Predictive Modeling Through Logistic Regression in R

Jomari Manto   
*College of Computer Studies & Engineering*  
*Jose Rizal University*Mandaluyong City, Philippines  
[jomari.manto@my.jru.edu](mailto:jomari.manto@my.jru.edu)

***Abstract*—This report presents the results of a predictive modeling activity involving logistic regression, conducted as part of an academic exercise under the course module on machine learning and predictive analytics. The task involved using the built-in “airquality" dataset in R to design, train, and evaluate logistic regression models aimed at predicting whether a day’s temperature falls within a defined “ideal” range. The exercise also included exploratory data analysis (EDA), variable selection using stepwise regression, and the evaluation of model performance using metrics such as accuracy, precision, recall, F1-score, and confusion matrix analysis. Different modeling approaches were explored and compared to identify the most effective configuration. All steps and experiments were documented and interpreted in this report.(***Abstract***)**

***Keywords—logistic regression, predictive modeling, stepwise selection, r programming, binary classification, confusion matrix(****key words****)***

# Introduction

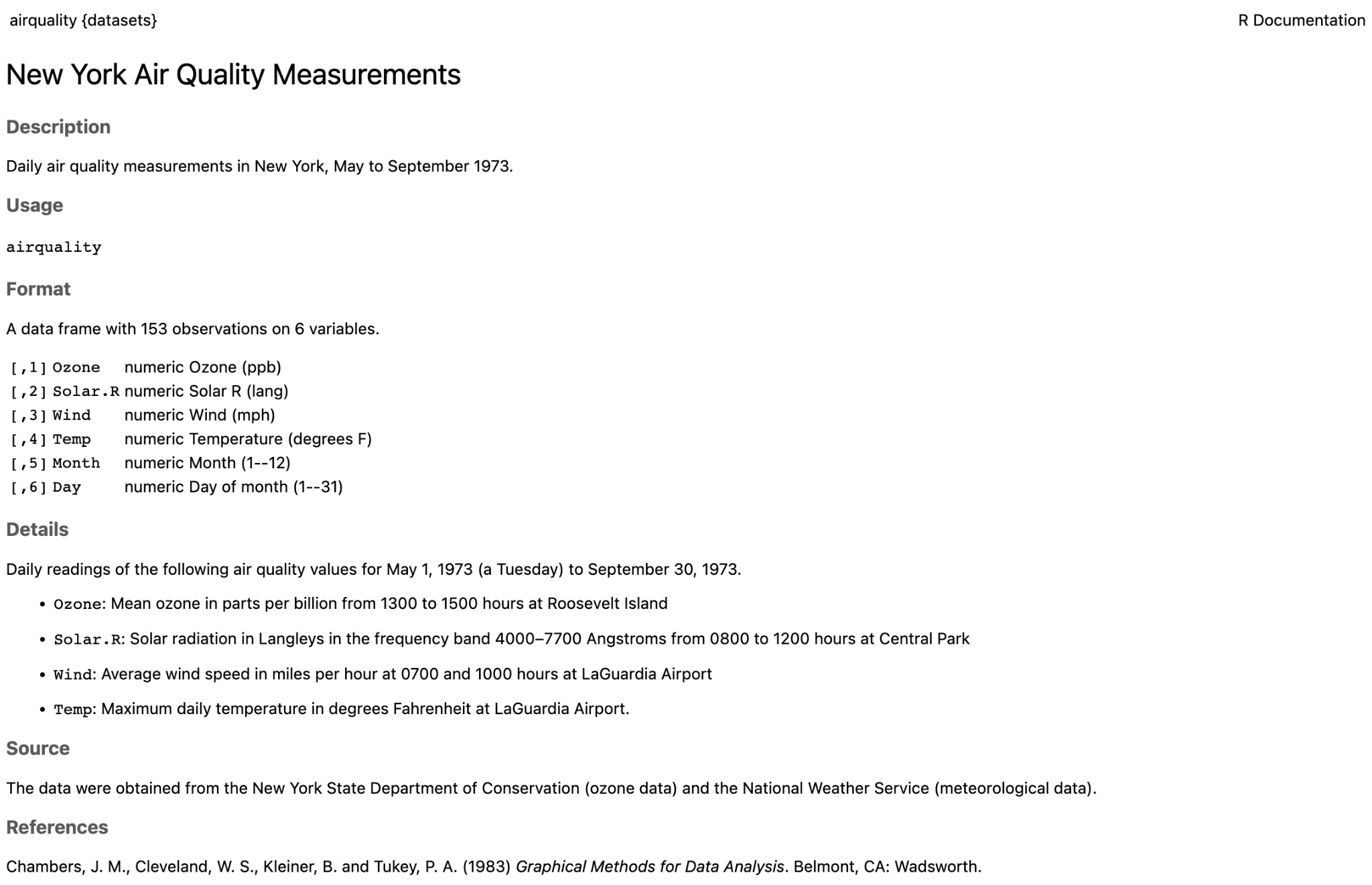
Logistic regression is a statistical method used for modeling the relationship between a set of independent variables and a binary outcome. Unlike linear regression, which predicts continuous values, logistic regression predicts the probability of a certain event occurring. The output is a value between 0 and 1, transformed by the logistic (sigmoid) function. Logistic regression is primarily used for classification tasks, where the goal is to categorize data into distinct classes.

