. ARIMA and similar models enhanced the accuracy of economic forecasting by incorporating components that could account for different types of data patterns and variations over time.

As the late 20th and early 21st centuries approached, the field of stock price prediction experienced a transformative shift with the rise of machine learning and artificial intelligence. Traditional statistical models began to be complemented and, in some cases, replaced by machine learning techniques such as Support Vector Machines (SVM), decision trees, and neural networks. These methods offered enhanced capabilities for modeling nonlinear relationships and managing large and complex datasets.

The introduction of deep learning models further revolutionized the field. Techniques such as Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNNs) emerged as powerful tools for stock price forecasting. LSTMs, with their ability to capture long-term temporal dependencies, proved effective in modeling sequential data and predicting stock prices based on historical trends. Meanwhile, CNNs, known for their strength in extracting spatial features from data, contributed to improved predictive accuracy by identifying patterns that traditional models might miss.

The evolution of stock price forecasting reflects a gradual progression from statistical methods to advanced machine learning and deep learning techniques. Each stage of this development has contributed to more accurate and sophisticated predictive models, driven by technological advancements and the increasing complexity of financial data.

## 2.2 Key Challenges in Forecasting Trends

Forecasting stock prices and subscriber growth involves several challenges:

- **Data Quality and Availability:** The accuracy of forecasting models depends heavily on the quality and completeness of historical data. Inconsistent data collection methods, missing standards, and data noise can significantly degrade model performance.
- **Market Volatility:** Many unpredictable factors, including economic indicators, political events, and market sentiment, influence financial markets, making accurate forecasting difficult.
- **Overfitting:** Machine learning models, especially complex ones, can easily overfit the training data, leading to inferences about new information that are not found with good results.
- **Feature Selection:** Identifying the most appropriate features from multiple potential predictors is critical for building robust models. This process often requires domain expertise and repeated testing.

## 2.3 Advancements & Model Comparisons

The field of stock price forecasting has undergone significant technological advances, evolving from traditional statistical methods to sophisticated machine learning and deep learning models. Each improvement reflects some progress in our understanding of economic data and our ability to model it effectively.

**Traditional Statistical Methods** have long been used to forecast stock prices due to the straightforward approach and ease of interpretation. Linear regression, one of the simplest and most commonly used methods, models the relationship between the dependent variable (such as stock price) and one or more independent variables. Its strength lies in its simplicity and clear interpretation of the results.

The Autoregressive Integrated Moving Average (ARIMA) model is another key component in time series forecasting. The ARIMA model is designed to capture temporal patterns in data, using past observations to predict future values. Autoregressive terms, moving averages, and differences for model stationary time series are included. While these techniques are effective for relatively simple linear trends and patterns, they often struggle with the complexities and nonlinearities of financial markets. Their reliance on assumptions of a linear nature of relationships limits their ability to capture phenomenal market dynamics.

**Machine Learning Models** have brought a new dimension to stock price prediction by addressing some of the limitations of traditional methods. Support Vector Machines (SVMs) represent a significant advance, capable of handling high-quality data and generating robust decision boundaries. SVMs use kernel functions to transform data into higher dimensional forms, making it possible to identify nonlinear separating hyperplanes. Decision trees provide a more interpretable way, dividing data into smaller groups based on feature values to make predictions.

However, single decision trees can easily overfit, resulting in non-general models. Ensemble methods, such as Random Forest and Gradient Boosting, address this issue by combining multiple decision trees to improve prediction performance and robustness. Random Forest aggregates the predictions of multiple decision trees, increasing accuracy and reducing variance, while Gradient Boosting executes sequences of models to correct errors in previous trees, optimizing performance. These machine learning techniques offer great flexibility and power in modeling complex interactions in data.

**Deep Learning Models** have marked a significant improvement in the ability to capture and predict stock price movements. Long Short-Term Memory (LSTM) networks, a type of recurrent neural network (RNN), are particularly effective for time series data. LSTMs are designed to capture long-term dependence and temporal patterns by incorporating mechanisms such as memory cells and gating units. This design enables LSTMs to store and manipulate information over long periods of time, making them ideally suited for modeling stock price trends and seasons. Convolutional Neural Networks (CNNs), originally developed for image analysis, have also been developed for financial forecasting. CNNs excel in recognizing local structures and features in data using convolutional layers representing spatial patterns. When applied to time series data, CNNs can detect striking patterns and trends, increasing the accuracy of forecasts.

The evolution from traditional statistical methods to modern machine learning and deep learning models reflects a significant improvement in the potential of stock price forecasting techniques. Each technological revolution brings with it advanced tools and techniques for understanding and forecasting financial markets, contributing to more accurate and reliable forecasts. While traditional methods provide simplicity and clarity, machine learning and deep learning techniques provide the sophistication needed to model complex and non-linear data, demonstrating continuous improvements in predictive analytics.

## 2.4 Previous Work in Stock Price Prediction

Many studies have explored different models and methods for stock price prediction, which show the progress and utility of various machine learning and deep learning techniques in this field.

In recent years, the interest in application of machine learning and deep learning techniques for stock price prediction has increased dramatically. Increasing technological capabilities and the availability of vast amounts of information have given rise to this trend. Researchers have increasingly focused on using these advanced models to improve the accuracy and reliability of stock market forecasts. The field has seen a shift from traditional statistical approaches to more sophisticated machine learning frameworks, which focus on the importance of selection, data preprocessing, and external factors such as public opinion and media sentiment will be combined.

Abhyuday, Akash, Shivam and Ashish conducted an extensive study on machine learning techniques for stock market pricing. Their work provides a comprehensive range of models, including traditional statistical methods, machine learning algorithms, and deep learning. Strengths and weaknesses of each method were emphasized, and it was noted that while traditional methods such as ARIMA are useful for linear data, machine learning models such as Support Vector Machines (SVM) and Random Forests (RF), provides high performance by capturing complex patterns in the data. Furthermore, the use of Long Short-Term Memory (LSTM) networks, Gated Recurrent Units (GRU), Recurrent Neural Networks (RNN), and K-Nearest Neighbors (KNN) for bank value forecasting was discussed. LSTM and GRU were noted for their ability to maintain long-term stability in time series data, while RNNs were emphasized for their effectiveness in sequential data processing. KNN was discussed as a non-parametric method that can be particularly useful for classification tasks. The authors emphasized the importance of selective features and data preprocessing in improving the accuracy of these models, and described techniques such as normalization, standardization, and Principal Component Analysis (PCA) features are extracted in detail [6].

In another important study, Shivani and Govinda Rao focused on the use of LSTM networks for stock price forecasting. Their research involves the collection and preprocessing of stock exchange data to generate time-series datasets suitable for LSTM models. A sliding window method was used to match input sequences and target values, allowing the LSTM network to learn from historical data and make future predictions. The preprocessing steps include dealing with missing values, normalizing the data to ensure that all features are on the same scale, and converting categorical variables to statistics. The study showed that LSTM models can store long-term memory and process sequential data, making them particularly suitable for forecasting economic time series data. The authors compared the performance of LSTM with other neural network algorithms and traditional machine learning models, showing that LSTM outperformed them in terms of prediction accuracy and robustness. Their method provided a comprehensive method for model training, hyperparameter tuning, and performance analysis [7].

Amzad Basha, Senthil Kumar, Martha Sucharitha, Samreen Ayesha and Macherla Bhagya, and explored a hybrid approach for time series analysis of stock price forecasting using stated machine learning and deep learning models together so. Their study combined methods such as ARIMA, LSTM, and hybrid models to capture various aspects of stock price movements. The analysis involved preprocessing the data to remove noise and normalize values, after which ARIMA was used to model the linear components and LSTM to capture non-linear patterns and temporal dependencies. Data preprocessing steps included removing outliers, scaling the data with Min-Max Scaler, and splitting the time series data into training and testing sets. The hybrid models were designed to combine the strengths of ARIMA with linear trends and LSTM with complex temporal relationship monitoring. The authors provided detailed implementation steps, including model training, hyperparameter tuning, and performance evaluation, showing that the hybrid approach yielded significant improvements in prediction accuracy compared to individual ones of the use of samples. They also highlighted the importance of combining external sources such as market sentiment and media analysis to increase forecast accuracy [8].

These studies highlight the advances in stock price forecasting techniques. The transition from traditional statistical models to sophisticated machine learning and deep learning marks a significant advance in this field. By leveraging the strengths of models, researchers have been able to achieve high levels of accuracy and reliability in forecasting stock prices, thereby providing valuable insights for investors and financial analysts.

## 2.5 Previous Work in Subscriber Growth Prediction

Predicting the growth of subscription-based services has become a particularly useful area of research and application, with the importance of understanding and anticipating consumer behaviour. With the rapid expansion of digital services and media platforms, accurate forecasting can provide valuable insights into strategic planning and decision-making. The challenge lies in how to effectively capture the dynamic patterns and factors that influence subscriber retention.

Advances in machine learning and deep learning technologies have provided new ways to address these challenges. Researchers are increasingly turning to sophisticated algorithms, such as Long Short-Term Memory (LSTM) networks, Convolutional Neural Networks (CNNs) and regression techniques, to model and predict subscriber growth. These techniques use historical data and temporal dependencies to increase predictive accuracy and provide a more nuanced understanding of user behavior and developmental trajectories. As the industry evolves, integrating these advanced approaches continues to play a key role in developing successful strategies for subscription-based services.

One notable study in this area was conducted by Harihara Reddy Pamuluri (2023), who investigated the use of deep learning techniques to predict user mobility, namely subscriber prediction will continue is closely related. The study used deep learning algorithms, including Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNNs), to analyze user behavior and movement patterns. Pamuluri's approach proved that deep learning models can effectively capture complex and temporal patterns in user data, leading to more accurate predictions of user mobility and, by extension, subscriber growth. The research emphasizes the significance of incorporating temporal and spatial data into models to enhance prediction capabilities. By using advanced neural network algorithms, the study demonstrated the feasibility of developing deep learning techniques for predicting client growth and development effectively [9].

In addition, a comprehensive study by Irfan Ullah, Basit Raza, Ahmad Kamran Malik, Muhammad Imran, Saif Ul Islam, and Sung Won Kim explored the use of Random Forest to predict churn in the telecom sector. This study highlights the effectiveness of Random

Forest models in identifying key factors affecting customer churn and improving forecast accuracy. The study highlights the importance of feature selection and preprocessing in improving the performance of predictive models. While the main focus was on churn prediction, the methods and insights presented are more relevant for understanding subscriber growth and retention in subscription-based services [10].

In conclusion, recent advances in machine learning and deep learning have greatly improved the ability to predict subscriber growth. Both studies demonstrate the effectiveness of models including LSTM, CNN, and Random Forest in capturing complex patterns and improving prediction accuracy. These methods provide valuable tools for predicting client mobility and creating strategic decisions in subscription-based services.

## 2.6 Advanced Techniques & Models

In recent years, advances in machine learning and deep learning have changed the financial forecasting landscape. This section examines these sophisticated mechanisms in more detail, focusing on how they have enhanced the prediction of stock price and subscriber growth. We explore the use of sophisticated models, particularly Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNNs), which have been shown to be highly effective in capturing complex patterns of time series in the 19th century. In addition, we explore machine learning models such as the Random Forest Regressor that show great promise in predicting user outcomes by efficiently handling large datasets and capturing nonlinear relationships.

### 2.6.1 Deep Learning for Stock Price Prediction

Deep learning models have emerged as powerful pricing tools due to their ability to model complex relationships in data. This subsection highlights two popular deep learning algorithms, LSTM and CNN, that have been successfully used to forecast stock prices, offering superior accuracy and robustness compared to traditional methods.

#### 2.6.1.1 Long Short-Term Memory (LSTM)

The LSTM network, which is a type of recurrent neural network (RNN), is specifically designed to model sequential data by capturing long time horizons, making it particularly effective in stock price forecasting because of ability to maintain and use in therefore the past for a long time. **Figure 2.1** shows the basic LSTM Network Architecture, which provides a detailed framework for understanding how the LSTM model works.

The architecture starts with the input data, which is first normalized to ensure that the input features are on a consistent scale. After normalization, the data goes into preparation to format it for incorporation into the LSTM network. The process is essentially LSTM model construction, where the network is built and the hyperparameters are optimized for optimal performance. Following this, the LSTM model is used to determine the closing value based on the processed data. Finally, the model's predictions are tested for accuracy and reliability, completing the sequence from data input to outcome generation. This basic framework is the foundation for LSTM models, helping to understand the process of stock price forecasting through different data models.

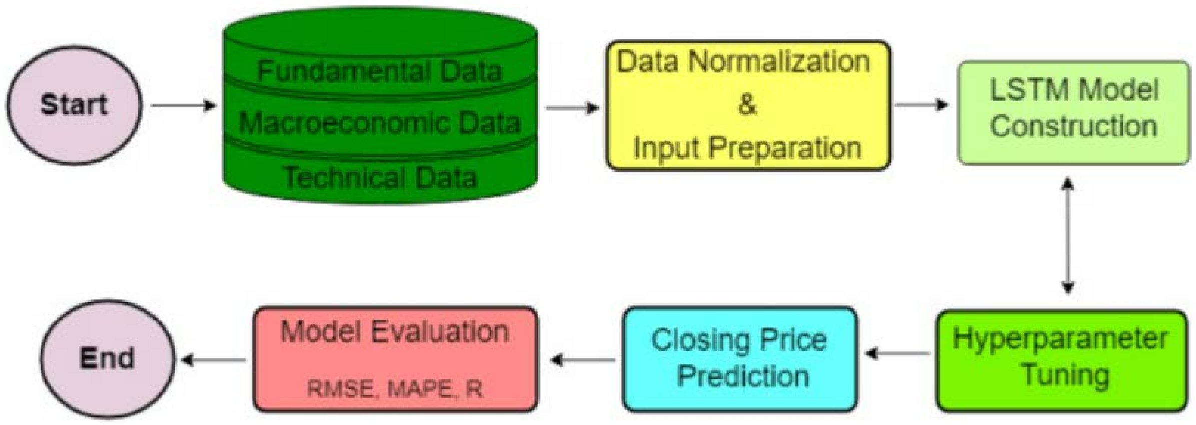


Figure 2.1: LSTM Network Architecture

Akash Patel, Devang Patel, and Seema Yadav conducted a comprehensive comparative analysis of stock price forecasting using different deep learning models, with a special focus on Long Short-Term Memory (LSTM) correlations. The aim of their study is to evaluate the effectiveness of LSTM in stock price forecasting compared to traditional statistical methods. The study carefully compared LSTM with other models, including linear regression and support vector machines, in terms of accuracy and robustness.

The authors used extensive historical stock price data to train and test their models, emphasizing the ability of LSTM networks to capture complex time dependencies in the data. Unlike conventional methods, which often struggle with complex structures and nonlinear relationships, LSTM networks excelled due to their design, which included memory cells designed to remember objects long-term dependence and the problem of gradient loss is mitigated.

Their study revealed that the LSTM networks not only improved forecasting accuracy but also exhibited greater robustness in the face of market volatility and noise. The success of the model was attributed to its ability to learn from changing market conditions and to adjust over time, which is crucial for effective stock price forecasting. This work highlights the benefits of applying advanced deep learning, such as LSTM, to economic forecasting and highlights its potential to outperform traditional methods in terms of accuracy and reliability [11].

### 2.6.1.2 Convolutional Neural Network (CNN)

CNNs commonly used in image recognition have also been applied to stock price forecasting by treating time series data as sequences that can be analyzed like images. By capturing local and global patterns in the data, CNNs can effectively learn from historical stock price movements. **Figure 2.2** shows the basic structure of the CNN model, which includes an input layer, convolutional layers, pooling layers for feature extraction, fully connected layers, and an output layer for classification.

The process starts at the input layer, where raw time series data are fed into the network. The convolutional layers then apply various filters to identify patterns or features in the data and focus on local regions to extract relevant information. Pooling layers follow, reducing the spatial dimensions of the data while preserving the most important features, and effectively summarizing the data to prevent overfitting. Fully connected layers combine the extracted features and process them to identify complex patterns and relationships. Finally, the output layer generates the forecast, which in stock price forecasting typically represents the distribution or regression of future price movements.

This algorithm enables CNNs to efficiently process and learn from sequences, making it a powerful tool for time series analysis and stock price forecasting.

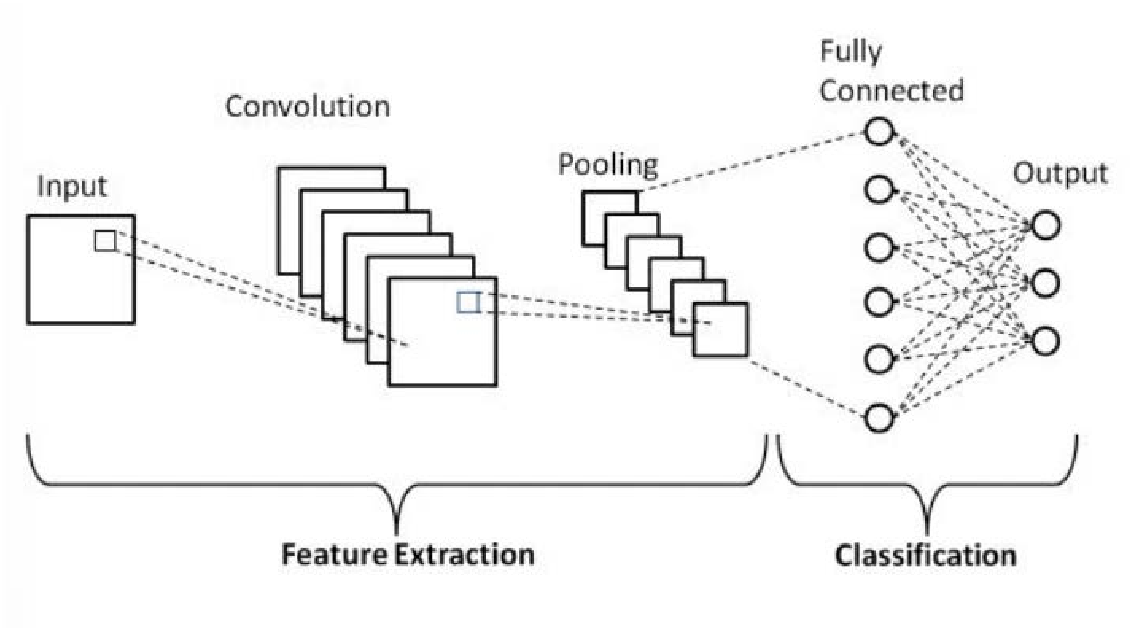


Figure 2.2: CNN Network Architecture

Rashi Jaiswal and Brijendra Singh conducted a comprehensive study on savings forecasting using Convolutional Neural Network (CNN) and Gated Recurrent Unit (GRU) models. Their research focused on combining CNN and GRU to exploit the strengths of both architectures for more accurate stock price forecasting.

In their study, Jaiswal and Singh used CNN layers for feature extraction, which enabled the model to capture complex and complex patterns in historical stock price data. The ability of CNN to apply a variety of filters and learn sequences from the data proved effective in detecting significant patterns and abnormalities that may be traditional the patterns will be lost.

To complement the feature extraction capability of the CNN, the authors added GRU layers to the model. GRUs are known to be adept at handling sequential data and capturing temporal dependencies, which are crucial for predicting long-term stock price trends. By combining the spatial feature extraction capabilities of CNNs with the time series modeling capabilities of GRUs, their hybrid model aims to enhance the overall forecasting accuracy.

The study of Jaiswal and Singh demonstrated the potential of hybrid deep learning models in stock market forecasting, and demonstrated how the integration of CNNs with GRUs can provide detailed analysis of economic data. Their approach highlighted the effectiveness of using advanced neural networks to improve forecasting performance and gain deeper insights into market trends [12].

### 2.6.2 Machine Learning for Subscriber Growth Prediction

This section explores the use of machine learning models in predicting user growth, which is central to performance forecasting. We focus on how these models, in particular the Random Forest Regressor, are used to analyze complex user behavior and predict subscriber growth trends. By handling large datasets and accounting for a variety of influencing factors, these models provide robust and accurate predictions, providing valuable insights for strategic decision-making.

#### 2.6.2.1 Random Forest Regressor

Random Forest Regressor, an ensemble learning method, is widely used for regression tasks due to its robustness and ability to handle high-dimensional data. It constructs multiple decision trees during training and outputs the average prediction of individual trees, reducing overfitting and improving generalizability. Below **Figure 2.3** shows a basic architecture for Random Forest Regressor.

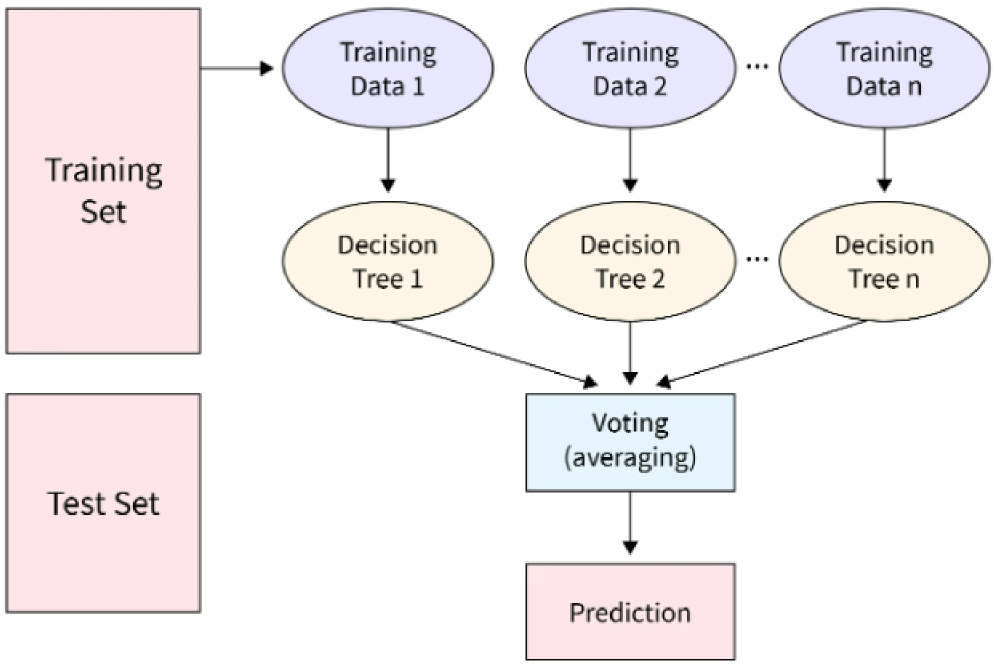


Figure 2.3: Random Forest Regressor Architecture

## 2.7 Challenges in Dataset Gathering

Ensuring data quality is key in predictive modeling, as the accuracy and precision of the data directly influences the confidence in the model results. Inconsistent data collection methods and the presence of missing values can severely compromise the integrity of the study, leading to skewed or erroneous predictions. To meet these challenges, a robust data validation and cleansing process must be implemented to ensure the accuracy and completeness of the data.

Feature engineering is another important aspect, which involves transforming raw data into meaningful features that improve the performance of the model. This process requires a deep understanding of the domain and often involves iterative testing to identify and build the most appropriate features. Effective feature engineering can greatly improve the model's ability to capture patterns and underlying relationships.

Furthermore, managing large datasets presents its own set of challenges, requiring large computing resources and highly efficient data management techniques. Handling large amounts of data requires optimized data storage, processing, and retrieval systems to ensure that models can perform efficiently and effectively. Balancing these considerations is essential to developing robust predictive models and gaining accurate insights that can guide strategic decisions.

## 2.8 Model Integration with Power BI

Integrating machine learning models with visualization tools like Power BI is essential for translating complex predictive analytics into actionable insights. Power BI offers robust features for real-time data visualization, enabling stakeholders to interact with and explore the predictions generated by LSTM, CNN, and Random Forest models.

- **Real-time Data Integration:** Power BI can connect to various data sources, including databases and APIs, to continuously update and display predictions.
- **Interactive Dashboards:** Users can create interactive dashboards to visualize stock price predictions and subscriber growth forecasts, facilitating data-driven decision-making.
- **Custom Visuals:** Power BI supports custom visuals, allowing for the integration of advanced charts and graphs that illustrate model performance and predictions.

## 2.9 Advancements & Current Era Work

Recent developments in stock price forecasting and subscriber growth forecasting have increasingly focused on the use of deep learning algorithms and hybrid models to increase forecast accuracy. These improvements include the integration of external data sources, such as social media sentiment and macroeconomic indicators, to provide additional context and increase the robustness of forecasts.

Sentiment Analysis has become an integral part of modern forecasting techniques. Researchers have sought to incorporate sentiment analysis from social media platforms, news articles, and other forms of public opinion. By analyzing the sentiment expressed in these external sources of information, analysts can gauge market sentiment and refine stock price forecasts. This approach helps to understand the consensus of market participants and its potential impact on stock movements.

Transfer Learning is another notable development that has gained momentum in recent years. This approach involves using pre-trained models developed on related tasks and adapting them to a specific dataset through optimization. Transfer learning has shown to be effective in improving prediction performance, especially when processing a small number of cases. By leveraging knowledge from large datasets and applying it across multiple specialties, researchers can achieve better results and reduce the need for extensive training data.

The field of stock price forecasting and subscriber growth forecasting has improved significantly with this development. Modern techniques, such as Long Short-Term Memory (LSTM) networks, Convolutional Neural Networks (CNNs), and Random Forest algorithms, as well as hybrid methods combining multiple models, offer promising solutions for predictions authentic and reliable. These methods have shown significant improvements in capturing robust patterns and temporal dependence in data.

Additionally, integrating these predictive models with visualization tools such as Power BI enhances their functionality by enabling real-time, interactive analytics of predictive insights. Visualization tools allow stakeholders to better understand and interpret results, making it easier to make informed decisions based on the latest information.

Despite the progress made, continued research and innovation is still needed to overcome existing limitations and exploit the full potential of these advanced models. Continued advances in machine learning, deep learning, and data integration are expected to drive further improvements in forecasting accuracy and usability and determine the future of forecasting in dynamic industry as banks and subscription-based services.

Chapter 3

# Methodology

This chapter provides details of the methodology used in our work predicting Netflix Inc.'s stock price and subscriber growth rates. We will describe in detail the software environment, tools, and languages used, the models chosen for the predictions, and the methods used in the analysis. In addition, we will dive deeper into the CRISP-DM methodology, guiding the entire process from data understanding to processing, adding diagrams to illustrate key steps and processes.

## 3.1 Software Environment

Software development involves the process of designing, developing, testing, and maintaining software applications and systems. It involves a set of steps aimed at developing software that meets specific needs and solves specific problems. In our project, software development is critical to building and implementing solutions that drive our data analytics and predictive modeling efforts. It provides a framework for translating complex requirements into functional tools, enabling efficient data processing, model training, and results visualization. Through the use of robust software development practices and tools, we ensure consistent and flexible deployment of our work, allowing for iterative development and seamless integration of various features. This approach not only enhances the reliability and performance of our solutions but also facilitates collaboration and improved flexibility throughout the project lifecycle.

For our project, we have picked a strong and adaptable software setup to handle and carry out our tasks effectively. The main parts of our software setup include:

- Google Colab
- Python

### 3.1.1 Google Colab

Google Colab is a cloud-based coding platform that lets you write and run Python code in your browser. This tool has many benefits. First, it's easy to use so you don't need to set up complex stuff on your computer. You can start coding right away, which speeds up your work. Also, Colab makes teamwork easy. Multiple people can work on the same notebook at the same time. What's more, Google Colab gives you access to powerful computers, including GPUs and TPUs. These are key to train deep learning models. Colab links with Google Drive too. This makes it simple to save and share notebooks and datasets. Because of this, you can always get to your work and know it's backed up. In addition, we used the GPU framework in Google Colab to accelerate the training of our deep learning model. This significantly reduced the training time for both LSTM and CNN models.

### 3.1.2 Python

Python is our primary coding language. We chose it because it is flexible and has many libraries for data analysis, modeling, and testing. We rely on libraries like Pandas and NumPy to work with the data and do the calculations. Pandas provide us with powerful tools such as DataFrames, which simplify the processing of big data. NumPy allows us to perform array operations, which is key for number crunching. When it comes to machine learning, we can't do it without Scikit-learn. It provides a wide range of algorithms and tools for storing data, selecting models, and evaluating efficiency. We also use TensorFlow and Keras to build and train our deep learning models. These libraries have easy-to-use interfaces and great docs, which help us build complex neural networks with minimal hassle. We used Python's matplotlib and seaborn libraries to create graphs to analyze the results of our models.

## 3.2 Libraries & Tools

In addition to Google Colab and Python, there are several specific tools and libraries that are important in our work, each playing a specific role in ensuring successful predictive modeling and effective data visualization. The tools and libraries used are:

- TensorFlow and Keras
- scikit-learn
- yFinance
- Power BI

We will discuss each of these components in detail to explain their specific roles and how they contribute to the success of our project.

### 3.2.1 TensorFlow and Keras

TensorFlow and Keras play key roles in building and training our deep learning models, like Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNNs). Google created TensorFlow, an open-source toolkit that gives us a full platform to create and use machine learning models. It helps with many deep learning jobs and has lots of tools to make models better and train them. Keras now part of TensorFlow as tf.keras, gives us an easy-to-use way to build and train neural networks. This simple way to do things helps when you want to test out different model designs.

In our project, we use TensorFlow and Keras to build LSTM and CNN models to predict Netflix Inc. (NFLX) stock prices in real-time. We selected the LSTM model because it can spot time patterns in data series, which makes it perfect to forecast stock prices. This model can study past data and factor in long-term trends and seasonal changes. The CNN model though for image data, is used to look at time-series data to find complex patterns and trends. By comparing how well these models work, we want to see which one gives more exact and dependable stock price predictions.

### 3.2.2 scikit-learn

scikit-learn is a powerful and widely used Python library for machine learning, which provides simple and efficient tools for data analysis and modeling. It supports several machine learning algorithms, including the Random Forest Regressor used in our work.

In our project, scikit-learn is used to use the Random Forest Regressor model to predict Netflix subscriber growth rates in different regions. This library provides the necessary functionality to build, train, and thoroughly test the model, ensuring that we can utilize the full potential of ensemble learning approaches in our prediction analysis.

### 3.2.3 yFinance

yFinance is a Python library that gives users easy access to historical stock market data from Yahoo Finance. This tool has a crucial impact on our project as it provides the historical stock prices, we need to train our LSTM and CNN models. By using yFinance, we can download comprehensive datasets that include historical and real-time price movements. These datasets are key to develop accurate forecasting models. This library makes the data acquisition process easier making sure we have the most relevant and current data to inform our predictions. In our project, we use yFinance to get historical stock data for Netflix. This data is the foundation for training our predictive models and checking how well they perform.

### 3.2.4 Power BI

Power BI, a tool Microsoft built for business analytics, lets users make interactive dashboards and visuals. It turns complex data into insights you can act on through easy-to-use interfaces. This tool connects different data sources, which helps create dynamic reports and visuals. These visuals make it easier to make good decisions.

In our project, we use Power BI to show historical stock price changes, expected subscriber growth, and how different models compare. This tool helps us present our data analysis results and lets people interact with them. This allows us to explore the data and grasp the insights from our predictive models. Power BI's interactive dashboards help us communicate results well and make smart choices based on the data we can see.

## 3.3 Models for Prediction

In our project, we use a variety of models to predict stock prices and forecast subscriber growth rates. The specific models used are outlined in detail below:

### 3.3.1 Long Short-Term Memory (LSTM)

LSTM networks are a type of recurrent neural network (RNN) that engineers created to model data that comes in sequences by capturing relationships over long periods. They work well to predict stock prices because they can remember information from the past for a long time. LSTM networks solve the vanishing gradient problem, which often causes issues for regular RNNs allowing them to learn from long data sequences. This ability is key to understand the patterns in time that show up in how stock prices change.

### 3.3.2 Convolutional Neural Network (CNN)

CNNs used to recognize images, have found a new role in predicting stock prices by treating time data as pictures. These networks can spot both small and big patterns in the data, which makes them good at learning from past stock price changes. By using different filters, CNNs can find important patterns and odd things that old methods might not see, which helps the model make better guesses about future prices.

### 3.3.3 Random Forest Regressor

To predict subscriber growth rates, we use the Random Forest Regressor. This method constructs multiple decision trees during training and outputs the average prediction of the individual trees. We pick the Random Forest Regressor because it's strong and can handle lots of data at once, which helps to avoid overfitting and makes the results more useful overall. It is effective in capturing complex relations between features, which is an important key for accurately predicting subscriber growth rates.

## 3.4 Data Sources

We acquire historical stock data from Yahoo Finance using the yFinance library. This data includes daily stock prices for Netflix (NFLX). To predict subscriber growth rate, we use data from the Netflix Userbase dataset, which has detailed information about Netflix subscribers, including demographics, subscription types, and how they use the service.

## 3.5 CRISP-DM Methodology

The CRISP-DM (Cross-Industry Standard Process for Data Mining) method is a popular framework for data mining and analytics projects. It offers a structured way to approach these projects helping teams move from understanding the problem to putting the solution into action. CRISP-DM consists of six main phases:

1. Business Understanding
2. Data Understanding
3. Data Preparation
4. Modeling
5. Evaluation
6. Deployment

**Figure 3.1** shows the CRISP-DM Life Cycle, which provides a visual representation of these six processes and shows how they interact with each other in an iterative process. These figures will provide the readers with an overview of a data mining project according to the CRISP-DM method.

