PREDICTING RAINFALL IN AUSTRALIA: EXPLORATORY ANALYSIS AND MACHINE LEARNING MODELLING

STATISTICS 4850G – ADVANCED DATA ANALYSIS

ANDY YUAN – 251159206

AIDAN DIGNAM – 251202245

DR. GUOWEN HUANG

APRIL 11, 2025

I. Introduction

Australia’s climate often sees extreme variability, making rainfall prediction challenging and important. In recent years, especially regarding impacts of climate change, Australia has seen recurring droughts, floods, and shifting weather patterns (Trewin, 2021). Accurate rainfall forecasting helps support agricultural planning, water resource management, and weather-related disaster response. This report presents a comprehensive analysis of rainfall data and employs advanced machine learning techniques to predict precipitation events across Australia. Recent research notes that linear regression, random forests, neural networks, XGBoost, and classification are considered robust methods for weather prediction (Zhang, Liu, Zhang, & Li, 2025). This report begins with an exploratory data analysis (EDA) to identify possible trends and correlations that will inform our model fitting processes. This will be followed by fitting logistic regression (including Elastic Net), random forest, and XGBoost models to the main effects of our available data. These models have been chosen because they are interpretable and have high predictive accuracy. The performance of these models will be evaluated through accuracy, precision, sensitivity, and the area under the Receiver Operating Characteristic (ROC) curve. Overall, the goal of this report is to improve the accuracy of rainfall predictions in Australia and highlight factors that impact rainfall patterns.

II. Dataset Exploration

The dataset used in this project, titled *“Rain in Australia,”* was sourced from Kaggle (Commonwealth of Australia, Bureau of Meteorology, 2021), a platform providing public datasets for data analysis. The data comprises daily weather observations collected from 49 unique weather stations across Australia, spanning from October 30th, 2007, to June 25th, 2017, and is copyrighted by the Commonwealth of Australia (2010), Bureau of Meteorology. The complete dataset includes 145,460 observations, averaging about 3,436 observations per weather station. There are 23 variables in total, each described in detail in Table 1. The primary response variable, *“RainTomorrow,”* indicates whether it will rain the following day and is coded categorically as *“Yes”* or *“No.”*

For clarity, all observations containing N/A values were omitted, which reduces the dataset's predictive power by leaving only 56,420 complete observations. This is sufficient for robust predictions and conclusions. A limitation of this dataset is the restricted timeframe over which data was collected, potentially limiting generalizability to periods experiencing significant climatic changes, such as severe global warming or an ice age. However, extrapolating to such scenarios would exceed the scope of this model. Overall, the dataset remains valuable for identifying key factors influencing rainfall in Australia.

Table 1: Variable Descriptions for “Rain in Australia” Dataset

Name	Type	Description
Date	Date	The date of an observation.

Location	Categorical	The name of the location of the weather station.
MinTemp	Continuous	The minimum temperature on a day (°C).
MaxTemp	Continuous	The maximum temperature on a day (°C).
Rainfall	Continuous	The amount of rainfall recorded for the day (mm).
Evaporation	Continuous	The Class A pan evaporation in the 24 hours before 9am (mm).
Sunshine	Continuous	The number of hours of bright sunshine in the day.
WindGustDir	Categorical	The direction of the strongest wind gust in the prior 24 hours.
WindGustSpeed	Continuous	The speed of the strongest wind gust in the prior 24 hours (km/h).
WindDir9am	Categorical	The direction of the wind at 9am.
WindDir3pm	Categorical	The direction of the wind at 3pm.
WindSpeed9am	Continuous	The average wind speed 10 minutes prior to 9am (km/hr).
WindSpeed3pm	Continuous	The average wind speed 10 minutes prior to 3pm (km/hr).
Humidity9am	Continuous	The humidity at 9am (in percent).
Humidity3pm	Continuous	The humidity at 3pm (in percent).
Pressure9am	Continuous	The atmospheric pressure reduced to mean sea level at 9am (hpa).
Pressure3pm	Continuous	The atmospheric pressure reduced to mean sea level at 3pm (hpa).
Cloud9am	Integer	The fraction of sky obscured by clouds at 9am (in Oktas). 0 indicates completely clear sky, 8 indicates completely overcast.
Cloud3pm	Integer	The fraction of sky obscured by clouds at 9am in Oktas.
Temp9am	Continuous	The temperature at 9am (°C).
Temp3pm	Continuous	The temperature at 3pm (°C).
RainToday	Categorical	1 if precipitation in the 24 hours to 9am exceeds 1mm, otherwise 0.
RainTomorrow	Categorical	1 if rain exceeded 1mm the next day, otherwise 0.

III. Methodology

a. Exploratory Data Analysis

The exploratory data analysis examines the rainfall dataset to identify distributions, patterns, and relationships among variables. Firstly, continuous variables were analyzed using density plots and boxplots. Density plots for variables such as minimum and maximum temperature, humidity, wind speed, and atmospheric pressure generally symmetric-shaped distributions. Looking closely, variables including evaporation, sunshine, wind gust speed, and rainfall exhibited skewness, reflecting rare weather events (see Figure 1). Notably, evaporation and rainfall have substantial right-skewness, indicating occasional high outliers in the data (see Figure 3). Figures 2 and 4 provide density plots and boxplots split between "*RainTomorrow*" as "*Yes*" or "*No*". Days with rain generally had higher humidity levels, lower sunshine hours, slightly lower pressure, and slightly cooler temperatures the day before compared to days without rain. In terms of outliers, variables including wind gust speed, evaporation, and rainfall suggests possible extreme weather conditions in Australia.

The categorical analysis displayed in Figures 5, 6, and 7 showed variability in rain occurrences across different weather station locations, highlighting geographical influences on rainfall patterns. Additionally, wind direction appeared to influence rain occurrences, with almost all directions correlating more frequently with rainfall events. The most visually significant categorical predictor is "*RainToday*", where it is highly likely that it will not rain tomorrow if it did not rain today.

Analysis of the response variable, "*RainTomorrow*," revealed some class imbalance. Only 22% of observations indicate rainfall on the following day (Figure 8). We will address this imbalance in the modelling stages of the report to combat potential bias in the predictive models.

b. Logistic Regression

Logistic regression is a binary classification model, which can be directly applied to predicting whether it will rain tomorrow based on weather conditions today (yes, or no). The logistic regression model generates probabilities that the response variable (*RainTomorrow*) equals "Yes" given a set of predictor variables. Logistic regression models use the logit $\eta_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_q x_{iq}$, which represents the log-odds of an outcome. To return a probability, we can express the logit under $P(Y_i = 1) = \frac{e^{\eta_i}}{1+e^{\eta_i}}$, where the β values are the coefficients from the logistic regression model obtained using maximum likelihood estimation, and the x values are the data used to model the relationship. Using the predictors in Table 1, a model will be fitted to predict the response variable, *RainTomorrow*. To interpret the coefficients, odds or probabilities can be used. For odds, any coefficient e^{β_i} represents the change in odds of an observation being in class one for a single unit increase of x_i, holding all other covariates fixed.

To convert predicted probabilities into binary outcomes ("*Yes*" or "*No*"), a classification threshold or cutoff is applied. Observations with predicted probabilities above the cutoff are classified as "*Yes*." Rather than using the default value of 0.5, we select an optimal cutoff. This is an effective method for improving performance given the class imbalance in the dataset.

c. Elastic Net Regression

Elastic Net regression combines elements of both Lasso and Ridge regression by including both penalties, respectively L_1 (absolute magnitude of coefficients) and L_2 (squared magnitude of coefficients), in the logistic regression framework. It aims to balance coefficient shrinkage and variable selection. The Elastic Net method is governed by two hyperparameters: λ , controlling the overall penalty strength and sparsity, and α, determining the balance between Ridge ($\alpha = 0$) and Lasso ($\alpha = 1$). This dual-penalty approach leverages the strengths of both Ridge and Lasso, making it effective in managing multicollinearity, selecting relevant variables, and stabilizing coefficient estimates. If the hyper parameter α is 1 (Lasso), then some predictors may be excluded from the final model. It is very rare for Ridge to possess this quality.

$$\hat{\beta}_{EN} = argmin_{\beta}\left\{\sum_{i=1}^{N}\left(y_{\mathrm{i}} - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j\right)^2 + \lambda\left(\alpha\sum_{j=1}^{p}\left|\beta_j\right| + (1-\alpha)\sum_{j=1}^{p}\beta_j^2\right)\right\}$$

d. Random Forests

Random Forest is a statistical method that performs classification using decision trees when training the model. In the context of weather predictions, the random forest process builds many trees with minor differences. The result is the class that majority of the trees arrive at. This classification method allows for randomness within the model; correspondingly, this avoids issues of overfitting, which typical linear models are prone to. The result from Random Forest is usually majority or average selection among the data.

Random Forest relies on bootstrap sampling for randomization. In each iteration, a random sample of data is used to create the decision tree. Each iteration uses a completely random subset of features to base how it splits the nodes of the tree. The benefit of Random Forest is that each individual iteration (individual trees) is unique from any other tree. This randomness allows to a more robust model.

Random Forest models indicate how features contribute to predicting an outcome, allowing for simple interpretation. In the case of rainfall prediction, we can test all predictors against the response variable to understand the importance of each predictor on whether it will rain tomorrow or not.

Random Forests can be mathematically represented as the following, where N is the total number of decision trees in the forest, $h_i(X)$ represents the prediction made by the ith tree, and the indicator function $I(h_i(\mathrm{X}) = 1)$ returns 1 if the tree predicts rain and 0 otherwise:

$$P(Y = 1|X) = \frac{1}{N}\sum_{i=1}^{N} I(h_i(\mathrm{X}) = 1)$$

e. XGBoost

Extreme Gradient Boosting (XGBoost) is classification method that can be applied to predicting whether it will rain tomorrow or not. Similar to Random Forest, XGBoost builds decision trees; however, the trees are sequential. In each new interation, the new tree corrects errors made in the previous trees. Updating the trees to remove previous errors results in strong accuracy, which will help in predicting weather patterns.

XGBoost is effective at discovering subtle relationships between variables and predictors. In the context of this report, relationships between predictors such as humidity, temperature, and wind speed help improve the decision trees. The updated trees are computed by using gradients and second-order derivatives. In the training process, the user can choose the number of boosting rounds, maximum tree depth, and the learning rate. Optimal selection of these parameters are tuned through cross validation, where the final numbers optimize the predictive performance regarding the response variable.

