REGESSION

AARUSHI PANDEY [1] BRANDON RUNYON [2] ZACHARY CANOOT [3] GRAY SIMPSON [4]

08 OCTOBER, 2022

WHAT IS OUR DATA?

This notebook explores song data from Kaggle (https://www.kaggle.com/datasets/budincsevity/szeged-weather). In particular, this is a Hungary dataset.

EXPLORING OUR DATA

Load the weatherHistory.csv file.

```
df <- read.csv("weatherHistory.csv")
df_temp <- df
str(df)
```

```
## 'data.frame':    96453 obs. of  12 variables:
##  $ Formatted.Date          : chr  "2006-04-01 00:00:00.000 +0200" "2006-
04-01 01:00:00.000 +0200" "2006-04-01 02:00:00.000 +0200" "2006-04-01
03:00:00.000 +0200" ...
##  $ Summary                 : chr  "Partly Cloudy" "Partly Cloudy" "Mostly
Cloudy" "Partly Cloudy" ...
##  $ Precip.Type             : chr  "rain" "rain" "rain" "rain" ...
##  $ Temperature..C.         : num  9.47 9.36 9.38 8.29 8.76 ...
##  $ Apparent.Temperature..C.: num  7.39 7.23 9.38 5.94 6.98 ...
##  $ Humidity                : num  0.89 0.86 0.89 0.83 0.83 0.85 0.95 0.89
0.82 0.72 ...
##  $ Wind.Speed..km.h.       : num  14.12 14.26 3.93 14.1 11.04 ...
##  $ Wind.Bearing..degrees.  : num  251 259 204 269 259 258 259 260 259 279
...
##  $ Visibility..km.         : num  15.8 15.8 15 15.8 15.8 ...
##  $ Loud.Cover              : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Pressure..millibars.    : num  1015 1016 1016 1016 1017 ...
##  $ Daily.Summary           : chr  "Partly cloudy throughout the day."
"Partly cloudy throughout the day." "Partly cloudy throughout the day."
"Partly cloudy throughout the day." ...
```

Calculate difference in Apparent Temperature and Temperature and add it as new data field.

```
df$Temperature.TempDiff <- df$Temperature..C. - df$Apparent.Temperature..C
str(df)
```

```
## 'data.frame':    96453 obs. of  13 variables:
##  $ Formatted.Date          : chr  "2006-04-01 00:00:00.000 +0200" "2006-
04-01 01:00:00.000 +0200" "2006-04-01 02:00:00.000 +0200" "2006-04-01
03:00:00.000 +0200" ...
##  $ Summary                 : chr  "Partly Cloudy" "Partly Cloudy" "Mostly
Cloudy" "Partly Cloudy" ...
##  $ Precip.Type             : chr  "rain" "rain" "rain" "rain" ...
##  $ Temperature..C.         : num  9.47 9.36 9.38 8.29 8.76 ...
##  $ Apparent.Temperature..C.: num  7.39 7.23 9.38 5.94 6.98 ...
##  $ Humidity                : num  0.89 0.86 0.89 0.83 0.83 0.85 0.95 0.89
0.82 0.72 ...
##  $ Wind.Speed..km.h.       : num  14.12 14.26 3.93 14.1 11.04 ...
##  $ Wind.Bearing..degrees.  : num  251 259 204 269 259 258 259 260 259 279
...
##  $ Visibility..km.         : num  15.8 15.8 15 15.8 15.8 ...
##  $ Loud.Cover              : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Pressure..millibars.    : num  1015 1016 1016 1016 1017 ...
##  $ Daily.Summary           : chr  "Partly cloudy throughout the day."
"Partly cloudy throughout the day." "Partly cloudy throughout the day."
"Partly cloudy throughout the day." ...
##  $ Temperature.TempDiff    : num  2.08 2.13 0 2.34 1.78 ...
```

Convert Precip.Type and Summary to factors (since they only have a few possible values)

```
df$Precip.Type <- as.factor(df$Precip.Type)
df$Summary <- as.factor(df$Summary)
str(df)
```

```
## 'data.frame':    96453 obs. of  13 variables:
##  $ Formatted.Date          : chr  "2006-04-01 00:00:00.000 +0200" "2006-04-01 01:00:00.000 +0200" "2006-04-01 02:00:00.000 +0200" "2006-04-01 03:00:00.000 +0200" ...
##  $ Summary                 : Factor w/ 27 levels "Breezy","Breezy and Dry",..: 20 20 18 20 18 20 20 20 20 20 ...
##  $ Precip.Type             : Factor w/ 3 levels "null","rain",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ Temperature..C.         : num  9.47 9.36 9.38 8.29 8.76 ...
##  $ Apparent.Temperature..C.: num  7.39 7.23 9.38 5.94 6.98 ...
##  $ Humidity                : num  0.89 0.86 0.89 0.83 0.83 0.85 0.95 0.89 0.82 0.72 ...
##  $ Wind.Speed..km.h.       : num  14.12 14.26 3.93 14.1 11.04 ...
##  $ Wind.Bearing..degrees.  : num  251 259 204 269 259 258 259 260 259 279 ...
##  $ Visibility..km.         : num  15.8 15.8 15 15.8 15.8 ...
##  $ Loud.Cover              : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Pressure..millibars.    : num  1015 1016 1016 1016 1017 ...
##  $ Daily.Summary           : chr  "Partly cloudy throughout the day." "Partly cloudy throughout the day." "Partly cloudy throughout the day." "Partly cloudy throughout the day." ...
##  $ Temperature.TempDiff    : num  2.08 2.13 0 2.34 1.78 ...
```

Our goal is to see if we can see how other weather factors, such as Wind Speed and Humidity, relate to the difference between Apparent Temperature and actual Temperature. Though we identify apparent temperature as a very good predictor of the difference, we do not use this in this assignment as we are interested in exploring more the other factors that influence the disparity.

##a. We'll divide the data into train and test.

```
set.seed(8)
i <- sample(1:nrow(df),nrow(df)*.8,replace=FALSE)
train <- df[i,]
test <- df[-i,]
```

##b. Exploring training data:

```
names(df)   # getting col names
```

```
##  [1] "Formatted.Date"           "Summary"
##  [3] "Precip.Type"              "Temperature..C."
##  [5] "Apparent.Temperature..C." "Humidity"
##  [7] "Wind.Speed..km.h."        "Wind.Bearing..degrees."
##  [9] "Visibility..km."          "Loud.Cover"
## [11] "Pressure..millibars."     "Daily.Summary"
## [13] "Temperature.TempDiff"
```

```
dim(df)   # getting number of rows and cols
```

```
## [1] 96453    13
```

```
head(df)  # getting first 6 rows
```

```
##                  Formatted.Date       Summary Precip.Type Temperature..C.
## 1 2006-04-01 00:00:00.000 +0200 Partly Cloudy        rain        9.472222
## 2 2006-04-01 01:00:00.000 +0200 Partly Cloudy        rain        9.355556
## 3 2006-04-01 02:00:00.000 +0200 Mostly Cloudy        rain        9.377778
## 4 2006-04-01 03:00:00.000 +0200 Partly Cloudy        rain        8.288889
## 5 2006-04-01 04:00:00.000 +0200 Mostly Cloudy        rain        8.755556
## 6 2006-04-01 05:00:00.000 +0200 Partly Cloudy        rain        9.222222
##   Apparent.Temperature..C. Humidity Wind.Speed..km.h.
Wind.Bearing..degrees.
## 1                 7.388889     0.89           14.1197
251
## 2                 7.227778     0.86           14.2646
259
## 3                 9.377778     0.89            3.9284
204
## 4                 5.944444     0.83           14.1036
269
## 5                 6.977778     0.83           11.0446
259
## 6                 7.111111     0.85           13.9587
258
##   Visibility..km. Loud.Cover Pressure..millibars.
## 1         15.8263          0              1015.13
## 2         15.8263          0              1015.63
## 3         14.9569          0              1015.94
## 4         15.8263          0              1016.41
## 5         15.8263          0              1016.51
## 6         14.9569          0              1016.66
##                       Daily.Summary Temperature.TempDiff
## 1 Partly cloudy throughout the day.             2.083333
## 2 Partly cloudy throughout the day.             2.127778
## 3 Partly cloudy throughout the day.             0.000000
## 4 Partly cloudy throughout the day.             2.344444
## 5 Partly cloudy throughout the day.             1.777778
## 6 Partly cloudy throughout the day.             2.111111
```

```
colMeans(df[4:11])   # calculating mean of linear cols
```

```
##          Temperature..C. Apparent.Temperature..C.                 Humidity
##                11.932678                10.855029                 0.734899
##        Wind.Speed..km.h.   Wind.Bearing..degrees.          Visibility..km.
##                10.810640               187.509232                10.347325
##               Loud.Cover     Pressure..millibars.
##                 0.000000              1003.235956
```

