Darrell_Nelson_HW9

Darrell Nelson II

March 27, 2019

```
# Darrell Nelson II
# HW# 09

# Step 1: Load airquality dataset
data(airquality)
air <- airquality
# Clean the data
# Check out which column contains the most NAs
colSums(is.na(air))
```

```
##   Ozone Solar.R    Wind    Temp   Month     Day
##      37       7       0       0       0       0
```

```
air <- na.omit(air) # removes all rows with NAs present

# Step 2: Create train and test datasets
# Randomly select 2/3rds of data for training, the other third will be for testing
randIndex <- sample(1:dim(air)[1]) # Create list/vector variable-random index
summary(randIndex) # verify indicies in randIndex
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     1.0    28.5    56.0    56.0    83.5   111.0
```

```
length(randIndex)
```

```
## [1] 111
```

```
head(randIndex)
```

```
## [1]  67 103  75  53   4  95
```

```
cutPoint2_3 <- floor(2*dim(air)[1]/3) # create 2/3rd cutpoint; floor rounds number to the lower integer
traindata <- air[randIndex[1:cutPoint2_3],] # create training data set
testdata <- air[randIndex[(cutPoint2_3+1):dim(air)[1]],] # create test data set

# Step 3: Build a KSVM (Kernel Support Vector Machine) & visualize the results
# Create SVM (support vector machine)
library("kernlab")
svmOutput <- ksvm(Ozone~., data=traindata, kernel="rbfdot", kpar="automatic", C=5, cross=3, prob.model='
svmOutput
```

```
## Support Vector Machine object of class "ksvm"
##
## SV type: eps-svr  (regression)
```

```
##  parameter : epsilon = 0.1  cost C = 5
##
## Gaussian Radial Basis kernel function.
##  Hyperparameter : sigma =  0.266490405307048
##
## Number of Support Vectors : 60
##
## Objective Function Value : -52.4697
## Training error : 0.08703
## Cross validation error : 556.386
## Laplace distr. width : 42.80243
```

```
# Predict how accurate we are w/test data when C=5
svmPred <- predict(svmOutput, testdata,type="response")
compTable <- data.frame(testdata[ ,1], svmPred)
# table(compTable)

diff <- compTable[ ,1] - compTable[ ,2] # difference b/w actual & predicted
squarediff <- diff^2
averagediff <- mean(squarediff)
sqrtdiff <- sqrt(averagediff)
message("RMSE = ", sqrtdiff)
```

```
## RMSE = 16.7025284332314
```

```
# Plot the results using a scatter plot
error <- diff # actual minus predicted
testdata$Error <- error
# Created error level parameter to transform error values into factor levels
testdata$ErrorLevel <- cut(testdata$Error, seq(-100, 100, 20), right = FALSE, labels = c(1:10))
library("ggplot2")
```

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

## The following object is masked from 'package:kernlab':
##
##     alpha
```

```
g1 <- ggplot(testdata, aes(x=Temp, y=Wind)) + geom_point(aes(color=ErrorLevel, size=ErrorLevel)) + ggti

# Create, compute, and visualize svm and lm models
# install.packages("e1071")
library("e1071")
mode.2 <- svm(Ozone~., data=traindata, cost=5, cross=3, probability=TRUE)
```

```
## Warning in cret$cresults * scale.factor: Recycling array of length 1 in vector-array arithmetic is d
##   Use c() or as.vector() instead.
```

```r
# Remove extra columns from testdata
testdata <- testdata[ , -length(testdata)+1:-length(testdata)] # revert testdata back to original form
# Predict how accurate we are w/test data when cost=5
svmPred2 <- predict(mode.2, testdata,type="response")
svmPred2 <- data.frame(svmPred2)
compTable2 <- data.frame(testdata[1], svmPred2)
diff <- compTable2[ ,1] - compTable2[ ,2] # difference b/w actual & predicted
sqrtdiff2 <- sqrt(mean(diff^2))
message("RMSE = ", sqrtdiff2)
```

