Did it rain in Seattle?

Rand

5/29/2020

Intorduction

This project aims to analyze the “Did it rain in Seattle” data set.

The original data set includes 25548 observations representing daily rainfall patterns in Seattle from January 1, 1948, to December 12, 2017, with five columns:

- DATE: The date yyyy-mm-dd.
- PRCP: Precipitation, in inches.
- TMAX: The highest temperature recorded on that day, in degrees Fahrenheit.
- TMIN: The lowest temperature recorded on that day, in degrees Fahrenheit.
- RAIN: TRUE if it has rained, FALSE if it has not.

The goal of the analysis is to predict whether it will rain on a specific day based on the expected TMAX, TMIN, and date. This means that this is a supervised classification problem with two classes (Rain, Not rain).

The steps of the analysis:

1- Exploring the relationship between predictors (TMAX, TMIN, month) and outcome.

2- Trying different machine learning classification algorithms (KNN, Regression trees, Random forests) to make predictions.

3- Choosing the optimum model based on the highest Accuracy.

The dataset was downloaded from Kaggle: https://www.kaggle.com/rtatman/did-it-rain-in-seattle-19482017 It was compiled by NOAA and is in the public domain.

Loading required libraries and rain dataset and turning it into a dataframe object:

```r
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
if(!require(lubridate)) install.packages("lubridate", repos = "http://cran.us.r-project.org")
load("./rain.rda")
rain <- as.data.frame(rain)
```

Analysis

Wrangling data:

Adding a separate column for month parsed from the DATE column, raining column which is of class factor to indicate if it rained or not and selecting relevant columns for the analysis.

```r
rain <- rain %>% mutate(month = month(DATE), year = year(DATE),
                        raini = ifelse(RAIN== TRUE, "YES", "NO"),
                        Rained = as.factor(raini)) %>%
  select(-raini) %>% select(-PRCP)
```

We are examining prevalence to ensure a balanced data set. At first, we notice the presence of NA values.

```r
mean(rain$RAIN)
```

```
## [1] NA
```

The number of NA values present in the dataset is 3. Since their number is very low compared to the number of rows in the dataset, they are unexpected to affect the analysis, so they are removed.

```r
sum(is.na(rain$RAIN))
```

```
## [1] 3
```

```r
rain<- na.omit(rain)
```

The prevalence after removing NAs is:

```r
mean(rain$RAIN)
```

```
## [1] 0.4266479
```

The prevalence indicates that our dataset is balanced, justifying using Accuracy to assess models' performances.

