SVM

Bishal Neupane, Saugat Gyawali

10/08/2022

Source:

https://www.kaggle.com/code/abhpasha/logistic-regression-predicting-rain-in-australia

Importing data and taking only first 15k because the data is two large more than 100k

```
df <- read.csv("weatherAUS.csv", header = TRUE)
df <- df[1: 15000,]
```

```
head(df)
```

```
##        Date Location MinTemp MaxTemp Rainfall Evaporation Sunshine WindGustDir
## 1 12/1/2008   Albury    13.4    22.9      0.6          NA       NA           W
## 2 12/2/2008   Albury     7.4    25.1      0.0          NA       NA         WNW
## 3 12/3/2008   Albury    12.9    25.7      0.0          NA       NA         WSW
## 4 12/4/2008   Albury     9.2    28.0      0.0          NA       NA          NE
## 5 12/5/2008   Albury    17.5    32.3      1.0          NA       NA           W
## 6 12/6/2008   Albury    14.6    29.7      0.2          NA       NA         WNW
##   WindGustSpeed WindDir9am WindDir3pm WindSpeed9am WindSpeed3pm Humidity9am
## 1            44          W        WNW           20           24          71
## 2            44        NNW        WSW            4           22          44
## 3            46          W        WSW           19           26          38
## 4            24         SE          E           11            9          45
## 5            41        ENE         NW            7           20          82
## 6            56          W          W           19           24          55
##   Humidity3pm Pressure9am Pressure3pm Cloud9am Cloud3pm Temp9am Temp3pm
## 1          22      1007.7      1007.1        8       NA    16.9    21.8
## 2          25      1010.6      1007.8       NA       NA    17.2    24.3
## 3          30      1007.6      1008.7       NA        2    21.0    23.2
## 4          16      1017.6      1012.8       NA       NA    18.1    26.5
## 5          33      1010.8      1006.0        7        8    17.8    29.7
## 6          23      1009.2      1005.4       NA       NA    20.6    28.9
##   RainToday RainTomorrow
## 1        No           No
## 2        No           No
## 3        No           No
## 4        No           No
## 5        No           No
## 6        No           No
```

#There are alot of column so removing columns with non numeric values.

```
df$Date<- NULL
df$WindGustDir<-NULL
df$WindGustDir <-NULL
df$WindDir3pm <- NULL
df$WindDir3pm <-NULL
df$Location <-NULL
df$Sunshine <-NULL
df$RainToday <- NULL
df$WindDir9am <-NULL
df$Evaporation <-NULL
```

Structure of Data Frame

```
str(df)
```

```
## 'data.frame':    15000 obs. of  15 variables:
##  $ MinTemp      : num  13.4 7.4 12.9 9.2 17.5 14.6 14.3 7.7 9.7 13.1 ...
##  $ MaxTemp      : num  22.9 25.1 25.7 28 32.3 29.7 25 26.7 31.9 30.1 ...
##  $ Rainfall     : num  0.6 0 0 0 1 0.2 0 0 0 1.4 ...
##  $ WindGustSpeed: int  44 44 46 24 41 56 50 35 80 28 ...
##  $ WindSpeed9am : int  20 4 19 11 7 19 20 6 7 15 ...
##  $ WindSpeed3pm : int  24 22 26 9 20 24 24 17 28 11 ...
##  $ Humidity9am  : int  71 44 38 45 82 55 49 48 42 58 ...
##  $ Humidity3pm  : int  22 25 30 16 33 23 19 19 9 27 ...
##  $ Pressure9am  : num  1008 1011 1008 1018 1011 ...
##  $ Pressure3pm  : num  1007 1008 1009 1013 1006 ...
##  $ Cloud9am     : int  8 NA NA NA 7 NA 1 NA NA NA ...
##  $ Cloud3pm     : int  NA NA 2 NA 8 NA NA NA NA NA ...
##  $ Temp9am      : num  16.9 17.2 21 18.1 17.8 20.6 18.1 16.3 18.3 20.1 ...
##  $ Temp3pm      : num  21.8 24.3 23.2 26.5 29.7 28.9 24.6 25.5 30.2 28.2 ...
##  $ RainTomorrow : chr  "No" "No" "No" "No" ...
```

Data Exploration

Names of Column

```
names(df)
```

```
##  [1] "MinTemp"       "MaxTemp"       "Rainfall"      "WindGustSpeed"
##  [5] "WindSpeed9am"  "WindSpeed3pm"  "Humidity9am"   "Humidity3pm"
##  [9] "Pressure9am"   "Pressure3pm"   "Cloud9am"      "Cloud3pm"
## [13] "Temp9am"       "Temp3pm"       "RainTomorrow"
```

Importing Package and using it to Change to factor

```
#install.packages("dplyr")
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
df <- mutate_if(df, is.character, as.factor)
```

