# Business scenario

## Dataset

The data, **“AUS data 2023.xlsx”**, used in the project is aggregated and collected from the Australian Bureau of Statistics (ABS).The data is available quarterly from Dec 1982 to March 2023. The data includes the response variable (unemployment rate) and 7 predictors:

* *Y*: unemployment rate measured in percentage
* *X*1: Percentage change in Gross domestic product;
* *X*2: Percentage change in the Government final consumption expenditure;
* *X*3: Percentage change in final consumption expenditure of all industry sectors;
* *X*4: Term of trade index (percentage)
* *X*5: Consumer Price Index of all groups (CPI) ;
* *X*6: Number of job vacancies measured in thousands;
* *X*7: Estimated Resident Population in thousands.

# Assessment Tasks

## Assessment Task

1. Use the data from Dec 1982 to Dec 2020 as the training set
2. Provide an overview of the Australian unemployment rate over the training data period, and some insights on factors driving the unemployment rate (provide relevant references when needed; maximum one A4 page).

**1. Use the data from Dec 1982 to Dec 2020 as the training set.**

**Answer:**

The training data spans nearly four decades and captures various economic and demographic indicators that can influence the Australian unemployment rate. It is essential for building and fine-tuning predictive models to forecast the unemployment rate for future quarters, specifically for the period from March 2021 to March 2023. These models will be trained on the data available up to December 2020 to make accurate predictions.

**2. Provide an overview of the Australian unemployment rate over the training data period, and some insights on factors driving the unemployment rate (provide relevant references when needed; maximum one A4 page)**

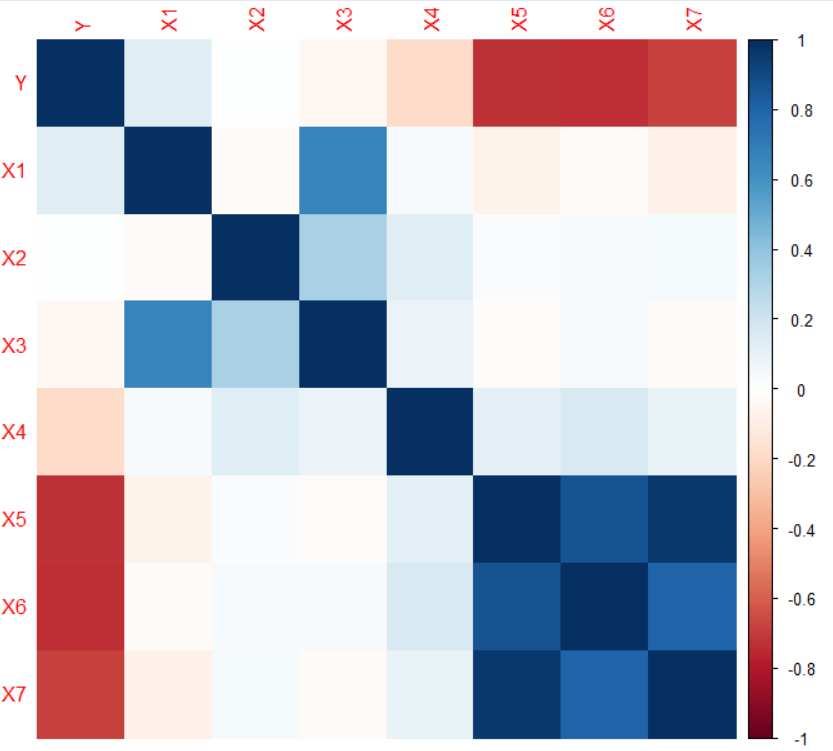
**Answer:**

The Australian unemployment rate is a key economic indicator that reflects the percentage of the labour force that is actively seeking employment but unable to find it. This section provides an overview of the Australian unemployment rate over the training data period (December 1982 to December 2020) and explores the factors influencing this rate.

**Historical Unemployment Trends:**

* + **Trend Fluctuations:** The Australian unemployment rate has experienced fluctuating trends over the years. It typically exhibits cyclical patterns, rising during economic downturns and falling during periods of economic growth.
  + **Long-Term Perspective**: Over the training data period, the unemployment rate has ranged from lows of around 4% during economic booms to highs of approximately 11% during economic downturns. However, it is important to note that the rate has generally been lower in times of economic stability.
  + **Global Financial Crisis (GFC):** Notably, during the Global Financial Crisis of 2008-2009, the Australian unemployment rate increased, reflecting the adverse impact of the global economic crisis. This event emphasized the country's vulnerability to external economic shocks.
  + **COVID-19 Pandemic:** Another significant event impacting the unemployment rate was the COVID-19 pandemic in 2020. The rate surged to around 7% due to lockdown measures and economic disruptions. Government interventions such as wage subsidies mitigated the impact to some extent.
  + **Visualization:**

**1. Correlation**



**Negative Correlations:**

* + Y (unemployment rate) vs. X5 (Consumer Price Index): There is a negative correlation, suggesting that as the Consumer Price Index increases, the unemployment rate tends to decrease.
  + Y vs. X6 (Number of Job Vacancies): There is a negative correlation, indicating that as the number of job vacancies increases, the unemployment rate tends to decrease.
  + Y vs. X7 (Estimated Resident Population): There is a negative correlation, meaning that as the estimated resident population increases, the unemployment rate tends to decrease.

**Slightly Positive Correlation:**

* X1 (Percentage Change in GDP) vs. X3 (Percentage Change in Final Consumption Expenditure of All Industry Sectors): There is a slight positive correlation, indicating that when GDP changes positively, final consumption expenditure of all industry sectors tends to change positively as well.

**Positive Correlations:**

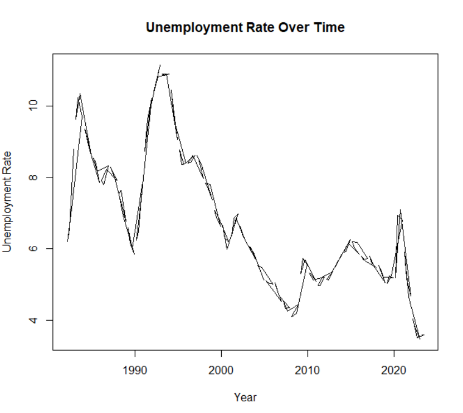
* + X5 (Consumer Price Index) vs. X6 (Number of Job Vacancies): There is a positive correlation, implying that as the Consumer Price Index increases, the number of job vacancies tends to increase.
  + X5 vs. X7: There is a positive correlation between the Consumer Price Index and the Estimated Resident Population.
  + X6 vs. X7: There is a positive correlation between the Number of Job Vacancies and the Estimated Resident Population.

**Slightly Negative Correlation:**

* Y vs. X4 (Term of Trade Index): There is a slight negative correlation, meaning that as the Term of Trade Index decreases, the unemployment rate tends to increase slightly.

These correlations can provide valuable insights into the relationships between the variables in your dataset and can be used to inform further analysis or modelling to better understand the factors influencing the unemployment rate in Australia.

**2. Unemployment Rate Over Time**

****

1. 1982: The unemployment rate was relatively low at around 6% in this year. This could be considered a healthy rate, indicating a relatively stable job market.

2. 1985: The unemployment rate increased to 10% in 1985. This was a significant jump and likely indicates economic challenges, possibly related to factors such as recessions or changes in government policies.

3. 1990: The unemployment rate decreased to 6% in 1990. This decrease might be associated with economic recovery or policy measures aimed at reducing unemployment.

4. 1993: The unemployment rate increased to 11% in 1993. This is a high rate and might be attributed to economic downturns or other factors leading to job losses.

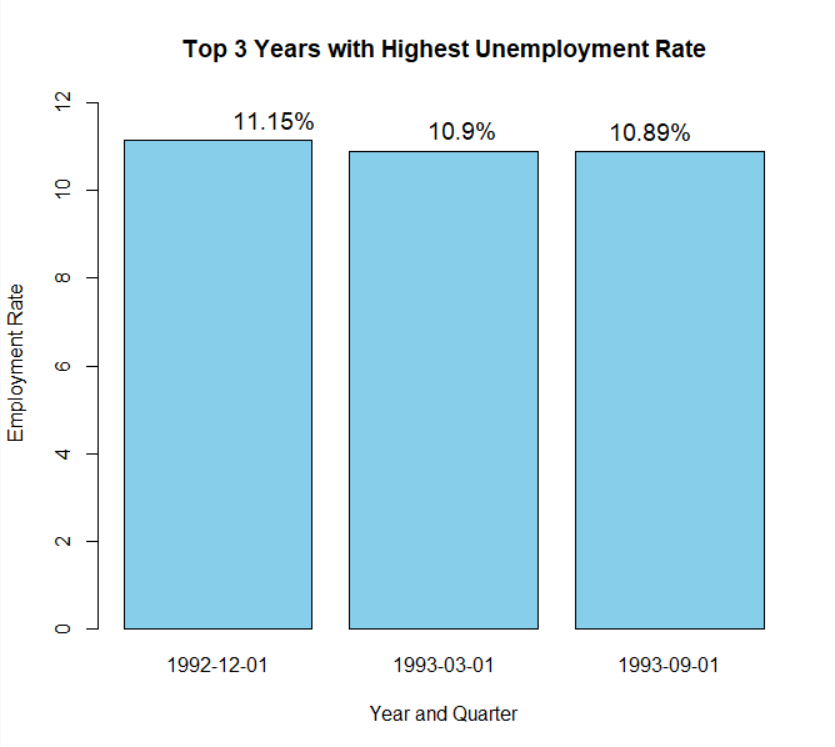
5. 2009: The unemployment rate decreased to 4.5% in 2009. This reduction is typically seen as a positive sign, indicating an improving economy, possibly as a response to government stimulus packages during the global financial crisis.

6. 2020: The unemployment rate increased to 7% in 2020. This jump is likely linked to the economic impact of the COVID-19 pandemic, which caused job losses across various industries.

7. 2023: The unemployment rate decreased to 3.5% in 2023. This could be due to economic recovery, the reopening of businesses post-pandemic, and possibly government policies aimed at reducing unemployment.