Steps to conduct logistic regression are explored using the built-in “airquality” dataset in R, which provides air quality measurements for New York. This dataset will be used to demonstrate the process of building a logistic regression model for classification tasks.

# Methodology— Pre-modeling Stage

## Dataset Description

The “airquality” dataset is a built-in dataset in R, containing 153 observations and 6 variables. It provides daily air quality readings from May 1, 1973, to September 30, 1973, for New York. Figure 1 shows the complete metadata for the dataset, detailing the variables and their descriptions.



1. Dataset description for “airquality.”

## Data Preprocessing

Missing values in the Ozone and Solar.R variables were imputed using column means to preserve dataset integrity. A new binary target variable named *Ideal* was created to indicate whether the temperature was within the defined comfortable range *(Ideal = 1)* or not *(Ideal = 0).* The target was converted into a factor for binary classification. Numeric predictor variables (Ozone, Solar.R, Wind, and Temp) were then standardized using z-score normalization to ensure all variables contributed equally during model training.

The dataset was partitioned into three subsets using a 60-30-10 split:

60% for training

30% for validation

10% for testing

Random sampling was performed using set.seed(123) to ensure reproducibility.

## Data Preprocessing

To understand the structure and relationships within the data, a comprehensive EDA was performed. Listed below are the steps that were performed during the exploratory data analysis stage.