Since Loud.Cover col has a mean of 0, it might have NA values.

```
colSums(is.na(df))
```

```
##           Formatted.Date                  Summary              Precip.Type
##                        0                        0                        0
##          Temperature..C. Apparent.Temperature..C.                 Humidity
##                        0                        0                        0
##        Wind.Speed..km.h.   Wind.Bearing..degrees.          Visibility..km.
##                        0                        0                        0
##               Loud.Cover     Pressure..millibars.            Daily.Summary
##                        0                        0                        0
##     Temperature.TempDiff
##                        0
```

```
sum(df$Loud.Cover)
```

```
## [1] 0
```

In actuality, there are no NA values in Loud.Cover col. But since all the values there are 0, we will not gain much from using it in the prediction model. So we'll ignore it.

```
summary(df)
```

```
##  Formatted.Date                   Summary      Precip.Type    
Temperature..C.
##  Length:96453       Partly Cloudy      :31733   null:  517   Min.   
:-21.822  
##  Class :character   Mostly Cloudy      :28094   rain:85224   1st Qu.: 
4.689  
##  Mode  :character   Overcast           :16597   snow:10712   Median : 
12.000  
##                     Clear              :10890                Mean   : 
11.933  
##                     Foggy              : 7148                3rd Qu.: 
18.839  
##                     Breezy and Overcast:  528                Max.   : 
39.906  
##                     (Other)            : 1463                          
##  Apparent.Temperature..C.    Humidity      Wind.Speed..km.h.
##  Min.   :-27.717          Min.   :0.0000   Min.   : 0.000   
##  1st Qu.:  2.311          1st Qu.:0.6000   1st Qu.: 5.828   
##  Median : 12.000          Median :0.7800   Median : 9.966   
##  Mean   : 10.855          Mean   :0.7349   Mean   :10.811   
##  3rd Qu.: 18.839          3rd Qu.:0.8900   3rd Qu.:14.136   
##  Max.   : 39.344          Max.   :1.0000   Max.   :63.853   
##                                                              
##  Wind.Bearing..degrees. Visibility..km.   Loud.Cover Pressure..millibars.
##  Min.   :  0.0          Min.   : 0.00   Min.   :0    Min.   :   0        
##  1st Qu.:116.0          1st Qu.: 8.34   1st Qu.:0    1st Qu.:1012        
##  Median :180.0          Median :10.05   Median :0    Median :1016        
##  Mean   :187.5          Mean   :10.35   Mean   :0    Mean   :1003        
##  3rd Qu.:290.0          3rd Qu.:14.81   3rd Qu.:0    3rd Qu.:1021        
##  Max.   :359.0          Max.   :16.10   Max.   :0    Max.   :1046        
##                                                                          
##  Daily.Summary      Temperature.TempDiff
##  Length:96453       Min.   :-4.811      
##  Class :character   1st Qu.: 0.000      
##  Mode  :character   Median : 0.000      
##                     Mean   : 1.078      
##                     3rd Qu.: 2.217      
##                     Max.   :10.183      
## 
```

```
summary(df$Summary)
```

```
##                               Breezy                       Breezy and Dry 
##                                   54                                    1 
##                     Breezy and Foggy             Breezy and Mostly Cloudy 
##                                   35                                  516 
##                  Breezy and Overcast             Breezy and Partly Cloudy 
##                                  528                                  386 
##                                Clear Dangerously Windy and Partly Cloudy 
##                                10890                                    1 
##                              Drizzle                                  Dry 
##                                   39                                   34 
##                Dry and Mostly Cloudy                Dry and Partly Cloudy 
##                                   14                                   86 
##                                Foggy              Humid and Mostly Cloudy 
##                                 7148                                   40 
##                   Humid and Overcast              Humid and Partly Cloudy 
##                                    7                                   17 
##                           Light Rain                        Mostly Cloudy 
##                                   63                                28094 
##                             Overcast                        Partly Cloudy 
##                                16597                                31733 
##                                 Rain                                Windy 
##                                   10                                    8 
##                        Windy and Dry                      Windy and Foggy 
##                                    1                                    4 
##              Windy and Mostly Cloudy                  Windy and Overcast 
##                                   35                                   45 
##              Windy and Partly Cloudy 
##                                   67
```

```
sum(df$Wind.Speed..km.h.==0)
```

```
## [1] 1297
```

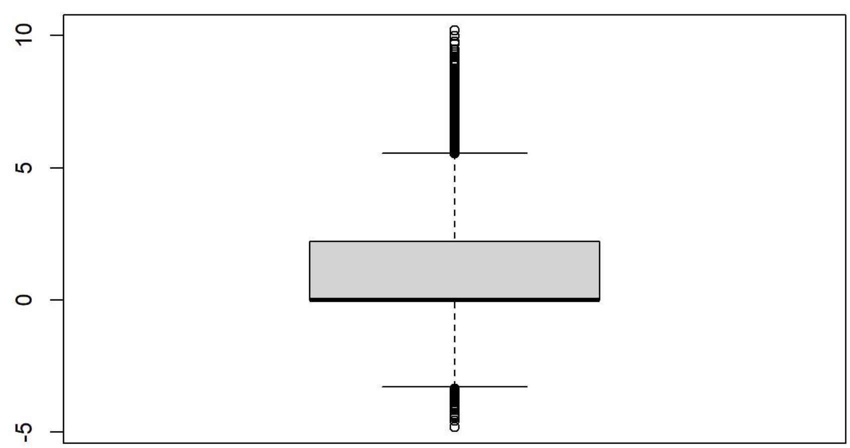
It is unlikely that there is absolutely no wind so some of this data may not be accurate.

We'll pull up some graphs to get a better idea of what we have to do, now. Yellow dots are null precipitation days, green is rain, and blue is snow.

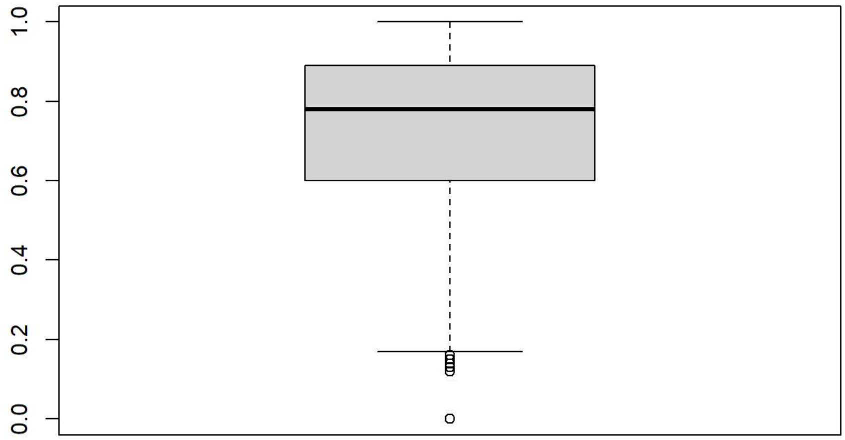
```
cor(df[4:7])
```

```
##                            Temperature..C. Apparent.Temperature..C.
Humidity
## Temperature..C.               1.000000000                0.9926286
-0.6322547
## Apparent.Temperature..C.      0.992628564                1.0000000
-0.6025710
## Humidity                     -0.632254675               -0.6025710
1.0000000
## Wind.Speed..km.h.             0.008956968               -0.0566497
-0.2249515
##                            Wind.Speed..km.h.
## Temperature..C.                 0.008956968
## Apparent.Temperature..C.       -0.056649698
## Humidity                       -0.224951456
## Wind.Speed..km.h.               1.000000000
```

