**Buoy Classification**

# **Why Classify Buoys?**

There are in total 72 buoys in the dataset, and they were not classified when we first received the data. Although data of a buoy in the dataset is arranged in time order, and we are able to separate data of a buoy from that of another by noticing the time gap that exists when the data jumps to another buoy and starts from an earlier date again, ambiguity in buoy labelling raises a critical question in real life: what if data of different buoys are collected and recorded at the same time? Then how can we possibly differentiate a buoy from another? It is true that location information of buoys are contained in the dataset, and since there is usually continuity in location of a buoy, previous location data can help us infer buoy index; however, we also notice that buoys tend not to get fixed at a certain location, they float around in the ocean and sometimes, a buoy can travel 5 longitude degrees away from its original location. As a result, location data alone is not a reliable indicator to predict buoy index, and thus we are required to use more complicated tools to clearly label buoys.

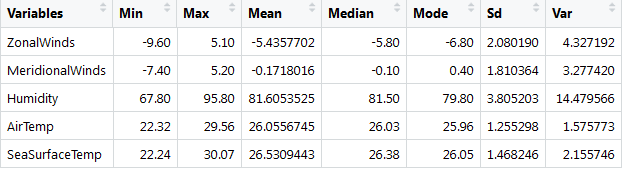
Owing to insufficient data of most buoys in the dataset (more than ⅔ of all buoys have fewer than 500 data points, and if we included them in classification, we would have had difficulty dealing with an imbalanced dataset), we will use 2 buoys for classification, which have more than 2,000 data points. Models we use in this report set an example for classification of more than 2 buoys in the future. By comparing results from logistic regression, KNN classification and Classification Tree results, we will know which model is best tuned to this task.

# **Descriptive Analysis**

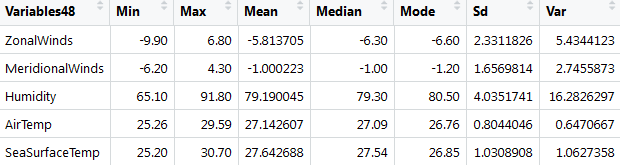
The “El Nino” dataset in the A1 includes the data from 72 buoys. In A2, we chose buoy 2 since it has the most data points (2298 entries). In this assignment, we decided to include buoy 48, which has the second most data (2240 entries). Because these two buoys collected a similar amount of the data, we developed logistic regression and K-NN analysis to predict the data source, from buoy 2 or buoy 48. It is also reasonable to discover the discrepancy of data from these two buoys during the same period. As the Sea.Surface.Temp is the key measurement of the occurrence of El Nino, we plotted Zonal.Winds, Meridional Winds, Humidity and Air Temp against Sea.Surface.Temp in the scatterplot. The blue line in the scatter plot is a linear fitted model, and the red one is to show the curvilinear relationships. We also compare the data of buoy 2 to that of buoy 48 side by side.

## **Buoy 2 and 48 Statistics Comparison**

### Buoy 2

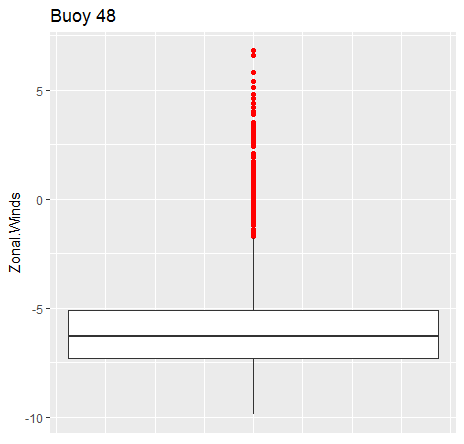
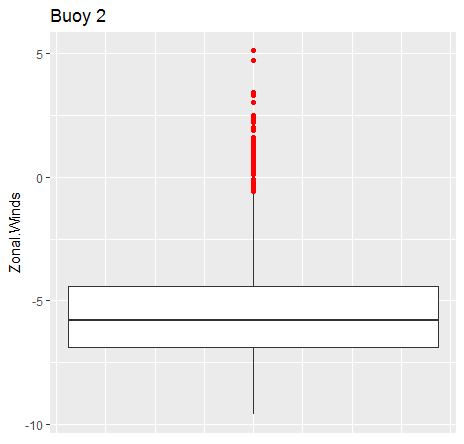


### Buoy 48

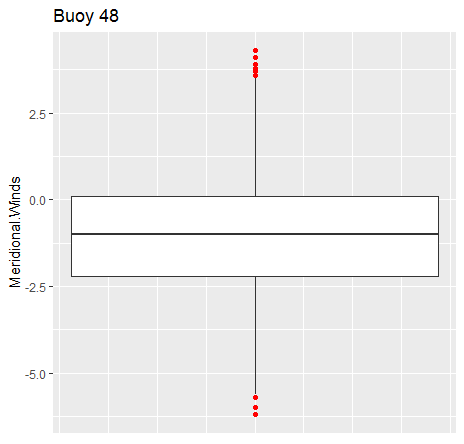
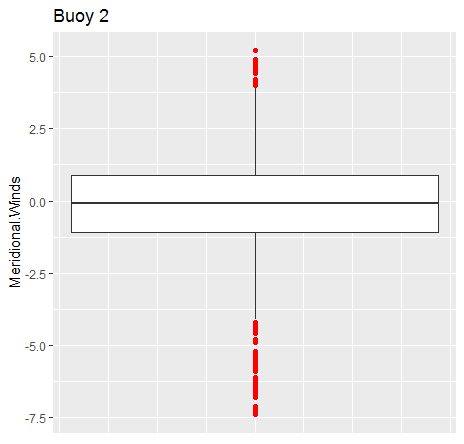


### Descriptive Plots

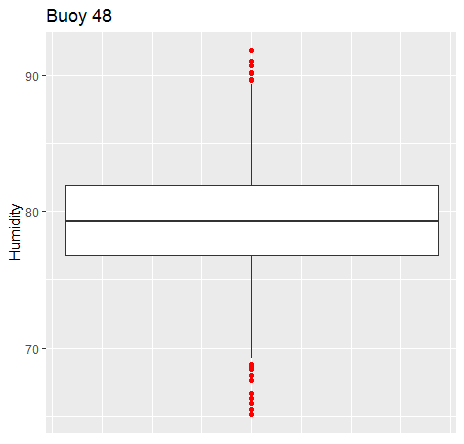
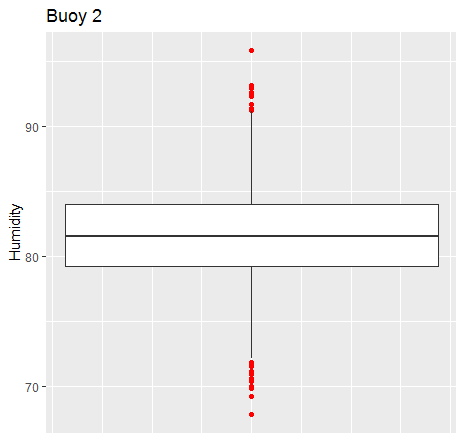
Zonal Winds



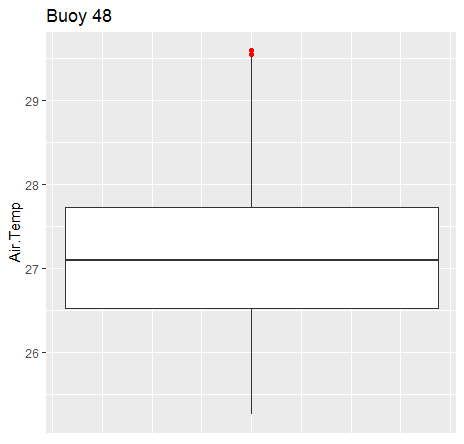
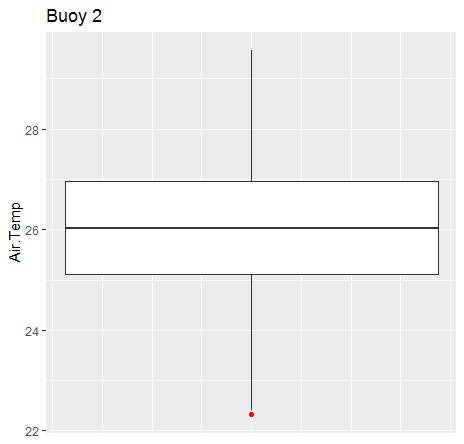
Meridional Winds