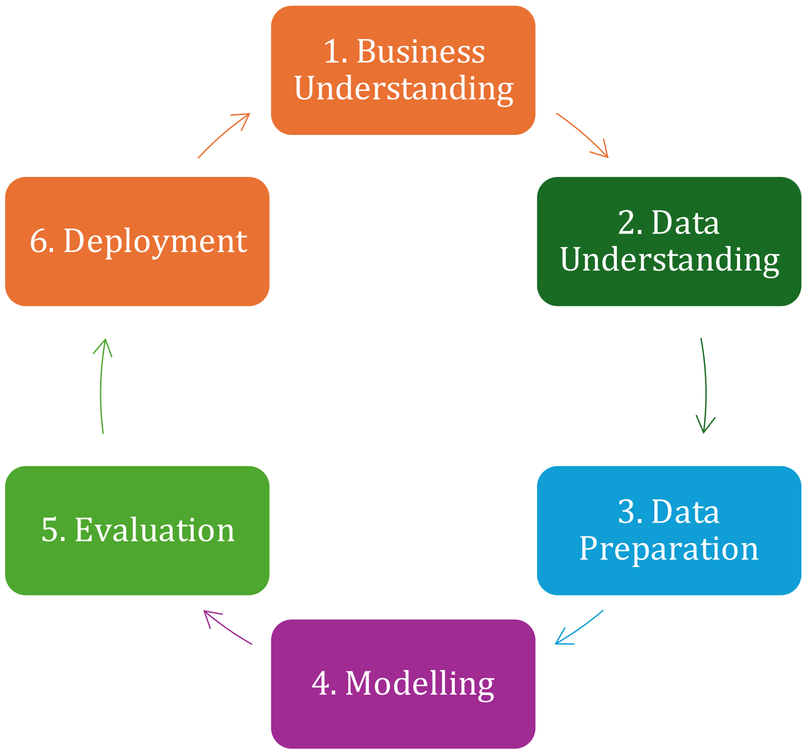


Figure 3.1: CRISP-DM Life Cycle

**Figure 3.6** presented in the deployment phase of this chapter shows the CRISP-DM Phases and Tasks, which describes the system in detail. This figure not only identifies the six key segments but also details the specific tasks associated with each segment. By bringing these figures to the end of the chapter, especially in the discussion of the deployment phase, we emphasize the importance of understanding each task within the CRISP-DM approach, especially as it relates to successful project management. The detailed breakdown in **Figure 3.6** highlights the complexities and details required at the implementation stage, ensuring that all aspects of the project are adequately considered and addressed.

Below is a breakdown of each stage in the CRISP-DM process and how it's used to predict Netflix stock prices and subscriber growth rates using Python and Google Colab.

### 3.5.1 Business Understanding

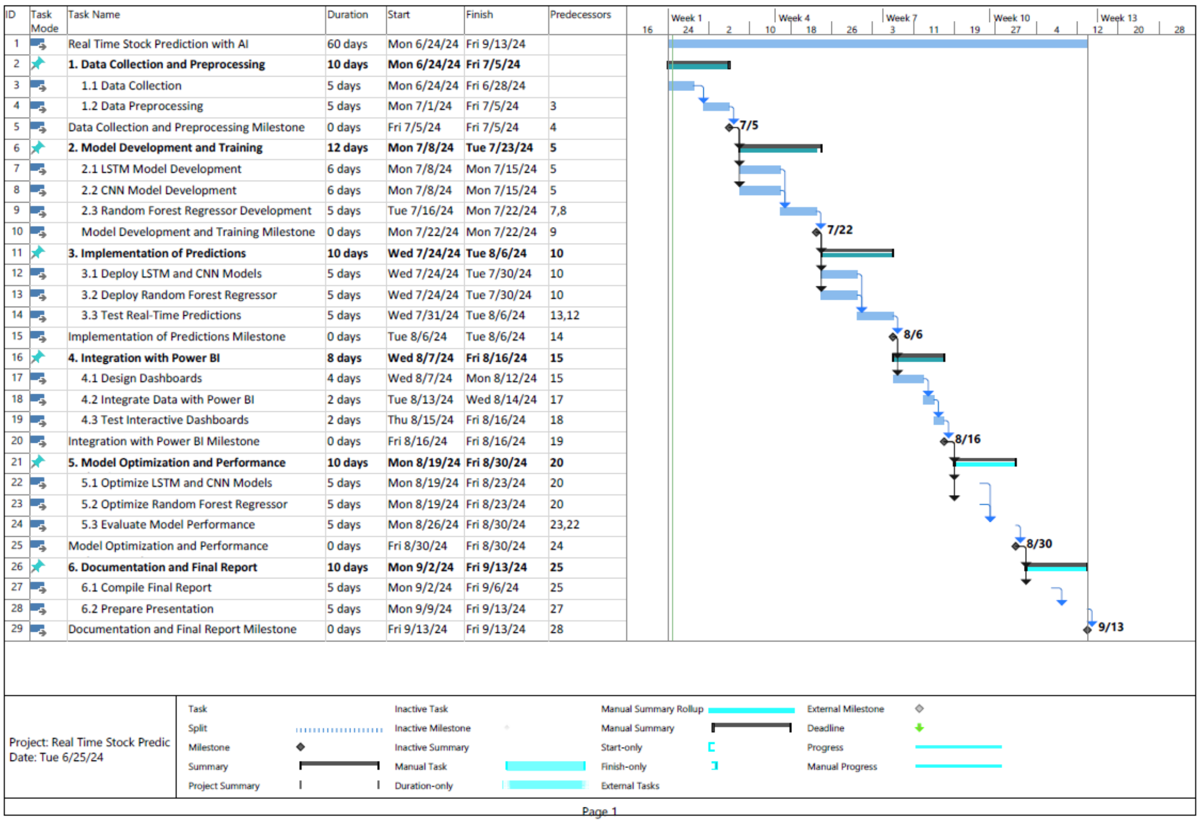
The Business Understanding phase is important as it sets the groundwork for the entire project. It involves understanding the goals, defining the problem, and setting the criteria for success. This section outlines the project objectives and outlines the detailed project plan. First, the objectives of the project are defined. The main objectives of the project are:

1. **Stock Price Prediction:** Create LSTM and CNN models to predict real-time stock price of Netflix Inc. (NFLX) using historical data obtained from Yahoo Finance and compare their performance for accuracy and reliability.
2. **Subscriber Growth Rate Prediction:** Implement a machine learning model as Random Forest Regressor, to predict subscriber growth in each region, and examine regional factors that influence subscriber trends to support strategic decision-making.
3. **Visualization with Power BI:** Integrate Power BI to visualize historical stock price movements, predicted subscriber growth rates, and comparative analysis, providing stakeholders with interactive dashboards to explore data insights and making informed decisions.

To determine the success criteria for this objective, we use specific performance metrics. For stock price prediction, the model is evaluated using Mean Squared Error (MSE) and Root Mean Squared Error (RMSE). In subscriber growth forecasting, R-squared and Root Mean Squared Error (RMSE) are the key metrics. These values are critical for assessing the accuracy and reliability of prediction models.

A detailed project plan is developed, specifying the tools, methods, and systems to be used. Python, with great libraries like TensorFlow, Keras, and scikit-learn, is basically a programming language. Google Colab was chosen as the development environment due to its ease of use and computing power. Stock price information is obtained from Yahoo Finance through the yfinance library, while subscriber growth information is obtained from publicly available Netflix data.

In addition, a project schedule plan has been developed to guide the execution of the project. This plan outlines the overall timeline, tasks, and milestones for the project, and ensures systematic progress and timely completion. **Figure 3.2** provides an overview of this project schedule plan.



| ID | Task Mode | Task Name | Duration | Start | Finish | Predecessors |
|---|---|---|---|---|---|---|
| 1 | ➜ | Real Time Stock Prediction with AI | 60 days | Mon 6/24/24 | Fri 9/13/24 | |
| 2 | 📌 | **1. Data Collection and Preprocessing** | **10 days** | **Mon 6/24/24** | **Fri 7/5/24** | |
| 3 | ➜ | 1.1 Data Collection | 5 days | Mon 6/24/24 | Fri 6/28/24 | |
| 4 | ➜ | 1.2 Data Preprocessing | 5 days | Mon 7/1/24 | Fri 7/5/24 | 3 |
| 5 | ➜ | Data Collection and Preprocessing Milestone | 0 days | Fri 7/5/24 | Fri 7/5/24 | 4 |
| 6 | 📌 | **2. Model Development and Training** | **12 days** | **Mon 7/8/24** | **Tue 7/23/24** | **5** |
| 7 | ➜ | 2.1 LSTM Model Development | 6 days | Mon 7/8/24 | Mon 7/15/24 | 5 |
| 8 | ➜ | 2.2 CNN Model Development | 6 days | Mon 7/8/24 | Mon 7/15/24 | 5 |
| 9 | ➜ | 2.3 Random Forest Regressor Development | 5 days | Tue 7/16/24 | Mon 7/22/24 | 7,8 |
| 10 | ➜ | Model Development and Training Milestone | 0 days | Mon 7/22/24 | Mon 7/22/24 | 9 |
| 11 | 📌 | **3. Implementation of Predictions** | **10 days** | **Wed 7/24/24** | **Tue 8/6/24** | **10** |
| 12 | ➜ | 3.1 Deploy LSTM and CNN Models | 5 days | Wed 7/24/24 | Tue 7/30/24 | 10 |
| 13 | ➜ | 3.2 Deploy Random Forest Regressor | 5 days | Wed 7/24/24 | Tue 7/30/24 | 10 |
| 14 | ➜ | 3.3 Test Real-Time Predictions | 5 days | Wed 7/31/24 | Tue 8/6/24 | 13,12 |
| 15 | ➜ | Implementation of Predictions Milestone | 0 days | Tue 8/6/24 | Tue 8/6/24 | 14 |
| 16 | 📌 | **4. Integration with Power BI** | **8 days** | **Wed 8/7/24** | **Fri 8/16/24** | **15** |
| 17 | ➜ | 4.1 Design Dashboards | 4 days | Wed 8/7/24 | Mon 8/12/24 | 15 |
| 18 | ➜ | 4.2 Integrate Data with Power BI | 2 days | Tue 8/13/24 | Wed 8/14/24 | 17 |
| 19 | ➜ | 4.3 Test Interactive Dashboards | 2 days | Thu 8/15/24 | Fri 8/16/24 | 18 |
| 20 | ➜ | Integration with Power BI Milestone | 0 days | Fri 8/16/24 | Fri 8/16/24 | 19 |
| 21 | 📌 | **5. Model Optimization and Performance** | **10 days** | **Mon 8/19/24** | **Fri 8/30/24** | **20** |
| 22 | ➜ | 5.1 Optimize LSTM and CNN Models | 5 days | Mon 8/19/24 | Fri 8/23/24 | 20 |
| 23 | ➜ | 5.2 Optimize Random Forest Regressor | 5 days | Mon 8/19/24 | Fri 8/23/24 | 20 |
| 24 | ➜ | 5.3 Evaluate Model Performance | 5 days | Mon 8/26/24 | Fri 8/30/24 | 23,22 |
| 25 | ➜ | Model Optimization and Performance | 0 days | Fri 8/30/24 | Fri 8/30/24 | 24 |
| 26 | 📌 | **6. Documentation and Final Report** | **10 days** | **Mon 9/2/24** | **Fri 9/13/24** | **25** |
| 27 | ➜ | 6.1 Compile Final Report | 5 days | Mon 9/2/24 | Fri 9/6/24 | 25 |
| 28 | ➜ | 6.2 Prepare Presentation | 5 days | Mon 9/9/24 | Fri 9/13/24 | 27 |
| 29 | ➜ | Documentation and Final Report Milestone | 0 days | Fri 9/13/24 | Fri 9/13/24 | 28 |

Project: Real Time Stock Predic
Date: Tue 6/25/24

Figure 3.2: Project Schedule for Netflix Price & Subscriber Growth Prediction

## 3.5.2 Data Understanding

The Data Understanding phase involves collecting and analyzing the data to gain insight into its structure and quality. This phase is necessary to identify potential issues and understand the quality of the data, which determines the next step. The main objectives addressed in this section include:

1. **Stock Price Prediction:** Collecting historical stock price data for Netflix (NFLX) from Yahoo Finance.
2. **Subscriber Growth Rate Prediction:** Obtaining subscriber growth data from publicly available Netflix datasets.
3. **Visualization with Power BI:** Identifying and collecting additional data that will be useful for visualization and comparative analysis in Power BI.

Once the datasets are collected, preliminary analysis is performed to understand its structure, such as the number of records, variables, and any missing values. visualizations such as histograms, trendlines and scatter plots are used to explore the distribution and relationships among variables. Statistical parameters such as mean, standard deviation, skewness, and kurtosis help to understand the central tendency and dispersion of the data.

## 3.5.3 Data Preparation

Data Preparation is an important step in pre-processing the data to ensure that it is clean and suitable for modeling. This phase includes the data cleaning, transformation, and quality checks described below:

### 3.5.3.1 Data Cleaning

Data cleaning involves handling missing values, correcting errors, and eliminating unnecessary variables. Inconsistencies in the data are identified and addressed to improve its quality. Techniques such as imputation for missing values and outlier detection are used to ensure data robustness. This process ensures that the dataset is accurate and reliable for subsequent analysis and sampling.

In the stock price forecasting model, data cleaning includes passing historical stock price information through the yfinance library and ensuring that the date formats are accurate. MinMaxScaler was used to scale the stock price to normalize the data, making it suitable for input into the machine learning models.

For the subscriber growth rate prediction, the cleaning process involved standardizing date formats to ensure reliability, handling invalid dates by dropping rows with such values. This step ensured that the datasets were clean and ready for subsequent data transformation and sampling.

### 3.5.3.2 Data Transformation

The data transformation is done for the models. Data transformation involves preparing the data to meet the specific requirements of the machine learning models. This process includes transforming categorical variables into numerical values, scaling, and normalization. In the stock price prediction model, MinMaxScaler was used to scale the stock prices to normalize the data, and the data were rearranged to rank the past stock prices as input features.

For the subscriber growth rate prediction, categorical features such as 'Subscription Type', 'Country', 'Gender', 'Device', and 'Plan Duration' were transformed into numeric values using LabelEncoder. New features were created from existing data to provide more information for the model, such as splitting date columns into 'Join Year', 'Join Month', and 'Join Day', calculate monthly subscriber count, and growth rates. This transformation step ensured that the data were in the appropriate format for the machine learning models, thereby improving their performance and prediction accuracy.

### 3.5.3.3 Data Quality Checks

Finally, the data quality is checked to ensure that the cleaned and transformed data is ready for modeling. Visualization techniques such as heatmaps are used to ensure that all missing features are properly handled and that the data are consistent. This study confirms that the data are of high quality and suitable for building reliable prediction models. For example, in the subscriber growth rate prediction model, the correlation matrix and feature importance were analyzed to understand the relationships between variables and to confirm that the most important features were correctly identified and addressed. This quality control check ensured that the data used in both models were robust and reliable, resulting in plausible and reliable predictions.

## 3.5.4 Modeling

The modeling phase involves selecting and applying appropriate algorithms to the prepared data to develop predictive models. This phase involves selecting suitable algorithms, training models, and evaluating their performance. It is an iterative process that requires experimentation to identify the models that deliver the best results.

### 3.5.4.1 Data Partitioning

In the modeling phase, data partitioning plays an important role in ensuring that the predictive model is properly trained, validated, and tested. This process usually involves splitting the dataset into training, validation, and test sets. Training data are used to fit the model, validation data help tuning hyperparameters and reduce overfitting, and test data provide a final assessment of model performance. Cross-validation methods can be used to further assess model generalizability and robustness.

The traditional method of partitioning the data into discrete training and testing was not used for the stock price forecasting model. Instead, the model uses real-time data updates, which reflect the dynamics of financial markets. In this approach, the model is always trained on the most recent data, allowing it to adapt to new data as it becomes available. This approach is the real-world application of stock price forecasting, where continuous learning of the latest information is more useful and relevant than static rail test splitting

Whereas, for the prediction of subscriber growth, the distribution of data followed the usual pattern. The dataset was divided into training and test sessions to train the Random Forest Regressor model and evaluate its performance. This process required a training set to match the model and a test set to evaluate the prediction accuracy. The separation between training and testing methods allowed a comprehensive assessment of the model's ability to predict subscriber growth based on historical data. This traditional classification method is appropriate for the structured dataset and static nature of subscriber growth forecasting.

### 3.5.4.2 Model Selection

Model selection is an important phase in developing predictive models, and the model is chosen based on the data and the specific objectives of the study. The selection includes:

- **Stock Price Prediction:**
    - LSTM (Long Short-Term Memory)

    - CNN (Convolutional Neural Networks)
- **Subscriber Growth Prediction:**
    - Random Forest Regressor

For stock price recognition, Long Short-Term Memory (LSTM) network and Convolutional Neural Networks (CNNs) are selected. The LSTM model is particularly well suited for time series forecasting because it excels in capturing temporal dependencies in data series. It is designed to recall long-term exposures and patterns, which are necessary to forecast stock prices that are influenced by historical trends. The LSTM model used in this study consists of two LSTM layers followed by complex layers, making it capable of handling a series of events efficiently and predicting future values based on past data.

However, CNNs are used because of their ability to identify locations in the data. In stock price forecasting, CNNs are used to analyze temporal patterns by using time series data as a one-dimensional signal. Implemented CNN models include Conv1D layers with specified filters and kernel sizes, which are more efficient in extracting features from the data. MaxPooling1D layers are used to reduce dimensionality, and Flatten layers convert the pooled features to a format suitable for dense layers. This approach helps capture complex patterns and trends in stock prices.

To predict subscriber growth, a Random Forest Regressor was selected. Random Forests are powerful ensemble methods that perform well on large datasets and are effective on datasets with strong correlations and relationships between features. These models build different decision trees and combine their predictions, yielding reliable estimates of growth rates based on a variety of factors. The Random Forest Regressor was chosen because of its ability to handle data types that can generate noise associated with subscriber metrics and its robust performance in regression tasks.

Each model was selected based on its strengths in addressing the specific requirements of the prediction tasks, leveraging their unique capabilities to achieve accurate and meaningful results.

#### 3.5.4.3 Model Building and Training

Model building and training are key steps in the development of predictive models, which involve applying selected algorithms to the prepared data. This section focuses on how to properly build and prepare samples to achieve optimal performance. The tools used include:

- **LSTM and CNN Models:**
    - TensorFlow and Keras
- **Random Forest Regressor:**
    - Scikit-learn

TensorFlow and Keras are used to implement the LSTM and CNN models. The LSTM model architecture contains two LSTM layers. The first LSTM layer has 50 units and is set to **return_sequences=True** in order to control the sequence of the data, which allows the second LSTM layer to process all the sequences. A second LSTM layer, with **return_sequences=False**, collects the sequences and populates them with a vector of fixed size. In addition to the LSTM layers, two Dense layers are used: the first one has 25 units and the last Dense layer has one unit for the prediction. The model is compiled using an Adam optimizer and a mean squared error loss function appropriate for regression tasks. The model is trained with a batch size of 1 and 1 epoch, which implies a focus on iterative training due to the real-time nature of stock price forecasting, where frequent updates are considered more useful than extended training periods.

The CNN model, which was also implemented with TensorFlow and Keras, is designed to capture spatial patterns in time series data. It includes a Conv1D layer with 64 filters and a kernel size of 2, which uses the ReLU activation function to introduce non-linearity. This is followed by a MaxPooling1D layer with a pool size of 2 to reduce dimensionality and a Flatten layer to transform the pooled features into a one-dimensional vector. The last layer consists of Dense layers similar to the LSTM model, with the last Dense layer resulting in a single prediction. The CNN model is also compiled using Adam optimizer and mean squared error loss function, and trained with batch size of 1 and 1 epoch.

For subscriber growth prediction, Random Forest Regressor was used, implemented by Scikit-learn. This model is well suited for regression tasks involving large datasets with many features. The Random Forest model was trained with 100 counts (trees) and a total of 42 random conditions for reproducibility. The model was trained on data divided into training and test sets, with hyperparameters adjusted to improve performance and ensure reliable predictions of growth rates.

Overall, the model building and training phase involves developing LSTM and CNN algorithms to capture temporal and spatial patterns, respectively, and using Random Forest Regressor to handle complex feature interactions in subscriber growth prediction. The selection of model designs, sequences, and training sets is tailored to the specific characteristics and requirements of each forecasting task, resulting in more accurate and reliable forecasts.

## 3.5.5 Evaluation

The evaluation phase is essential for assessing the performance and readiness of our models, ensuring they meet the project's objectives and success criteria. This phase encompasses three main areas: verifying model readiness for deployment, evaluating performance using metrics, and interpreting results through visualizations.

### 3.5.5.1 Model Readiness

The evaluation process addresses the following project objectives which were defined in the earlier phases:

1. **Stock Price Prediction:** Develop and compare LSTM and CNN models to forecast Netflix (NFLX) stock prices.
2. **Subscriber Growth Rate Prediction:** Implement and evaluate a Random Forest Regressor model to predict subscriber growth rates across different regions.
3. **Visualization with Power BI:** Create and analyze visualizations for historical stock price movements, predicted subscriber growth rates, and comparative analysis using Power BI.

### 3.5.5.2 Evaluation Metrics

The models are evaluated using the following performance criteria:

1. **Mean Squared Error (MSE)**
   - MSE is the average of the squares of the differences between the actual and predicted values. It quantifies the error between the predicted values and the actual values [13]. **Figure 3.3** illustrates the formula for Mean Squared Error (MSE).

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Figure 3.3: Mean Squared Error (MSE) Formula

   - where n is the number of data points, $y_i$ is the actual value, and y^$_i$ is the predicted value.
   - A low MSE indicates that the model performs well, as it measures the average squared difference between actual and predicted values, correcting larger errors more significantly.

2. **Root Mean Squared Error (RMSE)**
   - RMSE is the square root of the MSE. It gives an error metric in the same units as the output variable, making it more interpretable in the context of the data [14]. **Figure 3.4** shows the formula for Root Mean Squared Error (RMSE).

$$\text{RMSE} = \sqrt{\text{MSE}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Figure 3.4: Root Mean Squared Error (RMSE) Formula

   - Lower RMSE exhibits more accurate predictions, resulting in error measures that can be interpreted in the same categories as the target variables, making them easier to understand.

**3. R-squared ($R^2$)**

- $R^2$ is also known as the coefficient of determination, measures the difference in the predictability of the dependent variable from the independent variables [15]. **Figure 3.5** displays the formula for the R-squared ($R^2$) metric.

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

Figure 3.5: R-squared ($R^2$) Formula

- where $SS_{res}$ is the residual sum of squares and $SS_{tot}$ is the total sum of squares.
- $R^2$ values close to 1 indicate that the model explains most of the variance in the target variable, while values near to 0 suggest poor predictive power.

### 3.5.5.3 Visualization & Interpretation

Visualization techniques are critical for evaluating model performance and making informed decisions about model selection. In stock price prediction and subscriber growth rate prediction, specific models are used to analyze and better interpret the results.

In stock price prediction, time series plots are used to compare actual and predicted stock prices over time. These plots help visually inspect how well the LSTM and CNN models follow stock price trends and identify any discrepancies.

For the subscriber growth rate prediction, bar charts shows the predicted and actual growth rates in each region, providing a clear comparison of model performance across different regions. Line graphs show subscriber growth rates over time, making it easy to visualize changes and future trends.

Additionally, correlation heatmaps reveal relationships between various factors that influence subscriber growth. Feature importance plots highlight the influence of different features on the prediction.

Power BI is utilized to create interactive dashboards that visualize the insights. Allowing for a broader understanding of model performance and the factors influencing subscriber growth trends. By utilizing these visualizations, the evaluation phase addresses key questions:

- How accurately do the LSTM and CNN models predict Netflix stock prices?
- How well does the Random Forest Regressor predict regional subscriber growth rates?
- How effectively do the Power BI visualizations convey insights from the models?

These diagrams not only provide a detailed assessment of the model's performance but also facilitate an informed decision to select the most effective model to use.

### 3.5.6 Deployment

The Deployment phase involves implementing the final models and making it more useable. This phase ensures that the predictions of the model can be used to make appropriate decisions.

A deployment plan is developed, which outlines the steps for implementing the models in a production environment. This plan includes:

- Using Power BI, interactive dashboards are created to visualize forecasts and provide users with insights. These dashboards allow for easy interpretation of results and facilitate data-driven decision-making.
- Monitoring and maintenance strategies to ensure that the models continue to function properly over time.
- Regular reviews are conducted to evaluate the performance of the models and make necessary adjustments based on new information and changing conditions.

The CRISP-DM methodology provides a structured and systematic approach to the data mining process, ensuring that all aspects are properly addressed. By following each phase in detail, from business understanding to deployment, Netflix stock prices and subscriber growth rates are predicted efficiently, providing valuable insights and supporting informed decision-making. Using Python, Google Colab, and advanced machine learning techniques, the project achieves high level of accuracy and reliability in predictions.

Below **Figure 3.6** provides a general overview of CRISP-DM phases and their respective tasks.

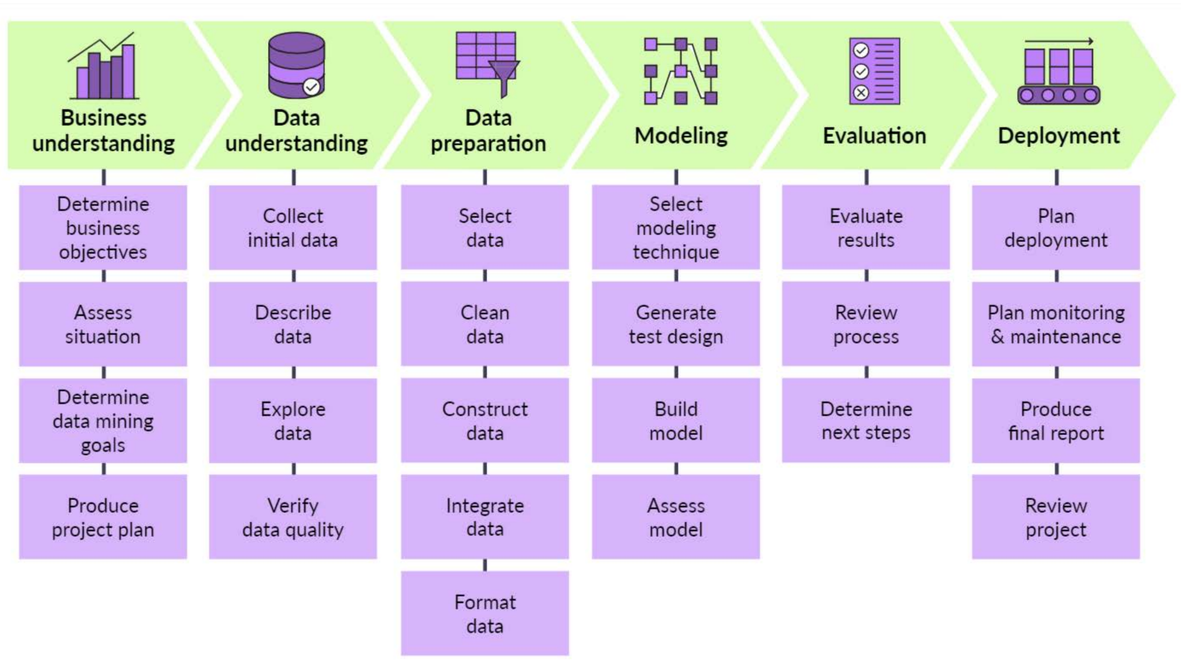


Figure 3.6: CRISP-DM Phases and Tasks

In this chapter, we focused on implementing the CRISP-DM phases through data preparation and analysis, which is an important step in setting up our research. In subsequent chapters, we will shift from the methods to a more focused discussion of how we use these methods in practice.

# Chapter 4

# Analysis & Design

In the previous chapter, we thoroughly explored the CRISP-DM approach, laying a solid foundation for our predictive modeling efforts. We focused on the data preparation and analysis phases, including data cleaning, adaptive transformation, and exploratory analysis. Having completed this foundational work, we are now ready to examine the analysis and design, which will be central to guiding our modeling efforts.

This chapter changes the focus from the methodology to a detailed analysis of how we analyze and develop our predictive models for Netflix's stock price and subscriber growth rates. We begin by categorizing the specific problems and objectives associated with our forecasting project and provide a clear understanding of the challenges we aim to address. This problem analysis will help refine our objectives and provide direction for our modeling efforts.

Next, we will analyze the system and model architecture, detailing the design and integration of the predictive models used in our work. This section will include a detailed flow diagram of the data processing pipeline, from initial data processing to final prediction. By understanding the structure of our models, we can appreciate how machine learning algorithms are used to obtain accurate and reliable predictions.

In addition, we will explore the various aspects of data management, focusing on how datasets are organized and prepared for analysis. This will require us to discuss data handling and design considerations that align with our research objectives. Understanding the data structure is important to ensure that the modeling process is based on well-structured and appropriate data.

Finally, we will summarize key insights and approaches from our analysis and design phases, and set the stage for subsequent chapters on modeling and evaluation. This summary will provide an integrated view of how the analysis and design work fits into the predictive modeling framework, and will establish the importance of these approaches in achieving the goals of our project.

By focusing on these design and evaluation elements, this chapter aims to establish a solid framework for the effective implementation and evaluation of the predictive models in Chapter 5 .

## 4.1 Problem Analysis

In this section, we examine the specific objectives and problems associated with our predictive model tasks for Netflix stock price and subscriber growth rates. This requires setting clear goals and understanding the challenges we face in achieving these goals.

### 4.1.1 Predictive Objectives

Our project is based on predicting Netflix stock prices and estimating subscriber growth rates. The project centered on the following main objectives:

1. **Predicting Netflix Stock Prices:**
   The goal is to use historical data to predict the future stock value of Netflix. This involves using past patterns and trends within the daily closing prices to build a robust forecasting model. By analyzing these historical data, the model aims to provide accurate predictions of future stock prices, identifying important patterns and trends.

2. **Estimating Subscriber Growth Rates:**
   The goal here is to predict future growth rates of Netflix subscribers based on historical trends and a variety of influencing factors. The main components are subscription type, region, and past growth. We used historical subscriber data, including monthly statistics and growth rates, to create a model that can estimate future growth rates. Including factors such as regional demographics and subscriptions contributes to the model's accuracy in predicting an increase in subscribers.

3. **Visualization of Data and Predictions:**
   Another key goal is to better visualize the data and predictions with Power BI. This requires interactive and informative dashboards that provide insight into Netflix stock price and subscriber growth. The visualizations help stakeholders understand trends, compare actual and forecasted values, and make data-driven decisions. We created a series of Power BI charts, such as line charts of stock price growth and bar charts of subscriber growth, to clearly show and communicate our findings.

### 4.1.2 Challenges

Several challenges need to be addressed to achieve above discussed objectives:

1. **Data Quality and Completeness:**
   Ensuring the accuracy and completeness of the dataset is critical. This requires handling missing values, identifying anomalies, and correcting data inconsistencies. In our method, we performed data cleaning by inputting missing values using appropriate methods and identifying outliers to ensure the accuracy of the dataset. For example, missing values in the number of subscribers were fixed based on historical data, while excess was handled to prevent skewed results.

2. **Feature Selection:**
   Identifying relevant factors that have the greatest impact on predictions is critical for model performance. In stock price forecasting, factors such as historical prices are selected based on their ability to capture trends and trends over time. Similarly, in predicting subscriber growth, factors such as country, subscriber type, and device type are selected based on their potential impact on growth rates. The importance of these factors was carefully assessed to ensure that they contributed meaningfully to the predictive power of the model.

3. **Regional Variations in Subscriber Growth:**
   Understanding regional changes is important for accurate subscriber growth rate predictions. Regions may exhibit development patterns that are influenced by local factors. We disaggregated the subscriber data by region to examine these variables and improve the model's ability to accurately predict growth rates. By incorporating local factors and historical trend data, we developed the model to account for regional differences and maximize forecast accuracy.

4. **Effective Data Visualization:**
   Creating effective visualizations is critical to delivering insights from data and predictions. Challenges here include choosing appropriate visualization techniques and ensuring that visualizations are clear and interpretable. We addressed these challenges by creating a variety of charts and dashboards in Power BI, such as line charts of stock prices and bar charts of growth rates. These graphs help present complexity and predictions in a logical way.

By addressing these objectives and challenges, we lay a solid foundation for the next phase of model design and development, ensuring that our predictive models are properly calibrated and that our visualizations effectively communicate the insights gain from the data.

## 4.2 System / Model Architecture

The design of our system for predicting Netflix stock price and subscriber growth is designed to optimize the data pipeline, from initial data collection to the point where the data is ready for predictive modeling. The system is modular, with a series of interconnected components, each performing a specific task in the data pipeline. This modular design ensures that each part of the process is handled independently and can be executed efficiently without affecting the overall business process.