Mathematically, we can represent XGBoost as a function of the sum of predictions from individual trees:

$$\widehat{y_i} = \sigma\left(\sum_{k=1}^{K} f_k(x_i)\right)$$

where $\widehat{y_i}$ is the final predicted probability for observation i, K is the total number of boosting rounds (or trees), and $f_k(x_i)$ is the prediction from the kth tree. The logistic function σ is applied to convert the summed output into a probability between 0 and 1, suitable for binary classification tasks like predicting rainfall.

f. Backward Model Selection

Backward model selection is a method that simplifies regression models by sequentially removing predictors from a model. Once the predictor is removed, the contribution of the predictor is evaluated. If there is minimal impact, the predictor is removed from the model. The goal of backward model selection is finding a balance between predictive power and interpretability. This is useful for exploring weather-related prediction processes, where unnecessary predictors can create confusion in identifying relationships.

In this analysis, Akaike Information Criterion (AIC) is used on a full main effects logistic regression model. AIC quantifies the trade-off between complexity and goodness-of-fit by penalizing overly complex models, outlined by the $2k$ term. The backward selection process iteratively removes individual predictors and chooses whether to exclude them from the final model with the final intention of minimizing the AIC value, which is comprised on L, the likelihood of the model, and k number of predictors:

$$AIC = -2\log(L) + 2k$$

The final model retains only predictors that are statistically significant in contributing to forecasting rainfall the next day. Given the regularization terms included in the elastic net regression, logistic regression models are not directly compared using a Hosmer-Lemeshow test or Cox calibration. The inputted models violate the assumptions of these tests, which can make the comparisons unreliable. Overall, the final logistic regression model is based on predictive accuracy, not direct hypothesis testing.

g. Performance Evaluation

The performance of each model is assessed by comparing its ability to predict rainfall status based on new data. To quantify this performance, analysis of the Receiver Operating Characteristics curve (ROC), confusion matrices, sensitivity, specificity, precision, negative predictive value, and overall accuracy will be used.

For each model, the predicted probabilities of tomorrows rain status are produced. Then, ROC curves are plotted, exploring the relationship between the true positive rate (sensitivity) false positive rate (1 – specificity) at different threshold levels. To summarize the performance, the area under the ROC curve (AUC) measures the model's ability to distinguish rain status the proceeding day. An AUC close to 1 is considered to have string predictive accuracy.

Based on the ROC curve analysis using Youden's statistic, the optimal classification threshold is selected, maximizing the difference between the true positive rate and false positive rate. The optimal threshold level is used to convert predicted probabilities into a binary classification, "*Yes*" and *"No*" in for the prediction of whether it will rain tomorrow. This approach aims to lessen the impact of the class imbalance in the data.

Based on the classifications, a confusion matrix summarizes the classifications of the model. The matrix stores counts for true positives, false positives, true negatives, and false negatives. These values can be defined as the following:

- Sensitivity (true positive rate): the proportion of actual rainy days correctly predicted.
- Specificity (true negative rate): the proportion of dry days correctly predicted.
- Precision (positive predictive value): the proportion of predicted rainy days that were rainy.
- Negative predictive value (NPV): the proportion of predicted dry days that were dry.
- Accuracy: the overall proportion of correct predictions across all observations.

These values are calculated for reach model fitted in previous sections. ROC and metrics from the confusion matrix allow for standardized comparisons across models, even if there are structural differences. For example, these metrics can be applied to elastic net regression and ensemble construction in random forests and boosting, all with the same interpretation. This is important to note because traditional calibration tests like the Hosmer-Lemeshow test cannot be used on models with additional components/assumptions (i.e. elastic net regression previously mentioned).

To examine interpretability, feature importance metrics such as permutation-based metrics applied to Random Forest models and gain-based metrics applied to XGBoost help us understand which variables contribute the most to rainfall predictions. The variables with the highest mean decrease accuracy (how much predictive accuracy decreases when permuted across out-of-bag samples) and mean decrease Gini (total reduction in Gini impurity caused by splits on that variable, averaged across all trees) for Random Forest and the highest importance in XGBoost are considered the most important variables, meaning that if they are not present, then the predictive power of the model decreases significantly.

IV. Results

a. Exploratory Data Analysis (EDA)

The EDA phase provided insights to the distributions and behaviours of the predictors used to explain tomorrow's rain prediction in Australia. With a series of density plots, boxplots, and bar plots, we can make visual assumptions about the predictors and their relationship with the response variable, RainTomorrow.

For the continuous variables, including all variations of Evaporation, Sunshine, Rainfall, and Wind Gust Speed, the distributions appear slightly right skewed. These variables can include extreme weather events, as seen in density plots (Figure 1) and boxplots presenting many outliers (Figures 3 and 4). For example, the summary statistics report that Rainfall has a maximum of 206.2 mm and a median of 0 mm, confirming that most days have no precipitation. Similarly, Wind Gust Speed ranges up to 124 km/h and Evaporation's high extreme is 81.2 mm, suggesting some extreme weather observations.

Variables such as MinTemp, MaxTemp, Humidity9am/3pm, and Pressure9am/3pm exhibited more symmetric distributions. Summary statistics show MinTemp ranges from -6.7°C to 31.4°C, while MaxTemp extends up to 48.1°C, indicating Australia's climate variability due to seasonality and/or extreme weather. When the continuous predictors are separated by the RainTomorrow outcome (Figure 2), we observe that rainy days are associated with higher humidity levels at both 9am and 3pm, less sunshine hours, slightly cooler temperatures, and lower pressure levels.

For categorical variables, bar plots demonstrated variability in rain distribution across locations, wind directions, and RainToday (Figures 5, 6, and 7). Rain patterns seem to vary with wind direction, and if it rained today, it's substantially more likely it will rain again tomorrow.

It is important to note that the imbalance in RainTomorrow is large, with only 12,427 "Yes" responses out of 56,420 total observations (22%) (see Figure 8). This is expected given Australia's dry climate; however, for modelling purposes, this imbalance highlights the importance of using evaluation metrics such as AUC, precision, and threshold for tuning in our classification models. Figure 9 shows the proportion of rainy days across time, indicating no consistent seasonal or temporal pattern in the likelihood of rainfall the next day.

b. Main Data Analysis

To evaluate the predictive power of Logistic Regression (AIC-based), Elastic Net Logistic Regression, Random Forest, and XGBoost for rainfall forecasting, each model was trained on 70% of the cleaned dataset and tested on the remaining 30%, with performance evaluation following. The key metrics used for comparison are AUC (area under the ROC curve), accuracy, sensitivity, specificity, and precision, all summarized in Table 2. We will also interpret the threshold/cutoff of each model. Precision was lower than what we want, which is a by-product of not adequately accounting for class imbalance. A better threshold would be higher to eliminate the number of false positives at the trade-off for a lower performing sensitivity.

Table 2: Model Performance Comparison

Model	Cutoff	AUC	Accuracy	Sensitivity	Specificity	Precision

Logistic Regression (AIC)	0.1848	0.8845	0.7852	0.8232	0.7744	0.5096
Elastic Net	0.1889	0.8847	0.7871	0.8168	0.7787	0.5124
Random Forest	0.2845	0.8974	0.8101	0.8094	0.8103	0.5485
XGBoost	0.2108	0.9012	0.8138	0.8198	0.8121	0.5540

XGBoost achieved the highest AUC (0.9012) and overall accuracy (81.38%), outperforming the other models on almost all metrics (see Figure 15). Random Forest and XGBoost also demonstrated higher precision, suggesting better reliability when predicting rainy days. Variable importance rankings are displayed in Figures 13 and 14 for Random Forest (Mean Decrease Accuracy and ranking of variables). It is likely that tree-based learning is best for weather prediction based on the results of these models.

Comparing the models Logistic Regression (AIC) and Elastic Net offer simple interpretation but are slightly less effective at capturing nonlinear interactions in the dataset. Random Forest provided balanced sensitivity and specificity and performed better than the logistic-based models in handling high dimension interactions. XGBoost demonstrated the strongest performance across all metrics, demonstrating its strength in modeling complex relationships within the weather data. Figure 10 presents the ROC curve for the AIC-selected logistic regression model, with an AUC of 0.8845 and an optimal threshold of 0.1848. The ROC curve for the Elastic Net model is shown in Figure 11, which produced an AUC of 0.8847 and a threshold of 0.1889.

These findings suggest that logistic regression remains valuable for interpretation and hypothesis generation, but tree-based ensemble models deliver superior predictive performance for rainfall classification tasks.

V. Conclusion/Discussion

This report applied several machine learning techniques to forecast rainfall in Australia, using ten years of historical weather data from the Bureau of Meteorology. Among Logistic Regression (AIC-selected), Elastic Net, Random Forest, and XGBoost, XGBoost performed the best, achieving the highest AUC (0.9012), accuracy (81.38%), sensitivity (81.98%), and precision (55.40%). These results suggest that XGBoost is an effective model choice for assessing how environmental variables influence rainfall (see Figure 15). Random Forest also demonstrated strong performance, with an AUC of 0.8974, accuracy of 81.01%, sensitivity of 80.94%, and precision of 54.85%. If a goal of analysis is to have more interpretable feature rankings, Random Forest would be a strong candidate. The logistic regression models, specifically the AIC model, achieved an AUC of 0.8845, accuracy of 78.52%, and sensitivity of 82.32%. This model outperformed the tree-based models in sensitivity but resulted in lower specificity (77.44%) and precision (50.96%). This is likely due to the lower classification threshold (0.1848). The Elastic Net model performed very similar to the AIC model. Overall, the logistic models are useful for interpreting individual predictor effects, but they are less effective at predicting the next day's rain status.

Regarding feature importance, Random Forest and XGBoost emphasized similar variables. In Random Forest, Humidity3pm had the highest Mean Decrease in Accuracy (134.5), followed by Sunshine (120.9), WindGustSpeed (101.5), and Pressure3pm (99.5). Similarly, in the XGBoost model, Humidity3pm accounted for 33.1% of total gain in predictive accuracy, followed by Sunshine (18.1%), Pressure3pm (9.5%), and WindGustSpeed (7.0%). These predictors make logical sense because high humidity and low sunshine levels often precede rain, an observation most humans can infer based on lived experience. As well, wind gusts often lead to some form of atmospheric changes. Figure 16 illustrates XGBoost's variable importance plot, with Humidity3pm and Sunshine emerging as the most influential predictors based on gain.