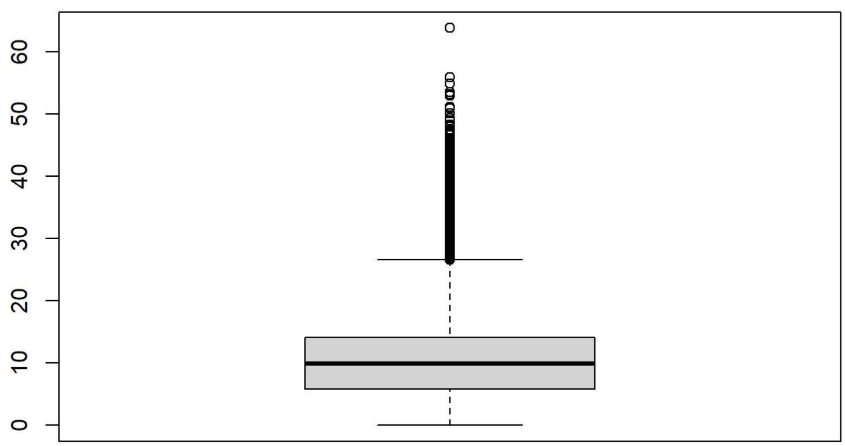
```
boxplot(df$Temperature.TempDiff)
```



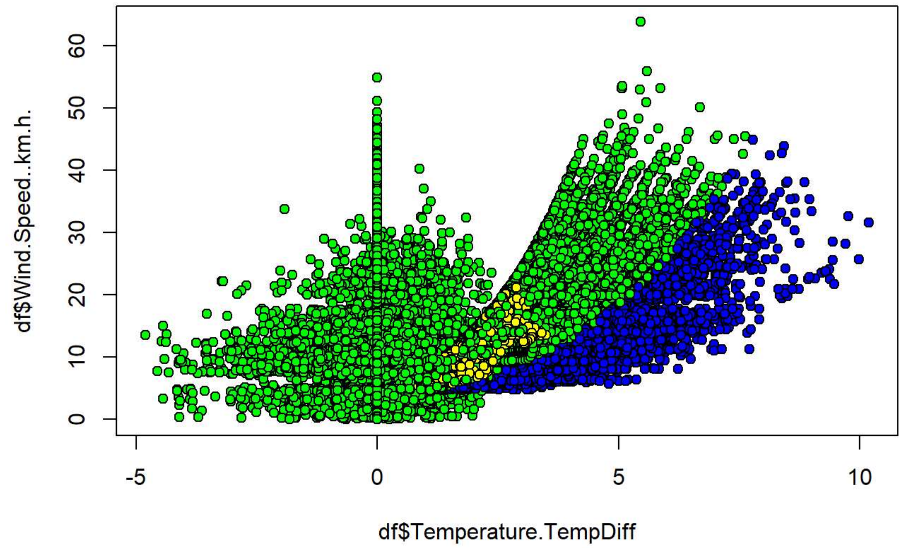

```
boxplot(df$Humidity)
```



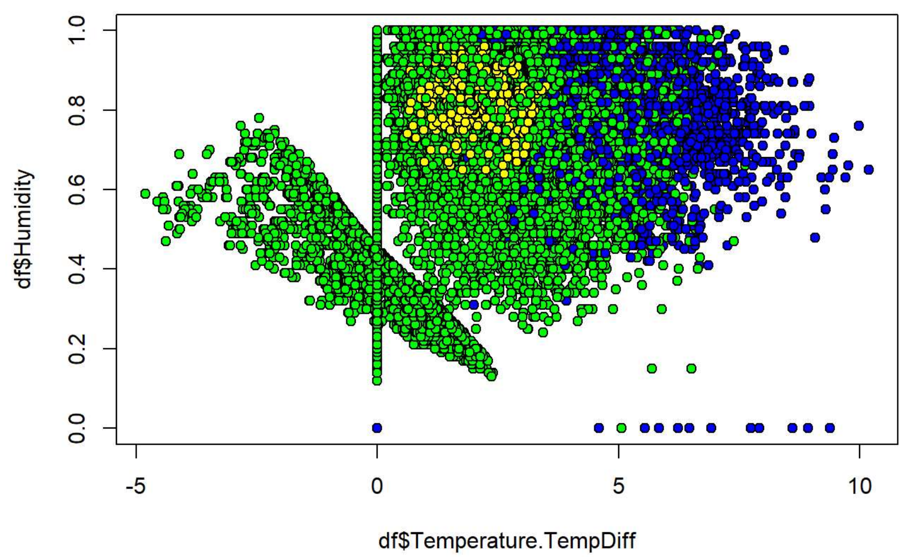

```
boxplot(df$Wind.Speed..km.h.)
```



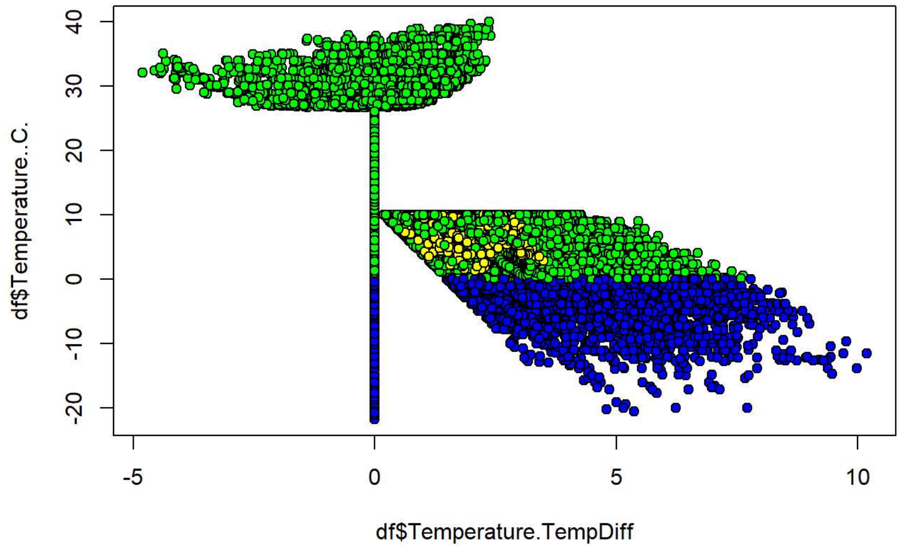

```
plot(df$Temperature.TempDiff,df$Wind.Speed..km.h.,pch=21,bg=c("yellow","green
       ","blue")[as.integer(df$Precip.Type)])  # lots of 0 values
```




```
plot(df$Temperature.TempDiff,df$Humidity,pch=21,bg=c("yellow","green","blue")
        [as.integer(df$Precip.Type)])  # lots of 0 values
```




```
plot(df$Temperature.TempDiff,df$Temperature..C.,pch=21,bg=c("yellow","green",
        "blue")[as.integer(df$Precip.Type)])  # lots of 0 values
```