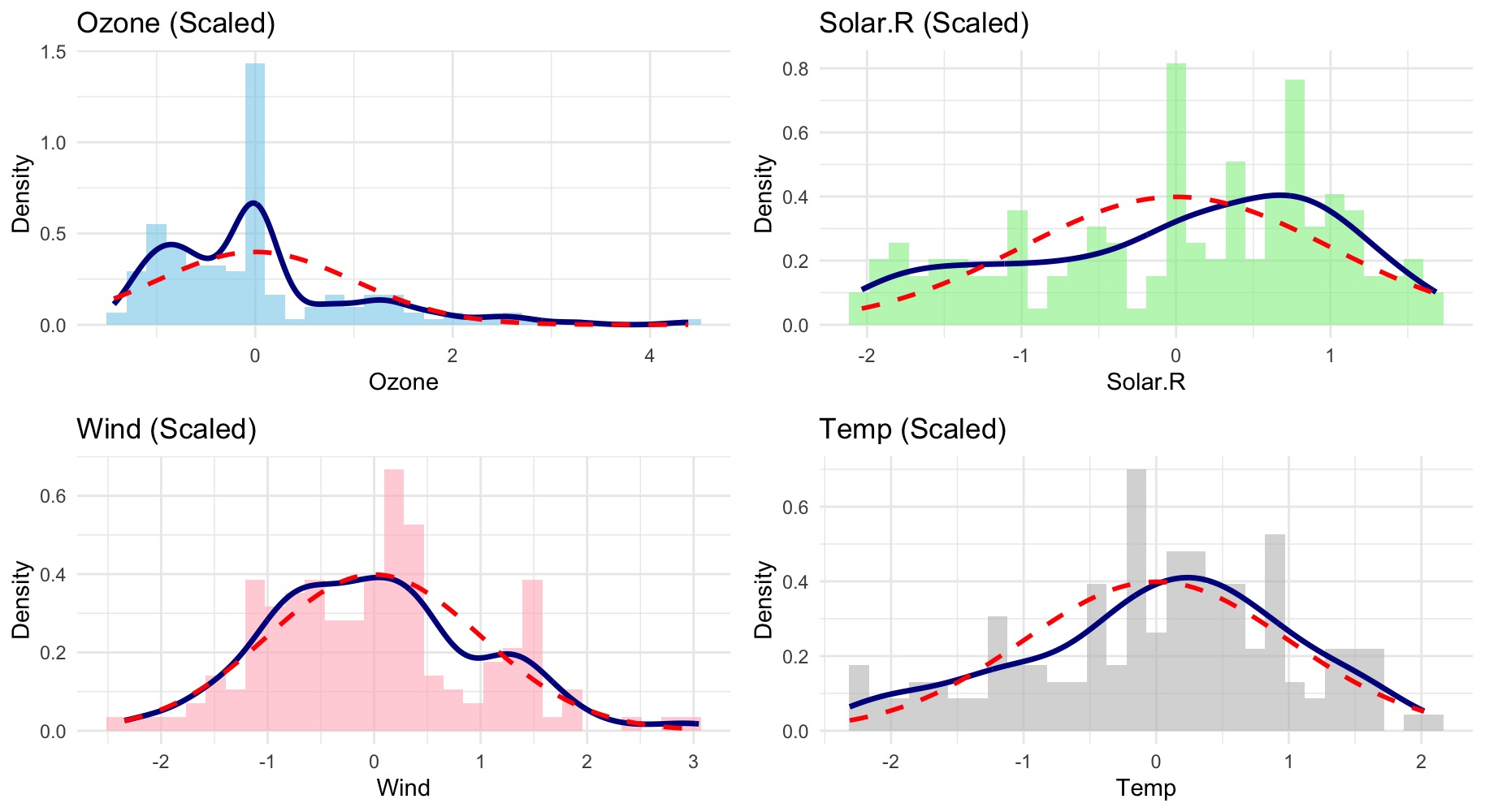
1. **Class Imbalance Check**—the distribution of the binary target variable *Ideal* was examined to assess whether the dataset was balanced. This step is crucial in classification tasks, as imbalanced data can bias the model toward the majority class, resulting in poor recall for the minority class.
2. Results Of Imbalance Check

| **Class** | **Count** | **Proportion** |
| --- | --- | --- |
| Not Ideal (0) | 117 | 76.50% |
| Ideal (1) | 36 | 23.50% |
| Total | 153 | 100% |

1. Results of imbalance check on the dataset.

Table I shows that most of the days in the dataset (76.5%) are labeled as *Not Ideal*, while only 23.5% are considered *Ideal*. In this context, *Ideal* days are those with temperatures between 68°F and 76°F, while days outside this range are labeled as *Not Ideal*. This imbalance means the model is more likely to correctly predict the majority class (Not Ideal) but may struggle to identify the fewer Ideal days. Because of this, we focused not only on accuracy but also on recall, precision, and F1-score to better understand the model’s performance on both classes.

1. **Distribution Plots of Predictors**—Histograms and density plots were created for each of the standardized numeric predictors (Ozone, Solar.R, Wind, and Temp) to inspect their distribution. This helped in identifying skewness, spread, and potential outliers that might influence model performance.