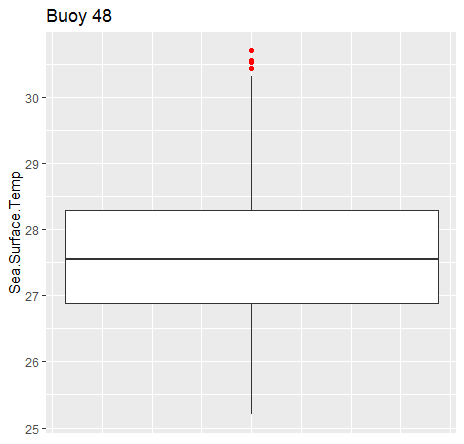
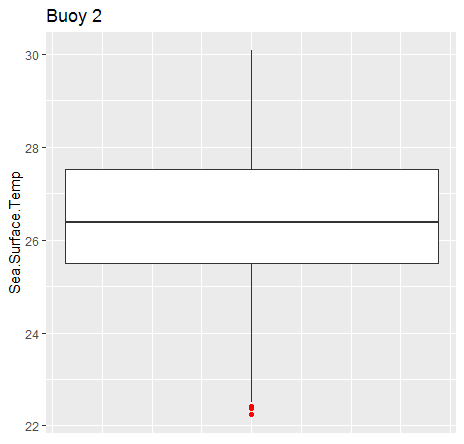
Humidity



Air Temp

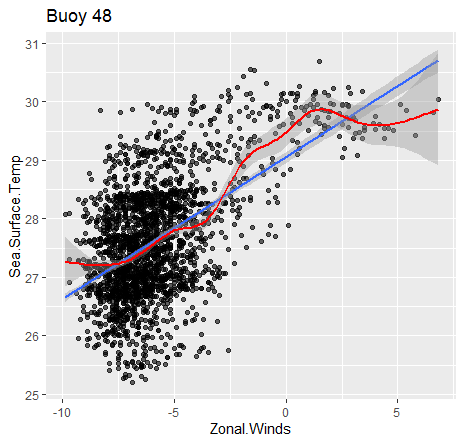
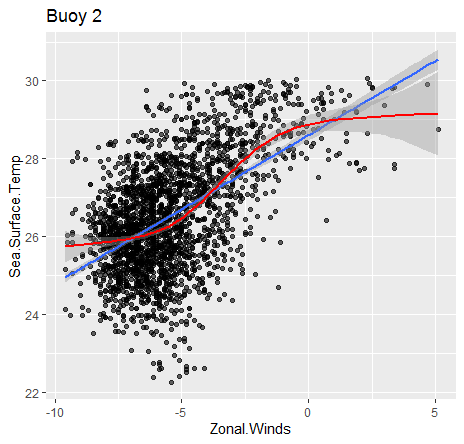


Sea Surface Temp

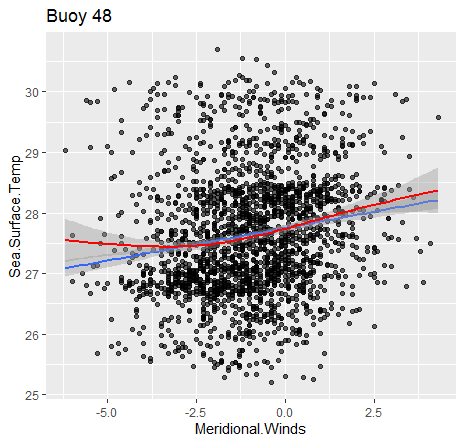
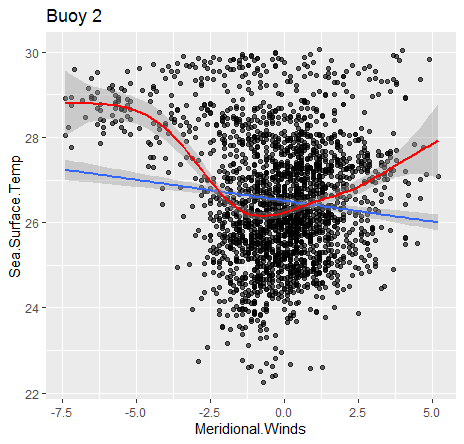


## Plots

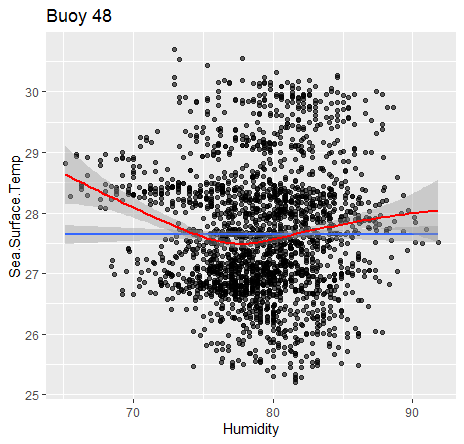
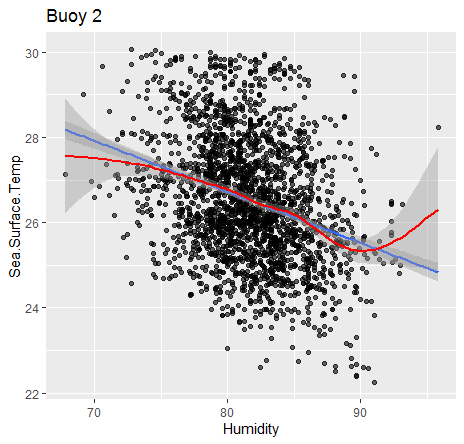
### Zonal Winds



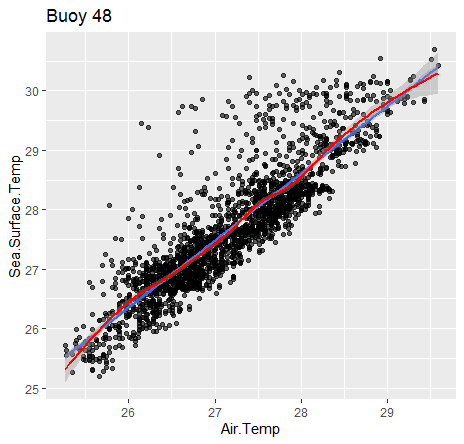
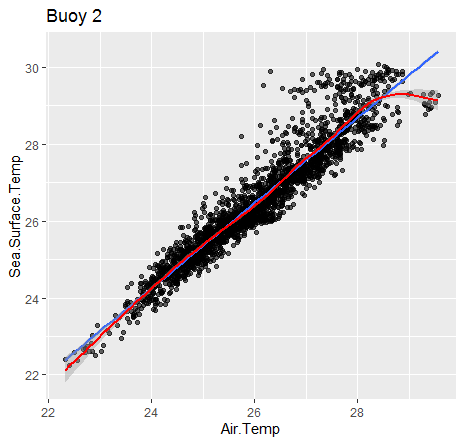
### Meridional Winds



