Project 2 Regression

Data can be found on Kaggle: https://www.kaggle.com/jsphyg/weather-dataset-rattle-package The data is weather data in Australia and I am using this to predict the chances of Rain occurring tomorrow or not. I used the predictors of Humidity, Pressure, and WindSPeed to predict the chance of rain

Some of the steps that had to be made to cleanup the data was to re-do how the data was presented. The chance of rain was given in percent, but even a small chance of rain is enough for it to predict wheather it'll rain or not so I redid the whole column of RainToday and RainTomorrow to 1 and 0 if there is rain or no rain. In order to do that I used the class of tidyverse and after converting, had to reclassify the column to numerical data so that I could evaluate on it. Alot of the rows had N/A so to fix that I had to exclude any rows that did not have data because I couldn't evaluate on it. In addition to that, I removed a majority of the columns that I thought were irrelevant such as Date, Evaporation, Sunshine, Clouds, and Location. It could rain anywhere, and I'm not trying to predict a certain location in Australia, It can rain with the Sun, with or without clouds, and most of the evaporation or Risk MM data were N/A so i removed the whole whole thing.

```
# Reading in Data
df3 <- read.csv("weatherAUS.csv")

# Data Cleaning
df3 <- na.omit(df3)
df3$Date<-NULL
df3$Evaporation<-NULL
df3$Sunshine<-NULL
df3$Cloud9am<-NULL
df3$Cloud3pm<-NULL
df3$RISK_MM<-NULL
df3$Location<-NULL

library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.0.6     v dplyr   1.0.4
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
df3$RainToday<-str_replace_all(df3$RainToday,"No","0")
df3$RainToday<-str_replace_all(df3$RainToday,"Yes","1")
df3$RainTomorrow<-str_replace_all(df3$RainTomorrow,"No","0")
df3$RainTomorrow<-str_replace_all(df3$RainTomorrow,"Yes","1")
```

```
df3$MinTemp<-as.numeric(df3$MinTemp)
df3$RainToday<-as.numeric(df3$RainToday)
df3$RainTomorrow<-as.numeric(df3$RainTomorrow)



colSums(is.na(df3))
```

```
##       MinTemp       MaxTemp      Rainfall   WindGustDir WindGustSpeed
##             0             0             0             0             0
##    WindDir9am    WindDir3pm  WindSpeed9am  WindSpeed3pm   Humidity9am
##             0             0             0             0             0
##   Humidity3pm   Pressure9am   Pressure3pm       Temp9am       Temp3pm
##             0             0             0             0             0
##     RainToday  RainTomorrow
##             0             0
```