Dimensions of df

```
dim(df)
```

```
## [1] 15000    15
```

```
str(df)
```

```
## 'data.frame':    15000 obs. of  15 variables:
##  $ MinTemp      : num  13.4 7.4 12.9 9.2 17.5 14.6 14.3 7.7 9.7 13.1 ...
##  $ MaxTemp      : num  22.9 25.1 25.7 28 32.3 29.7 25 26.7 31.9 30.1 ...
##  $ Rainfall     : num  0.6 0 0 0 1 0.2 0 0 0 1.4 ...
##  $ WindGustSpeed: int  44 44 46 24 41 56 50 35 80 28 ...
##  $ WindSpeed9am : int  20 4 19 11 7 19 20 6 7 15 ...
##  $ WindSpeed3pm : int  24 22 26 9 20 24 24 17 28 11 ...
##  $ Humidity9am  : int  71 44 38 45 82 55 49 48 42 58 ...
##  $ Humidity3pm  : int  22 25 30 16 33 23 19 19 9 27 ...
##  $ Pressure9am  : num  1008 1011 1008 1018 1011 ...
##  $ Pressure3pm  : num  1007 1008 1009 1013 1006 ...
##  $ Cloud9am     : int  8 NA NA NA 7 NA 1 NA NA NA ...
##  $ Cloud3pm     : int  NA NA 2 NA 8 NA NA NA NA NA ...
##  $ Temp9am      : num  16.9 17.2 21 18.1 17.8 20.6 18.1 16.3 18.3 20.1 ...
##  $ Temp3pm      : num  21.8 24.3 23.2 26.5 29.7 28.9 24.6 25.5 30.2 28.2 ...
##  $ RainTomorrow : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 2 1 ...
```

Statistics Summary of Each column

```
summary(df)
```

```
##     MinTemp          MaxTemp         Rainfall       WindGustSpeed
##  Min.   :-3.30   Min.   : 6.8   Min.   :  0.000   Min.   :  7.00
##  1st Qu.: 7.10   1st Qu.:19.4   1st Qu.:  0.000   1st Qu.: 28.00
##  Median :12.50   Median :24.4   Median :  0.000   Median : 35.00
##  Mean   :12.24   Mean   :24.7   Mean   :  2.393   Mean   : 36.43
##  3rd Qu.:17.40   3rd Qu.:29.5   3rd Qu.:  0.200   3rd Qu.: 43.00
##  Max.   :29.70   Max.   :47.3   Max.   :371.000   Max.   :117.00
##  NA's   :70      NA's   :62     NA's   :342       NA's   :586
##   WindSpeed9am   WindSpeed3pm    Humidity9am      Humidity3pm
##  Min.   : 0.0   Min.   : 0.00   Min.   :  5.00   Min.   :  1.00
##  1st Qu.: 7.0   1st Qu.:11.00   1st Qu.: 54.00   1st Qu.: 30.00
##  Median :13.0   Median :15.00   Median : 67.00   Median : 45.00
##  Mean   :12.8   Mean   :16.22   Mean   : 67.15   Mean   : 46.35
##  3rd Qu.:19.0   3rd Qu.:20.00   3rd Qu.: 82.00   3rd Qu.: 61.00
##  Max.   :56.0   Max.   :61.00   Max.   :100.00   Max.   :100.00
##  NA's   :412    NA's   :407     NA's   :211      NA's   :215
##   Pressure9am      Pressure3pm        Cloud9am        Cloud3pm
##  Min.   : 989.8   Min.   : 982.9   Min.   :0.000   Min.   :0.000
##  1st Qu.:1013.7   1st Qu.:1011.0   1st Qu.:1.000   1st Qu.:1.000
##  Median :1018.2   Median :1015.5   Median :4.000   Median :5.000
##  Mean   :1018.2   Mean   :1015.5   Mean   :4.042   Mean   :4.301
##  3rd Qu.:1022.7   3rd Qu.:1019.9   3rd Qu.:7.000   3rd Qu.:7.000
##  Max.   :1039.9   Max.   :1036.8   Max.   :8.000   Max.   :8.000
##  NA's   :514      NA's   :520      NA's   :6435    NA's   :6067
##     Temp9am         Temp3pm      RainTomorrow
##  Min.   : 0.30   Min.   : 6.40   No  :11815
##  1st Qu.:13.20   1st Qu.:18.20   Yes : 2843
##  Median :18.00   Median :22.80   NA's:  342
##  Mean   :17.62   Mean   :23.29
##  3rd Qu.:22.10   3rd Qu.:27.90
##  Max.   :37.70   Max.   :46.70
##  NA's   :73      NA's   :73
```

Exploring Missing values

```
sum(is.na(df))
```

```
## [1] 16329
```

Removing the row with target value NA

```
df <- subset(df,RainTomorrow  != "NA")
```

Dimension after removing rows with NA as Rain Tomorrow

```
dim(df)
```

```
## [1] 14658    15
```

```
str(df)
```

```
## 'data.frame':    14658 obs. of  15 variables:
##  $ MinTemp      : num  13.4 7.4 12.9 9.2 17.5 14.6 14.3 7.7 9.7 13.1 ...
##  $ MaxTemp      : num  22.9 25.1 25.7 28 32.3 29.7 25 26.7 31.9 30.1 ...
##  $ Rainfall     : num  0.6 0 0 0 1 0.2 0 0 0 1.4 ...
##  $ WindGustSpeed: int  44 44 46 24 41 56 50 35 80 28 ...
##  $ WindSpeed9am : int  20 4 19 11 7 19 20 6 7 15 ...
##  $ WindSpeed3pm : int  24 22 26 9 20 24 24 17 28 11 ...
##  $ Humidity9am  : int  71 44 38 45 82 55 49 48 42 58 ...
##  $ Humidity3pm  : int  22 25 30 16 33 23 19 19 9 27 ...
##  $ Pressure9am  : num  1008 1011 1008 1018 1011 ...
##  $ Pressure3pm  : num  1007 1008 1009 1013 1006 ...
##  $ Cloud9am     : int  8 NA NA NA 7 NA 1 NA NA NA ...
##  $ Cloud3pm     : int  NA NA 2 NA 8 NA NA NA NA NA ...
##  $ Temp9am      : num  16.9 17.2 21 18.1 17.8 20.6 18.1 16.3 18.3 20.1 ...
##  $ Temp3pm      : num  21.8 24.3 23.2 26.5 29.7 28.9 24.6 25.5 30.2 28.2 ...
##  $ RainTomorrow : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 2 1 ...
```

