Australian Rainfall Report

Hanna Damarjian

12/13/2021

I. Introduction.

Meteorology is the study of the atmosphere, particularly the conditions of the weather. There are many factors that contribute to predicting if it will rain tomorrow including the current day's atmospheric pressure, humidity, temperature, wind speed, and direction. This project emphasizes weather as I determine what factor(s) contribute to predicting if it will rain (tomorrow) in Australia. The output is the binary classification for weather - 0 for no rain and 1 for rainfall. There are nearly 24 input variables contained in the Kaggle data set, and several machine learning techniques are emphasized to determine the optimal model that can be used for the test data.

II. Methodology.

A. Exploratory Data Analysis.

When initially analyzing the data on Kaggle, it appears that multiple variables have missing entries. When imported onto R, there appears to be 34,191 rows, 24 columns $\rightarrow$ 820,584 entries total. There is also a total of 112,769 missing entries so nearly 86.26% of the data is filled. When analyzing the variables that contain NA, I decided to omit the variables **evaporation** and **sunshine** for two reasons: (1) each random variable has over 90% of missing entries and (2) from the introduction these two factors do not seem to be heavily involved with predicting the weather for the following day (in general). Unless there is a heat wave (but the definition of sunshine does not imply heat wave), but this did not impace my choice. I also deleted the two variables **cloud9am** and **cloud9pm** because they contain nearly 30-50% of missing entries. These two variables were not clear and I felt there were other variables that provided more meaningful information for the classification problem. Hence, they were omitted. Prior to imputation, I did two steps: (1) I converted every character variable into a factor variable prior to imputation. This would allow the imputation to recognize that if a variable is a factor, then the missing entries filled would be labeled as levels of the factor and not characters (or strings). (2) I analyzed the top five missing variables (which happened to be numeric). I wanted to analyze the relationship amongst these missing entries and created an aggregate which functions as accumulating the number of missing entries per row. The top five missing variables are: **pressure9am, pressure3pm, winddir9am, windgustspeed, and windgustdir.**

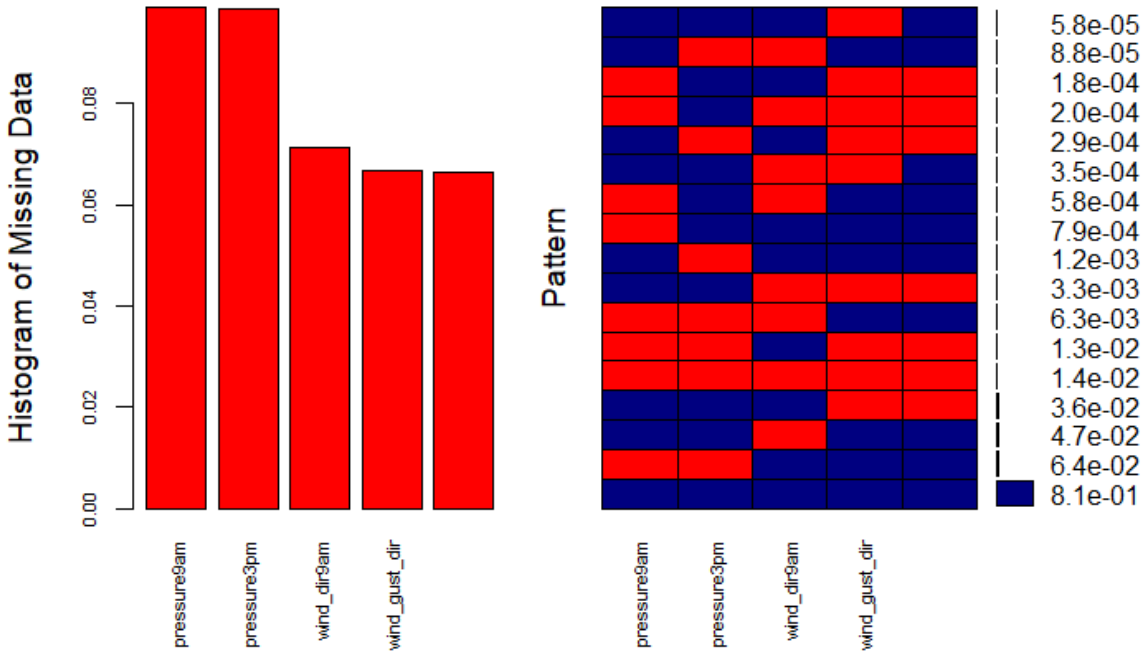

Figure 1: Top Five Missing Entries

For example, it seems like there was nearly 81% filled entries for all five variables and 6.4% missing entries for both **pressure9am** and **pressure3pm**. From here, my last step in EDA was to perform a **Multivariate Imputation by Chained Equations** (or mice) using the random forest method on R. I used random forest because the missing entries were of both categorical and numerical.