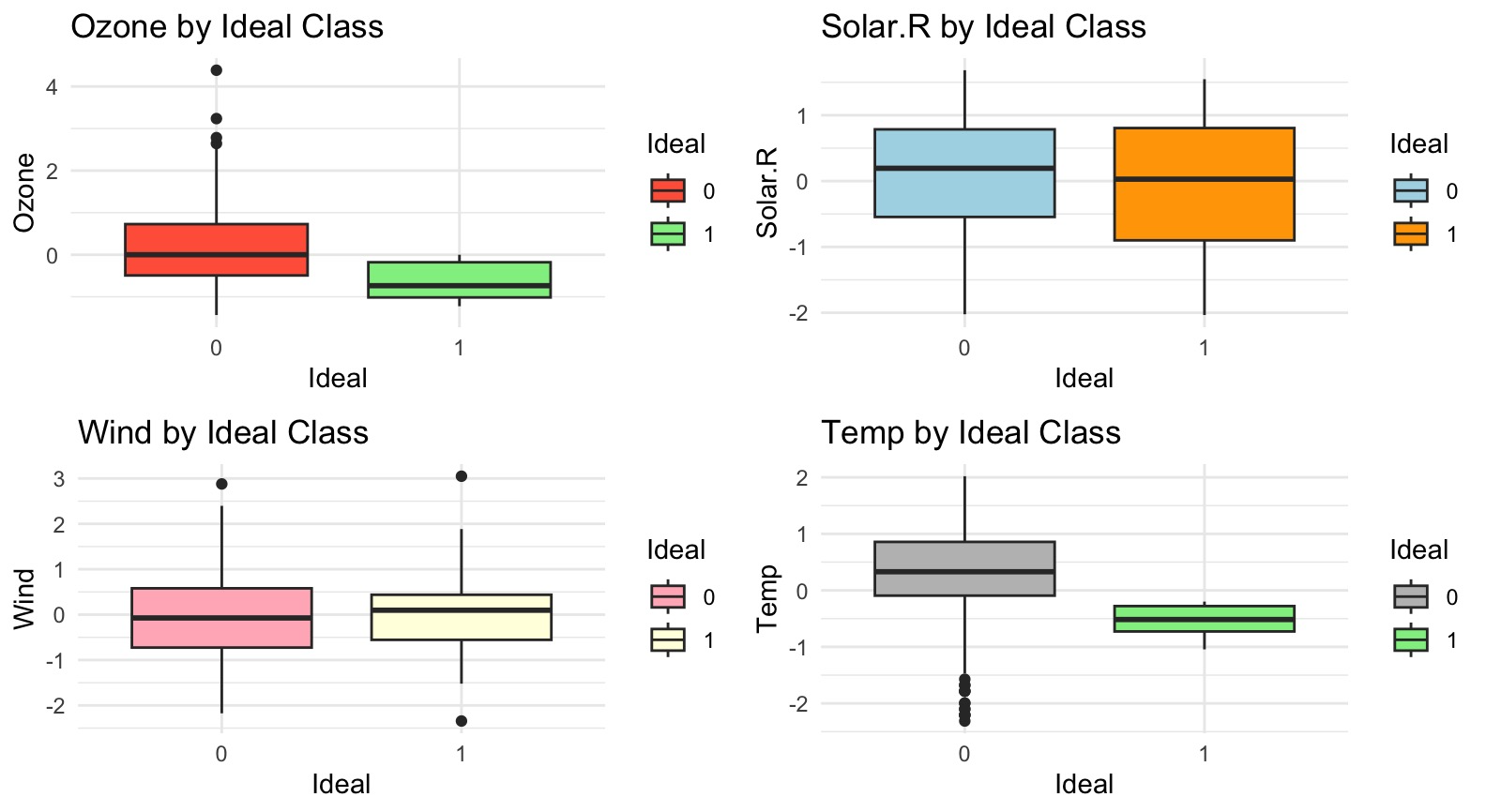
1. Distribution of Predictors

Figure 3 shows the distributions of the predictors (Ozone, Solar.R, Wind, and Temp). The shaded bars represent the actual data, while the solid blue line shows the real distribution. The dashed red line represents an ideal normal distribution for comparison.

* Ozone has many low values with some higher outliers, making it uneven (skewed).
* Solar.R is somewhat balanced but still slightly uneven.
* Wind is fairly balanced and matches the normal shape closely.
* Temp also shows a balanced distribution similar to the normal curve.

This suggests that while Wind and Temp have evenly distributed values, Ozone and Solar.R have less consistent data, which might affect the model's accuracy or suggest adjustments like transforming these variables.

1. **Boxplots by Ideal Class**—Boxplots were used to easily compare how the values of each predictor (Ozone, Solar.R, Wind, Temp) differed between days considered *Ideal (1)* and *Not Ideal (0)*. This helped show which variables were good at separating the two groups.