DATA EXPLORATION

```
str(df3)
```

```
## 'data.frame':    56420 obs. of  17 variables:
##  $ MinTemp      : num  17.9 18.4 19.4 21.9 24.2 27.1 23.3 16.1 19 19.7 ...
##  $ MaxTemp      : num  35.2 28.9 37.6 38.4 41 36.1 34 34.2 35.5 35.5 ...
##  $ Rainfall     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ WindGustDir  : chr  "SSW" "S" "NNE" "WNW" ...
##  $ WindGustSpeed: int  48 37 46 31 35 43 41 37 48 41 ...
##  $ WindDir9am   : chr  "ENE" "SSE" "NNE" "WNW" ...
##  $ WindDir3pm   : chr  "SW" "SSE" "NNW" "WSW" ...
##  $ WindSpeed9am : int  6 19 30 6 17 7 17 15 30 15 ...
##  $ WindSpeed3pm : int  20 19 15 6 13 20 19 6 9 17 ...
##  $ Humidity9am  : int  20 30 42 37 19 26 33 25 46 61 ...
##  $ Humidity3pm  : int  13 8 22 22 15 19 15 9 28 14 ...
##  $ Pressure9am  : num  1006 1013 1012 1013 1011 ...
##  $ Pressure3pm  : num  1004 1012 1009 1009 1007 ...
##  $ Temp9am      : num  26.6 20.3 28.7 29.1 33.6 30.7 25 20.7 23.4 24 ...
##  $ Temp3pm      : num  33.4 27 34.9 35.6 37.6 34.3 31.5 32.8 33.3 33.6 ...
##  $ RainToday    : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ RainTomorrow : num  0 0 0 0 0 0 0 0 0 0 ...
##  - attr(*, "na.action")= 'omit' Named int [1:89040] 1 2 3 4 5 6 7 8 9 10 ...
##   ..- attr(*, "names")= chr [1:89040] "1" "2" "3" "4" ...
```

```
names(df3)
```

```
##  [1] "MinTemp"       "MaxTemp"       "Rainfall"      "WindGustDir"
##  [5] "WindGustSpeed" "WindDir9am"    "WindDir3pm"    "WindSpeed9am"
##  [9] "WindSpeed3pm"  "Humidity9am"   "Humidity3pm"   "Pressure9am"
## [13] "Pressure3pm"   "Temp9am"       "Temp3pm"       "RainToday"
## [17] "RainTomorrow"
```

```
summary(df3)
```

```
##     MinTemp          MaxTemp          Rainfall      WindGustDir
##  Min.   :-6.70   Min.   : 4.10   Min.   :  0.00   Length:56420
##  1st Qu.: 8.60   1st Qu.:18.70   1st Qu.:  0.00   Class :character
##  Median :13.20   Median :23.90   Median :  0.00   Mode  :character
##  Mean   :13.46   Mean   :24.22   Mean   :  2.13
##  3rd Qu.:18.40   3rd Qu.:29.70   3rd Qu.:  0.60
##  Max.   :31.40   Max.   :48.10   Max.   :206.20
##  WindGustSpeed     WindDir9am         WindDir3pm         WindSpeed9am
##  Min.   :  9.00   Length:56420       Length:56420       Min.   : 2.00
##  1st Qu.: 31.00   Class :character   Class :character   1st Qu.: 9.00
##  Median : 39.00   Mode  :character   Mode  :character   Median :15.00
##  Mean   : 40.88                                        Mean   :15.67
##  3rd Qu.: 48.00                                        3rd Qu.:20.00
##  Max.   :124.00                                        Max.   :67.00
##   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        Temp9am        Temp3pm        RainToday
##  Min.   : 977.1   Min.   :-0.7   Min.   : 3.70   Min.   :0.0000
##  1st Qu.:1010.1   1st Qu.:13.1   1st Qu.:17.40   1st Qu.:0.0000
##  Median :1014.7   Median :17.8   Median :22.40   Median :0.0000
##  Mean   :1014.8   Mean   :18.2   Mean   :22.71   Mean   :0.2209
##  3rd Qu.:1019.4   3rd Qu.:23.3   3rd Qu.:27.90   3rd Qu.:0.0000
##  Max.   :1038.9   Max.   :39.4   Max.   :46.10   Max.   :1.0000
##   RainTomorrow
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.2203
##  3rd Qu.:0.0000
##  Max.   :1.0000
```

```
dim(df3)
```

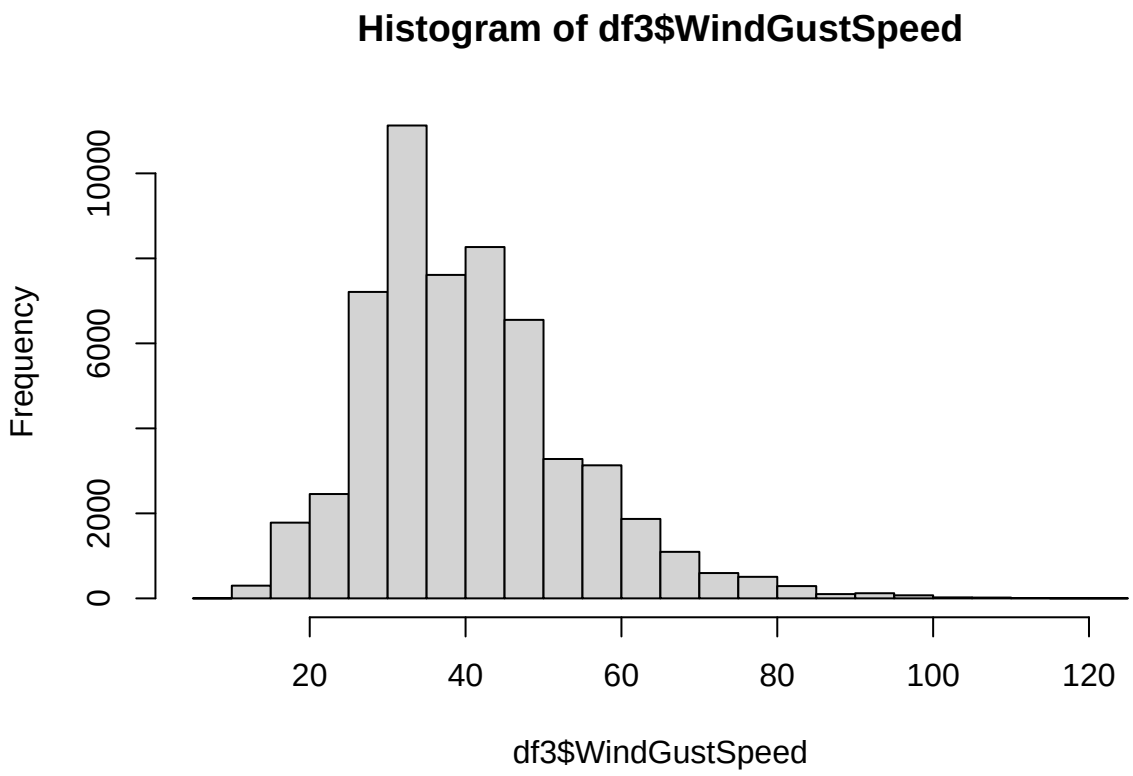
```
## [1] 56420    17
```

```
head(df3)
```