### Humidity

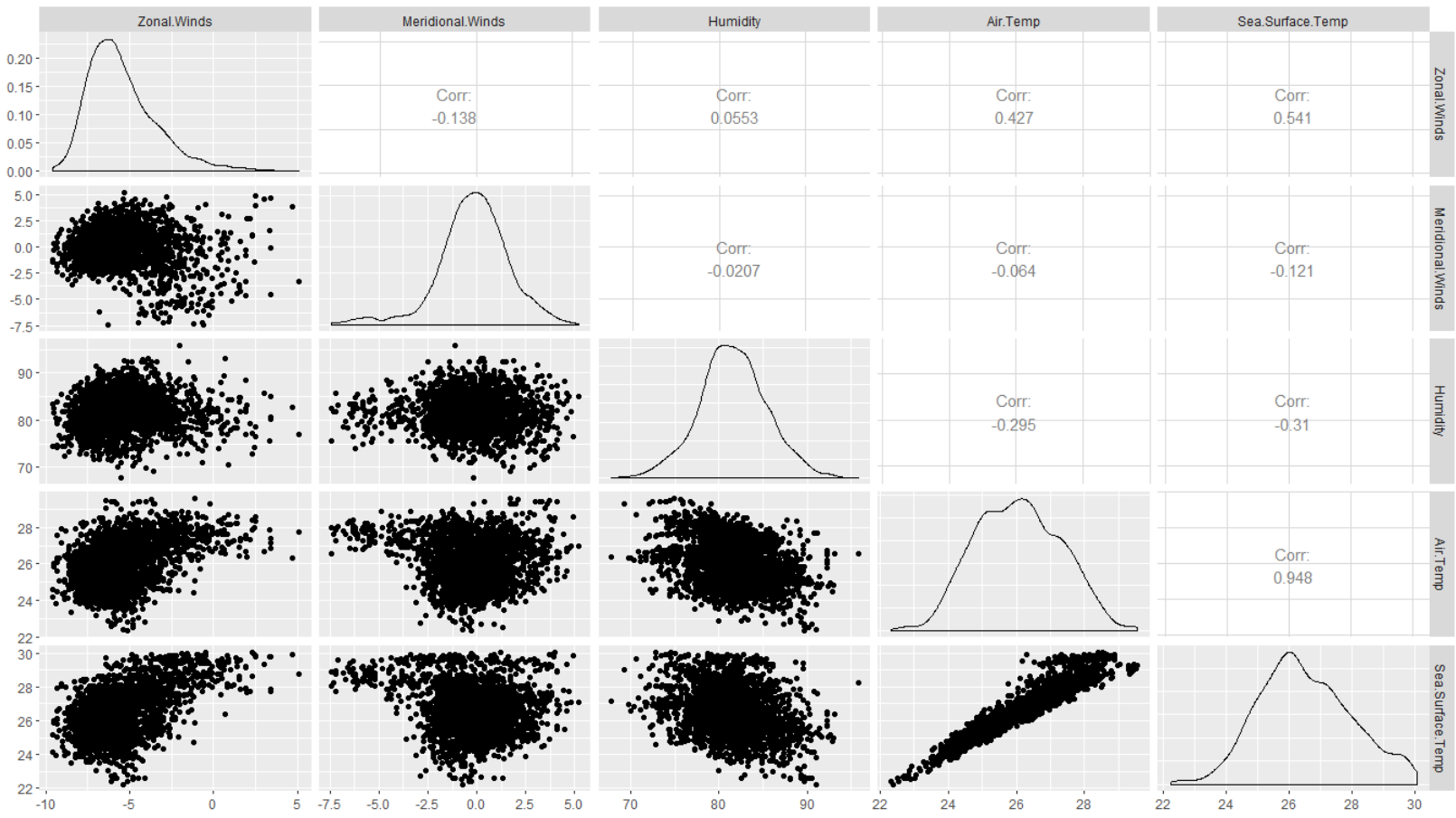


### Air Temp

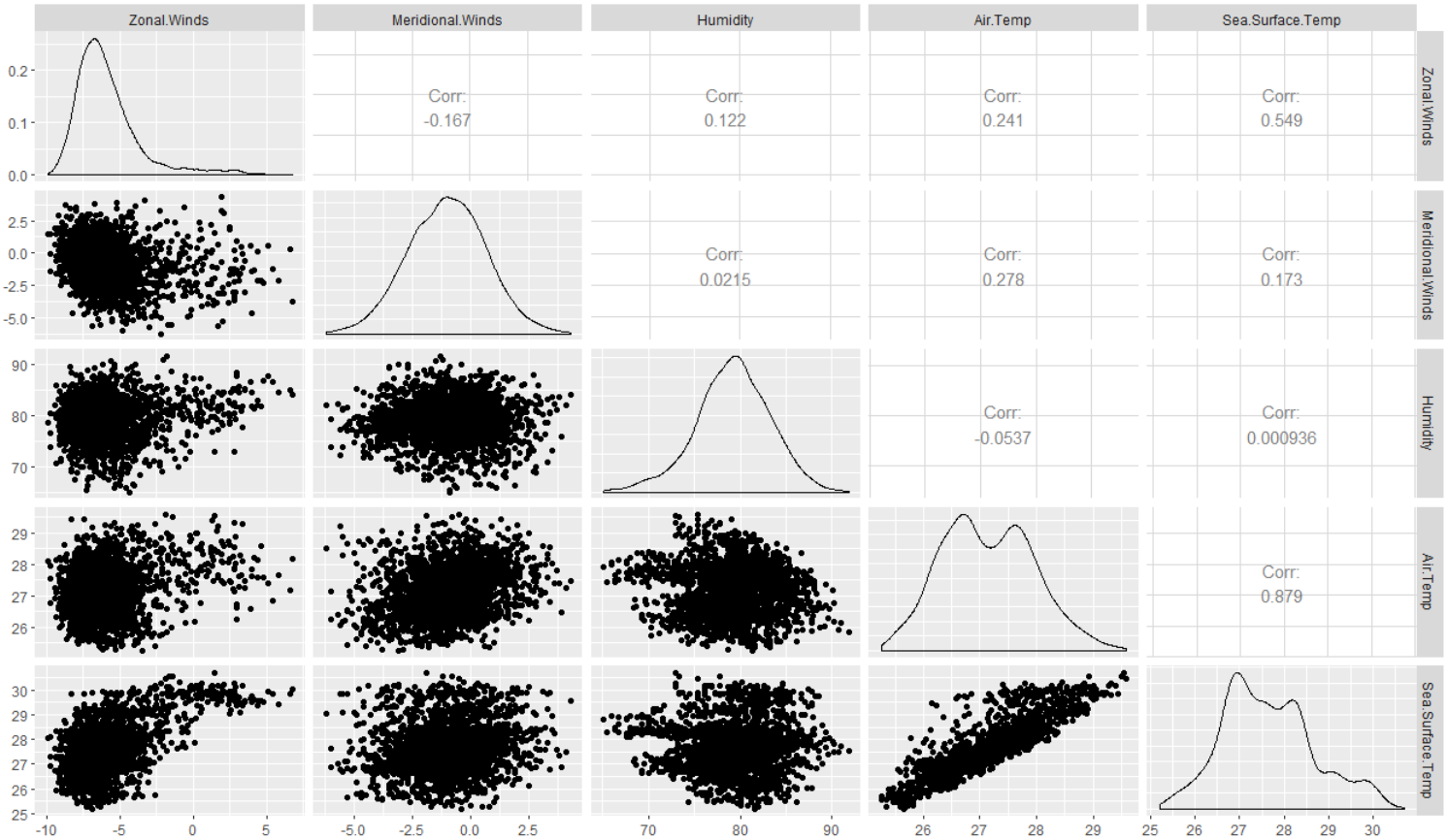


### Matrix Plot

Buoy 2

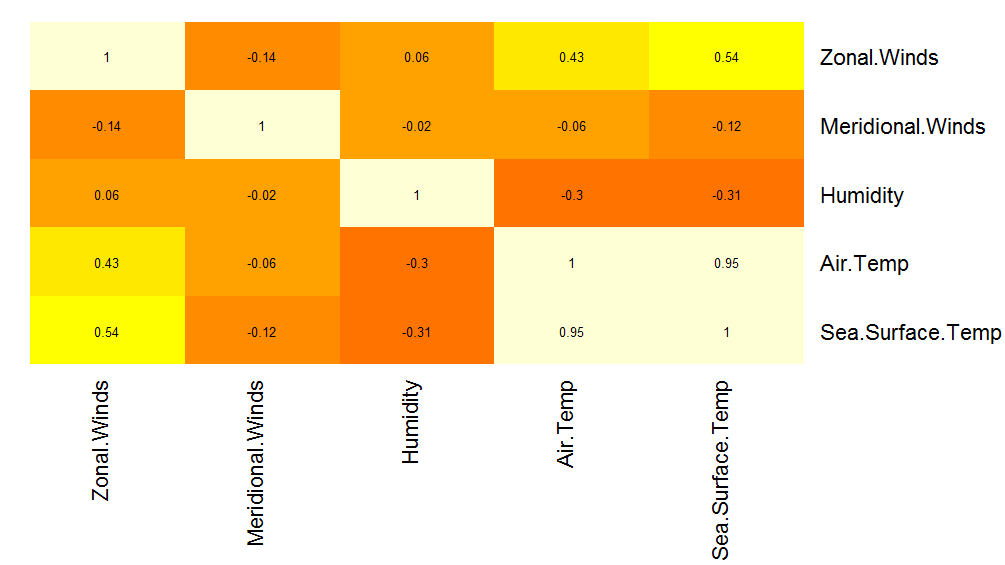


Buoy 48

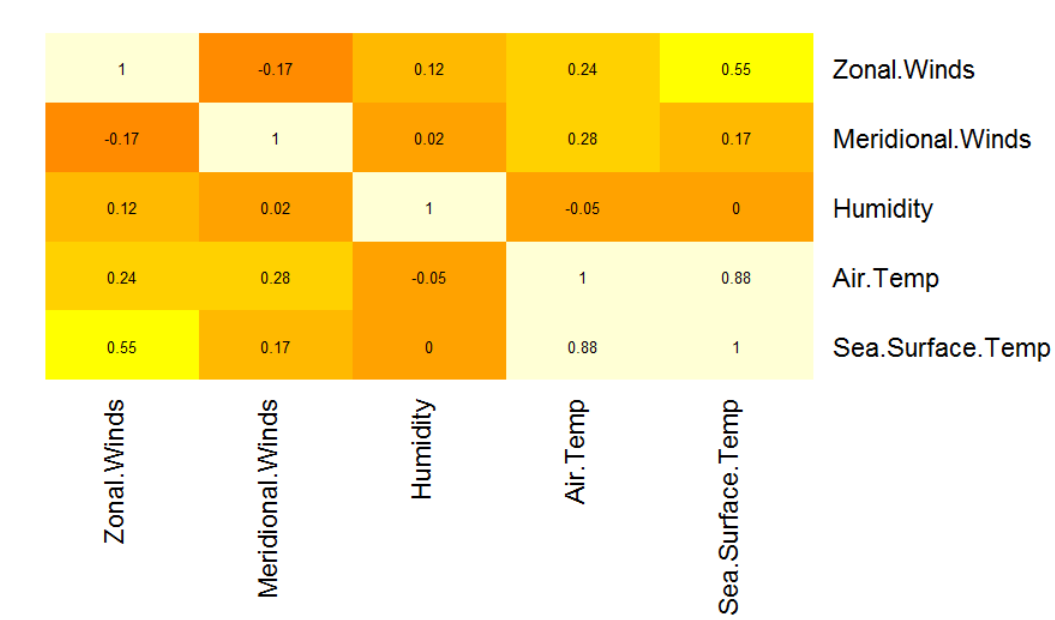


### Correlation Matrix

Buoy 2



Buoy 48

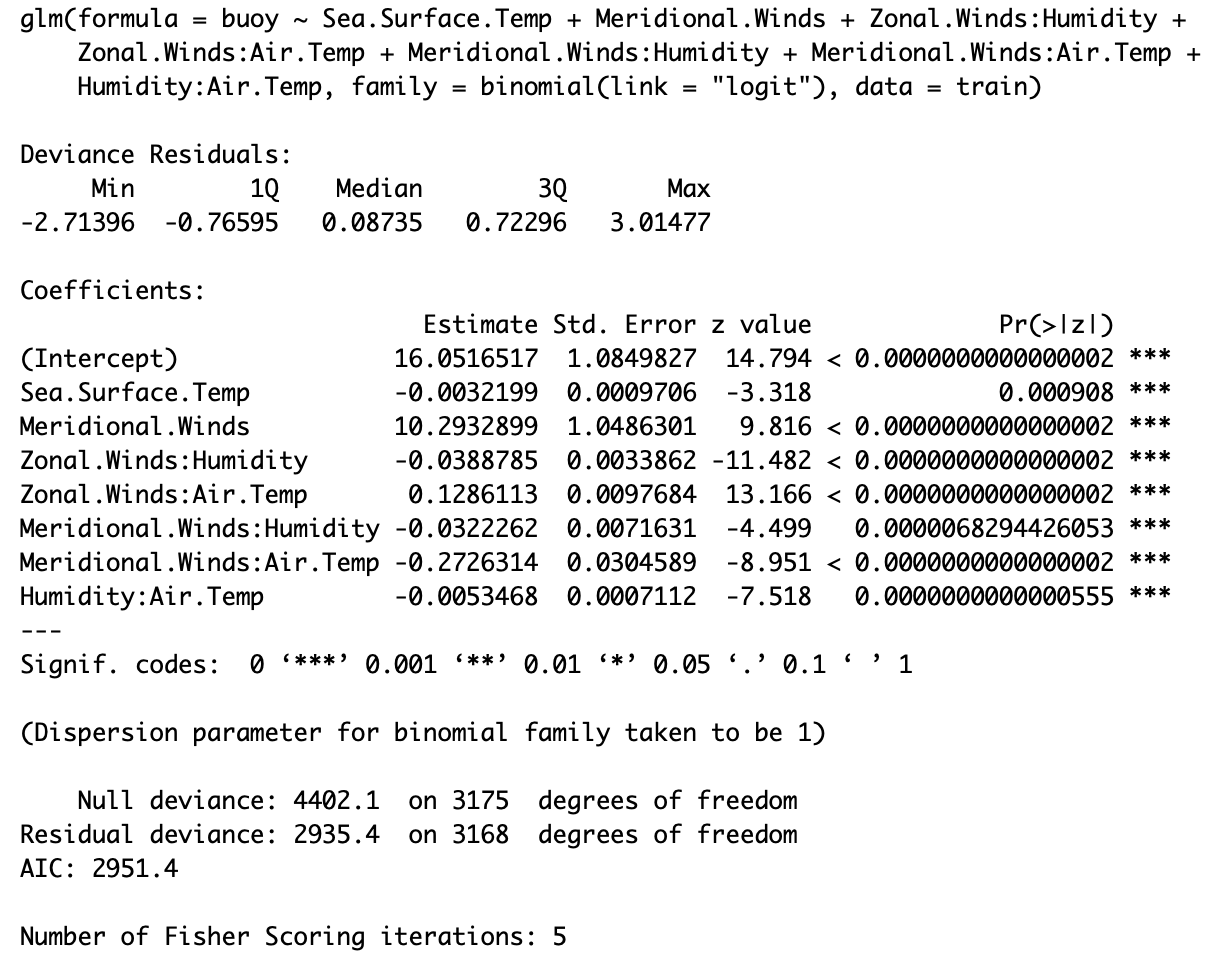


# **Logistic Regression**

Logistic regression is used to model dichotomous outcome variables, and it calculates the odds ratio, the likelihood that the event will occur. The odds ratio represents the constant effect of predictor variables, on the likelihood that the outcome will occur. We can only use logistic regression when we suspect that some certain data, missing the record of their buoy, belong to a certain buoy.