Replacing NA's with mean of a column

```
#install.packages('tidyr')
for(i in 1:ncol(df)){
  df[is.na(df[,i]), i] <- mean(df[,i], na.rm = TRUE)
}
```

```
## Warning in mean.default(df[, i], na.rm = TRUE): argument is not numeric or
## logical: returning NA
```

Summary after replacing NA's with mean

```
summary(df)
```

```
##     MinTemp          MaxTemp          Rainfall       WindGustSpeed
##  Min.   :-3.30   Min.   : 6.80   Min.   :  0.000   Min.   :  7.00
##  1st Qu.: 7.10   1st Qu.:19.40   1st Qu.:  0.000   1st Qu.: 28.00
##  Median :12.40   Median :24.30   Median :  0.000   Median : 35.00
##  Mean   :12.19   Mean   :24.65   Mean   :  2.395   Mean   : 36.37
##  3rd Qu.:17.30   3rd Qu.:29.40   3rd Qu.:  0.400   3rd Qu.: 43.00
##  Max.   :29.70   Max.   :47.30   Max.   :371.000   Max.   :117.00
##   WindSpeed9am    WindSpeed3pm    Humidity9am      Humidity3pm
##  Min.   : 0.00   Min.   : 0.00   Min.   :  5.00   Min.   :  1.00
##  1st Qu.: 7.00   1st Qu.:11.00   1st Qu.: 54.00   1st Qu.: 30.00
##  Median :12.74   Median :15.00   Median : 67.13   Median : 45.00
##  Mean   :12.74   Mean   :16.24   Mean   : 67.13   Mean   : 46.35
##  3rd Qu.:19.00   3rd Qu.:20.00   3rd Qu.: 81.00   3rd Qu.: 61.00
##  Max.   :56.00   Max.   :61.00   Max.   :100.00   Max.   :100.00
##   Pressure9am      Pressure3pm        Cloud9am        Cloud3pm
##  Min.   : 989.8   Min.   : 982.9   Min.   :0.000   Min.   :0.000
```

```
##  1st Qu.:1013.9   1st Qu.:1011.1   1st Qu.:3.000   1st Qu.:3.000
##  Median :1018.2   Median :1015.5   Median :4.032   Median :4.292
##  Mean   :1018.2   Mean   :1015.5   Mean   :4.032   Mean   :4.292
##  3rd Qu.:1022.6   3rd Qu.:1019.7   3rd Qu.:5.000   3rd Qu.:6.000
##  Max.   :1039.9   Max.   :1036.8   Max.   :8.000   Max.   :8.000
##     Temp9am         Temp3pm      RainTomorrow
##  Min.   : 0.30   Min.   : 6.40   No :11815
##  1st Qu.:13.10   1st Qu.:18.20   Yes: 2843
##  Median :17.90   Median :22.80
##  Mean   :17.58   Mean   :23.24
##  3rd Qu.:22.10   3rd Qu.:27.80
##  Max.   :37.70   Max.   :46.70
```