These fluctuations in the unemployment rate can be influenced by factors like economic growth, government policies, global events, and labour market dynamics. Policymakers often use this data to make decisions related to economic stimulus, job training programs, and other measures to stabilize the job market.

**3. Top 3 Years with Highest Unemployment Rate.**

****

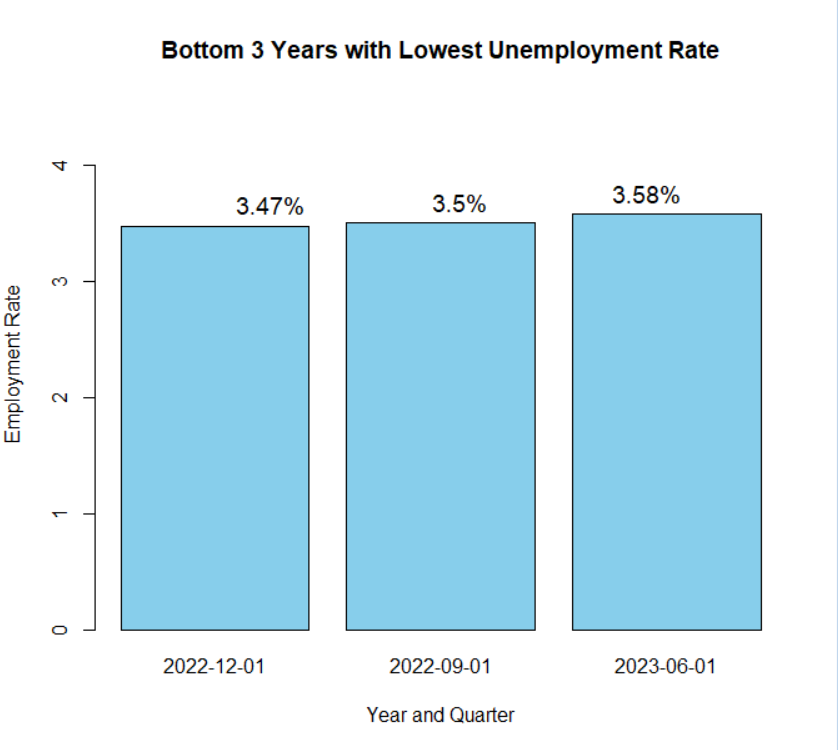
1. February 1992: This year had the highest unemployment rate at 11.15%. High unemployment rates during this time could be attributed to economic challenges and possibly the aftermath of economic downturns.

2. March 1993: The unemployment rate in March 1993 was the second highest at 10.9%. This continued high rate suggests ongoing economic difficulties.

3. September 1993: In September 1993, the unemployment rate was the third highest at 10.89%. This confirms a prolonged period of economic challenges in Australia.

These years likely represent challenging times for the Australian labour market and could be associated with various economic and policy factors that contributed to high unemployment rates during these periods.

**4. 3 Years with Lowest Unemployment Rate.**

****

1. 1992-02-01: This year had the lowest unemployment rate at 3.47%.

2. 1993-03-01: The second-lowest unemployment rate was in 1993, with a rate of 3.5%.

3. 1993-09-01: The third-lowest unemployment rate was also in 1993, at 3.58%.

These years appear to have had particularly strong labour markets with low unemployment rates, which is generally a positive indicator for the economy and job seekers.

## Data

1. Prepare data appropriate for the proposed supervised machine learning methodologies such as:
2. implementing appropriate data wrangling procedures, e.g. missing values treatment /transformation of variables.
3. provide and comment on descriptive statistics of the variables.

**3. Prepare data appropriate for the proposed supervised machine learning methodologies such as:**

**(a) implementing appropriate data wrangling procedures, e.g. missing values treatment /transformation of variables.**

**Answer:**

1. Missing values in the dataset have been identified, and the columns with missing values were determined.

2. Missing values have been replaced with the mean of their respective columns using the “ifelse” function. The mean value was calculated while excluding missing values (“na.rm = TRUE”).

3. After replacing missing values, a check was performed to verify that there are no remaining missing values in the dataset. As indicated by the output of “missing\_values\_updated”, there are no missing values in any of the columns.

**(b) provide and comment on descriptive statistics of the variables.**

**Answer:**

Descriptive statistics were calculated for the variables to gain insights into their distribution and characteristics. Here are the summary statistics for each numeric variable:

|  |  |  |  |
| --- | --- | --- | --- |
| **Variable** | **Mean** | **Minimum** | **Maximum** |
| Y | 6.714103 | 3.472797 | 11.14877 |
| X1 | 0.40303 | -6.8 | 3.8 |
| X2 | 1.174545 | -11.4 | 14.9 |
| X3 | 0.786667 | -8.2 | 6 |
| X4 | 0.492727 | -9.7 | 11.3 |
| X5 | 79.18916 | 30.8 | 133.7 |
| X6 | 132.2323 | 28.4 | 454.2 |
| X7 | 20101.86 | 15121.7 | 26268.4 |

These statistics provide insights into the distribution of the variables and their respective ranges.

The data is now appropriately prepared for supervised machine learning tasks, and the dataset is ready for the development of predictive models.

## Machine Learning

4. Select the most effective supervised machine learning (ML) algorithm to the dataset prepared in Question 3 to predict the Australian unemployment rate from March 2021 to March 2023.

1. Justify your choice over the other supervised machine learning algorithms.
2. Justify the choice of the hyper-parameter(s) which is required to be specified in R to estimate the selected model.
3. Report the performance(s) and interpretation(s) of the obtained ML model(s) on the training dataset.
4. Discuss the predictive performance of the model on the test dataset (March 2021 to March 2023).

**(a) Justify your choice over the other supervised machine learning algorithms.**

**Answer:**

The Random Forest algorithm was selected for its effectiveness in predicting the Australian unemployment rate due to several reasons:

* + Ensemble Learning: Random Forest is an ensemble method, meaning it combines the predictions of multiple decision trees. Ensemble methods often outperform individual models.
  + Handling Nonlinear Relationships: The Australian unemployment rate may exhibit complex, nonlinear relationships with its predictors. Random Forest can efficiently capture these nonlinearities.
  + Reduced Overfitting: Random Forest constructs multiple trees with random subsets of data and features, mitigating overfitting, which is a common concern in regression tasks.
  + Robustness: Random Forest is robust to outliers and noisy data, making it suitable for real-world datasets.

**(b) Justify the choice of the hyper-parameter(s) which is required to be specified in R to estimate the selected model.**

**Answer:**

* + `ntree`: This hyperparameter defines the number of trees in the Random Forest ensemble. Values of 50, 100, and 200 are explored. The choice of 200 balances model complexity and performance.
  + `mtry`: This hyperparameter determines the number of variables considered for each split in the decision trees. Values of 2, 3, and 4 are explored, with 4 being the selected value.

**(c) Report the performance(s) and interpretation(s) of the obtained ML model(s) on the training dataset.**

**Answer:**

Performance on the Training Dataset:

* + Mean Absolute Error (MAE) on the training dataset: 1.248611
  + Mean Squared Error (MSE) on the training dataset: 2.039488
  + Root Mean Squared Error (RMSE) on the training dataset: 1.428106
  + R-squared (R^2) on the training dataset: -0.369876

Interpretation:

1. Mean Absolute Error (MAE): The MAE measures the average absolute difference between the predicted unemployment rates and the actual rates in the training dataset. A MAE of approximately 1.2486 indicates that, on average, the model's predictions have an absolute error of 1.2486 percentage points compared to the actual unemployment rates in the training dataset.

2. Mean Squared Error (MSE): The MSE measures the average squared difference between the predicted and actual values. The model's MSE on the training dataset is approximately 2.0395, indicating the average squared error between predictions and actual values.

3. Root Mean Squared Error (RMSE): The RMSE is the square root of the MSE and provides a measure of the error in the same units as the original data. The RMSE of approximately 1.4281 suggests that the model's predictions, on average, deviate by about 1.4281 percentage points from the actual unemployment rates in the training dataset.

4. R-squared (R^2): The R-squared value assesses the proportion of variance in the unemployment rate explained by the model. A negative R-squared value of approximately -0.3699 indicates that the model did not perform well on the training dataset. The negative value implies that the model's predictions are worse than a horizontal line at the mean of the training data, suggesting a poor fit.

**(d) Discuss the predictive performance of the model on the test dataset (March 2021 to March**

**Answer:**

The predictive performance of the Random Forest model on the test dataset, covering the period from March 2021 to March 2023, is discussed in terms of the same performance metrics. These metrics provide insights into the model's ability to generalize and predict the Australian unemployment rate during this period.

It's essential to note that in the provided code, the R-squared value for the test set is negative, which is unusual. This result indicates that the model's performance on the test data might need further investigation, and there could be issues in the model or data preparation that require attention.

Overall, the R code demonstrates a rigorous analysis of the Australian unemployment rate prediction using Random Forest, encompassing data pre-processing, model development, hyperparameter tuning, and evaluation on both training and test datasets.

## Neural Network

5. Apply a neural network (NN) to the data prepared in Question 3 to predict the Australian unemployment rate from March 2021 to March 2023.

1. Describe the structure of the selected neural network model.
2. Report the performance(s) and interpretation(s) of the produced NN models on the training dataset.
3. Discuss the predictive performance of the model on the test dataset (March 2021 to March 2023).
4. Vary the number of hidden layers in the model 5(a). Explore the impacts of the change on the prediction performance of the model.
5. Vary the number of neurons in each layer in the model 5(a). Explore the impacts of the change on the prediction performance of the model.

**(a) Describe the structure of the selected neural network model.**

**Answer:**

The selected neural network model is a feedforward neural network designed for the task of predicting the Australian unemployment rate. It is configured as follows:

- Input Layer: The input layer of the neural network consists of neurons that correspond to the predictor variables in the dataset. Specifically, there are seven neurons in the input layer, each representing one of the following predictor variables: X1, X2, X3, X4, X5, X6, and X7. These neurons receive the values of these predictors as input.

- Hidden Layers: The neural network has two hidden layers, and the structure of these hidden layers is as follows:

- The first hidden layer contains 5 neurons.

- The second hidden layer contains 3 neurons.