In the following example, we selected the data measured by Buoy 2 and Buoy 48, and used logistic regression to determine whether the data belong to Buoy 2 or not. First, we combined the data from Buoy 2 and Buoy 48, randomly selected 60% of them for training and 40% for validation. For the logistic regression, we ran the regression on “buoy” against “Sea.Surface.Temp”, “Air.Temp”, “Zonal.Winds”, “Meridional.Winds”, “Humidity”, and the interaction between these variables, which were all measured by the buoy and proven to be relevant to El Nino in our last assignment.

Here is the summary of the logistic regression:

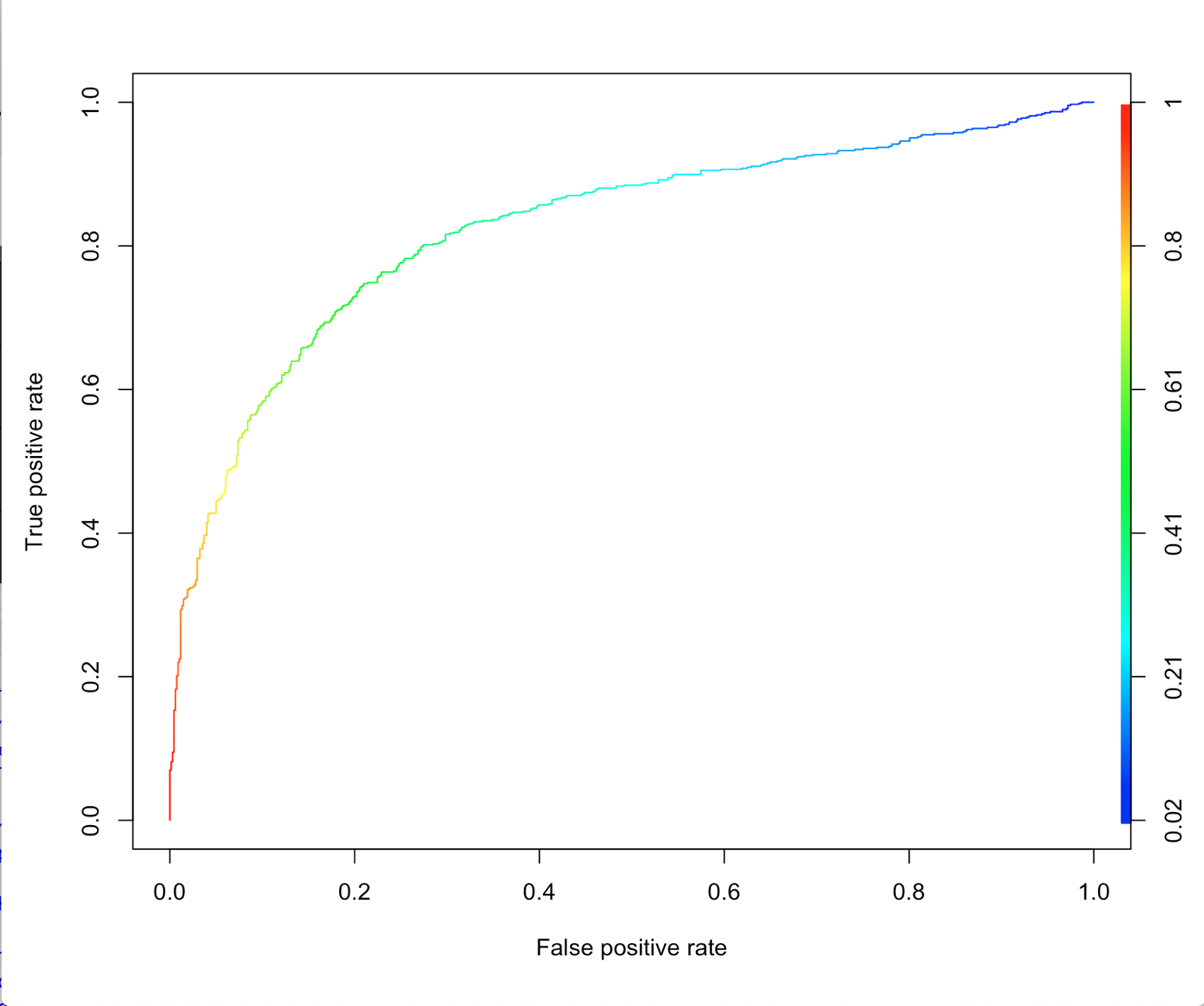
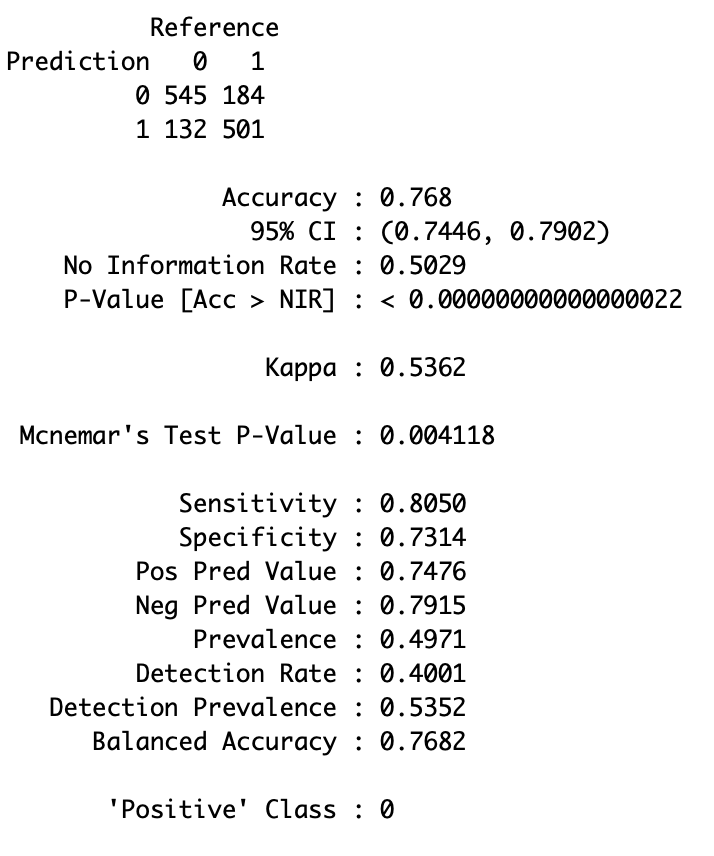