Data Visualization

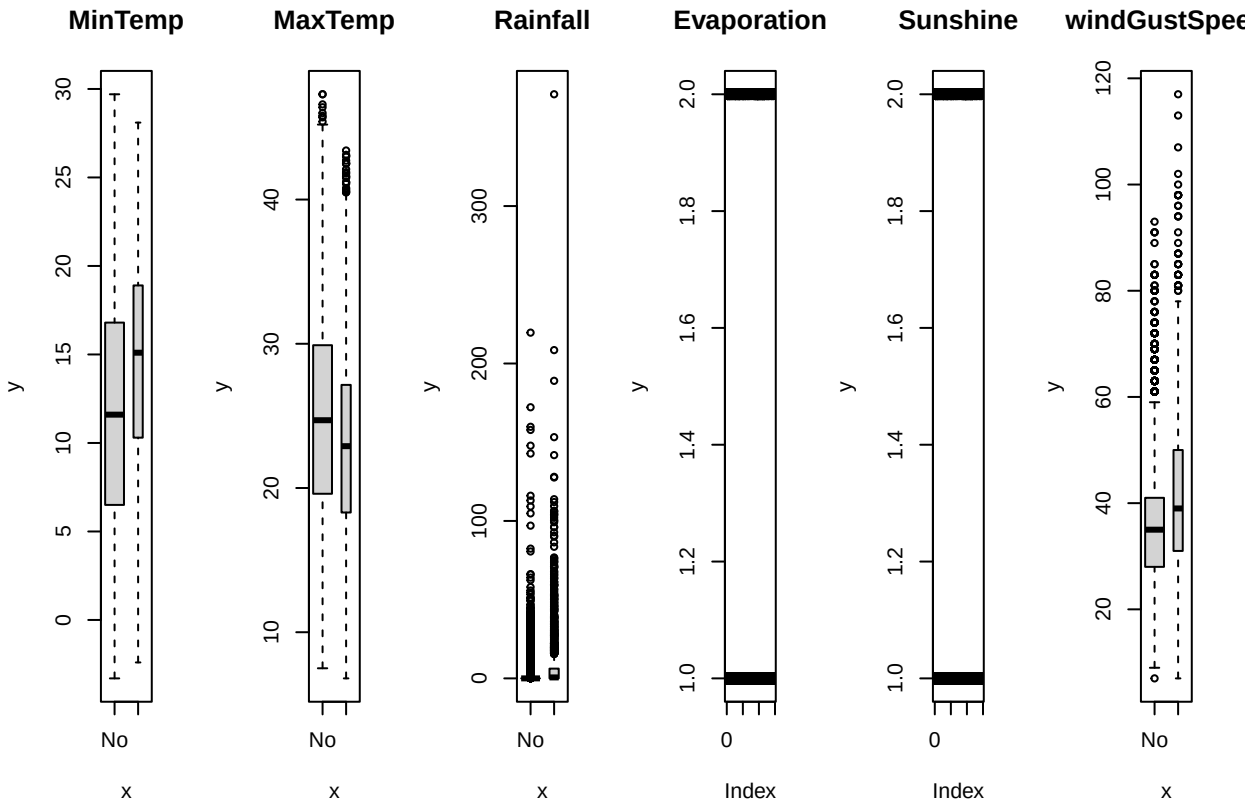
```
par(mfrow=c(1,6))
plot(df$RainTomorrow, df$MinTemp, data=df, main="MinTemp",
varwidth=TRUE)
plot(df$RainTomorrow, df$MaxTemp, data=df, main="MaxTemp", varwidth=TRUE)
plot(df$RainTomorrow, df$Rainfall, data=df, main="Rainfall", varwidth=TRUE)
plot(df$RainTomorrow, df$Evaporation, data=df, main="Evaporation", varwidth=TRUE)
```

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

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

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

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

## Warning in axis(side = side, at = at, labels = labels, ...): "data" is not a
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "varwidth" is not a
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "data" is not a
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "varwidth" is not a
## graphical parameter

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

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

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

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

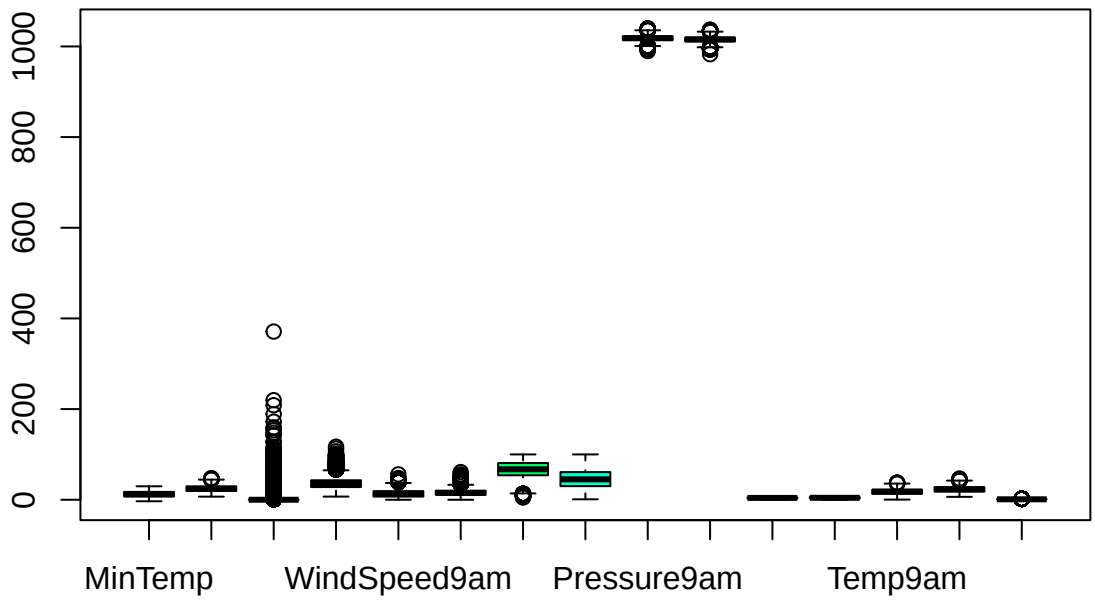
```
plot(df$RainTomorrow, df$Sunshine, data=df, main="Sunshine", varwidth=TRUE)
```

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

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

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

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

## Warning in axis(side = side, at = at, labels = labels, ...): "data" is not a
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "varwidth" is not a
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "data" is not a
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "varwidth" is not a
## graphical parameter

