logistic_regression.R

Magilan

Mon Oct 08 16:36:47 2018

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.0.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.6
## v tidyr   0.8.1     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0
```

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

```
library(boot)
library(forecast)
library(tseries)
library(caret)
```

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

```
##
## Attaching package: 'lattice'
```

```
## The following object is masked from 'package:boot':
##
##     melanoma
```

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

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

```
library(ROCR)
```

```
## Loading required package: gplots
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
library(psych)
```

```
##
## Attaching package: 'psych'
```

```
## The following object is masked from 'package:boot':
##
##     logit
```

```
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

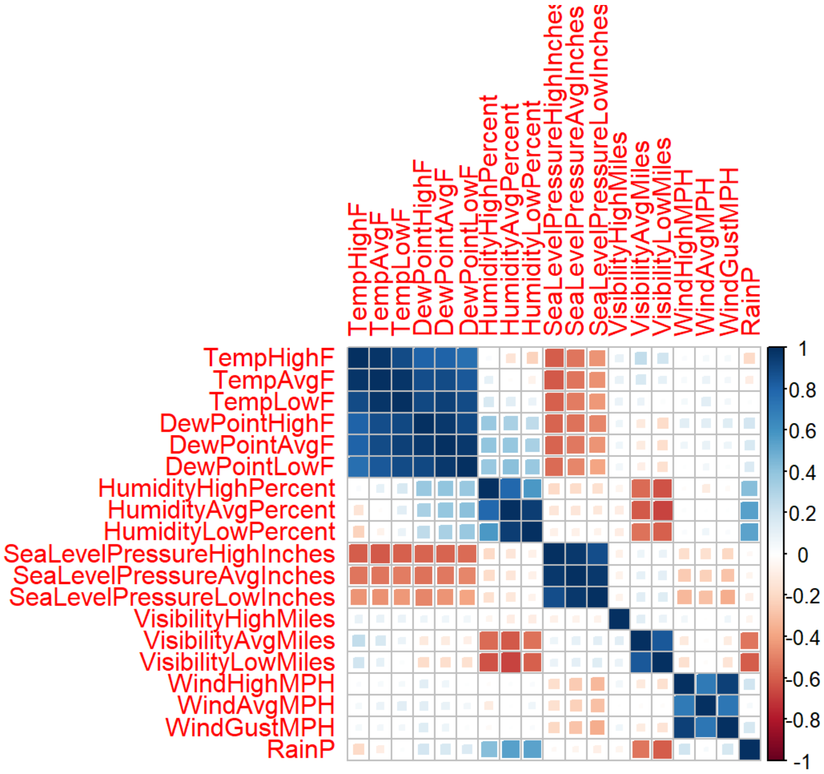
```
# Data Input

data <- read.csv("C:/Users/Magilan/Desktop/ML_project/austin_weather.csv",header = TRUE)
data1=na.omit(data,invert=FALSE)
attach(data1)
summary(data1)
```

```
##           Date        TempHighF         TempAvgF         TempLowF
##  01-01-2014:   1   Min.   : 32.00   Min.   :29.00   Min.   :19.00
##  01-01-2015:   1   1st Qu.: 72.00   1st Qu.:62.00   1st Qu.:49.00
##  01-02-2014:   1   Median : 83.00   Median :73.00   Median :62.00
##  01-02-2015:   1   Mean   : 80.79   Mean   :70.56   Mean   :59.82
##  01-02-2016:   1   3rd Qu.: 92.00   3rd Qu.:83.00   3rd Qu.:73.00
##  01-02-2017:   1   Max.   :107.00   Max.   :93.00   Max.   :81.00
##  (Other)   :1299
##  DewPointHighF    DewPointAvgF    DewPointLowF   HumidityHighPercent
##  Min.   :13.00   Min.   : 8.00   Min.   : 2.00   Min.   : 37.00
##  1st Qu.:53.00   1st Qu.:46.00   1st Qu.:38.00   1st Qu.: 85.00
##  Median :66.00   Median :61.00   Median :56.00   Median : 90.00
##  Mean   :61.52   Mean   :56.64   Mean   :50.94   Mean   : 87.83
##  3rd Qu.:73.00   3rd Qu.:69.00   3rd Qu.:65.00   3rd Qu.: 94.00
##  Max.   :80.00   Max.   :76.00   Max.   :75.00   Max.   :100.00
##
##  HumidityAvgPercent HumidityLowPercent SeaLevelPressureHighInches
##  Min.   :27.00      Min.   :10.00      Min.   :29.63
##  1st Qu.:59.00      1st Qu.:33.00      1st Qu.:29.99
##  Median :67.00      Median :44.00      Median :30.08
##  Mean   :66.66      Mean   :44.98      Mean   :30.11
##  3rd Qu.:74.00      3rd Qu.:55.00      3rd Qu.:30.21
##  Max.   :97.00      Max.   :93.00      Max.   :30.83
##
##  SeaLevelPressureAvgInches SeaLevelPressureLowInches VisibilityHighMiles
##  Min.   :29.55             Min.   :29.41             Min.   : 5.000
##  1st Qu.:29.91             1st Qu.:29.82             1st Qu.:10.000
##  Median :30.00             Median :29.91             Median :10.000
##  Mean   :30.02             Mean   :29.93             Mean   : 9.992
##  3rd Qu.:30.10             3rd Qu.:30.02             3rd Qu.:10.000
##  Max.   :30.74             Max.   :30.61             Max.   :10.000
##
##  VisibilityAvgMiles VisibilityLowMiles  WindHighMPH      WindAvgMPH
##  Min.   : 2.000     Min.   : 0.000     Min.   : 6.00   Min.   : 1.000
##  1st Qu.: 9.000     1st Qu.: 3.000     1st Qu.:10.00   1st Qu.: 3.000
##  Median :10.000     Median : 9.000     Median :13.00   Median : 5.000
##  Mean   : 9.162     Mean   : 6.843     Mean   :13.25   Mean   : 5.009
##  3rd Qu.:10.000     3rd Qu.:10.000     3rd Qu.:15.00   3rd Qu.: 6.000
##  Max.   :10.000     Max.   :10.000     Max.   :29.00   Max.   :12.000
##
##   WindGustMPH    PrecipitationSumInches  Rain          RainP
##  Min.   : 9.00   Min.   :0.0000         no :859   Min.   :0.0000
##  1st Qu.:17.00   1st Qu.:0.0000         yes:446   1st Qu.:0.0000
##  Median :21.00   Median :0.0000                   Median :0.0000
##  Mean   :21.38   Mean   :0.1248                   Mean   :0.3418
##  3rd Qu.:25.00   3rd Qu.:0.0800                   3rd Qu.:1.0000
##  Max.   :57.00   Max.   :5.2000                   Max.   :1.0000
##
```