All p-values of variables are < 0.05. The regression is a fair fit to the classification so we used this regression to predict the buoy of the validation data.

Here is a sample of the outcome of the prediction. “1” in Column Actual means that the buoy is Buoy 2, and “0” (though not in this table) for not Buoy 2. In Column Predicted, the results are predicted as whether the data are from Buoy 2. If the predicted probability > 0.5, the buoy is predicted to be Buoy 2.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **actual** | **predicted** |  |
|  |  | 1 | 0 |  |
|  |  | 1 | 1 |  |
|  |  | 1 | 1 |  |
|  |  | 1 | 0 |  |
|  |  | 1 | 1 |  |

Here are the confusion matrix of the prediction and ROC curve:



The confusion matrix describes the performance of the logistic regression. We are specifically concerned with True Positive on the left up grid, the true prediction of positive cases; and True Negative on the right down grid, the true prediction of negative cases. By Observing the confusion matrix, we can calculate accuracy, the proportion of true prediction among all predictions; sensitivity, the proportion of observed positives that were predicted to be positive; and specificity, the proportion of observed negatives that were predicted to be negatives. Other two grids are both false predictions. ROC curve plots sensitivity, (True positive rate), against 1-specificity (False positive rate)

From the results we notice that the following:

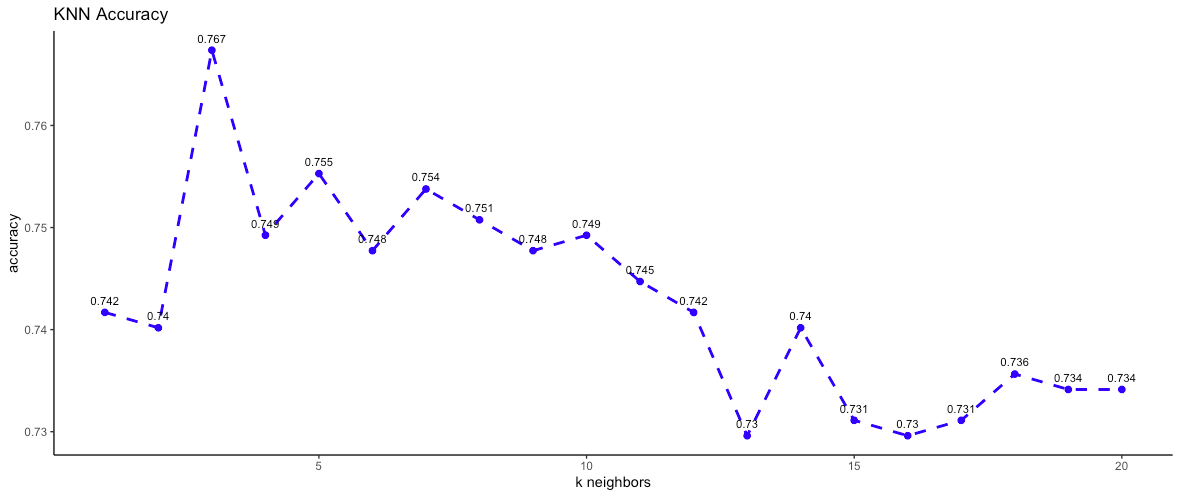
* Accuracy = 0.768
* Sensitivity = 0.805
* Specificity = 0.7314

Since the Accuracy, Sensitivity, and Specificity are high, the ROC curve is close to the left up corner, the logistic regression is effective in classifying buoys. We can also use this model to classify other buoys.

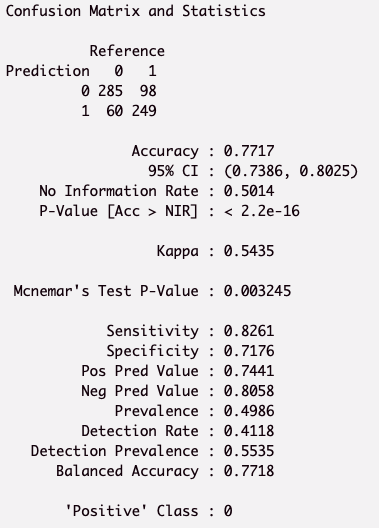
# **KNN**

We also used the K-Nearest Neighbor (KNN) algorithm for the classification of Buoy 2 or Bouy 48. KNN is a Supervised machine learning algorithm that stores all the available cases and classifies the new data or case based on feature similarity. It is mostly used to classify a data point based on how its neighbours are classified. Since there is no structured method to find the best value for *“K”,* we need to find out various values by trial and error and assuming that training data is unknown.

In this analysis, we increased the complexity of the model by increasing the number of predictors to include the initial six predictors, their squared and interaction terms leading to 21 predictors. We split the data into 70% training data, 15% validating data and 15% testing data. We then calculated the distance between two data points using the Euclidean distance method. Inorder to choose the best *“K”*, we run the KNN algorithm on the validating data for different value of *“K*” from 1 to 20. Below is a graph showing the accuracy (true positives and true negatives) derived from the confusion matrix for different values of *“K”.* The graph clearly shows that a *“K”* of **3** yields the highest accuracy of 76.73%.



Using a *“K”* = 3, we run the KNN algorithm on the testing data and this yields a marginally higher accuracy of 77.17%. With this result, overfitting is slightly minimized since the difference between validating and training data accuracy is rather marginal. This model neither underfits nor overfits. Below is a snippet showing the confusion matrix from this model.



From the results we notice that the following results:

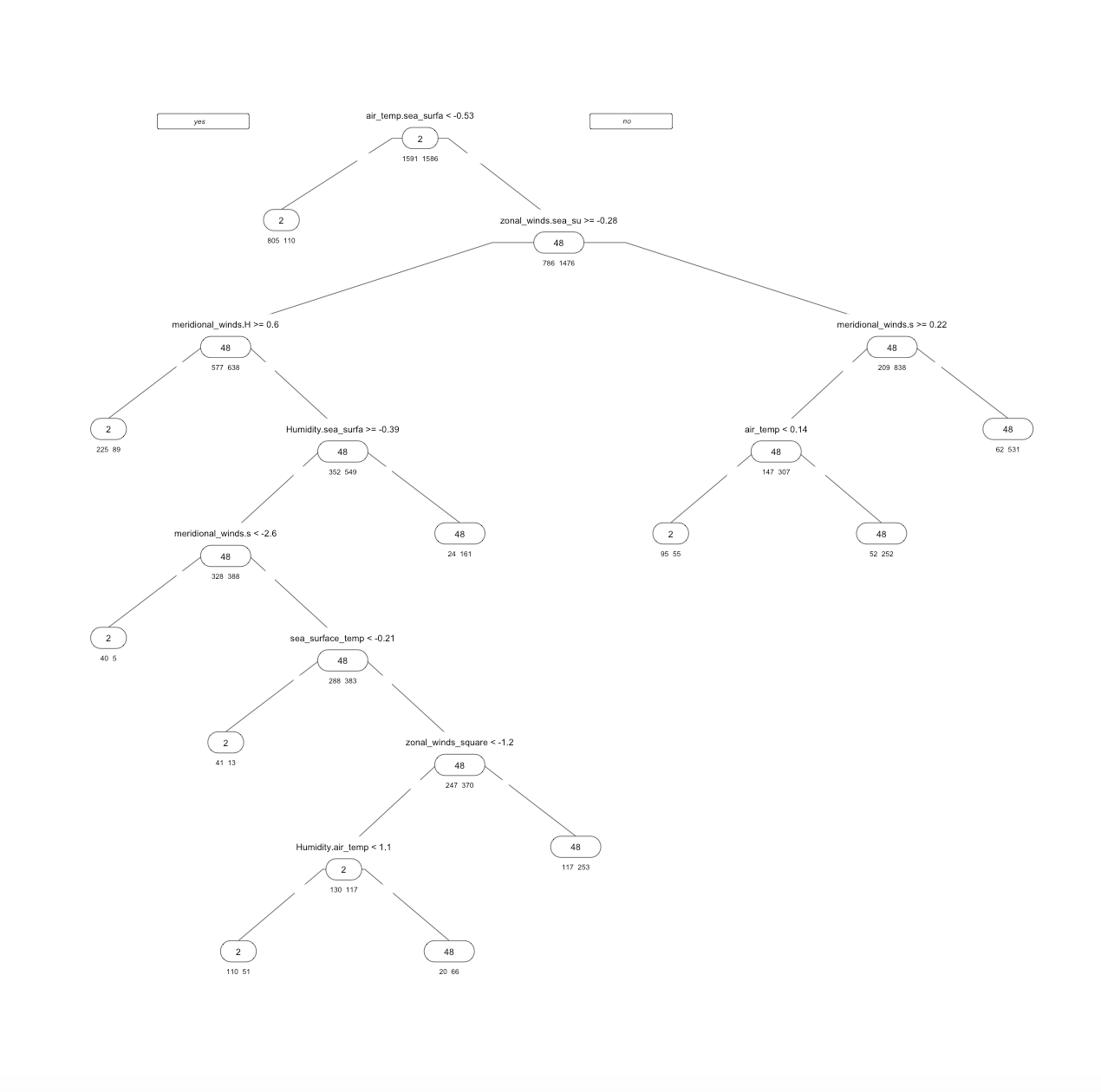
* Accuracy = 0.7717
* Specificity = 0.7176
* Sensitivity = 0.8261