B. Feature Selection and Model Building.

After imputation, I re-uploaded the data set and skimmed it using **skim()**. This allowed me to verify that the uploaded data was not being read correctly. That is, R interpreted my factor variables as character variables (strings), so they had to be converted into factors. Now, with the whole data set filled, I decided to perform an additional step that could have been done in EDA − change the data column by month. That is, the original variable was recorded as y/m/d, so I decided to use the **lubridate** package in order to change the date variable into a numeric variable such that the random value was the month. The reason I did this was because I thought that the month of the year could be of significance when predicting if it would rain tomorrow. An additional note is that since Australia has different climates (some wetter or dryer than others), I suspected that the month of the year would be better to consider compared to using the specific date. The final data touch-up I made was converted all of the factor variables into dummy variables (e.g. **location, windgustdir, winddir3pm, winddir9pm**). This would aid with the feature

selection and model selection.

1. First Model - Binomial Logistic Regression.

The first step when considering this model was to analyze the significance of all of the variables. Thus, I first built a binomial regression model including all variables. From there, I ranked the significance of each variable according to the (1) p-value and (2) VIF (see table on **Results**). I picked the top seven variables that were highly significant when considering all variables in the model. I then created the trained model and perform $9-$fold cross validation because the sample size (number of rows) is $n = 34,191$ is divisible by 9. Thus, for each fold, I had 8 training data sets and 1 tested. For each of 9 combinations, I built a model that first included the most significant variable $\rightarrow$ created that trained model $\rightarrow$ found the tested classification rate. Then, I repeated this for the two most significant variables, and then the next three most signficiant, and so on. The top seven variables I chose and the training classification rate (correct percentage classified) graph are shown under **Results**, respectively.

2. Second Model - Random Forest.

Similar to the first model, I did the same process. Only three variables that were significant in the binomial logistic regression model were also significant in the random forest (**humidity3pm, location, and pressure3pm**). The table and graph (similar to 1.) are shown under **Results**.

3. Third Model - Support Vector Machine (SVM).

I tried to look into alternative functions to train a model using the support vector machine method, however, the R software I have downloaded could not take two fast svm functions $-$ liquidSVM and selectSVMs $-$ to perform classification. Thus, I used the same **svm()** from this course. It was very slow, and I only decided to focus on the three significant variables that performed well under Binomial Logistic Regression and Random Forest. The graph is shown under results.

III. Results.

A. Binomial Logistic Regression.

Variable	VIF
Humidity 3 PM	35.92804
Wind Gust Speed	28.700968
Pressure 3 PM	14.6315
locationMountGinini	10.99914
Wind Speed 3 PM	10.807768
locationWollongong	9.05221770
Rain Today	7.909935