```
## RMSE = 16.6816213318266
```

```r
# Plot the results using a scatter plot
error <- diff # actual minus predicted
testdata$Error <- error
# Created error level parameter to transform error values into factor levels
testdata$ErrorLevel <- cut(testdata$Error, seq(-100, 100, 20), right = FALSE, labels = c(1:10))
library("ggplot2")
g2 <- ggplot(testdata, aes(x=Temp, y=Wind)) + geom_point(aes(color=ErrorLevel, size=ErrorLevel)) + ggti

# Use linear modeling (regression)
mode.3 <- lm(formula=Ozone~., data=traindata)
# Remove extra columns from testdata
testdata <- testdata[ , -length(testdata)+1:-length(testdata)] # revert testdata back to original form
# Predict how accurate we are w/test data when cost=5
svmPred3 <- predict(mode.3, testdata,type="response")
svmPred3 <- data.frame(svmPred3)
compTable3 <- data.frame(testdata[1], svmPred3)
diff <- compTable3[ ,1] - compTable3[ ,2] # difference b/w actual & predicted
sqrtdiff3 <- sqrt(mean(diff^2))
message("RMSE = ", sqrtdiff3)
```

```
## RMSE = 18.8240188350757
```

```r
modelnames <- c("KSVM", "SVM", "LM")
RMSE <- c(sqrtdiff, sqrtdiff2, sqrtdiff3)
data.frame(modelnames, RMSE)
```

```
##   modelnames     RMSE
## 1       KSVM 16.70253
## 2        SVM 16.68162
## 3         LM 18.82402
```

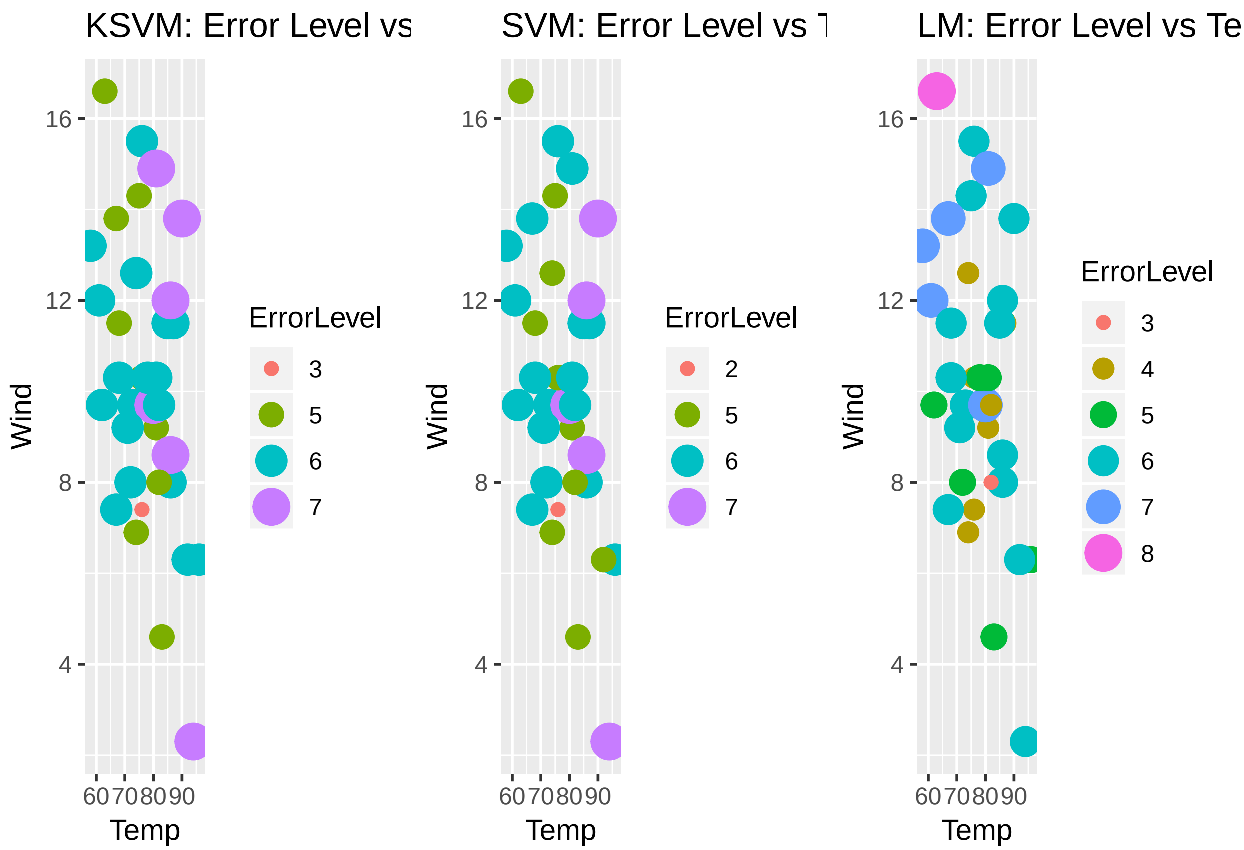
```r
# Plot the results using a scatter plot
error <- diff # actual minus predicted
testdata$Error <- error
# Created error level parameter to transform error values into factor levels
testdata$ErrorLevel <- cut(testdata$Error, seq(-100, 100, 20), right = FALSE, labels = c(1:10))
library("ggplot2")
g3 <- ggplot(testdata, aes(x=Temp, y=Wind)) + geom_point(aes(color=ErrorLevel, size=ErrorLevel)) + ggti
```