Despite these results, the analysis has limitations. Roughly two-thirds of the original dataset (145,460 rows) were excluded due to missing values, leaving only 56,420 complete cases for training and testing. Omitting this amount of data can reduce generalizability. Additionally, the data spans from 2007 to 2017, meaning recent climate shifts (climate change) and extreme weather patterns would not be captured in the training set. Accordingly, model extrapolation to future climate conditions might be less reliable than the model suggests. Additionally, since class imbalance was not fully addressed, the models may be more optimistic than the most correct model in determining whether the next day will have rain. This may be from overlap in the characteristics of the days before it will rain or not rain or could be from solely class imbalance. We see the effect of this in the low precision metrics across all models. One more limitation of the analysis is the lack of interactions between the covariates. We were limited in computational power to examine the effects of potential interactions between variables. Using basic meteorologic knowledge, potential pairs of covariates like wind speed and direction, humidity and evaporation, or pressure and rain today may be better predictors together rather than apart.

In conclusion, this analysis demonstrates that tree-based models like Random Forest and XGBoost are effective at predicting rainfall patterns in Australia with strong accuracy and reliability. While XGBoost offers the highest overall performance, Random Forest provides a well-balanced alternative with better interpretability. The best model should be selected based on the goals of the model. Logistic models remain valuable for understanding the role of individual predictors and how they affect rainfall the next day. Overall, these statistical methods provide a robust framework for building predictive systems that can support real-world decision-making in meteorology, agriculture, and disaster preparedness.

VI. References

Bibliography

Commonwealth of Australia, Bureau of Meteorology. (2021). *Rain in Australia* . Retrieved from Kaggle: https://www.kaggle.com/datasets/jsphyg/weather-dataset-rattle-package/data

Trewin, B. (2021, September 17). *What's happening to Australia's rainfall?* . Retrieved from Australian Academy of Science : https://www.science.org.au/curious/policy-features/whats-happening-australias-rainfall

Zhang, H., Liu, Y., Zhang, C., & Li, N. (2025, January 14). *Machine Learning Methods for Weather Forecasting: A Survey* . Retrieved from Multidisciplinary Digital Publishing Institute: https://www.mdpi.com/2073-4433/16/1/82

VII. Appendix

Table of Contents

Data Preprocessing

Import Required Libraries

```
# Libraries to import
 suppressWarnings(suppressMessages({
   library(factoextra), library(dplyr), library(doParallel),
library(ggplot2), library(reshape2),library(tidyr), library(stringr),
library(faraway), library(glmnet),library(foreach),library(doRNG),
library(data.table), library(pROC), library(randomForest), library(xgboost),
   library(caret)}))
 # Use all available resources of computer
 number_cores <- detectCores()-1; registerDoParallel(cores = number_cores)
```

Load and Clean Data (Omit NA, Encode Categorical Variables)

```
# Load data
 csvdata <- read.csv("weatherAUS.csv")
# Omit NA values
 csvdata <- na.omit(csvdata)
 # Converting to factor the categorical variables
 csvdata$Location <- as.factor(csvdata$Location)
 csvdata$WindGustDir <- as.factor(csvdata$WindGustDir)
 csvdata$WindDir9am <- as.factor(csvdata$WindDir9am)
 csvdata$WindDir3pm <- as.factor(csvdata$WindDir3pm)
 csvdata$RainToday <- as.factor(csvdata$RainToday)
 csvdata$RainTomorrow <- as.factor(csvdata$RainTomorrow)
```

Exploratory Analysis

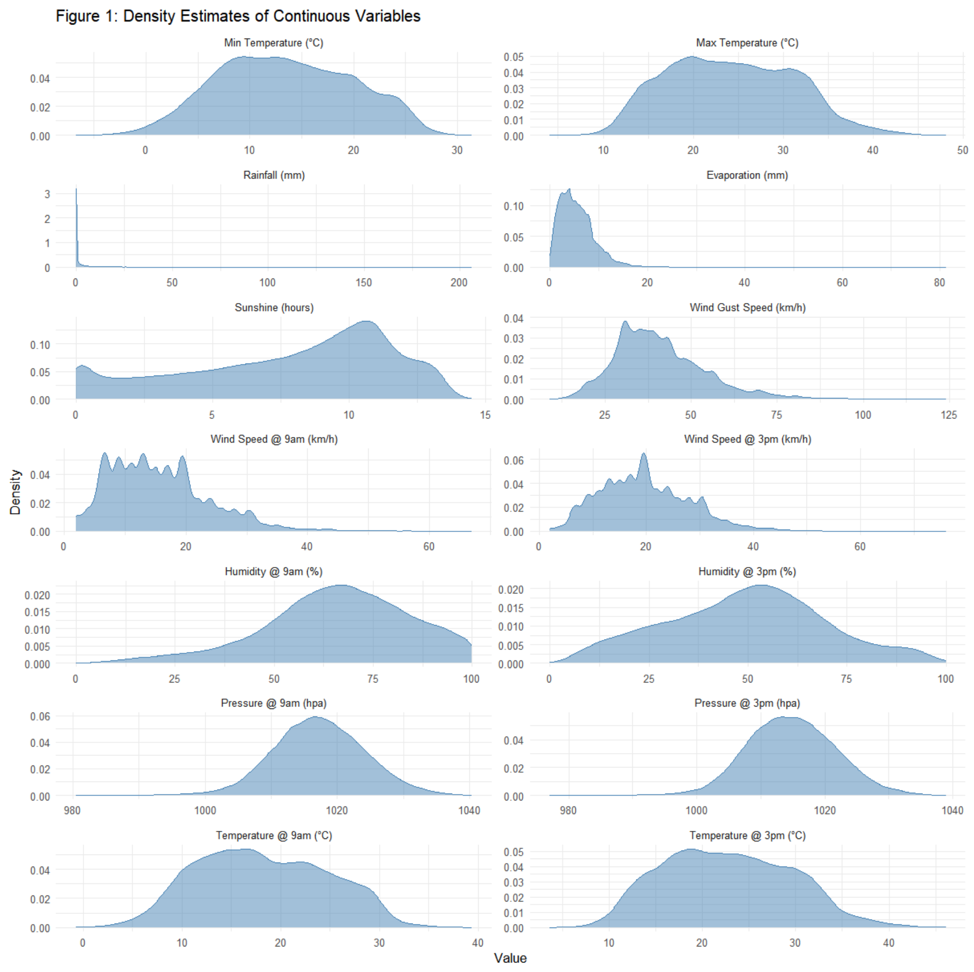
```
summary(csvdata) # Summary of data before deeper analysis
##      Date                       Location        MinTemp         MaxTemp
##  Length:56420       Darwin          : 3062   Min.   :-6.70   Min.   : 4.10
##  Class :character   Perth           : 3025   1st Qu.: 8.60   1st Qu.:18.70
##  Mode  :character   Brisbane        : 2953   Median :13.20   Median :23.90
##                     MelbourneAirport: 2929   Mean   :13.46   Mean   :24.22
##                     PerthAirport    : 2913   3rd Qu.:18.40   3rd Qu.:29.70
##                     SydneyAirport   : 2870   Max.   :31.40   Max.   :48.10
##                     (Other)         :38668
##     Rainfall        Evaporation        Sunshine       WindGustDir
##  Min.   :  0.00   Min.   : 0.000   Min.   : 0.000   E      : 4516
##  1st Qu.:  0.00   1st Qu.: 2.800   1st Qu.: 5.000   N      : 4210
##  Median :  0.00   Median : 5.000   Median : 8.600   W      : 4161
##  Mean   :  2.13   Mean   : 5.503   Mean   : 7.736   SW     : 4052
##  3rd Qu.:  0.60   3rd Qu.: 7.400   3rd Qu.:10.700   ENE    : 4028
##  Max.   :206.20   Max.   :81.200   Max.   :14.500   SE     : 3930
##                                                     (Other):31523
##  WindGustSpeed      WindDir9am      WindDir3pm     WindSpeed9am
##  Min.   :  9.00   N      : 4967   SE     : 4153   Min.   : 2.00
##  1st Qu.: 31.00   E      : 4456   S      : 4109   1st Qu.: 9.00
##  Median : 39.00   ENE    : 3932   SW     : 4012   Median :15.00
##  Mean   : 40.88   SSE    : 3893   ENE    : 3946   Mean   :15.67
##  3rd Qu.: 48.00   SE     : 3880   W      : 3922   3rd Qu.:20.00
##  Max.   :124.00   W      : 3707   WSW    : 3856   Max.   :67.00
##                   (Other):31585   (Other):32422
##   WindSpeed3pm    Humidity9am      Humidity3pm      Pressure9am
##  Min.   : 2.00   Min.   :  0.00   Min.   :  0.0   Min.   : 980.5
##  1st Qu.:13.00   1st Qu.: 55.00   1st Qu.: 35.0   1st Qu.:1012.7
##  Median :19.00   Median : 67.00   Median : 50.0   Median :1017.2
##  Mean   :19.79   Mean   : 65.87   Mean   : 49.6   Mean   :1017.2
##  3rd Qu.:26.00   3rd Qu.: 79.00   3rd Qu.: 63.0   3rd Qu.:1021.8
##  Max.   :76.00   Max.   :100.00   Max.   :100.0   Max.   :1040.4
##
##   Pressure3pm       Cloud9am        Cloud3pm        Temp9am
##  Min.   : 977.1   Min.   :0.000   Min.   :0.000   Min.   :-0.7
##  1st Qu.:1010.1   1st Qu.:1.000   1st Qu.:2.000   1st Qu.:13.1
##  Median :1014.7   Median :5.000   Median :5.000   Median :17.8
##  Mean   :1014.8   Mean   :4.242   Mean   :4.327   Mean   :18.2
##  3rd Qu.:1019.4   3rd Qu.:7.000   3rd Qu.:7.000   3rd Qu.:23.3
##  Max.   :1038.9   Max.   :8.000   Max.   :9.000   Max.   :39.4
##
##     Temp3pm      RainToday   RainTomorrow
 ##  Min.   : 3.70   No :43958   No :43993
##  1st Qu.:17.40   Yes:12462   Yes:12427
##  Median :22.40
##  Mean   :22.71
##  3rd Qu.:27.90
##  Max.   :46.10
```