Now, we'll clean up the data according to what we found. We'll clean up only what is referenced, but we will delete what we are uncertain about, since we have such a large amount of data.

```
df[,6:7][df[,6:7]==0] <- NA  # change 0s to NA values in Humidity and Wind
       Speed cols
df[,13:13][df[,13:13]==0] <- NA  # change 0s to NA values in TempDiff col
df <- na.omit(df)  # since we have enough data we can omit those which have
       NA values
summary(df)
```

```
##  Formatted.Date                      Summary      Precip.Type
Temperature..C.
##  Length:40660       Partly Cloudy      :11421   null:  237   Min.
:-20.556
##  Class :character   Mostly Cloudy      :10907   rain:32750   1st Qu.:
1.139
##  Mode  :character   Overcast           : 9062   snow: 7673   Median :
5.122
##                     Clear              : 4167                Mean   :
7.924
##                     Foggy              : 3969                3rd Qu.:
8.867
##                     Breezy and Overcast:  375                Max.   :
39.906
##                     (Other)            :  759
##  Apparent.Temperature..C.    Humidity      Wind.Speed..km.h.
##  Min.   :-27.717          Min.   :0.1300   Min.   : 0.1288
##  1st Qu.: -2.267          1st Qu.:0.6500   1st Qu.: 8.1788
##  Median :  2.544          Median :0.8200   Median :11.2217
##  Mean   :  5.370          Mean   :0.7524   Mean   :12.8271
##  3rd Qu.:  6.839          3rd Qu.:0.9100   3rd Qu.:15.4721
##  Max.   : 39.344          Max.   :1.0000   Max.   :63.8526
##
##  Wind.Bearing..degrees. Visibility..km.    Loud.Cover Pressure..millibars.
##  Min.   :  0.0          Min.   : 0.000   Min.   :0    Min.   :   0
##  1st Qu.:129.0          1st Qu.: 6.311   1st Qu.:0    1st Qu.:1012
##  Median :175.0          Median : 9.982   Median :0    Median :1017
##  Mean   :186.1          Mean   : 9.471   Mean   :0    Mean   :1001
##  3rd Qu.:280.0          3rd Qu.:11.270   3rd Qu.:0    3rd Qu.:1022
##  Max.   :359.0          Max.   :16.100   Max.   :0    Max.   :1046
##
##  Daily.Summary      Temperature.TempDiff
##  Length:40660       Min.   :-4.811
##  Class :character   1st Qu.: 1.483
##  Mode  :character   Median : 2.589
##                     Mean   : 2.554
##                     3rd Qu.: 3.628
##                     Max.   :10.183
##
```

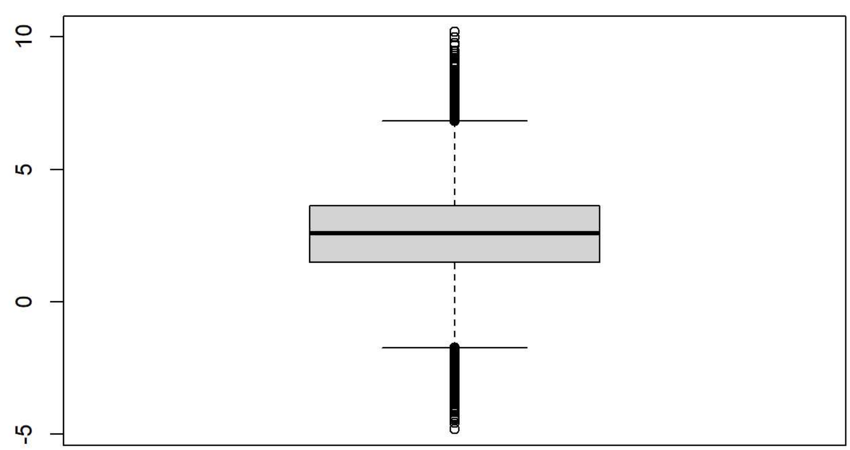
```
df_temp <- df
```

Make the graphs again.

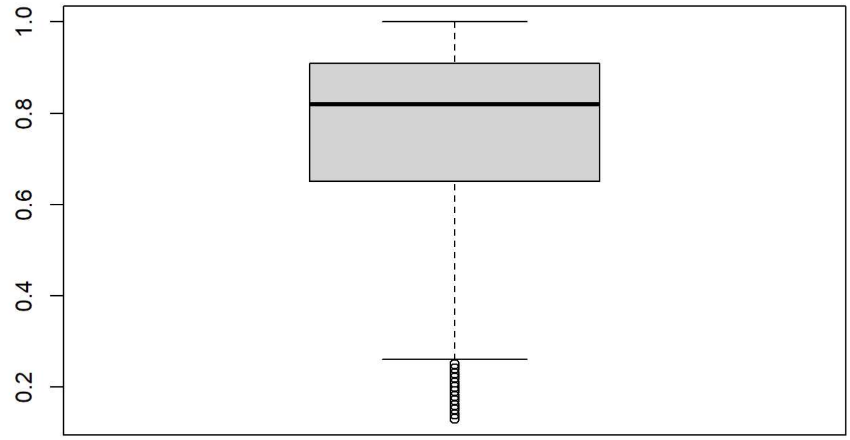
```
cor(df[4:7])
```

```
##                           Temperature..C. Apparent.Temperature..C.
Humidity
## Temperature..C.                1.00000000                0.9957386
-0.78282238
## Apparent.Temperature..C.       0.99573857                1.0000000
-0.75587228
## Humidity                      -0.78282238               -0.7558723
1.00000000
## Wind.Speed..km.h.             -0.08571425               -0.1545778
-0.06160538
##                           Wind.Speed..km.h.
## Temperature..C.                 -0.08571425
## Apparent.Temperature..C.        -0.15457779
## Humidity                        -0.06160538
## Wind.Speed..km.h.                1.00000000
```

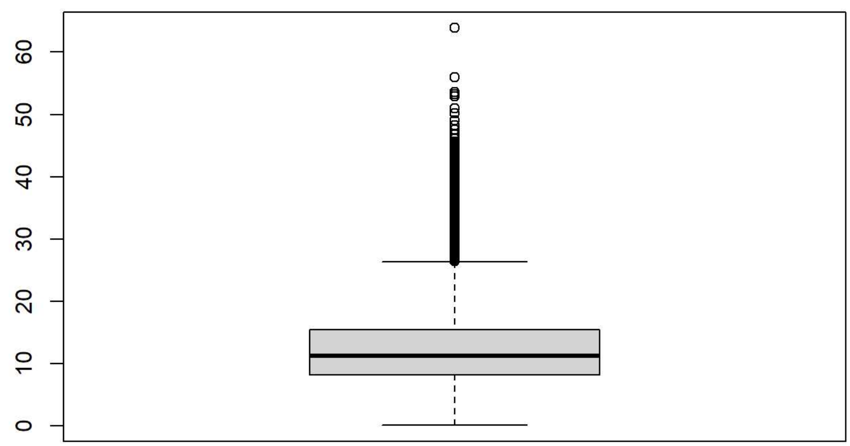
```
boxplot(df$Temperature.TempDiff)
```