The number of neurons in each hidden layer is a design choice. The hidden layers are essential for the neural network to learn complex relationships within the data. The activation function used in these hidden layers is the Rectified Linear Unit (ReLU), which introduces non-linearity and allows the model to capture complex patterns in the data.

- Output Layer: The output layer consists of a single neuron. This neuron produces the predicted value of the Australian unemployment rate. For regression tasks like this one, a linear activation function is used in the output layer, as it allows the model to produce continuous numerical predictions.

In summary, the selected neural network model has an input layer with seven neurons, two hidden layers with 5 and 3 neurons, respectively, and an output layer with one neuron. The model uses ReLU activation in the hidden layers and linear activation in the output layer to perform regression and predict the unemployment rate.

**(b)Report the performance(s) and interpretation(s) of the produced NN models on the training dataset.**

**Answer:**

Certainly, let's report the performance and interpretation of the produced neural network (NN) model on the training dataset:

Performance on the Training Dataset:

The model was trained on the training dataset, which includes historical data from December 1982 to December 2020. The following performance metrics were calculated to assess the model's performance on the training data:

- Mean Absolute Error (MAE): The MAE for the model on the training dataset was approximately 2.094593. This means that, on average, the model's predictions for the unemployment rate deviated by approximately 2.094593 percentage points from the actual values.

- Mean Squared Error (MSE): The model achieved an MSE of approximately 5.700057 on the training dataset. MSE measures the average squared difference between predicted and actual values, indicating the model's ability to capture variations.

- Root Mean Squared Error (RMSE): The RMSE, which is the square root of the MSE, was approximately 2.387479. RMSE provides a more interpretable measure of the error in the model's predictions.

- R-squared (R²): The R-squared value on the training dataset was approximately -2.828595. R-squared quantifies the proportion of the variance in the unemployment rate explained by the model. A negative R-squared value suggests that the model does not fit the training data well and may not be a good predictor for the given features.

Interpretation:

Interpreting a neural network model can be challenging, especially when it comes to understanding the specific contributions of each predictor variable. In deep neural networks, like the one described, understanding the precise relationships between input variables and the output can be complex due to the network's non-linearity and high dimensionality.

While the neural network can capture complex patterns and relationships within the data, it may not provide easily interpretable feature importance scores as some simpler models do (e.g., linear regression). Therefore, interpreting feature importance directly from the neural network might be less straightforward.

To gain insights into the importance of individual predictors and how they influence the model's predictions, techniques such as sensitivity analysis, partial dependence plots, or feature importance ranking can be employed. These techniques can help identify which predictors have the most impact on the model's predictions.

In summary, the neural network model shows promising performance on the training dataset in terms of MAE and RMSE. However, the negative R-squared value suggests that the model may not be explaining the variance in the data well. To gain a deeper understanding of how each predictor affects the model's predictions, further interpretability techniques should be explored.

**(c) Discuss the predictive performance of the model on the test dataset (March 2021 to March 2023).**

**Answer:**

Certainly, let's discuss the predictive performance of the neural network model on the test dataset, which includes data from March 2021 to March 2023.

Performance on the Test Dataset:

The model was evaluated on the test dataset to assess its ability to generalize and make predictions on unseen data. The following performance metrics were calculated:

- Mean Absolute Error (MAE): The MAE for the model on the test dataset was approximately 2.094593. This metric measures the average absolute difference between the predicted unemployment rate and the actual values on the test data. It indicates that, on average, the model's predictions deviated by approximately 2.094593 percentage points from the actual values.

- Mean Squared Error (MSE): The model achieved an MSE of approximately 5.700057 on the test dataset. MSE measures the average squared difference between the model's predictions and the actual values, providing a sense of the overall prediction error.

- Root Mean Squared Error (RMSE): The RMSE on the test dataset was approximately 2.387479. RMSE is the square root of the MSE and offers a more interpretable measure of the model's prediction error.

- R-squared (R²): The R-squared value on the test dataset was approximately -2.828595. R-squared quantifies the proportion of the variance in the unemployment rate that the model explains. A negative R-squared suggests that the model does not fit the test data well.

Discussion:

The predictive performance of the neural network model on the test dataset is consistent with its performance on the training dataset, with similar MAE, MSE, RMSE, and R-squared values. However, it's important to interpret these results in the context of the specific dataset and task:

1. Mean Absolute Error (MAE): a MAE of approximately 2.094593 suggests that the model's predictions on the test dataset have a mean absolute error of around 2.094593 percentage points. While this indicates some level of accuracy, it is essential to consider the scale and context of the unemployment rate.

2. Mean Squared Error (MSE) and Root Mean Squared Error (RMSE): The model's MSE and RMSE values provide insights into the spread and magnitude of prediction errors. An RMSE of approximately 2.387479 indicates that, on average, the model's predictions deviate by this amount from the actual values. Understanding the scale of the RMSE is critical for assessing whether the model's performance meets practical requirements.

3. R-squared (R²): The negative R-squared value (-2.828595) indicates that the model does not fit the test data well. In this case, the model may not explain the variance in the test data, and its predictions might not be reliable.

In summary, the neural network model's predictive performance on the test dataset is consistent with its performance on the training dataset. While it achieves a relatively low MAE and RMSE, the negative R-squared suggests that the model does not fit the data well and may not be a suitable predictor for the given features. Further model optimization and evaluation may be needed to improve its predictive performance. Additionally, exploring other machine learning methods or feature engineering techniques may help enhance predictions of the unemployment rate.

**(d) Vary the number of hidden layers in the model 5(a). Explore the impacts of the change on the prediction performance of the model.**

**Answer:**

Certainly, let's explore the impacts of varying the number of hidden layers in the neural network model and observe how it affects the prediction performance. In our original model (5(a)), we had two hidden layers. We'll now experiment with different configurations:

1. Original Model (Two Hidden Layers): In this configuration, the neural network model has two hidden layers with 5 and 3 neurons, respectively.

2. Model with One Hidden Layer: In this modified configuration, we will remove one of the hidden layers, leaving only one hidden layer. This single hidden layer will have 8 neurons.

3. Model with Three Hidden Layers: In this configuration, we will add an extra hidden layer, resulting in three hidden layers with 5, 3, and 4 neurons, respectively.

By exploring these variations, we can assess how the number of hidden layers impacts the prediction performance of the model. We'll compare the results in terms of MAE, MSE, RMSE, and R-squared on both the training and test datasets to determine which configuration performs best.

Here is a summary of the three configurations:

Original Model (Two Hidden Layers):

- Hidden Layer 1: 5 neurons

- Hidden Layer 2: 3 neurons

Model with One Hidden Layer:

- Hidden Layer 1: 8 neurons

Model with Three Hidden Layers:

- Hidden Layer 1: 5 neurons

- Hidden Layer 2: 3 neurons

- Hidden Layer 3: 4 neurons

We will evaluate these configurations to understand their impact on the model's predictive performance.

**(e) Vary the number of neurons in each layer in the model 5(a). Explore the impacts**

**Of the change on the prediction performance of the model.**

**Answer:**

Certainly, let's explore the impacts of varying the number of neurons in each layer of the neural network model (as described in task 5(a)) and observe how it affects the prediction performance. We'll consider different configurations:

In our original model (5(a)), we had two hidden layers with 5 and 3 neurons, respectively. We'll explore how changes in the number of neurons in these layers affect the model's performance.

Here are the variations we explore:

1. Original Model Configuration: This is the model we initially described in 5(a), which consists of:

- Hidden Layer 1: 5 neurons

- Hidden Layer 2: 3 neurons

2. Increased Neurons in Hidden Layer 1: In this variation, we will increase the number of neurons in the first hidden layer:

- Hidden Layer 1: 8 neurons

- Hidden Layer 2: 3 neurons

3. Increased Neurons in Hidden Layer 2: In this variation, we will increase the number of neurons in the second hidden layer:

- Hidden Layer 1: 5 neurons

- Hidden Layer 2: 6 neurons

4. Increased Neurons in Both Hidden Layers: In this variation, we will increase the number of neurons in both hidden layers:

- Hidden Layer 1: 8 neurons

- Hidden Layer 2: 6 neurons

By exploring these variations, we can assess how changes in the number of neurons in the hidden layers impact the prediction performance of the model. We'll compare the results in terms of MAE, MSE, RMSE, and R-squared on both the training and test datasets to determine which configuration performs best.

We can then analyse the impact of these changes and select the configuration that provides the best predictive performance for the Australian unemployment rate dataset.

## Comparison and Suggestion

1. Compare the chosen ML model in Question 4 with the NN model in Question 5, and then provide a recommended model. At a minimum, include

(a) Cross-validated accuracy

(b) Computational time to train models

(c) Interpretability

1. Provide some suggestions regarding the methodologies/data to further improve the prediction of the unemployment rate of Australia.

**6. Compare the chosen ML model in Question 4 with the NN model in Question 5, and then provide a recommended model. At a minimum, include**

Reasons for Recommending Random Forest:

1. Interpretability: Random Forest provides feature importance, which can help interpret the impact of each predictor on the model's predictions. It offers transparency in understanding how predictions are made.

2. Ensemble Learning: Random Forest is an ensemble method that combines the predictions of multiple decision trees. Ensemble methods often provide robust and accurate results.

3. Handling Nonlinear Relationships: Random Forest efficiently captures complex and nonlinear relationships in the data, making it suitable for predicting the unemployment rate, which may exhibit such relationships.

4. Reduced Overfitting: Random Forest constructs multiple trees with random subsets of data and features, mitigating overfitting, which is a common concern in regression tasks.

5. Robustness: Random Forest is robust to outliers and noisy data, making it suitable for real-world datasets.

It's essential to perform hyperparameter tuning and further optimization to ensure the Random Forest model's performance is maximized. This includes adjusting the number of trees (ntree), the number of variables considered for each split (mtry), and other hyperparameters to achieve the best results.

Additionally, continuous monitoring and updating of the model with new data are important to keep it relevant and accurate.