### 4.2.1 System Overview

The system has several key components, each of which plays an important role in transforming raw data into actionable insights. These components include data processing, preprocessing, feature engineering, model training and validation, prediction and analysis, visualization and reporting. By integrating these features into an interconnected system, we can maintain a smooth flow of data and ensure the reliability of the forecasts. To conduct this study, we followed a systematic approach to refine and improve our implementation process. The system overview is shown in **Figure 4.1** and includes the following steps:

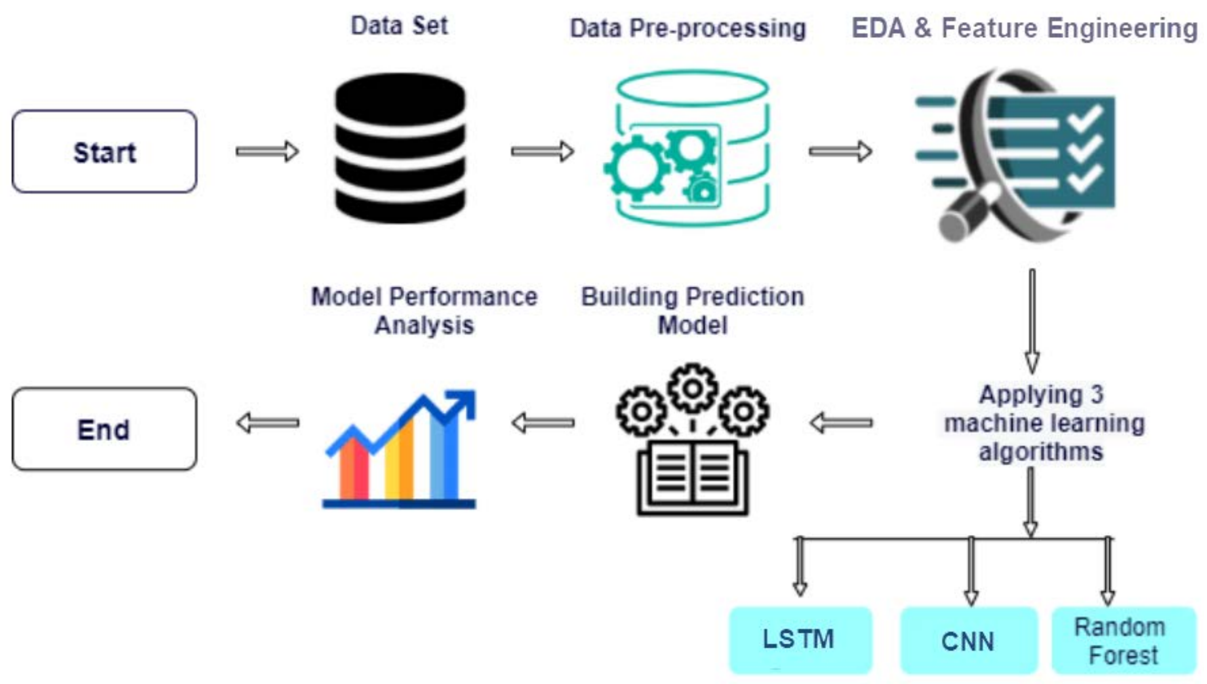

Data Set
Data Pre-processing
EDA & Feature Engineering
Start
Model Performance Analysis
Building Prediction Model
End
Applying 3 machine learning algorithms
LSTM
CNN
Random Forest

Figure 4.1: System Overview

1. **Data Ingestion:**
Data is collected from a variety of sources, including historical stock prices using the Yahoo Finance API and subscriber data from a Kaggle dataset as a CSV file. The data is then loaded into the system for further processing.

2. **Data Processing and Cleaning:**
Once processed, the data are preprocessed to address missing values, outliers, and inconsistencies. This step is necessary to ensure the quality of data used in predictive models. The data is cleaned and ready for further analysis.

3. **Exploratory Data Analysis (EDA) and Feature Engineering:**
Exploratory Data Analysis (EDA) is conducted to understand data structure and distribution. In this phase, a special step is taken to enhance the predictive power of the models. This includes generating time-series features for stock prices and calculating growth rate metrics for subscriber data.

4. **Applying Machine Learning Models:**
Predictive models are developed and trained using different algorithms. For stock price prediction, Long Short-Term Memory (LSTM) and Convolutional Neural Networks (CNN) are used. To predict the increase in subscribers, models such as

Random Forest are used. The model is trained on historical data and validated by business decisions.

5. **Prediction and Evaluation:**
After training, the models make predictions based on new or unseen information. These predictions are compared with actual values to assess model performance. Metrics such as RMSE, R-squared, and others are used to quantify the accuracy of forecasts.

6. **Model Performance Analysis:**
The results are analyzed to assess the effectiveness of each model. This includes analyzing the accuracy of the forecast, the robustness of the model, and visualizing the results using tools such as Power BI. The insights gained are used to refine the models and improve future predictions.

## 4.2.2 Flow Diagrams

Both the systematic/modeling processes for Netflix stock price forecasts and subscriber growth rate forecasts involve a series of well-defined steps that transform raw data into actionable insights. **Figure 4.2** and **Figure 4.3** show the different architectures used for these two prediction tasks. **Figure 4.2** details the framework for predicting Netflix stock price.

The process starts with **Data Loading**, where the **yfinance** API for the NFLX ticker retrieves historical and real-time stock price data. This data includes fields like Open, High, Low, Close, Adj Close and Volume, which are important for analysis.

In **Data Preprocessing** step, the **create_dataset()** function is used to create a data set from the raw stock value. This sequence is initially in 2D format (number of samples x time step). To make the data suitable for model input, it is reconstructed into a 3D array of dimensions (patterns, time steps, features). This restructuring is important because LSTM models expect 3D input to handle the sequential characteristics of time-series data, while CNNs also require 3D input to perform convolutional processing in terms of both time steps and features. The data is scaled using the MinMaxScaler to normalize the values, ensuring consistency in feature scaling.

**Modeling** involves two main techniques, LSTM and CNN. The LSTM model has an LSTM layer of 50 units, followed by two Dense layers of 25 and 1 unit(s). The model is trained with a batch size of 1 and epoch 1, with the aim of predicting stock prices from historical data. Similarly, the CNN model includes Conv1D and MaxPooling1D layers in addition to Dense layers, which are configured with the same training parameters. Both models output stock price predictions based on preprocessed data. For predictions, the LSTM model provides **lstm_predictions**, while the CNN model provides **cnn_predictions**. These forecasts show the stock price forecast provided by each model, based on input historical data.

In **Evaluation & Deployment** process, the performance of the model is evaluated using metrics such as MSE and RMSE. Results are visualized through Power BI, providing interactive dashboards to demonstrate and verify predictions.

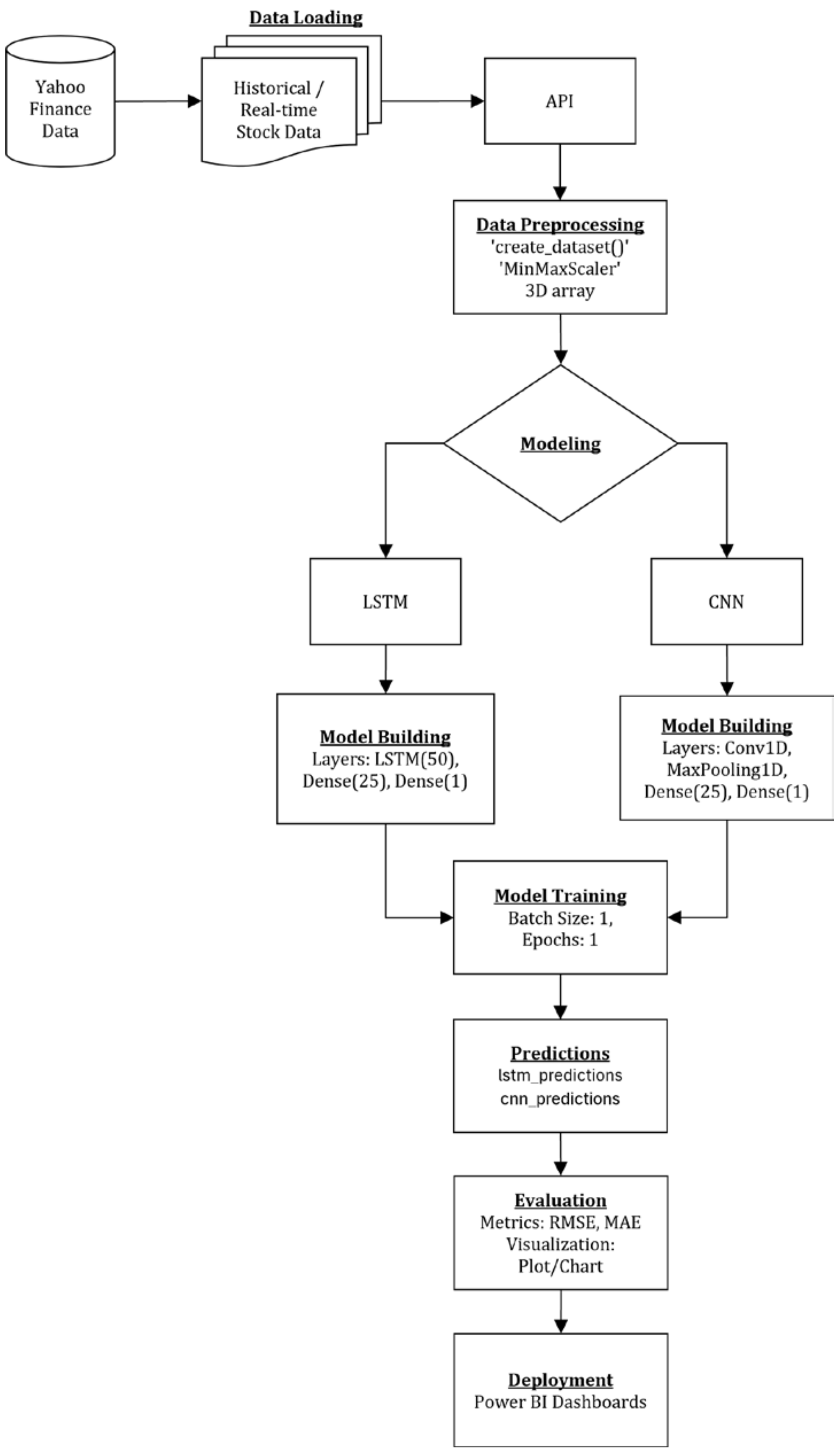


Figure 4.2: System Architecture for Netflix Stock Price Prediction

Below **Figure 4.3** outlines the architecture for predicting subscriber growth rates. The process begins with Data Loading, where subscriber data is imported from CSV files. Data is integrated through manual input.

**Data Analysis & Preprocessing** involves several critical steps to prepare the data for modeling. Initially, the data is cleaned and transformed to handle missing values, outliers, and inconsistencies, ensuring high-quality inputs for analysis. This includes handling any data integrity issues and standardizing the data format. Once the data is cleaned, various visualizations are employed to gain insights and understand the data better. This involves plotting charts and graphs to explore patterns, trends, and relationships within the data. Visualization tools help in identifying significant features and understanding the distribution of subscriber growth across different regions and time periods. Following this exploratory data analysis, monthly subscriber count and growth rates are computed. This step provides a clear view of subscriber behavior over time. Categorical features such as Country, Subscription Type, Gender, Device, and Plan Duration are encoded using label encoding to convert them into a numerical format suitable for machine learning models. The data is then split into training and test sets for model evaluation.

In the **Modeling** phase, we used the Random Forest algorithm to predict subscriber growth rates. The features included in the model are Country, Subscription Type, Monthly Revenue, Age, Gender, Device, Plan Duration, and Monthly Subscriber Count, with the Growth Rate (%) as the target variable.

The **Model Training** involves training the Random Forest model to predict the Growth Rate (%) through the specified features. The model is trained on the training dataset, and its performance is evaluated by comparing predictions with test data to ensure accuracy and efficiency.

In **Evaluation & Deployment** phase, metrics such as R-squared and RMSE are used to measure model performance. Diagrams are made to compare predictions with actual values and to emphasize their importance. The final model is exported for integration with Power BI, which simplifies interactive analysis and reporting.

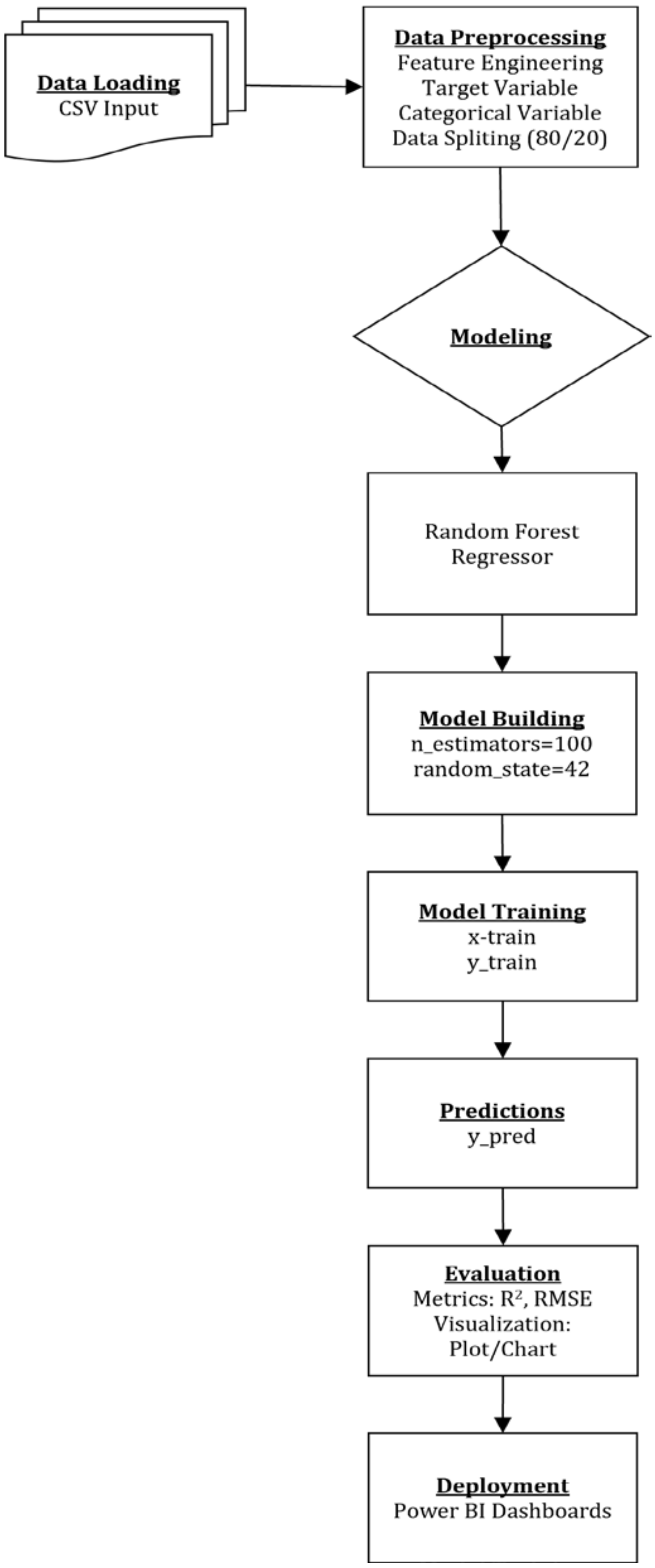


Figure 4.3: System Architecture for Subscriber Growth Rate Prediction

These diagrams provide a framework of the system architecture for both predictive tasks, and show the data flow, model building, and evaluation processes importance for reliable and actionable forecasting.

## 4.3 Data Understanding & Exploration

In this phase of the analysis, we begin with a thorough analysis of the data to establish a solid foundation for our predictive modeling efforts. This requires a description of the datasets we are using, including their origin, size, format, and the quality of their records. We provide a detailed description of the data type and transformation, ensuring a clear understanding of the structure and content of each dataset. After this preliminary analysis, we proceed to the exploratory phase, where we conduct a visual analysis and statistical summary. The purpose of this study is to reveal key insights, identify patterns, and explore relationships between the independent variables and the target variables. This step is important for guidance in subsequent sampling procedures and for proper use of the data.

### 4.3.1 Data Description

This section provides an overview of the datasets used in our work. It details the data sources, characteristics and variables for each dataset and provides a detailed description of both Netflix stock price and subscriber growth data. This foundation is important for understanding the structure and content of the data, which will inform subsequent analysis and model development.

#### 4.3.1.1 Data Sources

- **Netflix Stock Prices**: Used the yfinance API to obtain historical stock price data from Yahoo Finance. This dataset provides detailed information about Netflix's stock performance over time.
- **Subscriber Growth Data**: Subscriber data was obtained from a CSV file on Kaggle. This dataset includes detailed records of subscriber metrics, providing insights into growth trends and revenue.

#### 4.3.1.2 Data Details

- **Netflix Stock Prices:** The dataset contains several characteristics important for modeling stock prices. **Table 4.1** provides these characteristics and a brief description.

| ATTRIBUTE | DESCRIPTION |
|---|---|
| **DATE** | Date of the stock price record |
| **OPENING PRICE** | Price of the stock at market open |
| **CLOSING PRICE** | Price of the stock at market close |
| **HIGH** | Highest price of the stock during the day |
| **LOW** | Lowest price of the stock during the day |
| **VOLUME** | Number of shares traded |

Table 4.1: Attributes & Descriptions of Netflix Stock Prices Dataset

- **Subscriber Growth Data:** The subscriber growth dataset includes attributes important for analyzing subscriber trends. **Table 4.2** includes these characteristics along with its description.

| ATTRIBUTE | DESCRIPTION |
|---|---|
| **USER ID** | Unique identifier for each subscriber |
| **SUBSCRIPTION TYPE** | Type of subscription (e.g., Basic, Standard, Premium) |
| **MONTHLY REVENUE** | Revenue generated per month from the subscriber |
| **JOIN DATE** | Date when the subscriber joined |
| **LAST PAYMENT DATE** | Date of the most recent payment |
| **COUNTRY** | Geographic location of the subscriber |
| **AGE** | Age of the subscriber |
| **GENDER** | Gender of the subscriber |
| **DEVICE** | Type of device used by the subscriber (Laptop, Smart TV, Smart Phone, Tablet) |
| **PLAN DURATION** | Duration of the subscription plan |

Table 4.2: Attributes & Descriptions of Subscriber Growth Dataset

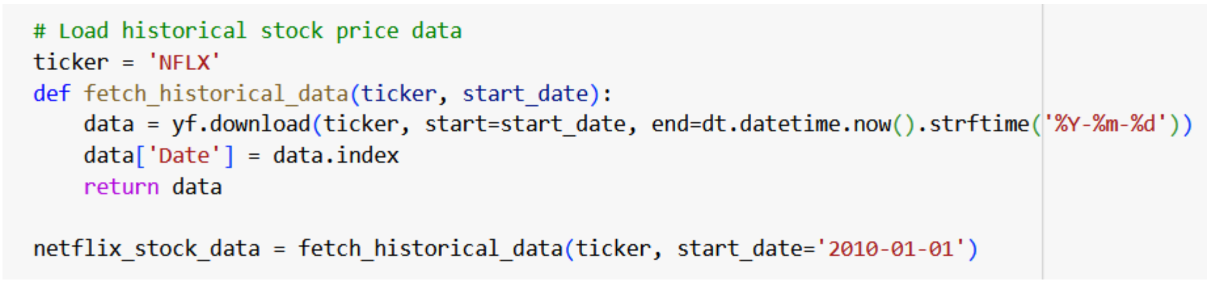
Data from these sources were thoroughly examined and processed to ensure their suitability for analysis. This includes dealing with missing values, performing data cleansing, and performing Exploratory Data Analysis (EDA) analysis to gain insights into trends and relationships in the data.

## 4.3.2 Data Loading

Before starting the exploratory analysis, we first loaded and reviewed the data. Below are the steps we went through to load the data, along with the statistics showing the code and the results.

### 4.3.2.1 Netflix Stock Prices

The Netflix stock price data was obtained from Yahoo Finance and was entered using the yfinance library in Python. The data includes historical records with attributes such as Date, Open, High, Low, Close, and Volume. Below **Figure 4.4** shows the Python code to retrieve and load the Netflix stock price data.

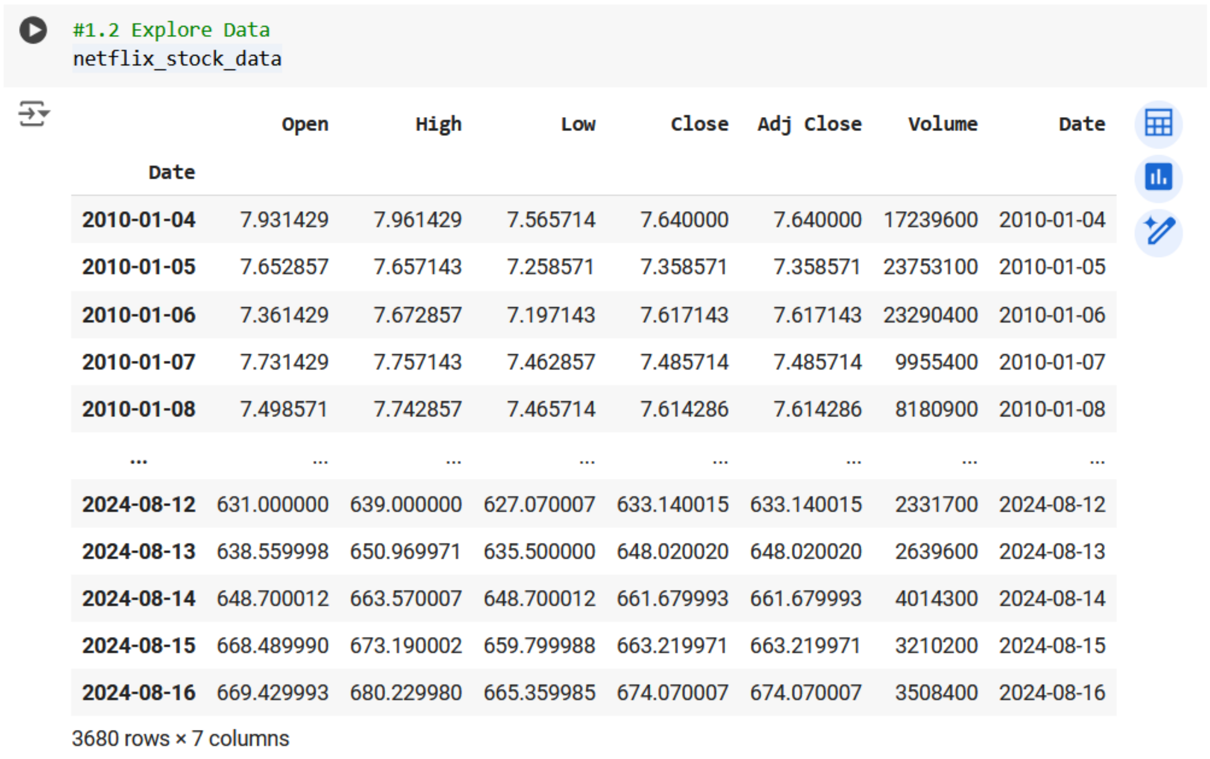


```python
# Load historical stock price data
ticker = 'NFLX'
def fetch_historical_data(ticker, start_date):
    data = yf.download(ticker, start=start_date, end=dt.datetime.now().strftime('%Y-%m-%d'))
    data['Date'] = data.index
    return data

netflix_stock_data = fetch_historical_data(ticker, start_date='2010-01-01')
```

Figure 4.4: Code for Loading Netflix Stock Data

This code uses the **yfinance** library to download historical stock price data for Netflix. The **fetch_historical_data** function retrieves the data from Yahoo Finance, ensuring its spans from January 2010 to the current date. The data is stored in the **netflix_stock_data** data frame, with the Date column selected as the index.

**Figure 4.5** presents a code snippet accompanying the output, showing the initial rows of the input dataset using **netflix_stock_data**. The dataset consists of 3,680 rows and 7 columns.



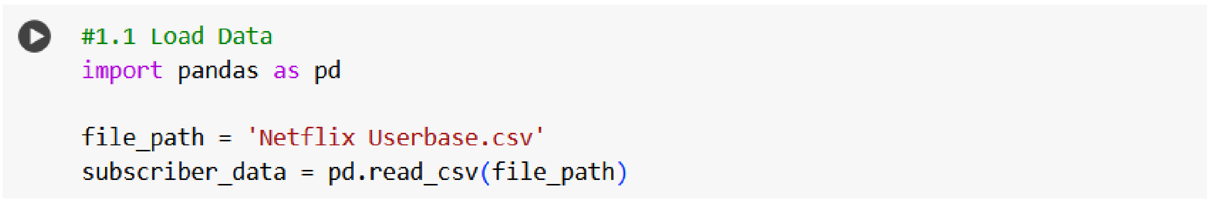
```
#1.2 Explore Data
netflix_stock_data
```

```
                 Open        High         Low       Close   Adj Close    Volume        Date
      Date
2010-01-04   7.931429    7.961429    7.565714    7.640000    7.640000  17239600  2010-01-04
2010-01-05   7.652857    7.657143    7.258571    7.358571    7.358571  23753100  2010-01-05
2010-01-06   7.361429    7.672857    7.197143    7.617143    7.617143  23290400  2010-01-06
2010-01-07   7.731429    7.757143    7.462857    7.485714    7.485714   9955400  2010-01-07
2010-01-08   7.498571    7.742857    7.465714    7.614286    7.614286   8180900  2010-01-08
    ...           ...         ...         ...         ...         ...       ...         ...
2024-08-12 631.000000  639.000000  627.070007  633.140015  633.140015   2331700  2024-08-12
2024-08-13 638.559998  650.969971  635.500000  648.020020  648.020020   2639600  2024-08-13
2024-08-14 648.700012  663.570007  648.700012  661.679993  661.679993   4014300  2024-08-14
2024-08-15 668.489990  673.190002  659.799988  663.219971  663.219971   3210200  2024-08-15
2024-08-16 669.429993  680.229980  665.359985  674.070007  674.070007   3508400  2024-08-16
3680 rows × 7 columns
```

Figure 4.5: Output of Loaded Netflix Stock Data

#### 4.3.2.2 Netflix Subscriber Growth

To obtain Netflix subscriber trend data, the following Python code in **Figure 4.6** was used to read a CSV file with attributes for subscribers. The data is loaded into a Pandas Data Frame for later analysis.

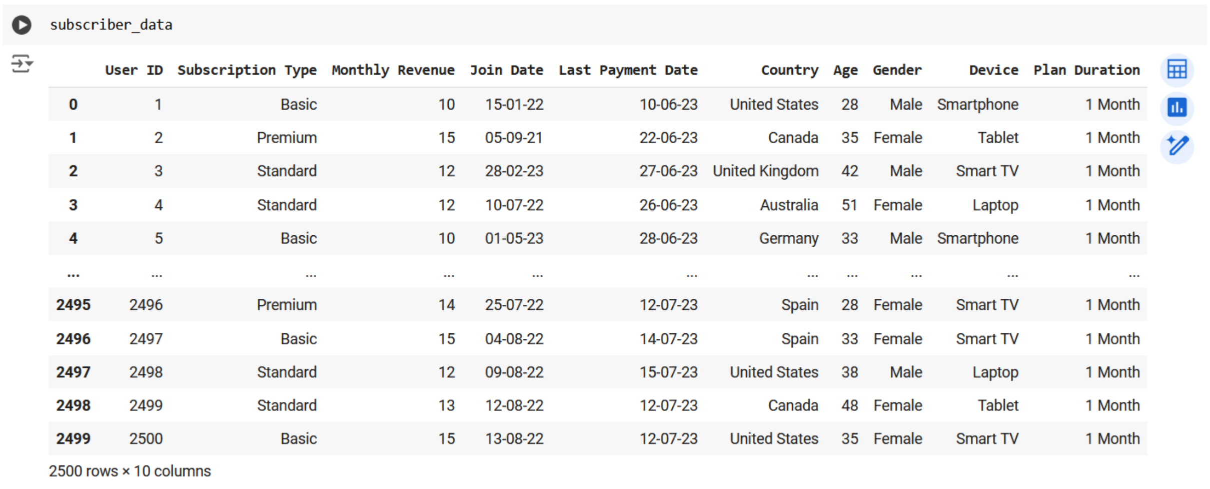


```
#1.1 Load Data
import pandas as pd

file_path = 'Netflix Userbase.csv'
subscriber_data = pd.read_csv(file_path)
```

Figure 4.6: Code for Loading Netflix Subscriber Growth Data

In the code provided, the **pd.read_csv()** function is used to load the subscriber growth data from the CSV file into a Pandas data frame.

To ensure that the data is loaded correctly, the data frame **subscriber_data** is examined directly, and its contents are shown in **Figure 4.7**. This dataset contains 2,500 rows and 10 columns allowing a first look at the structure and value of the dataset.



```
subscriber_data
```

| | User ID | Subscription Type | Monthly Revenue | Join Date | Last Payment Date | Country | Age | Gender | Device | Plan Duration |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | Basic | 10 | 15-01-22 | 10-06-23 | United States | 28 | Male | Smartphone | 1 Month |
| **1** | 2 | Premium | 15 | 05-09-21 | 22-06-23 | Canada | 35 | Female | Tablet | 1 Month |
| **2** | 3 | Standard | 12 | 28-02-23 | 27-06-23 | United Kingdom | 42 | Male | Smart TV | 1 Month |
| **3** | 4 | Standard | 12 | 10-07-22 | 26-06-23 | Australia | 51 | Female | Laptop | 1 Month |
| **4** | 5 | Basic | 10 | 01-05-23 | 28-06-23 | Germany | 33 | Male | Smartphone | 1 Month |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **2495** | 2496 | Premium | 14 | 25-07-22 | 12-07-23 | Spain | 28 | Female | Smart TV | 1 Month |
| **2496** | 2497 | Basic | 15 | 04-08-22 | 14-07-23 | Spain | 33 | Female | Smart TV | 1 Month |
| **2497** | 2498 | Standard | 12 | 09-08-22 | 15-07-23 | United States | 38 | Male | Laptop | 1 Month |
| **2498** | 2499 | Standard | 13 | 12-08-22 | 12-07-23 | Canada | 48 | Female | Tablet | 1 Month |
| **2499** | 2500 | Basic | 15 | 13-08-22 | 12-07-23 | United States | 35 | Female | Smart TV | 1 Month |

2500 rows × 10 columns

Figure 4.7: Output of Loaded Netflix Subscriber Growth Data

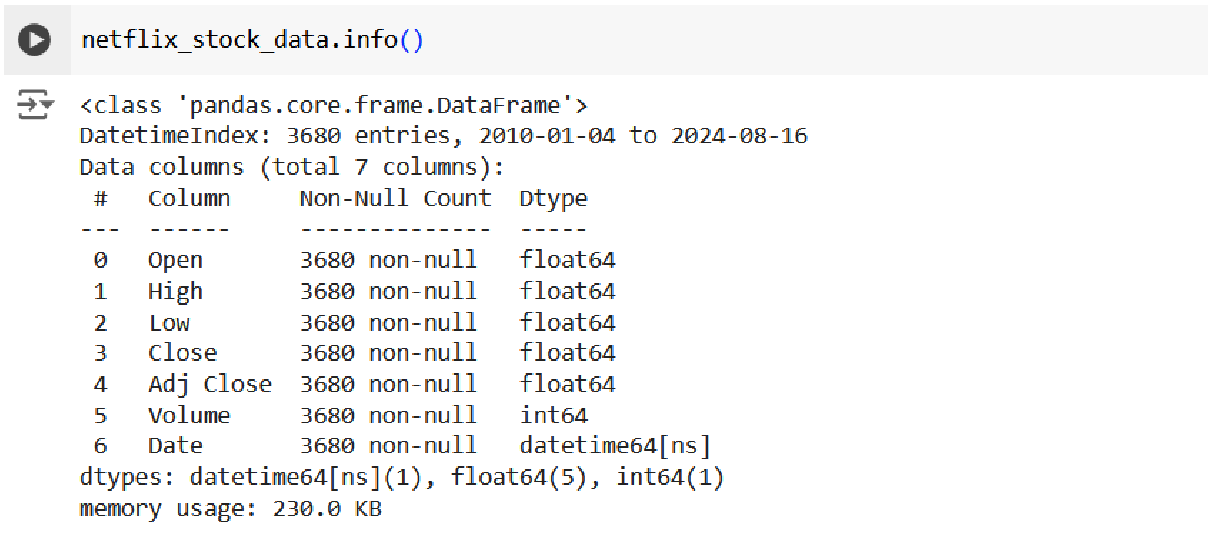
## 4.3.3 Exploratory Analysis

Exploratory Data Analysis (EDA) is essential to understanding the structure and nature of the data, and enables us to identify key patterns, trends, and relationships. In this section, we detail the EDA conducted on Netflix stock price data and Netflix Subscriber growth data, and integrate Python for advanced analysis and Power BI for interactive visualizations. This process provided important insights that informed our predictive modeling efforts.