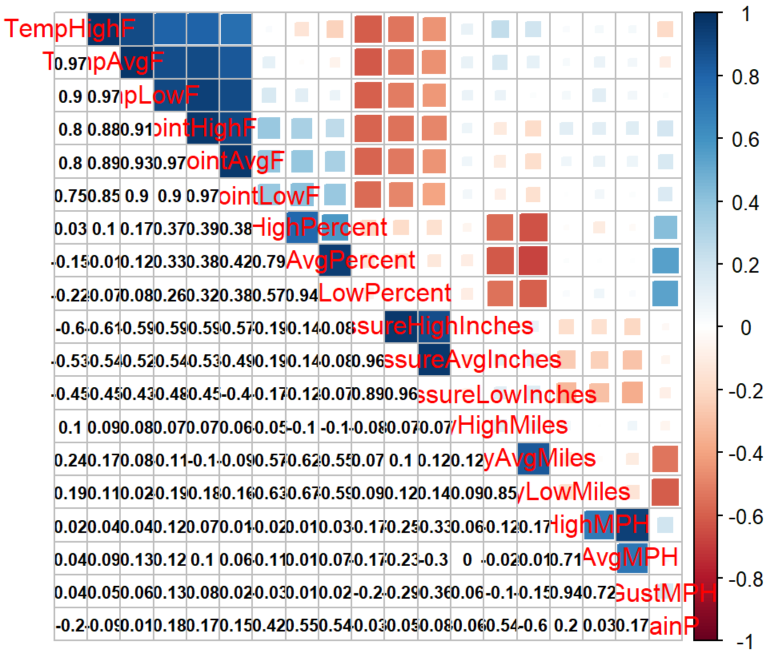
```
summary(Rain)
```

```
##  no yes
## 859 446
```

```
mat=cor(data1[,-c(1,20,21)],method = "spearman")


corrplot(mat,method = "square")
```




```
corrplot.mixed(mat, lower.col = "black",upper = "square", number.cex = .7)
```



```r
# Data Partitioning

index <- createDataPartition(Rain, p = 0.7, list = FALSE)
# Training set
train.df <- data1[index,]
# Testing dataset
test.df <- data1[-index,]

summary(train.df)
```

```
##          Date       TempHighF        TempAvgF       TempLowF
##  01-01-2014:  1   Min.   : 32.00   Min.   :29.00   Min.   :19.00
##  01-01-2015:  1   1st Qu.: 71.00   1st Qu.:61.00   1st Qu.:49.00
##  01-02-2014:  1   Median : 83.00   Median :73.00   Median :62.00
##  01-02-2015:  1   Mean   : 80.62   Mean   :70.45   Mean   :59.78
##  01-03-2014:  1   3rd Qu.: 92.00   3rd Qu.:83.00   3rd Qu.:73.00
##  01-03-2015:  1   Max.   :107.00   Max.   :93.00   Max.   :80.00
##  (Other)   :909
##  DewPointHighF    DewPointAvgF    DewPointLowF   HumidityHighPercent
##  Min.   :13.00   Min.   :11.00   Min.   : 4.00   Min.   : 37.00
##  1st Qu.:52.00   1st Qu.:46.00   1st Qu.:38.00   1st Qu.: 84.50
##  Median :66.00   Median :61.00   Median :56.00   Median : 90.00
##  Mean   :61.35   Mean   :56.52   Mean   :50.87   Mean   : 87.82
##  3rd Qu.:73.00   3rd Qu.:69.00   3rd Qu.:65.00   3rd Qu.: 94.00
##  Max.   :80.00   Max.   :76.00   Max.   :75.00   Max.   :100.00
##
##  HumidityAvgPercent HumidityLowPercent SeaLevelPressureHighInches
##  Min.   :27.00      Min.   :10         Min.   :29.63
##  1st Qu.:59.00      1st Qu.:32         1st Qu.:30.00
##  Median :67.00      Median :44         Median :30.08
##  Mean   :66.68      Mean   :45         Mean   :30.12
##  3rd Qu.:75.00      3rd Qu.:55         3rd Qu.:30.21
##  Max.   :97.00      Max.   :93         Max.   :30.83
##
##  SeaLevelPressureAvgInches SeaLevelPressureLowInches VisibilityHighMiles
##  Min.   :29.55             Min.   :29.42             Min.   : 8.000
##  1st Qu.:29.92             1st Qu.:29.83             1st Qu.:10.000
##  Median :30.00             Median :29.92             Median :10.000
##  Mean   :30.03             Mean   :29.94             Mean   : 9.993
##  3rd Qu.:30.11             3rd Qu.:30.02             3rd Qu.:10.000
##  Max.   :30.74             Max.   :30.61             Max.   :10.000
##
##  VisibilityAvgMiles VisibilityLowMiles  WindHighMPH      WindAvgMPH
##  Min.   : 2.000     Min.   : 0.000     Min.   : 7.00   Min.   : 1.000
##  1st Qu.: 9.000     1st Qu.: 3.000     1st Qu.:10.00   1st Qu.: 3.000
##  Median :10.000     Median : 9.000     Median :13.00   Median : 5.000
##  Mean   : 9.158     Mean   : 6.902     Mean   :13.23   Mean   : 5.019
##  3rd Qu.:10.000     3rd Qu.:10.000     3rd Qu.:15.00   3rd Qu.: 6.000
##  Max.   :10.000     Max.   :10.000     Max.   :29.00   Max.   :11.000
##
##   WindGustMPH    PrecipitationSumInches  Rain          RainP
##  Min.   : 9.00   Min.   :0.0000         no :602   Min.   :0.0000
##  1st Qu.:17.00   1st Qu.:0.0000         yes:313   1st Qu.:0.0000
##  Median :21.00   Median :0.0000                   Median :0.0000
##  Mean   :21.38   Mean   :0.1164                   Mean   :0.3421
##  3rd Qu.:25.00   3rd Qu.:0.0800                   3rd Qu.:1.0000
##  Max.   :57.00   Max.   :4.9300                   Max.   :1.0000
##
```

```r
summary(test.df)
```

```
##          Date       TempHighF        TempAvgF        TempLowF
##  01-02-2016:  1   Min.   : 36.0   Min.   :29.00   Min.   :22.00
##  01-02-2017:  1   1st Qu.: 73.0   1st Qu.:62.00   1st Qu.:51.00
##  01-05-2016:  1   Median : 83.0   Median :73.00   Median :62.00
##  01-08-2016:  1   Mean   : 81.2   Mean   :70.81   Mean   :59.92
##  01-10-2015:  1   3rd Qu.: 92.0   3rd Qu.:82.00   3rd Qu.:72.00
##  01-10-2016:  1   Max.   :104.0   Max.   :92.00   Max.   :81.00
##  (Other)   :384
##  DewPointHighF    DewPointAvgF    DewPointLowF   HumidityHighPercent
##  Min.   :15.00   Min.   : 8.00   Min.   : 2.00   Min.   : 44.00
##  1st Qu.:54.25   1st Qu.:47.00   1st Qu.:38.00   1st Qu.: 85.00
##  Median :66.00   Median :61.00   Median :55.00   Median : 91.00
##  Mean   :61.90   Mean   :56.91   Mean   :51.13   Mean   : 87.86
##  3rd Qu.:73.00   3rd Qu.:69.75   3rd Qu.:65.00   3rd Qu.: 94.00
##  Max.   :78.00   Max.   :74.00   Max.   :73.00   Max.   :100.00
##
##  HumidityAvgPercent HumidityLowPercent SeaLevelPressureHighInches
##  Min.   :27.00      Min.   :10.00      Min.   :29.65
##  1st Qu.:60.00      1st Qu.:33.00      1st Qu.:29.99
##  Median :67.00      Median :44.00      Median :30.08
##  Mean   :66.62      Mean   :44.94      Mean   :30.10
##  3rd Qu.:74.00      3rd Qu.:54.00      3rd Qu.:30.19
##  Max.   :97.00      Max.   :93.00      Max.   :30.80
##
##  SeaLevelPressureAvgInches SeaLevelPressureLowInches VisibilityHighMiles
##  Min.   :29.56             Min.   :29.41             Min.   : 5.000
##  1st Qu.:29.91             1st Qu.:29.81             1st Qu.:10.000
##  Median :30.00             Median :29.91             Median :10.000
##  Mean   :30.01             Mean   :29.92             Mean   : 9.987
##  3rd Qu.:30.10             3rd Qu.:30.01             3rd Qu.:10.000
##  Max.   :30.68             Max.   :30.50             Max.   :10.000
##
##  VisibilityAvgMiles VisibilityLowMiles  WindHighMPH      WindAvgMPH
##  Min.   : 2.000     Min.   : 0.000     Min.   : 6.00   Min.   : 1.000
##  1st Qu.: 9.000     1st Qu.: 2.000     1st Qu.:10.00   1st Qu.: 3.000
##  Median :10.000     Median : 9.000     Median :13.00   Median : 5.000
##  Mean   : 9.172     Mean   : 6.705     Mean   :13.28   Mean   : 4.987
##  3rd Qu.:10.000     3rd Qu.:10.000     3rd Qu.:15.00   3rd Qu.: 6.000
##  Max.   :10.000     Max.   :10.000     Max.   :25.00   Max.   :12.000
##
##   WindGustMPH   PrecipitationSumInches  Rain         RainP
##  Min.   : 9.0   Min.   :0.0000         no :257   Min.   :0.000
##  1st Qu.:17.0   1st Qu.:0.0000         yes:133   1st Qu.:0.000
##  Median :21.0   Median :0.0000                   Median :0.000
##  Mean   :21.4   Mean   :0.1445                   Mean   :0.341
##  3rd Qu.:25.0   3rd Qu.:0.0600                   3rd Qu.:1.000
##  Max.   :43.0   Max.   :5.2000                   Max.   :1.000
##
```

```
# Logistic regression

colnames(data1)
```

```
##  [1] "Date"                       "TempHighF"
##  [3] "TempAvgF"                   "TempLowF"
##  [5] "DewPointHighF"              "DewPointAvgF"
##  [7] "DewPointLowF"               "HumidityHighPercent"
##  [9] "HumidityAvgPercent"         "HumidityLowPercent"
## [11] "SeaLevelPressureHighInches" "SeaLevelPressureAvgInches"
## [13] "SeaLevelPressureLowInches"  "VisibilityHighMiles"
## [15] "VisibilityAvgMiles"         "VisibilityLowMiles"
## [17] "WindHighMPH"                "WindAvgMPH"
## [19] "WindGustMPH"                "PrecipitationSumInches"
## [21] "Rain"                       "RainP"
```

```r
model <- glm(Rain ~ TempHighF+TempAvgF+TempLowF+DewPointHighF+DewPointAvgF+DewPointLowF+HumidityHighPercent+
HumidityAvgPercent+HumidityLowPercent+SeaLevelPressureHighInches+SeaLevelPressureAvgInches+VisibilityLowMile
s+VisibilityHighMiles+VisibilityAvgMiles+WindGustMPH+WindHighMPH+WindAvgMPH, data = train.df, family = binom
ial)

summary(model)
```

```
##
## Call:
## glm(formula = Rain ~ TempHighF + TempAvgF + TempLowF + DewPointHighF +
##     DewPointAvgF + DewPointLowF + HumidityHighPercent + HumidityAvgPercent +
##     HumidityLowPercent + SeaLevelPressureHighInches + SeaLevelPressureAvgInches +
##     VisibilityLowMiles + VisibilityHighMiles + VisibilityAvgMiles +
##     WindGustMPH + WindHighMPH + WindAvgMPH, family = binomial,
##     data = train.df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7676  -0.4454  -0.1951   0.3632   2.7014
##
## Coefficients:
##                             Estimate Std. Error z value Pr(>|z|)
## (Intercept)                -40.99834   29.17451  -1.405   0.1599
## TempHighF                   -0.04739    0.20958  -0.226   0.8211
## TempAvgF                    -0.35795    0.41079  -0.871   0.3836
## TempLowF                     0.33203    0.20893   1.589   0.1120
## DewPointHighF                0.08541    0.03887   2.197   0.0280 *
## DewPointAvgF                 0.08855    0.06375   1.389   0.1648
## DewPointLowF                -0.05499    0.03454  -1.592   0.1114
## HumidityHighPercent         -0.09692    0.07935  -1.221   0.2219
## HumidityAvgPercent           0.13546    0.15352   0.882   0.3776
## HumidityLowPercent          -0.04748    0.07819  -0.607   0.5437
## SeaLevelPressureHighInches   2.36409    3.07149   0.770   0.4415
## SeaLevelPressureAvgInches   -1.10078    3.20509  -0.343   0.7313
## VisibilityLowMiles          -0.36397    0.05300  -6.868 6.52e-12 ***
## VisibilityHighMiles          0.22131    0.75491   0.293   0.7694
## VisibilityAvgMiles           0.29502    0.12463   2.367   0.0179 *
## WindGustMPH                  0.06403    0.05829   1.099   0.2719
## WindHighMPH                  0.25335    0.10057   2.519   0.0118 *
## WindAvgMPH                  -0.43272    0.08616  -5.023 5.10e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1175.60  on 914  degrees of freedom
## Residual deviance:  585.35  on 897  degrees of freedom
## AIC: 621.35
##
## Number of Fisher Scoring iterations: 6
```

```r
predicted_values <- predict(model, test.df[,-c(1,20,21,22)], type = "response")
head(predicted_values)
```

```
##           1           3           6           9          15          16
## 0.868356607 0.004696376 0.296961362 0.676644194 0.090897401 0.769032689
```

```r
# Validation

table(Rain)
```

```
## Rain
##  no yes
## 859 446
```

```r
nrows_prediction<-nrow(test.df)
prediction <- data.frame(c(1:nrows_prediction))
colnames(prediction) <- c("Rain")
str(prediction)
```

```
## 'data.frame':    390 obs. of  1 variable:
##  $ Rain: int  1 2 3 4 5 6 7 8 9 10 ...
```