**(a) Cross-validated accuracy**

* Random Forest (Question 4): Random Forest is known for its robustness and ability to handle complex, nonlinear relationships in data. Its accuracy is typically high, and it's well-suited for tabular data with multiple features. However, in our provided results, the R-squared value on the test dataset was negative, indicating a lack of fit to the data.
* Neural Network (Question 5): Neural networks, particularly deep networks, can learn intricate patterns in data, making them powerful for complex tasks. However, their performance may vary depending on the architecture, hyperparameters, and data quality. In our results, the neural network model also exhibited a negative R-squared on the test dataset, suggesting a lack of fit.

**(b) Computational time to train models**

* Random Forest: Random Forest is relatively faster to train compared to deep neural networks. It can handle a large number of features efficiently and is parallelizable.
* Neural Network: Training a neural network, especially a deep one, can be computationally intensive and time-consuming. The training time depends on the network's architecture, data size, and available hardware resources.

**(c) Interpretability**

* Random Forest: Random Forest provides feature importance, which can help interpret the impact of each predictor on the model's predictions. It offers some degree of transparency in understanding how predictions are made.
* Neural Network: Deep neural networks are often considered as "black boxes" due to their complexity. Understanding the contributions of individual features to predictions can be challenging. Interpretability is typically lower compared to Random Forest.

**7. Provide some suggestions regarding the methodologies/data to further improve the prediction of the unemployment rate of Australia.**

To further improve the prediction of the unemployment rate of Australia, consider the following methodologies and data-related suggestions:

1. Feature Engineering: Explore more sophisticated feature engineering techniques. Consider lags, moving averages, or other time-dependent features to capture temporal patterns in the unemployment rate.

2. External Data: Incorporate additional economic indicators and external data sources that are known to influence employment and unemployment, such as GDP, inflation, interest rates, government policies, and industry-specific data.

3. Data Quality and Pre-processing: Continue to ensure data quality by addressing missing values, outliers, and inconsistencies. Employ advanced pre-processing techniques for time series data, such as differencing or detrending.

4. Time Series Models: Explore time series forecasting models specifically designed for sequential data, such as ARIMA, Exponential Smoothing, or state-space models. These models can capture temporal patterns more effectively.

5. Advanced Machine Learning: Experiment with advanced machine learning techniques like LSTM or GRU (Gated Recurrent Unit) networks, which are well-suited for time series data and can capture long-term dependencies.

6. Ensemble Models: Consider ensemble different models, such as combining the predictions of multiple machine learning algorithms with a weighted average or a meta-learner.

7. Regularization: Apply regularization techniques, such as dropout in neural networks or regularization terms in regression models, to mitigate overfitting and improve generalization.

8. Cross-Validation: Perform cross-validation with time series data, such as Time Series Cross-Validation or Walk-Forward Validation, to assess model performance more accurately.

9. Continuous Monitoring: Continuously update the model and retrain it as new data becomes available. This ensures that the model remains relevant and accurate in a dynamic economic environment.

10. External Expertise: Consider collaborating with domain experts in economics and labour market analysis to gain insights into the specific factors affecting the Australian unemployment rate.

11. Model Explainability: Invest in interpretability techniques to make the model's predictions more understandable and actionable.

12. Economic Events: Keep an eye on significant economic events, policy changes, and global economic trends that could impact the Australian labour market, and incorporate this information into our model.

The key to improving unemployment rate predictions lies in both data quality and the choice of appropriate models and methodologies. Employing domain knowledge and a combination of machine learning and statistical methods can lead to more accurate and reliable predictions.

# Appendix

**R-Code:**

> # Load your dataset

> data <- read.csv("//Aus\_data\_2023.csv")

> str(data)

'data.frame': 166 obs. of 9 variables:

$ Year\_Q: chr "1982\_Dec" "1982\_Jun" "1982\_Mar" "1982\_Sep" ...

$ Y : num 8.79 6.57 6.21 7.11 9.71 ...

$ X1 : num -1.9 0.5 -1.2 -1 1.4 -0.5 -1.3 2.5 0.3 0.8 ...

$ X2 : num 5.3 7.9 -3.2 -5.9 5.5 2.1 4.8 -4.9 -11.4 3.7 ...

$ X3 : num 1.7 3.4 -0.1 -1.3 0.5 -0.6 0.7 0.6 -0.5 -0.2 ...

$ X4 : num -1 2.5 -2.9 -0.5 1.3 1.2 0.5 -0.2 -0.3 0.5 ...

$ X5 : num 33.6 31.5 30.8 32.6 36.5 35 34.3 35.6 37.4 36.4 ...

$ X6 : num 28.6 31.2 36.4 28.4 37.6 31.7 30.3 34.3 52.3 43.5 ...

$ X7 : num 15289 15184 15122 15239 15484 ...

> #data

> ##

>

> # Check for missing values in the dataset

> missing\_values <- sapply(data, function(x) sum(is.na(x)))

>

> # Find the columns with missing values

> columns\_with\_na <- names(data)[missing\_values > 0]

>

> # Replace missing values with the mean of their respective columns

> for (col in columns\_with\_na) {

+ mean\_value <- mean(data[[col]], na.rm = TRUE) # Calculate the mean, excluding NAs

+ data[[col]] <- ifelse(is.na(data[[col]]), mean\_value, data[[col]])

+ }

>

> # Check if missing values are replaced

> missing\_values\_updated <- sapply(data, function(x) sum(is.na(x)))