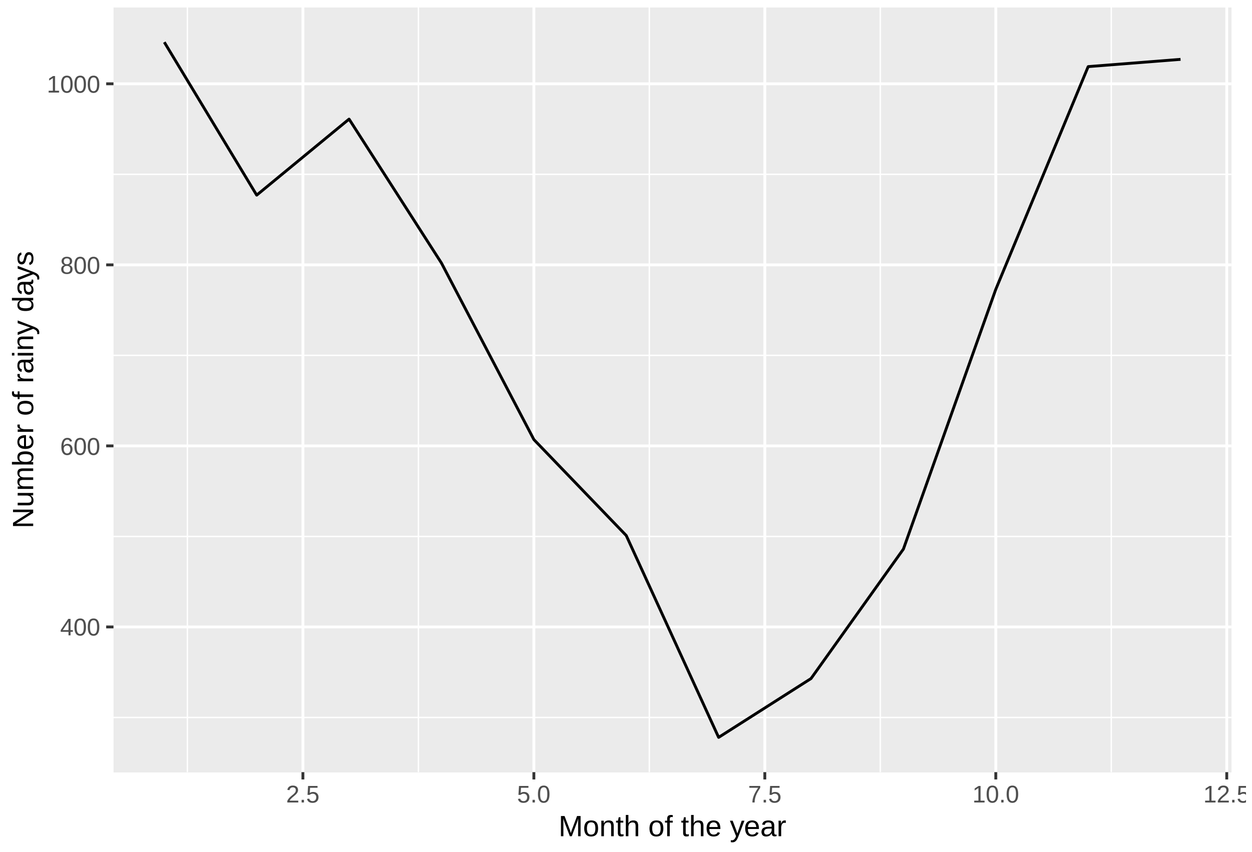
Lets split the data into training set and testing set using typical (80/20) percentage. There is no reason to split it otherwise.

```r
suppressWarnings(set.seed(1998, sample.kind = "Rounding"))
test_index <- createDataPartition(rain$Rained, times = 1, p = 0.2, list = FALSE)
test_set <- rain[test_index, ]
train_set <- rain[-test_index, ]
```

Exploratory Data Visualization:

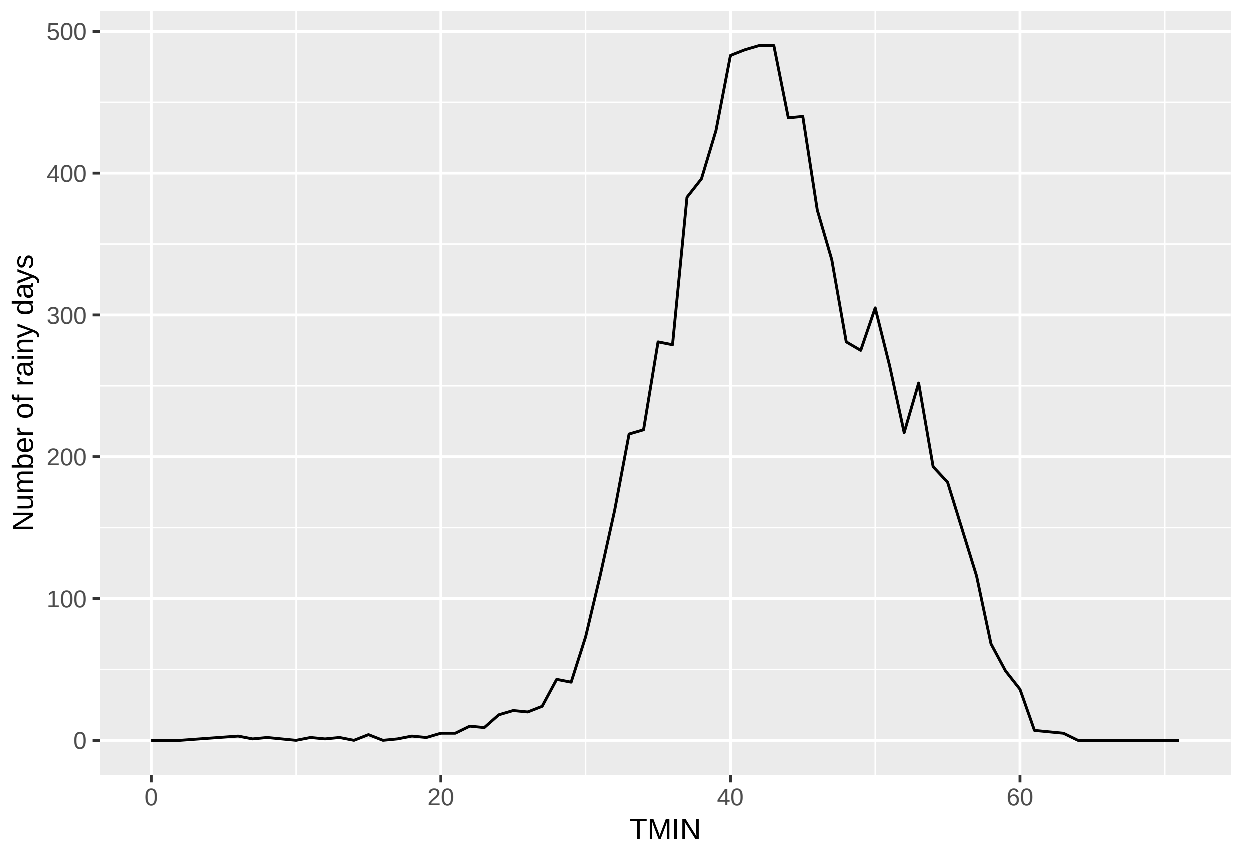
We have three predictors in the data set: TMAX, TMIN, and month. Let's examine whether there is a relationship between each of them and the classes:

First, The month versus the number of rainy days in that month across the years:



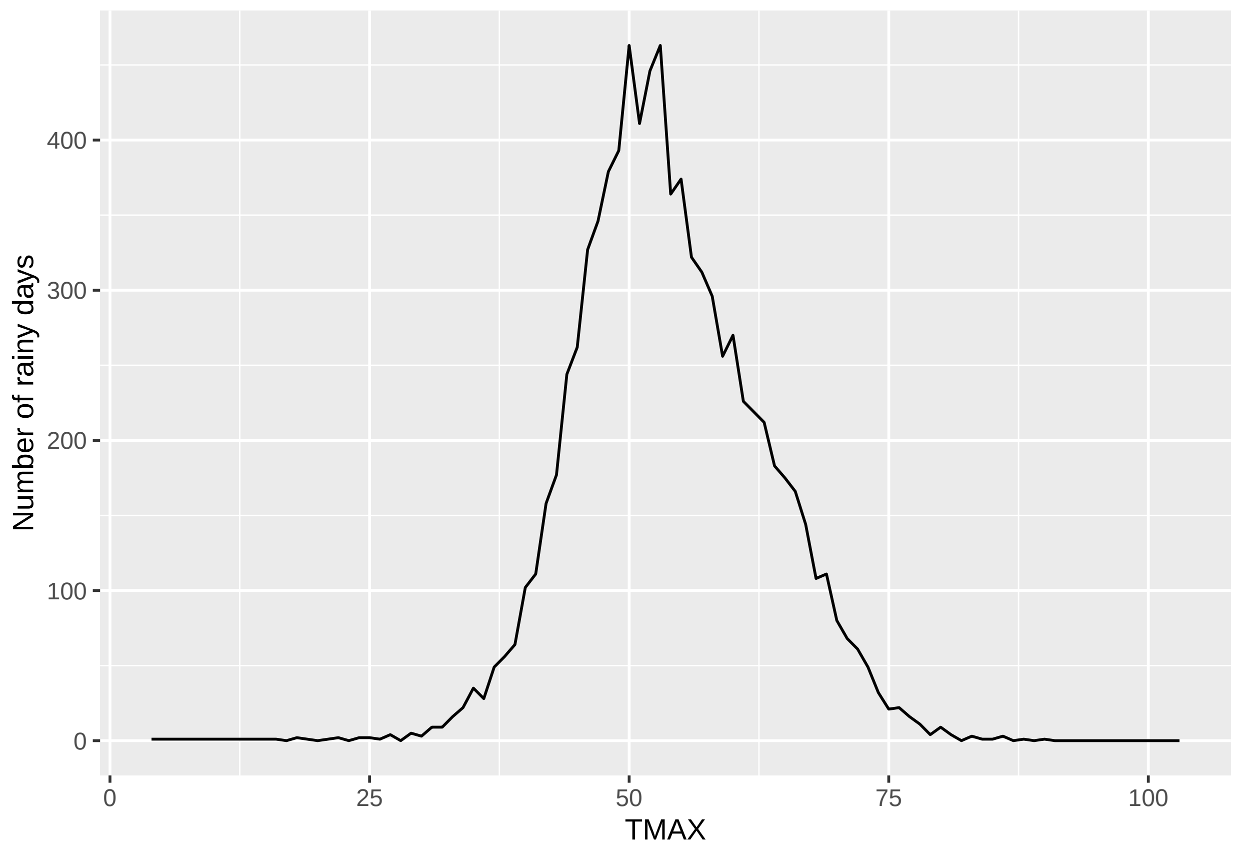
We notice that it rained more days in the winter (Jan, Dec) than summer (Jul, Aug).