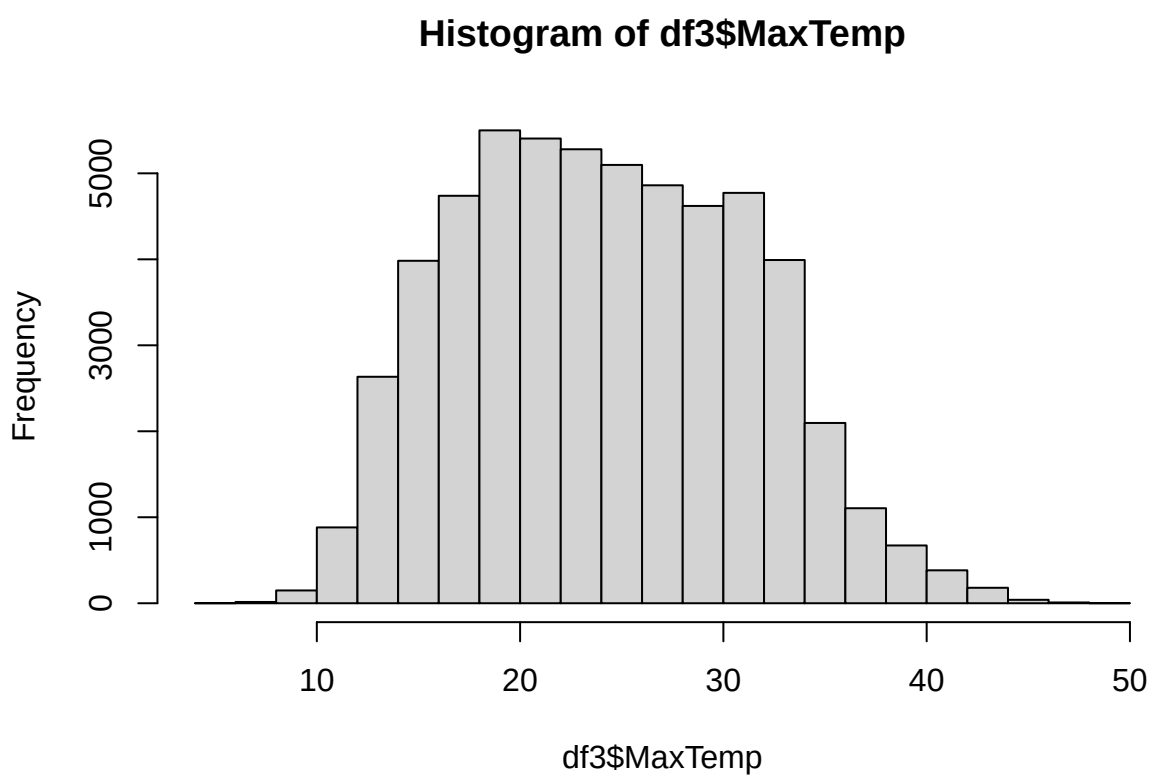
```
##      MinTemp MaxTemp Rainfall WindGustDir WindGustSpeed WindDir9am WindDir3pm
## 6050    17.9    35.2        0         SSW            48        ENE         SW
## 6051    18.4    28.9        0           S            37        SSE        SSE
## 6053    19.4    37.6        0         NNE            46        NNE        NNW
## 6054    21.9    38.4        0         WNW            31        WNW        WSW
## 6055    24.2    41.0        0         WNW            35         NW        WNW
## 6056    27.1    36.1        0           N            43          N        WNW
##      WindSpeed9am WindSpeed3pm Humidity9am Humidity3pm Pressure9am Pressure3pm
```

```
## 6050            6           20           20           13       1006.3       1004.4
## 6051           19           19           30            8       1012.9       1012.1
## 6053           30           15           42           22       1012.3       1009.2
## 6054            6            6           37           22       1012.7       1009.1
## 6055           17           13           19           15       1010.7       1007.4
## 6056            7           20           26           19       1007.7       1007.4
##      Temp9am Temp3pm RainToday RainTomorrow
## 6050    26.6    33.4         0            0
## 6051    20.3    27.0         0            0
## 6053    28.7    34.9         0            0
## 6054    29.1    35.6         0            0
## 6055    33.6    37.6         0            0
## 6056    30.7    34.3         0            0
```