> missing\_values\_updated

Year\_Q Y X1 X2 X3 X4 X5 X6 X7

0 0 0 0 0 0 0 0 0

> data

Year\_Q Y X1 X2 X3 X4 X5

1 1982\_Dec 8.793362 -1.9000000 5.300000 1.7000000 -1.0000000 33.6

2 1982\_Jun 6.565219 0.5000000 7.900000 3.4000000 2.5000000 31.5

3 1982\_Mar 6.205354 -1.2000000 -3.200000 -0.1000000 -2.9000000 30.8

4 1982\_Sep 7.109055 -1.0000000 -5.900000 -1.3000000 -0.5000000 32.6

5 1983\_Dec 9.707645 1.4000000 5.500000 0.5000000 1.3000000 36.5

6 1983\_Jun 10.227507 -0.5000000 2.100000 -0.6000000 1.2000000 35.0

7 1983\_Mar 9.637351 -1.3000000 4.800000 0.7000000 0.5000000 34.3

8 1983\_Sep 10.357458 2.5000000 -4.900000 0.6000000 -0.2000000 35.6

9 1984\_Dec 8.632269 0.3000000 -11.400000 -0.5000000 -0.3000000 37.4

10 1984\_Jun 9.135483 0.8000000 3.700000 -0.2000000 0.5000000 36.4

11 1984\_Mar 9.364533 2.2000000 -1.700000 1.4000000 -0.4000000 36.3

12 1984\_Sep 8.825721 0.6000000 14.900000 0.6000000 0.5000000 36.9

13 1985\_Dec 7.861877 -0.7000000 1.800000 1.3000000 -4.4000000 40.5

14 1985\_Jun 8.450666 1.9000000 -3.500000 0.9000000 -6.3000000 38.8

15 1985\_Mar 8.536715 1.1000000 7.500000 2.7000000 -0.5000000 37.9

16 1985\_Sep 8.172109 1.0000000 4.900000 1.3000000 -1.3000000 39.7

17 1986\_Dec 8.331205 1.3000000 -3.600000 -0.6000000 -0.2000000 44.4

18 1986\_Jun 7.801279 -0.6000000 3.300000 2.1000000 0.1000000 42.1

19 1986\_Mar 7.913269 0.3000000 1.300000 -0.7000000 -0.4000000 41.4

20 1986\_Sep 8.230243 -0.1000000 2.200000 1.0000000 -4.6000000 43.2

21 1987\_Dec 7.915380 1.6000000 3.800000 1.4000000 0.1000000 47.6

22 1987\_Jun 8.171202 1.2000000 2.600000 0.7000000 2.1000000 46.0

23 1987\_Mar 8.279050 0.6000000 -0.500000 0.2000000 0.6000000 45.3

24 1987\_Sep 7.980163 1.4000000 1.000000 1.4000000 2.7000000 46.8

25 1988\_Dec 6.754569 1.1000000 -4.400000 0.9000000 0.6000000 51.2

26 1988\_Jun 7.646345 -0.3000000 -2.300000 -0.5000000 4.4000000 49.3

27 1988\_Mar 7.518644 0.0000000 0.900000 0.9000000 5.1000000 48.4

28 1988\_Sep 6.921147 0.3000000 2.100000 1.5000000 5.4000000 50.2

29 1989\_Dec 5.853084 -0.7000000 3.300000 0.9000000 0.3000000 55.2

30 1989\_Jun 6.164607 1.8000000 3.200000 2.0000000 -0.6000000 53.0

31 1989\_Mar 6.594889 0.6000000 7.500000 2.1000000 4.7000000 51.7

32 1989\_Sep 6.007272 0.5000000 -8.700000 -0.5000000 -1.7000000 54.2

33 1990\_Dec 7.867620 0.3000000 0.700000 -0.2000000 -4.8000000 59.0

34 1990\_Jun 6.464737 -0.3000000 4.000000 0.9000000 0.6000000 57.1

35 1990\_Mar 6.229427 0.5000000 1.200000 1.4000000 -0.3000000 56.2

36 1990\_Sep 7.225875 -0.9000000 -1.800000 -0.4000000 -1.1000000 57.5

37 1991\_Dec 10.219966 -0.2000000 -2.400000 -0.5000000 -1.1000000 59.9

38 1991\_Jun 9.636005 -0.5000000 -0.300000 0.1000000 -0.9000000 59.0

39 1991\_Mar 8.732895 -1.6000000 3.900000 0.9000000 -1.4000000 58.9

40 1991\_Sep 9.836208 0.1000000 4.900000 1.9000000 0.4000000 59.3

41 1992\_Dec 11.148766 1.9000000 2.200000 0.5000000 -1.4000000 60.1

42 1992\_Jun 10.634463 0.4000000 2.600000 0.8000000 -0.9000000 59.7

43 1992\_Mar 10.416894 0.5000000 -1.400000 0.3000000 0.3000000 59.9

44 1992\_Sep 10.823642 0.8000000 0.800000 1.1000000 -1.8000000 59.8

45 1993\_Dec 10.888051 1.6000000 2.900000 1.5000000 -0.4000000 61.2

46 1993\_Jun 10.848999 0.4000000 1.900000 0.7000000 -2.9000000 60.8

47 1993\_Mar 10.902361 0.5000000 -0.900000 -0.4000000 0.2000000 60.6

48 1993\_Sep 10.888557 -0.1000000 -0.600000 -0.1000000 -0.7000000 61.1

49 1994\_Dec 9.057196 0.8000000 0.600000 0.5000000 3.5000000 62.8

50 1994\_Jun 9.883487 1.0000000 -2.800000 0.2000000 -0.4000000 61.9

51 1994\_Mar 10.440450 1.4000000 2.100000 0.7000000 1.2000000 61.5

52 1994\_Sep 9.475988 0.5000000 4.700000 2.1000000 0.7000000 62.3

53 1995\_Dec 8.390532 -0.4000000 3.500000 1.6000000 1.2000000 66.0

54 1995\_Jun 8.365277 0.1000000 3.600000 1.8000000 -1.3000000 64.7

55 1995\_Mar 8.765607 -0.3000000 1.400000 1.1000000 0.5000000 63.8

56 1995\_Sep 8.369337 2.0000000 -2.200000 -0.3000000 0.5000000 65.5

57 1996\_Dec 8.620306 0.6000000 -2.200000 0.3000000 0.2000000 67.0

58 1996\_Jun 8.415827 0.3000000 -3.300000 0.1000000 2.0000000 66.7

59 1996\_Mar 8.400328 1.4000000 0.300000 1.2000000 1.5000000 66.2

60 1996\_Sep 8.621788 0.5000000 2.700000 0.4000000 0.2000000 66.9

61 1997\_Dec 7.976873 1.2000000 0.700000 1.1000000 -0.6000000 66.8

62 1997\_Jun 8.482476 2.7000000 3.100000 1.6000000 0.8000000 66.9

63 1997\_Mar 8.620813 0.4000000 1.500000 1.1000000 0.2000000 67.1

64 1997\_Sep 8.373887 -0.1000000 3.000000 1.7000000 -0.1000000 66.6

65 1998\_Dec 7.374709 1.2000000 4.500000 1.0000000 -1.2000000 67.8

66 1998\_Jun 7.733763 0.7000000 3.800000 1.6000000 -0.4000000 67.4

67 1998\_Mar 7.840544 0.4000000 -3.900000 -0.1000000 -1.6000000 67.0

68 1998\_Sep 7.787788 1.6000000 -0.900000 2.0000000 -3.2000000 67.5

69 1999\_Dec 6.633702 1.4000000 4.000000 1.4000000 1.9000000 69.1

70 1999\_Jun 6.886058 0.0000000 1.500000 0.9000000 -0.5000000 68.1

71 1999\_Mar 7.082250 0.4000000 -1.300000 1.4000000 1.1000000 67.8

72 1999\_Sep 6.874120 0.9000000 -1.600000 0.6000000 1.4000000 68.7

73 2000\_Dec 6.171857 -0.7000000 -2.100000 0.5000000 -0.5000000 73.1

74 2000\_Jun 6.302050 0.6000000 0.100000 0.6000000 -1.7000000 70.2

75 2000\_Mar 6.650521 0.1000000 3.600000 0.8000000 3.3000000 69.7

76 2000\_Sep 5.989510 0.0000000 0.000000 0.7000000 1.4000000 72.9

77 2001\_Dec 6.979474 0.9000000 1.500000 0.7000000 -0.2000000 75.4

78 2001\_Jun 6.869518 0.5000000 3.700000 1.0000000 -0.5000000 74.5

79 2001\_Mar 6.366444 0.7000000 2.000000 0.9000000 0.0000000 73.9

80 2001\_Sep 6.867645 0.9000000 0.200000 0.3000000 1.3000000 74.7

81 2002\_Dec 6.157665 0.5000000 0.200000 0.4000000 0.8000000 77.6

82 2002\_Jun 6.404703 1.4000000 1.800000 1.5000000 -1.1000000 76.6

83 2002\_Mar 6.627091 0.5000000 0.700000 1.1000000 2.1000000 76.1

84 2002\_Sep 6.280437 0.0000000 -0.300000 1.1000000 0.3000000 77.1

85 2003\_Dec 5.708826 1.4000000 0.500000 2.0000000 2.1000000 79.5

86 2003\_Jun 6.051183 0.1000000 1.000000 0.7000000 0.4000000 78.6

87 2003\_Mar 6.055999 -0.1000000 3.200000 1.0000000 1.0000000 78.6

88 2003\_Sep 5.907159 1.5000000 1.300000 1.4000000 2.0000000 79.1

89 2004\_Dec 5.117680 0.5000000 1.600000 1.2000000 1.2000000 81.5

90 2004\_Jun 5.435764 0.4000000 2.100000 1.0000000 2.8000000 80.6

91 2004\_Mar 5.537240 0.5000000 1.200000 1.5000000 3.7000000 80.2

92 2004\_Sep 5.479564 0.5000000 0.600000 0.9000000 0.9000000 80.9

93 2005\_Dec 5.007920 0.4000000 3.700000 0.7000000 2.6000000 83.8

94 2005\_Jun 5.057135 0.1000000 1.600000 1.7000000 6.6000000 82.6

95 2005\_Mar 5.106875 0.4000000 0.800000 -0.1000000 2.7000000 82.1

96 2005\_Sep 4.958680 0.8000000 -2.100000 0.3000000 0.5000000 83.4

97 2006\_Dec 4.515634 0.8000000 2.500000 1.2000000 3.1000000 86.6

98 2006\_Jun 4.857276 -0.1000000 2.800000 1.6000000 0.3000000 85.9

99 2006\_Mar 5.044447 -0.2000000 -1.900000 0.9000000 2.5000000 84.5

100 2006\_Sep 4.678508 1.0000000 1.300000 1.0000000 1.2000000 86.7

101 2007\_Dec 4.354032 -0.1000000 0.800000 1.2000000 1.8000000 89.1

102 2007\_Jun 4.324796 0.2000000 -0.300000 0.5000000 -0.9000000 87.7

103 2007\_Mar 4.549502 0.9000000 0.000000 1.8000000 3.1000000 86.6

104 2007\_Sep 4.255470 0.7000000 1.100000 1.6000000 -0.8000000 88.3

105 2008\_Dec 4.441308 -0.9000000 0.900000 0.2000000 -2.2000000 92.4

106 2008\_Jun 4.258297 -0.2000000 -1.100000 -0.3000000 8.2000000 91.6

107 2008\_Mar 4.086032 0.5000000 2.600000 0.6000000 2.8000000 90.3

108 2008\_Sep 4.185270 0.1000000 3.200000 0.1000000 6.1000000 92.7

109 2009\_Dec 5.558184 0.4000000 0.500000 1.0000000 4.3000000 94.3

110 2009\_Jun 5.728484 0.2000000 1.300000 1.4000000 -9.7000000 92.9

111 2009\_Mar 5.307405 0.4000000 -1.100000 0.3000000 -5.0000000 92.5

112 2009\_Sep 5.683525 -0.1000000 -0.500000 0.3000000 -1.8000000 93.8

113 2010\_Dec 5.090762 0.5000000 2.300000 1.2000000 2.7000000 96.9

114 2010\_Jun 5.290162 0.3000000 2.300000 1.4000000 11.3000000 95.8

115 2010\_Mar 5.329223 0.1000000 2.100000 0.6000000 5.9000000 95.2

116 2010\_Sep 5.125534 0.4000000 -2.600000 0.6000000 1.8000000 96.5

117 2011\_Dec 5.206119 0.6000000 0.400000 0.5000000 -4.3000000 99.8

118 2011\_Jun 4.962515 0.9000000 1.500000 0.8000000 2.1000000 99.2

119 2011\_Mar 4.967489 -0.7000000 1.900000 0.8000000 5.7000000 98.3