Second, The minimum tempreture versus the number of rainy days:



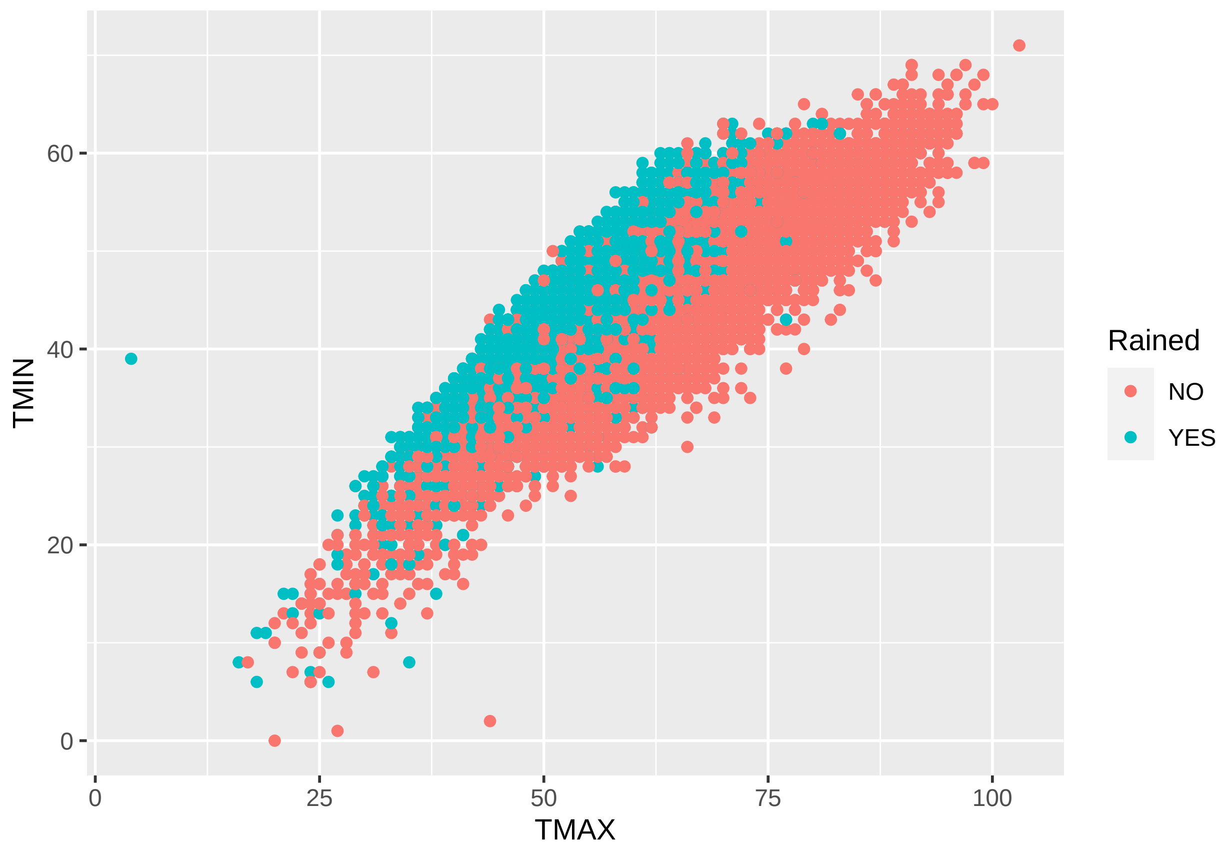
We notice that the data follows approximately a normal distribution centered around ~ 42.