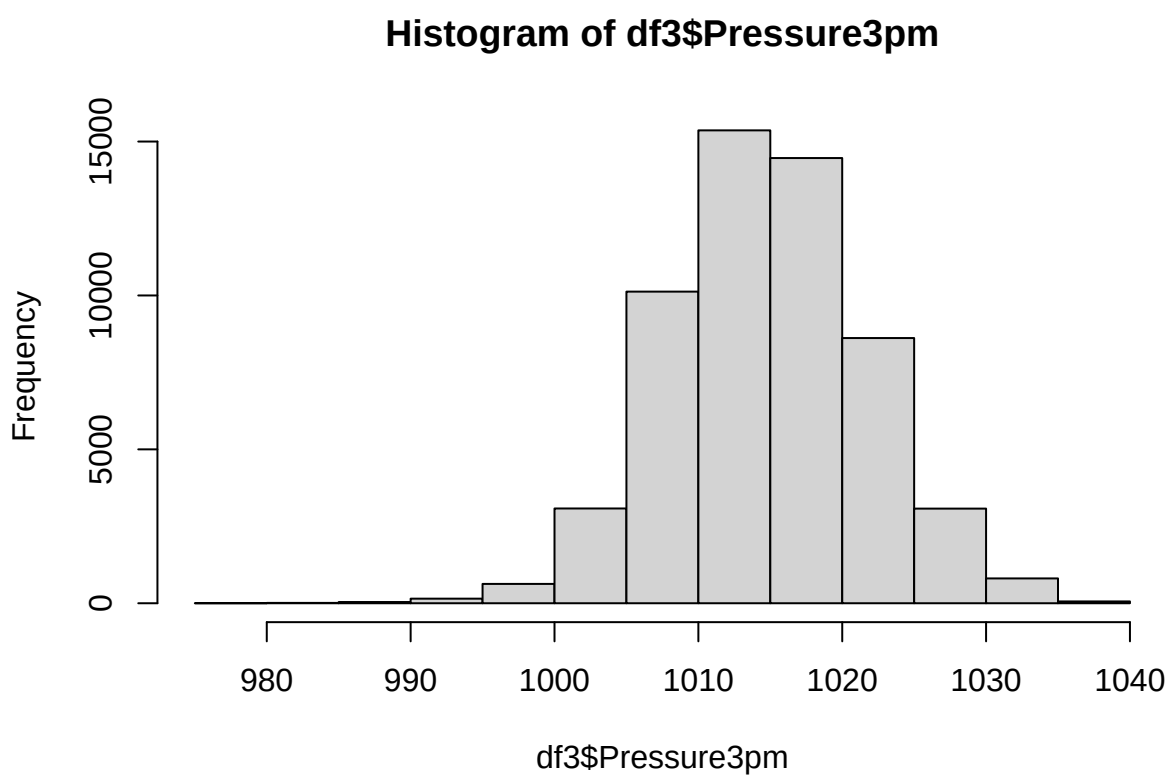
```
# Data Visualization #
hist(df3$WindGustSpeed)
```