Continuous Variables

```r
# Define continuous variables
 continuous_vars <- c("MinTemp", "MaxTemp", "Rainfall", "Evaporation",
                      "Sunshine", "WindGustSpeed", "WindSpeed9am",
                      "WindSpeed3pm", "Humidity9am", "Humidity3pm",
                      "Pressure9am", "Pressure3pm", "Temp9am", "Temp3pm")
```

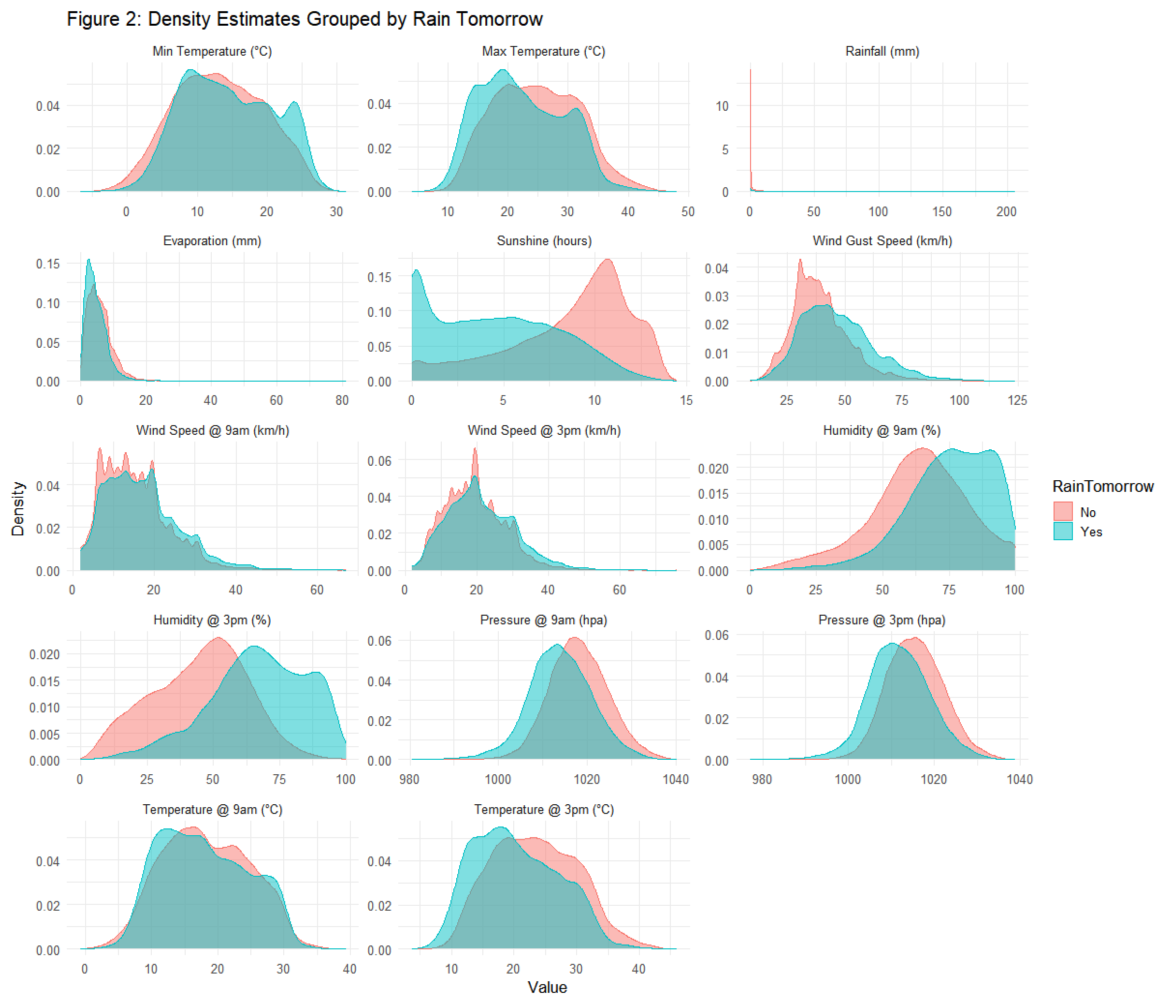
We can use density plots for each continuous variable to visualize their distributions, providing information about the spread, skewness, and modality in the data.

```r
# Reshape dataframe for continuous variables
 df_long <- reshape2::melt(csvdata, measure.vars = continuous_vars)
 # Plot density plots
 ggplot(df_long, aes(x = value)) +
   geom_density(color = "steelblue", fill = "steelblue", alpha = 0.5) +
   facet_wrap(~ variable,scales = "free",ncol = 2, labeller =
labeller(variable = c(MinTemp = "Min Temperature (°C)", MaxTemp = "Max
Temperature (°C)", Rainfall = "Rainfall (mm)", Evaporation = "Evaporation
(mm)", Sunshine = "Sunshine (hours)", WindGustSpeed = "Wind Gust Speed
(km/h)", WindSpeed9am = "Wind Speed @ 9am (km/h)", WindSpeed3pm = "Wind Speed
@ 3pm (km/h)", Humidity9am   = "Humidity @ 9am (%)", Humidity3pm = "Humidity
@ 3pm (%)", Pressure9am = "Pressure @ 9am (hpa)", Pressure3pm = "Pressure @
3pm (hpa)", Temp9am = "Temperature @ 9am (°C)", Temp3pm = "Temperature @ 3pm
(°C)"))) + labs(title = "Figure 1: Density Estimates of Continuous
Variables", x = "Value", y = "Density") + theme(axis.text.x =
element_text(size = 12,angle = 45, hjust = 1), panel.spacing = unit(1,
"lines"), strip.text = element_text(size = 12), plot.title =
element_text(hjust = 0.5, face="bold")) + theme_minimal(base_size = 14)
```



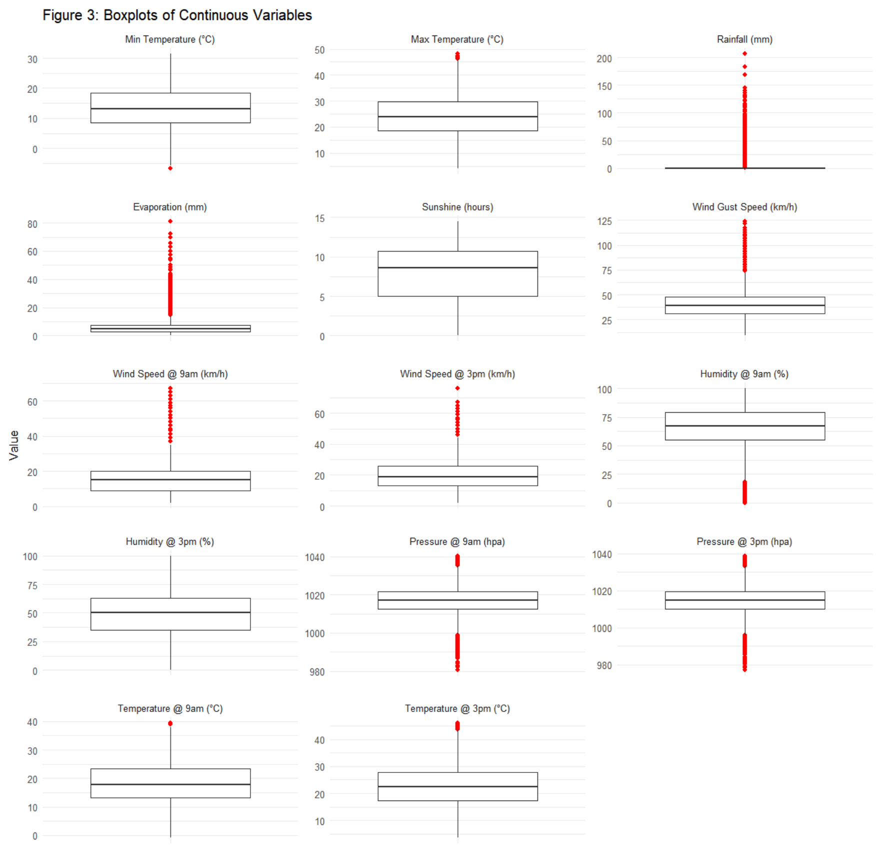
Similar to the previous plots, can also separate the data based on RainTomorrow.

```
ggplot(df_long, aes(x = value, fill = RainTomorrow, color = RainTomorrow)) +
   geom_density(alpha = 0.5) + facet_wrap(~ variable, scales = "free", ncol =
3, labeller = labeller(variable = c(MinTemp = "Min Temperature (°C)", MaxTemp
= "Max Temperature (°C)", Rainfall = "Rainfall (mm)", Evaporation =
"Evaporation (mm)", Sunshine = "Sunshine (hours)", WindGustSpeed = "Wind Gust
Speed (km/h)", WindSpeed9am = "Wind Speed @ 9am (km/h)", WindSpeed3pm = "Wind
Speed @ 3pm (km/h)", Humidity9am = "Humidity @ 9am (%)", Humidity3pm =
"Humidity @ 3pm (%)", Pressure9am = "Pressure @ 9am (hpa)", Pressure3pm =
"Pressure @ 3pm (hpa)", Temp9am = "Temperature @ 9am (°C)", Temp3pm =
"Temperature @ 3pm (°C)"))) + labs(title = "Figure 2: Density Estimates
Grouped by Rain Tomorrow", x = "Value",y = "Density") + theme(axis.text.x  =
element_text(angle = 45, hjust = 1), axis.text.y  = element_text(angle = 0,
size = 10), panel.spacing = unit(1.5, "lines"), strip.text =
element_text(size = 11), plot.title = element_text(hjust = 0.5, size = 14,
face = "bold"), legend.position = "bottom") + theme_minimal(base_size = 14)
```



