Regression SVM

Jonathan Ho

October 20, 2022

Load packages and data

```
library(e1071)
library(MASS)
df <- read.csv("weatherHistory.csv", header=TRUE)
```

Divide into train, test, and validate

Load in only 10,000 randoms rows of data due to long loading times of SVM kernels.

```
set.seed(420)
spec <- c(train = 0.6, test = 0.2, validate = 0.2)
i <- sample(cut(1:nrow(df), nrow(df)*cumsum(c(0,spec)), labels=names(spec)))
train <- df[i=="train",]
test <- df[i=="test",]
vali <- df[i=="validate",]
```

Data exploration

View all columns within the dataset.

```
str(train)
```

```
## 'data.frame':    57871 obs. of  12 variables:
##  $ Formatted.Date          : chr  "2006-04-01 01:00:00.000 +0200" "2006-04-01 02:00:00.000 +0
200" "2006-04-01 03:00:00.000 +0200" "2006-04-01 05:00:00.000 +0200" ...
##  $ Summary                 : chr  "Partly Cloudy" "Mostly Cloudy" "Partly Cloudy" "Partly Clo
udy" ...
##  $ Precip.Type             : chr  "rain" "rain" "rain" "rain" ...
##  $ Temperature..C.         : num  9.36 9.38 8.29 9.22 7.73 ...
##  $ Apparent.Temperature..C.: num  7.23 9.38 5.94 7.11 5.52 ...
##  $ Humidity                : num  0.86 0.89 0.83 0.85 0.95 0.89 0.54 0.69 0.77 0.66 ...
##  $ Wind.Speed..km.h.       : num  14.26 3.93 14.1 13.96 12.36 ...
##  $ Wind.Bearing..degrees.  : num  259 204 269 258 259 260 316 163 152 149 ...
##  $ Visibility..km.         : num  15.83 14.96 15.83 14.96 9.98 ...
##  $ Loud.Cover              : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Pressure..millibars.    : num  1016 1016 1016 1017 1017 ...
##  $ Daily.Summary           : chr  "Partly cloudy throughout the day." "Partly cloudy througho
ut the day." "Partly cloudy throughout the day." "Partly cloudy throughout the day." ...
```

Check for NAs.

```
sapply(df, function(y) sum(is.na(y)))
```

```
##           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
```

Display the number of rows and columns in the dataset.

```
dim(df)
```

```
## [1] 96453    12
```

Summary of each column.