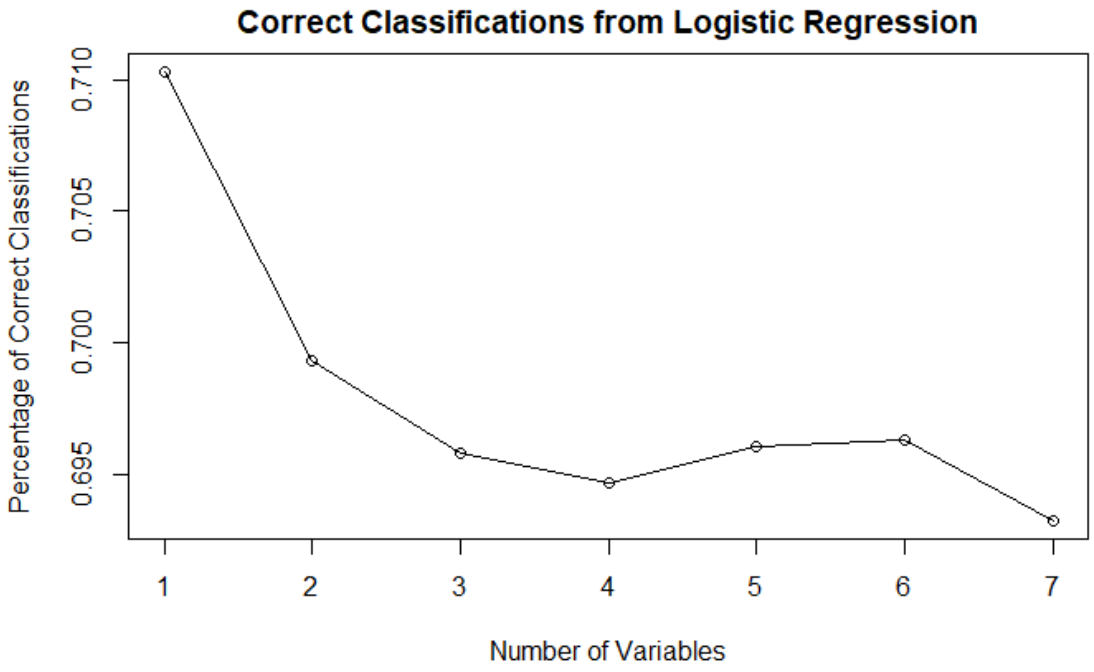

Figure 2: Classification Rate under Binomial Logistic Regression

As mentioned in the **Methodology** section in detail, I considered the VIF for the top seven most significant variables and created a table ranking those values. The higher the VIF, the more significant and highly contributable variable in the model.

From the correctly classified rate graph, it appears that the best binomial regression model includes only the one-most significant variable **humidity 3 PM** with classification rate at about 71%.

B. Random Forest.

The **Variable Importance Plot** is a function that utilizes the random forest variables and ranks