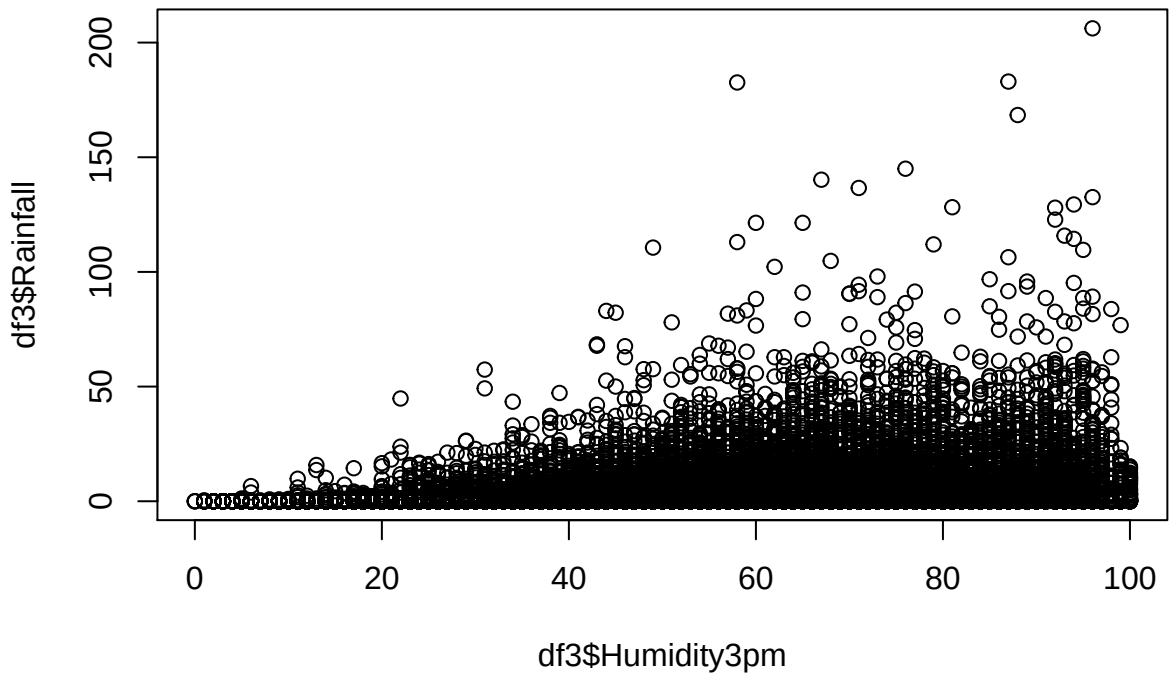

```
hist(df3$MaxTemp)
```



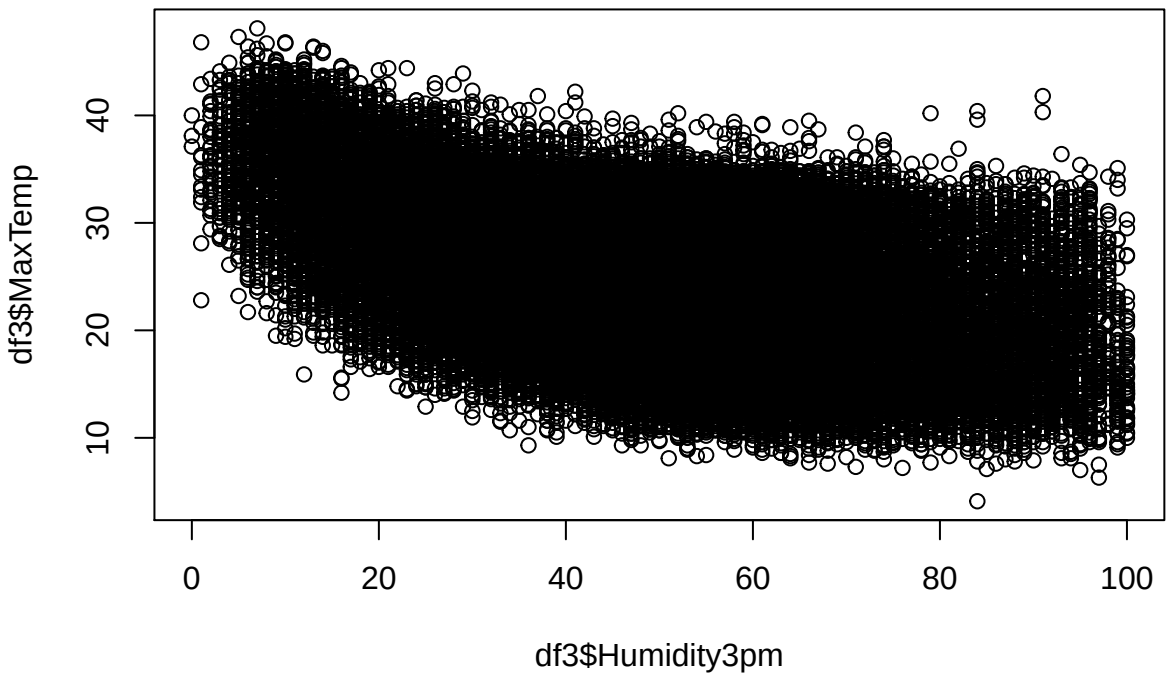

```
hist(df3$Pressure3pm)
```



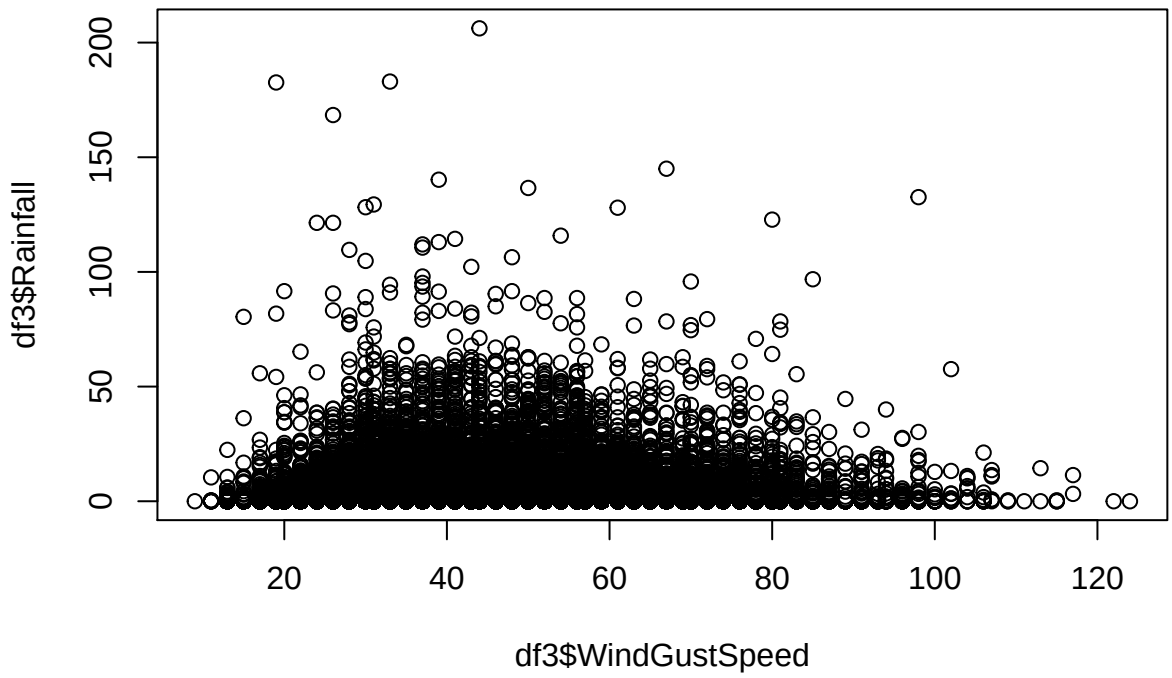

```