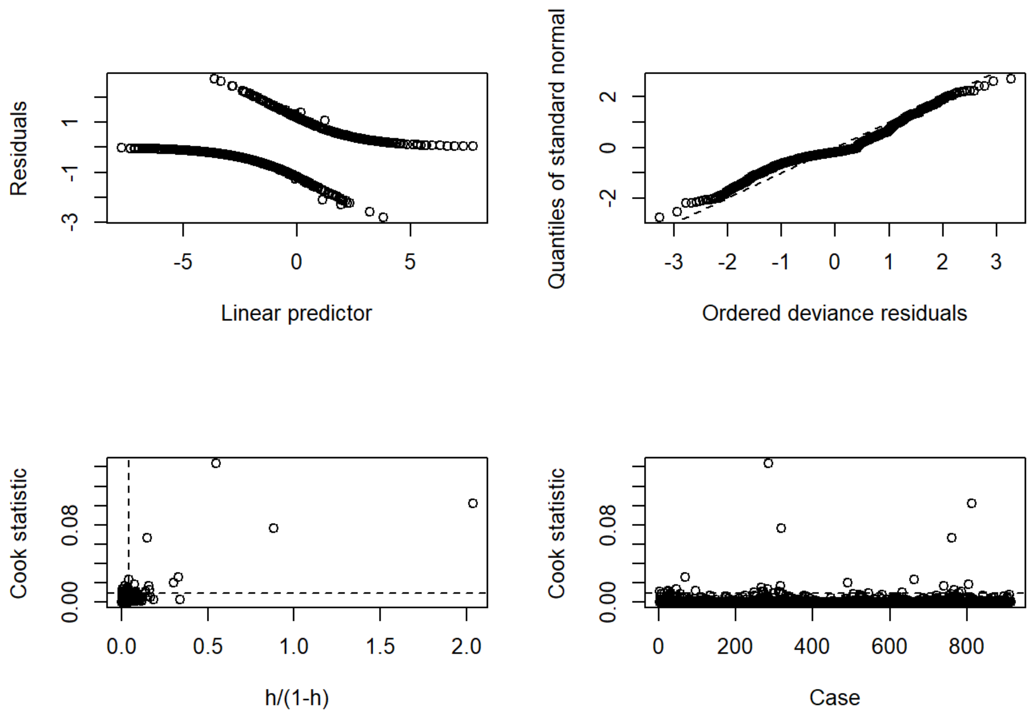
```r
prediction$Rain <- as.character(prediction$Rain)
prediction$Rain <- "yes"
prediction$Rain[ predicted_values < 0.5] <- "no"
prediction$Rain <- as.factor(prediction$Rain)

#Confusion Matrix

table(prediction$Rain, test.df$Rain)
```

```
##
##        no yes
##   no  230  31
##   yes  27 102
```

```r
confusionMatrix(prediction$Rain,test.df$Rain)
```

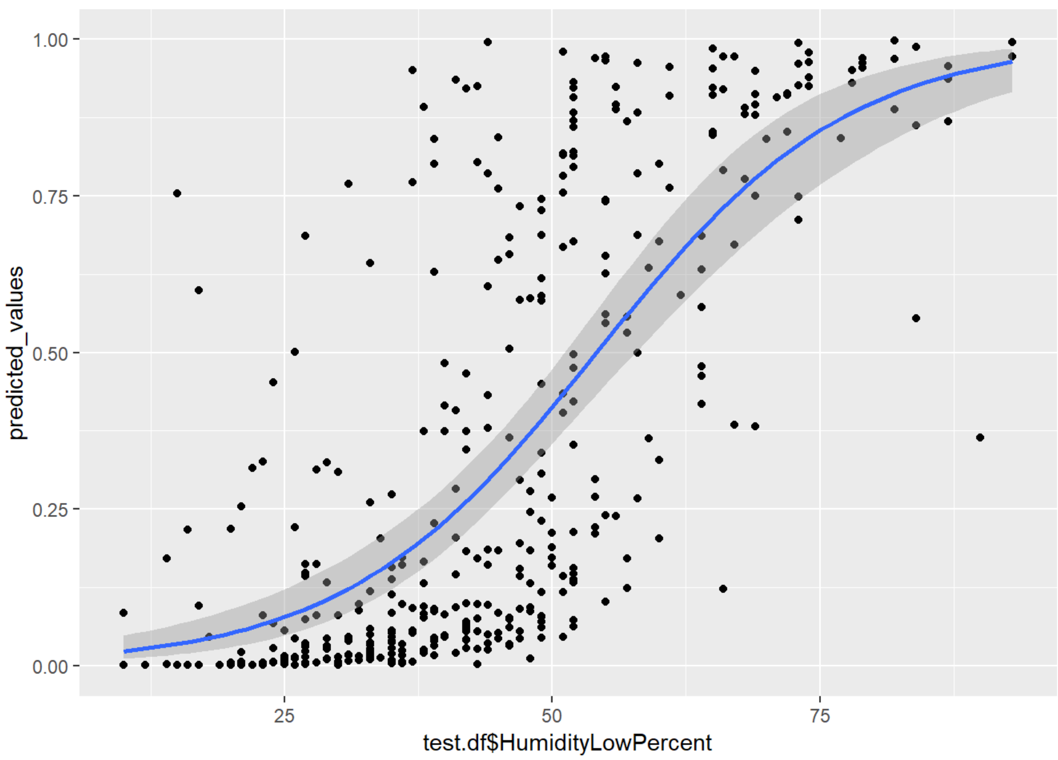
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  230  31
##        yes  27 102
##
##                Accuracy : 0.8513
##                  95% CI : (0.812, 0.8851)
##     No Information Rate : 0.659
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.6667
##  Mcnemar's Test P-Value : 0.6936
##
##             Sensitivity : 0.8949
##             Specificity : 0.7669
##          Pos Pred Value : 0.8812
##          Neg Pred Value : 0.7907
##              Prevalence : 0.6590
##          Detection Rate : 0.5897
##    Detection Prevalence : 0.6692
##       Balanced Accuracy : 0.8309
##
##        'Positive' Class : no
##
```

```r
glm.diag.plots(model)
```




```
ggplot(test.df, aes(x = test.df$HumidityLowPercent, y = predicted_values))+
  geom_point() + # add points
  geom_smooth(method = "glm", # plot a regression...
            method.args = list(family = "binomial"))
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial
## glm!
```



knn_CV.R

Magilan

Mon Oct 08 16:27:47 2018

```
library(caret)
```

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

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

```
library(sp)
library(class)

# Data Input

data <- read.csv("C:/Users/Magilan/Desktop/ML_project/austin_weather.csv",header = TRUE)
data1=na.omit(data,invert=FALSE)
attach(data1)

# Scalling the Data

standardized.X=scale(data1[,-c(1,20,21,22)])

# Data Partitioning

index <- createDataPartition(Rain, p = 0.7, list = FALSE)
train.X=standardized.X[index,]
test.X=standardized.X[-index,]
train.Y=Rain[index]
test.Y=Rain[-index]

# Knn Model

knn.pred=knn(train.X,test.X,train.Y,k=1)
head(data.frame(knn.pred,test.Y))
```

```
##   knn.pred test.Y
## 1      yes    yes
## 2       no     no
## 3       no    yes
## 4       no     no
## 5       no     no
## 6      yes    yes
```

```
confusionMatrix(knn.pred,test.Y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  227  58
##        yes  30  75
##
##                Accuracy : 0.7744
##                  95% CI : (0.7296, 0.8149)
##     No Information Rate : 0.659
##     P-Value [Acc > NIR] : 4.51e-07
##
##                   Kappa : 0.4711
##  Mcnemar's Test P-Value : 0.003999
##
##             Sensitivity : 0.8833
##             Specificity : 0.5639
##          Pos Pred Value : 0.7965
##          Neg Pred Value : 0.7143
##              Prevalence : 0.6590
##          Detection Rate : 0.5821
##    Detection Prevalence : 0.7308
##       Balanced Accuracy : 0.7236
##
##        'Positive' Class : no
##
```

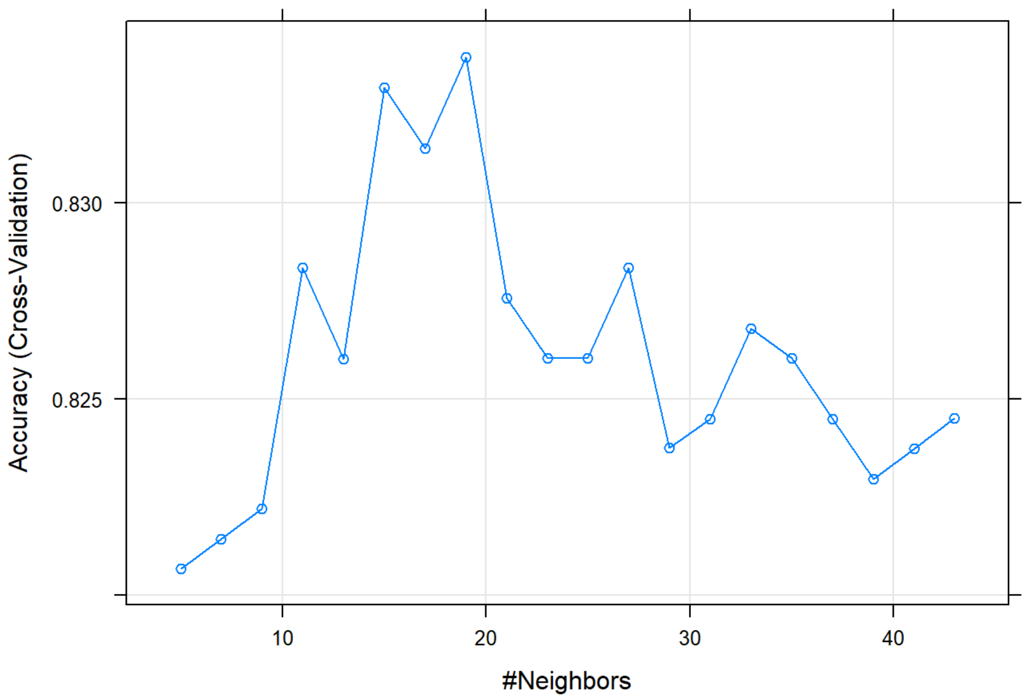
```
knn.pred1=knn(train.X,test.X,train.Y,k=2)
confusionMatrix(knn.pred1,test.Y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  221  53
##        yes  36  80
##
##                Accuracy : 0.7718
##                  95% CI : (0.7269, 0.8125)
##     No Information Rate : 0.659
##     P-Value [Acc > NIR] : 8.062e-07
##
##                   Kappa : 0.4761
##  Mcnemar's Test P-Value : 0.08989
##
##             Sensitivity : 0.8599
##             Specificity : 0.6015
##          Pos Pred Value : 0.8066
##          Neg Pred Value : 0.6897
##              Prevalence : 0.6590
##          Detection Rate : 0.5667
##    Detection Prevalence : 0.7026
##       Balanced Accuracy : 0.7307
##
##        'Positive' Class : no
##
```