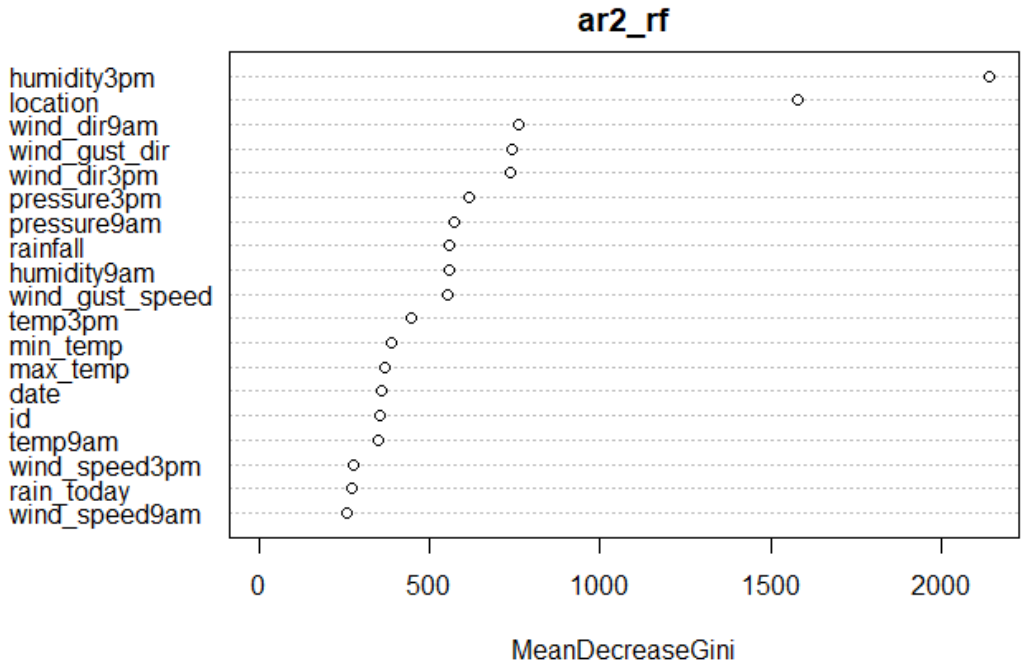

Figure 3: VIF under Random Forest

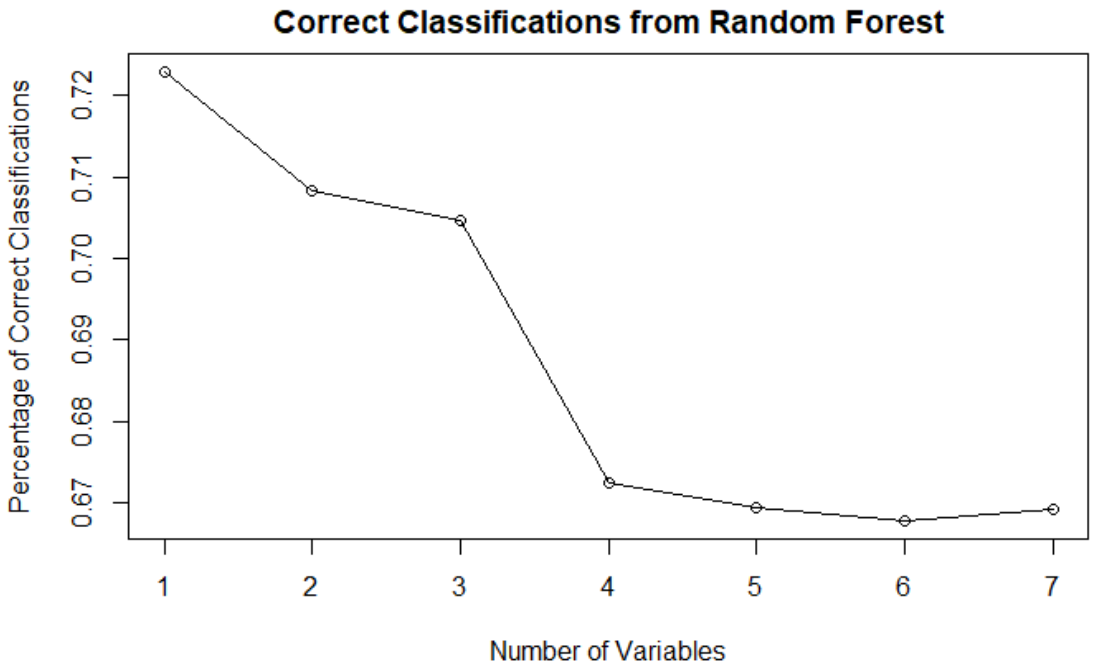

Figure 4: Classification Rate under Random Forest

them according to the MeanDecreaseGini value. The higher this value, the more significant the variable was. For example, **humidity 3 PM and location** were the two most significant variables from this graph. Also, it appears that the best random forest includes only the one-most significant variable **humidity 3 PM** with classification rate at about 73%.