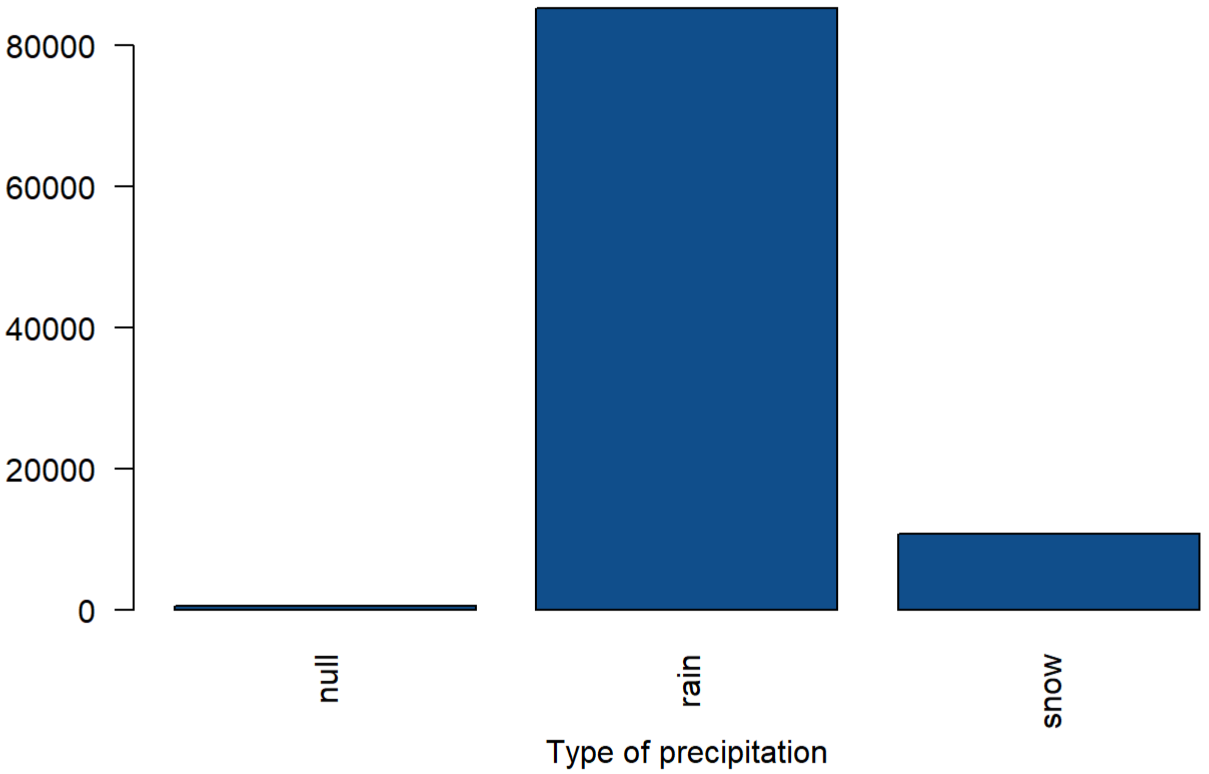
```
summary(df)
```

```
##  Formatted.Date       Summary          Precip.Type        Temperature..C.
##  Length:96453       Length:96453       Length:96453       Min.   :-21.822
##  Class :character   Class :character   Class :character   1st Qu.:  4.689
##  Mode  :character   Mode  :character   Mode  :character   Median : 12.000
##                                                           Mean   : 11.933
##                                                           3rd Qu.: 18.839
##                                                           Max.   : 39.906
##  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     
##  Length:96453      
##  Class :character  
##  Mode  :character  
##                    
##                    
## 
```