### 4.3.3.1 Data Exploration

#### 1. Netflix Stock Prices

We started by analyzing Netflix stock price data to understand its structure and content. The **netflix_stock_data** dataset was parsed using the **info()** function, which provided information about data types, missing values, and key statistics. This first step, illustrated in **Figure 4.8**, confirmed the completeness and accuracy of the data, and ensured that it was ready for further analysis.



```
netflix_stock_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 3680 entries, 2010-01-04 to 2024-08-16
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Open       3680 non-null   float64
 1   High       3680 non-null   float64
 2   Low        3680 non-null   float64
 3   Close      3680 non-null   float64
 4   Adj Close  3680 non-null   float64
 5   Volume     3680 non-null   int64
 6   Date       3680 non-null   datetime64[ns]
dtypes: datetime64[ns](1), float64(5), int64(1)
memory usage: 230.0 KB
```

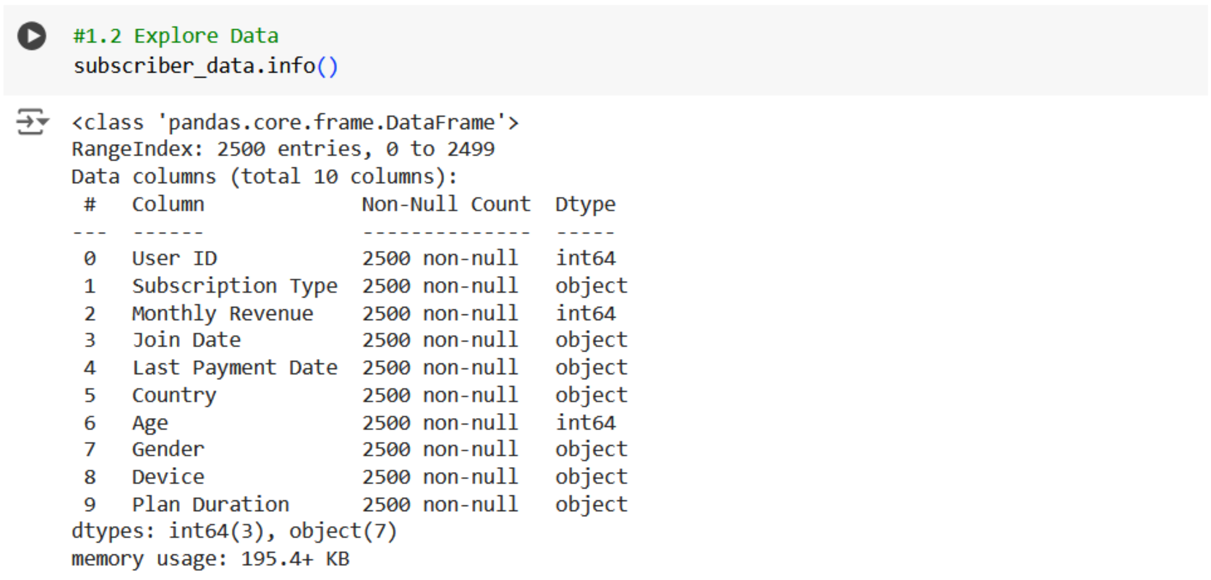
Figure 4.8: Overview of Netflix Stock Price Data Structure

**Figure 4.8** provides a detailed description of the Netflix stock price dataset. The dataset contains 3,680 events, from January 4, 2010, to January 16, 2024. It has 7 columns: Open, High, Low, Close, Adj Close, Volume, and Date. All columns are complete with no missing values, and the data types include float64 for value-related columns and datetime64[ns] for Date columns. These parameters confirm that the dataset is ready for subsequent exploratory analysis and sampling.

**2. Netflix Subscriber Growth**

The analysis of subscriber growth data required similar steps to understand the characteristics and characteristics of the dataset. Using the **info()** function on the **subscriber_data** dataset, we checked for important attributes and checked for missing values. This preliminary analysis is necessary to ensure the accuracy of the data before continuing with further research.

Below **Figure 4.9** provides a summary of the **subscriber_data** dataset, revealing that it contains 2,500 10-column fields, including attributes such as User ID, Subscription Type, Monthly Revenue, and others. The dataset has no missing values, and the data types include int64 for numeric columns, int of integer values and object for categorical columns. This general perspective supports the information prepared for in-depth analysis.



```
#1.2 Explore Data
subscriber_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2500 entries, 0 to 2499
Data columns (total 10 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   User ID            2500 non-null   int64
 1   Subscription Type  2500 non-null   object
 2   Monthly Revenue    2500 non-null   int64
 3   Join Date          2500 non-null   object
 4   Last Payment Date  2500 non-null   object
 5   Country            2500 non-null   object
 6   Age                2500 non-null   int64
 7   Gender             2500 non-null   object
 8   Device             2500 non-null   object
 9   Plan Duration      2500 non-null   object
dtypes: int64(3), object(7)
memory usage: 195.4+ KB
```

Figure 4.9: Overview of Subscriber Growth Data Structure

### 4.3.3.2 Data Visualization in Google Colab

In this section, we explore the use of Python in Google Colab to visualize Netflix stock price and subscriber growth rate data. The visualizations provided a detailed analysis of the patterns and trends in the data, and provided a foundation for our predictive analysis. The section covers two main areas:

1. **Netflix Stock Prices:** Using the Python libraries in Google Colab, we created plots to analyze Netflix stock price data. These included a linear chart of historical price movements, a candlestick chart of long-term price fluctuations, and area chart of long-term price trends. These graphs helped identify key patterns, changes, and trends, and provided a deeper understanding of how Netflix stock price has moved.

2. **Netflix Subscriber Growth Rate:** Similarly, we used Python to create graphs for Netflix's subscriber growth data. This required line charts showing subscriber trends over time and bar charts to compare the number of subscribers in different regions. These graphs allowed us to examine subscriber growth in more detail, revealing insights into the drivers of change in Netflix's user base.

Overall, the visualizations in Google Colab played an important role in our exploratory analysis, helping inform our modeling efforts and enhancing our understanding of the behavior of the data.

**Netflix Stock Prices:**

For Netflix stock prices plots below were created using python in Google Colab to reveal patterns and underlying trends in Netflix stock price data:

1. **Time Series Plot of Netflix Stock Prices:** The plot depicted in **Figure 4.10** visualizes the evolution of Netflix stock price over time. It provides a clear view of the stock's historical performance, highlighting volatility and long-term trends. In the early years, the stock was relatively undervalued, reflecting the company's initial market presence. Over time, its price gradually increased, indicating the company's growth and expansion in the market. Significant peak values were observed in early 2022, indicating an all-time high in stock prices. However, beyond this peak, the value of the stock fell sharply after 2023, highlighting a period of market correction or firm-specific challenges. This plot gives a clear picture of Netflix's historical performance, highlighting both the volatility and long-term trends in its stock price.

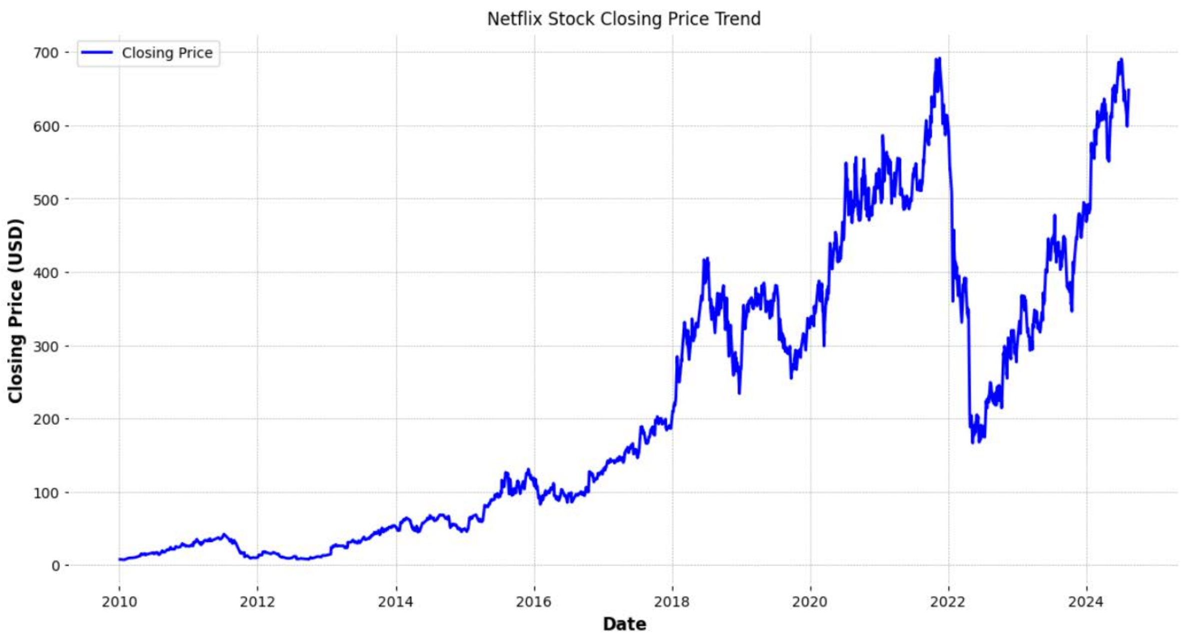


Figure 4.10: Time Series Plot of Netflix Stock Closing Prices

2. **Candlestick Plot with Moving Averages:** The candlestick plot shown in **Figure 4.11** incorporates both 20-day (MA20) and 30-day (MA30) moving averages. The MA20 line provides a rough idea of a stock's price action over the past 20 days, highlighting short-term trends and helping to identify recent trends. Meanwhile, the MA30 line provides a longer-term perspective, smoothing 30-day price volatility to reflect consistent trends. Taken together, these continuous distributions help to analyze price action, filter out short-term volatility, and better understand long-term trends.

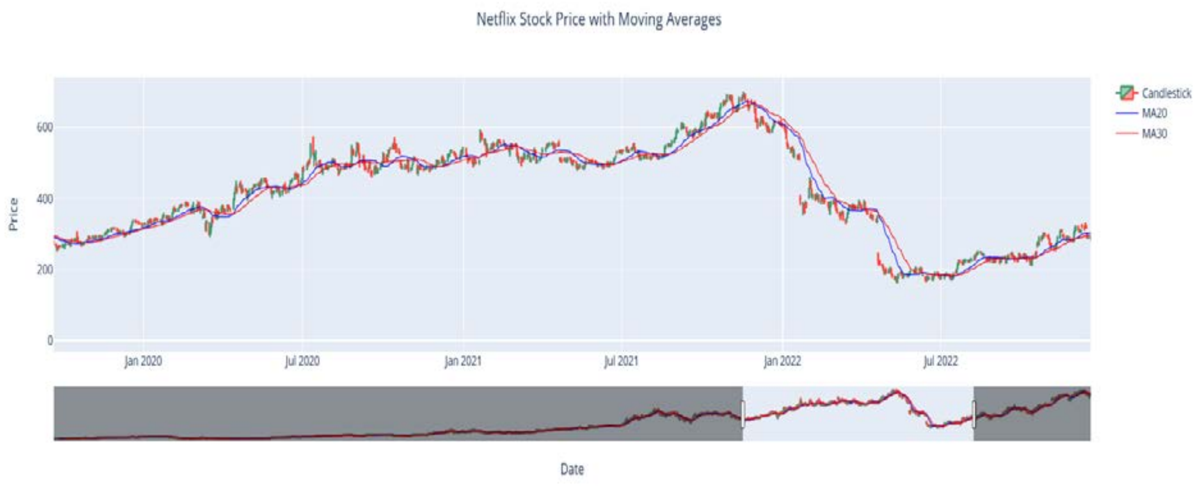


Figure 4.11: Candlestick Plot with 20-Day and 30-Day Moving Averages

3. **Daily Returns Plot: Figure 4.12** shows the daily returns in Netflix stock, calculated as a percentage change in closing price. This plot is useful for stock volatility analysis and understanding daily business fluctuations. We can see significant growth in 2012 and early 2022, marking periods of significant negative returns. Conversely, a sharp increase in daily profitability is evident in 2013, reflecting a period of greater efficiency. This analysis helps to identify periods of significant change and to understand the factors responsible for these changes.

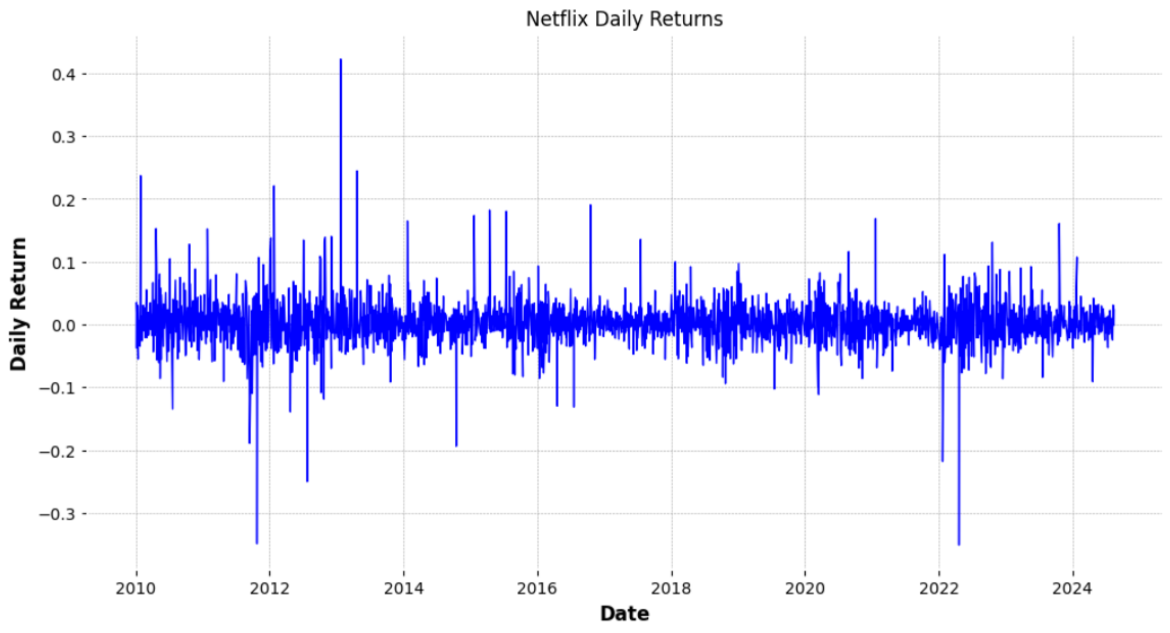


Figure 4.12: Daily Returns of Netflix Stock

4. **Stock Prices and Trading Volume Plot: Figure 4.13** shows a dual-axis plot of stock price versus trading volume. This diagram allows us to examine the relationship between price movements and trading activity. Notably, in 2012, when trading volume was high, stock prices were relatively weak, mirroring previous years' price declines.

Conversely, from 2016 to 2022, stock prices rose even as trading volume remained low, suggesting a period of higher prices despite a decline in trading activity. This indicates that inflation was not significantly related to trading volume in these later years.

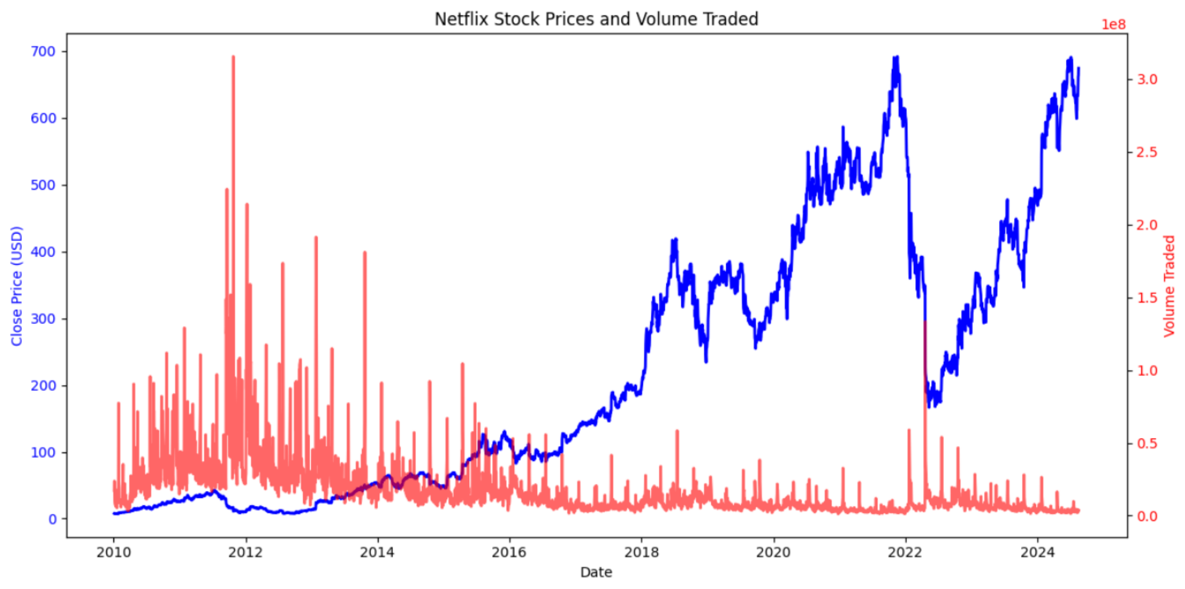


Figure 4.13: Dual-Axis Plot of Stock Prices and Trading Volume

5. **Correlation Matrix Heatmap:** The heat map in **Figure 4.14** shows the relationships between the various statistical features in the dataset. It reveals the strength of relationship between attributes such as opening price, closing price, and volume. High correlations between variables may indicate significant relationships that can influence stock prices. Notably, the close price has the lowest correlation with volume, around -0.5, indicating an inverse and relatively weak relationship between these two variables.

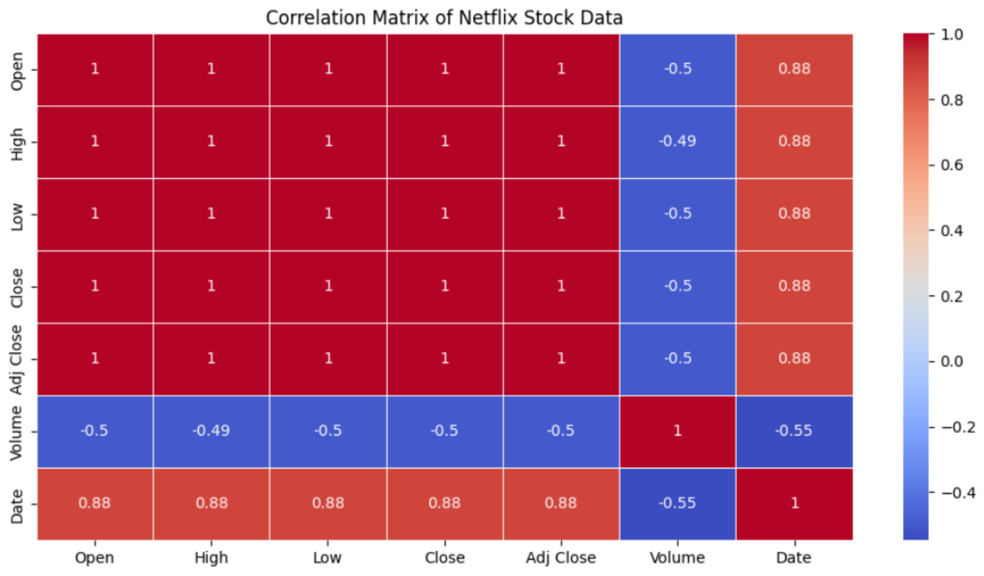


Figure 4.14: Correlation Matrix Heatmap of Netflix Stock Data

**Netflix Subscriber Growth Rate:**

For subscriber growth rate data, Python was used to create visualizations to gain insights into user demographics, subscription types and revenue distribution:

1. **Number of Users by Country:** The bar chart in **Figure 4.15** shows the number of Netflix users in each country. By aggregating data by country and counting the number of users, this chart gives a clear picture of the global distribution of Netflix subscribers. It reveals that countries such as Canada, Spain and the United States have the highest number of users. This visualization is useful for identifying regional market sizes and key geographic areas with significant user bases.

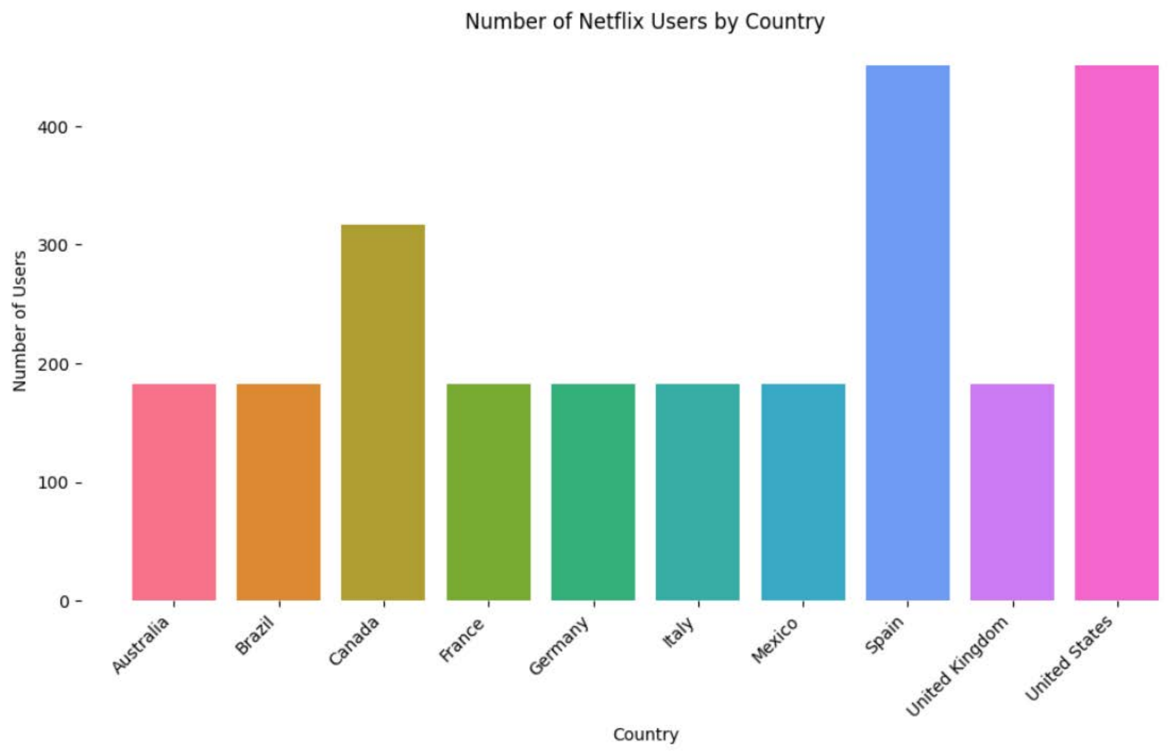


Figure 4.15: Number of Netflix Users by Country

2. **Distribution of Subscribers by Age: Figure 4.16** shows the age distribution of Netflix subscribers by histogram. This graph reveals a wide range of ages among Netflix users, with significant positions in different age groups. The most notable age group is the 30s to 40s, although there are also significant populations enrolled in the 40s-50s. Understanding this age distribution is valuable for improving user engagement strategies and aligning content according to different age groups.

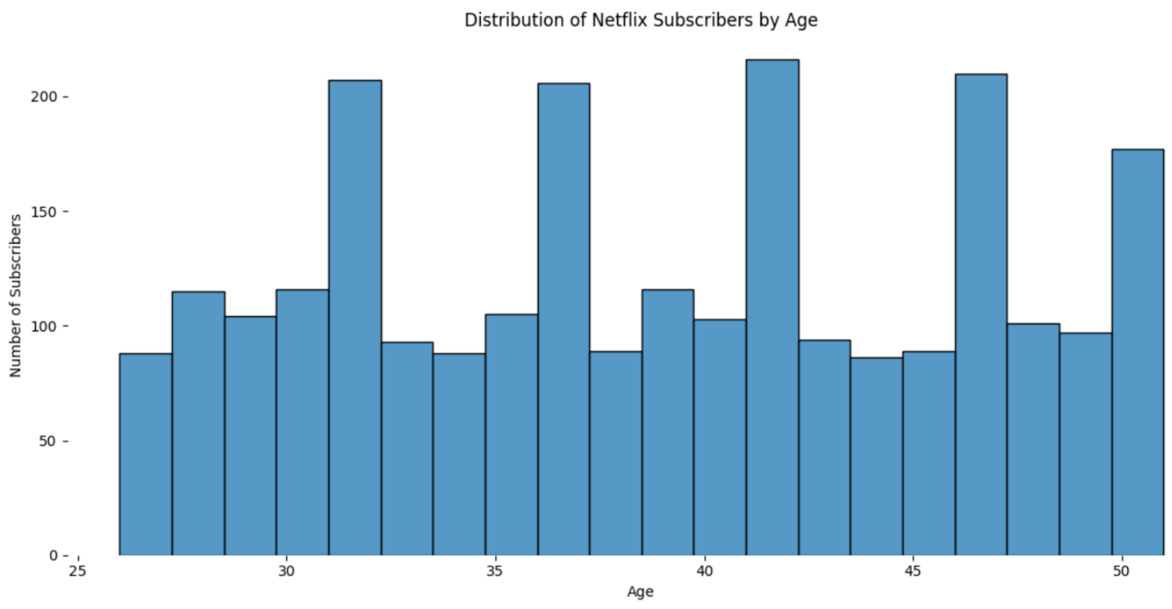


Figure 4.16: Distribution of Netflix Subscribers by Age

3. **Subscription Type Distribution: Figure 4.17** shows how Netflix subscribers are broken down into subscription types, including Basic, Standard, and Premium plans. This bar chart highlights customer preferences, revealing that the Basic subscription type has the highest number of subscribers, while the Premium plan has the lowest. Understanding these trends is important for Netflix to tailor its offering and pricing strategies to better meet consumer needs.

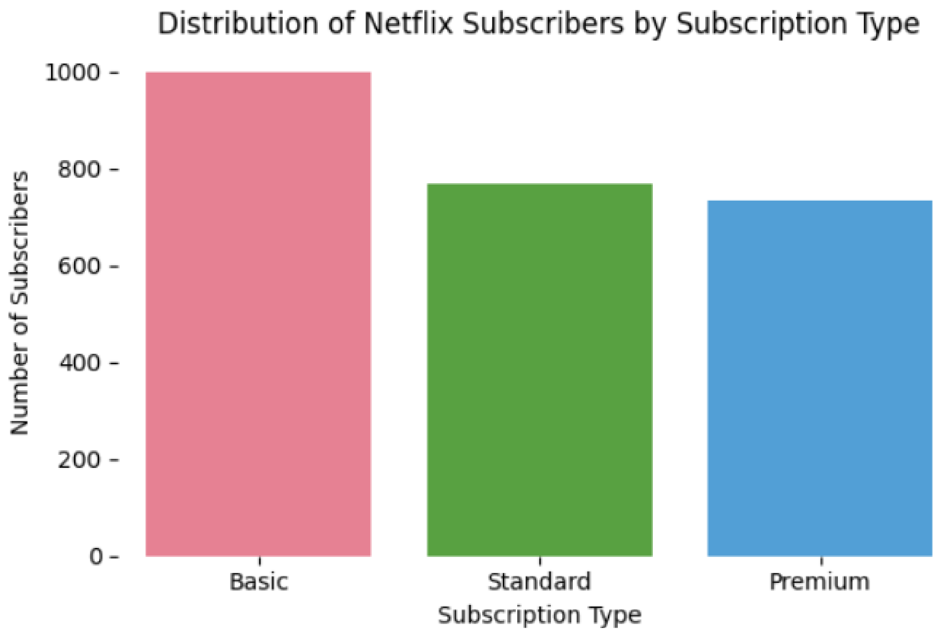


Figure 4.17: Distribution of Netflix Subscribers by Subscription Type

4. **Subscription Plans by Country: Figure 4.18** provides a summary bar chart showing the distribution of Netflix subscribers for each type of subscription in different countries. This chart highlights regional preferences for subscription policy and

provides a comparative analysis. Notably, Spain has the largest number of Premium subscribers, while the United States and Italy have the most Basic subscribers. Both the United Kingdom and Mexico exhibit significant numbers of Standard subscribers, more than other countries. Conversely, these two countries have the lowest number of Basic subscribers and no Premium subscribers at all. This information is important for understanding the regional subscription trend, which can inform target trade and regional strategies.

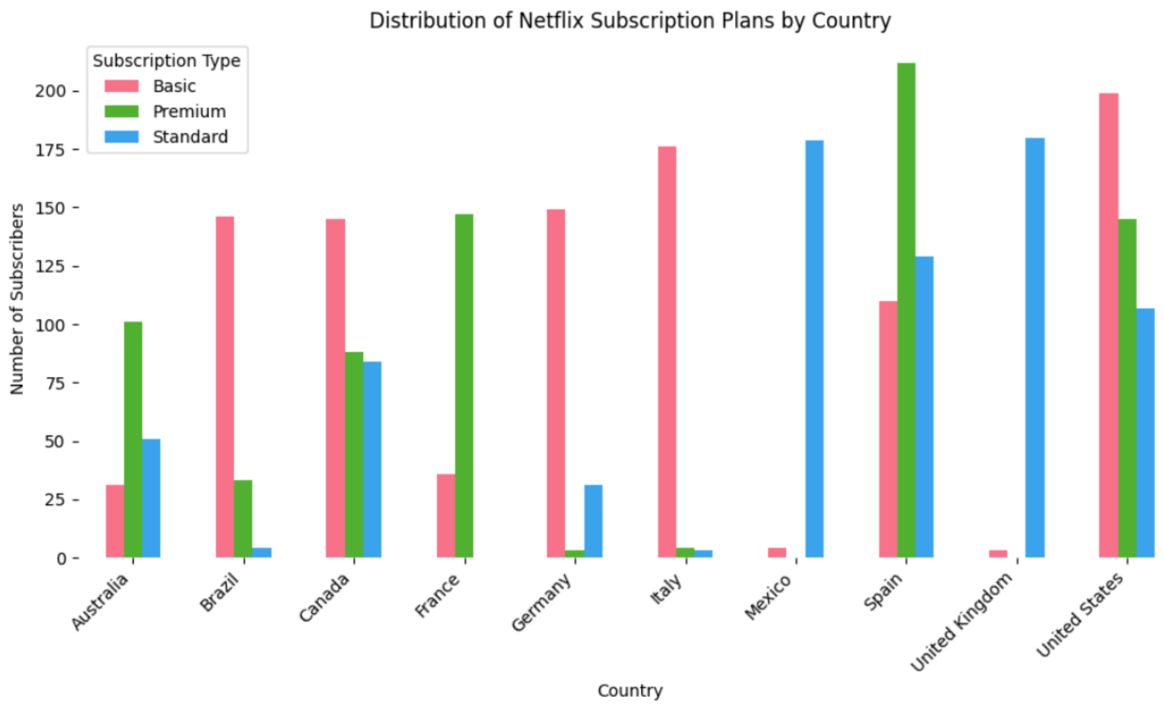


Figure 4.18: Distribution of Netflix Subscription Plans by Country

7. **Monthly Revenue Distribution Pie Chart:** The pie chart in **Figure 4.19** shows the proportion of total monthly revenue generated by each type of Netflix subscription. The chart shows that the Basic plan offers the highest revenue share at 39.9%, followed by the Standard plan at 30.6%, and the Premium plan at 29.5%. This chart provides a clear comparison of revenue contributions, focusing the dominant revenue reason across different subscription plans. This insight is valuable for revenue analysis and financial forecasting, helping to identify which plans are generating the most income and informs pricing and marketing strategies.

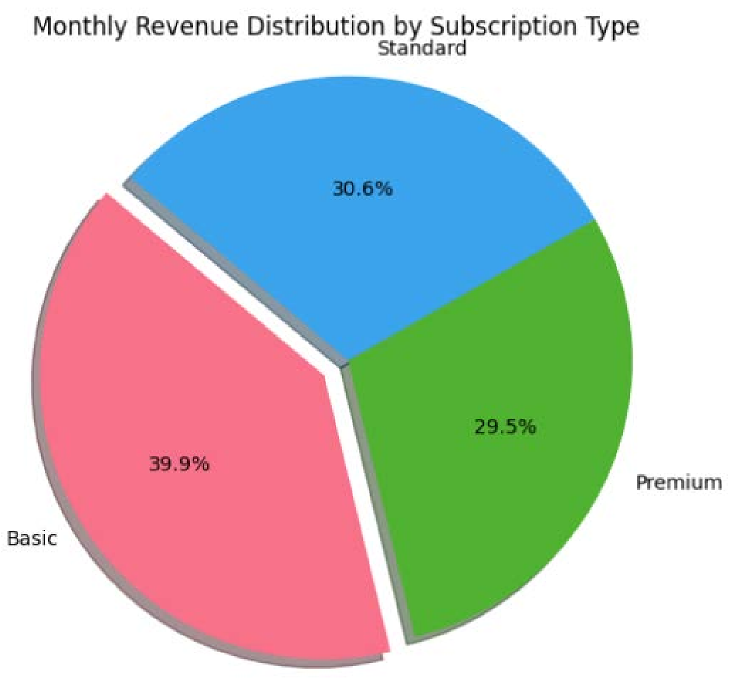


Figure 4.19: Monthly Revenue Distribution by Subscription Type (Pie Chart)

8. **Number of Users Joining Over Time:** The line plot in **Figure 4.20** shows the monthly rate of new subscribers to Netflix. By converting the 'Join Date' to a datetime format and aggregating the data on a monthly basis, this chart reveals trends and seasonal patterns in user acquisition. The plot shows a marked increase in enrollment starting after April 2022, and increasing after July 2022 . This analysis helps identify periods of greater growth and decline, and provides insights for strategic planning.