```
knn.pred2=knn(train.X,test.X,train.Y,k=100)
confusionMatrix(knn.pred2,test.Y)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction  no yes
##        no  238  66
##        yes  19  67
##                                           
##                Accuracy : 0.7821          
##                  95% CI : (0.7377, 0.822) 
##     No Information Rate : 0.659           
##     P-Value [Acc > NIR] : 7.248e-08       
##                                           
##                   Kappa : 0.4699          
##  Mcnemar's Test P-Value : 6.057e-07       
##                                           
##             Sensitivity : 0.9261          
##             Specificity : 0.5038          
##          Pos Pred Value : 0.7829          
##          Neg Pred Value : 0.7791          
##              Prevalence : 0.6590          
##          Detection Rate : 0.6103          
##    Detection Prevalence : 0.7795          
##       Balanced Accuracy : 0.7149          
##                                           
##        'Positive' Class : no              
## 
```

```
# Cross Validation to find the value of K with highest Accuracy

tr=cbind(standardized.X,Rain)

model <- train(
  Rain ~., data = data1[,-c(1,20,22)], method = "knn",
  trControl = trainControl("cv", number = 10),
  preProcess = c("center","scale"),
  tuneLength = 20
)
plot(model)
```




```
k=model$bestTune
k
```

```
##    k
## 8 19
```

```
knn.pred3=knn(train.X,test.X,train.Y,k= model$bestTune)
confusionMatrix(knn.pred3,test.Y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  237  54
##        yes  20  79
##
##                Accuracy : 0.8103
##                  95% CI : (0.7678, 0.848)
##     No Information Rate : 0.659
##     P-Value [Acc > NIR] : 2.799e-11
##
##                   Kappa : 0.5501
##  Mcnemar's Test P-Value : 0.000125
##
##             Sensitivity : 0.9222
##             Specificity : 0.5940
##          Pos Pred Value : 0.8144
##          Neg Pred Value : 0.7980
##              Prevalence : 0.6590
##          Detection Rate : 0.6077
##    Detection Prevalence : 0.7462
##       Balanced Accuracy : 0.7581
##
##        'Positive' Class : no
##
```

pca.R

Magilan

Mon Oct 08 15:05:45 2018

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.0.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.6
## v tidyr   0.8.1     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0
```

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

```
library(boot)
library(forecast)
library(tseries)
library(caret)
```

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

```
##
## Attaching package: 'lattice'
```

```
## The following object is masked from 'package:boot':
##
##     melanoma
```

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

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

```
library(ROCR)
```

```
## Loading required package: gplots
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
library(psych)
```

```
##
## Attaching package: 'psych'
```

```
## The following object is masked from 'package:boot':
##
##     logit
```

```
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```r
library(devtools)
library(ggbiplot)
```

```
## Loading required package: plyr
```

```
## ------------------------------------------------------------------------------
```

```
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)
```

```
## ------------------------------------------------------------------------------
```

```
##
## Attaching package: 'plyr'
```

```
## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

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

```
## Loading required package: scales
```

```
##
## Attaching package: 'scales'
```

```
## The following objects are masked from 'package:psych':
##
##     alpha, rescale
```

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

```
## The following object is masked from 'package:readr':
##
##     col_factor
```

```
## Loading required package: grid
```

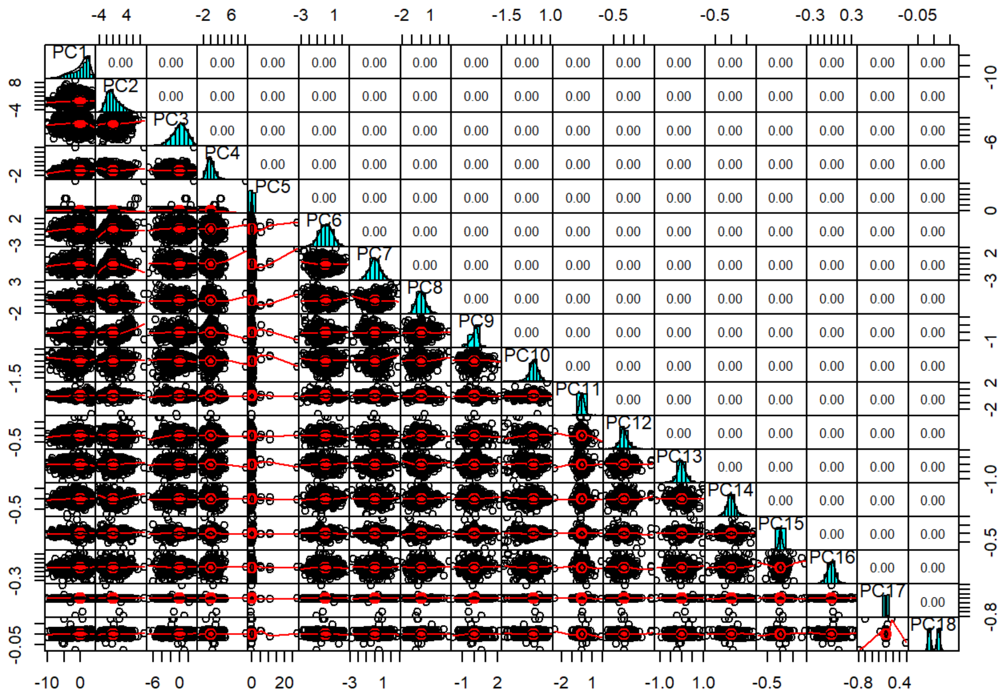
```r
library(sp)
library(class)


data <- read.csv("C:/Users/Magilan/Desktop/ML_project/austin_weather.csv",header = TRUE)
data1=na.omit(data,invert=FALSE)
attach(data1)

# Principal Component analysis

pc = prcomp(data1[,-c(1,20,21,22)],
            center=TRUE,
            scale. = TRUE)
pc$center
```

```
##                 TempHighF                  TempAvgF
##                 80.792337                 70.557854
##                  TempLowF              DewPointHighF
##                 59.819923                 61.516475
##               DewPointAvgF               DewPointLowF
##                 56.636782                 50.944061
##        HumidityHighPercent         HumidityAvgPercent
##                 87.833716                 66.662835
##         HumidityLowPercent SeaLevelPressureHighInches
##                 44.983908                 30.112337
##  SeaLevelPressureAvgInches  SeaLevelPressureLowInches
##                 30.022835                 29.931609
##        VisibilityHighMiles         VisibilityAvgMiles
##                  9.991571                  9.162452
##         VisibilityLowMiles                WindHighMPH
##                  6.842912                 13.245211
##                WindAvgMPH                WindGustMPH
##                  5.009195                 21.383908
```

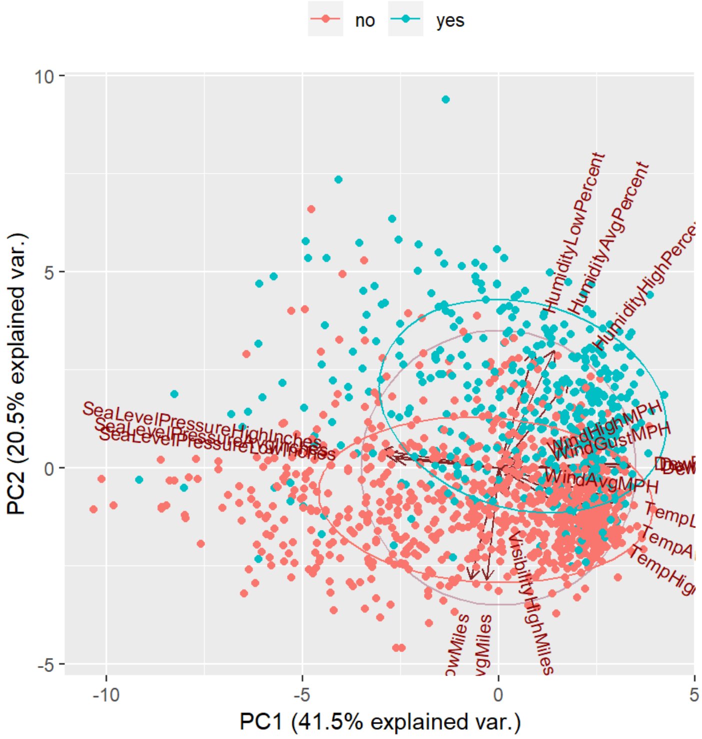
```r
summary(pc)
```

```
## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6
## Standard deviation     2.7336 1.9211 1.6574 1.09700 0.98168 0.81429
## Proportion of Variance 0.4152 0.2050 0.1526 0.06686 0.05354 0.03684
## Cumulative Proportion  0.4152 0.6202 0.7728 0.83967 0.89321 0.93005
##                            PC7    PC8     PC9    PC10    PC11    PC12
## Standard deviation     0.69570 0.5597 0.40987 0.31203 0.25950 0.21910
## Proportion of Variance 0.02689 0.0174 0.00933 0.00541 0.00374 0.00267
## Cumulative Proportion  0.95693 0.9743 0.98367 0.98908 0.99282 0.99549
##                           PC13    PC14    PC15    PC16    PC17    PC18
## Standard deviation     0.20639 0.14985 0.09830 0.07064 0.03619 0.01485
## Proportion of Variance 0.00237 0.00125 0.00054 0.00028 0.00007 0.00001
## Cumulative Proportion  0.99785 0.99910 0.99964 0.99992 0.99999 1.00000
```