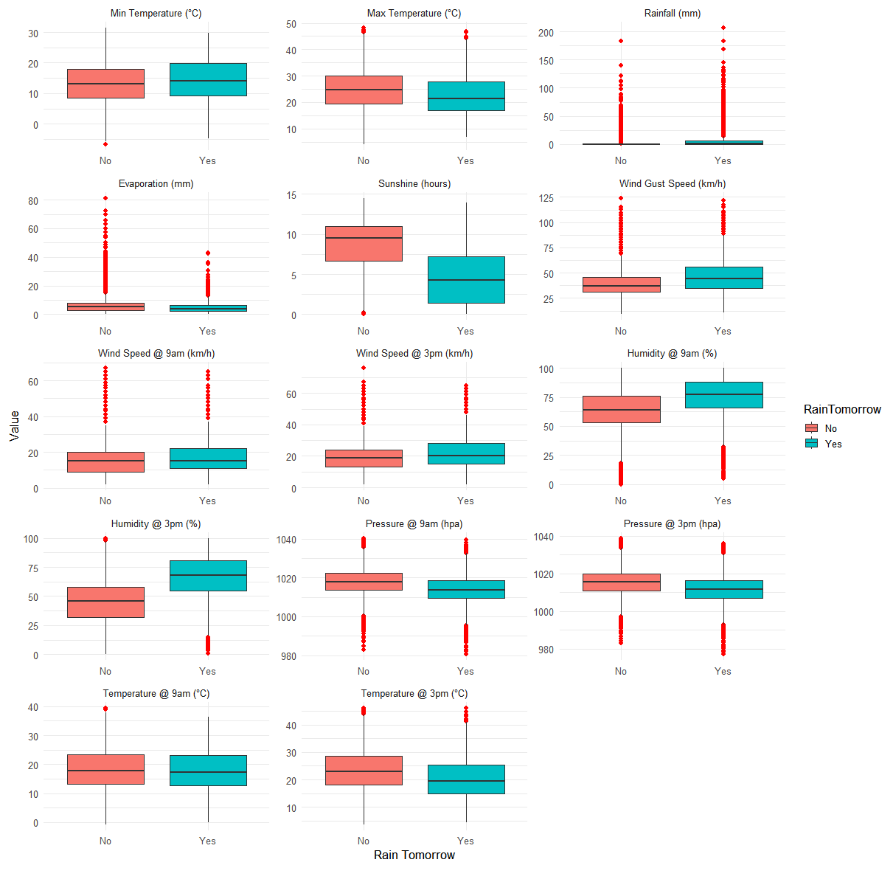
To identify outliers, we can use boxplots across each variable.

```
# box plots for all continuous variables
 ggplot(df_long, aes(x = "", y = value)) + geom_boxplot(outlier.colour =
"red", outlier.shape = 16, outlier.size = 2) + facet_wrap(~ variable, scales
= "free", ncol = 3,  # Try 2 or 3 columns to make each panel larger
     labeller = labeller(variable = c(MinTemp = "Min Temperature (°C)",
MaxTemp = "Max Temperature (°C)", Rainfall = "Rainfall (mm)", Evaporation =
"Evaporation (mm)", Sunshine = "Sunshine (hours)", WindGustSpeed = "Wind Gust
Speed (km/h)", WindSpeed9am = "Wind Speed @ 9am (km/h)", WindSpeed3pm = "Wind
Speed @ 3pm (km/h)", Humidity9am = "Humidity @ 9am (%)", Humidity3pm =
"Humidity @ 3pm (%)", Pressure9am = "Pressure @ 9am (hpa)", Pressure3pm =
"Pressure @ 3pm (hpa)", Temp9am = "Temperature @ 9am (°C)", Temp3pm =
"Temperature @ 3pm (°C)"))) + labs(title = "Figure 3: Boxplots of Continuous
Variables", x = "", y = "Value") + theme(plot.title = element_text(hjust =
0.5, size = 16, face = "bold"),axis.text.x = element_blank(),      # Hides
the blank x-axis label axis.ticks.x = element_blank(), strip.text =
element_text(size = 12) # Adjust facet label size as desired
   ) + theme_minimal(base_size = 14)
```



```
# Plot all boxplots, grouped by RainTomorrow ggplot(df_long, aes(x =
RainTomorrow, y = value, fill = RainTomorrow)) + geom_boxplot(outlier.colour
= "red", outlier.shape = 16, outlier.size = 2) + facet_wrap(~ variable,
scales = "free", ncol = 3,  # fewer columns -> bigger panels labeller =
labeller(variable = c(MinTemp = "Min Temperature (°C)", MaxTemp = "Max
Temperature (°C)", Rainfall = "Rainfall (mm)", Evaporation = "Evaporation
(mm)", Sunshine = "Sunshine (hours)", WindGustSpeed = "Wind Gust Speed
(km/h)", WindSpeed9am  = "Wind Speed @ 9am (km/h)", WindSpeed3pm  = "Wind
Speed @ 3pm (km/h)",Humidity9am = "Humidity @ 9am (%)",Humidity3pm =
"Humidity @ 3pm (%)", Pressure9am   = "Pressure @ 9am (hpa)" ,Pressure3pm   =
"Pressure @ 3pm (hpa)", Temp9am = "Temperature @ 9am (°C)", Temp3pm       =
"Temperature @ 3pm (°C)"))) + labs(itle = "Figure 4: Boxplots of Continuous
Variables by Rain Tomorrow",x = "Rain Tomorrow",y = "Value") +
theme(plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
axis.text.x = element_text(angle = 45, hjust = 1), # rotate category labels
strip.text = element_text(size = 12)) + theme_minimal(base_size = 14)
```

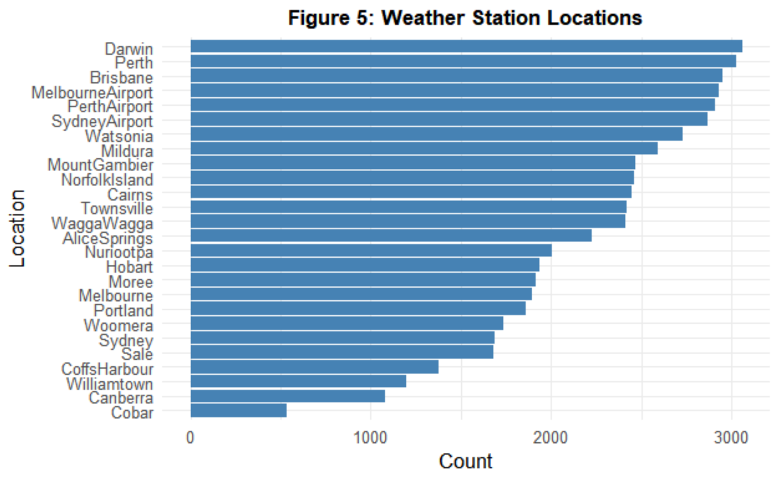
Figure 4: Boxplots of Continuous Variables by Rain Tomorrow



Categorical Variables

For categorical variables, bar plots can demonstrate the count of each category within each variable. Below, we see the number of observations per location:

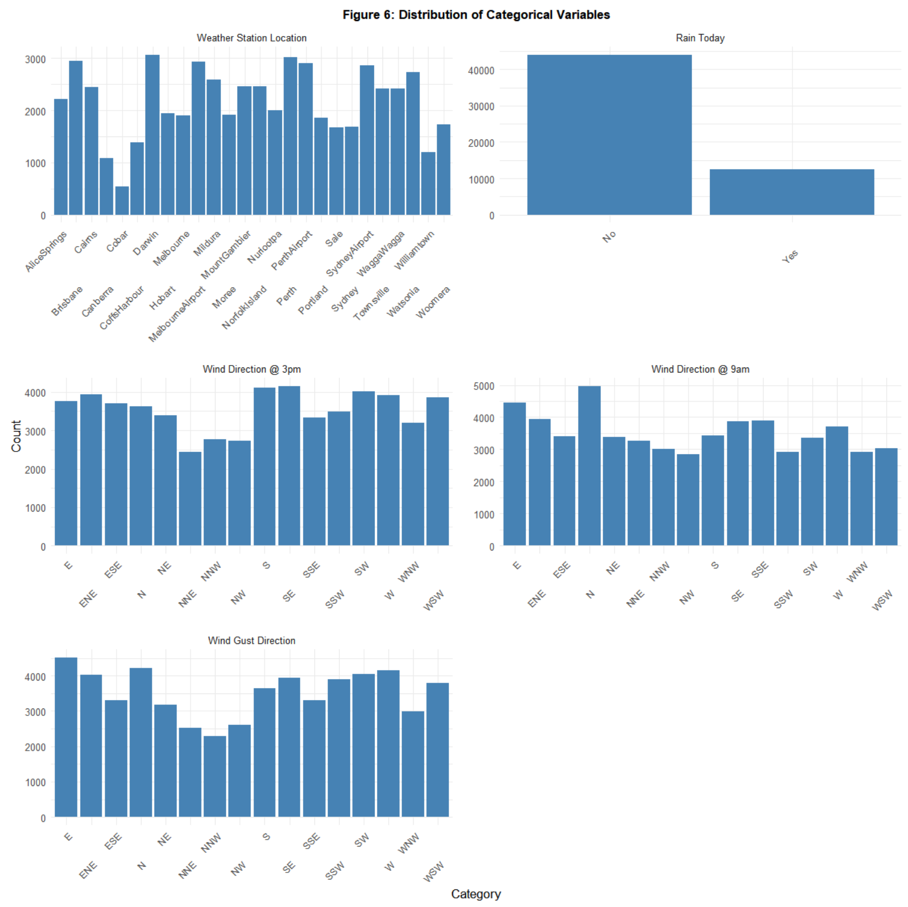
```
# Define the categorical variables
 categorical_vars <- c("Location", "WindGustDir", "WindDir9am", "WindDir3pm",
                      "RainToday"); df_cat <- csvdata[, categorical_vars]
 df_cat_long <- pivot_longer(df_cat, cols = everything(), names_to =
"Variable", values_to = "Category")
location_data <- subset(df_cat_long, Variable == "Location")
# Plot bar plot based on location
 ggplot(location_data, aes(x = reorder(Category, table(Category)[Category])))
+ geom_bar(fill = "steelblue") + coord_flip() + labs(title = "Figure 5: Weather
Station Locations", x = "Location", y = "Count") + theme_minimal(base_size =
14) + theme(plot.title = element_text(hjust = 0.5, face = "bold", size = 14))
```



s

We see something similar when counting the frequency of observations for other variables