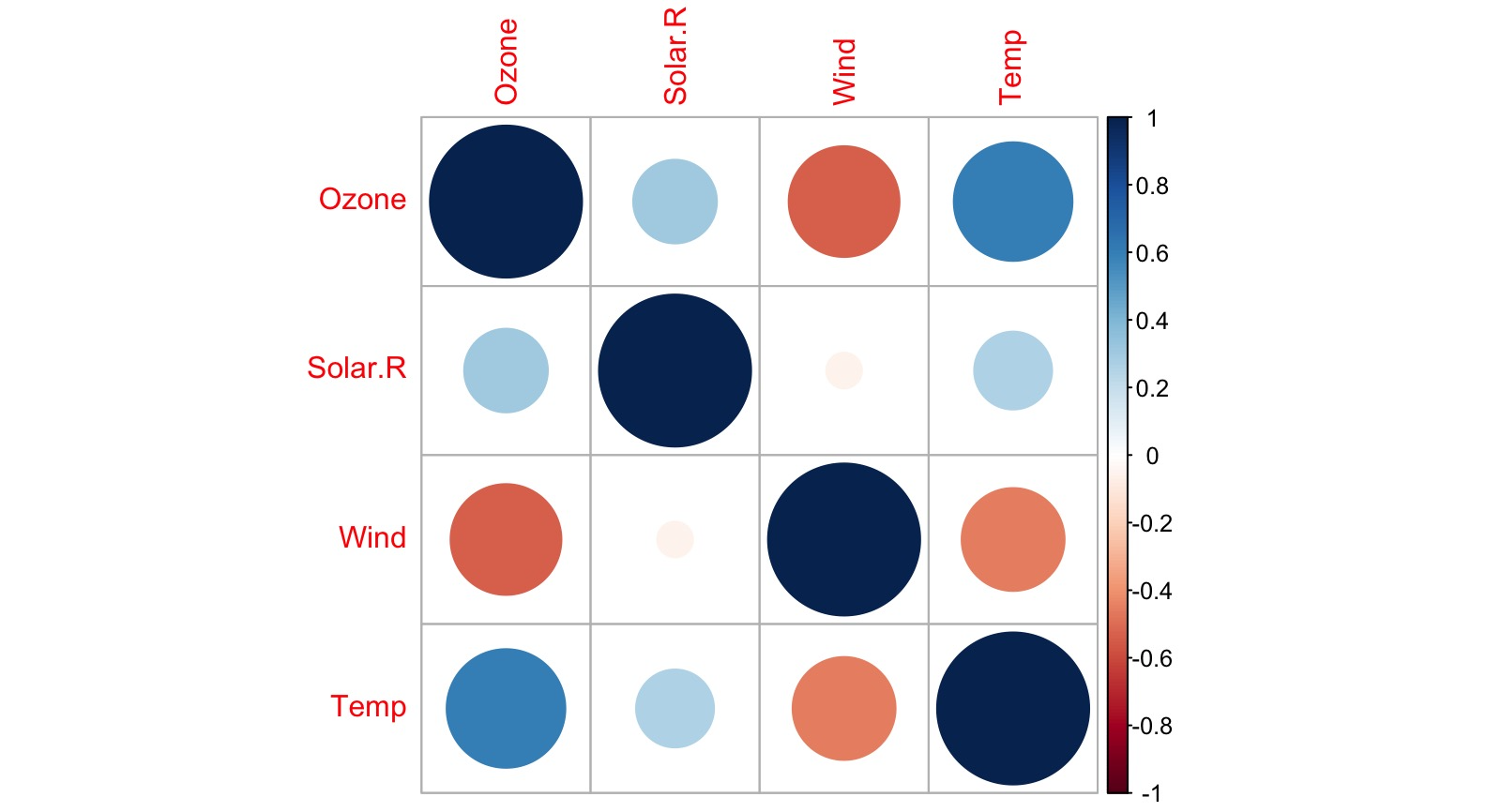
1. Distribution of Predictors

Figure 4 compares predictor variables (Ozone, Solar.R, Wind, Temp) between Ideal (1) days (68°F–76°F) and Not Ideal (0) days:

* Ozone: *Ideal* days have lower ozone levels compared to *Not Ideal* days.
* Solar.R: Solar radiation is similar for both Ideal and *Not Ideal* days, showing little difference.
* Wind: Wind speed is slightly different, but mostly overlaps, providing limited distinction.
* Temp: *Ideal* days have more consistent temperatures, while *Not Ideal* days vary widely.

Overall, Temp clearly separates the two groups (by design), Ozone moderately helps differentiate them, while Solar.R and Wind have less impact.