Data visualization

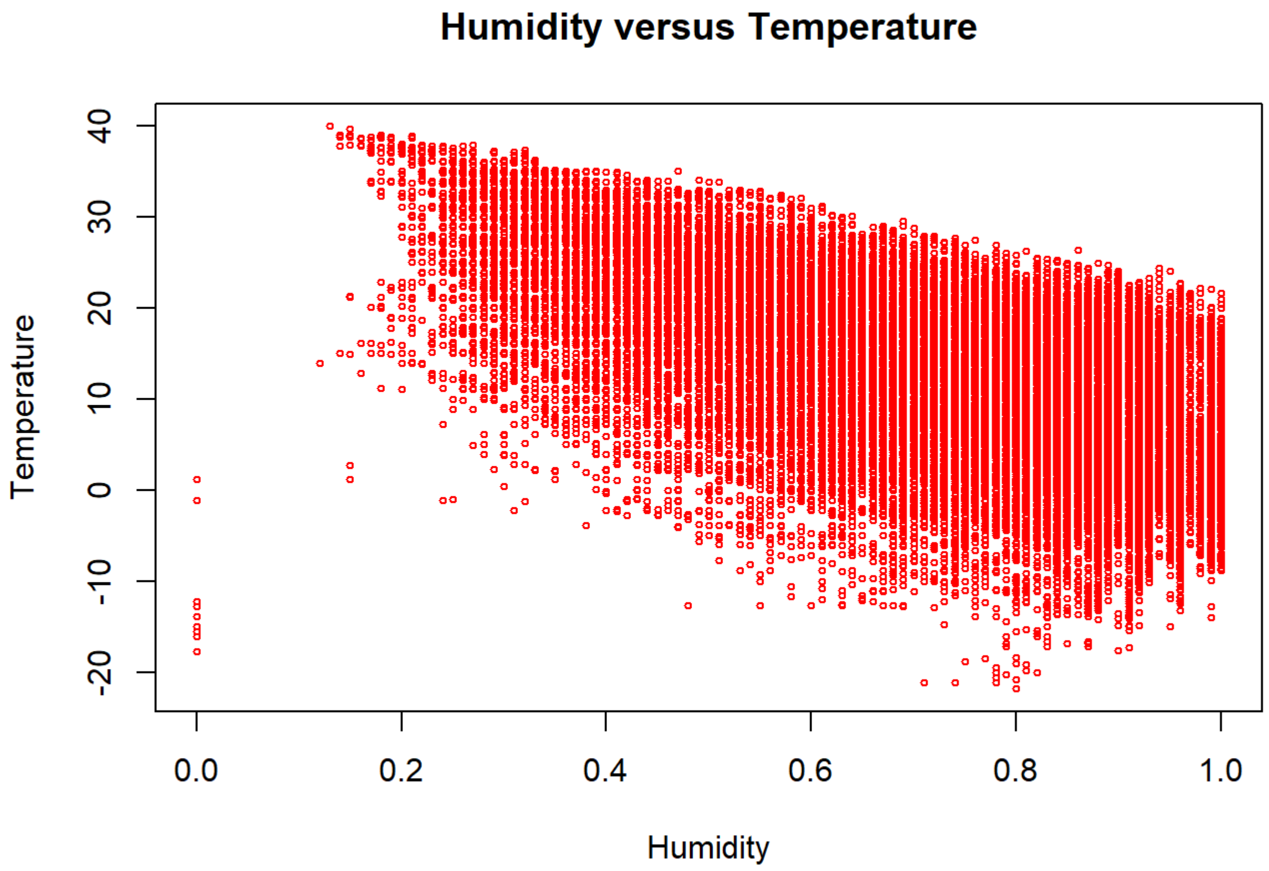
Bar plot of the type of precipitation.

```
counts <- table(df$Precip.Type)
barplot(counts, xlab="Type of precipitation", ylab="", col="dodgerblue4", las=2) # las=2 display
s all the Platforms
```



Scatter plot of Humidity versus Temperature.

```
plot(df$Humidity, df$Temperature..C., pch=1, col="red", cex=0.5,
     main="Humidity versus Temperature", xlab="Humidity", ylab="Temperature")
```