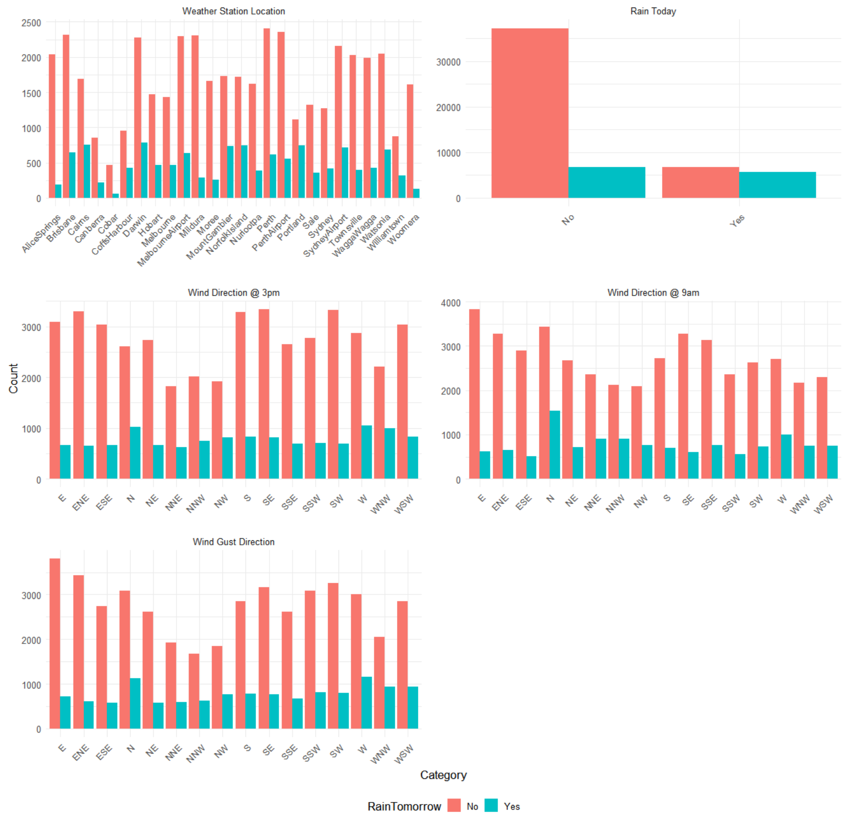
```
df_cat$Location <- str_wrap(df_cat$Location, width = 12)
# readjust previous data frame
 df_cat_long <- pivot_longer(df_cat,cols = everything(), names_to =
"Variable", values_to = "Category")
 # Plot for other variables
 ggplot(df_cat_long, aes(x = Category)) + geom_bar(fill = "steelblue") +
facet_wrap(~ Variable,scales = "free",ncol = 2,labeller = labeller(Variable =
c(Location = "Weather Station Location", WindGustDir = "Wind Gust Direction",
WindDir9am = "Wind Direction @ 9am", WindDir3pm="Wind Direction @ 3pm",
RainToday= "Rain Today"))) + scale_x_discrete(guide = guide_axis(angle = 45,
n.dodge = 2)) +labs(title = "Figure 6: Distribution of Categorical
Variables", x = "Category", y = "Count”) +theme_minimal(base_size = 14) +
theme(panel.spacing = unit(1.2, "lines"),plot.title = element_text(hjust =
0.5, size = 14, face = "bold"))
```



We can group the variables by RainTomorrow's value and use bar plots again.

```
# creat data frame of categorical variables and RainTomorrow
 df_cat2 <- csvdata[, c(categorical_vars, "RainTomorrow")]
 df_cat_long2 <- pivot_longer(df_cat2,cols = -all_of("RainTomorrow"),names_to
= "Variable",values_to = "Category")
# Plot grouped bar charts
 ggplot(df_cat_long2, aes(x = Category, fill = RainTomorrow)) +
geom_bar(position = "dodge") +facet_wrap(~ Variable,scales = "free",ncol = 2,
# Fewer columns for increased horizontal space
labeller = labeller(Variable = c(Location = "Weather Station Location",
WindGustDir = "Wind Gust Direction", WindDir9am= "Wind Direction @ 9am",
WindDir3pm = "Wind Direction @ 3pm", RainToday= "Rain Today”))) +
labs(title = "Figure 7: Categorical Variables Grouped by Rain Tomorrow",x =
"Category",y = "Count") +theme_minimal(base_size = 14) +theme(axis.text.x
= element_text(angle = 45, hjust = 1),panel.spacing = unit(1.2, "lines"),
plot.title    = element_text(hjust = 0.5, size = 14, face = "bold"),
legend.position = "bottom")
```

Figure 7: Categorical Variables Grouped by Rain Tomorrow

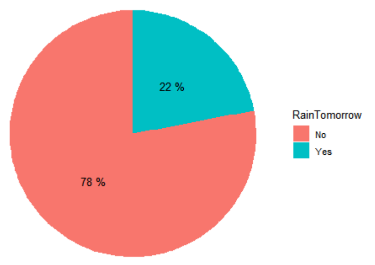


The Response

A pie chart displays the proportions of the response variable.

```
# Calculate counts and proportions for RainTomorrow
 rain_counts <- table(csvdata$RainTomorrow)
 rain_df <- data.frame(RainTomorrow = names(rain_counts), Count =
as.vector(rain_counts))
 rain_df$Percent <- round(rain_df$Count / sum(rain_df$Count) * 100, 1)
ggplot(rain_df, aes(x = "", y = Percent, fill = RainTomorrow)) +
   geom_bar(stat = "identity", width = 1) + coord_polar(theta = "y") +
   geom_text(aes(label = paste(Percent, "%")), position =
position_stack(vjust = 0.5)) + theme_void() + labs(title = "Figure 8:
Distribution of Rain Tomorrow")
```

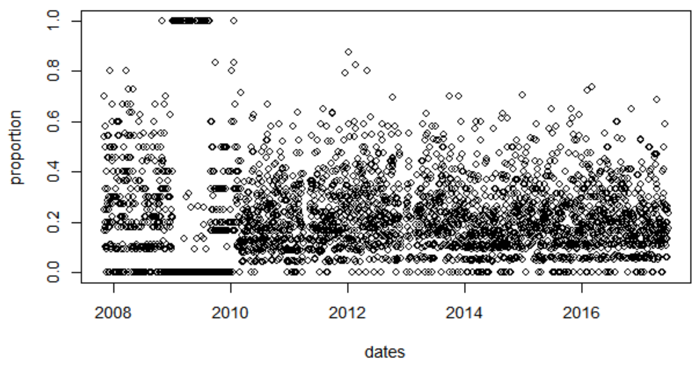
Figure 8: Distribution of Rain Tomorrow



A plot for the proportion of “Yes” of the response to Date shows that there is no specific pattern to the proportion it rains tomorrow across Australia.

```r
# Plot for proportion of "Yes" to RainTomorrow by date
 dates1 <- as.Date(csvdata$Date); dates <- unique(as.Date(csvdata$Date))
 raintomorrownumber <- as.numeric(as.factor((csvdata$RainTomorrow)))-1
 proportion <- tapply(raintomorrownumber, dates1, mean)
plot(proportion~dates,
     main = "Figure 9: Proportion of Rain Across Australia by Date")
```

Figure 9: Proportion of Rain Across Australia by Date



```r
# Mean of the proportion of "Yes" in RainTomorrow response variable
 mean(raintomorrownumber)
```

```
## [1] 0.2202588
```

Main Data analysis

Split into Train and Test Sets

```r
# Dropping irrelevant Date variable
 csvdata <- subset(csvdata, select = -Date)
 # Scaling data
 csvdata_scaled <- csvdata; numeric_cols <- sapply(csvdata, is.numeric)
 csvdata_scaled[numeric_cols] <- lapply(csvdata_scaled[numeric_cols],
function(x) as.numeric(scale(x))); weatheraus <- csvdata_scaled
set.seed(123)
 #dim(weatheraus) ### 56420    22
 # Split into train and test sets
 trainsize <- dim(weatheraus)[1]*0.7; train_ind <-
sample(1:dim(weatheraus)[1], replace=FALSE, size=trainsize); train <-
weatheraus[train_ind,]; test <- weatheraus[-train_ind,]
```

Logistic-based Models

“Vanilla” Logistic Regression

```
# Target variable is the RainTomorrow variable
 # Create logistic regression model
 logmod <- glm(RainTomorrow ~. , data=train, family = binomial)
 # Setting covariates and responses for following models
 x <- model.matrix(RainTomorrow ~., data = train)[,-1]; y <-
as.numeric(train$RainTomorrow) - 1
```

Logistic Tuning

Elastic Net Regularization

```
# Uses cross validation, thus do not need a validation set
 alpha_grid <- seq(0, 1, by = 0.05) # creates a sequence of alpha values to
try
 set.seed(123) # set for reproducibility
 foldid <- sample(rep(1:10, length.out = nrow(x))) # samples for k-fold CV

# performing a foreach loop that considers each alpha value in a
# cross-validated glmnet for elastic net regression
 results <- foreach(alpha_val = alpha_grid, .combine = rbind,
.packages="glmnet") %do% {cv_fit <- cv.glmnet(x, y, alpha = alpha_val, family
= "binomial", type.measure = "auc", foldid = foldid); data.frame(alpha =
alpha_val, auc = max(cv_fit$cvm))}
best_row <- results[which.max(results$auc), ]; best_alpha <- best_row$alpha
 best_alpha # if 1, LASSO, if 0, Ridge

## [1] 1

# Best lambda for elastic-net
 cvfitelastic <- cv.glmnet(x, y, alpha = best_alpha, type.measure = "auc",
family = "binomial"); elastic_minlambda <- cvfitelastic$lambda.min
 elastic_minlambda # best lambda value

## [1] 0.0001222504

# Best regularization model
 elastic.mod.best <- glmnet(x, y, alpha = best_alpha, lambda =
elastic_minlambda, family=binomial)
```

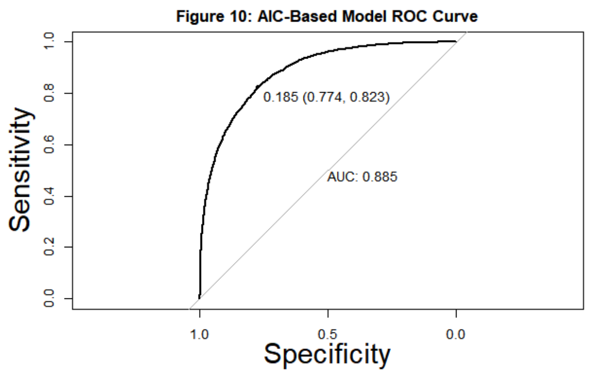
Backwards Selection by AIC

```
# AIC Selection for the large logistic model at the beginning
 best_logis_model <- step(logmod, trace=0)
```