1. **Correlation Matrix—**A correlation matrix was created to check how similar or related the predictors were to each other. This helped identify if any two variables were providing the same information, which could make the model less effective.



1. Correlation Matrix

Figure 5 shows the correlation (relationship) between the predictor variables (Ozone, Solar.R, Wind, and Temp).

* Dark Blue circles represent strong positive relationships (both variables increase or decrease together).
* Dark Red circles represent strong negative relationships (one variable increases while the other decreases).
* Small or lighter circles indicate weak or no clear relationships.

From the figure:

* Ozone and Temp show a noticeable positive relationship (as temperature rises, ozone tends to increase).
* Wind and Temp show a noticeable negative relationship (higher temperatures usually mean lower wind speeds).
* Other pairs (Solar.R with other variables) have weaker relationships, indicating less impact on each other.

This visualization helps confirm which variables provide unique or overlapping information, ensuring the model isn't overloaded with redundant data.

1. **Multicollinearity Check (VIF)—**A statistical check (VIF) was performed to ensure that the predictors weren't too similar to each other. All variables passed the test easily, showing they each gave unique and useful information to the model.
2. Variance Inflation Factor (VIF) Results

| **Predictor Variable** | **VIF Value** | **Interpretation** |
| --- | --- | --- |
| Ozone | 1.54 | No multicollinearity |
| Solar.R | 1.13 | No multicollinearity |
| Wind | 1.24 | No multicollinearity |
| Temp | 1.24 | No multicollinearity |

1. VIF results. VIF values less than 5 indicate no concern regarding multicollinearity.

All predictor variables had low VIF values (below 2), indicating no issues with multicollinearity. Each variable provided unique information to the logistic regression model. After performing the EDA stage, the next part of the process is to create the model.

# Model Building (Training and Evaluation)

A logistic regression model was used to predict whether a day’s temperature fell within the defined “Ideal” range (68°F to 76°F). The initial model included all four standardized predictors: Ozone, Solar.R, Wind, and Temp. This baseline model was trained on the 60% training set and evaluated on the 30% validation set using classification metrics such as accuracy, precision, recall, F1-score, and confusion matrix analysis.

To improve model efficiency, stepwise selection was applied using forward, backward, and bidirectional methods based on Akaike Information Criterion (AIC). All three approaches produced the same reduced model, selecting only Ozone and Wind as predictors. This stepwise model achieved the same validation accuracy as the full model but with fewer variables and a lower AIC, making it the more efficient choice.

Both the full and stepwise models were evaluated using the 10% held-out test set to assess their final performance. Evaluation focused not only on overall accuracy but also on recall and F1-score, due to the imbalance in the target variable.

1. Comparative Table of Results

| **Predictors** | **Val Accuracy** | **Val Recall** | **Val Precision** | **Val F1** | **Test Accuracy** | **AIC** |
| --- | --- | --- | --- | --- | --- | --- |
| Ozone, Solar.R, Wind, Temp | 0.7391 | 0.0833 | 0.5 | ~0.14 | 0.75 | ~84 |
| Ozone, Wind | 0.7391 | 0.0833 | 0.5 | ~0.14 | 0.75 | 80.05 |
| Ozone, Wind (0.4 Threshold) | 0.6957 | 0.0833 | 0.25 | ~0.125 | Same as Exp 2 | 80.05 |