C. Support Vector Machine (SVM).

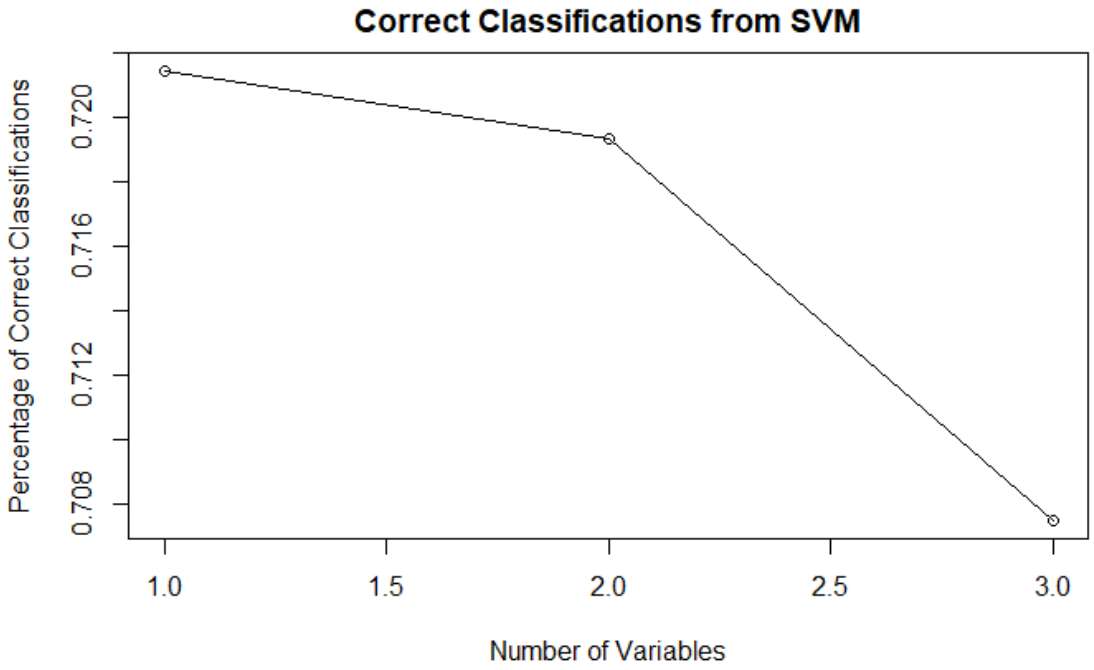

Figure 5: Classification Rate under Random Forest

It appears that the best random forest includes only the one-most significant variable **humidity 3 PM** with classification rate at about 73%. Note that the only variables tested were **humidity 3 PM, pressure 3PM, and location**. This came from taking the intersection of the top seven significant variables according to the first two models.

IV. Conclusion.

Out of the three classification models I have depicted, I chose to use **Random Forest** with the **humidity 3pm** variable alone for the tested data. It was because of the less run time compared to SVM even if both had nearly the same correct classifcation rate. When I tested the random forest using the data on Kaggle (test.csv), the log loss value was 0.69314, so I believe my model was at the benchmark but unsuccessful at being a good predictive model. I am not surprised with how significant humidity can be

when predicting if it would rain tomorrow in Australia. The location (again due to higher/lower elevation and drier/cooler/colder climates) and atmospheric pressure are also significant factors. I am happy to be able to learn and take away important information from these methods!

Future Direction:

- Consider imputing the two dropped variable **cloud 3 PM and cloud 9 PM**. I would like to see how significant these two variables are as well.
- Consider a different mice method. Possibly, consider imputing each variable (with missing entries) using a different method. This depends on the distribution (e.g. normal, skewed?) of that numeric random variable or testing other alternative mice methods for a categorical variable. I tried several other mice methods (for example, I used KNN and lasso), but they were unsuccessful. I am not sure why, but this is a great question to further research as I want to improve my accuracy.
- Perform Recursive Backwards Elimination due to the amount of variables (including dummy variables) that were contained in this model. This can be a better method to selecting the optimal variables. Another method I would try to improve this project would be univariate selection (i.e. building all possible univariate models and selecting the top seven most significant according to the p-value). Researching and learning more feature selection methods would be a good idea.
- Consider other machine learning models for classification such as KNN and LDA (or QDA).