plot(df3$Humidity3pm, df3$Rainfall)
```



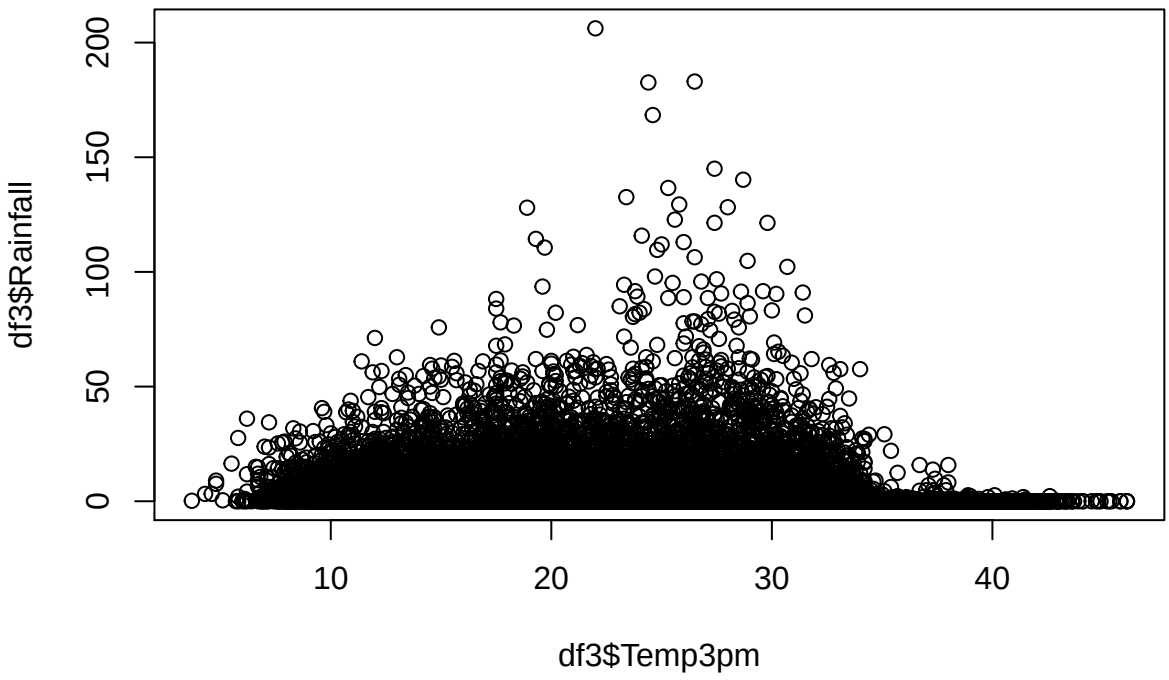

```
plot(df3$Humidity3pm, df3$MaxTemp)
```




```
plot(df3$WindGustSpeed, df3$Rainfall)
```




```
plot(df3$Temp3pm, df3$Rainfall)
```