```r
# Orthogonality of PC

pairs.panels(pc$x,gap=0,pch=21)
```



```
g <- ggbiplot(pc,
              obs.scale = 1,
              var.scale = 1,
              groups = data1$Rain,
              ellipse = TRUE,
              circle = TRUE,
              ellipse.prob = 0.68)
g <- g + scale_color_discrete(name = '')
g <- g + theme(legend.direction = 'horizontal',
               legend.position = 'top')
print(g)
```



```r
pc.df=data.frame(pc$x)

index <- createDataPartition(Rain, p = 0.7, list = FALSE)
# Training set
train.df <- pc.df[index,]
train.Y = data1[index,22]
train.Y1 = data1[index,21]
train = cbind(train.df,train.Y)

# Testing dataset
test.df <- pc.df[-index,]
test.Y = data1[-index,22]
test.Y1 =data1[-index,21]
test = cbind(test.df,test.Y)

# Logistic Regression With PCA

model <- glm(train$train.Y ~. , data = train)
summary(model)
```

```
##
## Call:
## glm(formula = train$train.Y ~ ., data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.0002  -0.1913  -0.0299   0.1747   0.9579
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.341247   0.011161  30.575  < 2e-16 ***
## PC1          0.025835   0.004132   6.253 6.23e-10 ***
## PC2          0.148393   0.005669  26.178  < 2e-16 ***
## PC3         -0.039171   0.006708  -5.839 7.33e-09 ***
## PC4         -0.020073   0.010145  -1.979  0.04817 *
## PC5         -0.011699   0.010288  -1.137  0.25578
## PC6         -0.088247   0.014063  -6.275 5.43e-10 ***
## PC7         -0.032025   0.016442  -1.948  0.05176 .
## PC8          0.147705   0.020477   7.213 1.16e-12 ***
## PC9         -0.142736   0.027550  -5.181 2.73e-07 ***
## PC10        -0.078385   0.035364  -2.217  0.02691 *
## PC11        -0.011828   0.042141  -0.281  0.77903
## PC12        -0.195926   0.050709  -3.864  0.00012 ***
## PC13         0.229570   0.053763   4.270 2.16e-05 ***
## PC14        -0.036846   0.072598  -0.508  0.61191
## PC15        -0.033466   0.118913  -0.281  0.77844
## PC16        -0.040601   0.157613  -0.258  0.79678
## PC17        -0.118198   0.266033  -0.444  0.65693
## PC18        -0.578076   0.745721  -0.775  0.43843
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.113517)
##
##     Null deviance: 205.93  on 914  degrees of freedom
## Residual deviance: 101.71  on 896  degrees of freedom
## AIC: 626.6
##
## Number of Fisher Scoring iterations: 2
```

```r
predicted_values <- predict(model, test.df, type = "response")
head(predicted_values)
```

```
##            1            3           22           24           28
##  0.849981638 -0.071333565  0.276149681  0.331586724  0.140525906
##           32
## -0.008151737
```

```r
#Vlaidation
table(Rain)
```

```
## Rain
##  no yes
## 859 446
```

```r
nrows_prediction<-nrow(test.df)
prediction <- data.frame(c(1:nrows_prediction))
colnames(prediction) <- c("Rain")
str(prediction)
```

```
## 'data.frame':    390 obs. of  1 variable:
##  $ Rain: int  1 2 3 4 5 6 7 8 9 10 ...
```

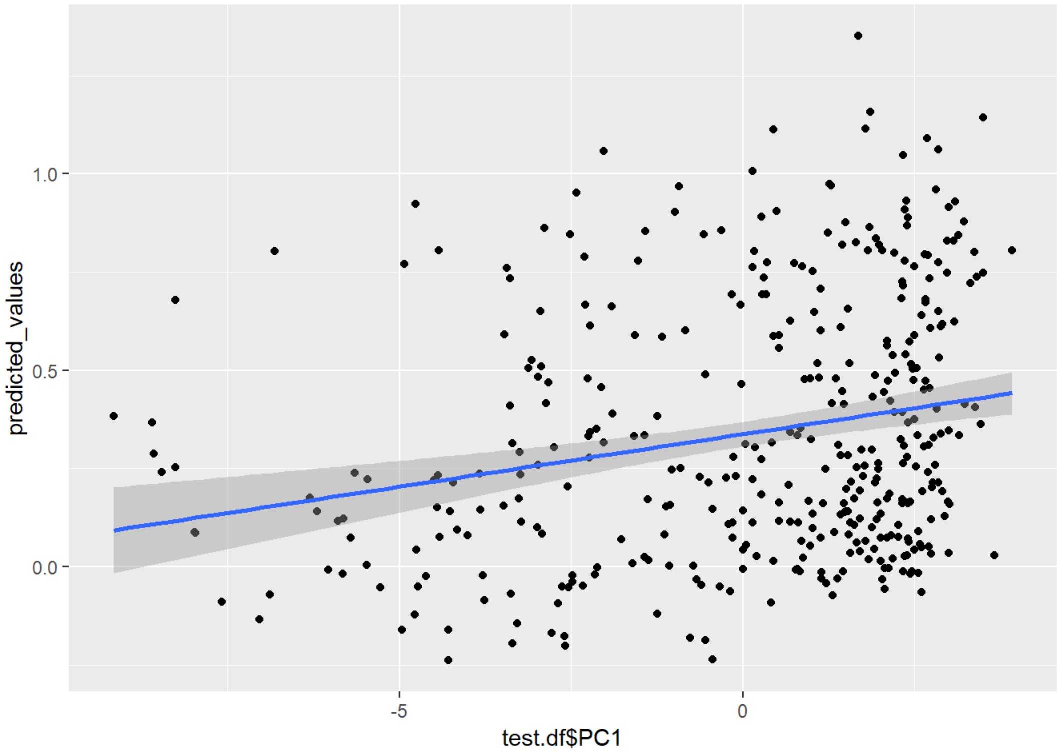
```r
prediction$Rain <- as.character(prediction$Rain)
prediction$Rain <- "yes"
prediction$Rain[ predicted_values < 0.5] <- "no"
prediction$Rain <- as.factor(prediction$Rain)

#Confusion Matrix

table(prediction$Rain, test.Y1)
```

```
##      test.Y1
##        no yes
##   no  232  40
##   yes  25  93
```

```r
confusionMatrix(prediction$Rain,test.Y1)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction  no yes
##        no  232  40
##        yes  25  93
## 
##                Accuracy : 0.8333
##                  95% CI : (0.7926, 0.869)
##     No Information Rate : 0.659
##     P-Value [Acc > NIR] : 1.045e-14
## 
##                   Kappa : 0.6188
##  Mcnemar's Test P-Value : 0.08248
## 
##             Sensitivity : 0.9027
##             Specificity : 0.6992
##          Pos Pred Value : 0.8529
##          Neg Pred Value : 0.7881
##              Prevalence : 0.6590
##          Detection Rate : 0.5949
##    Detection Prevalence : 0.6974
##       Balanced Accuracy : 0.8010
## 
##        'Positive' Class : no
## 
```

```r
#Plotting

ggplot(test, aes(x = test.df$PC1, y = predicted_values))+
  geom_point() + # add points
  geom_smooth(method = "lm", # plot a regression...
              method.args = list())
```



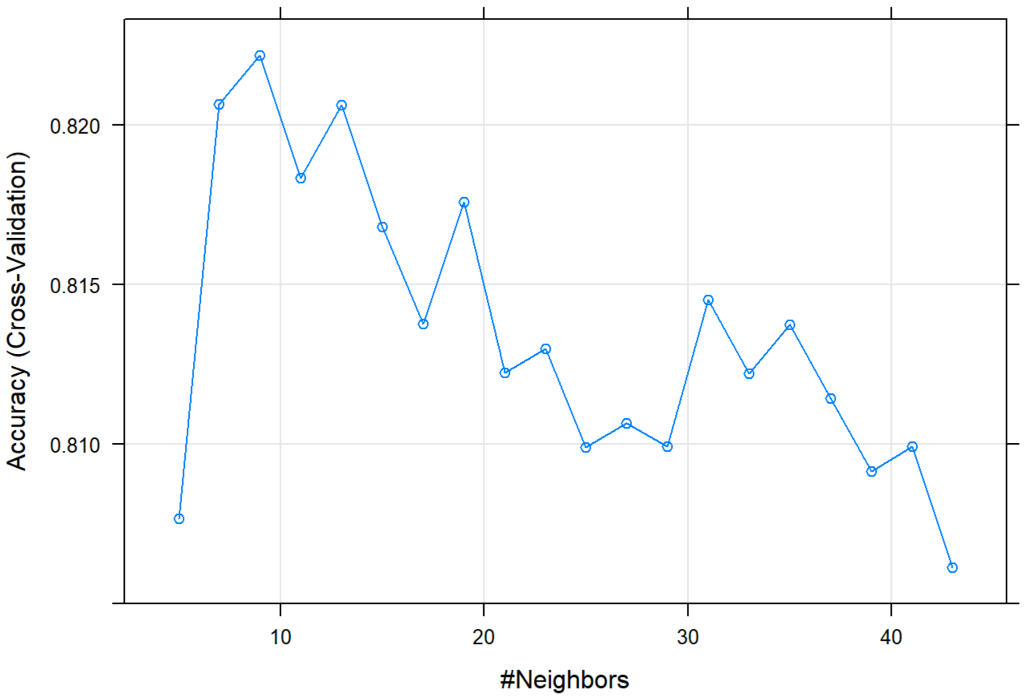

```
# KNN After PCA


model.knn = knn(train.df,test.df,train.Y1,k=1)
head(data.frame(model.knn,test.Y1))
```

```
##   model.knn test.Y1
## 1       yes     yes
## 2        no      no
## 3        no      no
## 4        no      no
## 5        no      no
## 6        no      no
```

```
confusionMatrix(model.knn,test.Y1)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  219  50
##        yes  38  83
##
##                Accuracy : 0.7744
##                  95% CI : (0.7296, 0.8149)
##     No Information Rate : 0.659
##     P-Value [Acc > NIR] : 4.51e-07
##
##                   Kappa : 0.4868
##  Mcnemar's Test P-Value : 0.241
##
##             Sensitivity : 0.8521
##             Specificity : 0.6241
##          Pos Pred Value : 0.8141
##          Neg Pred Value : 0.6860
##              Prevalence : 0.6590
##          Detection Rate : 0.5615
##    Detection Prevalence : 0.6897
##       Balanced Accuracy : 0.7381
##
##        'Positive' Class : no
##
```