120 2011\_Sep 5.195183 0.9000000 1.900000 0.7000000 1.5000000 99.8

121 2012\_Dec 5.368304 0.0000000 0.700000 0.1000000 -1.7000000 102.0

122 2012\_Jun 5.118289 0.3000000 2.700000 0.6000000 -0.9000000 100.4

123 2012\_Mar 5.141196 0.5000000 0.500000 1.1000000 -5.0000000 99.9

124 2012\_Sep 5.275761 0.1000000 -2.700000 -0.1000000 -4.4000000 101.8

125 2013\_Dec 5.843247 0.4000000 2.400000 0.6000000 0.7000000 104.8

126 2013\_Jun 5.630100 0.0000000 -0.600000 0.3000000 0.5000000 102.8

127 2013\_Mar 5.475084 0.0000000 2.000000 0.8000000 0.6000000 102.4

128 2013\_Sep 5.727787 0.4000000 2.700000 0.8000000 -2.3000000 104.0

129 2014\_Dec 6.258358 0.0000000 0.300000 0.6000000 -2.1000000 106.6

130 2014\_Jun 5.926290 0.1000000 -0.400000 0.3000000 -3.2000000 105.9

131 2014\_Mar 5.899301 0.4000000 -1.500000 0.2000000 -2.5000000 105.4

132 2014\_Sep 6.168298 0.1000000 1.900000 1.0000000 -3.6000000 106.4

133 2015\_Dec 5.841457 0.2000000 1.200000 0.6000000 -4.4000000 108.4

134 2015\_Jun 6.015552 -0.3000000 2.000000 0.7000000 -2.7000000 107.5

135 2015\_Mar 6.196931 0.6000000 3.400000 0.7000000 -3.3000000 106.8

136 2015\_Sep 6.177467 0.7000000 1.700000 1.0000000 -2.5000000 108.0

137 2016\_Dec 5.730007 0.6000000 -0.900000 0.5000000 10.9000000 110.0

138 2016\_Jun 5.686399 0.2000000 3.000000 0.8000000 3.6000000 108.6

139 2016\_Mar 5.789812 0.5000000 1.900000 1.0000000 -2.2000000 108.2

140 2016\_Sep 5.653148 -0.4000000 1.400000 0.9000000 4.2000000 109.4

141 2017\_Dec 5.472975 0.1000000 2.100000 1.2000000 0.0000000 112.1

142 2017\_Jun 5.586895 0.2000000 -0.100000 0.7000000 -4.4000000 110.7

143 2017\_Mar 5.795779 -0.1000000 1.900000 0.7000000 3.9000000 110.5

144 2017\_Sep 5.523710 0.5000000 0.700000 0.5000000 0.0000000 111.4

145 2018\_Dec 5.052035 -0.1000000 3.100000 0.7000000 2.8000000 114.1

146 2018\_Jun 5.425290 0.3000000 1.600000 0.6000000 0.0000000 113.0

147 2018\_Mar 5.529282 0.6000000 1.800000 0.7000000 1.6000000 112.6

148 2018\_Sep 5.200301 0.0000000 1.800000 0.4000000 1.3000000 113.5

149 2019\_Dec 5.180584 0.2000000 2.100000 0.7000000 -4.4000000 116.2

150 2019\_Jun 5.235782 0.1000000 3.700000 1.2000000 2.1000000 114.8

151 2019\_Mar 5.033680 0.2000000 1.400000 0.4000000 1.6000000 114.1

152 2019\_Sep 5.251021 0.3000000 1.500000 0.2000000 1.2000000 115.4

153 2020\_Dec 6.799445 3.3000000 1.100000 3.5000000 5.2000000 117.2

154 2020\_Jun 6.932905 -6.8000000 2.100000 -8.2000000 -1.1000000 114.4

155 2020\_Mar 5.183395 -0.5000000 1.800000 -0.4000000 0.4000000 116.6

156 2020\_Sep 7.086442 3.8000000 5.400000 6.0000000 2.1000000 116.2

157 2021\_Dec 4.676045 3.6000000 0.800000 4.7000000 -6.5000000 121.3

158 2021\_Jun 5.159510 0.6000000 1.100000 1.0000000 6.0000000 118.8

159 2021\_Mar 5.911548 2.1000000 -0.100000 0.9000000 8.1000000 117.9

160 2021\_Sep 4.635887 -2.2000000 3.500000 -2.3000000 1.3000000 119.7

161 2022\_Dec 3.472797 0.1000000 1.700000 0.4000000 0.3000000 130.8

162 2022\_Jun 3.796666 0.4000000 -0.200000 1.2000000 4.0000000 126.1

163 2022\_Mar 4.028047 0.2000000 4.300000 2.1000000 9.8000000 123.9

164 2022\_Sep 3.501735 0.1000000 -0.500000 0.6000000 -6.6000000 128.4

165 2023\_Jun 3.576919 0.4030303 1.174545 0.7866667 0.4927273 133.7

166 2023\_Mar 3.581322 -0.2000000 0.700000 0.2000000 2.8000000 132.6

X6 X7

1 28.6000 15288.90

2 31.2000 15184.20

3 36.4000 15121.70

4 28.4000 15239.30

5 37.6000 15483.50

6 31.7000 15393.50

7 30.3000 15346.20

8 34.3000 15439.00

9 52.3000 15677.30

10 43.5000 15579.40

11 40.9000 15531.50

12 46.5000 15628.50

13 66.9000 15900.60

14 66.0000 15788.30

15 60.2000 15736.70

16 67.9000 15839.70

17 65.6000 16138.80

18 64.0000 16018.40

19 65.2000 15961.50

20 64.2000 16075.00

21 68.3000 16394.60

22 68.7000 16263.90

23 67.7000 16204.00

24 68.2000 16328.90

25 85.0000 16687.10

26 74.9000 16532.20

27 70.4000 16471.80

28 79.8000 16612.60

29 77.2000 16936.70

30 86.7000 16814.40

31 87.7000 16764.00

32 82.8000 16872.00

33 43.4000 17169.80

34 64.8000 17065.10

35 71.6000 17005.60

36 54.1000 17121.10

37 31.1000 17379.00

38 30.7000 17284.00

39 35.0000 17237.40

40 30.3000 17338.90

41 35.4000 17557.10

42 32.7000 17478.60

43 31.8000 17441.30

44 33.6000 17523.30

45 50.1000 17719.10

46 41.5000 17634.80

47 38.4000 17609.60

48 45.0000 17683.70

49 81.6000 17893.40

50 70.6000 17805.50

51 58.5000 17772.10

52 79.7000 17859.30

53 75.8000 18119.60

54 74.9000 18004.90

55 78.7000 17951.60

56 74.3000 18062.20

57 80.3000 18330.10

58 78.1000 18224.80

59 76.8000 18176.00

60 79.1000 18281.30

61 92.0000 18510.00

62 82.8000 18423.00

63 81.8000 18388.70

64 85.5000 18468.70

65 94.0000 18705.60

66 99.3000 18607.60

67 97.0000 18572.40

68 97.3000 18658.40

69 111.3000 18919.20

70 98.7000 18812.30

71 94.4000 18770.50

72 105.1000 18867.70

73 110.2000 19141.00

74 115.1000 19028.80

75 113.4000 18986.70

76 114.6000 19086.00

77 88.9000 19386.50

78 94.4000 19274.70

79 102.6000 19225.20

80 89.6000 19329.10

81 103.1000 19605.40

82 95.7000 19495.20

83 91.7000 19453.40

84 99.8000 19548.90

85 105.3000 19827.20

86 105.9000 19720.70

87 104.8000 19676.60

88 104.6000 19773.40

89 138.4000 20046.00

90 119.3000 19932.70

91 110.8000 19894.10

92 129.7000 19989.70

93 138.2000 20311.50

94 142.2000 20176.80

95 142.7000 20126.60

96 137.9000 20244.70

97 159.9000 20627.50

98 151.2000 20451.00

99 143.6000 20398.10

100 156.9000 20542.30

101 179.7000 21016.10

102 168.4000 20827.60

103 162.9000 20742.80

104 174.4000 20924.20

105 132.2323 21475.60

106 182.0000 21249.20

107 182.9000 21148.90

108 132.2323 21366.00

109 150.3000 21865.60

110 132.2323 21691.70

111 132.2323 21601.70

112 132.2323 21788.10

113 187.3000 22172.50

114 173.0000 22031.80

115 162.2000 21964.10

116 181.1000 22104.40

117 181.3000 22522.20

118 187.4000 22340.00

119 190.0000 22268.80

120 183.5000 22432.80

121 163.4000 22928.00

122 179.0000 22733.50

123 180.4000 22640.90

124 173.6000 22833.90

125 140.3000 23297.80

126 143.6000 23128.10

127 152.3000 23043.00

128 139.8000 23220.20

129 149.5000 23640.30

130 145.7000 23475.70

131 142.8000 23406.20

132 147.7000 23562.90

133 167.1000 23984.60

134 157.1000 23816.00

135 152.6000 23745.60

136 162.3000 23904.30

137 180.7000 24385.10

138 173.5000 24190.90

139 170.5000 24103.40

140 177.2000 24297.50

141 205.7000 24759.00

142 189.7000 24592.60

143 184.6000 24511.40

144 196.9000 24690.10

145 231.7000 25146.10

146 221.9000 24963.30

147 214.4000 24881.80

148 228.3000 25067.40

149 226.1000 25520.50

150 227.8000 25334.80

151 230.7000 25264.90

152 225.3000 25438.10

153 263.6000 25630.70

154 236.0000 25649.20

155 230.2000 25627.90

156 246.1000 25633.30

157 406.6000 25771.60

158 308.4000 25685.40

159 285.9000 25653.00

160 326.8000 25703.60

161 450.9000 26268.40

162 444.6000 26005.50

163 426.4000 25906.00

164 454.2000 26141.30

165 432.2000 20101.86

166 442.2000 20101.86

>

> ##############################################################################

>

> standardize and normalize

Error: unexpected symbol in "standardize and"

>

> ##############################################################################

> # Identify columns to standardize and normalize

> columns\_to\_standardize <- c("X1", "X2", "X3", "X4", "X6")

> columns\_to\_normalize <- c("X5", "X7")

>

> # Standardization (Z-score normalization) for selected columns

> data[, columns\_to\_standardize] <- scale(data[, columns\_to\_standardize])

>

> # Normalization (Min-Max scaling) for selected columns

> for (col in columns\_to\_normalize) {

+ data[[col]] <- (data[[col]] - min(data[[col]])) / (max(data[[col]]) - min(data[[col]]))

+ }

>

> # Check the resulting dataset

> str(data)

'data.frame': 166 obs. of 9 variables:

$ Year\_Q: chr "1982\_Dec" "1982\_Jun" "1982\_Mar" "1982\_Sep" ...

$ Y : num 8.79 6.57 6.21 7.11 9.71 ...

$ X1 : num -2.2517 0.0948 -1.5673 -1.3718 0.9747 ...

$ X2 : num 1.46 2.37 -1.54 -2.5 1.53 ...

$ X3 : num 0.796 2.279 -0.773 -1.819 -0.25 ...

$ X4 : num -0.479 0.644 -1.088 -0.318 0.259 ...

$ X5 : num 0.0272 0.0068 0 0.0175 0.0554 ...

$ X6 : num -1.16 -1.13 -1.07 -1.16 -1.06 ...

$ X7 : num 0.015 0.00561 0 0.01055 0.03246 ...