```
## Group all plots together
library("gridExtra")
grid.arrange(g1, g2, g3, nrow = 1)
```

```
## Warning: Using size for a discrete variable is not advised.

## Warning: Using size for a discrete variable is not advised.

## Warning: Using size for a discrete variable is not advised.
```




```
# Step 4: Create a 'goodOzone' variable
air2 <- air # create new dataset to remove all added columns
meanOzone <- mean(air2$Ozone)
air2$OzoneCategorical <- cut(air2$Ozone, c(min(air2$Ozone),meanOzone, max(air2$Ozone)), right = FALSE,
# Make sure there were no NAs created
colSums(is.na(air2))
```

```
##            Ozone          Solar.R             Wind             Temp
##                0                0                0                0
##            Month              Day OzoneCategorical
##                0                0                1
```

```r
# Find and properly assign NA
napoint <- which(is.na(air2$OzoneCategorical))
air2$Ozone[napoint]
```

```
## [1] 168
```

```r
air2$OzoneCategorical[napoint] <- 1

##################################################################################



### Step 5: Predict 'good' and 'bad' days
# create new testing & training data with categorical data included
randIndex <- sample(1:dim(air2)[1]) # Create list/vector variable-random index
summary(randIndex) # verify indicies in randIndex
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     1.0    28.5    56.0    56.0    83.5   111.0
```

```r
cutPoint2_3 <- floor(2*dim(air2)[1]/3) # create 2/3rd cutpoint; floor rounds number to the lower intege
traindata_cat <- air2[randIndex[1:cutPoint2_3],] # create training data set
testdata_cat <- air2[randIndex[(cutPoint2_3+1):dim(air2)[1]],] # create test data set


# Create KSVM (support vector machine)
svmOutput_cat <- ksvm(OzoneCategorical~., data=traindata_cat, kernel="rbfdot", kpar="automatic", C=5, c
svmOutput_cat
```

```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 5
##
## Gaussian Radial Basis kernel function.
##  Hyperparameter : sigma =  0.138871509950104
##
## Number of Support Vectors : 24
##
## Objective Function Value : -35.322
## Training error : 0
## Cross validation error : 0.080556
## Probability model included.
```

```r
# Predict how accurate we are w/test data when C=5
svmPred_cat <- predict(svmOutput_cat, testdata_cat,type="response") # Use type="response" NOT type="vot
compTable_cat <- data.frame(testdata_cat$OzoneCategorical, svmPred_cat)
a <- table(compTable_cat)