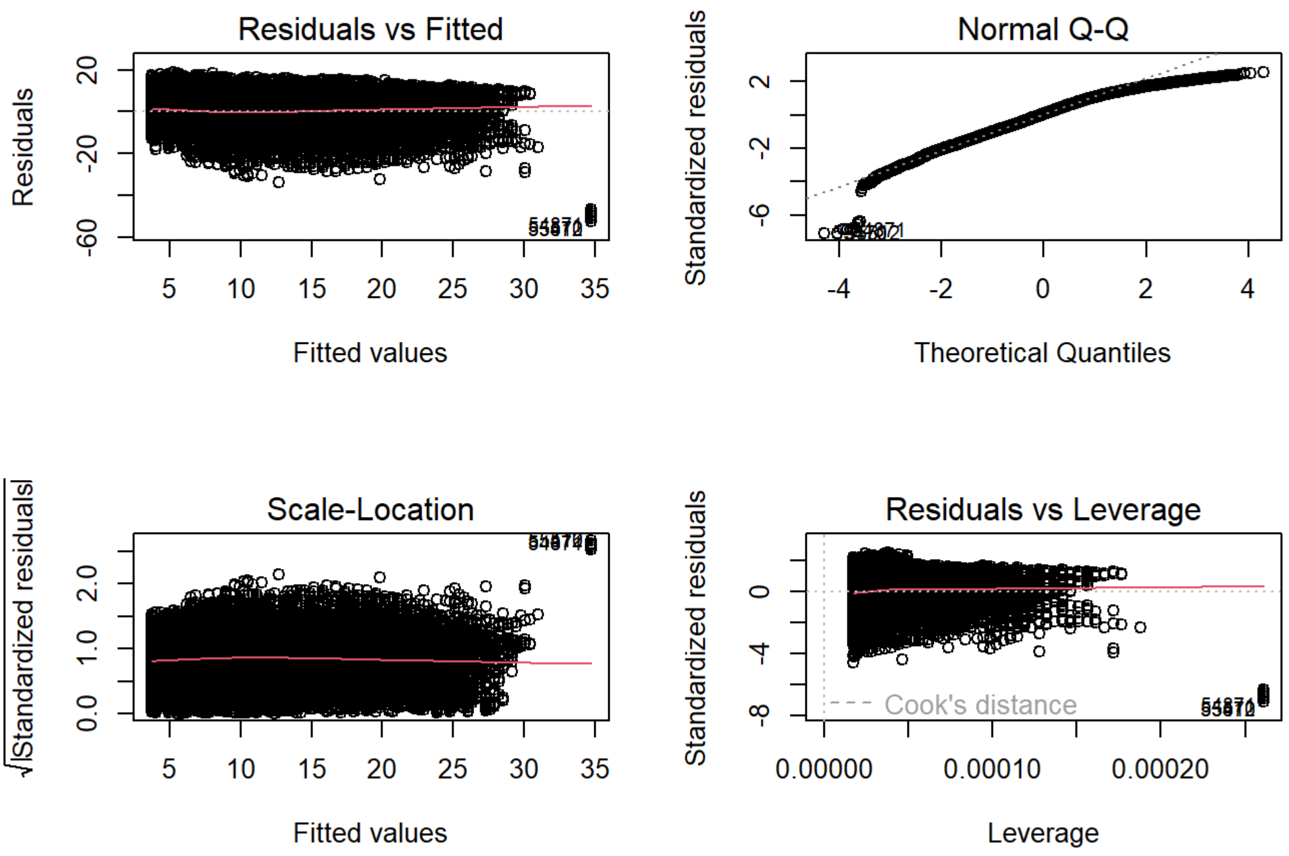


Linear regression

```
lm1 <- lm(Temperature..C.~Humidity, data=train)
summary(lm1)
```

```
##
## Call:
## lm(formula = Temperature..C. ~ Humidity, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -52.522  -5.084   0.354   5.736  18.805
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  34.7446     0.1196   290.5   <2e-16 ***
## Humidity    -31.0580     0.1574  -197.3   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.404 on 57869 degrees of freedom
## Multiple R-squared:  0.4022, Adjusted R-squared:  0.4022
## F-statistic: 3.893e+04 on 1 and 57869 DF,  p-value: < 2.2e-16
```

```
par(mfrow=c(2,2))
plot(lm1)
```



Making prediction (summary at end)

```
pred <- predict(lm1, newdata=test)
cor_lm1 <- cor(pred, test$Temperature..C.)
mse_lm1 <- mean((pred-test$Temperature..C.)^2)
```

Linear Kernel

Binary classification of Temperature and Humidity. Tuning is done to try and get the best cost. Gamma is not done since it is for non-linear kernels. A prediction is also done on the best linear svm.

```
tune_lsvm <- tune(svm, Temperature..C.~Humidity, data=vali, kernel="linear", range=list(cost=c(
0.001, 0.01, 0.1, 1)))
summary(tune_lsvm)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##   cost
##  0.001
##
## - best performance: 53.88654
##
## - Detailed performance results:
##    cost    error dispersion
## 1 0.001 53.88654   1.553197
## 2 0.010 54.16318   1.646645
## 3 0.100 54.21211   1.659913
## 4 1.000 54.21707   1.661722
```

```
pred <- predict(tune_lsvm$best.model, newdata=test)
cor_lm2 <- cor(pred, test$Temperature..C.)
mse_lm2 <- mean((pred - test$Temperature..C.)^2)
```