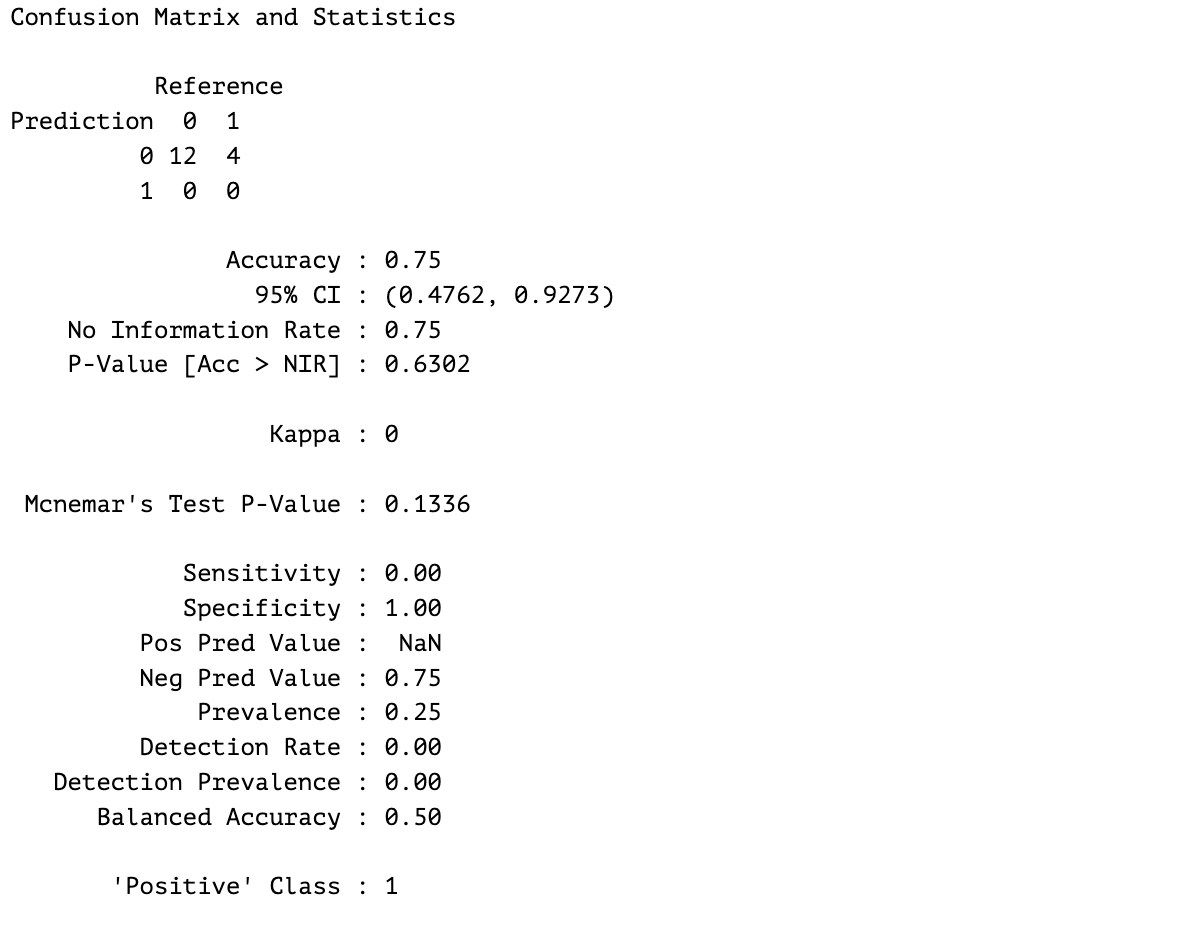
1. Comparative table of results.

To better understand which logistic regression setup worked best, three models were created and compared.

* **Model 1 (Full Model):** This used all four predictors—Ozone, Solar.R, Wind, and Temp. It served as our baseline and performed reasonably well in terms of accuracy but struggled to detect “*Ideal*” days (low recall).
* **Model 2 (Stepwise Model):** To make the model more efficient, we applied stepwise regression, which selected only Ozone and Wind. This simpler model gave the same accuracy as the full model but with fewer variables and a lower AIC, making it more efficient and easier to interpret.
* **Model 3 (Threshold Tuning):** Since our dataset was imbalanced (only 23.5% of days were “*Ideal*”), we tried lowering the classification threshold from 0.5 to 0.4 to improve recall. However, recall remained the same, and precision dropped, showing that threshold tuning alone did not improve the model’s ability to catch Ideal days.

The table comparing the three models shows that while accuracy stayed consistent, recall stayed low across all models. This means that the models had trouble identifying Ideal days, which were the minority. The stepwise model stood out as the most efficient because it performed the same as the full model but used fewer variables and had the lowest AIC.  
 These experiments showed that simplifying the model did not hurt performance, but solving the class imbalance would need other techniques, such as resampling, which was no longer applied in the exercise.   
 After comparing all three models, the stepwise model using only Ozone and Wind was chosen as the final model. It had the same accuracy as the full model but was simpler and easier to understand. Even though recall stayed low across all models because of class imbalance, the stepwise model had the lowest AIC, making it the most efficient choice. This final model was then used to make predictions on the test set to check how well it performs on new data.