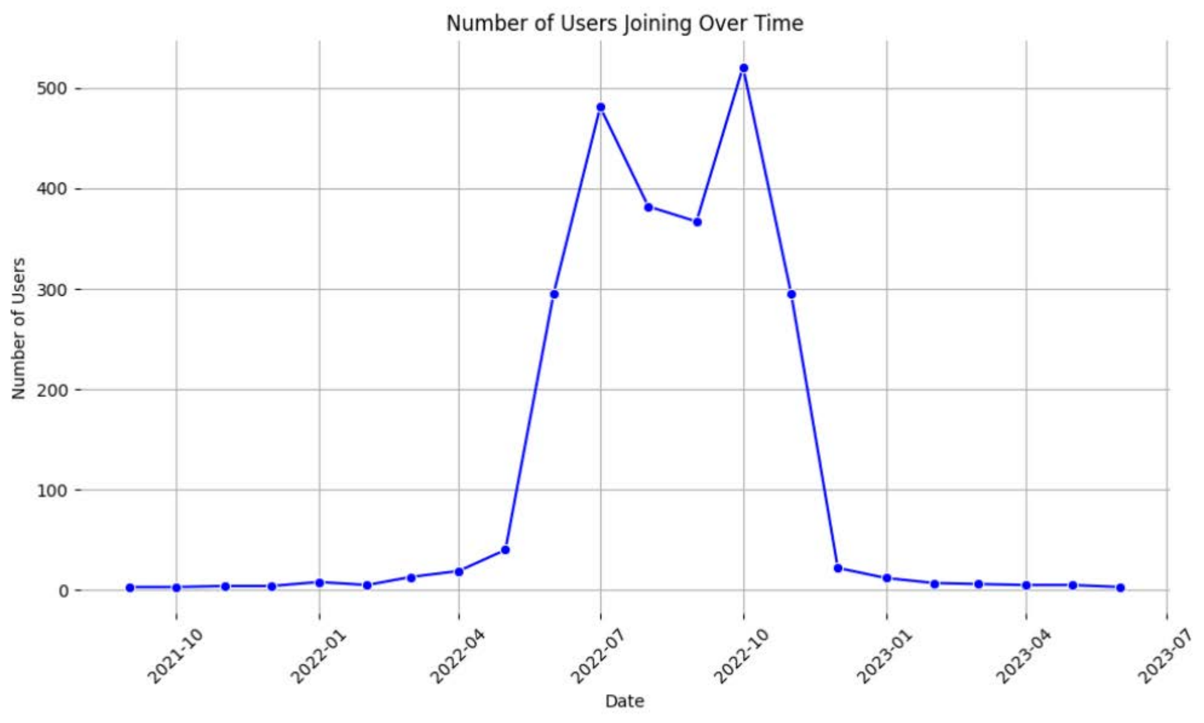


Figure 4.20: Number of Users Joining Over Time

### 4.3.3.3 Data Visualization in Power BI

In this section, we explore how to create and use interactive dashboards in Power BI to visualize Netflix stock price and Netflix subscriber growth rate data. Power BI played a key role in enhancing our understanding of stock price trends and subscriber growth patterns by transforming raw data into insightful and actionable visualizations. The section covers two main areas:

1. **Netflix Stock Price**: We integrated real-time stock price data in Power BI using the "**Get Data** > **Web**" feature, to access the Yahoo Finance web data link. This integration ensured that our dashboard reflected current market trends. We created interactive charts, with line charts to track historical price movements, bar charts for comparison, and other graphs to provide a more comprehensive view of Netflix stock price movements.
2. **Netflix Subscriber Growth Rate**: In addition to stock price data, we also visualized Netflix subscriber growth rate. The dashboard included charts showing trends, comparisons across geographies, and the number of items affecting subscribers. This allowed us to provide a comprehensive analysis of subscriber growth and provide valuable insights into the drivers of this change.

Together, these visualizations in Power BI provide a better view of Netflix's financial and subscriber metrics, allowing users to explore and analyze the data interactively.

**Netflix Stock Prices:**

**Figure 4.21** shows a Power BI dashboard with the Netflix Stock Historical Data View, highlighting charts and graphs designed to provide detailed analysis of stock price data. Each chart affecting the dashboard is described in more detail below, providing insight into how these visualizations contribute to a clearer understanding of historical stock price trends.

1. **Yearly Closing Price Trend:** The line chart is a graphical representation of how Netflix's closing prices are trending over the years, showing long-term price trends and seasonal trends. Initially, the stock price was relatively low but showed a gradual increase, indicative of Netflix's growth and market expansion. A sharp decline is seen around 2021, likely due to market fluctuations or external factors. Following this, the price started to rise again after 2022, indicating a recovery and renewed investor confidence.

2. **Stock Trading Volume by Year:** The column chart shows Netflix's trading volumes on an annual basis, highlighting trading activity patterns over time. Trading volume increased primarily between 2010 and 2013, indicating increased investor interest

and activity during this period. This increase may be related to significant developments in Netflix's business model or market presence during those years.

3. **Annual Open, High, and Low Prices:** shows the yearly Netflix stock opening, high, and low prices. Initially low, prices gradually increased, reflecting the growth of the company. In 2021, the stock rose to its highest value, followed by a sharp decline around 2022. This diagram is very important for understanding price volatility and overall dynamics, and highlights key moments of growth and decline.

4. **Daily Stock Price Movement with Moving Averages:** A candlestick chart overlaid with 20-day and 30-day moving averages provides a clear view of Netflix's daily stock price fluctuations. Continuous averages help smooth short-term changes, making it easier to spot trends and potential changes. This chart is valuable for analyzing daily price patterns in addition to longer-term trends, providing insight into potential support and resistance levels.

5. **Annual Closing Price Sum:** The ribbon chart is Netflix's cumulative closing prices on an annualized basis, and highlights the stock's overall performance trends over the years. The cumulative stock price reflects substantial ups and downs, especially substantial increases after 2015, reflecting Netflix's expanding market share and business growth.

6. **Monthly Closing Prices and Trading Volume:** The line and clustered column chart compares monthly closing prices to trading volume. This graph provides a clear picture of how stock prices and trading volume change each month, and provides insight into their relationship over time.

Additionally, the dashboard includes stock closing price as $633.94.

Through these visualizations, we gained a deeper understanding of Netflix stock data, identifying key trends, correlations, and anomalies. This exploratory analysis was helpful in informing our predictive modeling and increasing the accuracy of our forecasts.

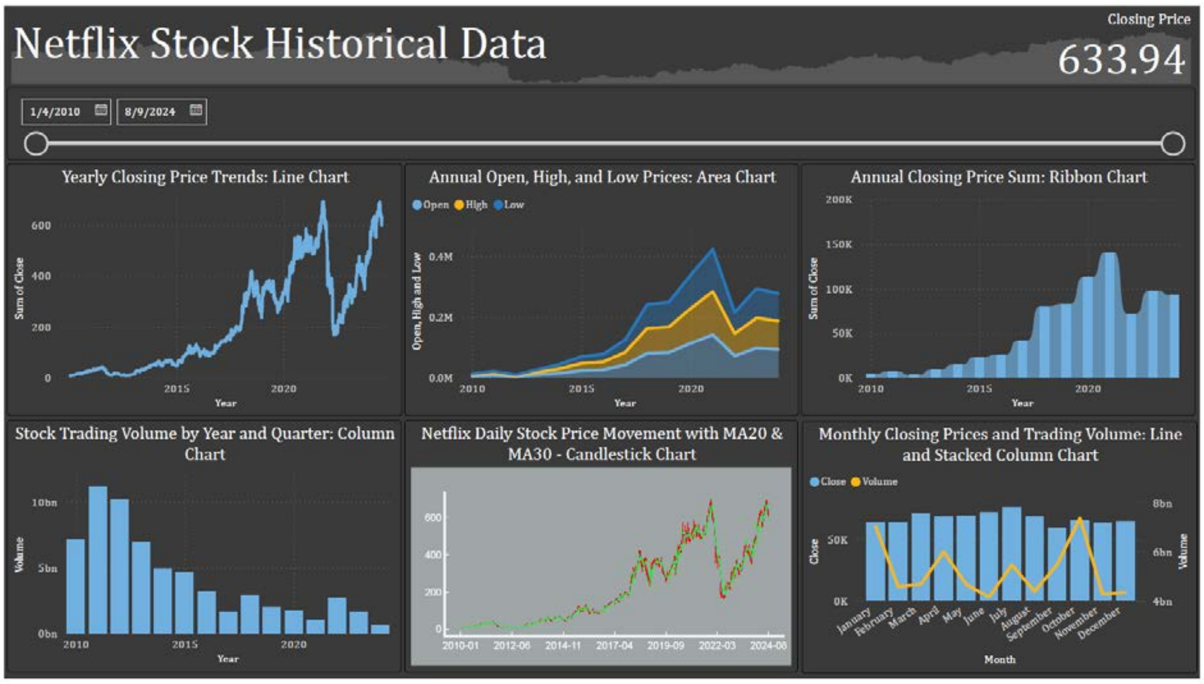


Figure 4.21: Netflix Stock Historical Data Power BI Dashboard

**Netflix Subscriber Growth Rate:**

In this section, we explore visualizations created in Power BI to analyze Netflix subscriber growth data. These interactive charts and graphs were developed to provide a comprehensive view of how subscribers are progressing, enabling detailed analytics and strategic insights. **Figure 4.22** shows the Netflix Subscriber Growth Rate Dashboard, along with graphs showing regional and temporal trends in subscriber numbers.

1. **Quarterly Analysis of Subscriber Count by Region:** The clustered column chart provides a quarterly survey of subscriptions by region. The data reveals that Spain has the highest number of subscribers in both Q1 and Q3, while the United States shows strong orders this quarter, leading the way in Q2. This type of visualization is essential to understand region based performance data on a quarterly basis.

2. **Subscriber Growth by Month:** The line chart shows monthly subscriber growth, highlighting June as the peak month. This visualization is invaluable for key moments of increasing clients, which can inform marketing strategies and product distribution.

3. **Subscriber Growth by Country:** The map shows growth rates by country, showing the United States and Italy as the most expanded, while Australia and Germany exhibit the lowest. These maps help identify regions that differ in development dynamics, which is important for targeted expansion strategies.

4. **Monthly Revenue by Country:** The funnel chart highlights monthly revenue by country, with the United States and Spain leading the way in revenue contribution. This type of visualization is important for understanding the economic impact of different regions and prioritizing high-income markets.

5. **Regional Monthly Subscriber Trends and Growth Analysis:** The line and stacked column chart analyzes regional monthly subscriber trends and growth rates, showing the highest growth rates in the United States and Italy. This integrated visualization provides a comprehensive view of subscriber trends and progress, assisting in both regional analysis and strategic planning.

6. **Subscription Type by Country:** The pie chart details the distribution of subscription types, and reveals that Basic subscriptions are 39.96% of the total, Standard subscriptions are 30.72%, and Premium subscriptions are 29.32%. This division analysis helps to understand customer preferences in different regions, and guides supply chain decisions.

The dashboard also shows key values such as:

- **Subscriber Growth Rate:** 111.89K
- **Monthly Subscriber Count:** 33,179

These Power BI visualizations provide detailed and interactive analysis of Netflix subscriber growth data, providing critical insights for data-driven decision making.

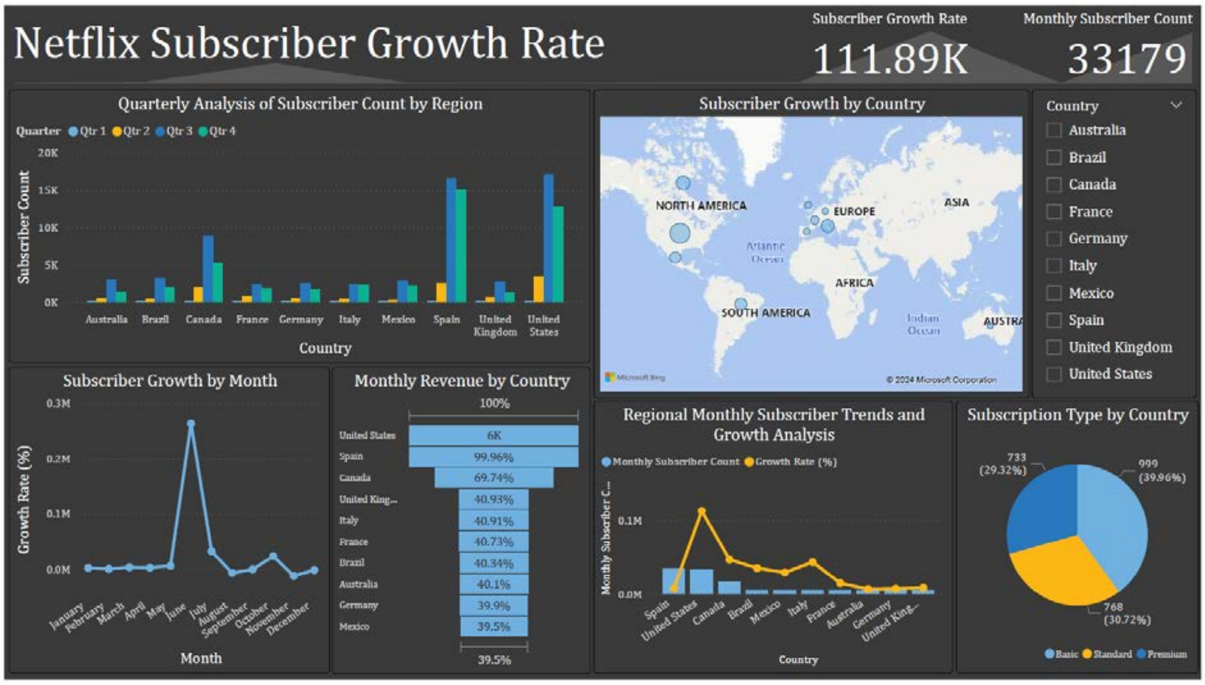

Netflix Subscriber Growth Rate
Subscriber Growth Rate
111.89K
Monthly Subscriber Count
33179
Quarterly Analysis of Subscriber Count by Region
Quarter ● Qtr 1 ● Qtr 2 ● Qtr 3 ● Qtr 4
Subscriber Count
Country
Subscriber Growth by Country
Country
Australia
Brazil
Canada
France
Germany
Italy
Mexico
Spain
United Kingdom
United States
Subscriber Growth by Month
Growth Rate (%)
Month
Monthly Revenue by Country
Regional Monthly Subscriber Trends and Growth Analysis
● Monthly Subscriber Count ● Growth Rate (%)
Subscription Type by Country
733 (29.32%)
999 (39.96%)
768 (30.72%)
● Basic ● Standard ● Premium

Figure 4.22: Netflix Subscriber Growth Rate Power BI Dashboard

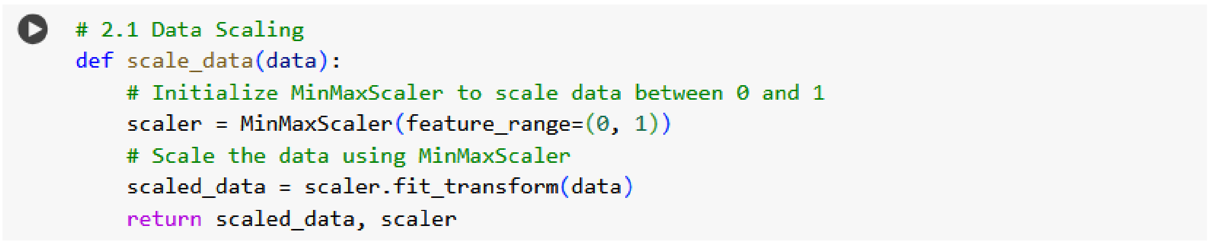
## 4.3.4 Data Preparation

The Data Preparation phase is important to ensure that the data is clean, accurate, and ready for modeling. This section describes methods for preparing both Netflix stock price data and subscriber growth data for predictive analysis. Adequate data preparation is an important factor in the accuracy and reliability of the model.

### 4.3.4.1 Netflix Stock Prices

#### 1. Data Scaling & Normalization

In order to prepare Netflix stock price data for modeling, the first step is to use MinMaxScaler to standardize the data. This option ensures that all feature values are normalized to the range between 0 and 1, which is important for models such as LSTM and CNN that are sensitive to scale of input features. The MinMaxScaler was applied to the Close value of the stock data. The function **scale_data** handles this scaling. **Figure 4.23** shows a code snippet used for data scaling and normalization.

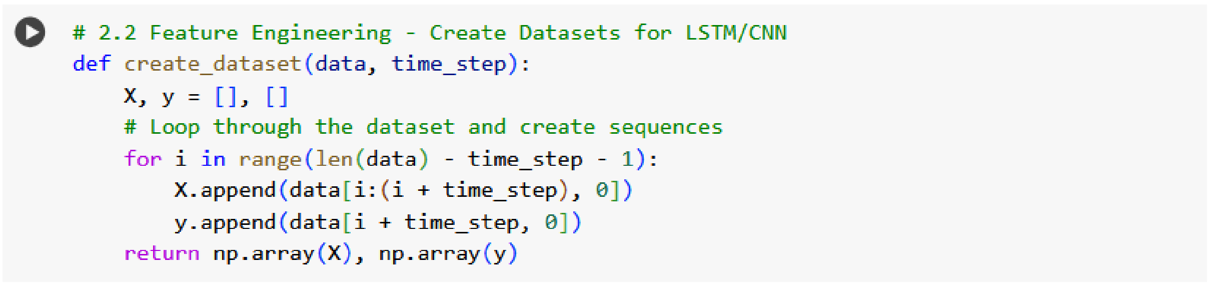


```python
# 2.1 Data Scaling
def scale_data(data):
    # Initialize MinMaxScaler to scale data between 0 and 1
    scaler = MinMaxScaler(feature_range=(0, 1))
    # Scale the data using MinMaxScaler
    scaled_data = scaler.fit_transform(data)
    return scaled_data, scaler
```

Figure 4.23: Code for Data Scaling and Normalization

#### 2. Feature Engineering

In time series forecasting, feature engineering involves transforming raw data into sequences that the model can learn from. The **create_dataset** function performs this task by generating sequences of past closing prices and target values. Each sequence of **time_step** historical data point serves as an input to the model, with the value being the target. **Figure 4.24** provides the code snippet for feature engineering performed for stock prices.



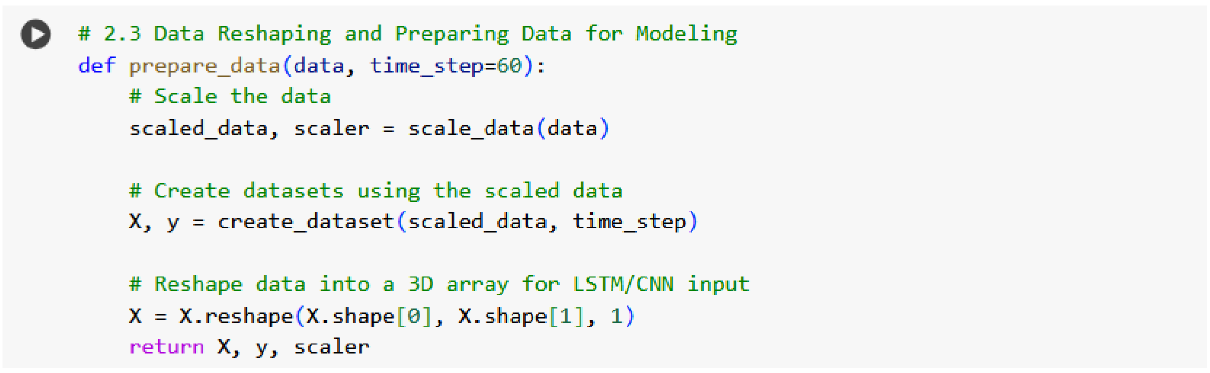
```python
# 2.2 Feature Engineering - Create Datasets for LSTM/CNN
def create_dataset(data, time_step):
    X, y = [], []
    # Loop through the dataset and create sequences
    for i in range(len(data) - time_step - 1):
        X.append(data[i:(i + time_step), 0])
        y.append(data[i + time_step, 0])
    return np.array(X), np.array(y)
```

Figure 4.24: Code for Creating Datasets for LSTM/CNN

#### 3. Data Reshaping

Once the sequences are generated, the data must be reshaped to the input requirements of the LSTM and CNN models. Specifically, the input data must be in 3D array format as:

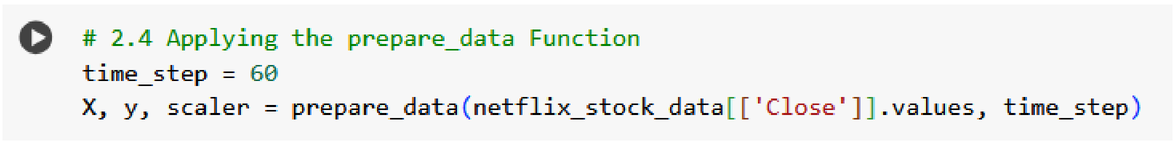
[shapes, time steps, objects]. This reshaping is handled by the **prepare_data** function, which scales the data, creates the necessary sequences, and reshapes the input as required. The reshaping ensures that the input dimensions meet the model's expectations. This reshaping step is shown in **Figure 4.25**.



```
# 2.3 Data Reshaping and Preparing Data for Modeling
def prepare_data(data, time_step=60):
    # Scale the data
    scaled_data, scaler = scale_data(data)

    # Create datasets using the scaled data
    X, y = create_dataset(scaled_data, time_step)

    # Reshape data into a 3D array for LSTM/CNN input
    X = X.reshape(X.shape[0], X.shape[1], 1)
    return X, y, scaler
```

Figure 4.25: Code snippet for Preparing Data for Modeling

**Figure 4.26** shows an example of the use of **prepare_data** function, where a 60-day time step is called to create a series from historical stock price data. The Figure shows how the function scales the data, creates sequences, and reshapes it into a format suitable for LSTM/CNN models.



```
# 2.4 Applying the prepare_data Function
time_step = 60
X, y, scaler = prepare_data(netflix_stock_data[['Close']].values, time_step)
```

Figure 4.26: Prepared Data for LSTM/CNN

This steps ensures that the stock price data is correctly scaled, sequenced, and reshaped, making it ready to be used in LSTM and CNN models to predict future stock prices.

#### 4.3.4.2 Subscriber Growth Data

The subscriber growth data went through more extensive cleaning and transformation process. The following steps were performed:

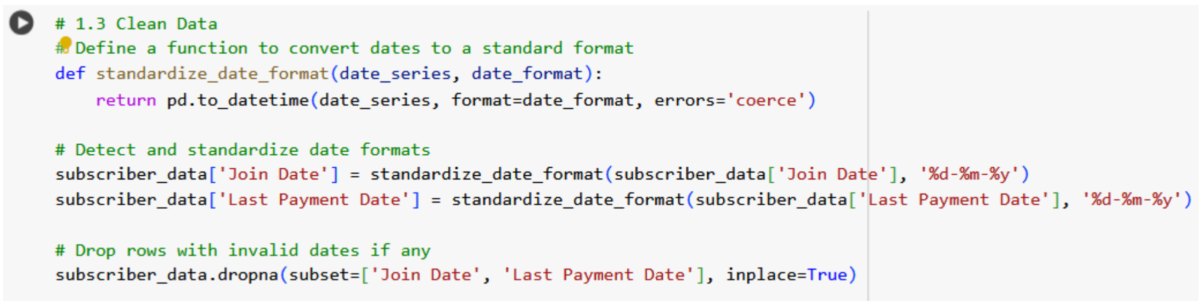
**1. Date Standardization**

Given that the date columns (Join Date and Last Payment Date) had inconsistent formats, a custom function was used to standardize these dates in a uniform format (%d-%m-%y). This function also handled errors by converting incorrect date information to NaT (Not a Time).

**2. Invalid Date Handling**

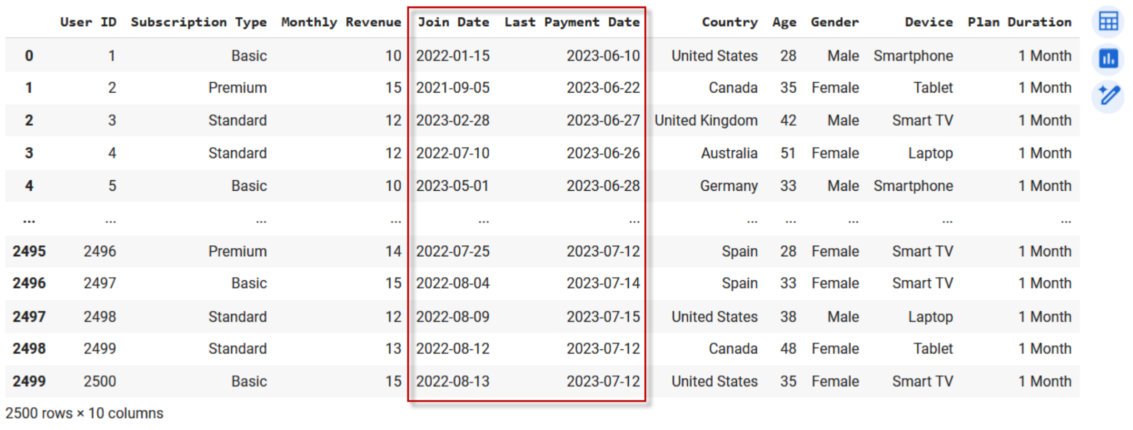
After standardizing the date formats, rows with invalid dates were dropped from the dataset to maintain data integrity.

See **Figure 4.27** for code snippets showing invalid date standardization and date handling changes and **Figure 4.28** for the resulting cleaned data.



```
# 1.3 Clean Data
# Define a function to convert dates to a standard format
def standardize_date_format(date_series, date_format):
    return pd.to_datetime(date_series, format=date_format, errors='coerce')

# Detect and standardize date formats
subscriber_data['Join Date'] = standardize_date_format(subscriber_data['Join Date'], '%d-%m-%y')
subscriber_data['Last Payment Date'] = standardize_date_format(subscriber_data['Last Payment Date'], '%d-%m-%y')

# Drop rows with invalid dates if any
subscriber_data.dropna(subset=['Join Date', 'Last Payment Date'], inplace=True)
```

Figure 4.27: Code for Date Standardization and Cleaning



|  | User ID | Subscription Type | Monthly Revenue | Join Date | Last Payment Date | Country | Age | Gender | Device | Plan Duration |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | Basic | 10 | 2022-01-15 | 2023-06-10 | United States | 28 | Male | Smartphone | 1 Month |
| 1 | 2 | Premium | 15 | 2021-09-05 | 2023-06-22 | Canada | 35 | Female | Tablet | 1 Month |
| 2 | 3 | Standard | 12 | 2023-02-28 | 2023-06-27 | United Kingdom | 42 | Male | Smart TV | 1 Month |
| 3 | 4 | Standard | 12 | 2022-07-10 | 2023-06-26 | Australia | 51 | Female | Laptop | 1 Month |
| 4 | 5 | Basic | 10 | 2023-05-01 | 2023-06-28 | Germany | 33 | Male | Smartphone | 1 Month |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 2495 | 2496 | Premium | 14 | 2022-07-25 | 2023-07-12 | Spain | 28 | Female | Smart TV | 1 Month |
| 2496 | 2497 | Basic | 15 | 2022-08-04 | 2023-07-14 | Spain | 33 | Female | Smart TV | 1 Month |
| 2497 | 2498 | Standard | 12 | 2022-08-09 | 2023-07-15 | United States | 38 | Male | Laptop | 1 Month |
| 2498 | 2499 | Standard | 13 | 2022-08-12 | 2023-07-12 | Canada | 48 | Female | Tablet | 1 Month |
| 2499 | 2500 | Basic | 15 | 2022-08-13 | 2023-07-12 | United States | 35 | Female | Smart TV | 1 Month |

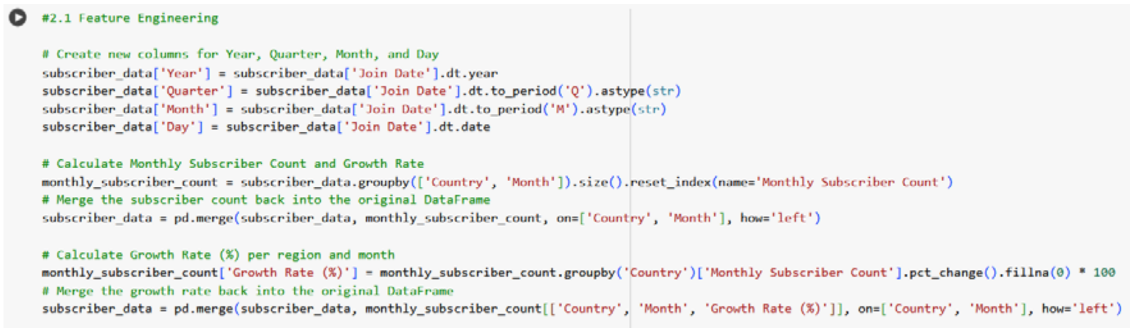
2500 rows × 10 columns

Figure 4.28: Cleaned Subscriber Data After Date Standardization

## 3. Feature Engineering

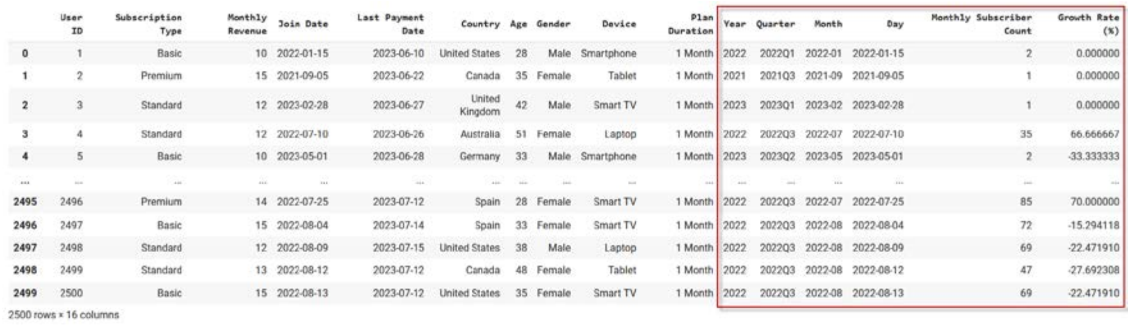
Additional insights were developed to improve the dataset:

- **Year, Quarter, Month, and Day:** These features were derived from the Join Date to allow for granular temporal analysis.
- **Monthly Subscriber Count and Growth Rate (%):** The dataset is expanded with columns monthly subscriber counts and growth rate. This involved grouping the data by country and month, calculating the number of subscribers each month, and determining the month-over-month growth rate. These features are important for understanding trends and patterns in subscriber growth. **Figure 4.29** and **Figure 4.30** show the code snippets and outputs for feature engineering steps performed for subscriber growth data.



```
#2.1 Feature Engineering

# Create new columns for Year, Quarter, Month, and Day
subscriber_data['Year'] = subscriber_data['Join Date'].dt.year
subscriber_data['Quarter'] = subscriber_data['Join Date'].dt.to_period('Q').astype(str)
subscriber_data['Month'] = subscriber_data['Join Date'].dt.to_period('M').astype(str)
subscriber_data['Day'] = subscriber_data['Join Date'].dt.date

# Calculate Monthly Subscriber Count and Growth Rate
monthly_subscriber_count = subscriber_data.groupby(['Country', 'Month']).size().reset_index(name='Monthly Subscriber Count')
# Merge the subscriber count back into the original DataFrame
subscriber_data = pd.merge(subscriber_data, monthly_subscriber_count, on=['Country', 'Month'], how='left')

# Calculate Growth Rate (%) per region and month
monthly_subscriber_count['Growth Rate (%)'] = monthly_subscriber_count.groupby('Country')['Monthly Subscriber Count'].pct_change().fillna(0) * 100
# Merge the growth rate back into the original DataFrame
subscriber_data = pd.merge(subscriber_data, monthly_subscriber_count[['Country', 'Month', 'Growth Rate (%)']], on=['Country', 'Month'], how='left')
```

Figure 4.29: Code for Feature Engineering in Subscriber Growth Data



| | User ID | Subscription Type | Monthly Revenue | Join Date | Last Payment Date | Country | Age | Gender | Device | Plan Duration | Year | Quarter | Month | Day | Monthly Subscriber Count | Growth Rate (%) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | Basic | 10 | 2022-01-15 | 2023-06-10 | United States | 28 | Male | Smartphone | 1 Month | 2022 | 2022Q1 | 2022-01 | 2022-01-15 | 2 | 0.000000 |
| 1 | 2 | Premium | 15 | 2021-09-05 | 2023-06-22 | Canada | 35 | Female | Tablet | 1 Month | 2021 | 2021Q3 | 2021-09 | 2021-09-05 | 1 | 0.000000 |
| 2 | 3 | Standard | 12 | 2023-02-28 | 2023-06-27 | United Kingdom | 42 | Male | Smart TV | 1 Month | 2023 | 2023Q1 | 2023-02 | 2023-02-28 | 1 | 0.000000 |
| 3 | 4 | Standard | 12 | 2022-07-10 | 2023-06-26 | Australia | 51 | Female | Laptop | 1 Month | 2022 | 2022Q3 | 2022-07 | 2022-07-10 | 35 | 66.666667 |
| 4 | 5 | Basic | 10 | 2023-05-01 | 2023-06-28 | Germany | 33 | Male | Smartphone | 1 Month | 2023 | 2023Q2 | 2023-05 | 2023-05-01 | 2 | -33.333333 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 2495 | 2496 | Premium | 14 | 2022-07-25 | 2023-07-12 | Spain | 28 | Female | Smart TV | 1 Month | 2022 | 2022Q3 | 2022-07 | 2022-07-25 | 85 | 70.000000 |
| 2496 | 2497 | Basic | 15 | 2022-08-04 | 2023-07-14 | Spain | 33 | Female | Smart TV | 1 Month | 2022 | 2022Q3 | 2022-08 | 2022-08-04 | 72 | -15.294118 |
| 2497 | 2498 | Standard | 12 | 2022-08-09 | 2023-07-15 | United States | 38 | Male | Laptop | 1 Month | 2022 | 2022Q3 | 2022-08 | 2022-08-09 | 69 | -22.471910 |
| 2498 | 2499 | Standard | 13 | 2022-08-12 | 2023-07-12 | Canada | 48 | Female | Tablet | 1 Month | 2022 | 2022Q3 | 2022-08 | 2022-08-12 | 47 | -27.692308 |
| 2499 | 2500 | Basic | 15 | 2022-08-13 | 2023-07-12 | United States | 35 | Female | Smart TV | 1 Month | 2022 | 2022Q3 | 2022-08 | 2022-08-13 | 69 | -22.471910 |