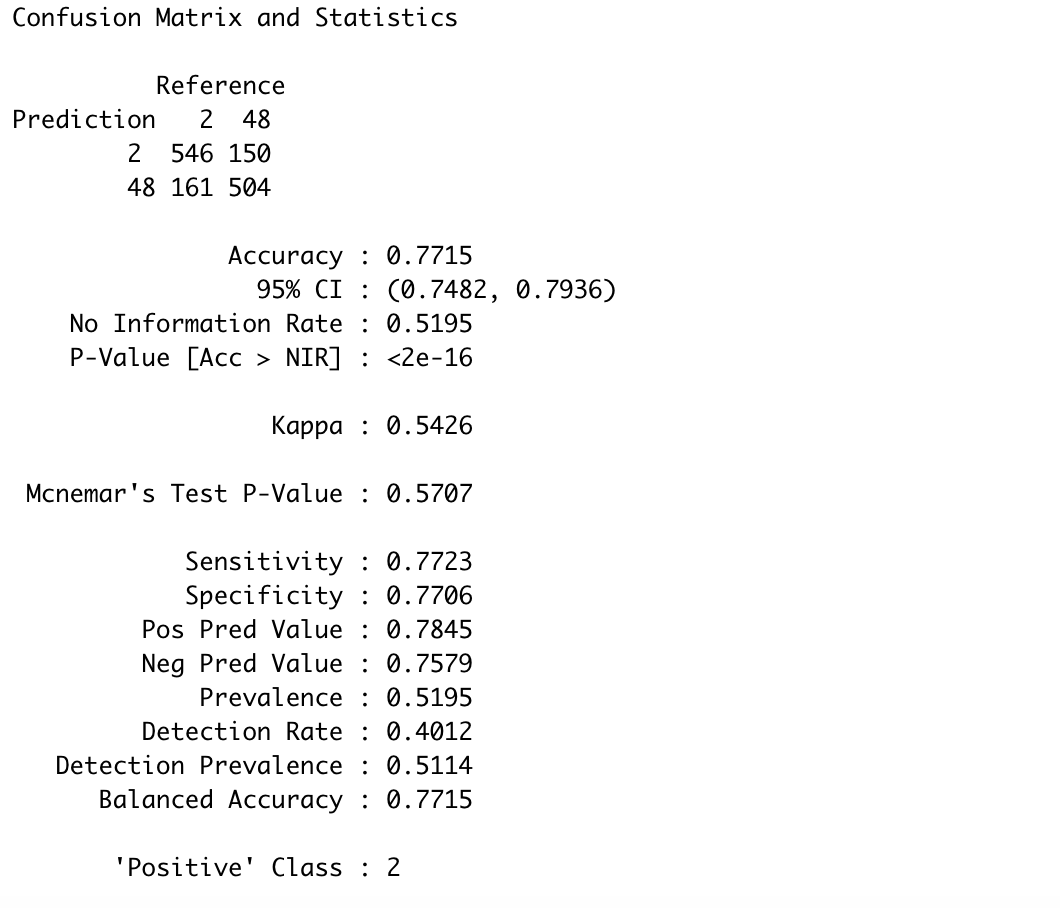
With KNN, the Accuracy is slightly higher than in logistic regression. Based on these results, we can conclude that KNN is as effective in classifying the buoys.

# **Classification Tree**

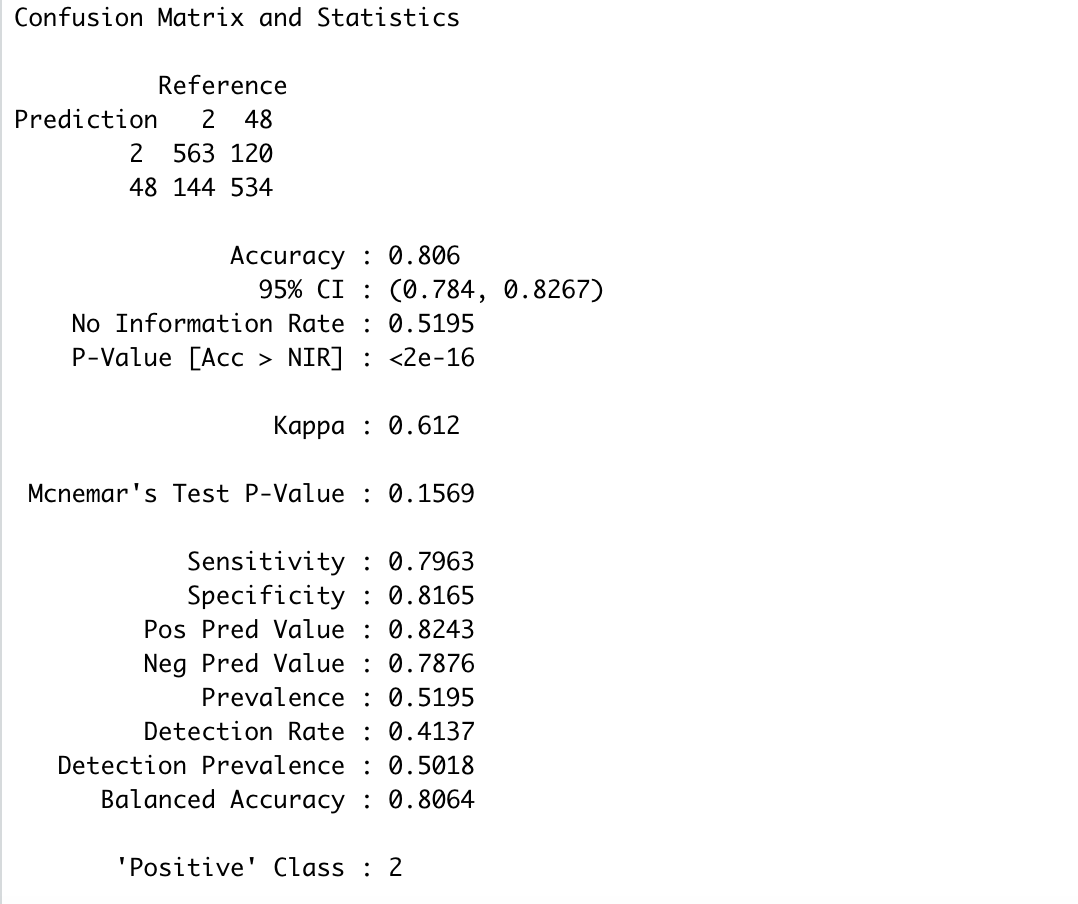
In this part, we used a number of classification tree methods to determine a certain data point belongs to buoy 2 or 48. In addition, on top of the 6 variables given in the dataset, we created interaction and squared variables. Moreover, we also created a variable called ‘month’ reflecting the month in which the data is recorded to capture possible seasonal effect of the time series data. We partitioned train and test data by 70% threshold.

We first used a simple tree to classify the two buoys. Many parameters in the rpart() function are set to default. The tree plot and confusion are shown below.