Third, The maximum tempreture versus the number of rainy days:



We notice that the data follows approximately a normal distribution centered around ~ 52.

Finally, Let's get an idea of the distribution of classes based on both TMIN and TMAX:



We notice here that there are clusters of rainy and not-rainy days present in the graph, although they are not perfect. But, a line would not seperate them in a good way so I will not try linear regression.

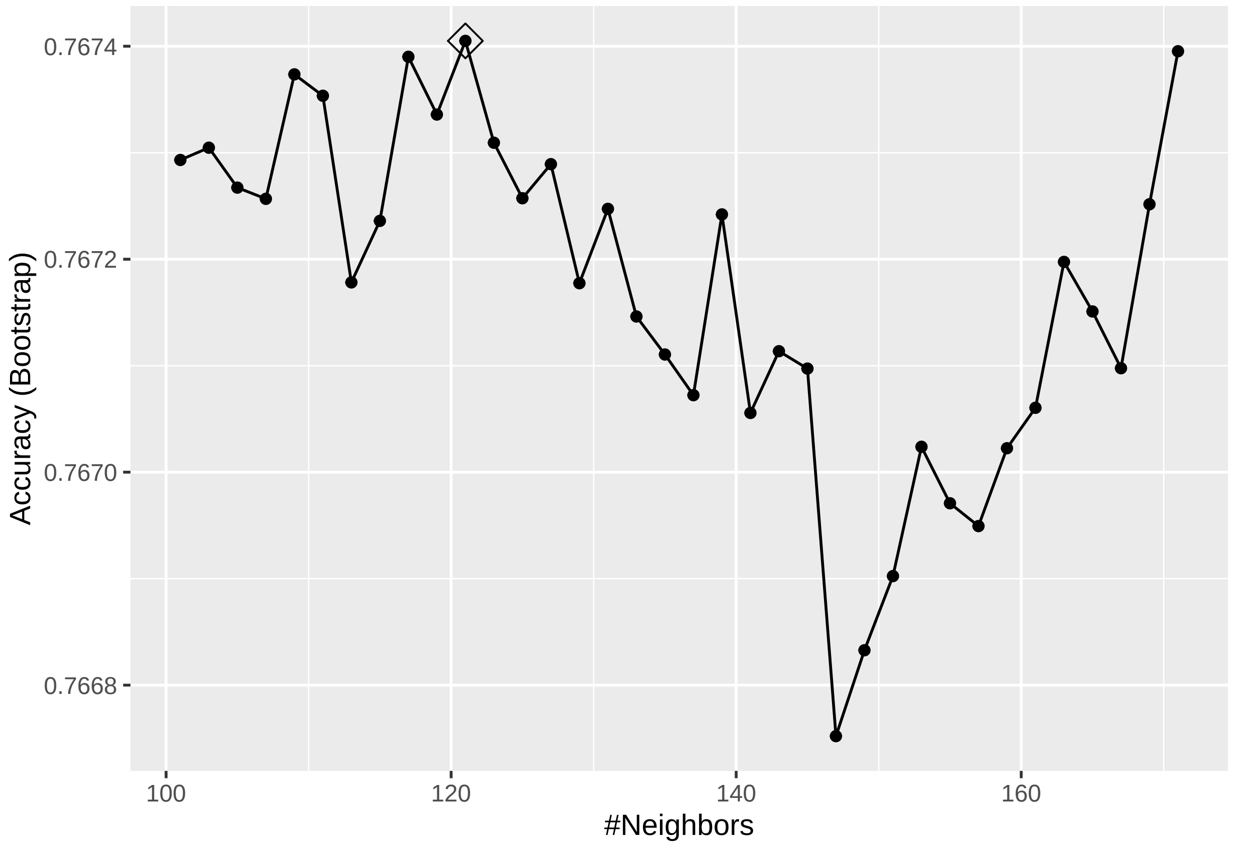
Trying different classification algorithms

Trying different classification algorithms to predict whether it rained or not based on TMAX, TMIN, and month:

KNN nearest neighborhoods After tuning k value with odd numbers from 121 to 221,

```
fit_knn <-   train(Rained ~ TMAX+TMIN+month, method = "knn",
                   data = train_set,
                   tuneGrid = data.frame(k = seq(101,171,2)))
```