V. Appendix. All of my code was commented to easily compile this document. My PC is very slow when it comes to compiling code under SVM and random forests.

```
# ar <- read.csv("C://Users//johnd.LAPTOP-35N364TU//OneDrive//Desktop//Predictive Analytics//train.csv")
# # View(ar) --> all columns included from Kaggle
#
#
# # Methodology:
#
# # 1. Exploratory Data Analysis:
# # A. Dataframe
# nrow(ar) # 34,191 rows
# length(ar) # 24 columns
# total_entries <- nrow(ar)*length(ar)
# total_entries # 820,584 total entries
# # summary(ar) --> summary statistics for each variable
# library(skimr)
# # skim(ar) --> shows the data type, mean, sd, and hist for each variable
#
#
# # B. Missing values (NA):
# sum(is.na(ar)) #  112,769 missing entries
# percentage_of_filled_entries <-
#   100*(total_entries-sum(is.na(ar)))/total_entries
# percentage_of_filled_entries # about 86.26% of cells are filled.
#
# # Which variables (columns) contain NA?
# # What is the percentage of NA's for each variable?
#
# colSums(is.na(ar))/nrow(ar)
# # I will omit evaporation and sunshine as % of missing values is close to 100.
#
#
# # "cloud9am" and "cloud3pm" are the only variables where 0.1 < x < 0.90.
# # I will omit these variables as well. Why?
#
# # Drop the two variables:
```

```
# ar <- subset(ar, select=-c(evaporation,sunshine,cloud3pm, cloud9am))
# length(names(ar)) # ensure the four variables are dropped
#
#
# # C. Change the string variables into factors:
#
# # Change all character variables into factor variables:
# # Use skim(ar) to determine which string variables needed to be factored:
# # skim(ar)
# ar$location <- as.factor(ar$location)
# ar$wind_gust_dir <- as.factor(ar$wind_gust_dir)
# ar$wind_dir9am <- as.factor(ar$wind_dir9am)
# ar$wind_dir3pm <- as.factor(ar$wind_dir3pm)
# ar$rain_today <- as.factor(ar$rain_today)
# ar$rain_tomorrow <- as.factor(ar$rain_tomorrow)
# skim(ar)
#
# # D. Imputation (MICE):
# # Imputating the numeric variables using "" method:
# library(mice)
#
#
# ar2 <- ar # keep an extra copy of original dataset to compare
# #arNEW <- subset()
# library(VIM) # used for aggregate plot
#
# # Top five variables with missing data are
# # (1) pressure9am, (2) pressure3pm, (3) wind_dir9am,
# # (4) wind_gust_speed, and (5) wind_gust_dir.
# top_5_me <- subset(ar, select=c('pressure9am', 'pressure3pm',
#                                 'wind_dir9am', 'wind_gust_speed',
#                                 'wind_gust_dir'))
# aggr_plot <- aggr(top_5_me, col=c("navyblue","red"), numbers=TRUE, sortVars=TRUE,
#                   labels=names(top_5_me), cex.axis=0.7, gap=3, ylab =
#                     c("Histogram of Missing Data","Pattern"))
# # Explain.
#
# # Finally, impute using random forest (what else could I have done differently?):
#
```

```
# imputed_ar <- mice(ar, m=3, method="rf")
# # View(complete(imputed_ar)) check out the dataset and compare to original "ar"
# finished_imputed_ar <- complete(imputed_ar,1)
# dim(finished_imputed_ar)
# sapply(finished_imputed_ar, function(x) sum(is.na(x)))
#
# write.csv(finished_imputed_ar,
#           "C:/Users/johnd.LAPTOP-35N364TU/OneDrive/Desktop/Predictive Analytics/fi_ar.csv")
#
# # Australian Rain dataset after imputation:
# ar <- read.csv("C:/Users/johnd.LAPTOP-35N364TU/OneDrive/Desktop/Predictive Analytics/fi_ar.csv")
# sum(is.na(ar))
#
# library(skimr)
# skim(ar)
#
# # 1. Factorize the variables:
# ar$location <- as.factor(ar$location)
# ar$wind_gust_dir <- as.factor(ar$wind_gust_dir)
# ar$wind_dir9am <- as.factor(ar$wind_dir9am)
# ar$wind_dir3pm <- as.factor(ar$wind_dir3pm)
# ar$rain_today <- as.factor(ar$rain_today)
# ar$rain_tomorrow <- as.factor(ar$rain_tomorrow)
# skim(ar)
# View(ar) # omit the first column, X
#
# # 2. Omit the row-counter column:
# ar2 <- subset(ar, select=-c(X))
# View(ar2)
#
#
# # 3. Change the date column to specify the month only:
#
# library(lubridate)
# ar2$date <- month(as.POSIXlt(ar2$date, format="%Y-%m-%d"))
#
#
# # 4. Create dummy variables:
# library(fastDummies)
```