Testing Logistic-Based Models

AIC-Based model

```
# Splitting the response and covariates in test data
 weatheraus.test <- test$RainTomorrow; test_covariates <- subset(test, select
= -RainTomorrow)
 # Making new predictions using the AIC-based model
 yhat.log <- predict(best_logis_model, newdata=test_covariates,
type="response")
 # Defining the ROC curve for AIC model
 roc_obj <- roc(response=weatheraus.test, predictor=yhat.log)

# Obtaining the AUC value for the ROC curve
 AUC <- auc(roc_obj)
 # Defining the metrics of the ROC curve
 roc_logistic <- c(coords(roc_obj, "b", ret=c("threshold","se","sp",
"accuracy"), best.method="youden"),AUC); names(roc_logistic) <-
c("Threshold","Sensitivity","Specificity","Accuracy","AUC")
 # Showing the metrics of the ROC curve
 t(roc_logistic) # Threshold: 0.1848469, AUC = 0.8845454

# Plotting the ROC curve for AIC model
 plot(roc_obj,legacy.axes=FALSE,print.auc=TRUE,print.thres=TRUE,cex.lab=2,
      main = "Figure 10: AIC-Based Model ROC Curve")
```



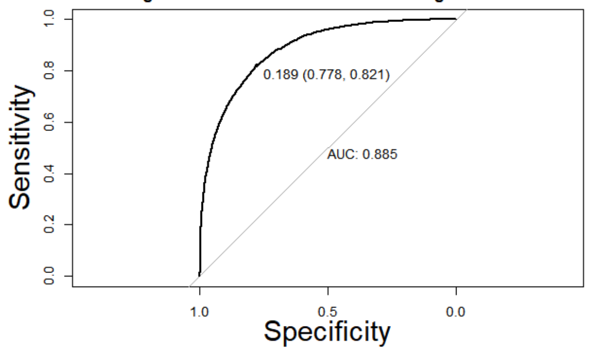
```
# Building the confusion matrix for AIC-based model
 weatherdata <- data.frame(weatheraus.test)
 weatherdatanext <- mutate(weatherdata, predprob=yhat.log)
 weatherdatanext <- mutate(weatherdatanext, predout=ifelse(predprob <
roc_logistic["Threshold"], "no", "yes"))
 # Showing the confusion matrix for the AIC-based model
 (cm <- xtabs( ~ weatheraus.test + predout, weatherdatanext))

##                 predout
 ## weatheraus.test    no    yes
##             No  10203  2972
##             Yes   663  3088

# Metrics of the confusion matrix
 tpr <- cm[2,2] / (cm[2,2]+cm[2,1]); fpr <- cm[1,2] / (cm[1,2]+cm[1,1])
 tnr <- cm[1,1] / (cm[1,1]+cm[1,2]); ppv <- cm[2,2] / (cm[2,2]+cm[1,2])
 npv <- cm[1,1] / (cm[1,1]+cm[2,1]); acc <- (cm[2,2]+cm[1,1]) /
(cm[2,2]+cm[2,1]+cm[1,2]+cm[1,1])
 cat("Sensitivity: ", tpr, ". FPV: ", fpr, ". Specificity: ", tnr,
    ". Precision: ", ppv, ". NegPredVal: ", npv, ". Accuracy: ", acc)

## Sensitivity:  0.8232471 . FPV:  0.2255787 . Specificity:  0.7744213 .
Precision:  0.509571 . NegPredVal:  0.938984 . Accuracy:  0.7852416
```

Elastic Net Regularized Model

ROC curve for elastic net model

```
# Splitting the response and covariates in test data
 weatheraus.test <- as.numeric(test$RainTomorrow)-1
 test_covariates <- model.matrix(RainTomorrow ~. , data = test)[,-1]
 # Making new predictions using the elastic net model
 yhat.elastic <- predict(elastic.mod.best, newx=test_covariates,
type="response")
 # Defining the ROC curve for elastic net
 roc_obj <- roc(response=weatheraus.test,predictor=as.numeric(yhat.elastic))

# Obtaining the AUC value for the ROC curve
 AUC <- auc(roc_obj)
 # Defining the metrics of the ROC curve
 roc_logistic <- c(coords(roc_obj, "b",
ret=c("threshold","se","sp","accuracy"), best.method="youden"),AUC)
 names(roc_logistic) <- c("Threshold","Sensitivity","Specificity",
                          "Accuracy","AUC")
 # Showing the metrics of the ROC curve
 t(roc_logistic) # Threshold: 0.1888685, AUC: 0.8846742

# Plotting the ROC curve for elastic net
 plot(roc_obj,legacy.axes=FALSE,print.auc=TRUE,print.thres=TRUE,cex.lab=2,
     main = "Figure 11: ROC Curve for Elastic Net Logistic Model")
```

Figure 11: ROC Curve for Elastic Net Logistic Model



```
# Building the confusion matrix for elastic net
 weatherdata <- data.frame(weatheraus.test)
 weatherdatanext_elastic <- mutate(weatherdata, predprob=yhat.elastic)
 weatherdatanext_elastic <- mutate(weatherdatanext, predout=ifelse(predprob <
roc_logistic["Threshold"], "no", "yes"))
 # Showing the confusion matrix for the elastic net model
 (cm <- xtabs( ~ weatheraus.test + predout, weatherdatanext_elastic))

##                predout
 ## weatheraus.test    no   yes
 ##              No 10259  2916
 ##              Yes   687  3064

# Metrics of the confusion matrix; reuse variable names from previous
 cat("Sensitivity: ", tpr, ". FPV: ", fpr, ". Specificity: ", tnr,
    ". Precision: ", ppv, ". NegPredVal: ", npv, ". Accuracy: ", acc)

## Sensitivity:  0.8168488 . FPV:  0.2213283 . Specificity:  0.7786717 .
Precision:  0.5123746 . NegPredVal:  0.9372373 . Accuracy:  0.7871322
```

Random Forest

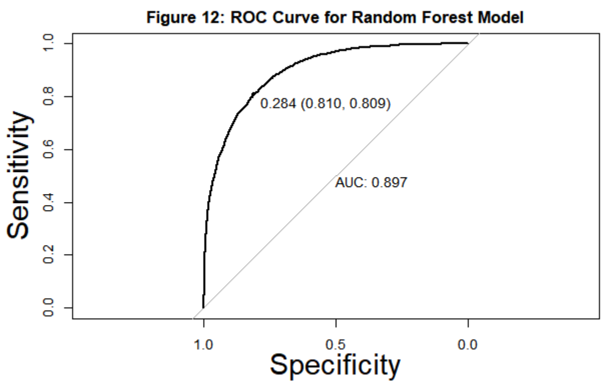
Random Forest Tuning

```
# parameter grid for random forest
 tuneGrid <- expand.grid(mtry = seq(2, sqrt(ncol(train)-1), length.out = 3),
   nodesize = c(1, 5, 10),ntree = c(500, 1000))
 best_accuracy <- 0; best_params <- list()
# for loop that performs grid search for the best random forest model
 for(i in 1:nrow(tuneGrid)) {set.seed(123); model <-randomForest(
RainTomorrow~.,data = train, ntree = tuneGrid$ntree[i], mtry =
tuneGrid$mtry[i],nodesize = tuneGrid$nodesize[i], importance =
TRUE);current_accuracy <- 1 - model$err.rate[model$ntree, "OOB"];
  if(current_accuracy > best_accuracy) {best_accuracy <- current_accuracy;
best_params <- list(mtry = tuneGrid$mtry[i], nodesize = tuneGrid$nodesize[i],
ntree = tuneGrid$ntree[i] }}
```

```r
set.seed(123)
 # Using the final parameters, run the random forest model
 rf.final <- randomForest(RainTomorrow~.,data = train,ntree =
best_params$ntree,mtry = best_params$mtry,nodesize = best_params$nodesize,
importance = TRUE)
```

Random forest testing

```r
# Splitting test response and covariates
 weatheraus.test <- test$RainTomorrow
 test_covariates <- subset(test, select = -RainTomorrow)
 # Predicting on the random forest model
 yhat.rf = predict(rf.final,newdata=test_covariates, type="prob")[, "Yes"]
 # defining ROC curve for random forest
 roc_obj <- roc(response=weatheraus.test, predictor=yhat.rf)

# Obtaining AUC of ROC curve
 AUC <- auc(roc_obj)
 # Obtaining metrics of ROC curve
 roc_rf <- c(coords(roc_obj, "b", ret=c("threshold","se","sp","accuracy"),
                   best.method="youden"),AUC)
 names(roc_rf) <-c("Threshold","Sensitivity","Specificity","Accuracy","AUC")
 # Showing metrics of ROC curve for random forest
 t(roc_rf) # Threshold: 0.2845, AUC: 0.897426

# Plotting ROC curve
 plot(roc_obj,legacy.axes=FALSE,print.auc=TRUE,print.thres=TRUE,cex.lab=2,
      main= "Figure 12: ROC Curve for Random Forest Model")
```



```r
# Building confusion matrix of random forest
 weatherdata <- data.frame(weatheraus.test)
 weatherdatanext <- mutate(weatherdata, predprob=yhat.rf)
 weatherdatanext <- mutate(weatherdatanext,predout=ifelse(predprob <
roc_rf['Threshold'], "no", "yes"))
```

```
 # Showing confusion matrix for random forest
 (cm <- xtabs( ~ weatheraus.test + predout, weatherdatanext))
##                 predout
 ## weatheraus.test    no   yes
 ##              No  10676  2499
 ##              Yes   715  3036
```

```
# Metrics of the confusion matrix; reuse variables from last model
 cat("Sensitivity: ", tpr, ". FPV: ", fpr, ". Specificity: ", tnr,
    ". Precision: ", ppv, ". NegPredVal: ", npv, ". Accuracy: ", acc)

## Sensitivity:  0.8093842 . FPV:  0.1896774 . Specificity:  0.8103226 .
Precision:  0.5485095 . NegPredVal:  0.9372311 . Accuracy:  0.8101146
```

```
# importance scores of the random forest
 importance(rf.final) # cut to top 10 for space, MD=MeanDecrease

##                      No          Yes  MDAccuracy  MDGini
 ## Location      173.73077 -15.87916463 164.11633   1066.4670
 ## MinTemp        57.56354   4.17358363 60.56405    295.8711
 ## MaxTemp        50.15693   2.82588232 53.22253    277.4048
 ## Rainfall       39.03759  47.85419408 56.34189    498.8398
 ## Evaporation    52.28141  -3.85814603 52.87798    273.5478
 ## Sunshine       87.68533  87.31686520 120.93480   1353.3294
 ## WindGustDir   121.91124 -40.10826329 100.39746   768.5614
 ## WindGustSpeed  85.57415  69.65502102 101.47685   548.7361
 ## WindDir9am     88.94198 -17.25686656 72.36780    790.7373
 ## WindDir3pm    115.43020 -21.77958930 98.73046    770.6976
```

```
par(mar = c(5, 10, 4, 2)); par(cex = 0.8); varImpPlot(rf.final,sort=TRUE,type
= 1,  main = "Figure 13: Random Forest Variable Importance by Mean Decrease
Accuracy"); varImpPlot(rf.final,sort=TRUE,type = 2, main = "Figure 14: Random
Forest Variable Importance by Mean Decrease Gini")
```