2500 rows × 16 columns

Figure 4.30: Output of Feature Engineering for Subscriber Growth Data

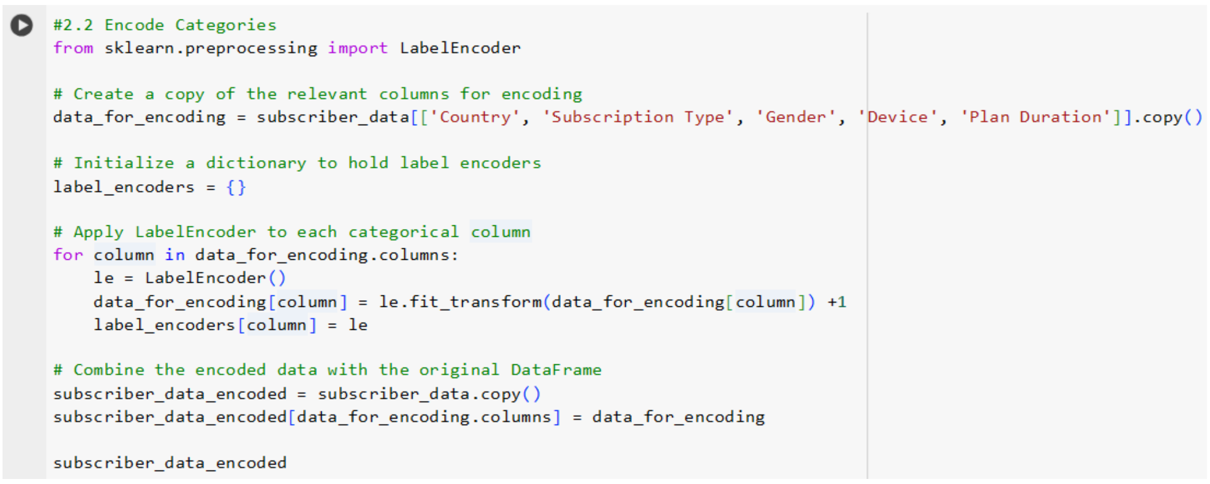
**4. Category Encoding**

Categorical columns such as Country, Subscription Type, Gender, Device, and Plan Duration were encoded using **LabelEncoder**. This process converts categorical values to numerical, making them suitable for machine learning models. **Table 4.3** shows these columns and their respective categories that were encoded, such as 'Country' converted into numeric labels representing each country, also Subscription Type, Gender, Device, and Plan Duration. This ensures that the machine learning algorithms can effectively understand and process these categorical inputs.

| DESCRIPTION | CATEGORIES |
|---|---|
| **COUNTRY** | Australia, Brazil, Canada, France, Germany, Italy, Mexico, Spain, United Kingdom, United States |
| **SUBSCRIPTION** | Basic, Premium, Standard |
| **GENDER** | Male, Female |
| **DEVICE** | Laptop, Smart Tv, Smart Phone, Tablet Plan |
| **DURATION** | 1 Month |

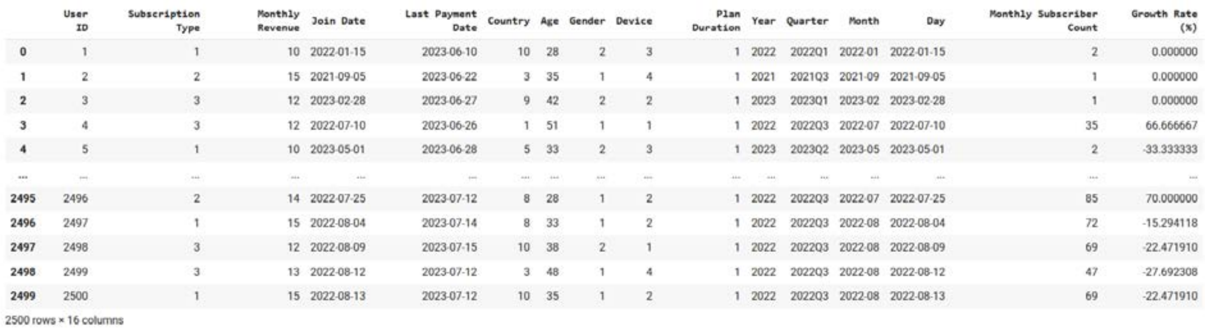
Table 4.3: Categories for Encoding

**Figure 4.31** shows the code snippet for encoding categories, while **Figure 4.32** displays the output after encoding process. This step was crucial for transforming categorical features into a format suitable for machine learning models.



```python
#2.2 Encode Categories
from sklearn.preprocessing import LabelEncoder

# Create a copy of the relevant columns for encoding
data_for_encoding = subscriber_data[['Country', 'Subscription Type', 'Gender', 'Device', 'Plan Duration']].copy()

# Initialize a dictionary to hold label encoders
label_encoders = {}

# Apply LabelEncoder to each categorical column
for column in data_for_encoding.columns:
    le = LabelEncoder()
    data_for_encoding[column] = le.fit_transform(data_for_encoding[column]) +1
    label_encoders[column] = le

# Combine the encoded data with the original DataFrame
subscriber_data_encoded = subscriber_data.copy()
subscriber_data_encoded[data_for_encoding.columns] = data_for_encoding

subscriber_data_encoded
```

Figure 4.31: Code for Encoding Categorical Variables



| | User ID | Subscription Type | Monthly Revenue | Join Date | Last Payment Date | Country | Age | Gender | Device | Plan Duration | Year | Quarter | Month | Day | Monthly Subscriber Count | Growth Rate (%) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 10 | 2022-01-15 | 2023-06-10 | 10 | 28 | 2 | 3 | 1 | 2022 | 2022Q1 | 2022-01 | 2022-01-15 | 2 | 0.000000 |
| 1 | 2 | 2 | 15 | 2021-09-05 | 2023-06-22 | 3 | 35 | 1 | 4 | 1 | 2021 | 2021Q3 | 2021-09 | 2021-09-05 | 1 | 0.000000 |
| 2 | 3 | 3 | 12 | 2023-02-28 | 2023-06-27 | 9 | 42 | 2 | 2 | 1 | 2023 | 2023Q1 | 2023-02 | 2023-02-28 | 1 | 0.000000 |
| 3 | 4 | 3 | 12 | 2022-07-10 | 2023-06-26 | 1 | 51 | 1 | 1 | 1 | 2022 | 2022Q3 | 2022-07 | 2022-07-10 | 35 | 66.666667 |
| 4 | 5 | 1 | 10 | 2023-05-01 | 2023-06-28 | 5 | 33 | 2 | 3 | 1 | 2023 | 2023Q2 | 2023-05 | 2023-05-01 | 2 | -33.333333 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 2495 | 2496 | 2 | 14 | 2022-07-25 | 2023-07-12 | 8 | 28 | 1 | 2 | 1 | 2022 | 2022Q3 | 2022-07 | 2022-07-25 | 85 | 70.000000 |
| 2496 | 2497 | 1 | 15 | 2022-08-04 | 2023-07-14 | 8 | 33 | 1 | 2 | 1 | 2022 | 2022Q3 | 2022-08 | 2022-08-04 | 72 | -15.294118 |
| 2497 | 2498 | 3 | 12 | 2022-08-09 | 2023-07-15 | 10 | 38 | 2 | 1 | 1 | 2022 | 2022Q3 | 2022-08 | 2022-08-09 | 69 | -22.471910 |
| 2498 | 2499 | 3 | 13 | 2022-08-12 | 2023-07-12 | 3 | 48 | 1 | 4 | 1 | 2022 | 2022Q3 | 2022-08 | 2022-08-12 | 47 | -27.692308 |
| 2499 | 2500 | 1 | 15 | 2022-08-13 | 2023-07-12 | 10 | 35 | 1 | 2 | 1 | 2022 | 2022Q3 | 2022-08 | 2022-08-13 | 69 | -22.471910 |

2500 rows × 16 columns

Figure 4.32: Output of Encoded Subscriber Growth Data

The transformations and feature engineering ensured that the subscriber growth data was well-prepared, with features relevant for modeling and analysis process.

## 5. Feature Selection

For the subscriber growth data, feature selection involves below steps:

- **Correlation Matrix:** To identify the most suitable features for predicting subscriber growth, a correlation matrix was calculated. This matrix shows the relationship between features and the target variable. Features with higher correlation values with the target variable are generally considered more appropriate.
- **Feature Selection:** Based on the correlation matrix, the features that are significantly correlated with the target variable (Growth Rate) were selected. This step ensures that the model focuses on the most relevant variables, potentially improving its application and interpretation.
- **Target Variable:** Growth Rate (%) was the selected target variable, which refers to the percentage change in the number of subscribers from one month to the next.

**Figure 4.33** shows the correlation matrix for the subscriber growth data. The matrix highlights the strength and relationships between features and the target variable, which helps to select factors with the greatest impact.

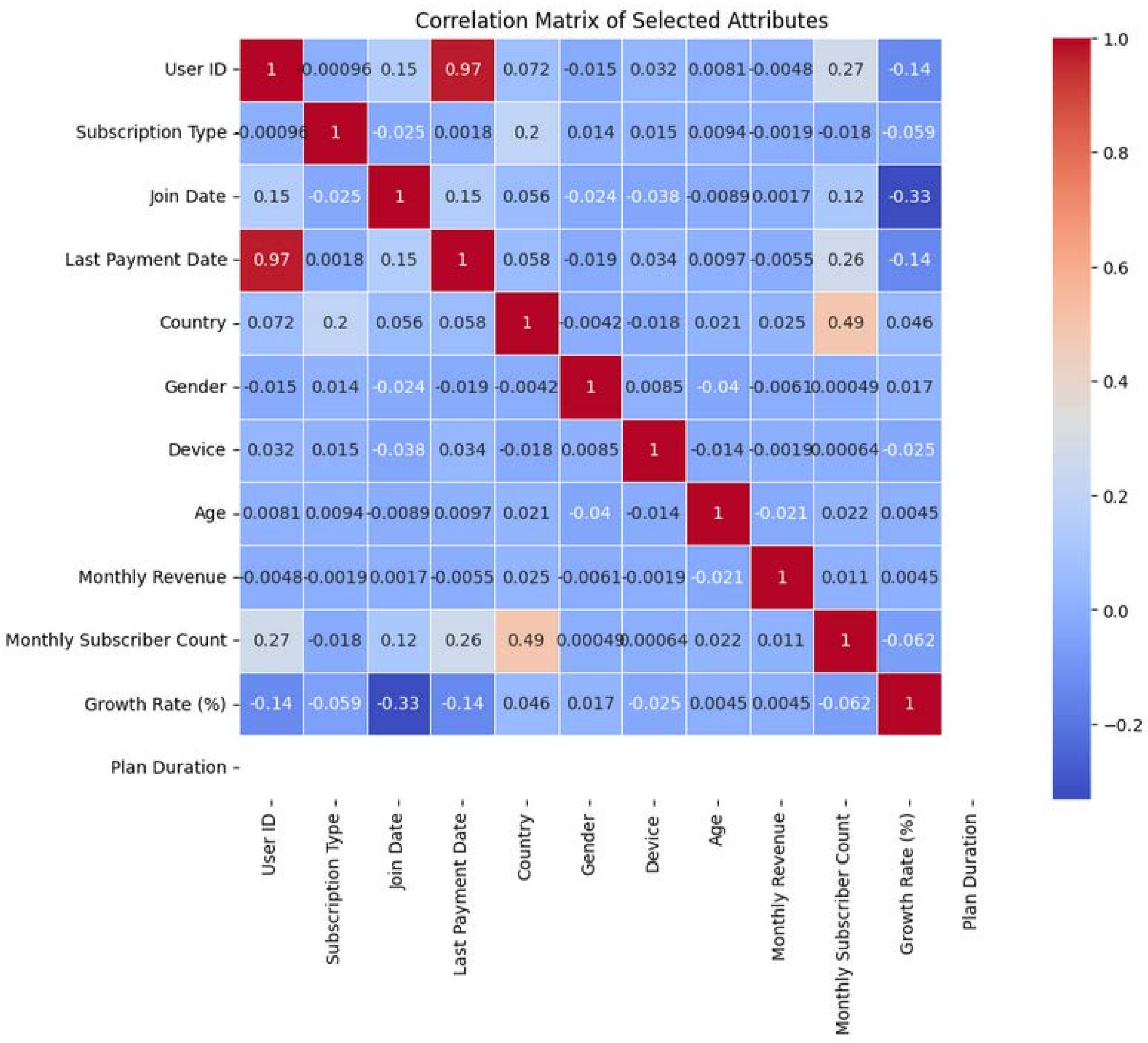


Figure 4.33: Correlation Matrix

The following features were selected from the given correlation matrix because of their relevance to the target:

- **Country** (Correlation: 0.045854)
- **Gender** (Correlation: 0.017306)
- **Age** (Correlation: 0.004507)
- **Monthly Revenue** (Correlation: 0.004462)
- **Device** (Correlation: -0.024586)
- **Subscription Type** (Correlation: -0.058909)
- **Monthly Subscriber Count** (Correlation: -0.061929)

These factors, although showing relatively low correlations, were included because they may contribute to model performance when combined with other variables or may capture patterns that might not be evident from individual correlations.

However, the following features were not selected due to their higher negative correlations with the target variable, which suggests that they might not provide valuable information for predicting Growth Rate (%). Including these features could potentially introduce noise into the model, thereby reducing its accuracy and interpretability.

- **User ID** (Correlation: -0.136552)
- **Last Payment Date** (Correlation: -0.140974)
- **Join Date** (Correlation: -0.331325)
- **Plan Duration**: This item was not selected because it contains only one category, '1 Month'. Due to the lack of variables, it does not provide meaningful information to determine the Growth rate (%) and has no relationship with the target variable. Also, it can introduce unnecessary noise into the image.

**Table 4.4** lists the selected features and target variables along with their specifications.

| FEATURES | DESCRIPTION |
|---|---|
| **COUNTRY** | Region or country of the subscribers |
| **SUBSCRIPTION TYPE** | Type of subscription (e.g., basic, premium) |
| **MONTHLY REVENUE** | Revenue generated per month from subscriptions |
| **AGE** | Age of the subscriber |
| **GENDER** | Gender of the subscriber |
| **DEVICE** | Device used to access the service |
| **MONTHLY SUBSCRIBER COUNT** | Count of subscribers each month |
| **TARGET VARIABLE** | Growth Rate - Percentage change in subscribers |

Table 4.4: Selected Features & Target Variable for Subscriber Growth Data

**Figure 4.34** shows a code snippet for feature selection based on correlation analysis. This strategy involves selecting the factors that have the greatest impact on the target variable, ensuring that the model is trained on the most relevant information.

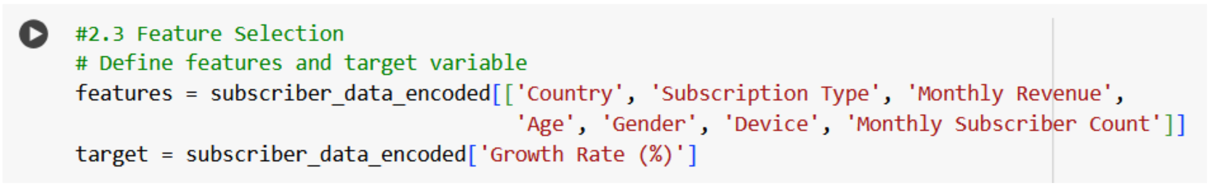
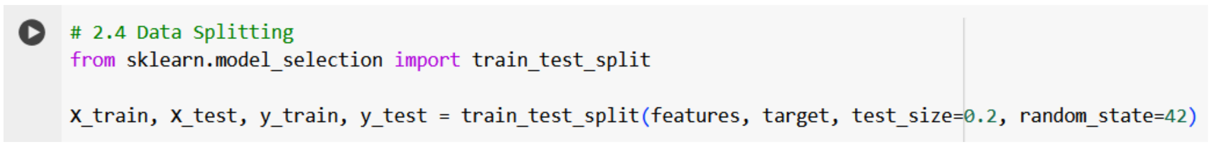

```
#2.3 Feature Selection
# Define features and target variable
features = subscriber_data_encoded[['Country', 'Subscription Type', 'Monthly Revenue',
                                    'Age', 'Gender', 'Device', 'Monthly Subscriber Count']]
target = subscriber_data_encoded['Growth Rate (%)']
```

Figure 4.34: Feature Selection Code Snippet

## 6. Data Splitting

Data splitting involves partitioning the dataset into subsets of training and testing to see how the model performs on unseen data. **Figure 4.35** shows a code snippet for data partitioning and shows how the dataset is divided into training and testing subsets.



```
# 2.4 Data Splitting
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.2, random_state=42)
```

Figure 4.35: Code for Data Splitting

The dataset is divided as follows:

- **Training Set (80%):** Used for training the machine learning model.
- **Testing Set (20%):** Used for evaluating the model's performance.

The data preparation step carefully prepared both Netflix stock price and subscriber growth datasets, ensuring they were optimized for accurate modeling and analysis. This thorough preparation minimized the potential for error and greatly enhanced the performance of the models in subsequent analyses.

In conclusion, Chapter 4 detailed the important steps involved in preparing Netflix stock price and subscriber growth datasets for modeling. The step ensured that the data was formatted properly for use in machine learning models. Through careful selection of appropriate features and appropriate methodologies, the foundation was laid for an accurate and efficient forecasting model. In the upcoming Chapter 5, we will build and train various models, including LSTM, CNN, and Random Forest, to predict Netflix stock price and subscriber growth.

Chapter 5

# Implementation & Evaluation

In the previous chapter, we prepared the data for analysis and modeling by cleaning, scaling, and organizing both the Netflix stock price and subscriber growth datasets. Through feature engineering and extensive selection, we identified the most appropriate variables to predict stock prices and subscriber growth rates, setting a solid foundation for the modeling tasks described in this chapter.

This chapter focuses on the use of predictive models and training on Netflix stock price and subscriber growth rates. Includes development of Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN) models for stock price prediction, and Random Forest Regressor modeling and training to predict subscriber growth. Detailed code snippets are provided, along with descriptions of the modeling choices and configurations used.

For Netflix stock price forecasting, we used two deep learning models: LSTM and CNN. The LSTM model was chosen because of its ability to capture the sequence of dependencies in time series data, which is crucial for accurate stock price forecasting. Its structure of several LSTM layers followed by Dense layers was designed to improve the prediction performance of the model. The model is trained using the Mean Squared Error (MSE) loss function and the Adam optimizer, which is well suited to the use of complex stock price data.

In addition to LSTM, a CNN model was developed to evaluate the ability to detect trends and patterns in time series data. The CNN algorithm includes convolutional layers for feature extraction, pooling layers for down sampling, and Dense layers for final prediction. These models were trained with the Adam optimizer and the MSE loss function, similar to LSTM, and verified for methodological consistency.

For the Netflix subscriber growth forecast, the Random Forest Regressor was used due to its robustness and efficiency in large-scale datasets with a wide range of features. The Random Forest algorithm builds multiple decision trees during training and combines their predictions for accuracy and robustness. This chapter describes in detail the training of the Random Forest model, including its performance evaluation on the test dataset.

The main goal of this chapter is to bridge the gap between data preparation and actionable insights by implementing and training predictive models for both stock prices and subscriber growth. It describes the use of LSTM and CNN models for stock forecasting and the Random Forest Regressor for subscriber growth forecasting, and allows for detailed evaluation and optimization in the following chapters.

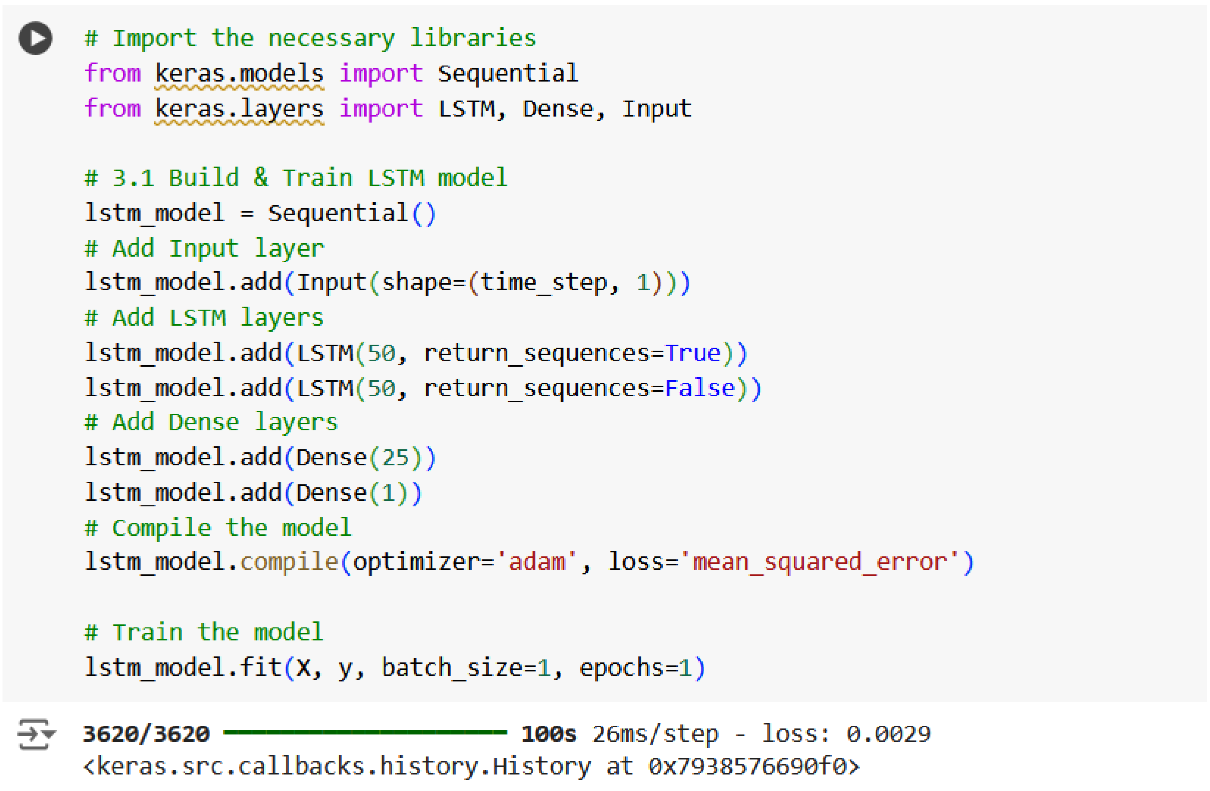
## 5.1 Model Building & Training

This section focuses on the development and refinement of prediction models developed for our objectives. This phase involves selecting appropriate algorithms, setting model parameters, and training the models on historical data to ensure accurate capture of underlying patterns and trends. By iteratively testing and optimizing these models, we aim to achieve robust performance that generalizes well to new, unseen data. This section describes in detail the steps taken to build, train, and test the selected models, and sets the stage for their use in prediction and analysis.

### 5.1.1 Long Short-Term Memory (LSTM)

The Long Short-Term Memory (LSTM) model is well-suited for time series forecasting tasks, such as stock price forecasting, due to its ability to capture long-term dependency in sequential data. The LSTM model was created using the Sequential API from Keras.

**Figure 5.1** shows the code snippet for building and training the LSTM model.



```
# Import the necessary libraries
from keras.models import Sequential
from keras.layers import LSTM, Dense, Input

# 3.1 Build & Train LSTM model
lstm_model = Sequential()
# Add Input layer
lstm_model.add(Input(shape=(time_step, 1)))
# Add LSTM layers
lstm_model.add(LSTM(50, return_sequences=True))
lstm_model.add(LSTM(50, return_sequences=False))
# Add Dense layers
lstm_model.add(Dense(25))
lstm_model.add(Dense(1))
# Compile the model
lstm_model.compile(optimizer='adam', loss='mean_squared_error')

# Train the model
lstm_model.fit(X, y, batch_size=1, epochs=1)
```

```
3620/3620 ━━━━━━━━━━━━━━━━━━━━ 100s 26ms/step - loss: 0.0029
<keras.src.callbacks.history.History at 0x7938576690f0>
```

Figure 5.1: LSTM Model Building & Training

### 5.1.1.1 Model Architecture

- **Input Layer:** The model starts with an input layer defined in **Input(shape = (time_step, 1))**. Here, time_step represents the number of previous days' values against the next day's value. This option ensures that the model considers a fixed sequence length for each forecast, and captures the temporal dependence in the data.
- **LSTM Layers:** Two LSTM layers are used in the model. Each layer consists of 50 units. **return_sequences=True** parameter is set for the first LSTM layer to ensure that the entire sequence is passed to the next layer, which is necessary for capturing all relevant temporal information at the input data. The second LSTM layer is irreversible, which is appropriate for the transition from sequence acquisition to a short position before the final complex sequence.
- **Dense Layers:** In addition to the LSTM layers, two Dense layers are added. The last dense layer produces a single value, which represents the stock price forecast for the next day. Dense layers are used to study nonlinear combinations of features extracted by the LSTM layers, terminating in the final prediction.
- **Compilation and Training:** The number of classes in training is adjusted with the Adam optimizer, and the Mean Squared Error (MSE) loss function, which is commonly used in regression tasks to minimize the average squared difference between predicted and actual values.

### 5.1.1.2 Training Parameters

- **Batch Size:** The model is trained with a batch size of 1, which means that the model updates its weight after every single training sample. While this may make the training process more computationally intensive, it allows for more granular adjustments to the model parameters and can be useful for time series data where each sample is critical. Although different batch sizes can be tested for efficiency, in this case batch size 1 was chosen to simplify the process and to facilitate accurate replication after each data point.
- **Epochs:** The model is initially trained for epoch 1. An epoch specifies one complete process through the entire training dataset. Training repeatedly typically improves the performance of the model, but the initial training run once is a preliminary test to ensure that the model is working correctly and check the initial loss of value. While different numbers of periods can be tried to obtain optimal results, starting from a single period helps to quickly examine the basic functionality of the model and make necessary adjustments.

#### 5.1.1.3 Output of Modeling & Training

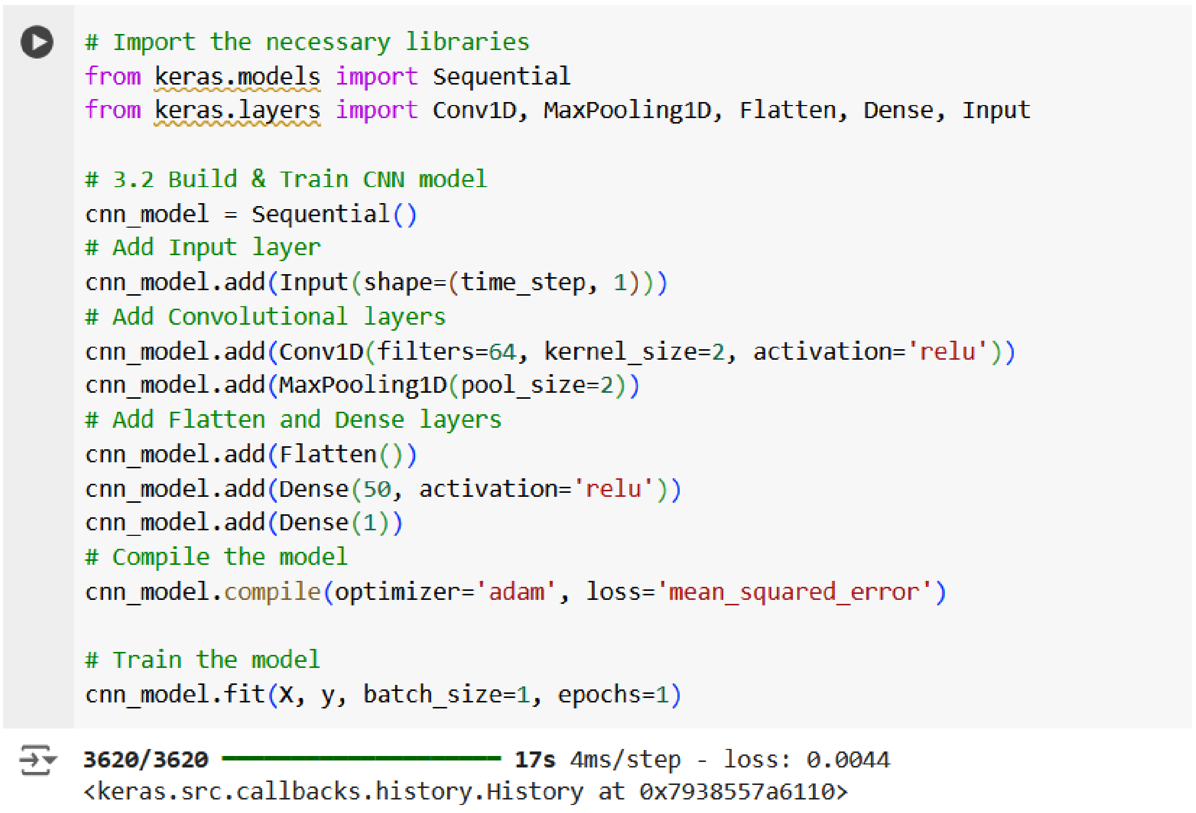
The training process of the LSTM model is examined by tracking the loss value, which measures the accuracy of the model predictions with the actual stock price. **Figure 5.2** shows the results of the training process, where a summary of the training is expressed as:

3620/3620 ━━━━━━━━━━━━━━━━━━━━ 100s 26ms/step - loss: 0.0029

These results indicate that the model has handled all 3,620 data points well, and the final loss value is 0.0029. The loss value is important for understanding the performance of the model, as it represents the average squared difference between the predicted and actual stock prices. A low loss value indicates that the model is well suited to the training data, indicating that the LSTM model has made accurate predictions during this initial training phase. Further research and analysis will provide deeper insights into the model's predictive accuracy and performance.

### 5.1.2 Convolutional Neural Network (CNN)

In this section, we develop and train a Convolutional Neural Network (CNN) for predicting Netflix stock price. CNNs are commonly used in visualization but can also be successfully applied to time-series data by using convolutional layers to capture local patterns. **Figure 5.2** shows the code snippet to build and train the CNN model.



```
# Import the necessary libraries
from keras.models import Sequential
from keras.layers import Conv1D, MaxPooling1D, Flatten, Dense, Input

# 3.2 Build & Train CNN model
cnn_model = Sequential()
# Add Input layer
cnn_model.add(Input(shape=(time_step, 1)))
# Add Convolutional layers
cnn_model.add(Conv1D(filters=64, kernel_size=2, activation='relu'))
cnn_model.add(MaxPooling1D(pool_size=2))
# Add Flatten and Dense layers
cnn_model.add(Flatten())
cnn_model.add(Dense(50, activation='relu'))
cnn_model.add(Dense(1))
# Compile the model
cnn_model.compile(optimizer='adam', loss='mean_squared_error')

# Train the model
cnn_model.fit(X, y, batch_size=1, epochs=1)
```

```
3620/3620 ━━━━━━━━━━━━━━━━━━━━ 17s 4ms/step - loss: 0.0044
<keras.src.callbacks.history.History at 0x7938557a6110>
```

Figure 5.2: CNN Model Building & Training

### 5.1.2.1 Model Architecture & Training

The CNN model was designed using the following architecture:

- **Input Layer:** It is defined by **Input (shape=(time_step, 1))**, where **time_step** specifies the number of historical days used to determine the next day's stock price. This assumption ensures that the input sequence is properly ordered for a CNN Model.
- **Convolutional Layers:**
    - o The first layer is a Conv1D layer with 64 filters and a kernel size of 2, using the relu activation function. This layer extracts local features from the time-series data.
    - o A MaxPooling1D layer with a pool size of 2, reduces the size of the data while preserving the most important features.
- **Flatten Layer:** Converts the 2D feature maps from the convolutional and pooling layers into a 1D vector, which is required for subsequent Dense layers.
- **Dense Layers:**
    - o A Dense layer with 50 units and relu activation is added to capture complex patterns.
    - o The final Dense layer results in a single value, which represents the predicted stock price.
- **Compilation:** The model is compiled using the Adam optimizer and mean squared error (MSE) loss function. Adam corrects the learning rate dynamically, while MSE measures the average squared difference between actual and predicted values, making it fit for regression.
- **Training:** The model is trained with a batch size of 1 and for 1 epoch. A batch size of 1 means that the model updates its weights after each sample, which can provide more fine-grained updates but also leads to longer training times. The epoch count indicates that the model processes the entire training dataset once.

### 5.1.2.2 Output of Modeling & Training

The results of the training process for the CNN model are shown in **Figure 5.2** as:

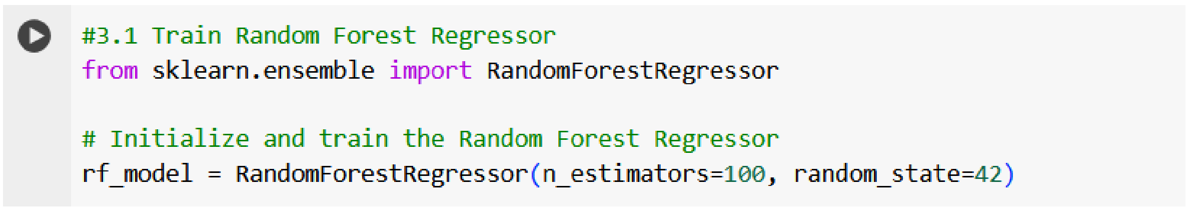
```
3620/3620 ━━━━━━━━━━━━━━━━━━━━ 17s 4ms/step - loss: 0.0044
```

This means that the model was trained on all 3,620 data points in 17 seconds, resulting in a final loss of 0.0044. This loss value indicates the model's performance in predicting stock prices based on training data. Further analyzes were conducted with the same dataset to evaluate the model's predictive accuracy and robustness, focusing on key performance metrics such as MSE and RMSE.