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

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

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

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

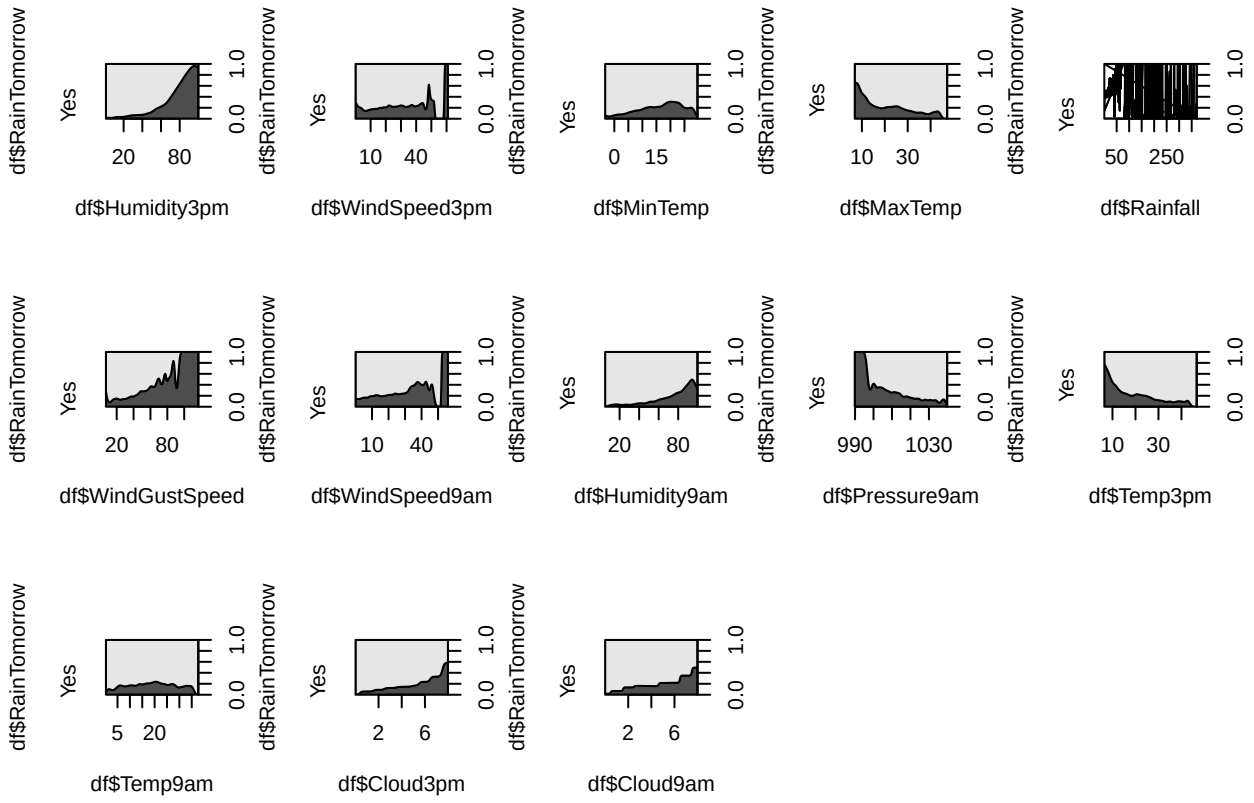
```
plot(df$RainTomorrow, df$WindGustSpeed, data=df, main="windGustSpeed",
varwidth=TRUE)
```




```
boxplot(df, col = rainbow(ncol(df)))
```




```
par(mfrow=c(3,5))
cdplot(df$RainTomorrow~df$Humidity3pm)
cdplot(df$RainTomorrow~df$WindSpeed3pm)
cdplot(df$RainTomorrow~df$MinTemp)
cdplot(df$RainTomorrow~df$MaxTemp)
cdplot(df$RainTomorrow~df$Rainfall)
cdplot(df$RainTomorrow~df$WindGustSpeed)
cdplot(df$RainTomorrow~df$WindSpeed9am)
cdplot(df$RainTomorrow~df$Humidity9am)
cdplot(df$RainTomorrow~df$Pressure9am)
cdplot(df$RainTomorrow~df$Temp3pm)
cdplot(df$RainTomorrow~df$Temp9am)
cdplot(df$RainTomorrow~df$Cloud3pm)
cdplot(df$RainTomorrow~df$Cloud9am)
```



##Spliting into train and test set

```
set.seed(1234)
i <- sample(1:nrow(df), 0.80*nrow(df), replace=FALSE)
train <-df[i,]
test <- df[-i,]
```

```
library(e1071)
trainForSVM <- train
trainForSVM$RainTomorrow <- NULL
head(trainForSVM)
```

```
##      MinTemp MaxTemp Rainfall WindGustSpeed WindSpeed9am WindSpeed3pm
## 7568     5.6    18.2      0.0            24            9            7
## 8132    18.0    30.0      0.4            48           31           28
## 7276     3.7    16.6      0.0            26            7           15
## 8202    22.0    31.5      0.4            37           17            6
## 7383     4.7    23.6      0.0            26           20            7
## 9328     7.6    22.6      0.0            56           13           26
##      Humidity9am Humidity3pm Pressure9am Pressure3pm Cloud9am Cloud3pm Temp9am
## 7568          79          42      1028.4      1025.9        7        3    11.0
## 8132          41          13      1013.7      1010.6        1        1    21.7
## 7276          60          35      1025.3      1021.7        1        3    10.3
## 8202          65          40      1013.4      1011.3        8        8    23.6
## 7383          30          12      1021.0      1016.2        1        0    14.2
## 9328          21          11      1010.1      1006.0        1        0    18.0
```

```
##      Temp3pm
## 7568    17.5
## 8132    28.7
## 7276    15.8
## 8202    31.0
## 7383    22.6
## 9328    21.9
```