ML ALGORITHIMS

```
# Multiple Linear Regression
df3$RainTomorrow <- as.numeric(df3$RainTomorrow)

library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
set.seed(1234)
i <- sample(1:nrow(df3), nrow(df3)*.75, replace = FALSE)
train <- df3[i,]
test <- df3[-i,]

lm1 <- lm(RainTomorrow~WindGustSpeed+Humidity3pm+Pressure3pm, data=train)
summary(lm1)
```

```
##
## Call:
## lm(formula = RainTomorrow ~ WindGustSpeed + Humidity3pm + Pressure3pm,
##     data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.97835 -0.24214 -0.08066  0.14170  1.18665
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    9.992e+00  2.735e-01   36.54   <2e-16 ***
## WindGustSpeed  5.699e-03  1.374e-04   41.49   <2e-16 ***
## Humidity3pm    9.569e-03  8.387e-05  114.10   <2e-16 ***
## Pressure3pm   -1.033e-02  2.673e-04  -38.64   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3482 on 42311 degrees of freedom
## Multiple R-squared:  0.2942, Adjusted R-squared:  0.2941
## F-statistic:  5878 on 3 and 42311 DF,  p-value: < 2.2e-16
```

```
probs_rain <- predict(lm1, newdata=test)
pred_rain <- ifelse(probs_rain>.5, 1, 0)
table(test$RainTomorrow, pred_rain>.5)
```

```
##
##     FALSE  TRUE
##   0 10658   344
##   1  1851  1252
```

```
cor1 <- cor(pred_rain, test$RainTomorrow)
mse1 <- mean((pred_rain - test$RainTomorrow)^2)
print("Correlation: ")
```

```
## [1] "Correlation: "
```

```
cor1
```

```
## [1] 0.4867333
```

```
print("MSE: ")
```

```
## [1] "MSE: "
```

```
mse1
```

```
## [1] 0.1556186
```

I chose to run a Linear Regression because I am trying to find the best model for correlation for when it rains given the Wind, Humidity, and Pressure. Linear regression works well for a model to find a linear correlation between variables, and given the following variables, I was able to create a correlation of: 48% That means that given the variables, the model predicted that there is a 48% correlation between rain happening tomorrow given the wind, humidity, and pressure. There was also a MSE of .15, given that the the rainfall is either 1 or 0, which is close to an almost perfect model.

```
# KNN REGRESSION
library(class)
#train_scaled <-  train[, 1:3,8:15]
train_scaled <-  train[c(5,11,13)]
means <- sapply(train_scaled, mean)
stdvs <- sapply(train_scaled, sd)
train_scaled <- scale(train_scaled, center = means, scale=stdvs)
#test_scaled <- scale(test[, 1:3,8:15], center = means, scale = stdvs)
test_scaled <- scale(test[c(5,11,13)], center = means, scale = stdvs)

fit <- knnreg(train_scaled, train$RainTomorrow, k=5)
pred1 <- predict(fit, test_scaled)
cor_knn1 <- cor(pred1, test$RainTomorrow)
mse_knn1 <- mean((pred1 - test$RainTomorrow)^2)
print(cor_knn1)
```

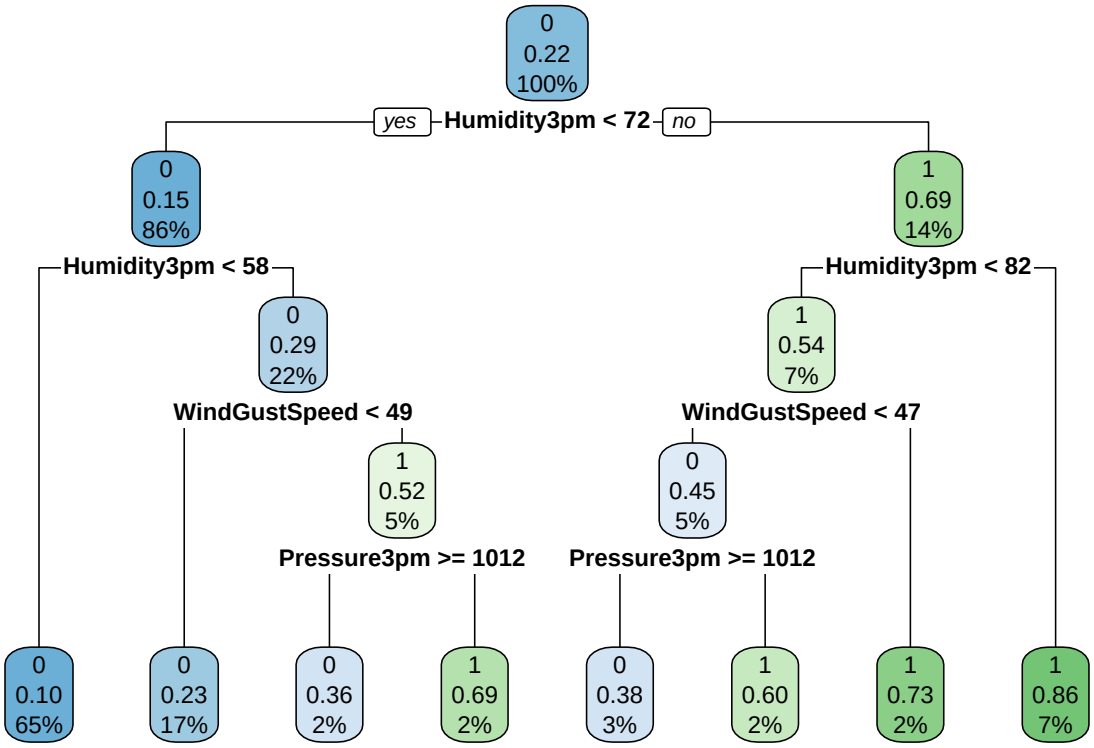
```
## [1] 0.5254814
```

```
print(mse_knn1)
```

```
## [1] 0.1287944
```