```
tr=cbind(pc.df,Rain)

model.cv <- train(
  Rain ~., data = tr, method = "knn",
  trControl = trainControl("cv", number = 10),
  preProcess = c("center","scale"),
  tuneLength = 20
)
plot(model.cv)
```




```
K=model.cv$bestTune
K
```

```
##   k
## 3 9
```

```
model.knn = knn(train.df,test.df,train.Y1,k=K)
head(data.frame(model.knn,test.Y1))
```

```
##   model.knn test.Y1
## 1       yes     yes
## 2        no      no
## 3        no      no
## 4        no      no
## 5        no      no
## 6        no      no
```

```
confusionMatrix(model.knn,test.Y1)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  235  53
##        yes  22  80
##
##                Accuracy : 0.8077
##                  95% CI : (0.765, 0.8456)
##     No Information Rate : 0.659
##     P-Value [Acc > NIR] : 6.184e-11
##
##                   Kappa : 0.5466
##  Mcnemar's Test P-Value : 0.000532
##
##             Sensitivity : 0.9144
##             Specificity : 0.6015
##          Pos Pred Value : 0.8160
##          Neg Pred Value : 0.7843
##              Prevalence : 0.6590
##          Detection Rate : 0.6026
##    Detection Prevalence : 0.7385
##       Balanced Accuracy : 0.7580
##
##        'Positive' Class : no
##
```

decision_tree.R

Magilan

Mon Oct 08 16:17:50 2018

```
library(tree)
library(rpart)
library(rpart.plot)
library(caret)
```

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

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

```
library(bst)
```

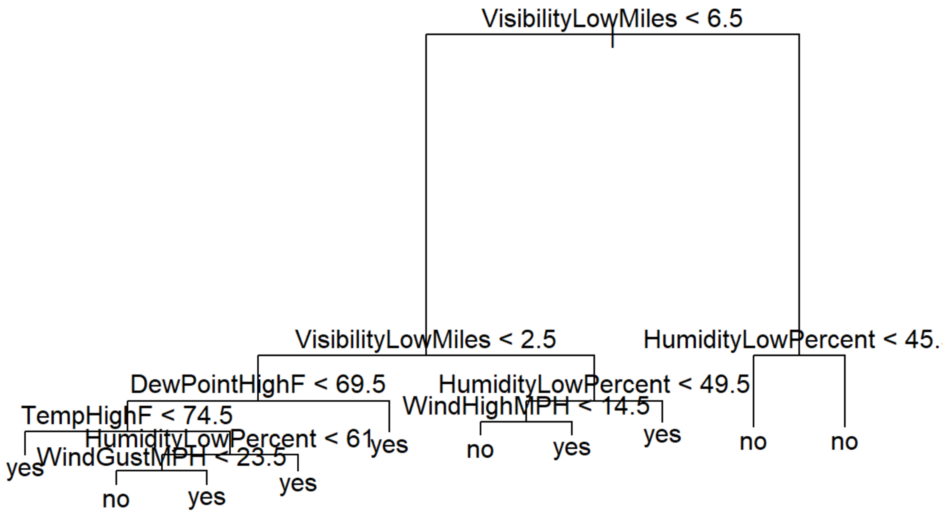
```
## Loading required package: gbm
```

```
## Loaded gbm 2.1.4
```

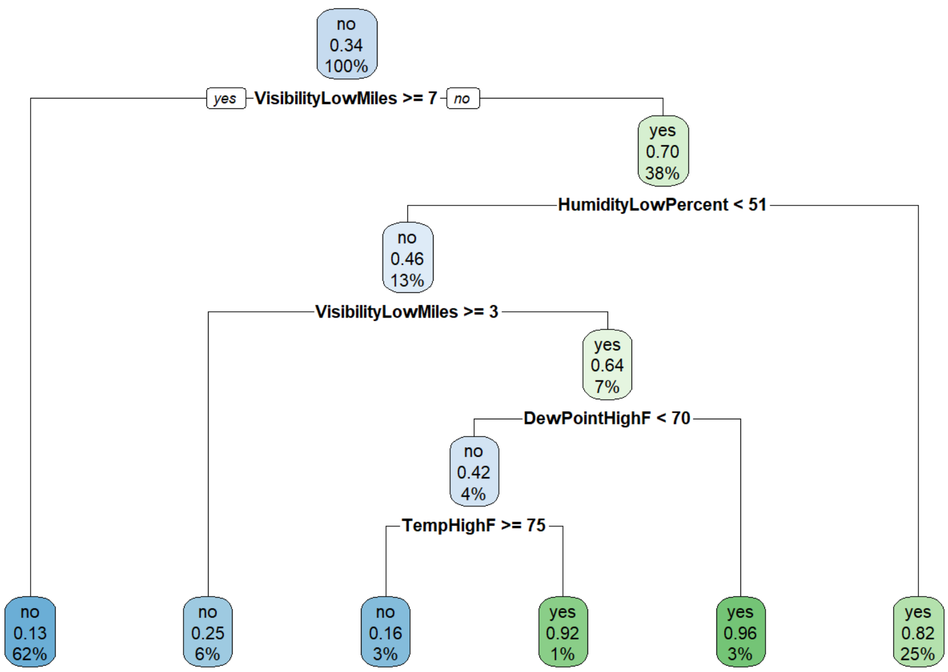
```
#Data Input

data <- read.csv("C:/Users/Magilan/Desktop/ML_project/austin_weather.csv",header = TRUE)
data1=na.omit(data,invert=FALSE)
attach(data1)

data2=data1[,-c(1,20,22)]
tree.model =tree(Rain ~. , data2,method = "class" )
summary(tree.model)
```

```
##
## Classification tree:
## tree(formula = Rain ~ ., data = data2, method = "class")
## Variables actually used in tree construction:
## [1] "VisibilityLowMiles" "DewPointHighF"      "TempHighF"
## [4] "HumidityLowPercent" "WindGustMPH"        "WindHighMPH"
## Number of terminal nodes:  10
## Residual mean deviance:  0.703 = 910.4 / 1295
## Misclassification error rate: 0.1479 = 193 / 1305
```

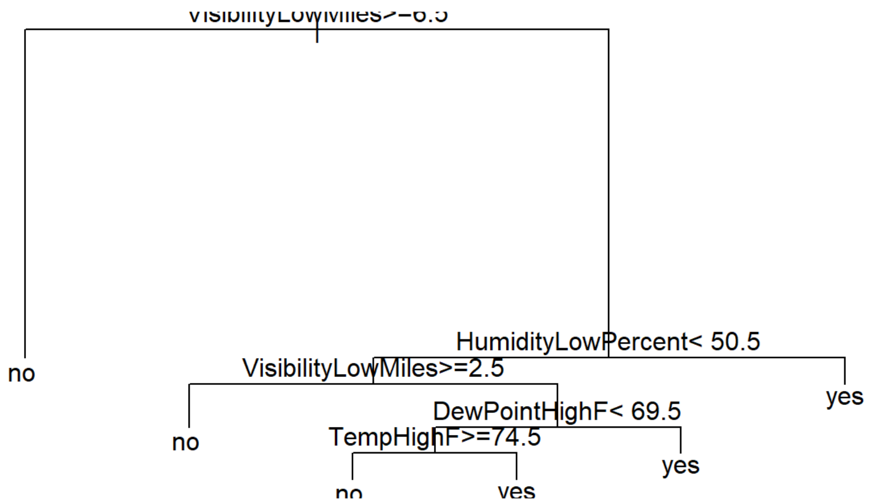
```
plot(tree.model )
text(tree.model ,pretty =0)
```




```
# Train And Test Data

index <- createDataPartition(Rain, p = 0.7, list = FALSE)
train = data1[index,-c(1,20,22)]
test = data1[-index,-c(1,20,22)]
test.Y = Rain[-index]

# Tree Model

tree.model1 = rpart(Rain ~ . ,data = train, method = "class")
rpart.plot(tree.model1)
```




```
plot(tree.model1)
text(tree.model1,pretty = 0)
```




```
tree.pred = predict(tree.model1 ,test, type = "class")
table(tree.pred,test.Y)
```

```
##          test.Y
## tree.pred  no yes
##       no  232  44
##       yes  25  89
```

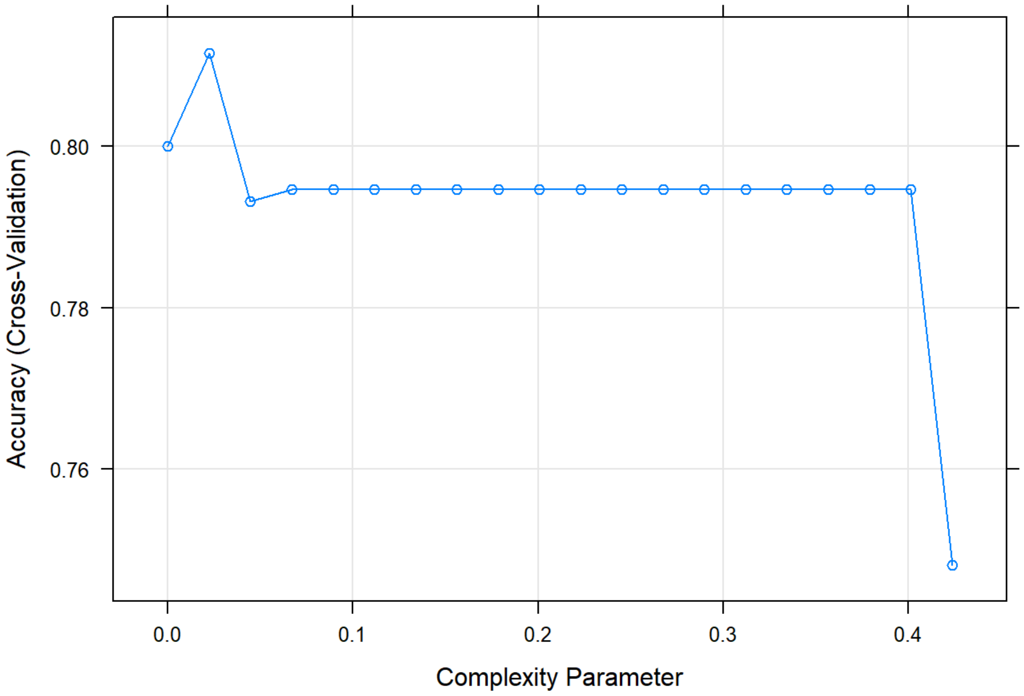
```
confusionMatrix(tree.pred,test.Y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  232  44
##        yes  25  89
##
##                Accuracy : 0.8231
##                  95% CI : (0.7815, 0.8597)
##     No Information Rate : 0.659
##     P-Value [Acc > NIR] : 4.143e-13
##
##                   Kappa : 0.5923
##  Mcnemar's Test P-Value : 0.03024
##
##             Sensitivity : 0.9027
##             Specificity : 0.6692
##          Pos Pred Value : 0.8406
##          Neg Pred Value : 0.7807
##              Prevalence : 0.6590
##          Detection Rate : 0.5949
##    Detection Prevalence : 0.7077
##       Balanced Accuracy : 0.7859
##
##        'Positive' Class : no
##
```