Figure 13: Random Forest Variable Importance by Mean Decrease Accuracy

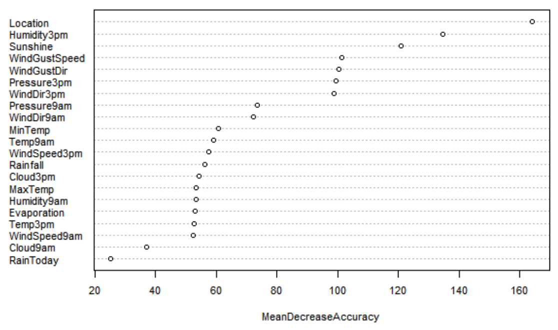
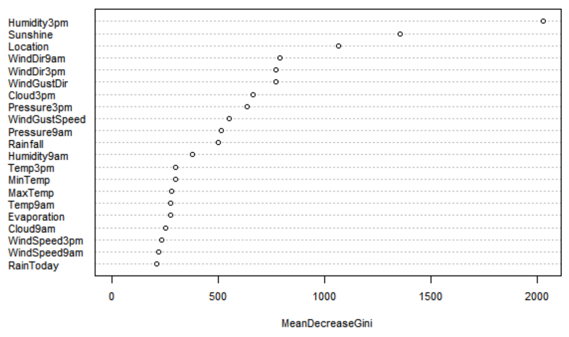


Figure 14: Random Forest Variable Importance by Mean Decrease Gini



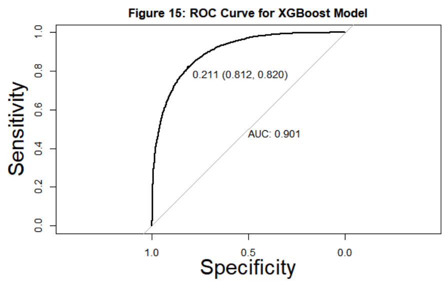
XGBoost (Extreme Gradient Boosting)

XGBoost Tuning parameters

```
# Data cleaning for XGBoost # For train data
 numeric_rain_train <- as.numeric(train$RainTomorrow)-1
 train_covariates <- subset(train, select = -RainTomorrow)
 train_covariates$Location <- as.numeric(train_covariates$Location)-1
 train_covariates$RainToday <- as.numeric(train_covariates$RainToday)-1
 train_covariates$WindDir3pm <- as.numeric(train_covariates$WindDir3pm)-1
 train_covariates$WindDir9am <- as.numeric(train_covariates$WindDir9am)-1
 train_covariates$WindGustDir <- as.numeric(train_covariates$WindGustDir)-1
# For test data
 test_covariates <- subset(test, select = -RainTomorrow)
 test_covariates$Location <- as.numeric(test_covariates$Location)-1
 test_covariates$RainToday <- as.numeric(test_covariates$RainToday)-1
 test_covariates$WindDir3pm <- as.numeric(test_covariates$WindDir3pm)-1
 test_covariates$WindDir9am <- as.numeric(test_covariates$WindDir9am)-1
 test_covariates$WindGustDir <- as.numeric(test_covariates$WindGustDir)-1;
set.seed(123)
 # Create boosting matrix of covariates
 train_data_boost <- xgb.DMatrix(data=as.matrix(train_covariates), label =
numeric_rain_train)
# To tune the other parameters (shrinkage and splits) via grid search
 depth_grid <- c(1, 2, 4, 6, 8) # splits; eta_grid <- c(0.0001, 0.001, 0.01,
0.05, 0.1, 0.2, 0.3) # shrinkage; results <- data.frame()
# CV for nrounds (number of trees)
 for (eta in eta_grid) {for (depth in depth_grid) {cv <- xgb.cv(data =
train_data_boost,params = list(objective = "binary:logistic", eval_metric =
"auc",eta = eta,max_depth = depth, nrounds = 1000,nfold = 10,
early_stopping_rounds = 10,verbose = 0); best_auc <-
max(cv$evaluation_log$test_auc_mean); best_nrounds <- cv$best_ntreelimit ;
     results <- rbind(results,data.frame(eta,depth,best_auc,best_nrounds))}}
# Best combination
 (best_combo <- results[which.max(results$best_auc), ])

##     eta depth best_auc best_nrounds
 ## 19 0.05     6 0.904917          442

# extracting best results for parameters
 best_tree_number <- results[which.max(results$best_auc), ]$best_nrounds;
best_eta <- results[which.max(results$best_auc), ]$eta;
 best_depth <- results[which.max(results$best_auc), ]$depth;

# best XGBoost model using cross-validated and grid search parameters
 boost.weatheraus <- xgboost(data=train_data_boost, nrounds=best_tree_number,
params = list(objective = "binary:logistic",eval_metric = "auc", eta =
best_eta, max_depth = best_depth , verbose=0)
```

XGBoost testing

```
# Preparation for ROC curve plot
 weatheraus.test <- as.numeric(test$RainTomorrow)-1;  at.boost <-
predict(boost.weatheraus,newdata=as.matrix(test_covariates)); roc_obj <-
roc(response=weatheraus.test, predictor=yhat.boost)

AUC <- auc(roc_obj); roc_boost <- c(coords(roc_obj, "b",
ret=c("threshold","se","sp","accuracy"),best.method="youden"),AUC);
names(roc_boost) <- c("Threshold","Sensitivity","Specificity","Accuracy",
"AUC")
 # Metrics for ROC curve on boosting
 t(roc_boost) c# Threshold: 0.2108152, AUC: 0.9011854

# ROC curve plot for boosting
 plot(roc_obj,legacy.axes=FALSE,print.auc=TRUE,print.thres=TRUE,cex.lab=2,
      main = "Figure 15: ROC Curve for XGBoost Model")
```



```
# Plotting confusion matrix for boosting
 weatherdata <- data.frame(weatheraus.test)
 weatherdatanext <- mutate(weatherdata, predprob=yhat.boost)
 weatherdatanext <- mutate(weatherdatanext,predout=ifelse(predprob <
roc_boost["Threshold"], "no", "yes"));(cm <- xtabs( ~ weatheraus.test +
predout, weatherdatanext))

##                predout
 ## weatheraus.test    no   yes
 ##                0 10699  2476
 ##                1   676  3075

# Metrics of the confusion matrix; reuse variables from last model
 cat("Sensitivity: ", tpr, ". FPV: ", fpr, ". Specificity: ", tnr,
    ". Precision: ", ppv, ". NegPredVal: ", npv, ". Accuracy: ", acc)
```

```
## Sensitivity:  0.8197814 . FPV:  0.1879317 . Specificity:  0.8120683 .
Precision:  0.5539542 . NegPredVal:  0.9405714 . Accuracy:  0.8137776
```

```
# Importance scores for XGBoost
 importance_scores_xgb <- xgb.importance(feature_names =
colnames(subset(test, select = -RainTomorrow)),model = boost.weatheraus
 ); importance_scores_xgb # Cut to top 10 for space
```

```
##            Feature        Gain      Cover  Frequency
 ##             <char>       <num>      <num>      <num>
 ##  1:    Humidity3pm 0.330877226 0.11879735 0.07269247
 ##  2:       Sunshine 0.181032877 0.08720889 0.07600373
 ##  3:    Pressure3pm 0.094619564 0.12157013 0.09995861
 ##  4: WindGustSpeed 0.070173690 0.09252476 0.06244826
 ##  5:       Location 0.040319438 0.07622481 0.06281043
 ##  6:       Cloud3pm 0.031867305 0.03366300 0.02297185
 ##  7:       Rainfall 0.027180482 0.02174821 0.03771730
 ##  8:    Pressure9am 0.025403197 0.07084240 0.06462127
 ##  9:        MinTemp 0.022564260 0.06028476 0.05799876
 ## 10:    Humidity9am 0.019608994 0.02880681 0.05142798
 ##            Feature        Gain      Cover  Frequency
```

```
xgb.plot.importance(importance_matrix = importance_scores_xgb, cex=1.2, main
= "Figure 16: XGBoost Variable Importance Plot", top_n = 10, xlab = "Gain")
```

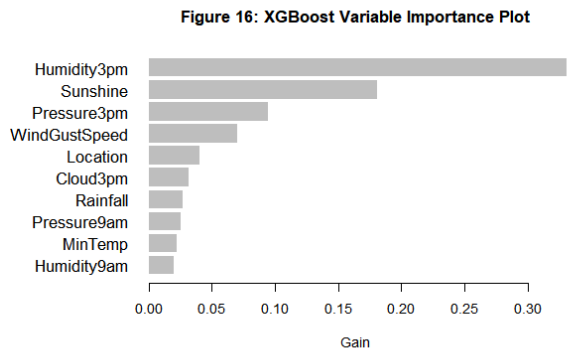


Figure 16: XGBoost Variable Importance Plot
Humidity3pm
Sunshine
Pressure3pm
WindGustSpeed
Location
Cloud3pm
Rainfall
Pressure9am
MinTemp
Humidity9am
Gain