Model Building and Prediction on Test set

```
trainForSVMLabels <- train$RainTomorrow
testForSVM <- test
testForSVM$RainTomorrow <- NULL
testLabelForSVM <- test$RainTomorrow
svm1 <- svm(train$RainTomorrow~., data = train, kernel = "linear", cost = 50, scale = TRUE)
summary(svm1)
```

```
##
## Call:
## svm(formula = train$RainTomorrow ~ ., data = train, kernel = "linear",
##     cost = 50, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  50
##
## Number of Support Vectors:  3664
##
##  ( 1834 1830 )
##
##
## Number of Classes:  2
##
## Levels:
##  No Yes
```

```
pred <- predict(svm1,newdata=test)
table(pred,test$RainTomorrow)
```

```
##
## pred    No  Yes
##   No  2261  298
##   Yes   73  300
```

Confusion Matrix and Statistics of Linear SVM

```
library(caret)
```

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

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

```
confusionMatrix(as.factor(pred),as.factor(test$RainTomorrow))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  2261  298
##        Yes   73  300
##
##                Accuracy : 0.8735
##                  95% CI : (0.8609, 0.8853)
##     No Information Rate : 0.796
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5469
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9687
##             Specificity : 0.5017
##          Pos Pred Value : 0.8835
##          Neg Pred Value : 0.8043
##              Prevalence : 0.7960
##          Detection Rate : 0.7711
##    Detection Prevalence : 0.8728
##       Balanced Accuracy : 0.7352
##
##        'Positive' Class : No
##
```

Tuning the value for C

```
set.seed(1234)
i <- sample(1:nrow(train), 0.20*nrow(train), replace=FALSE)

validTune <- train[i,]
train <- train[-1,]
validTuneLabels <- validTune$RainTomorrow
```

Getting best C

```
par(mfrow=c(1,3))
tune.out <- tune(svm,RainTomorrow~., data=validTune, kernel="linear",
ranges=list(cost=c(0.001, 0.01, 0.1, 1, 5, 10, 100)))
summary(tune.out)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##     1
##
## - best performance: 0.1304837
##
## - Detailed performance results:
##    cost     error dispersion
## 1 1e-03 0.1859284 0.02149188
## 2 1e-02 0.1347590 0.02046146
## 3 1e-01 0.1309093 0.02491476
## 4 1e+00 0.1304837 0.02733512
## 5 5e+00 0.1309093 0.02717170
## 6 1e+01 0.1309093 0.02717170
## 7 1e+02 0.1309093 0.02717170
```

Choosing best model based upon the C value

```
best_model1 <- tune.out$best.model
summary(best_model1)
```

```
##
## Call:
## best.tune(method = svm, train.x = RainTomorrow ~ ., data = validTune,
##     ranges = list(cost = c(0.001, 0.01, 0.1, 1, 5, 10, 100)), kernel = "linear")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  1
##
## Number of Support Vectors:  739
##
##  ( 372 367 )
##
##
## Number of Classes:  2
##
## Levels:
##  No Yes
```

Model Building and predicting with Best model

```
pred <- predict(best_model1,newdata=test)
```

Confusion Matrix of best C

```
library(caret)
confusionMatrix(as.factor(pred),as.factor(test$RainTomorrow))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  2276  315
##        Yes   58  283
##
##                Accuracy : 0.8728
##                  95% CI : (0.8602, 0.8846)
##     No Information Rate : 0.796
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5337
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9751
##             Specificity : 0.4732
##          Pos Pred Value : 0.8784
##          Neg Pred Value : 0.8299
##              Prevalence : 0.7960
##          Detection Rate : 0.7763
##    Detection Prevalence : 0.8837
##       Balanced Accuracy : 0.7242
##
##        'Positive' Class : No
##
```

Model Building and prediction of SVM polynomial with cost =1 and gamma =1

```
svm2 <- svm(train$RainTomorrow~., data = train, kernel = "polynomial", cost = 1,gamma = 1, scale = TRUE)
summary(svm2)
```

```
##
## Call:
## svm(formula = train$RainTomorrow ~ ., data = train, kernel = "polynomial",
##     cost = 1, gamma = 1, scale = TRUE)
##
##
```

```
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  polynomial
##        cost:  1
##      degree:  3
##      coef.0:  0
##
## Number of Support Vectors:  3164
##
##  ( 1707 1457 )
##
##
## Number of Classes:  2
##
## Levels:
##  No Yes
```

```
pred <- predict(svm2,newdata=test)
table(pred,test$RainTomorrow)
```

```
##
## pred    No  Yes
##   No  2256  327
##   Yes   78  271
```

Confusion Matrix of SVM kernal = Polynomial

```
library(caret)
confusionMatrix(as.factor(pred),as.factor(test$RainTomorrow))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  2256  327
##        Yes   78  271
##
##                Accuracy : 0.8619
##                  95% CI : (0.8488, 0.8742)
##     No Information Rate : 0.796
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4967
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9666
##             Specificity : 0.4532
##          Pos Pred Value : 0.8734
##          Neg Pred Value : 0.7765
##              Prevalence : 0.7960
```

```
##          Detection Rate : 0.7694
##    Detection Prevalence : 0.8810
##       Balanced Accuracy : 0.7099
##
##        'Positive' Class : No
##
```