To evaluate the final model, a confusion matrix was generated using the test set. The matrix provides a detailed view of how the model classified each case. As shown in Figure 8, while the model correctly identified all Not Ideal days (class 0), it failed to detect any *Ideal* days (class 1). This reinforces the earlier findings that, despite high overall accuracy, the model struggled with recall due to class imbalance.



1. Confusion Matrix of chosen model

The final model was applied to the test set to generate predictions. Each observation was assigned a probability of being an Ideal day. Using a threshold of 0.5, these probabilities were converted into binary predictions (0 = Not Ideal, 1 = Ideal). The predicted values were then compared to the actual class labels to evaluate model performance.

1. Actual vs. Predicted Classification of Ideal Days

| **Index** | **Ozone** | **Wind** | **Actual (Ideal)** | **Predicted Class** | **Predicted Probability** |
| --- | --- | --- | --- | --- | --- |
| 1 | -0.03936 | -0.72595 | 0 | 0 | 0.1793 |
| 3 | -1.05004 | 0.75007 | 1 | 0 | 0.3979 |
| 18 | -1.25915 | 2.39639 | 0 | 0 | 0.2275 |
| 45 | ~0 | 1.09068 | 0 | 0 | 0.0468 |
| 57 | ~0 | -0.55564 | 0 | 0 | 0.1493 |
| 61 | ~0 | -0.55564 | 0 | 0 | 0.1493 |
| 66 | 0.76222 | -1.52073 | 0 | 0 | 0.0636 |
| 77 | 0.2046 | -0.86787 | 0 | 0 | 0.1241 |
| 102 | ~0 | -0.38533 | 0 | 0 | 0.1333 |
| 110 | -0.66668 | -0.72595 | 1 | 0 | 0.4686 |
| 114 | -1.1546 | 1.23261 | 1 | 0 | 0.3647 |
| 119 | ~0 | -1.20849 | 0 | 0 | 0.2253 |
| 121 | 2.64419 | -2.17358 | 0 | 0 | 0.0017 |
| 128 | 0.16975 | -0.72595 | 0 | 0 | 0.1207 |
| 138 | -1.01519 | 0.43783 | 1 | 0 | 0.4377 |
| 140 | -0.84094 | 1.09068 | 0 | 0 | 0.2418 |

1. Actual vs. Predicted Classification of Ideal Days Using the Stepwise Logistic Regression Model (Test Set)

Table IV table compares the model’s predictions to the actual values in the test set. While the model correctly predicted Not Ideal days (0), it failed to identify any *Ideal* days (1), even when some probabilities were close. This supports the confusion matrix results and shows the model struggled with detecting *Ideal* days.

##### Conclusion

While this activity focused on a single dataset, it effectively demonstrated how logistic regression can be applied to predictive modeling involving categorical outcomes. However, it is important to conduct preliminary checks before applying the method directly. In this case, class imbalance significantly affected the model’s ability to detect minority outcomes. In real-world scenarios, such issues should be given greater attention, as they can lead to biased or misleading predictions if left unaddressed.

This activity also highlighted the importance of exploratory data analysis (EDA) in preparing for classification tasks. Understanding variable distributions, relationships, and class balance played a key role in guiding model design. In addition, stepwise logistic regression proved useful in selecting the most relevant predictors, helping to simplify the model without sacrificing performance. These steps are essential in developing interpretable and efficient models, especially when working with real-world data.

##### Acknowledgment

The student acknowledges the sources of instructional and technical support that contributed to the successful completion of this activity. The same support was also instrumental in previous related tasks, and continued assistance is sincerely appreciated. Sincere gratitude is extended to the following:

**The Creator** for the gift of life and abilities—still a work in progress, yet central to the pursuit of learning and future opportunities.

**Rodolfo C. Raga Jr., PhDCS**, for his insightful feedback and for encouraging the exploration of ideas beyond conventional limits.

**Innovators and technologists** whose tools and platforms have enabled efficient data processing, analysis, and visualization throughout this activity.

**Researchers and academics** whose foundational work and published insights guided the direction of this research.

**Readers and peers** for their time and constructive feedback; and

**AI-based tools like ChatGPT**, which served as a supplemental resource for clarification, troubleshooting, and support throughout the research process.