Polynomial Kernel

Using a Polynomial Kernel and making a prediction.

```
svm2 <- svm(Temperature..C.~Humidity, data=train, kernel="polynomial", cost=1, scale=TRUE)
summary(svm2)
```

```
##
## Call:
## svm(formula = Temperature..C. ~ Humidity, data = train, kernel = "polynomial",
##     cost = 1, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  polynomial
##        cost:  1
##      degree:  3
##       gamma:  1
##      coef.0:  0
##     epsilon:  0.1
##
##
## Number of Support Vectors:  53227
```

```
pred <- predict(svm2, newdata=test)
cor_lm3 <- cor(pred, test$Temperature..C.)
mse_lm3 <- mean((pred - test$Temperature..C.)^2)
```

Radial Kernel

Tuning hyperparameters with different costs and gamma to find the best cost and gamma.

```
set.seed(420)
tune.out <- tune(svm, Temperature..C.~Humidity, data=vali, kernel="radial",
                 ranges=list(cost=c(0.1,1),
                             gamma=c(0.5,1)))
summary(tune.out)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##     1     1
##
## - best performance: 53.24874
##
## - Detailed performance results:
##   cost gamma    error dispersion
## 1  0.1   0.5 53.47366   1.483534
## 2  1.0   0.5 53.42845   1.518858
## 3  0.1   1.0 53.40792   1.513519
## 4  1.0   1.0 53.24874   1.556791
```

Using best cost and gamma to do a prediction.

```
svm4 <- svm(Temperature..C.~Humidity, data=train, kernel="radial", cost=0.1, gamma=0.5, scale=TR
UE)
summary(svm4)
```

```
##
## Call:
## svm(formula = Temperature..C. ~ Humidity, data = train, kernel = "radial",
##     cost = 0.1, gamma = 0.5, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  radial
##        cost:  0.1
##       gamma:  0.5
##     epsilon:  0.1
##
##
## Number of Support Vectors:  52497
```

```
pred <- predict(svm4, newdata=test)
cor_lm4 <- cor(pred, test$Temperature..C.)
mse_lm4 <- mean((pred - test$Temperature..C.)^2)
```

Summary of Results

```
cat("Linear Regression:\n")
```

```
## Linear Regression:
```

```
print(paste('cor: ', cor_lm1))
```

```
## [1] "cor:  0.620367853918574"
```

```
print(paste('mse: ', mse_lm1))
```

```
## [1] "mse:  55.6797485555949"
```

```
cat("\nLinear Kernel:\n")
```

```
##
## Linear Kernel:
```

```
print(paste('cor: ', cor_lm2))
```

```
## [1] "cor:  0.620367853918574"
```

```
print(paste('mse: ', mse_lm2))
```

```
## [1] "mse:  56.2101782838416"
```

```
cat("\nPolynomial Kernel:\n")
```

```
##
## Polynomial Kernel:
```

```
print(paste('cor: ', cor_lm3))
```

```
## [1] "cor:  0.488008588872778"
```

```
print(paste('mse: ', mse_lm3))
```

```
## [1] "mse:  70.7355906041254"
```

```
cat("\nRadial Kernel:\n")
```

```
##
## Radial Kernel:
```

```
print(paste('cor: ', cor_lm4))
```

```
## [1] "cor:  0.628400190703289"
```

```
print(paste('mse: ', mse_lm4))
```

```
## [1] "mse:  55.4472918468278"
```

Results Discussion

With the given metrics, it is seen that Radial Kernal gives the highest correlation. However, the lowest mse would be from Linear Regression. Unsurprisingly correlation for Linear Regression and Kernel are the same, most likely cause the data fits pretty linearly. Mse might be slightly higher on a Linear Kernel because it may have assumed a few data points were SVMs that it should not have. As for why Radial Kernal had technically the highest correlation, there were probably a few outliers that Linear Regression and Kernel took into account that Radial left out. Polynomial Kernal had the lowest correlation and mse most likely since it tried to fit a polynomial function to something inheritantly linear.