Tuning svm polynomial

```
set.seed(1234)
tune.out <- tune(svm,RainTomorrow~.,data = validTune[1:200,], kernel="polynomial",scale=TRUE,
ranges=list(cost=c(0.001, 0.01, 0.1, 1, 5, 10),gamma=c(0.5,1,2,3)))
summary(tune.out)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##  0.01   0.5
##
## - best performance: 0.145
##
## - Detailed performance results:
##     cost gamma error dispersion
## 1  1e-03   0.5 0.165 0.08181958
## 2  1e-02   0.5 0.145 0.07245688
## 3  1e-01   0.5 0.200 0.08819171
## 4  1e+00   0.5 0.270 0.08881942
## 5  5e+00   0.5 0.270 0.08881942
## 6  1e+01   0.5 0.270 0.08881942
## 7  1e-03   1.0 0.145 0.07245688
## 8  1e-02   1.0 0.190 0.09067647
## 9  1e-01   1.0 0.260 0.08432740
## 10 1e+00   1.0 0.270 0.08881942
## 11 5e+00   1.0 0.270 0.08881942
## 12 1e+01   1.0 0.270 0.08881942
## 13 1e-03   2.0 0.180 0.08232726
## 14 1e-02   2.0 0.245 0.10124228
## 15 1e-01   2.0 0.270 0.08881942
## 16 1e+00   2.0 0.270 0.08881942
## 17 5e+00   2.0 0.270 0.08881942
## 18 1e+01   2.0 0.270 0.08881942
## 19 1e-03   3.0 0.220 0.09775252
## 20 1e-02   3.0 0.270 0.08881942
## 21 1e-01   3.0 0.270 0.08881942
## 22 1e+00   3.0 0.270 0.08881942
## 23 5e+00   3.0 0.270 0.08881942
## 24 1e+01   3.0 0.270 0.08881942
```

Getting Best model

```
best_model2 <- tune.out$best.model
summary(best_model2)
```

```
##
## Call:
## best.tune(method = svm, train.x = RainTomorrow ~ ., data = validTune[1:200,
##     ], ranges = list(cost = c(0.001, 0.01, 0.1, 1, 5, 10), gamma = c(0.5,
##     1, 2, 3)), kernel = "polynomial", scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  polynomial
##        cost:  0.01
##      degree:  3
##      coef.0:  0
##
## Number of Support Vectors:  88
##
##  ( 55 33 )
##
##
## Number of Classes:  2
##
## Levels:
##  No Yes
```

Prediction for best value of C and gamma

```
pred <- predict(best_model2,newdata=test)
table(pred,test$RainTomorrow)
```

```
##
## pred    No  Yes
##   No  2247  379
##   Yes   87  219
```

Confusion matrix and Statistics of best model svm(Kenrnel = polynomial)

```
library(caret)
confusionMatrix(as.factor(pred),as.factor(test$RainTomorrow))
```

```
## Confusion Matrix and Statistics
##
##           Reference
```

```
## Prediction   No  Yes
##        No  2247  379
##        Yes   87  219
##
##                Accuracy : 0.8411
##                  95% CI : (0.8273, 0.8541)
##     No Information Rate : 0.796
##     P-Value [Acc > NIR] : 2.711e-10
##
##                   Kappa : 0.4019
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9627
##             Specificity : 0.3662
##          Pos Pred Value : 0.8557
##          Neg Pred Value : 0.7157
##              Prevalence : 0.7960
##          Detection Rate : 0.7664
##    Detection Prevalence : 0.8956
##       Balanced Accuracy : 0.6645
##
##        'Positive' Class : No
##
```

Model Building and prediction for SVM Kernel = radial

```
svm3 <- svm(train$RainTomorrow~., data = train, kernel = "radial",scale= TRUE, cost = 1,gamma = 1)
summary(svm3)
```

```
##
## Call:
## svm(formula = train$RainTomorrow ~ ., data = train, kernel = "radial",
##     cost = 1, gamma = 1, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  8831
##
##  ( 6615 2216 )
##
##
## Number of Classes:  2
##
## Levels:
##  No Yes
```

```
pred <- predict(svm3,newdata=test)
table(pred,test$RainTomorrow)
```

```
##
## pred    No  Yes
##   No  2303  473
##   Yes   31  125
```

Confusion Matrix and Statistics of SVM kernal = radial and C = 1 and Gamma = 1

```
library(caret)
confusionMatrix(as.factor(pred),as.factor(test$RainTomorrow))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  2303  473
##        Yes   31  125
##
##                Accuracy : 0.8281
##                  95% CI : (0.814, 0.8416)
##     No Information Rate : 0.796
##     P-Value [Acc > NIR] : 6.196e-06
##
##                   Kappa : 0.27
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9867
##             Specificity : 0.2090
##          Pos Pred Value : 0.8296
##          Neg Pred Value : 0.8013
##              Prevalence : 0.7960
##          Detection Rate : 0.7855
##    Detection Prevalence : 0.9468
##       Balanced Accuracy : 0.5979
##
##        'Positive' Class : No
##
```