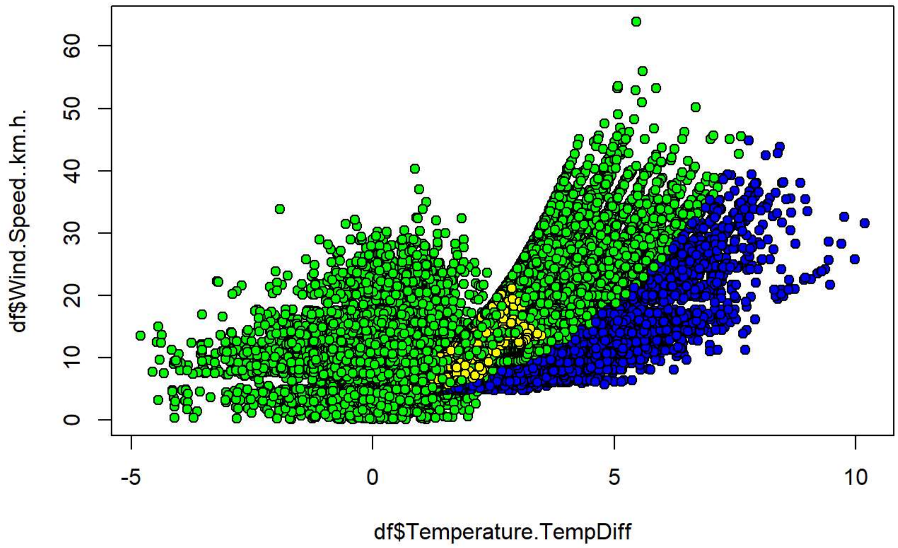

```
boxplot(df$Humidity)
```



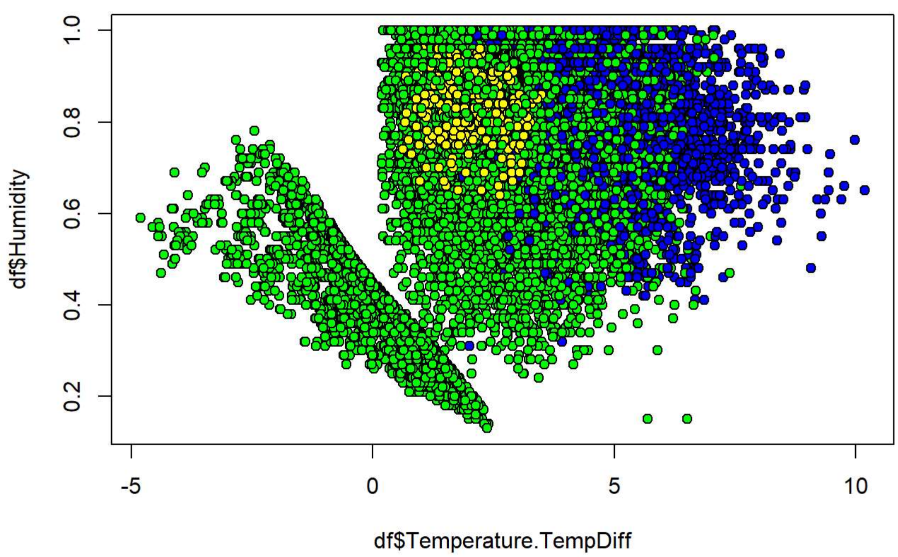
```
boxplot(df$Wind.Speed..km.h.)
```



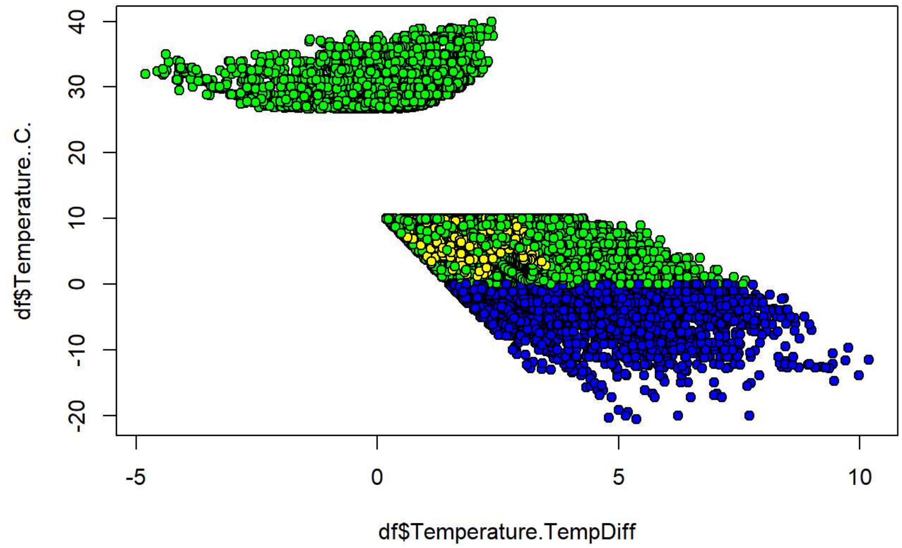
```
plot(df$Temperature.TempDiff,df$Wind.Speed..km.h.,pch=21,bg=c("yellow","green
       ","blue")[as.integer(df$Precip.Type)])  # lots of 0 values
```




```
plot(df$Temperature.TempDiff,df$Humidity,pch=21,bg=c("yellow","green","blue")
        [as.integer(df$Precip.Type)])  # lots of 0 values
```




```
plot(df$Temperature.TempDiff,df$Temperature..C.,pch=21,bg=c("yellow","green",
        "blue")[as.integer(df$Precip.Type)])  # lots of 0 values
```



We’ll clean up the train and test data again (removing the rows that had NA values).

```
trainindex <- sample(1:nrow(df),nrow(df)*.8,replace=FALSE)
train <- df[trainindex,]
test <- df[-trainindex,]
```

REGRESSION ALGORITHMS

LINEAR REGRESSION (MULTIPLE COLUMNS)

We’ll use a combination of predictors, interaction effects, and polynomial regression to see if we can get an accurate regression model.

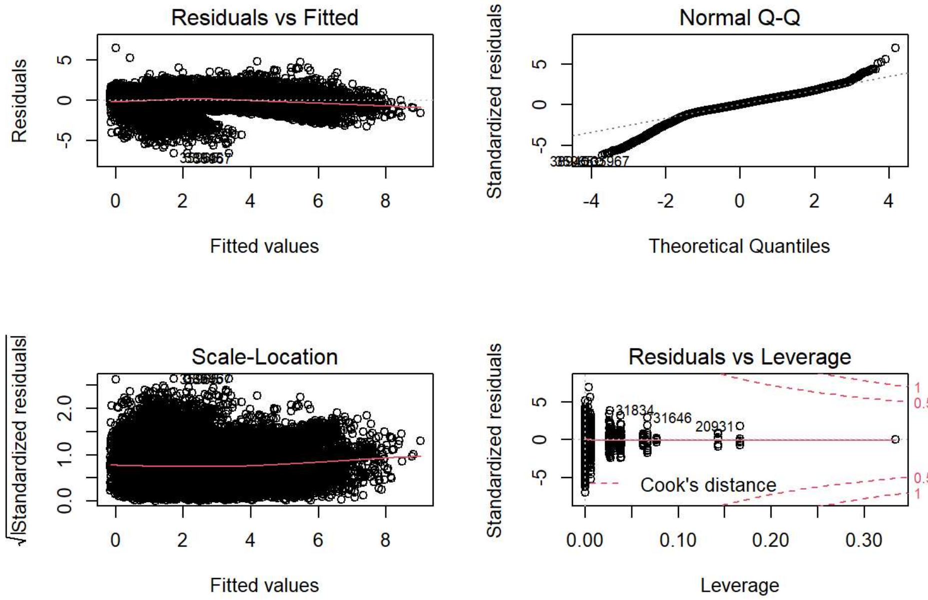
```
linreg <-
       lm(Temperature.TempDiff~poly(Humidity*Wind.Speed..km.h.)+Precip.Type+Summary,d
       ata=train)

summary(linreg)
```

```
##
## Call:
## lm(formula = Temperature.TempDiff ~ poly(Humidity * Wind.Speed..km.h.) +
##     Precip.Type + Summary, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.5629 -0.4885  0.0405  0.6044  6.5009
##
## Coefficients:
##                                             Estimate Std. Error t value
## (Intercept)                                  1.83657    0.16975  10.820
## poly(Humidity * Wind.Speed..km.h.)         223.15195    1.13970 195.799
## Precip.Typerain                              0.11849    0.06925   1.711
## Precip.Typesnow                              1.91715    0.07000  27.387
## SummaryBreezy and Foggy                     -1.35087    0.23819  -5.671
## SummaryBreezy and Mostly Cloudy             -0.34490    0.16878  -2.043
## SummaryBreezy and Overcast                  -0.89024    0.16369  -5.438
## SummaryBreezy and Partly Cloudy              0.16717    0.17086   0.978
## SummaryClear                                 0.41696    0.15612   2.671
## SummaryDangerously Windy and Partly Cloudy  -1.64055    0.95182  -1.724
## SummaryDrizzle                              -0.28248    0.30304  -0.932
## SummaryDry                                   0.92062    0.28184   3.266
## SummaryDry and Mostly Cloudy                 1.04067    0.38764   2.685
## SummaryDry and Partly Cloudy                 1.05392    0.22780   4.626
## SummaryFoggy                                 0.25100    0.15595   1.610
## SummaryLight Rain                           -0.72484    0.24044  -3.015
## SummaryMostly Cloudy                         0.36121    0.15535   2.325
## SummaryOvercast                              0.32325    0.15522   2.083
## SummaryPartly Cloudy                         0.13868    0.15563   0.891
## SummaryRain                                 -0.37107    0.56394  -0.658
## SummaryWindy                                 0.11952    0.41334   0.289
## SummaryWindy and Dry                        -1.09844    0.95185  -1.154
## SummaryWindy and Foggy                      -2.93145    0.95211  -3.079
## SummaryWindy and Mostly Cloudy              -1.46781    0.28760  -5.104
## SummaryWindy and Overcast                   -2.29974    0.24091  -9.546
## SummaryWindy and Partly Cloudy              -0.43923    0.21698  -2.024
##                                            Pr(>|t|)
## (Intercept)                                 < 2e-16 ***
## poly(Humidity * Wind.Speed..km.h.)          < 2e-16 ***
## Precip.Typerain                             0.08706 .
## Precip.Typesnow                             < 2e-16 ***
## SummaryBreezy and Foggy                    1.43e-08 ***
## SummaryBreezy and Mostly Cloudy             0.04101 *
## SummaryBreezy and Overcast                 5.41e-08 ***
## SummaryBreezy and Partly Cloudy             0.32787
## SummaryClear                                0.00757 **
## SummaryDangerously Windy and Partly Cloudy  0.08479 .
## SummaryDrizzle                              0.35126
## SummaryDry                                  0.00109 **
## SummaryDry and Mostly Cloudy                0.00727 **
## SummaryDry and Partly Cloudy               3.73e-06 ***
## SummaryFoggy                                0.10752
## SummaryLight Rain                           0.00257 **
## SummaryMostly Cloudy                        0.02007 *
## SummaryOvercast                             0.03730 *
## SummaryPartly Cloudy                        0.37290
## SummaryRain                                 0.51054
## SummaryWindy                                0.77247
## SummaryWindy and Dry                        0.24850
## SummaryWindy and Foggy                      0.00208 **
## SummaryWindy and Mostly Cloudy             3.35e-07 ***
## SummaryWindy and Overcast                   < 2e-16 ***
## SummaryWindy and Partly Cloudy              0.04295 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9391 on 32502 degrees of freedom
## Multiple R-squared:  0.6957, Adjusted R-squared:  0.6955
## F-statistic:  2973 on 25 and 32502 DF,  p-value: < 2.2e-16
```

```
par(mfrow=c(2,2))
plot(linreg)
```

```
## Warning: not plotting observations with leverage one:
##   20297, 24839, 28039
```