We see that the Accuracy beaked at k :



best k is:

```
fit_knn$bestTune$k
```

```
## [1] 121
```

with Accuracy equals:

```
max(fit_knn$results$Accuracy)
```

```
## [1] 0.767405
```

Now, let's predict classes of the test data set using our model:

```
y_hat_knn <- predict(fit_knn, test_set, type="raw")
```

After comparing our predictions to the actual values, we obtain an accuracy of:

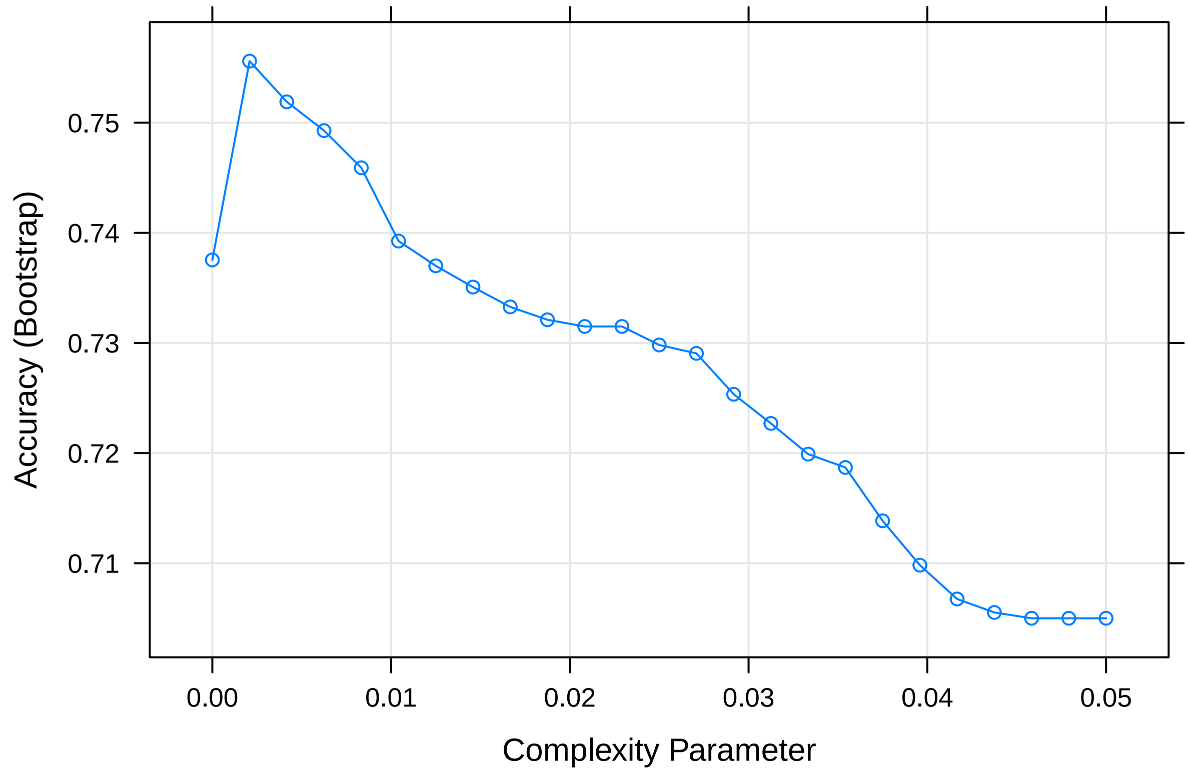
```
cmknn <- confusionMatrix(y_hat_knn, factor(test_set$Rained))
cmknn$overall["Accuracy"]
```

```
##  Accuracy
## 0.7655577
```

Regression tree algorithm: After tuning cp value with 25 numbers between 0 to 0.05,

```
fit_rpart <-   train(Rained ~ TMAX+TMIN+month,
                     method = "rpart",
                     tuneGrid = data.frame(cp = seq(0, 0.05, len = 25)),
                     data = train_set)
```

We see that the Accuracy beaked at cp :



best cp is:

```
fit_rpart$bestTune$cp
```

```
## [1] 0.002083333
```

with Accuracy equals:

```
max(fit_rpart$results$Accuracy)
```

```
## [1] 0.755592
```