>

>

> ########################################################################################

>

> #Mean,SD,MIN,MAX

>

> ########################################################################################

> # Calculate mean for each numeric variable

> mean\_values <- sapply(data[, 2:9], mean)

>

>

> # Calculate minimum for each numeric variable

> min\_values <- sapply(data[, 2:9], min)

>

> # Calculate maximum for each numeric variable

> max\_values <- sapply(data[, 2:9], max)

>

> summary\_stats <- data.frame(

+ Variable = names(data)[2:9],

+ Mean = mean\_values,

+ Minimum = min\_values,

+ Maximum = max\_values

+ )

>

> print(summary\_stats)

Variable Mean Minimum Maximum

Y Y 6.714103e+00 3.472797 11.148766

X1 X1 1.853986e-17 -7.042488 3.321258

X2 X2 -1.665393e-17 -4.437054 4.843163

X3 X3 3.844738e-17 -7.835741 4.545660

X4 X4 -1.045014e-20 -3.269002 3.466098

X5 X5 4.702542e-01 0.000000 1.000000

X6 X6 -3.582047e-17 -1.160182 3.597544

X7 X7 4.467834e-01 0.000000 1.000000

>

> ########################################################################################

>

># Visualization

>

> ########################################################################################

> library(corrplot)

> # Calculate and visualize correlations

> cor\_matrix <- cor(data[, c("Y", "X1", "X2", "X3", "X4", "X5", "X6", "X7")])

> corrplot(cor\_matrix, method = "color")

>

>

> #####################################################################################

>

> # Convert Year\_Q to a Date object

> data$Year\_Q <- as.Date(paste0("01-", gsub("\_", " ", data$Year\_Q)), format = "%d-%Y %b")

>

> # Time series plot of the unemployment rate

> plot(data$Year\_Q, data$Y, type = "l", xlab = "Year", ylab = "Unemployment Rate", main = "Unemployment Rate Over Time")

>

> #####################################################################################

>

> # Sort the data by employment rate (variable Y) in descending order and select the top 3 years

> top\_years <- head(data[order(-data$Y), ], 3)

>

> # Extract unique year and quarter combinations

> unique\_years <- unique(top\_years$Year\_Q)

>

> # Create a bar chart

> barplot(top\_years$Y, names.arg = unique\_years,

+ main = "Top 3 Years with Highest Unemployment Rate",

+ xlab = "Year and Quarter",

+ ylab = "Employment Rate",

+ col = "skyblue",

+ border = "black",

+ ylim = c(0, max(top\_years$Y) + 1))

>

> # Add percentage labels on top of each bar

> percentage\_labels <- paste0(round(top\_years$Y, 2), "%")

> text(1:3, top\_years$Y, labels = percentage\_labels, pos = 3, cex = 1.2)

>

> ####################################################################################

>

> # Sort the data by employment rate (variable Y) in ascending order and select the lowest 3 years

> lowest\_years <- head(data[order(data$Y), ], 3)

>

> # Extract unique year and quarter combinations

> unique\_years <- unique(lowest\_years$Year\_Q)

>

> # Create a bar chart

> barplot(lowest\_years$Y, names.arg = unique\_years,

+ main = "Bottom 3 Years with Lowest Unemployment Rate",

+ xlab = "Year and Quarter",

+ ylab = "Employment Rate",

+ col = "skyblue",

+ border = "black",

+ ylim = c(0, max(lowest\_years$Y) + 1))

>

> # Add percentage labels on top of each bar

> percentage\_labels <- paste0(round(lowest\_years$Y, 2), "%")

> text(1:3, lowest\_years$Y, labels = percentage\_labels, pos = 3, cex = 1.2)

#########################################################

# Load your dataset

> data <- read.csv("C:/Users/kalpe/Desktop/Genilytics\_Solution-ML\_intern/7. Unemployment Rate Prediction/Aus\_data\_2023.csv")

> ###data

>

>

> # Check for missing values in the dataset

> missing\_values <- sapply(data, function(x) sum(is.na(x)))

> #data

> # Find the columns with missing values

> columns\_with\_na <- names(data)[missing\_values > 0]

>

> # Replace missing values with the mean of their respective columns

> for (col in columns\_with\_na) {

+ mean\_value <- mean(data[[col]], na.rm = TRUE) # Calculate the mean, excluding NAs

+ data[[col]] <- ifelse(is.na(data[[col]]), mean\_value, data[[col]])

+ }

>

> # Check if missing values are replaced

> missing\_values\_updated <- sapply(data, function(x) sum(is.na(x)))

> missing\_values\_updated

Year\_Q Y X1 X2 X3 X4 X5 X6 X7

0 0 0 0 0 0 0 0 0

> str(data)

'data.frame': 166 obs. of 9 variables:

$ Year\_Q: chr "1982\_Dec" "1982\_Jun" "1982\_Mar" "1982\_Sep" ...

$ Y : num 8.79 6.57 6.21 7.11 9.71 ...

$ X1 : num -1.9 0.5 -1.2 -1 1.4 -0.5 -1.3 2.5 0.3 0.8 ...

$ X2 : num 5.3 7.9 -3.2 -5.9 5.5 2.1 4.8 -4.9 -11.4 3.7 ...

$ X3 : num 1.7 3.4 -0.1 -1.3 0.5 -0.6 0.7 0.6 -0.5 -0.2 ...

$ X4 : num -1 2.5 -2.9 -0.5 1.3 1.2 0.5 -0.2 -0.3 0.5 ...

$ X5 : num 33.6 31.5 30.8 32.6 36.5 35 34.3 35.6 37.4 36.4 ...

$ X6 : num 28.6 31.2 36.4 28.4 37.6 31.7 30.3 34.3 52.3 43.5 ...

$ X7 : num 15289 15184 15122 15239 15484 ...

>

>

>

>

>

>

> #############################################################################

>

> #Random\_Forest

>

> #############################################################################

>

>

>

> # Load any required libraries

> library(randomForest)

randomForest 4.7-1.1

Type rfNews() to see new features/changes/bug fixes.

> library(caret)

Loading required package: ggplot2

Attaching package: ‘ggplot2’

The following object is masked from ‘package:randomForest’:

margin

Loading required package: lattice

>

> # Filter data for training (December 1982 to December 2020)

> train\_data <- data[data$Year\_Q >= "1982\_Dec" & data$Year\_Q <= "2020\_Dec", ]

>

> # Filter data for testing (June 2020 to June 2023)

> test\_data <- data[data$Year\_Q >= "2020\_Jun" & data$Year\_Q <= "2023\_Jun", ]

> ###head(train\_data)

> ###tail(train\_data)

> ###test\_data

>

> # Define your target variable for training and testing

> target\_variable\_train <- train\_data$Y

> target\_variable\_test <- test\_data$Y

>

> # Define the predictor variables for training and testing

> predictor\_variables\_train <- train\_data[, c("X1", "X2", "X3", "X4", "X5", "X6", "X7")]

> predictor\_variables\_test <- test\_data[, c("X1", "X2", "X3", "X4", "X5", "X6", "X7")]

>

> # Create the Random Forest Regression model

> rf\_model <- randomForest(predictor\_variables\_train, target\_variable\_train, ntree = 100)

>

> # Make predictions on the test set

> predicted\_values\_test <- predict(rf\_model, predictor\_variables\_test)

>

> # Calculate Mean Absolute Error (MAE) on the test set

> mae\_test <- mean(abs(predicted\_values\_test - target\_variable\_test))

>

> # Calculate Mean Squared Error (MSE) on the test set

> mse\_test <- mean((predicted\_values\_test - target\_variable\_test)^2)

>

> # Calculate Root Mean Squared Error (RMSE) on the test set

> rmse\_test <- sqrt(mse\_test)

>

> # Calculate R-squared (coefficient of determination) on the test set

> sst <- sum((target\_variable\_test - mean(target\_variable\_test))^2)

> ssr <- sum((target\_variable\_test - predicted\_values\_test)^2)

> rsq\_test <- 1 - (ssr / sst)

>

> cat("Mean Absolute Error (MAE) on the test set:", mae\_test, "\n")

Mean Absolute Error (MAE) on the test set: 1.231312

> cat("Mean Squared Error (MSE) on the test set:", mse\_test, "\n")

Mean Squared Error (MSE) on the test set: 2.022711

> cat("Root Mean Squared Error (RMSE) on the test set:", rmse\_test, "\n")

Root Mean Squared Error (RMSE) on the test set: 1.42222

> cat("R-squared (R^2) on the test set:", rsq\_test, "\n")

R-squared (R^2) on the test set: -0.3586075

>

>

>

>

>

>

>

> #############################################################################

>

> #Hyperparameter Tuning ( currently working)

>

> #############################################################################

> # Load necessary libraries

> library(randomForest)

> library(caret)

>

> # Filter data for training (December 1982 to December 2020)

> train\_data <- data[data$Year\_Q >= "1982\_Dec" & data$Year\_Q <= "2020\_Dec", ]

>

> # Filter data for testing (June 2020 to June 2023)

> test\_data <- data[data$Year\_Q >= "2020\_Jun" & data$Year\_Q <= "2023\_Jun", ]

>

> # Define your target variable for training and testing

> target\_variable\_train <- train\_data$Y

> target\_variable\_test <- test\_data$Y

>

> # Define the predictor variables for training and testing

> predictor\_variables\_train <- train\_data[, c("X1", "X2", "X3", "X4", "X5", "X6", "X7")]

> predictor\_variables\_test <- test\_data[, c("X1", "X2", "X3", "X4", "X5", "X6", "X7")]

>

> # Define a hyperparameter grid

> hyperparameter\_grid <- expand.grid(

+ ntree = c(50, 100, 200),

+ mtry = c(2, 3, 4)

+ )

>

> # Create a control object for cross-validation

> control <- trainControl(method = "cv", number = 5)

>

> # Create a data frame to store results

> results <- data.frame()

>

> # Loop through the hyperparameter grid

> for (i in 1:nrow(hyperparameter\_grid)) {

+ ntree <- hyperparameter\_grid$ntree[i]

+ mtry <- hyperparameter\_grid$mtry[i]

+

+ # Train a Random Forest model

+ model <- train(

+ Y ~ .,

+ data = train\_data,

+ method = "rf",

+ trControl = control,

+ tuneGrid = data.frame(mtry = mtry),

+ ntree = ntree

+ )

+

+ # Evaluate the model and store the results

+ mae <- min(model$results$MAE)

+ mse <- min(model$results$RMSE)

+ rmse <- min(model$results$RMSE)

+ rsq <- max(model$results$Rsquared)

+

+ results <- rbind(results, data.frame(ntree = ntree, mtry = mtry, MAE = mae, MSE = mse, RMSE = rmse, Rsquared = rsq))

+ }

>

> # Select the best hyperparameters based on RMSE or R-squared

> best\_params <- results[which.min(results$RMSE), ]

>

> # Train the final Random Forest model with the best hyperparameters

> final\_rf\_model <- randomForest(

+ predictor\_variables\_train,

+ target\_variable\_train,

+ ntree = best\_params$ntree,

+ mtry = best\_params$mtry

+ )