We understand from our data exploration that Humidity, Wind Speed, and Precipitation Type all relate to the data in different ways. We can find different trends depending on what we're looking at, so we can ask the model to reference all of that data when its processing now. When the precipitation type was rain, it didn't add much to figuring things out, but knowing that it was in the snow range was very helpful. In addition, we added Summary as well as an interaction effect with precipitation. We made this decision based on the cloud of Partly Cloudy values that didn't seem to follow other data, and we can see that some specific Summary values were quite helpful in the result, and some were not.

R^2 is almost 0.7, which is a good value. (We want R^2 to be close to 1.) The p-value is very low, and the RSE is low as well (less than 1 y-unit).

KNN REGRESSION

Load required library for prediction Since kNN model does not like factors, we will exclude it when making the model. We will only use numeric cols for prediction, and scale them too (since it produces better results that way). We will need to find the best k value before making the model.

```
library(caret)
```

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

```
## Loading required package: ggplot2
```

```
## Warning: The package `ellipsis` (>= 0.3.2) is required as of rlang 1.0.0.
```

```
## Warning: replacing previous import 'lifecycle::last_warnings' by
## 'rlang::last_warnings' when loading 'tibble'
```

```
## Warning: replacing previous import 'ellipsis::check_dots_unnamed' by
## 'rlang::check_dots_unnamed' when loading 'tibble'
```

```
## Warning: replacing previous import 'ellipsis::check_dots_used' by
## 'rlang::check_dots_used' when loading 'tibble'
```

```
## Warning: replacing previous import 'ellipsis::check_dots_empty' by
## 'rlang::check_dots_empty' when loading 'tibble'
```

```
## Warning: replacing previous import 'vctrs::data_frame' by
'tibble::data_frame'
## when loading 'dplyr'
```

```
str(df)
```

```
## 'data.frame':    40660 obs. of  13 variables:
##  $ Formatted.Date          : chr  "2006-04-01 00:00:00.000 +0200" "2006-
04-01 01:00:00.000 +0200" "2006-04-01 03:00:00.000 +0200" "2006-04-01
04:00:00.000 +0200" ...
##  $ Summary                 : Factor w/ 27 levels "Breezy","Breezy and
Dry",..: 20 20 20 18 20 20 20 20 20 20 ...
##  $ Precip.Type             : Factor w/ 3 levels "null","rain",..: 2 2 2 2
2 2 2 2 2 2 ...
##  $ Temperature..C.         : num  9.47 9.36 8.29 8.76 9.22 ...
##  $ Apparent.Temperature..C.: num  7.39 7.23 5.94 6.98 7.11 ...
##  $ Humidity                : num  0.89 0.86 0.83 0.83 0.85 0.95 0.89 0.66
0.79 0.82 ...
##  $ Wind.Speed..km.h.       : num  14.1 14.3 14.1 11 14 ...
##  $ Wind.Bearing..degrees.  : num  251 259 269 259 258 259 260 149 180 161
...
##  $ Visibility..km.         : num  15.8 15.8 15.8 15.8 15 ...
##  $ Loud.Cover              : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Pressure..millibars.    : num  1015 1016 1016 1017 1017 ...
##  $ Daily.Summary           : chr  "Partly cloudy throughout the day."
"Partly cloudy throughout the day." "Partly cloudy throughout the day."
"Partly cloudy throughout the day." ...
##  $ Temperature.TempDiff    : num  2.08 2.13 2.34 1.78 2.11 ...
##  - attr(*, "na.action")= 'omit' Named int [1:55793] 3 9 10 11 12 13 14 15
16 17 ...
##   ..- attr(*, "names")= chr [1:55793] "3" "9" "10" "11" ...
```

```
df <- df_temp
#df$Summary <- as.character(df$Summary)
#df$Precip.Type <- as.character(df$Precip.Type)
#str(df)
#colnames(df)
df <- df[-10]
str(df)
```

```
## 'data.frame':    40660 obs. of  12 variables:
##  $ Formatted.Date          : chr  "2006-04-01 00:00:00.000 +0200" "2006-
04-01 01:00:00.000 +0200" "2006-04-01 03:00:00.000 +0200" "2006-04-01
04:00:00.000 +0200" ...
##  $ Summary                 : Factor w/ 27 levels "Breezy","Breezy and
Dry",..: 20 20 20 18 20 20 20 20 20 20 ...
##  $ Precip.Type             : Factor w/ 3 levels "null","rain",..: 2 2 2 2
2 2 2 2 2 2 ...
##  $ Temperature..C.         : num  9.47 9.36 8.29 8.76 9.22 ...
##  $ Apparent.Temperature..C.: num  7.39 7.23 5.94 6.98 7.11 ...
##  $ Humidity                : num  0.89 0.86 0.83 0.83 0.85 0.95 0.89 0.66
0.79 0.82 ...
##  $ Wind.Speed..km.h.       : num  14.1 14.3 14.1 11 14 ...
##  $ Wind.Bearing..degrees.  : num  251 259 269 259 258 259 260 149 180 161
...
##  $ Visibility..km.         : num  15.8 15.8 15.8 15.8 15 ...
##  $ Pressure..millibars.    : num  1015 1016 1016 1017 1017 ...
##  $ Daily.Summary           : chr  "Partly cloudy throughout the day."
"Partly cloudy throughout the day." "Partly cloudy throughout the day."
"Partly cloudy throughout the day." ...
##  $ Temperature.TempDiff    : num  2.08 2.13 2.34 1.78 2.11 ...
```

```
trainindex <- sample(1:nrow(df),nrow(df)*.8,replace=FALSE)
train <- df[trainindex,]
test <- df[-trainindex,]

#Scaling training data
train_scaled <- train[,4:10]
means <- sapply(train_scaled, mean)
stdevs <- sapply(train_scaled, sd)
train_scaled <- scale(train_scaled, center=means, scale=stdevs)
test_scaled <- scale(test[,4:10], center=means, scale=stdevs)

#Finding the best k
#Try various values of k and plot the results.
cor_k <- rep(0, 20)
mse_k <- rep(0, 20)
i <- 1
for (k in seq(1, 39, 2)){
  fit_k <- knnreg(train_scaled,train$Temperature.TempDiff, k=k)
  pred_k <- predict(fit_k, test_scaled)
  cor_k[i] <- cor(pred_k, test$Temperature.TempDiff)
  mse_k[i] <- mean((pred_k - test$Temperature.TempDiff)^2)
  print(paste("k=", k, cor_k[i], mse_k[i]))
  i <- i + 1
}
```

```
## [1] "k= 1 0.990547972358438 0.0542421344118343"
## [1] "k= 3 0.993875806812961 0.0360420770707115"
## [1] "k= 5 0.994030804084363 0.0362040924033286"
## [1] "k= 7 0.994155641631011 0.036520442576753"
## [1] "k= 9 0.993942035613177 0.0385949512827067"
## [1] "k= 11 0.993721519465398 0.0408233633122127"
## [1] "k= 13 0.993424162441494 0.0432657587444784"
## [1] "k= 15 0.993168337935517 0.0455659292236941"
## [1] "k= 17 0.992932931142222 0.0478173606879386"
## [1] "k= 19 0.992675941824261 0.050030571327206"
## [1] "k= 21 0.992414635318523 0.0522207721593291"
## [1] "k= 23 0.992083748423219 0.054941149016596"
## [1] "k= 25 0.991809950221527 0.0573030565850389"
## [1] "k= 27 0.991544212885121 0.0594900090659074"
## [1] "k= 29 0.991392851256225 0.0610145464318253"
## [1] "k= 31 0.991154208868928 0.0629774953076742"
## [1] "k= 33 0.990842533004479 0.0654000593156279"
## [1] "k= 35 0.990601375287482 0.0674041296079488"
## [1] "k= 37 0.99036098613237 0.0693673884640314"
## [1] "k= 39 0.990117403948413 0.0712841438840286"
```