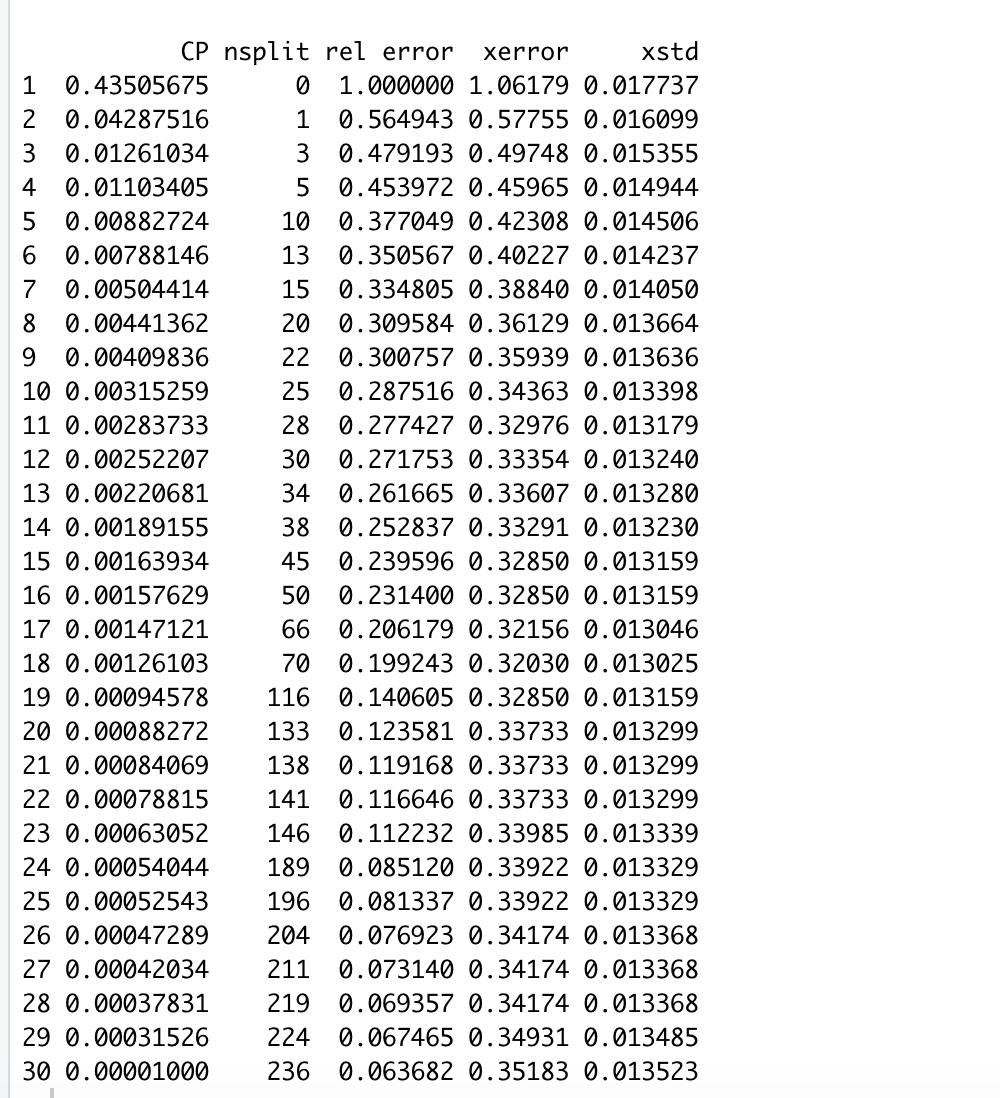


The tree looks simple, however, the result is not bad. The accuracy on test data of this model is 77.15%, while sensitivity and specificity are on a similar level. We would continue the experiment to more complex trees. In this case, we set cp = 0 and minsplit = 1 in the rpart() function and significantly increase the complexity of a single tree. The confusion matrix is shown below.

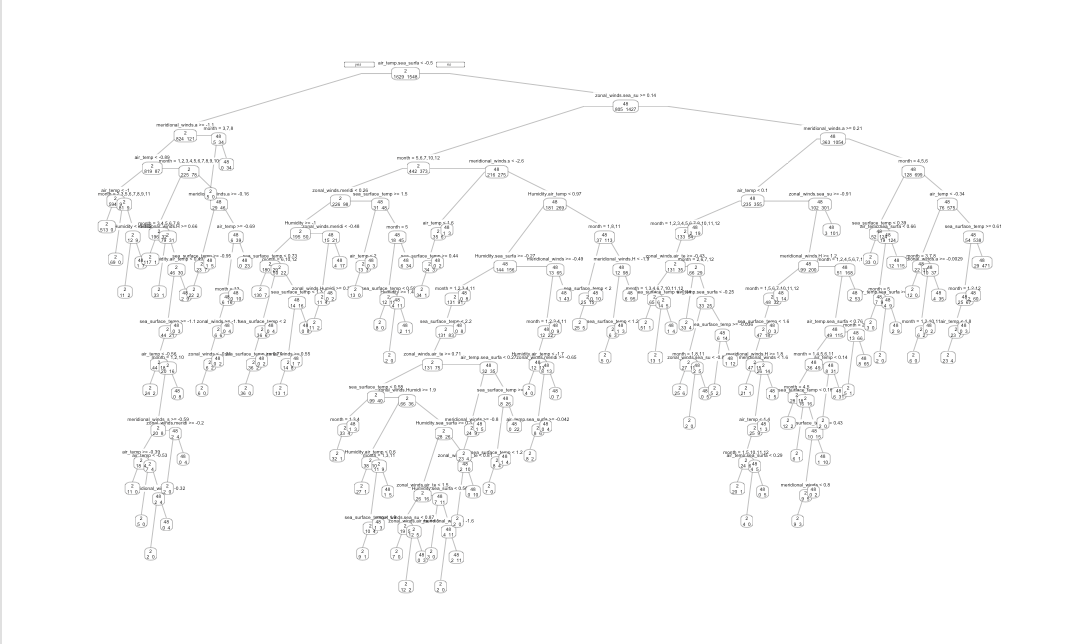


The accuracy increases to 80% consequently.

We then used the built-in cross validation method to find the optimal complexity and nsplit parameters for rpart() function. The table for errors is shown below.