I chose to run a KNN Regression because the data given to predict when it rains is entirely Numerical. This makes it easy for KNN regression to predict a numerical target based on similarity measurements to predict if it'll rain tomorrow or not. I was able to create a correlation of: 52% This means that given the variables, the model predicted that there is a 52% correlation between rain happening given the wind, humidity, and pressure. There was also a MSE of .12, given that the the rainfall is either 1 or 0, which is close to an almost perfect model.

```
#decision tree
library(rpart)
library(rpart.plot)
decision_tree <- rpart(RainTomorrow~WindGustSpeed+Humidity3pm+Pressure3pm, data = test, method = "class
rpart.plot(decision_tree, uniform=TRUE)
```




```
pred_tree <- predict(decision_tree, newdata=test)
print(paste('correlation: ', cor(pred_tree, test$RainTomorrow)))
```

```
## [1] "correlation:  -0.542789859404947" "correlation:  0.542789859404947"
```

```
rmse_tree <- sqrt(mean((pred_tree - test$RainTomorrow)^2))
print(paste('rmse: ', rmse_tree))
```

```
## [1] "rmse:  0.615597055896974"
```

I chose to run a Decision Tree Regression because the tree breaks down the set into smaller subsets to evaluate which nodes or leafs are the best for correlation. This makes it easy for Decision Tree Regression to predict a regression because all the values are numerical as well. Given this model, I was able to create a correlation of 54% This means that given the variables, the model predicted that there is a 54% correlation between rain happening tomorrow given the wind, humidity, and pressure. There was also a RMSE of .61, meaning we were about .61 off the rainfall percetange on average given that chance of rainfall is 0 or 1. THis is relatively close to an almost perfect model

RESULTS ANALYSIS

Ranking the algorithms from best to worst: 1. Decision Tree Regression 2. KNN Regression 3. Multiple Linear Regression

The reason why Linear Regression was ranked last was because it had the lowest correlation. I think this attributed to the fact that the predictors used can vary widely. For example, wind does not always guarantee rain, but humidity and pressure does. Wind can still bring rain, but it is not always guaranteed. Given this source of error, the Linear Regression had values that were not in line of correlation, causing the correlation to be lower than the others. The next best model was KNN Regression. I think this attributed to the fact that the observations are generally bunched up so it's easier for data to take the average target value. Still the correlation was about 52% and the reason why this wasn't a super high correlation was due to the fact that rain can come from different sources of attributes. A high humidity does not always guarantee rain, and neither does pressure. Yet this still performed better than the linear regression because it was able to identify an average of the values to create a regression line. The best model to perform was the decision tree Regression. I think this attributed to the fact that there was many variables that could be split up to identify a split in the data to create a regression. The RSS in this case was minimized to the best ability because all the predictors would cause rain, but by splitting the data the averages could be accounted for each unique possibility for the predictors and as a result we got the best model of 54%. The reason why there was such low correlation for all of them, is because predicting the weather is not always guaranteed. Even in modern weather predictions the chance of rain or snow is still wild and no weather channel always gets it right, but what weather channels can do is predict the chance of rain, which is essenttially what the models I created are doing given the observations. All the model scripts were able to learn from the data and this is useful to know because if you wanted to predict the weather given certain situations, you can determine which model would work best given the predictors you wanted to use. The decision tree regression would be best if you wanted the data to tell you the chances of rain or snow for each weather pattern.