```
# ar2_dummy <- dummy_cols(ar, select_columns = c('location','wind_gust_dir','wind_dir3pm','wind_dir9pm',
#                       remove_selected_columns=TRUE))
#
#
#
#
#
#
# # 5. Feature Selection by building several machine learning models.
#
# # A. Binomial Logistic Regression:
# ar2_glm <- glm(rain_tomorrow ~ ., data=ar2, family="binomial")
# summary(ar2_glm)
#
# # We can use the "caret" package to determine feature importance.
# library(caret)
# varImp(ar2_glm)
#
# # Next steps:
# # a. Create subsetted data for the 4-7 variables and create ML code for classification.
# # b. Create a graph for each number of significant variables and how many trained observations
# # were classified correctly.
#
# library(fastDummies)
# ar2_var <- subset(ar2_dummy, select=c(rain_tomorrow,humidity3pm,
#                       wind_gust_speed, pressure3pm,
#                       location_MountGinini, wind_speed3pm,
#                       location_Wollongong, rain_today))
#
# set.seed(123)
# var_count_per_cc <- c() # for i number of variables
# per_cc <- c() # for j training sets, we have the number of correct classifications
#
#
# for (i in 2:8){ # For predictor count
#   ar2_var_imp <- ar2_var[,1:i] # since 9*(k=3799) = 34191, we will have 8 trained
#                                       and 1 tested.We also start with 1 variable.
#   # 8*3799 = 30392
#   for (j in 1:9){ # For training/testing
```

```
#     rgifeo <- sample(rep(1:9, each = 3799), replace=FALSE)
#
#     # Pick training data:
#     train.subset <- ar2_var_imp[rgifeo != j, ]
#
#     # Pick testing data:
#     test.subset <- ar2_var_imp[rgifeo == j, ]
#
#     # Train the binomial logistic regression model:
#     ar2_glm <- glm(rain_tomorrow ~ ., data=train.subset, family="binomial")
#
#     # Predict the output (estimated probability):
#     ar2_glm.probs <- predict(ar2_glm, test.subset, type="response")
#
#     # Create a vector of test length filled with "No's":
#     glm.pred=rep(0,nrow(test.subset))
#
#     # Create the resulting vector where any P(rain_tom) > 0.5 --> "Yes":
#     glm.pred[ar2_glm.probs >.5]=1
#     glm.pred <- as.factor(glm.pred)
#
#     # Find percentage of correctly classified results:
#     per_cc[j] <- mean(glm.pred==ar2_var_imp$rain_tomorrow)
#   }
#
#   var_count_per_cc[i] <- median(per_cc)
#
# }
# var_count_per_cc <- var_count_per_cc[-1] # Make sure this is always length = 7!
#
# plot(1:7, var_count_per_cc, main = "Correct Classifications from Logistic Regression",
#      xlab = "Number of Variables", ylab = "Percentage of Correct Classifications")
# lines(1:7,var_count_per_cc)
#
# # B. Random Forest:
# # library(randomForest)
# #ar2_rf <- randomForest(rain_tomorrow ~ ., data=ar2)
# # importance(ar2_rf)
# # varImp(ar2_rf)
```

```
# #varImpPlot(ar2_rf)
#
# # Next steps:
# # a. Create subsetted data for the 4-7 variables and create ML code for classification.
# # b. Create a graph for each number of significant variables and how many trained observations
# # were classified correctly.
#
# set.seed(123)
# var_count_per_cc2 <- c() # for i number of variables
# per_cc2 <- c() # for j training sets, we have the number of correct classifications
#
# ar2_var2 <- subset(ar2_dummy, select=c(rain_tomorrow, humidity3pm, location, wind_dir9am,
#                                        wind_gust_dir, wind_dir3pm, pressure3pm,
#                                        pressure9am))
#
# for (i in 2:8){ # For predictor count
#   ar2_var_imp2 <- ar2_var2[,1:i] # since 9*(k=3799) = 34191, we will have 8 trained
#                                    and 1 tested.We also start with 1 variable.
#   # 8*3799 = 30392
#   for (j in 1:9){ # For training/testing
#     rgifeo2 <- sample(rep(1:9, each = 3799), replace=FALSE)
#
#     # Pick training data:
#     train.subset2 <- ar2_var_imp2[rgifeo2 != j, ]
#
#     # Pick testing data:
#     test.subset2 <- ar2_var_imp2[rgifeo2 == j, ]
#
#     # Train the random forest model:
#     ar2_rf <- randomForest(rain_tomorrow ~ ., data=train.subset2)
#
#     # Predict the output (estimated probability):
#     ar2_rf.class <- predict(ar2_rf, test.subset2, type="response")
#
#     # Find percentage of correctly classified results:
#     per_cc2[j] <- mean(ar2_rf.class==ar2_var_imp2$rain_tomorrow)
#   }
#
#   var_count_per_cc2[i] <- median(per_cc2)
```