```r
# Cross Validation

model <- train(
  Rain ~., data = data1[,-c(1,20,22)], method = "rpart",
  trControl = trainControl("cv", number = 10),
  preProcess = c("center","scale"),
  tuneLength = 20
  )
model
```

```
## CART
##
## 1305 samples
##   18 predictor
##    2 classes: 'no', 'yes'
##
## Pre-processing: centered (18), scaled (18)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1176, 1175, 1174, 1174, 1174, 1174, ...
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.00000000  0.8000565  0.5500857
##   0.02230352  0.8115894  0.5621310
##   0.04460703  0.7931803  0.5388793
##   0.06691055  0.7947070  0.5452138
##   0.08921407  0.7947070  0.5452138
##   0.11151758  0.7947070  0.5452138
##   0.13382110  0.7947070  0.5452138
##   0.15612462  0.7947070  0.5452138
##   0.17842813  0.7947070  0.5452138
##   0.20073165  0.7947070  0.5452138
##   0.22303517  0.7947070  0.5452138
##   0.24533868  0.7947070  0.5452138
##   0.26764220  0.7947070  0.5452138
##   0.28994572  0.7947070  0.5452138
##   0.31224923  0.7947070  0.5452138
##   0.33455275  0.7947070  0.5452138
##   0.35685627  0.7947070  0.5452138
##   0.37915978  0.7947070  0.5452138
##   0.40146330  0.7947070  0.5452138
##   0.42376682  0.7480188  0.3657507
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.02230352.
```

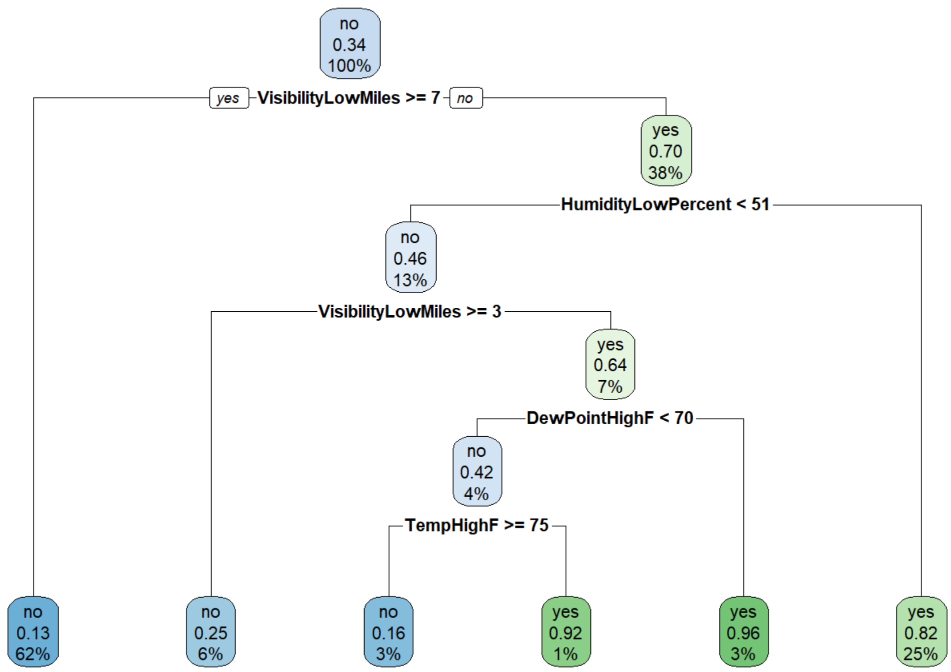
```r
plot(model)
```



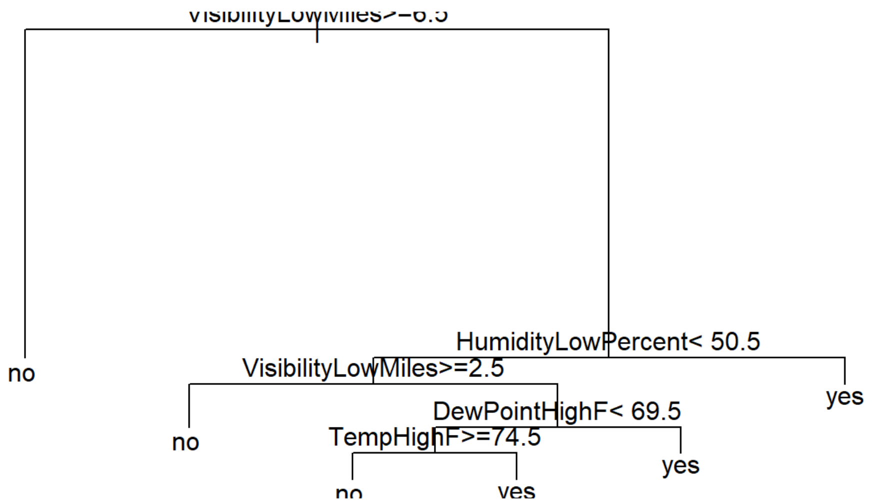

```
k=model$bestTune
k
```

```
##            cp
## 2 0.02230352
```

```
# Prunning

ptree<-  prune(tree.model1,cp=0.022303)
rpart.plot(ptree)
```




```
plot(ptree)
text(ptree,pretty = 0)
```



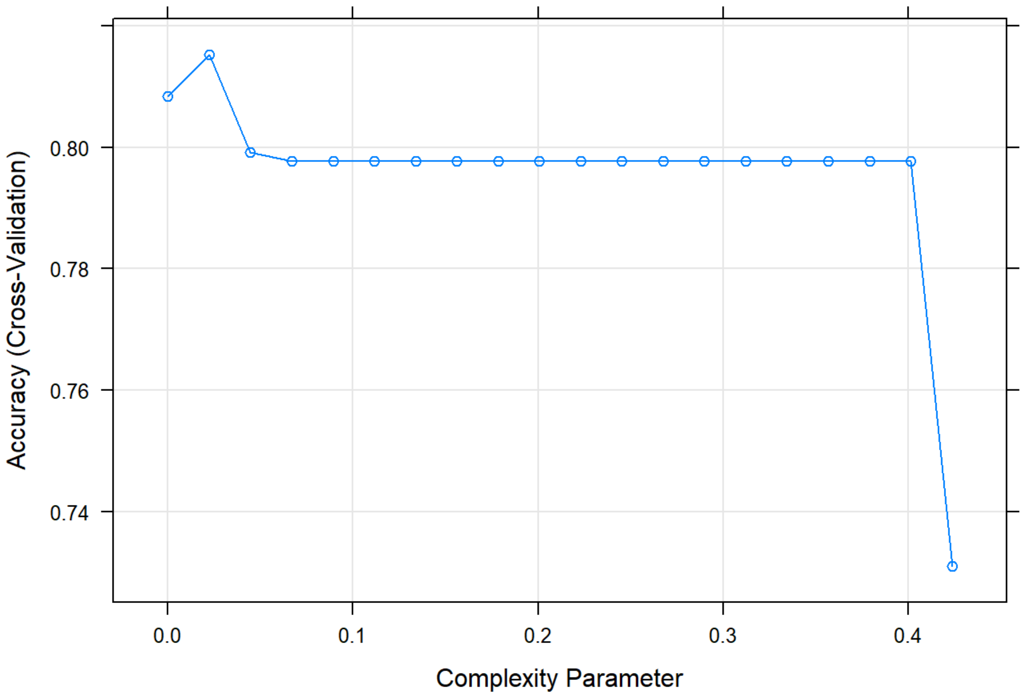

```r
ptree.pred = predict(ptree ,test, type = "class")
table(ptree.pred,test.Y)
```

```
##           test.Y
## ptree.pred  no yes
##        no  232  44
##        yes  25  89
```

```r
confusionMatrix(ptree.pred,test.Y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  232  44
##        yes  25  89
##
##                Accuracy : 0.8231
##                  95% CI : (0.7815, 0.8597)
##     No Information Rate : 0.659
##     P-Value [Acc > NIR] : 4.143e-13
##
##                   Kappa : 0.5923
##  Mcnemar's Test P-Value : 0.03024
##
##             Sensitivity : 0.9027
##             Specificity : 0.6692
##          Pos Pred Value : 0.8406
##          Neg Pred Value : 0.7807
##              Prevalence : 0.6590
##          Detection Rate : 0.5949
##    Detection Prevalence : 0.7077
##       Balanced Accuracy : 0.7859
##
##        'Positive' Class : no
##
```

```r
# Using Gini Indexing

model1 <- train(
  Rain ~., data = data1[,-c(1,20,22)],parms = list(split = "gini"),
  method = "rpart",
  trControl = trainControl("cv", number = 10),
  preProcess = c("center","scale"),
  tuneLength = 20
)
plot(model1)
```




```r
model1$bestTune
```

```
##           cp
## 2 0.02230352
```

```r
tree.pred.gini = predict(model1 ,test)
table(tree.pred.gini,test.Y)
```

```
##                test.Y
## tree.pred.gini  no yes
##            no  228  39
##            yes  29  94
```

```r
confusionMatrix(tree.pred.gini,test.Y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  228  39
##        yes  29  94
##
##                Accuracy : 0.8256
##                  95% CI : (0.7843, 0.862)
##     No Information Rate : 0.659
##     P-Value [Acc > NIR] : 1.695e-13
##
##                   Kappa : 0.6049
##  Mcnemar's Test P-Value : 0.2751
##
##             Sensitivity : 0.8872
##             Specificity : 0.7068
##          Pos Pred Value : 0.8539
##          Neg Pred Value : 0.7642
##              Prevalence : 0.6590
##          Detection Rate : 0.5846
##    Detection Prevalence : 0.6846
##       Balanced Accuracy : 0.7970
##
##        'Positive' Class : no
##
```

random_forest.R

Magilan

Mon Oct 08 17:27:38 2018