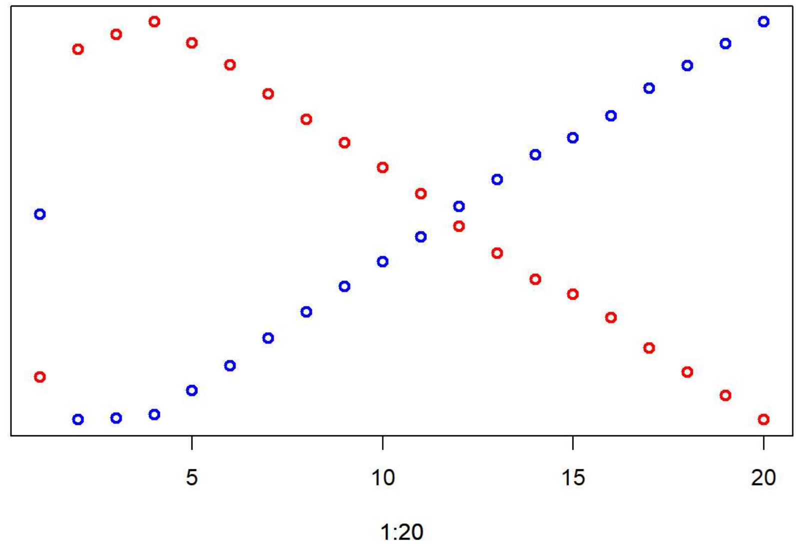
```
plot(1:20, cor_k, lwd=2, col='red', ylab="", yaxt='n')
par(new=TRUE)
plot(1:20, mse_k, lwd=2, col='blue', labels=FALSE, ylab="", yaxt='n')
```

```
## Warning in plot.window(...): "labels" is not a graphical parameter
```

```
## Warning in plot.xy(xy, type, ...): "labels" is not a graphical parameter
```

```
## Warning in box(...): "labels" is not a graphical parameter
```

```
## Warning in title(...): "labels" is not a graphical parameter
```




```
#Find the best k
which.min(mse_k)  # MSE is min when k = 3 (2nd array element)
```

```
## [1] 2
```

```
which.max(cor_k)  # COR is max when k = 5 (3rd array element)
```

```
## [1] 4
```

```
fit <- knnreg(train_scaled, train$Temperature.TempDiff, k=3)
```

Since the min MSE index (2) and max COR index (3) don't coincide, we will arbitrarily choose to have the minimum MSE index where k = 3.

DECISION TREE REGRESSION

Load required library for prediction

```
#install.packages("tree")
library(tree)
#install.packages("MASS")
library(MASS)
str(df)
```

```
## 'data.frame':    40660 obs. of  12 variables:
##  $ Formatted.Date          : chr  "2006-04-01 00:00:00.000 +0200" "2006-
04-01 01:00:00.000 +0200" "2006-04-01 03:00:00.000 +0200" "2006-04-01
04:00:00.000 +0200" ...
##  $ Summary                 : Factor w/ 27 levels "Breezy","Breezy and
Dry",..: 20 20 20 18 20 20 20 20 20 20 ...
##  $ Precip.Type             : Factor w/ 3 levels "null","rain",..: 2 2 2 2
2 2 2 2 2 2 ...
##  $ Temperature..C.         : num  9.47 9.36 8.29 8.76 9.22 ...
##  $ Apparent.Temperature..C.: num  7.39 7.23 5.94 6.98 7.11 ...
##  $ Humidity                : num  0.89 0.86 0.83 0.83 0.85 0.95 0.89 0.66
0.79 0.82 ...
##  $ Wind.Speed..km.h.       : num  14.1 14.3 14.1 11 14 ...
##  $ Wind.Bearing..degrees.  : num  251 259 269 259 258 259 260 149 180 161
...
##  $ Visibility..km.         : num  15.8 15.8 15.8 15.8 15 ...
##  $ Pressure..millibars.    : num  1015 1016 1016 1017 1017 ...
##  $ Daily.Summary           : chr  "Partly cloudy throughout the day."
"Partly cloudy throughout the day." "Partly cloudy throughout the day."
"Partly cloudy throughout the day." ...
##  $ Temperature.TempDiff    : num  2.08 2.13 2.34 1.78 2.11 ...
```

Can use all cols to predict the temperature difference in decision tree. Pruned the tree according to the best number of leaf nodes in the tree.

```
tree1 <- tree(Temperature.TempDiff~., data=train)
```

```
## Warning in tree(Temperature.TempDiff ~ ., data = train): NAs introduced by
## coercion
```

```
summary(tree1)
```

```
##
## Regression tree:
## tree(formula = Temperature.TempDiff ~ ., data = train)
## Variables actually used in tree construction:
## [1] "Apparent.Temperature..C." "Wind.Speed..km.h."
## [3] "Humidity"
## Number of terminal nodes:  12
## Residual mean deviance:  0.3301 = 10730 / 32520
## Distribution of residuals:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -4.16300 -0.37170 -0.02788  0.00000  0.35270  3.77400
```

```
treepred <- predict(tree1, newdata=test)
```

```
## Warning in pred1.tree(object, tree.matrix(newdata)): NAs introduced by
coercion
```