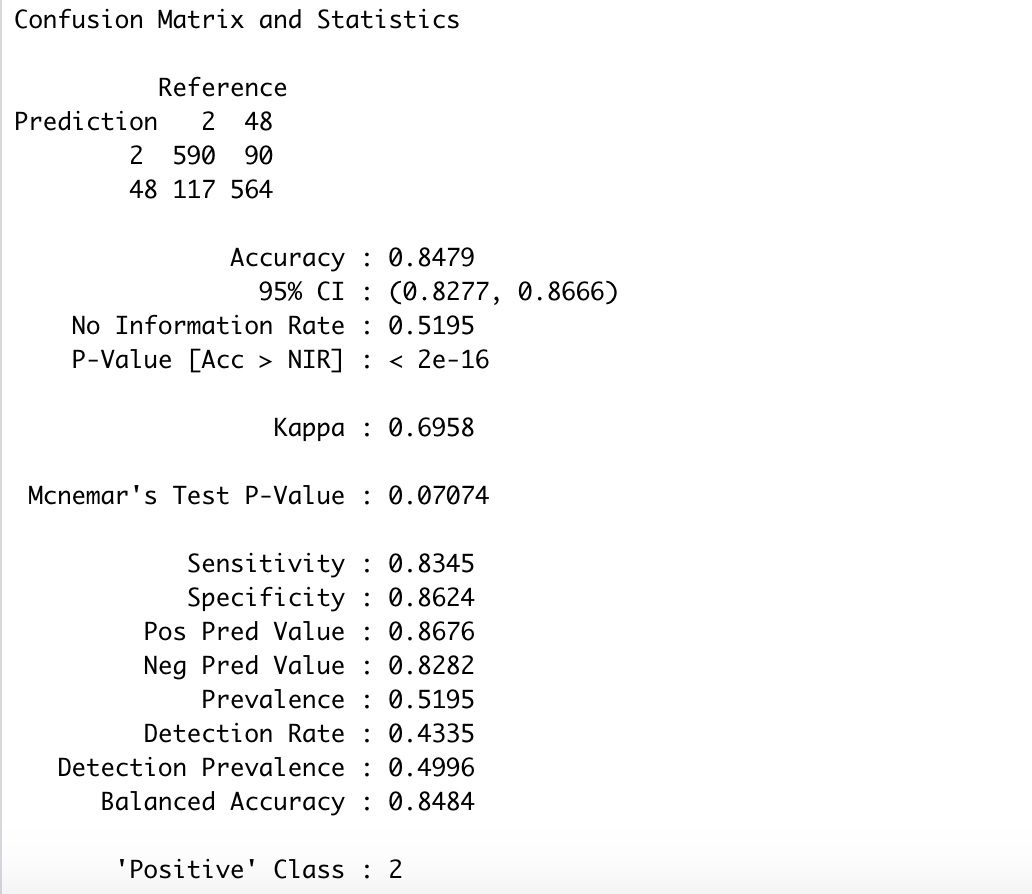


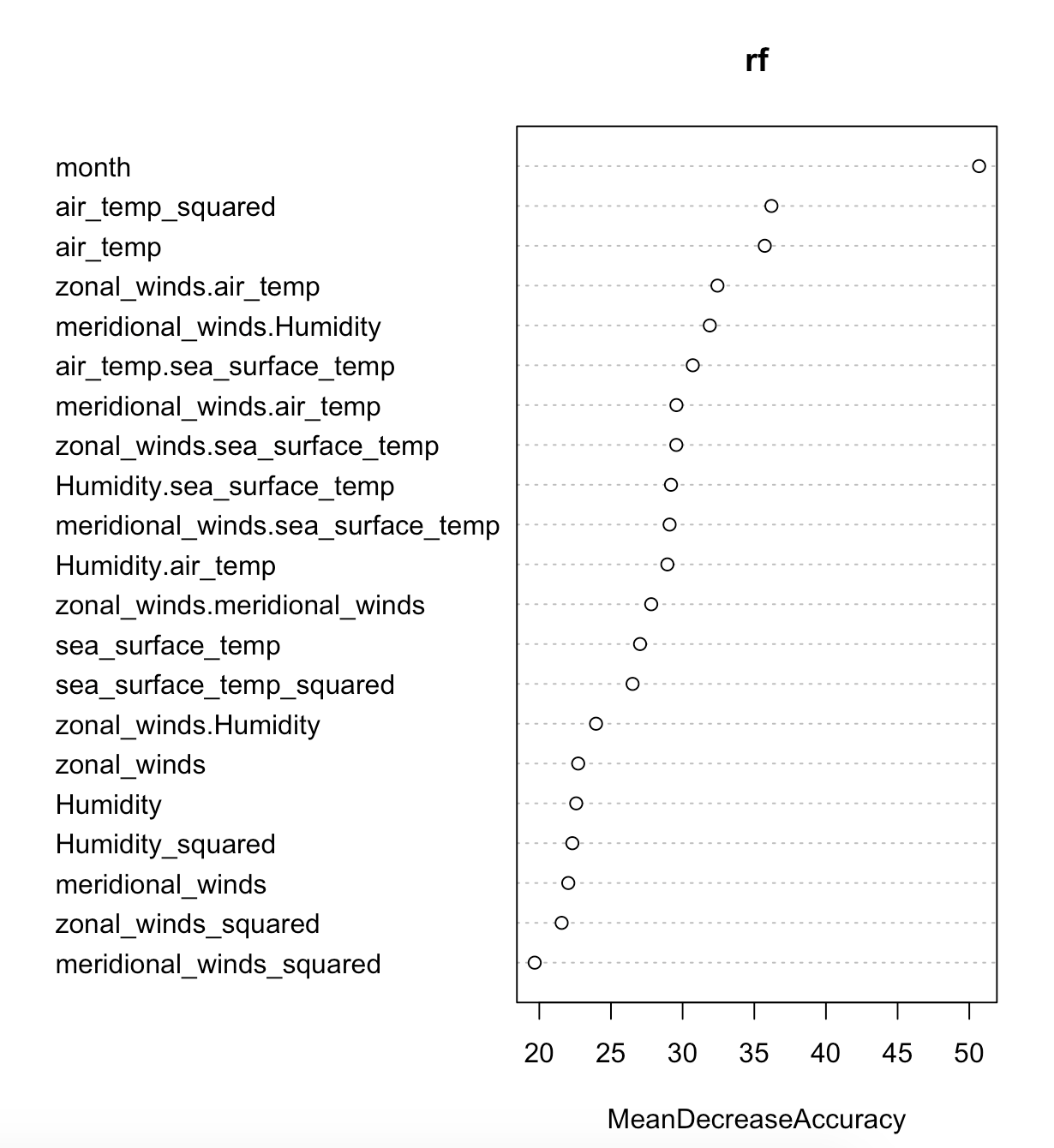
As shown in the table, the minimum xerror is at CP = 0.0013 and nsplit = 70. The R-Squared fitting score for this model is 1-0.20 ~ 80%. However, the tree plot looks a lot more complicated than the previous tree plot.



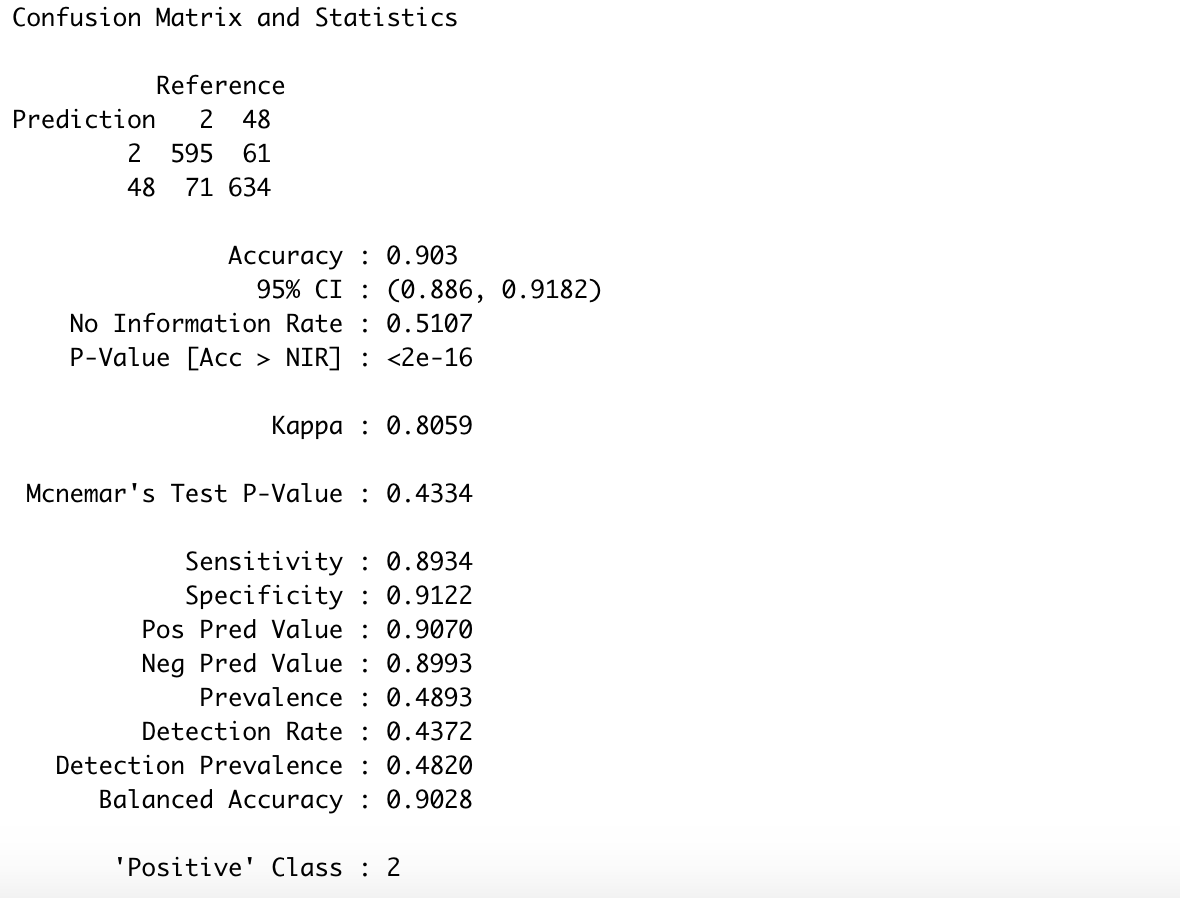
When we compare the results of the above three tree methods, we can see that in general, classification tree yields an accuracy of around 80%, and increasing complexity of trees does not significantly enhance the accuracy of the model.

Then we turned our sights to random forest and boosting tree methods. There are two tables shown below: the first table displays confusion matrix of random forest model’s predicting capability on test dataset; the second table shows features’ importance by mean decrease accuracy. Month turns out to be more significant than any other variables to predict the index of buoys. One interpretation is that time series data usually poses great seasonal effect, and capturing the seasonal effect usually enhances accuracy of models. The accuracy of the random forest method reaches 84.80%, better than that of a single tree.





Finally, we generated a boosting tree method to classify the two buoys. The accuracy is further enhanced to 90.3%.



# **Conclusion**

In this exercise, we applied logistic regression, KNN, and classification tree to classify two buoys. For each classification method, we used similar features, including variables provided in the dataset, squared terms and interactions among these variables; except in the decision tree part, we created and included a month variable. All these classification methods yield good enough results (accuracy > 75%) and set examples for real life using cases. In all of them, boosting tree method appears to have the best accuracy of more than 90%. However, this method is really computationally costly. It takes more than 1 minute for the model to be trained on the trained dataset containing only 3,000 data points. If we expand the model to include multiple buoys (more than 2) and more data points, we will need considerably more time to consolidate a boosting tree model. By that time, maybe a regular tree with enough leaves or other classification methods suffice to complete the task. Otherwise, if we already suspect that the data belongs to a certain buoy, we can choose to use logistic regression to investigate whether the data is really measured by that certain buoy or not.