Tuning SVM with Kernel = radial

```
set.seed(1234)
tune.out <- tune(svm,RainTomorrow~., data=validTune,scale= TRUE, kernel="radial",
ranges=list(cost=c(0.001, 0.01, 0.1, 1, 5, 10, 100),gamma=c(0.5,1,2,3,4)))
summary(tune.out)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##     1   0.5
##
## - best performance: 0.1578069
##
## - Detailed performance results:
##     cost gamma     error dispersion
## 1  1e-03   0.5 0.1850955 0.02595582
## 2  1e-02   0.5 0.1850955 0.02595582
## 3  1e-01   0.5 0.1850955 0.02595582
## 4  1e+00   0.5 0.1578069 0.03098865
## 5  5e+00   0.5 0.1671777 0.03234231
## 6  1e+01   0.5 0.1684561 0.03395415
## 7  1e+02   0.5 0.1701582 0.03405861
## 8  1e-03   1.0 0.1850955 0.02595582
## 9  1e-02   1.0 0.1850955 0.02595582
## 10 1e-01   1.0 0.1850955 0.02595582
## 11 1e+00   1.0 0.1812548 0.02924004
## 12 5e+00   1.0 0.1850937 0.03345660
## 13 1e+01   1.0 0.1846663 0.03251925
## 14 1e+02   1.0 0.1846663 0.03251925
## 15 1e-03   2.0 0.1850955 0.02595582
## 16 1e-02   2.0 0.1850955 0.02595582
## 17 1e-01   2.0 0.1850955 0.02595582
## 18 1e+00   2.0 0.1850955 0.02595582
## 19 5e+00   2.0 0.1859502 0.02620420
## 20 1e+01   2.0 0.1859502 0.02620420
## 21 1e+02   2.0 0.1859502 0.02620420
## 22 1e-03   3.0 0.1850955 0.02595582
## 23 1e-02   3.0 0.1850955 0.02595582
## 24 1e-01   3.0 0.1850955 0.02595582
## 25 1e+00   3.0 0.1850955 0.02595582
## 26 5e+00   3.0 0.1859484 0.02687757
## 27 1e+01   3.0 0.1859484 0.02687757
## 28 1e+02   3.0 0.1859484 0.02687757
## 29 1e-03   4.0 0.1850955 0.02595582
## 30 1e-02   4.0 0.1850955 0.02595582
## 31 1e-01   4.0 0.1850955 0.02595582
## 32 1e+00   4.0 0.1850955 0.02595582
## 33 5e+00   4.0 0.1859484 0.02687757
## 34 1e+01   4.0 0.1859484 0.02687757
## 35 1e+02   4.0 0.1859484 0.02687757
```

```
best_model3 <- tune.out$best.model
summary(best_model3)
```

```
##
## Call:
```

```
## best.tune(method = svm, train.x = RainTomorrow ~ ., data = validTune,
##     ranges = list(cost = c(0.001, 0.01, 0.1, 1, 5, 10, 100), gamma = c(0.5,
##         1, 2, 3, 4)), scale = TRUE, kernel = "radial")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  1618
##
##  ( 1194 424 )
##
##
## Number of Classes:  2
##
## Levels:
##  No Yes
```

Model Building and Prediction Best value of C and gamma for SVM with kernel = "radial"

```
pred <- predict(best_model3,newdata=test)
table(pred,test$RainTomorrow)
```

```
##
## pred    No  Yes
##   No  2292  453
##   Yes   42  145
```

Confusion Matrix and Statistics of Best model of SVM whose kernel = radial

```
library(caret)
confusionMatrix(as.factor(pred),as.factor(test$RainTomorrow))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  2292  453
##        Yes   42  145
##
##                Accuracy : 0.8312
##                  95% CI : (0.8171, 0.8446)
##     No Information Rate : 0.796
##     P-Value [Acc > NIR] : 7.852e-07
##
```

```
##                   Kappa : 0.3016
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9820
##             Specificity : 0.2425
##          Pos Pred Value : 0.8350
##          Neg Pred Value : 0.7754
##              Prevalence : 0.7960
##          Detection Rate : 0.7817
##    Detection Prevalence : 0.9362
##       Balanced Accuracy : 0.6122
##
##        'Positive' Class : No
##
```

Result Analysis:

This notebook has the experiment done on the weather dataset from Kaggle. The main point of this notebook is to predict whether it will rain or not tomorrow. I have used SVM classification with different kernel types along with various parameters and hyperparameters. Also, tuning of hyperperimenter is done to get the best model and the values of those hyperparameters which gives the model.

SVM with kernel linear, cost = 50 and scale = True

I was able to get accuracy of about 87 percent.The model had 3664 support vector classfier before tuning but after doing crossvalidation and using the best model, it was reduced to 739.

SVM with kernel polynomial cost =1 , gamma = 1 and Scale = TRUE

Without tunning I was able to get the accuracy of about 86 percent.Before tuning the number of support vectors was 3164 and later after tuning it was changed to 88.

SVM with kernel radial cost = 1, gamma = 1, scale = True

Without tuning the accuracy was 82 percent. Before tuning the number of Support vector was 8831 and later changed to 1618 after tuning.

It is clear from the above statistics that the accuracy was higher when the kernel was linear which means that the distance from both classes to the hyperplane was maximum in linear kernel. The run time for linear was less than for polynomial. It was taking a lot of time so I have used less data for tuning. This might be because it have to transform data to other planes. So, Linear kernel outperform for my dataset.