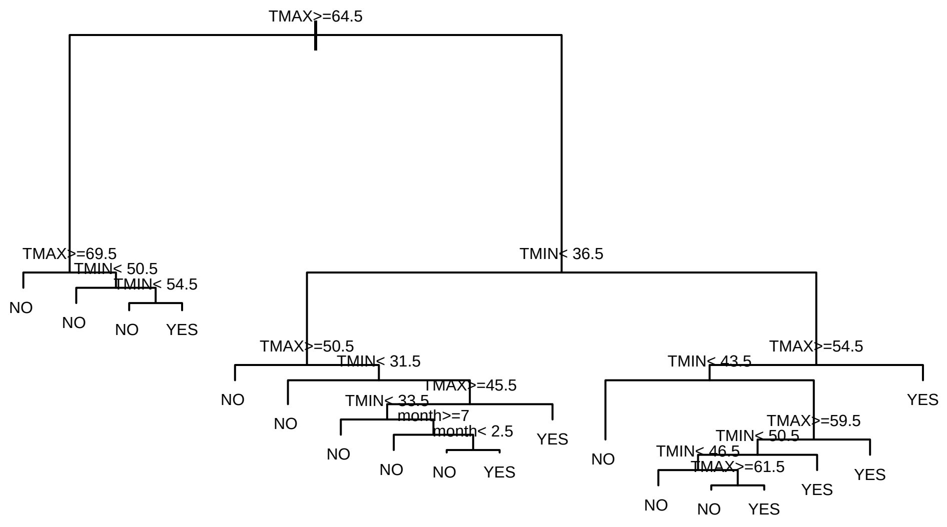
We notice from examining predictors importance that TMAX and TMIN are far more important than the month in the tree, possibly because months are highly correlated with temperature values:

```
fit_rpart$finalModel$variable.importance
```

```
##      TMAX      TMIN     month
## 2410.8725 2022.7720  226.9484
```

A visualization of the final tree:

```
plot(fit_rpart$finalModel, margin = 0.02)
text(fit_rpart$finalModel, cex = 0.5)
```



Now, let's predict classes of the test data set using our model:

```r
y_hat_rpart <- predict(fit_rpart, test_set, type="raw")
```

After comparing our predictions to the actual classes, we obtain an accuracy of:

```r
cmrp <- confusionMatrix(y_hat_rpart, factor(test_set$Rained))
cmrp$overall["Accuracy"]
```

```
##  Accuracy
## 0.7594912
```

Random forest model We begin by training the model, setting the seed to 1999 so the numbers are reproducable:

```r
suppressWarnings(set.seed(1999, sample.kind = "Rounding"))
fit_rf <- train(Rained ~ TMAX+TMIN+month,
                method = "rf",
                data = train_set)
```

```
## note: only 2 unique complexity parameters in default grid. Truncating the grid to 2 .
```

The Accuracy of the model equals:

```r
max(fit_rf$results$Accuracy)
```

```
## [1] 0.7328426
```

Now, let's predict classes of the test data set using our model:

```r
y_hat_rf <- predict(fit_rf, test_set, type="raw")
```

By comparing our predictions to the actual values, we obtain an accuracy of:

```r
cmrf <- confusionMatrix(y_hat_rf, factor(test_set$Rained))
cmrf$overall["Accuracy"]
```

```
##  Accuracy
## 0.7497065
```

Results

We see from above that the sorting of accuracies is as follows: KNN > Regression tree > Random forest.

In the beginning, we justified the model with the highest Accuracy is the best. Our highest Accuracy is ~ 0.765 produced by the KNN model.

But, it is worth mentioning that the KNN model took much more time to tune and run than the regression tree model, while it only improved Accuracy by around 0.6%.

The random forest model produced the lowest acuracy and took the longest time to run.

Conclusion

To conclude, this report aims to build a classification machine algorithm model to predict whether it will rain in Seattle or not based on expected TMAX, TMIN, and date (month of the year).

After trying several classification algorithms, KNN produced the best Accuracy.

Possible points to consider to improve results since the Accuracy of 76.5% is not very hight:

- Looking for additional predictors like humidity.
- Tuning the random forest model, which is beyond the computational capabilities of my computer.
- Using more advanced machine learning algorithms.

I hope you enjoyed the report. I am looking forward to the feedback.