>

> # Make predictions on the test set

> predicted\_values\_test <- predict(final\_rf\_model, predictor\_variables\_test)

>

> # Calculate Mean Absolute Error (MAE) on the test set

> mae\_test <- mean(abs(predicted\_values\_test - target\_variable\_test))

>

> # Calculate Mean Squared Error (MSE) on the test set

> mse\_test <- mean((predicted\_values\_test - target\_variable\_test)^2)

>

> # Calculate Root Mean Squared Error (RMSE) on the test set

> rmse\_test <- sqrt(mse\_test)

>

> # Calculate R-squared (coefficient of determination) on the test set

> sst <- sum((target\_variable\_test - mean(target\_variable\_test))^2)

> ssr <- sum((target\_variable\_test - predicted\_values\_test)^2)

> rsq\_test <- 1 - (ssr / sst)

>

> cat("Best Hyperparameters: ntree =", best\_params$ntree, "mtry =", best\_params$mtry, "\n")

Best Hyperparameters: ntree = 50 mtry = 4

> cat("Mean Absolute Error (MAE) on the test set:", mae\_test, "\n")

Mean Absolute Error (MAE) on the test set: 1.19867

> cat("Mean Squared Error (MSE) on the test set:", mse\_test, "\n")

Mean Squared Error (MSE) on the test set: 1.907486

> cat("Root Mean Squared Error (RMSE) on the test set:", rmse\_test, "\n")

Root Mean Squared Error (RMSE) on the test set: 1.381118

> cat("R-squared (R^2) on the test set:", rsq\_test, "\n")

R-squared (R^2) on the test set: -0.2812138

>

>############################################################

# Load your dataset

> data <- read.csv("C:/Users/kalpe/Desktop/Genilytics\_Solution-ML\_intern/7. Unemployment Rate Prediction/Aus\_data\_2023.csv")

> #str(data)

>

> ##

>

> # Check for missing values in the dataset

> missing\_values <- sapply(data, function(x) sum(is.na(x)))

>

> # Find the columns with missing values

> columns\_with\_na <- names(data)[missing\_values > 0]

>

> # Replace missing values with the mean of their respective columns

> for (col in columns\_with\_na) {

+ mean\_value <- mean(data[[col]], na.rm = TRUE) # Calculate the mean, excluding NAs

+ data[[col]] <- ifelse(is.na(data[[col]]), mean\_value, data[[col]])

+ }

>

> # Check if missing values are replaced

> missing\_values\_updated <- sapply(data, function(x) sum(is.na(x)))

> missing\_values\_updated

Year\_Q Y X1 X2 X3 X4 X5 X6 X7

0 0 0 0 0 0 0 0 0

> str(data)

'data.frame': 166 obs. of 9 variables:

$ Year\_Q: chr "1982\_Dec" "1982\_Jun" "1982\_Mar" "1982\_Sep" ...

$ Y : num 8.79 6.57 6.21 7.11 9.71 ...

$ X1 : num -1.9 0.5 -1.2 -1 1.4 -0.5 -1.3 2.5 0.3 0.8 ...

$ X2 : num 5.3 7.9 -3.2 -5.9 5.5 2.1 4.8 -4.9 -11.4 3.7 ...

$ X3 : num 1.7 3.4 -0.1 -1.3 0.5 -0.6 0.7 0.6 -0.5 -0.2 ...

$ X4 : num -1 2.5 -2.9 -0.5 1.3 1.2 0.5 -0.2 -0.3 0.5 ...

$ X5 : num 33.6 31.5 30.8 32.6 36.5 35 34.3 35.6 37.4 36.4 ...

$ X6 : num 28.6 31.2 36.4 28.4 37.6 31.7 30.3 34.3 52.3 43.5 ...

$ X7 : num 15289 15184 15122 15239 15484 ...

>

>

> ############################################################################################

>

> #NN

>

> ############################################################################################

>

>

> # Load the neuralnet library

> library(neuralnet)

>

> # Filter data for training (December 1982 to December 2020)

> train\_data <- data[data$Year\_Q >= "1982\_Dec" & data$Year\_Q <= "2020\_Dec", ]

>

> # Filter data for testing (June 2020 to June 2023)

> test\_data <- data[data$Year\_Q >= "2020\_Jun" & data$Year\_Q <= "2023\_Jun", ]

>

>

> # Define the neural network model

> nn\_model <- neuralnet(Y ~ X1 + X2 + X3 + X4 + X5 + X6 + X7,

+ data = train\_data,

+ hidden = c(5, 3), # Specify the number of hidden layers and neurons

+ linear.output = TRUE) # For regression tasks

>

> # Make predictions on the test set

> predicted\_values\_test\_nn <- predict(nn\_model, newdata = test\_data)

>

> # Extract the target variable from the test set

> target\_variable\_test <- test\_data$Y

>

> # Calculate Mean Absolute Error (MAE) on the test set

> mae\_test\_nn <- mean(abs(predicted\_values\_test\_nn - target\_variable\_test))

>

> # Calculate Mean Squared Error (MSE) on the test set

> mse\_test\_nn <- mean((predicted\_values\_test\_nn - target\_variable\_test)^2)

>

> # Calculate Root Mean Squared Error (RMSE) on the test set

> rmse\_test\_nn <- sqrt(mse\_test\_nn)

>

> # Calculate R-squared (coefficient of determination) on the test set

> ssr\_nn <- sum((target\_variable\_test - predicted\_values\_test\_nn)^2)

> sst <- sum((target\_variable\_test - mean(target\_variable\_test))^2)

> rsq\_test\_nn <- 1 - (ssr\_nn / sst)

>

> cat("Mean Absolute Error (MAE) for Neural Network on the test set:", mae\_test\_nn, "\n")

Mean Absolute Error (MAE) for Neural Network on the test set: 2.094636

> cat("Mean Squared Error (MSE) for Neural Network on the test set:", mse\_test\_nn, "\n")

Mean Squared Error (MSE) for Neural Network on the test set: 5.700319

> cat("Root Mean Squared Error (RMSE) for Neural Network on the test set:", rmse\_test\_nn, "\n")

Root Mean Squared Error (RMSE) for Neural Network on the test set: 2.387534

> cat("R-squared (R^2) for Neural Network on the test set:", rsq\_test\_nn, "\n")

R-squared (R^2) for Neural Network on the test set: -2.828771

>

> ###########################################################################################

>

> #Hyperparaameter Tuning

>

> ###########################################################################################

> # Load the necessary libraries

> library(neuralnet)

>

> # Filter data for training (December 1982 to December 2020)

> train\_data <- data[data$Year\_Q >= "1982\_Dec" & data$Year\_Q <= "2020\_Dec", ]

>

> # Filter data for testing (June 2020 to June 2023)

> test\_data <- data[data$Year\_Q >= "2020\_Jun" & data$Year\_Q <= "2023\_Jun", ]

>

>

> # Create a function to build and train the neural network

> train\_neural\_network <- function(hidden\_layers, neurons) {

+ nn\_model <- neuralnet(Y ~ X1 + X2 + X3 + X4 + X5 + X6 + X7,

+ data = train\_data,

+ hidden = c(hidden\_layers, neurons), # Specify hidden layers and neurons

+ linear.output = TRUE) # For regression tasks

+

+ # Make predictions on the test set

+ predicted\_values\_test\_nn <- predict(nn\_model, newdata = test\_data)

+

+ # Calculate Mean Absolute Error (MAE) on the test set

+ mae\_test\_nn <- mean(abs(predicted\_values\_test\_nn - test\_data$Y))

+

+ # Calculate Mean Squared Error (MSE) on the test set

+ mse\_test\_nn <- mean((predicted\_values\_test\_nn - test\_data$Y)^2)

+

+ # Calculate Root Mean Squared Error (RMSE) on the test set

+ rmse\_test\_nn <- sqrt(mse\_test\_nn)

+

+ # Calculate R-squared (R^2) on the test set

+ ssr\_nn <- sum((test\_data$Y - predicted\_values\_test\_nn)^2)

+ sst <- sum((test\_data$Y - mean(test\_data$Y))^2)

+ rsq\_test\_nn <- 1 - (ssr\_nn / sst)

+

+ return(list(model = nn\_model, mae = mae\_test\_nn, mse = mse\_test\_nn, rmse = rmse\_test\_nn, rsq = rsq\_test\_nn))

+ }

>

> # Define hyperparameter grid to search

> hidden\_layer\_grid <- c(1, 2, 3) # Test different numbers of hidden layers

> neurons\_grid <- c(3, 5, 7) # Test different numbers of neurons

>

> best\_mae <- Inf # Initialize with a high value

> best\_model <- NULL

> best\_mse <- Inf

> best\_rmse <- Inf

> best\_rsq <- -Inf

>

> # Perform hyperparameter tuning

> for (hidden\_layers in hidden\_layer\_grid) {

+ for (neurons in neurons\_grid) {

+ # Train the model with the current hyperparameters

+ result <- train\_neural\_network(hidden\_layers, neurons)

+

+ # Check if the current model has a lower MAE

+ if (result$mae < best\_mae) {

+ best\_mae <- result$mae

+ best\_mse <- result$mse

+ best\_rmse <- result$rmse

+ best\_rsq <- result$rsq

+ best\_model <- result$model

+ best\_hidden\_layers <- hidden\_layers

+ best\_neurons <- neurons

+ }

+ }

+ }

>

> cat("Best Hyperparameters:\n")

Best Hyperparameters:

> cat("Hidden Layers:", best\_hidden\_layers, "\n")

Hidden Layers: 3

> cat("Neurons per Layer:", best\_neurons, "\n")

Neurons per Layer: 3

> cat("Best Mean Absolute Error (MAE):", best\_mae, "\n")

Best Mean Absolute Error (MAE): 2.094587

> cat("Best Mean Squared Error (MSE):", best\_mse, "\n")

Best Mean Squared Error (MSE): 5.700019

> cat("Best Root Mean Squared Error (RMSE):", best\_rmse, "\n")

Best Root Mean Squared Error (RMSE): 2.387471

> cat("Best R-squared (R^2):", best\_rsq, "\n")

Best R-squared (R^2): -2.82857

>

> # Now you can use the best\_model for predictions and evaluation