```
library(party)
```

```
## Loading required package: grid
```

```
## Loading required package: mvtnorm
```

```
## Loading required package: modeltools
```

```
## Loading required package: stats4
```

```
## Loading required package: strucchange
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Loading required package: sandwich
```

```
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
library(caret)
```

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

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

```
##
## Attaching package: 'ggplot2'
```

```
## The following object is masked from 'package:randomForest':
##
##     margin
```

```r
# Data Input

data <- read.csv("C:/Users/Magilan/Desktop/ML_project/austin_weather.csv",header = TRUE)
data1=na.omit(data,invert=FALSE)
attach(data1)

# Data Partitioning

index <- createDataPartition(Rain, p = 0.7, list = FALSE)
train.df <- data1[index,-c(1,20,22)]
test.df <- data1[-index,-c(1,20,21,22)]
test.Y <- data1[-index,21]

# Random Forest

model.rf = randomForest(Rain ~ ., data= train.df)

pred <- predict(model.rf, test.df, type ="response")
head(pred)
```

```
##   1   6  11  18  19  22
## yes  no  no  no yes  no
## Levels: no yes
```

```r
confusionMatrix(pred,test.Y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  233  35
##        yes  24  98
##
##                Accuracy : 0.8487
##                  95% CI : (0.8092, 0.8828)
##     No Information Rate : 0.659
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.6566
##  Mcnemar's Test P-Value : 0.193
##
##             Sensitivity : 0.9066
##             Specificity : 0.7368
##          Pos Pred Value : 0.8694
##          Neg Pred Value : 0.8033
##              Prevalence : 0.6590
##          Detection Rate : 0.5974
##    Detection Prevalence : 0.6872
##       Balanced Accuracy : 0.8217
##
##        'Positive' Class : no
##
```

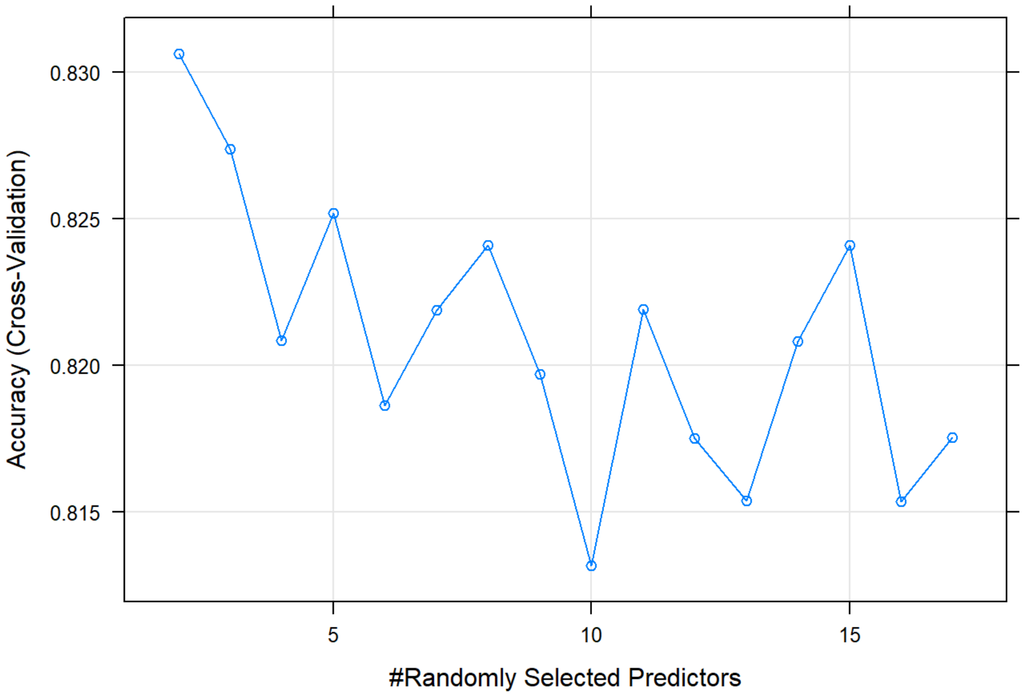
```r
# Cross Validation

model.rf <- train(
  Rain ~., data = train.df[,-c(1,20,22)], method = "rf",
  trControl = trainControl("cv", number = 10),
  preProcess = c("center","scale"),
  tuneLength = 20
)
```

```
## note: only 16 unique complexity parameters in default grid. Truncating the grid to 16 .
```

```r
model.rf
```

```
## Random Forest
##
## 915 samples
##  17 predictor
##   2 classes: 'no', 'yes'
##
## Pre-processing: centered (17), scaled (17)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 823, 823, 823, 823, 823, 824, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.8306259  0.6085034
##    3    0.8273650  0.6036578
##    4    0.8208313  0.5887250
##    5    0.8251911  0.5986349
##    6    0.8186335  0.5841256
##    7    0.8218705  0.5922942
##    8    0.8241042  0.5963846
##    9    0.8197086  0.5876869
##   10    0.8131629  0.5724501
##   11    0.8219183  0.5903565
##   12    0.8175227  0.5832722
##   13    0.8153727  0.5754351
##   14    0.8208194  0.5879672
##   15    0.8241042  0.5974383
##   16    0.8153488  0.5771795
##   17    0.8175466  0.5813555
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

```
plot(model.rf)
```




```
k=model.rf$bestTune
k
```

```
##   mtry
## 1    2
```

```r
pred.cv = predict(model.rf,test.df)
confusionMatrix(pred.cv,test.Y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  233  42
##        yes  24  91
##
##                Accuracy : 0.8308
##                  95% CI : (0.7898, 0.8666)
##     No Information Rate : 0.659
##     P-Value [Acc > NIR] : 2.692e-14
##
##                   Kappa : 0.6108
##  Mcnemar's Test P-Value : 0.03639
##
##             Sensitivity : 0.9066
##             Specificity : 0.6842
##          Pos Pred Value : 0.8473
##          Neg Pred Value : 0.7913
##              Prevalence : 0.6590
##          Detection Rate : 0.5974
##    Detection Prevalence : 0.7051
##       Balanced Accuracy : 0.7954
##
##        'Positive' Class : no
##
```

```r
model.rf1 = randomForest(Rain ~ ., data= train.df , mtry = 15)
pred1 <- predict(model.rf1, test.df, type ="response")
confusionMatrix(pred1,test.Y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  232  34
##        yes  25  99
##
##                Accuracy : 0.8487
##                  95% CI : (0.8092, 0.8828)
##     No Information Rate : 0.659
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.6578
##  Mcnemar's Test P-Value : 0.2976
##
##             Sensitivity : 0.9027
##             Specificity : 0.7444
##          Pos Pred Value : 0.8722
##          Neg Pred Value : 0.7984
##              Prevalence : 0.6590
##          Detection Rate : 0.5949
##    Detection Prevalence : 0.6821
##       Balanced Accuracy : 0.8235
##
##        'Positive' Class : no
##
```

svm_CV.R

Magilan

Mon Oct 08 22:48:08 2018

```r
library(e1071)
library(caret)
```

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

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

```r
# Data Input

data <- read.csv("C:/Users/Magilan/Desktop/ML_project/austin_weather.csv",header = TRUE)
data1=na.omit(data,invert=FALSE)
attach(data1)

# Data Partitioning

index <- createDataPartition(Rain, p = 0.7, list = FALSE)
train.df <- data1[index,-c(1,20,22)]
test.df <- data1[-index,-c(1,20,21,22)]
test.Y <- data1[-index,21]

# SVM Model with Linear Kernel

model.svm <- svm(Rain ~ . , data = train.df)

pred.svm <- predict(model.svm, test.df, type = "C-classification")
head(pred.svm)
```

```
##  2  4  5 10 12 13
## no no no no no no
## Levels: no yes
```

```r
confusionMatrix(pred.svm,test.Y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  232  39
##        yes  25  94
##
##                Accuracy : 0.8359
##                  95% CI : (0.7953, 0.8713)
##     No Information Rate : 0.659
##     P-Value [Acc > NIR] : 3.98e-15
##
##                   Kappa : 0.6254
##  Mcnemar's Test P-Value : 0.1042
##
##             Sensitivity : 0.9027
##             Specificity : 0.7068
##          Pos Pred Value : 0.8561
##          Neg Pred Value : 0.7899
##              Prevalence : 0.6590
##          Detection Rate : 0.5949
##    Detection Prevalence : 0.6949
##       Balanced Accuracy : 0.8047
##
##        'Positive' Class : no
##
```

```r
# Cross Validation

model.cv <- train(
  Rain ~., data = train.df[,-c(1,20,22)], method = "svmLinear",
  trControl = trainControl("cv", number = 10),
  preProcess = c("center","scale"),
  tuneLength = 20
)
model.cv
```

```
## Support Vector Machines with Linear Kernel
##
## 915 samples
##  17 predictor
##   2 classes: 'no', 'yes'
##
## Pre-processing: centered (17), scaled (17)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 824, 824, 823, 824, 823, 823, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.8502628  0.6619508
##
## Tuning parameter 'C' was held constant at a value of 1
```

```r
k=model.cv$bestTune
k
```

```
##   C
## 1 1
```

```r
pred.cv = predict(model.cv,test.df)
confusionMatrix(pred.cv,test.Y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  229  36
##        yes  28  97
##
##                Accuracy : 0.8359
##                  95% CI : (0.7953, 0.8713)
##     No Information Rate : 0.659
##     P-Value [Acc > NIR] : 3.98e-15
##
##                   Kappa : 0.6295
##  Mcnemar's Test P-Value : 0.3816
##
##             Sensitivity : 0.8911
##             Specificity : 0.7293
##          Pos Pred Value : 0.8642
##          Neg Pred Value : 0.7760
##              Prevalence : 0.6590
##          Detection Rate : 0.5872
##    Detection Prevalence : 0.6795
##       Balanced Accuracy : 0.8102
##
##        'Positive' Class : no
##
```

```r
# SVM AND RANDOM FOREST GIVES THE BEST ACCURACY APROX. 84.1%
```