```
#
# }
# var_count_per_cc2 <- var_count_per_cc2[-1] # Make sure this is always length = 7!
#
# plot(1:7, var_count_per_cc2, main = "Correct Classifications from Random Forest",
#      xlab = "Number of Variables", ylab = "Percentage of Correct Classifications")
# lines(1:7,var_count_per_cc2)
#
# # C. Support Vector Machine:
#
# library(e1071)
#
# set.seed(123)
# var_count_per_cc3 <- c() # for i number of variables
# per_cc3 <- c() # for j training sets, we have the number of correct classifications
#
# ar2_var3 <- subset(ar2_dummy, select=c(rain_tomorrow, humidity3pm, location, pressure3pm))
#
# for (i in 2:4){ # For predictor count
#   ar2_var_imp3 <- ar2_var3[,1:i]
#
#   for (j in 1:3){ # For training/testing
#     rgifeo3 <- sample(rep(1:9, each = 11397), replace=FALSE)
#
#     # Pick training data:
#     train.subset3 <- ar2_var_imp3[rgifeo3 != j, ]
#
#     # Pick testing data:
#     test.subset3 <- ar2_var_imp3[rgifeo3 == j, ]
#
#     # Train the random forest model:
#     ar2_svm <- svm(rain_tomorrow ~ ., kernel = "linear",
#                    cost = 0.1, data=train.subset3, scale=FALSE)
#
#     # Predict the output (estimated probability):
#     ar2_svm.class <- predict(ar2_svm, test.subset3)
#
#     # Find percentage of correctly classified results:
#     per_cc3[j] <- mean(ar2_svm.class==ar2_var_imp3$rain_tomorrow)
```

```
#   }
#
#   var_count_per_cc3[i] <- median(per_cc3)
#
# }
# var_count_per_cc3 <- var_count_per_cc3[-1] # Make sure this is always length = 7!
#
# plot(1:3, var_count_per_cc3, main = "Correct Classifications from SVM",
#      xlab = "Number of Variables", ylab = "Percentage of Correct Classifications")
# lines(1:3,var_count_per_cc3)
#
# # Final model: Use a random forest with one variable (humidity3pm):
# ar <- read.csv("C://Users//johnd.LAPTOP-35N364TU//OneDrive//Desktop//Predictive Analytics//test.csv")
# # View(ar) --> all columns included from Kaggle
#
#
# # Change the string variables into factors:
#
# # Change all character variables into factor variables:
# # Use skim(ar) to determine which string variables needed to be factored:
# # skim(ar)
# ar$location <- as.factor(ar$location)
# ar$wind_gust_dir <- as.factor(ar$wind_gust_dir)
# ar$wind_dir9am <- as.factor(ar$wind_dir9am)
# ar$wind_dir3pm <- as.factor(ar$wind_dir3pm)
# ar$rain_today <- as.factor(ar$rain_today)
#
# # D. Imputation (MICE):
# # Imputating the numeric variables using "" method:
# library(mice)
#
#
# imputed_ar <- mice(ar, m=1, method="rf")
# # View(complete(imputed_ar)) check out the dataset and compare to original "ar"
# finished_imputed_ar <- complete(imputed_ar,1)
# dim(finished_imputed_ar)
# sapply(finished_imputed_ar, function(x) sum(is.na(x)))
#
# write.csv(finished_imputed_ar,
```

```
#           "C:/Users/johnd.LAPTOP-35N364TU/OneDrive/Desktop/Predictive Analytics/test_ar.csv")
#
# # Test data set:
#
# ar <- read.csv("C:/Users/johnd.LAPTOP-35N364TU/OneDrive/Desktop/Predictive Analytics/test_ar.csv")
#
# library(randomForest)
# ar2_rf <- randomForest(rain_tomorrow ~ humidity3pm, data=ar2_var2)
#
# predictor.test <- subset(ar, select=c("humidity3pm"))
# ar2_rf.class <- predict(ar2_rf, predictor.test, type="response")
#
# id <- subset(ar, select=c('id'))
#
# arsubmission <- cbind(id,ar2_rf.class)
#
# write.csv(arsubmission,
#           "C:/Users/johnd.LAPTOP-35N364TU/OneDrive/Desktop/Predictive Analytics/final.csv")
```