## 5.1.3 Random Forest Regressor

The Random Forest Regressor is an ensemble learning model designed to improve prediction accuracy by comparing the outputs of multiple decision trees. The model works particularly well for complex datasets with many features, such as Netflix subscriber growth data, because it reduces overfitting and improves generalization.

**Figure 5.3** shows the code snippet used to create the Random Forest Regressor model:



```
#3.1 Train Random Forest Regressor
from sklearn.ensemble import RandomForestRegressor

# Initialize and train the Random Forest Regressor
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
```

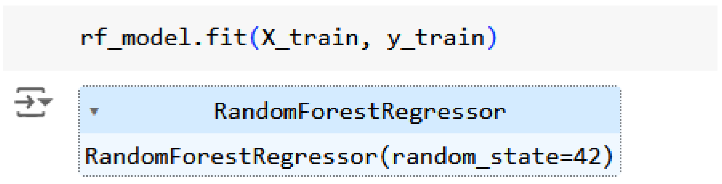
Figure 5.3: Random Forest Regressor Model Building

### 5.1.3.1 Model Architecture

- **n_estimators=100:** This parameter specifies the number of decision trees in the forest. A value of 100 is often used because it provides a good balance between model performance and computational effort. More trees generally improve efficiency, but they also require more computational resources.
- **random_state=42**: This parameter controls the randomness involved in training the model. Setting it to a default value such as 42 ensures that the results are repeatable across runs, which is important for model validation and consistency.

### 5.1.3.2 Model Training

The Random Forest model was trained using the prepared training dataset, as shown in **Figure 5.4**:



```
rf_model.fit(X_train, y_train)

▾ RandomForestRegressor
RandomForestRegressor(random_state=42)
```

Figure 5.4: Random Forest Regressor Model Training

The fit method trains the model using the **X_train** and **y_train** data which corresponding features and target values (subscriber growth rates), respectively. During this process, the model constructs multiple decision trees using different subsets and features of the data.

Each tree predicts individually, and the final prediction is the average of all results. This approach increases accuracy and reduces the likelihood of overfitting.

Through the use of an ensemble approach to the Random Forest Regressor, the model is trained to capture and predict the complex relationships within the subscriber growth data. The model architecture and training setup provide a solid foundation for accurate prediction, preparing the model for further evaluation in subsequent analyses.

## 5.2 Model Evaluation

In this section, we evaluate the performance of the models build and trained in the previous sections. The evaluation process involves calculating key metrics to ensure model accuracy, as well as visualizing the predicted parameters against the actual targets. This comprehensive analysis will provide insight into the effectiveness of LSTM, CNN, and Random Forest models in predicting Netflix stock price and subscriber growth rates.

### 5.2.1 Netflix Stock Price Prediction

To evaluate the performance of the LSTM and CNN models in predicting Netflix stock prices, we used two main parameters:

- Mean Squared Error (MSE)
- Root Mean Squared Error (RMSE)

These metrics show how well the models' predictions match the actual stock price, where MSE measures the squared difference, and RMSE gives the maximum error that can be interpreted in the same units as the data. The evaluation process involves the following major steps:

#### 5.2.1.1 Evaluation Function

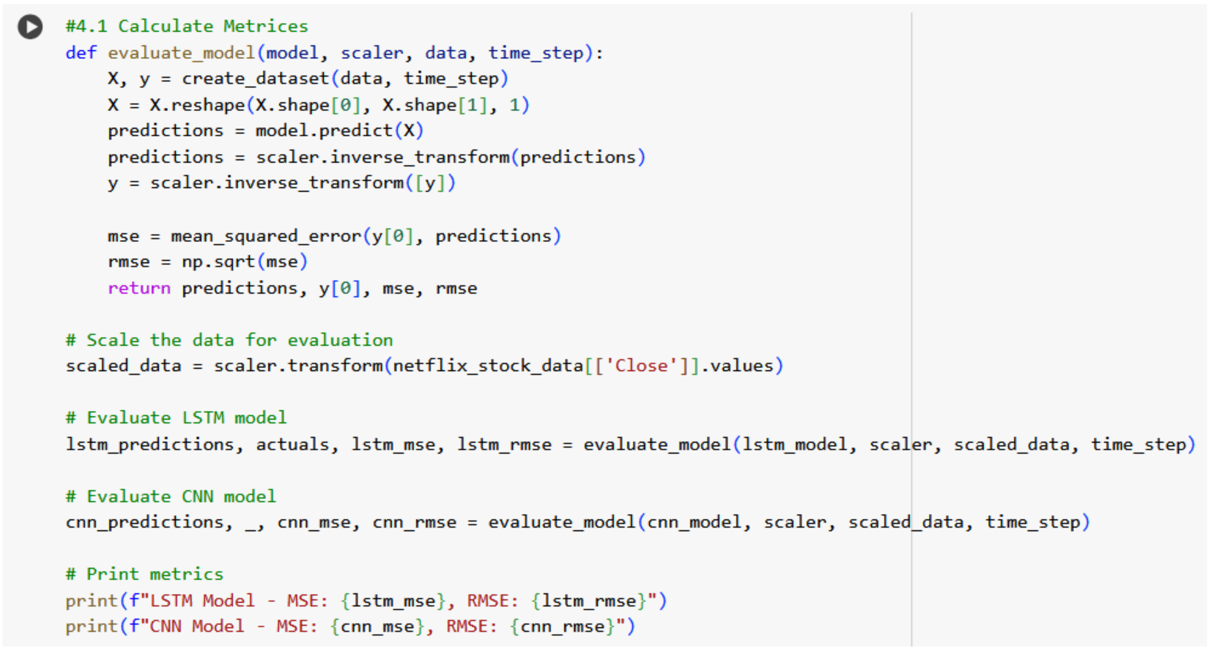
We developed the **evaluate_model** function to provide MSE and RMSE estimates for both models. This function takes the trained model, data scaler, stock price data, and time steps as inputs. It reformats the data to match the model's expected input format, generates predictions, and then inverts the scaling to restore the predictions to their original values. Finally, the function calculates the MSE and RMSE by comparing the model's prediction with the actual stock price. The code snippet for evaluating the LSTM and CNN models is illustrated in **Figure 5.5**, and performs several basic tasks to assess the performance of the models.

The **evaluate_model** function is designed to calculate evaluation metrics for the stock price prediction model. It begins with preparing the data for evaluation. The **create_dataset** function is used to create input features (X) and corresponding target values (y) based on the time step used for the model. The input data (X) is then transformed back into a 3D array to match the expected input shape of the model.

The model makes predictions on the reshaped data, and these predictions are returned to their original form using the **scaler.inverse_transform** method. This step is necessary when converting model results from the scaled range back to the actual price values. Similarly, the actual objective values (y) are converted back to their original scale.

Next, the function calculates two key performance metrics MSE and RMSE. The function also combines these metrics with predictions and actual targets for further analysis.

Once the evaluation function is defined, the data is scaled using the same scaler object used in training. The **evaluate_model** function is then called for both the LSTM and CNN models. Results included predictions, actual values, and analytical parameters for each model. The final metrics are published to provide an overview of how well each model performed in predicting stock prices.



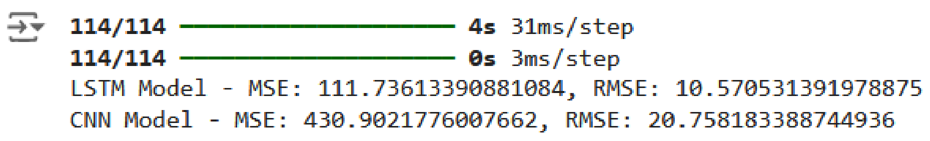
```
#4.1 Calculate Metrices
def evaluate_model(model, scaler, data, time_step):
    X, y = create_dataset(data, time_step)
    X = X.reshape(X.shape[0], X.shape[1], 1)
    predictions = model.predict(X)
    predictions = scaler.inverse_transform(predictions)
    y = scaler.inverse_transform([y])

    mse = mean_squared_error(y[0], predictions)
    rmse = np.sqrt(mse)
    return predictions, y[0], mse, rmse

# Scale the data for evaluation
scaled_data = scaler.transform(netflix_stock_data[['Close']].values)

# Evaluate LSTM model
lstm_predictions, actuals, lstm_mse, lstm_rmse = evaluate_model(lstm_model, scaler, scaled_data, time_step)

# Evaluate CNN model
cnn_predictions, _, cnn_mse, cnn_rmse = evaluate_model(cnn_model, scaler, scaled_data, time_step)

# Print metrics
print(f"LSTM Model - MSE: {lstm_mse}, RMSE: {lstm_rmse}")
print(f"CNN Model - MSE: {cnn_mse}, RMSE: {cnn_rmse}")
```

Figure 5.5: LSTM & CNN Models Evaluation

#### 5.2.1.2 Model Evaluation

The evaluation of the LSTM and CNN models focused on their prediction accuracy of Netflix stock prices, and the results are summarized in **Figure 5.6**. The LSTM model obtained an MSE of 111.74 and an RMSE of 10.57, indicating excellent accuracy with relatively low error. In contrast, the CNN model's MSE of 430.90 and RMSE of 20.76 show higher errors, indicating poorer performance compared to LSTM.

The performance logs in **Figure 5.6** confirm that the LSTM model took longer per step due to its computational complexity, while the CNN model was quicker. The metrics and logs show that the LSTM model is more accurate, while the predictions of the CNN model are less accurate.



```
114/114 ━━━━━━━━━━━━━━━━━━━━ 4s 31ms/step
114/114 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step
LSTM Model - MSE: 111.73613390881084, RMSE: 10.570531391978875
CNN Model - MSE: 430.9021776007662, RMSE: 20.758183388744936
```

Figure 5.6: Evaluation Metrics Output

### 5.2.1.3 Results Visualization

To visually check the model's performance, we plotted the actual stock price against the predictions from both models. **Figure 5.7** shows a comparison between actual prices and LSTM forecasts, while **Figure 5.8** compares these with the forecasts of the CNN model. This pattern highlights the good alignment of the LSTM model with the actual price movement, while the CNN model exhibits asymmetry.

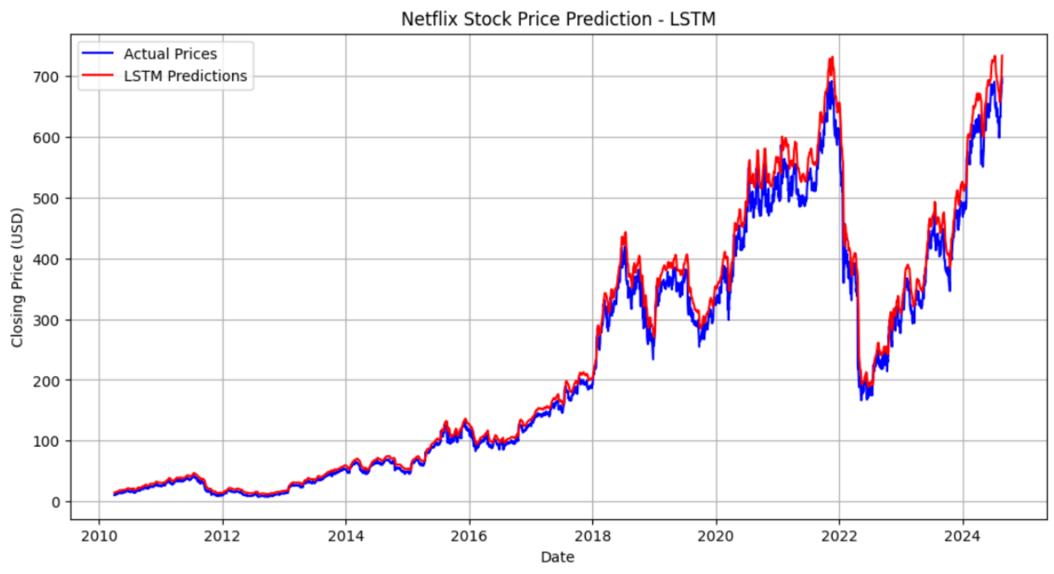


Figure 5.7: Actual Prices vs LSTM Model Predictions

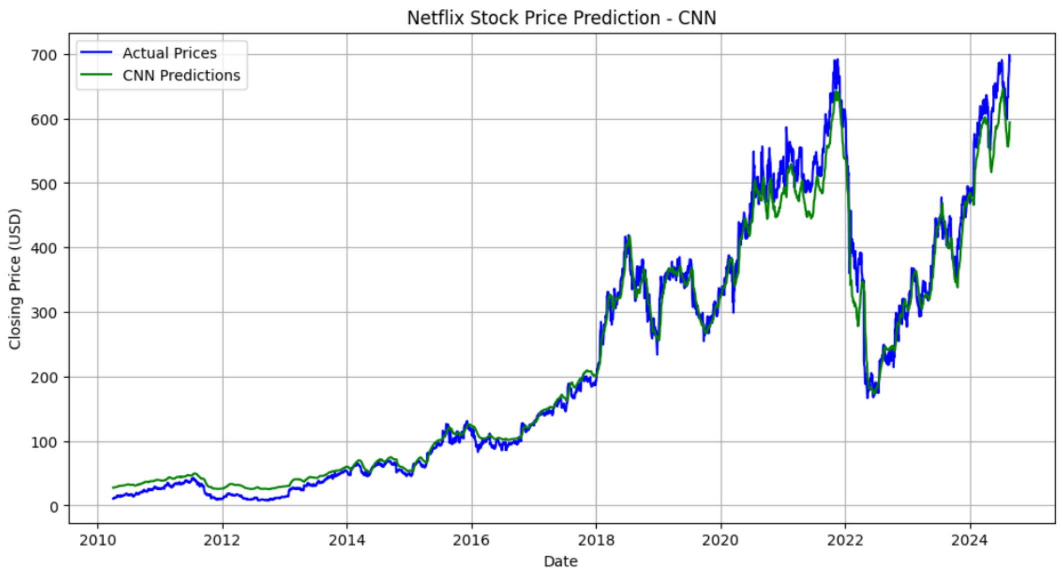


Figure 5.8: Actual Prices vs CNN Model Predictions

Below **Figure 5.9** presents a combined chart that compares actual stock prices with predictions from both the LSTM and CNN models. The chart clearly illustrates the difference between the predicted prices of the two models, highlighting how closely (or not) each relates with the actual prices. This visual comparison highlights the high accuracy of the LSTM model, with forecasts closely tracking the actual value trend, while the CNN model forecasts show significant differences.

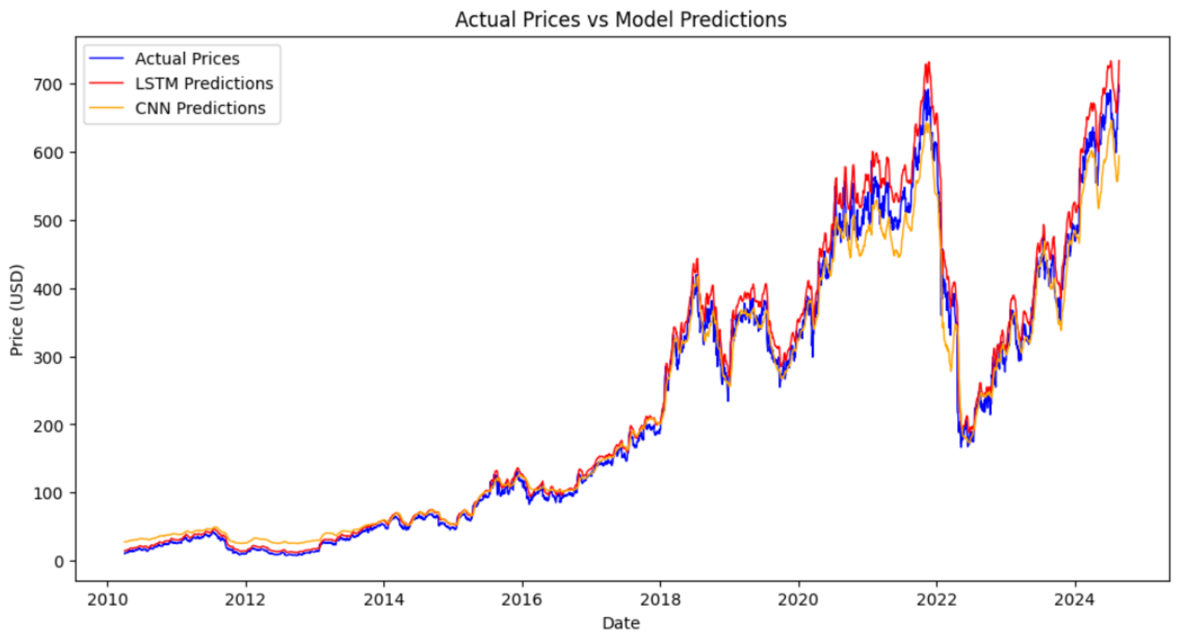


Figure 5.9: Actual Prices vs Model Predictions

## 5.2.2 Netflix Subscriber Growth Rate Prediction

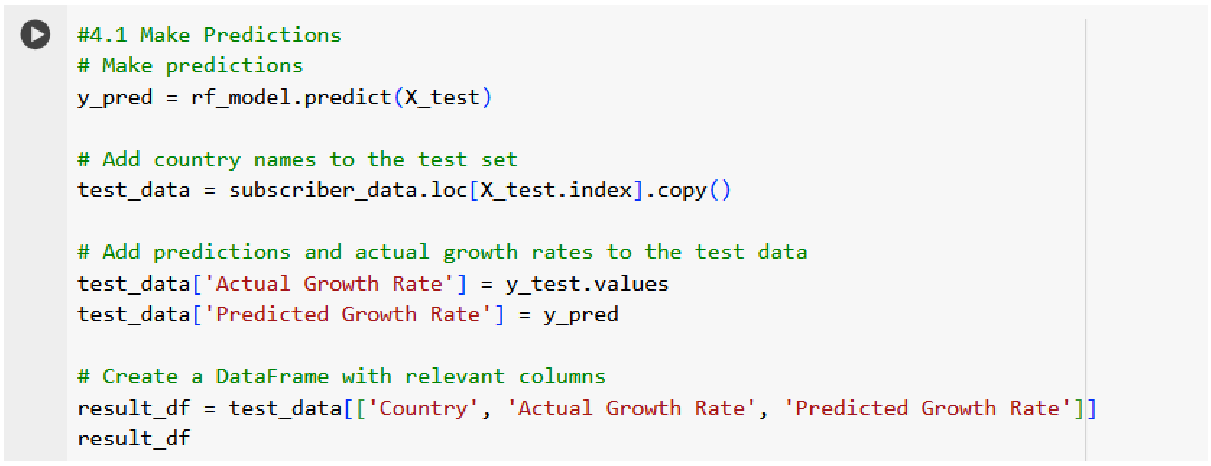
To evaluate the performance of the Random Forest model in predicting subscriber growth rates, we used two main parameters:

- R-squared ($R^2$)
- Root Mean Squared Error (RMSE)

R-squared ($R^2$) measures the predictability of the difference in the dependent variable from the independent variables, and indicates how well the model fits the data. A high $R^2$ value, close to 1, indicates a good fit. RMSE indicates the difference between predicted and observed values, and provides insight into the prediction accuracy of the model. Lower RMSE indicates more accurate predictions. The research process for predicting subscriber growth involves the following steps:

### 5.2.2.1 Model Predictions

Predictions of the Random Forest model were compared with actual growth for each country. To achieve this, we performed several steps, including importing predictions, matching them with actual values, and storing them in a Data Frame. **Figure 5.10** shows part of the code of the steps developed for the model predictions.



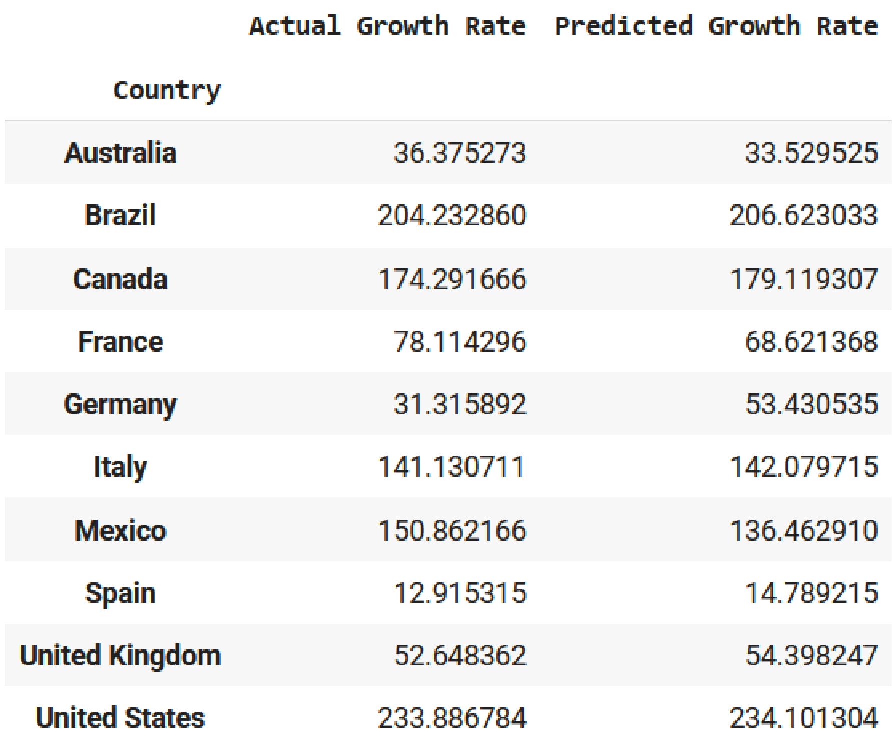
```
#4.1 Make Predictions
# Make predictions
y_pred = rf_model.predict(X_test)

# Add country names to the test set
test_data = subscriber_data.loc[X_test.index].copy()

# Add predictions and actual growth rates to the test data
test_data['Actual Growth Rate'] = y_test.values
test_data['Predicted Growth Rate'] = y_pred

# Create a DataFrame with relevant columns
result_df = test_data[['Country', 'Actual Growth Rate', 'Predicted Growth Rate']]
result_df
```

Figure 5.10: Code Snippet for Model Predictions

In this code, the first step is to use the prediction function of the trained Random Forest model **rf_model** to generate a prediction **y_pred** based on the test data **X_test**. These predictions represent the forecasted growth rates for each country. Next, we match the test data **X_test** with the corresponding country names from the original dataset **subscriber_data**. This strategy ensures that our forecasts and actual growth rates are accurately matched to each country.

We then add two new columns to the test data: Actual Growth Rate from **y_test** and Predicted Growth Rate from **y_pred**. This category allows a direct comparison between actual and predicted growth rates. Finally, we create a Data Frame as **result_df** that extracts and sorts the relevant columns: Country, Actual Results, and Proposed Results. This Data Frame provides a clear and structured view of the model's predictions in comparison to actual growth rates, making it easier to monitor model performance.

The **result_df** output, containing 500 rows and 3 columns, is summarized in **Figure 5.11**, which presents the data aggregated by country. The highest actual and predicted growth rates are seen in the United States, followed by Brazil and Canada. In contrast, the lowest predicted growth rates are found in Spain, Australia, and Germany. This cross-regional comparison provides insight into how well the model predicts growth, with notable accuracy in countries such as the United States and Brazil.

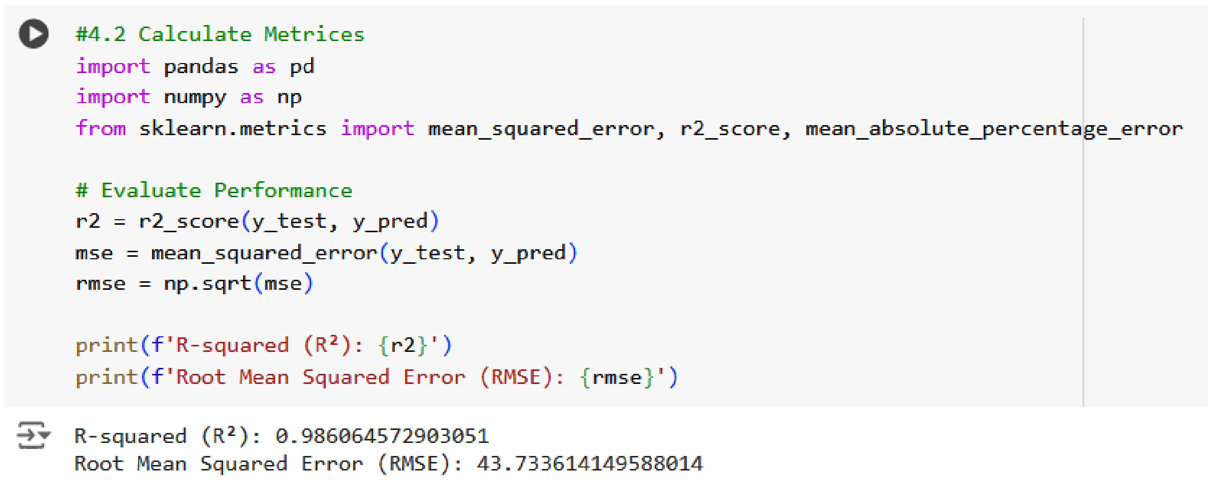


| Country | Actual Growth Rate | Predicted Growth Rate |
|---|---|---|
| Australia | 36.375273 | 33.529525 |
| Brazil | 204.232860 | 206.623033 |
| Canada | 174.291666 | 179.119307 |
| France | 78.114296 | 68.621368 |
| Germany | 31.315892 | 53.430535 |
| Italy | 141.130711 | 142.079715 |
| Mexico | 150.862166 | 136.462910 |
| Spain | 12.915315 | 14.789215 |
| United Kingdom | 52.648362 | 54.398247 |
| United States | 233.886784 | 234.101304 |

Figure 5.11: Output Result of Actual vs Predicted Growth Rate %

### 5.2.2.2 Model Evaluation

The performance of the Random Forest model was evaluated using R2 and RMSE metrics, which produced the following results: an R-squared ($R^2$) value of 0.9861 and a Root Mean Squared Error (RMSE) of 43.73. A high R2 value close to 1 indicates that the model explains almost all of the variance in the target variable, indicating strong predictive accuracy. Furthermore, the lowest RMSE value reflects a small error in the prediction of the model, further confirming its efficiency. The code and output for this calculation are shown in **Figure 5.12**.



```
#4.2 Calculate Metrices
import pandas as pd
import numpy as np
from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_percentage_error

# Evaluate Performance
r2 = r2_score(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)

print(f'R-squared (R²): {r2}')
print(f'Root Mean Squared Error (RMSE): {rmse}')
```

```
R-squared (R²): 0.986064572903051
Root Mean Squared Error (RMSE): 43.733614149588014
```

Figure 5.12: Evaluation Metrics for Random Forest Regressor Model

### 5.2.2.3 Results Visualization

Several visualizations were generated in Python to analyze the model's performance:

1. **Actual vs Predicted Growth Rates by Country: Figure 5.13** shows a bar plot comparing forecasted and actual growth rates across countries, and shows the accuracy of the model. The highest growth rates are seen in the USA, Brazil, and Canada, while the lowest rates are seen in Spain, Australia, and the United Kingdom.

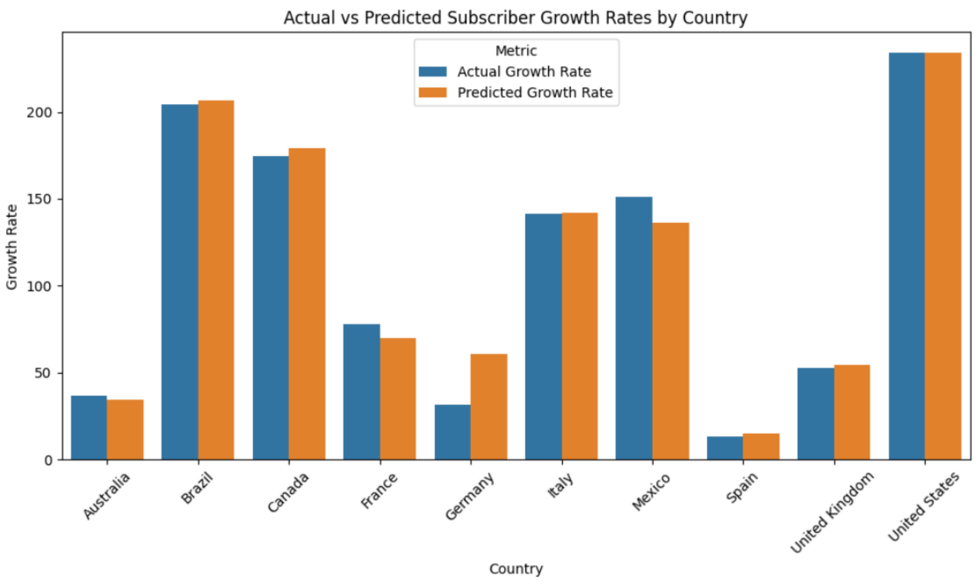


Figure 5.13: Actual vs Predicted Growth Rates by Country

2. **Feature Importance: Figure 5.14** provides a horizontal line showing the importance of the various factors used by the Random Forest model. Monthly subscriber count, country, and subscriber type are the most important factors, while gender, device, and monthly income are less influential.

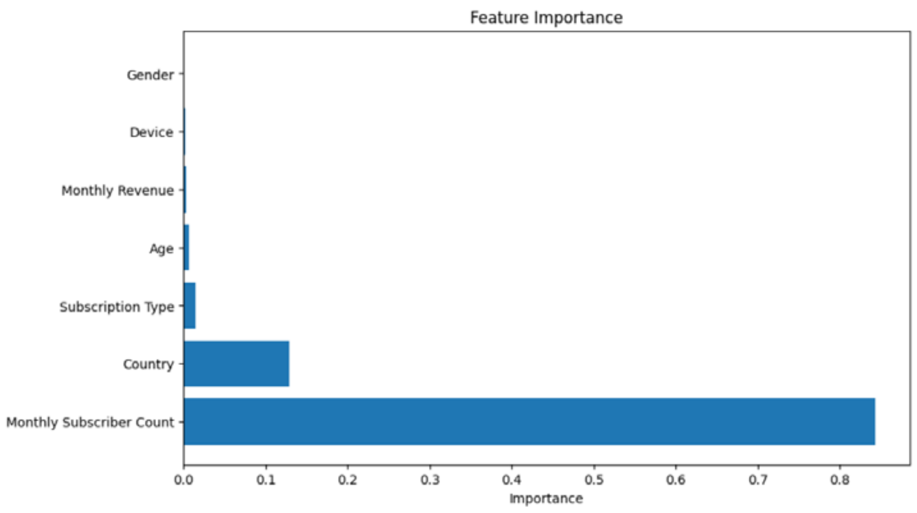


Figure 5.14: Feature Importance in Random Forest Model

3. **Monthly Subscriber Count by Quarter in Each Region: Figure 5.15** shows a bar plot of the average monthly subscriber counts on a quarterly basis for each country. Spain has the highest subscriber counts in 2022Q4, 2022Q3, 2022Q2, and 2023Q1.

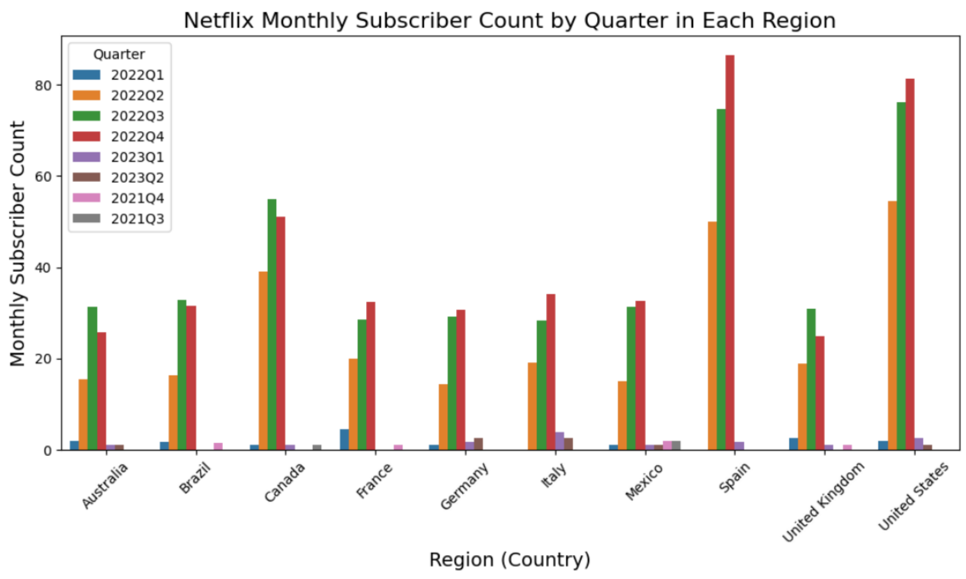


Figure 5.15: Monthly Subscriber Count by Quarter in Each Region

4. **Regional Trends: Figure 5.16** shows the relationship between monthly subscribers and the predicted growth rate for each country. The blue bars represent monthly subscriber counts, with Spain, USA and Canada showing the highest subscribers. The red line indicates the predicted growth rates, where the United States, Italy, and Brazil have the highest growth rates, while Spain, United Kingdom, Germany, Australia, and France show the lowest predicted growth rates. This dual-axis chart effectively emphasizes the difference between subscriber rates and growth rates across different regions.