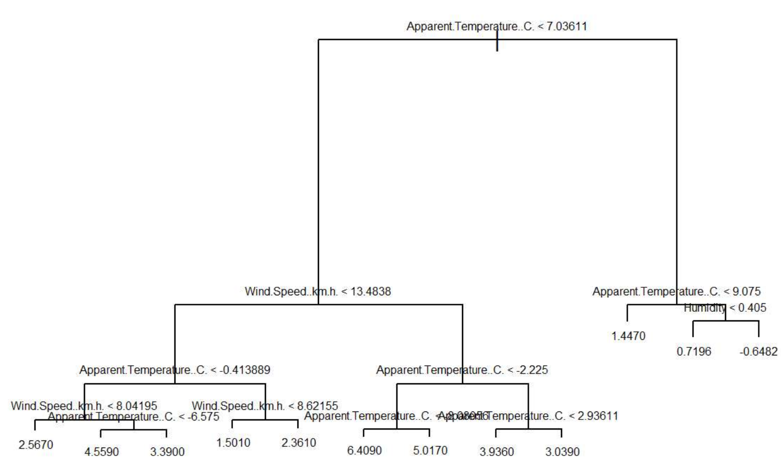
```
print(paste('correlation:', cor(treepred, test$Temperature.TempDiff)))
```

```
## [1] "correlation: 0.939530746710973"
```

```
rmse_tree <- sqrt(mean((treepred-test$Temperature.TempDiff)^2))
print(paste('rmse:', rmse_tree))
```

```
## [1] "rmse: 0.581177928558978"
```

```
plot(tree1)
text(tree1, cex=0.5, pretty=0)
```



```
# cross validation
cv_tree <- cv.tree(tree1)
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

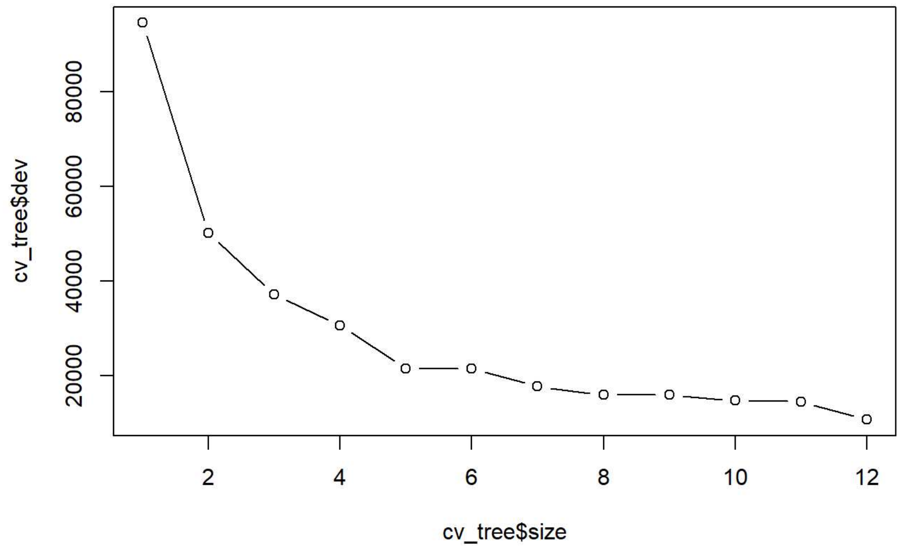
```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

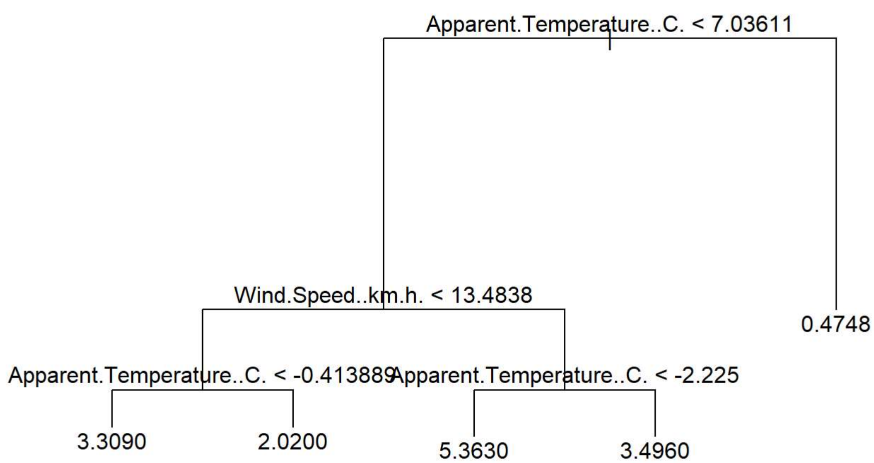
```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
plot(cv_tree$size, cv_tree$dev, type='b')
```




```
# prune the tree
tree_pruned <- prune.tree(tree1, best=5)
plot(tree_pruned)
text(tree_pruned, pretty=0)
```



PREDICTIONS

Using the three models, we will predict and evaluate using the metric correlation and MSE.

```
linregpred <- predict(linreg,newdata=test)
linregcor <- cor(linregpred,test$Temperature.TempDiff)
linregmse <- mean((linregpred-test$Temperature.TempDiff)^2)
linregrmse <- sqrt(linregmse)

#Output results
print("-------Linear Regression Model-------")
```

```
## [1] "-------Linear Regression Model-------"
```

```
print(paste("Correlation: ", linregcor))
```

```
## [1] "Correlation:  0.833165272388978"
```

```
print(paste("MSE: ", linregmse))
```

```
## [1] "MSE:  0.880883362938895"
```

```
print(paste("RMSE: ", linregrmse))
```

```
## [1] "RMSE:  0.938553867894057"
```

```
knnpred <- predict(fit, test_scaled)
knncor <- cor(knnpred, test$Temperature.TempDiff)
knnmse <- mean((knnpred-test$Temperature.TempDiff)^2)
knnrmse <- sqrt(knnmse)

#Output results
print("-------kNN Model-------")
```

```
## [1] "-------kNN Model-------"
```

```
print(paste("Correlation: ", knncor))
```

```
## [1] "Correlation:  0.993875806812961"
```

```
print(paste("MSE: ", knnmse))
```

```
## [1] "MSE:  0.0360420770707115"
```

```
print(paste("RMSE: ", knnrmse))
```

```
## [1] "RMSE:  0.18984751004612"
```

```
# test on the pruned tree
pred_pruned <- predict(tree_pruned, newdata=test)
```

```
## Warning in pred1.tree(object, tree.matrix(newdata)): NAs introduced by
coercion
```

```
cor_pruned <- cor(pred_pruned, test$Temperature.TempDiff)
mse_pruned <- mean((pred_pruned-test$Temperature.TempDiff)^2)
rmse_pruned <- sqrt(mse_pruned)

#Output results
print("-------Decision Tree Model-------")
```

```
## [1] "-------Decision Tree Model-------"
```

```
print(paste("Correlation: ", cor_pruned))
```

```
## [1] "Correlation:  0.865565194887986"
```

```
print(paste("MSE: ", mse_pruned))
```

```
## [1] "MSE:  0.72274777619241"
```

```
print(paste("RMSE: ", rmse_pruned))
```

```
## [1] "RMSE:  0.850145738207521"
```

The highest correlation was by the kNN regression model (0.99), followed by the decision tree regression model (0.87), and lastly the linear regression model (0.84). Unsurprisingly, the order for the lowest mean squared error and root mean squared error is in the same ordeR: kNN (0.034 MSE and 0.18 RMSE), decision tree (0.69 MSE and 0.83 RMSE), and linear model (0.85 MSE and 0.92 RMSE). By analyzing the results, it is easy to conclude that the kNN model is the best for predicting the difference in the actual and apparent temperatures.

#d. Conclusion and Analysis

Due to kNN not being interpretable, it is hard to pinpoint exactly why this model predicts better than other models. Something we did different when preparing the data for this model was scaling the numeric columns before using them to predict the difference in temperatures. This scaling of the data could have created better predictors than our original columns. The decision tree model also did quite well, despite not being the most accurate algorithm (since it is a greedy algorithm that can only divide data by making linear boundaries). However, it is highly interpretable and seemed to find significant predictors (apparent temperature and wind speed) easily. For the linear model, it was up to us to choose the predictors, and we chose humidity, wind speed, precipitation type, and summary. It is easy to understand that due to limited information and data exploration, we may have picked unnecessary predictors or excluded significant ones. In this case, the true relationship between the different temperatures may not have been linear which prevented this model from being more accurate.

---

1. Aarushi's Portfolio (https://aarushi-pandey.github.io/Portfolio_ML/) ↩
2. Brandon's Portfolio () ↩
3. Zaiquiri's Portfolio (https://zaiquiriw.github.io/ml-portfolio/) ↩
4. Gray's Porfolio (https://ecclysium.github.io/MachineLearning_Portfolio/) ↩