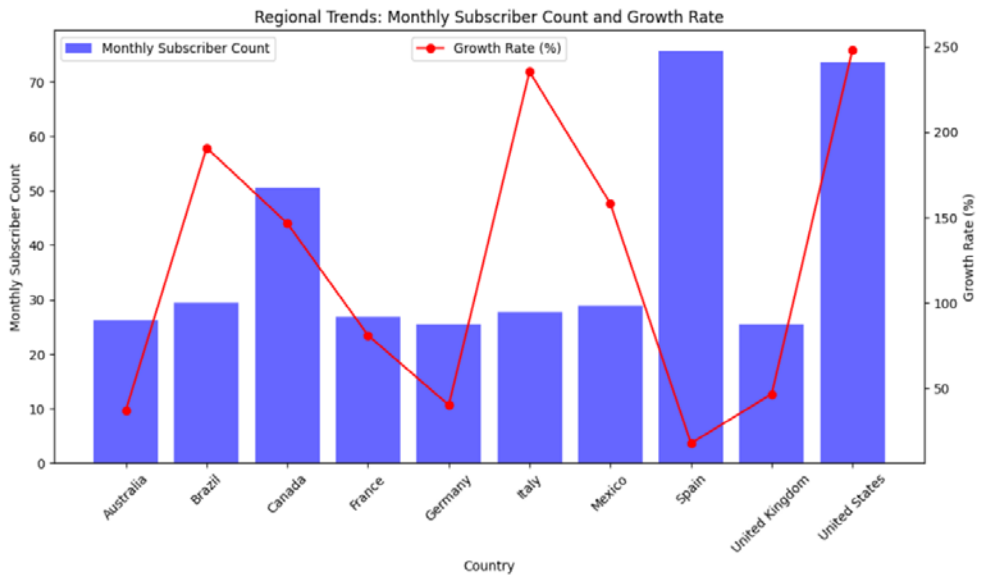


Figure 5.16: Regional Trends: Monthly Subscriber Count and Growth Rate

In conclusion, Chapter 5 provided a detailed explanation of the model implementation and evaluation process for predicting Netflix stock prices and subscriber growth rates. We thoroughly evaluated the performance of our model using a variety of metrics, confirming the accuracy and reliability of the model. Our detailed diagrams showed predictions of the model relative to actual values and the importance of different factors. These studies demonstrated the predictive performance of our model. By examining these results, we established a stronger understanding of the model's capabilities and areas for improvement. The insights gained from these experiments are critical for optimizing the model and preparing it for real-world applications.

In the next chapter, we'll transition into the deployment phase, where we'll focus on real-time forecasting of stock prices and putting all the results into Power BI dashboards. This will allow us to conceptualize and interact with the model's outputs, providing valuable insights for strategic decision making.

# Chapter 6

# Deployment

In Chapter 6, we transition from model evaluation to deployment, focusing on effectively applying our predictive model to real-time data and putting the results into actionable dashboards. This chapter covers several key areas: real-time forecasting of Netflix stock prices, integrating forecasts with Power BI for comprehensive visualization and analysis, and implementing monitoring and maintenance systems will include performance appraisal plan.

We start by developing a real-time forecasting approach, where we use our trained Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN) models to predict Netflix stock prices based on the most recent data. This requires us to take up-to-date stock information, process it through our models, and make actionable predictions.

Previously, our forecasts were based on historical data, which allowed us to use past data to develop and run our models. These historical forecasts were important to assess the accuracy and reliability of our models over time, and provided a solid foundation for understanding their performance. However, real-time forecasting presents a new dimension, where the goal is to predict future stock prices based on the most recent available information. Integrating real-time forecasting capabilities into our operations ensures delivering timely insights and actionable predictions relevant to immediate decision making, which is particularly important for stakeholders needing up-to-date information to make informed decisions.

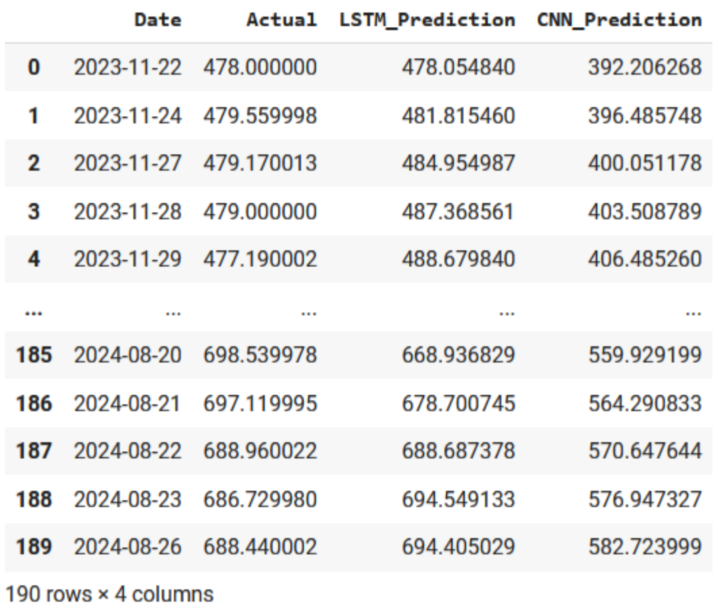
Following this, we combine the prediction results with Power BI, creating a comprehensive dashboard that provides interactive visualizations. These dashboards are designed to give stakeholders a clear and actionable view of both predictive performance and underlying data trends. In addition, we discuss the development of a monitoring and maintenance plan to ensure that the model continues to perform well over time, as well as a performance evaluation process for testing the model and adjusting it as needed. By combining real-time forecasting and detailed visual analysis, and using a robust maintenance and inspection algorithm, this chapter presents the synthesis of the results of our models the holistic approach has served well for strategic decision making and business insights.

## 6.1 Real-Time Netflix Stock Price Prediction

To improve the real-time performance of our predictive models, we developed an algorithm to predict Netflix stock price using both Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN) models. This section describes the process for forecasting real-time stock prices:

1. **Loading Data**: We fetched the most recent stock price data from the last year using the **fetch_historical_data** function. This step ensures that our predictions are based on the latest available information.
2. **Data Preparation**: We checked that there were no values in the data and applied scaling using a pre-fitted scaler to normalize the data. The data were then converted to a format suitable for model input by reformatting.
3. **Modeling**: We used the same LSTM and CNN models that were built and trained in Chapter 5. These models were applied to the prepared data to make predictions. The predictions were then converted to their original scale and arranged in a Data Frame, associating each prediction with its corresponding date.
4. **Evaluation**: Finally, we combined the predictions from both samples into a single Data Frame, made sure that both predictions were the same length, and saved the results as a CSV file for analysis and new combinations. **Figure 6.1** shows the results, showing the actual and predicted values from both models in a combined form.

Together, these steps enable real-time forecasting of Netflix stock prices, providing up-to-date forecasts for informed decision-making.



| | Date | Actual | LSTM_Prediction | CNN_Prediction |
|---|---|---|---|---|
| **0** | 2023-11-22 | 478.000000 | 478.054840 | 392.206268 |
| **1** | 2023-11-24 | 479.559998 | 481.815460 | 396.485748 |
| **2** | 2023-11-27 | 479.170013 | 484.954987 | 400.051178 |
| **3** | 2023-11-28 | 479.000000 | 487.368561 | 403.508789 |
| **4** | 2023-11-29 | 477.190002 | 488.679840 | 406.485260 |
| **...** | ... | ... | ... | ... |
| **185** | 2024-08-20 | 698.539978 | 668.936829 | 559.929199 |
| **186** | 2024-08-21 | 697.119995 | 678.700745 | 564.290833 |
| **187** | 2024-08-22 | 688.960022 | 688.687378 | 570.647644 |
| **188** | 2024-08-23 | 686.729980 | 694.549133 | 576.947327 |
| **189** | 2024-08-26 | 688.440002 | 694.405029 | 582.723999 |

190 rows × 4 columns

Figure 6.1: Real-Time Stock Prediction using LSTM & CNN Model

## 6.2 Integration with Power BI

In this section, we explore how the results of our predictive models were integrated into Power BI for comprehensive visualization and analysis. Power BI provides an interactive platform for creating insightful reports, making it an ideal tool to predict the outcome of our predictive modeling efforts.

### 6.2.1 Exporting & Importing Data Files

To facilitate Power BI integration, the following data files are exported:

**Netflix Stock Price:**

1. **Netflix Stock Price Predictions (netflix_stock_predictions.csv)**: This file contains historical and stock price predictions for Netflix, generated by the models trained in Chapter 5.
2. **Netflix Stock Real-Time Price Predictions (realtime_predictions.csv)**: This file contains real-time predictions of Netflix stock prices using LSTM and CNN models, and provides up-to-date predictions.
3. **Model Metrics (model_metrics.csv)**: This file summarizes the performance metrics (e.g., MSE, RMSE) of both LSTM and CNN models, and provides a comparison of their prediction accuracy.

**Netflix Subscriber Growth Rate:**

1. **Subscriber Data with Growth Rates (subscriber_data_with_growth_rates.csv)**: This file includes processed subscriber data with calculated monthly subscriber numbers and growth rates in different regions.
2. **Subscriber Growth Predictions (subscriber_growth_predictions.csv)**: This file contains predictions of subscriber growth, generated by the model.
3. **Subscriber Metrics (subscriber_metrics.csv)**: This file presents the key performance metrics (e.g., $R^2$, RMSE) for the subscriber growth model.
4. **Feature Importance (feature_importance.csv)**: This file highlights the importance of features in predicting subscriber growth rates.
5. **Correlation Matrix (correlation_matrix.csv)**: This file visualizes the relationships between the various factors affecting client growth, and provides insight into their relative interdependencies.

These files were imported into Power BI using the ***Get Data*** > ***Text/CSV*** option, one at a time. Each file was used to create different visualizations, allowing an in-depth analysis of the results.

### 6.2.2 Results Visualization in Power BI

In this section, we explore how the results of our predictive models are visualized in Power BI, transforming raw data and model outputs into interactive and insightful dashboards.

**Netflix Stock Price:**

To enhance our analysis and make it easier to use, we extended our analytical framework for stock price forecasting by adding predictive models to Power BI. This allowed us to visualize the results clearly, turning Python-based objects into usable insights. Power BI played a key role in the deployment phase, allowing us to present the LSTM and CNN models' predictions in an interactive and user-friendly format.

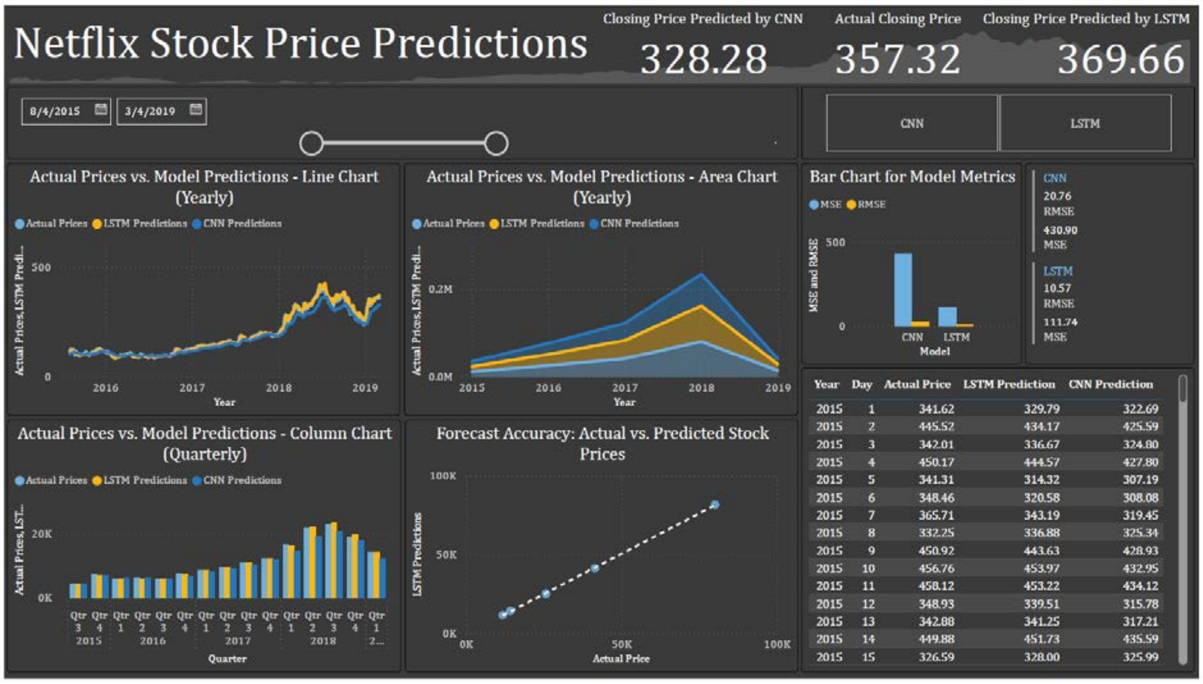
**Figure 6.2** shows the Power BI dashboard created for the Netflix Stock Price predictions using the results from both models. The dashboard has several icons, each serving a different purpose:

1. **Actual Prices vs. Model Predictions Line Chart (Yearly):** This chart shows yearly trends in actual and predicted prices, enabling a clear assessment of the accuracy of models over time. It shows how well the LSTM and CNN models match the long-run true stock price.
2. **Actual Prices vs. Model Predictions Column Chart (Quarterly):** This chart breaks down forecasts by quarter, focusing on more accurate short-term forecasts. It reveals how the models perform on a quarterly basis, and shows where the forecasts were more or less accurate. It helps to visualize the extent of the predicted error.
3. **Actual Prices vs. Model Predictions Area Chart (Yearly):** This chart highlights the variations between actual and predicted values, with shaded areas indicating periods where predictions varied significantly from actual prices. It helps to visualize the extent of the predicted error.
4. **Forecast Accuracy: Actual vs. Predicted Stock Prices:** This chart directly compares actual and forecasted stock prices, revealing how well the models' forecasts match real world prices.
5. **Bar Chart for Model Metrics:** This chart shows the Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) of both models, and visually demonstrates the superior performance of the LSTM model compared to the CNN model.
6. **Multi-row Card for Model Metrics:** This feature displays a summarized view of the key metrics (MSE and RMSE) for each model, making it easy to quickly compare their overall performance.
7. **Table View of Actual vs. Predicted Stock Prices:** This detailed table provides a breakdown of actual and forecasted stock prices, useful for granular analysis of forecast accuracy.

In Addition, the dashboard includes specific metrics for closing prices:

- **Actual Closing Price:** $357.32
- **LSTM Model Predicted closing price:** $369.66
- **CNN Model Predicted closing price:** $328.28

These values are important to understand the practical implications of the models' predictions. The predicted closing price of $328.28 for the CNN model and the predicted closing price of $369.66 for the LSTM model are compared to the actual closing price of $357.32, indicating the long-term performance of each model compared to real-world data. In this view we can easily see that the predicted closing prices of LSTM model are close to the actual prices, also better prediction than CNN model.



8/4/2015
3/4/2019
CNN
LSTM
Actual Prices vs. Model Predictions - Line Chart (Yearly)
Actual Prices
LSTM Predictions
CNN Predictions
Actual Prices, LSTM Predi...
Year
Actual Prices vs. Model Predictions - Area Chart (Yearly)
Actual Prices vs. Model Predictions - Column Chart (Quarterly)
Quarter
Forecast Accuracy: Actual vs. Predicted Stock Prices
LSTM Predictions
Actual Price
Bar Chart for Model Metrics
MSE
RMSE
MSE and RMSE
Model

Netflix Stock Price Predictions

Closing Price Predicted by CNN 328.28

Actual Closing Price 357.32

Closing Price Predicted by LSTM 369.66

CNN
20.76
RMSE
430.90
MSE

LSTM
10.57
RMSE
111.74
MSE

| Year | Day | Actual Price | LSTM Prediction | CNN Prediction |
|---|---|---|---|---|
| 2015 | 1 | 341.62 | 329.79 | 322.69 |
| 2015 | 2 | 445.52 | 434.17 | 425.59 |
| 2015 | 3 | 342.01 | 336.67 | 324.80 |
| 2015 | 4 | 450.17 | 444.57 | 427.80 |
| 2015 | 5 | 341.31 | 314.32 | 307.19 |
| 2015 | 6 | 348.46 | 320.58 | 308.08 |
| 2015 | 7 | 365.71 | 343.19 | 319.45 |
| 2015 | 8 | 332.25 | 336.88 | 325.34 |
| 2015 | 9 | 450.92 | 443.63 | 428.93 |
| 2015 | 10 | 456.76 | 453.97 | 432.95 |
| 2015 | 11 | 458.12 | 453.22 | 434.12 |
| 2015 | 12 | 348.93 | 339.51 | 315.78 |
| 2015 | 13 | 342.88 | 341.25 | 317.21 |
| 2015 | 14 | 449.88 | 451.73 | 435.59 |
| 2015 | 15 | 326.59 | 328.00 | 325.99 |

Figure 6.2: Power BI Dashboard for Netflix Stock Price Predictions

Overall, the Power BI visualizations and the specific closing price metrics provide a comprehensive view of the models' performance, providing both high-level of insights and detailed analysis of predictive accuracy.

**Netflix Subscriber Growth Rate:**

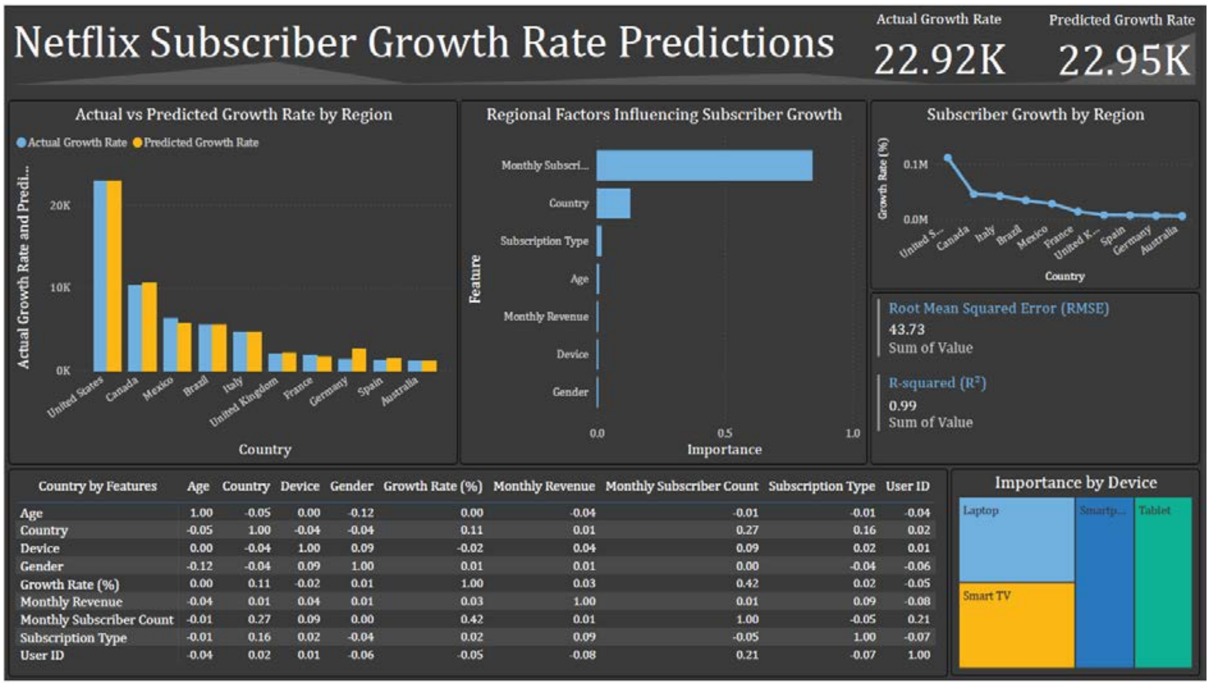
To support the deployment phase, we added a Power BI dashboard to the Python-based visualizations for the subscriber growth rate predictions to provide a comprehensive view of the models' performance. **Figure 6.3** shows the Power BI dashboard developed for Netflix Subscriber Growth Rate prediction. The dashboard has several visualizations that provide insight into the model's performance and the factors that influence subscriber growth rates across regions:

1. **Actual vs. Predicted Growth Rate by Region:** This bar chart shows yearly trends and comparison between actual and predicted growth rates. It highlights how well the model's predictions match real-world data, with the USA exhibiting the highest predicted growth rates.
2. **Regional Factors Influencing Subscriber Growth:** This chart analyses various factors that affect subscriber growth in different regions. It identifies monthly subscriber count, country, and age group as key drivers of growth rate variation.
3. **Correlation Matrix:** This chart visualizes the relationships between various factors affecting subscriber growth, and provides insight into how different factors are related.
4. **Subscriber Growth by Region:** A line chart that highlights growth rate variations between different regions, offering a clear view of how subscriber growth changes around different regions.
5. **Multi-row Card for Model Metrics:** This card summarizes key performance metrics, including R-squared ($R^2$) and Root Mean Squared Error (RMSE), which are 0.986 and 43.73, respectively, showing the model's accuracy.
6. **Feature Importance by Device:** This visualization focuses on the significance of different devices in predicting subscriber growth, with smartphones and tablets being the most influential, followed by laptops and smart TVs.

Additionally, the dashboard provides the following key insights:

- **Actual Subscriber Growth Rate:** 22.92K
- **Predicted Subscriber Growth Rate:** 22.95K

These visualizations provide a detailed overview of the model's predictive capabilities and accuracy, and better capture the distinctions of subscriber growth rate forecasting across regions.



| Country by Features | Age | Country | Device | Gender | Growth Rate (%) | Monthly Revenue | Monthly Subscriber Count | Subscription Type | User ID |
|---|---|---|---|---|---|---|---|---|---|
| Age | 1.00 | -0.05 | 0.00 | -0.12 | 0.00 | -0.04 | -0.01 | -0.01 | -0.04 |
| Country | -0.05 | 1.00 | -0.04 | -0.04 | 0.11 | 0.01 | 0.27 | 0.16 | 0.02 |
| Device | 0.00 | -0.04 | 1.00 | 0.09 | -0.02 | 0.04 | 0.09 | 0.02 | 0.01 |
| Gender | -0.12 | -0.04 | 0.09 | 1.00 | 0.01 | 0.01 | 0.00 | -0.04 | -0.06 |
| Growth Rate (%) | 0.00 | 0.11 | -0.02 | 0.01 | 1.00 | 0.03 | 0.42 | 0.02 | -0.05 |
| Monthly Revenue | -0.04 | 0.01 | 0.04 | 0.01 | 0.03 | 1.00 | 0.01 | 0.09 | -0.08 |
| Monthly Subscriber Count | -0.01 | 0.27 | 0.09 | 0.00 | 0.42 | 0.01 | 1.00 | -0.05 | 0.21 |
| Subscription Type | -0.01 | 0.16 | 0.02 | -0.04 | 0.02 | 0.09 | -0.05 | 1.00 | -0.07 |
| User ID | -0.04 | 0.02 | 0.01 | -0.06 | -0.05 | -0.08 | 0.21 | -0.07 | 1.00 |

Figure 6.3: Power BI Dashboard for Netflix Subscriber Growth Rate Predictions

## 6.3 Monitoring and Maintenance Strategies

To ensure the continued success and reliability of the predictive models, a comprehensive monitoring and maintenance strategy is required. This strategy includes:

- **Performance Monitoring**: Model performance is continuously monitored using key metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared ($R^2$) to track accuracy. Alerts can be set up to notify stakeholders of any delays or slowdowns in performance.
- **Data Updates and Model Retraining**: As new data becomes available, it will be added to the models to ensure it remains relevant to current trends. Additional training will be conducted periodically to address model drift and increase prediction accuracy.
- **System Health Checks**: Conduct regular assessments to ensure that models and supporting services are working properly. This review will focus on data simplicity, model execution times, and accuracy of real-time forecasts.

## 6.4 Regular Performance Reviews and Adjustments

The deployment plan includes a set of procedures for testing the performance of the model. Key activities include:

- **Quarterly Performance Reviews**: Every quarter, a comprehensive review of the model results can be conducted, focusing on forecast accuracy, precision, and adequacy. This analysis will use the Power BI dashboards to analyze trends and identify areas for improvement.

- **Adjustments and Updates**: Based on the results of the analysis, make necessary changes, including tuning model parameters, adding new features, or modifying data preprocessing techniques. This iterative process ensures that the patterns keep up with the dynamic environment of stock prices and subscriber growth.

In conclusion, Chapter 6 provides an overview of the deployment phase, where the focus shifted from the analysis of the predictive model to its application in a real-world setting. This chapter detailed the steps taken to generate a real-time forecast of Netflix stock price using the LSTM and CNN models, followed by combining these forecasts with Power BI to generate advanced visualization and analysis. In addition, we discussed the development of a monitoring and maintenance strategy, as well as the routine performance review processes to ensure the model remains accurate and efficient.

The next chapter, Conclusions and Recommendations, will discuss the main findings from our study, provide a comprehensive analysis of the intervention process, and suggest recommendations for future development.

# Chapter 7

# Conclusion & Recommendations

In this final chapter, we summarize the progress and findings of our work, which has focused on developing and applying predictive models for Netflix stock price and subscriber growth rates. We reflect on the successes, identify limitations, and propose directions for future work to improve the model and its application. This chapter consolidates the project results, evaluates its impact, and suggests steps for future research to continue the current work.

## 7.1 Conclusion

This project has successfully used advanced predictive modeling techniques to gain actionable insights into Netflix's stock price and subscriber growth rates. By using Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN) models, we were able to predict outstanding stock price accuracy. The LSTM model in particular exhibited superior performance compared to the CNN model, as indicated by its lower Mean Squared Error (MSE) and Root Mean Squared Error (RMSE). This application highlights the strengths of LSTM in handling sequential data and capturing temporal dependencies, making it ideally suited for financial forecasting.

In addition to forecasting stock prices, the project also used a Random Forest Regressor to forecast subscriber growth in various locations. The Random Forest model exhibited high accuracy, with an $R^2$ value close to 1, indicating strong predictive ability. This model proved effective in identifying regional variations in subscriber growth, and provided valuable insights into how different markets contribute to Netflix's overall growth track.

The successful implementation of these model for real-time forecasting shows significant improvement. By incorporating the latest stock and subscriber data into our models, we were able to make up-to-date forecasts that are critical for informed decisions. This real-time forecasting capability increases the importance of our insights, providing stakeholders with actionable information that reflects current market conditions.

Additionally, integrating our predictive models with Power BI for visualization and analysis significantly increased the impact of our work. Interactive dashboards created in Power BI transform complex model outputs into easy-to-interpret visualizations. These dashboards provided a comprehensive view of both historical and real-time forecasts, and facilitated in-depth analysis of progress and model performance. By providing a clear and actionable view of the data, Power BI dashboards enhanced the decision-making process and better communicated the predictive insights.

## 7.2 Recommendations for Future Work

While the project reached its ultimate goal, several areas for further exploration and improvement have been identified. These recommendations aim to improve the performance of the models, expand their applicability, and refine the overall assessment process.

1. **Hyperparameter Optimization**: Future work should focus on optimizing the hyperparameters of the LSTM and CNN models. Testing different batch sizes, time periods, and number of studies can improve model performance. The proper refinement of these parameters is important for the accuracy and efficiency of the model, as the choice of hyperparameters greatly influences the training process and the results.
2. **Feature Expansion**: Expanding the feature set used for model training can provide deeper insights and improve prediction accuracy. The inclusion of additional factors such as trading volumes, market sentiment, and macroeconomic indicators can enhance the ability of models to capture complex market dynamics. More detailed compositions can lead to stronger and more accurate forecasts.
3. **Model Variations and Comparisons**: Exploring alternative model architectures and hybrid approaches can yield new insights. Experimental models such as Transformers or combinations of features of LSTM and CNN architectures may reveal potential improvements in prediction accuracy. Comparison of these variables with existing models will help identify the most effective forecasting methods.
4. **Enhanced Data Sources**: Integrating data sources such as sentiment analysis from the media, social media trends, and global economic indicators can strengthen models. These additional data points can provide a more comprehensive view of market conditions and improve forecast accuracy by capturing a wider range of factors.
5. **Model Tuning and Regularization**: Using advanced techniques such as dropout, batch normalization, and learning rate scheduling can address issues such as overfitting and underfitting. These techniques contribute to the stability of model training and improve performance, resulting in more reliable predictions.
6. **Cross-Validation and Testing**: Cross-validation and testing of the model on different datasets, including different stock markets or sectors, can check robustness and

generalizability. Ensuring that the models perform well in different situations will increase their reliability and applicability in different situations.

7. **User Feedback and Dashboard Enhancements**: Collecting feedback from end users about Power BI dashboards can provide valuable insights for improvement. Customized dashboards with additional interactive elements, advanced graphics, can better meet the needs and preferences of the user.

## 7.3 Practical Implications

The results of this work highlight the importance of combining predictive analytics with advanced real-time data and visualization tools. The development of models and dashboards not only provides valuable insights for investment decisions but also supports strategic planning and implementation decisions. By providing a comprehensive overview of prediction and underlying data, the work demonstrates how predictive analytics can be effectively used to inform and guide decision-making processes.

In summary, this chapter has reviewed the progress and limitations of the project, and provided a comprehensive analysis of the results. Recommendations for future work aim to build on existing foundations, address areas for improvement and explore new opportunities to advance predictive modeling capabilities. By pursuing these recommendations, future research can further increase the accuracy, robustness, and applicability of predictive models, contributing to informed and efficient financial and business decision making.

# References

[1] J. Brownlee, Long Short-Term Memory Networks with Python: Develop Sequence Prediction Models with Deep Learning. Machine Learning Mastery, 2021.

[2] K.-j. Kim, "Financial time series forecasting using support," *Neurocomputing,* vol. 55, pp. 307-319, 2003.

[3] P. G. N. T. Aryendra Singh, "An Empirical Research and Comprehensive Analysis of Stock Market Prediction using Machine Learning and Deep Learning techniques," *IOP Conference Series: Materials Science and Engineering,* 2021.

[4] J. W. T. Z. Y. C. F. Y. Cheng Xu, "Prediction of prognosis and survival of patients with gastric cancer by a weighted improved random forest model: an application of machine learning in medicine," *National Library of Medicine,* 2021.

[5] C. Z. Z. Q. W. X. a. J. F. Xiaorui Wang, "A New Forest Growing Stock Volume Estimation Model Based on AdaBoost and Random Forest Model," *Forests,* 2024.

[6] A. D. S. S. A. K. M. Abhyuday, "A Survey on Machine Learning Techniques for," *ASU International Conference in Emerging Technologies for Sustainability and Intelligent Systems (ICETSIS),* p. 10, 2024.

[7] S. G. R. B Shivani, "Analysis of a Stock Exchange and Future Prediction Using LSTM," *4th International Conference on Recent Trends in Computer Science and Technology (ICRTCST-2021),* p. 7, 2021.

[8] P. S. K. M. M. S. S. A. M. B. L. Md Shaik Amzad Basha, "Implementation of Time-Series Analysis: Prediction of Stock Prices using Machine Learning and Deep learning models: A Hybrid Approach," *2022 IEEE North Karnataka Subsection Flagship International Conference (NKCon),* p. 8, 2022.

[9] H. R. Pamuluri, "Predicting User Mobility using Deep Learning Methods," 2020.

[10] B. R. A. K. M. M. I. S. U. I. A. S. W. K. IRFAN ULLAH, "A Churn Prediction Model Using Random Forest: Analysis of Machine Learning Techniques for Churn Prediction and Factor Identification in Telecom Sector," 2019.

[11] D. P. S. Y. Akash Patel, "Prediction of stock market using artificial intelligence," in *Proceedings of the 2021 International Conference on Computer Communication and Informatics (ICCCI)*, 2021.

[12] B. S. Rashi Jaiswal, "A Hybrid Convolutional Recurrent (CNN-GRU) Model for Stock Price Prediction," 2020.

[13] DataScience-Prof, "Understanding Mean Squared Error (MSE) in Machine Learning," 30 April 2024. [Online]. Available: https://medium.com/@TheDataScience-ProF/understanding-mean-squared-error-mse-in-machine-learning-442795910802.

[14] A. Chugh, "MAE, MSE, RMSE, Coefficient of Determination, Adjusted R Squared — Which Metric is Better?," Analytics Vidhya, 8 December 2020. [Online]. Available: https://medium.com/analytics-vidhya/mae-mse-rmse-coefficient-of-determination-adjusted-r-squared-which-metric-is-better-cd0326a5697e.

[15] S. Ahmed, "ML Series 5: Understanding R-squared in Regression Analysis," Medium, 14 February 2024. [Online]. Available: https://medium.com/@sahin.samia/understanding-r-squared-in-regression-analysis-2d8246a63dbb.

# Appendix

The implementation of the predictive models used in this project, including the LSTM, CNN, and RFR, was carried out using Google Colab. The notebook provides a step-by-step breakdown of data loading, preprocessing, model building, training, evaluation, and visualization steps. To ensure transparency and reproducibility, the following resources are available:

1. **Google Colab Code:** The complete code for the predictive models and evaluation. Link to Google Colab Code
2. **Netflix Subscriber Growth DataSource:** The dataset used for subscriber growth rate predictions. Link to Netflix Subscriber Growth DataSource
3. **Power BI Dashboards:** Interactive dashboards visualizing the stock price predictions and subscriber growth analysis. Link to Power BI Dashboards
4. **Project Schedule Plan:** Detailed project timeline and milestones schedule plan. Link to MS Project Schedule Plan
5. **Project Proposal Plan:** The initial project proposal outlining the objectives, scope, and methodologies. Link to Project Proposal Plan

These references provide a complete guide for exploring the project code, dataset, visualizations, schedule, and proposal plan, ensuring a deeper understanding and facilitating replication or further enhancement of the models used.