# Percent Ozone correctly predicted
goodozone_per <- 100*(a[1,1]/(a[1,1] + a[1,2]))
message("Percent of Good Ozone predicted = ", goodozone_per, "%")
```

```
## Percent of Good Ozone predicted = 95.4545454545455%
```

```r
# Create Good/Bad day labels
testdata_cat$Label <- rep(0,nrow(testdata_cat)) # input a Label column filled with 0s
i <- 1 # Create a while loop that basis Good/Bad Days on Ozone Categorical field
while (i < nrow(testdata_cat)+1) {
 if (testdata_cat$Ozone[i] < meanOzone)
{
  testdata_cat$Label[i] <- "Good Day"
 } else {testdata_cat$Label[i] <- "Bad Day"}
  i <- i+1
}

index <- which(compTable_cat[1] == compTable_cat[2])
testdata_cat$Correct_Pred[index] <- "Correct"
napoints <- which(is.na(testdata_cat$Correct_Pred))
testdata_cat$Correct_Pred[napoints] <- "Wrong"
# Plot the results using a scatter plot
g4 <- ggplot(testdata_cat, aes(x=Temp, y=Wind)) + geom_point(aes(shape=Label,color=Ozone, size=Correct_
##########################################################################################
##########################################################################################

# Re-create test and train data set from air2 dataset and remodel
# randIndex <- sample(1:dim(air2)[1]) # Create list/vector variable-random index
# summary(randIndex) # verify indicies in randIndex
# cutPoint2_3 <- floor(2*dim(air2)[1]/3) # create 2/3rd cutpoint; floor rounds number to the lower inte
# traindata_cat <- air2[randIndex[1:cutPoint2_3],] # create training data set
# testdata_cat <- air2[randIndex[(cutPoint2_3+1):dim(air2)[1]],] # create test data set



# Create SVM (support vector machine)
svm_cat <- svm(OzoneCategorical~., data=traindata_cat, cost=5, cross=3, prob.model=TRUE)
svm_cat
```

```
##
## Call:
## svm(formula = OzoneCategorical ~ ., data = traindata_cat, cost = 5,
##     cross = 3, prob.model = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  5
##       gamma:  0.1666667
##
## Number of Support Vectors:  24
```

```r
# Remove extra columns from testdata_cat
testdata_cat <- testdata_cat[ , -length(testdata_cat)+1:-length(testdata_cat)] # revert testdata_cat ba

# Predict how accurate we are w/test data when cost=5
```

```r
svmPred_cat <- predict(svm_cat, testdata_cat,type="response")
svmPred_cat <- data.frame(svmPred_cat)
CompTable4 <- data.frame(testdata_cat$OzoneCategorical, svmPred_cat)
b <- table(CompTable4)

# Percent Ozone correctly predicted
goodozone_per2 <- 100*(b[1,1]/(b[1,1] + b[1,2]))
message("Percent of Good Ozone predicted = ", goodozone_per2, "%")
```

```
## Percent of Good Ozone predicted = 95.4545454545455%
```

```r
# Create Good/Bad day labels
testdata_cat$Label <- rep(0,nrow(testdata_cat)) # input a Label column filled with 0s
i <- 1 # Create a while loop that basis Good/Bad Days on Ozone Categorical field
while (i < nrow(testdata_cat)+1) {
  if (testdata_cat$Ozone[i] < meanOzone)
  {
    testdata_cat$Label[i] <- "Good Day"
  } else {testdata_cat$Label[i] <- "Bad Day"}
  i <- i+1
}

index <- which(CompTable4[1] == CompTable4[2])
testdata_cat$Correct_Pred[index] <- "Correct"
napoints <- which(is.na(testdata_cat$Correct_Pred))
testdata_cat$Correct_Pred[napoints] <- "Wrong"
# Plot the results using a scatter plot
g5 <- ggplot(testdata_cat, aes(x=Temp, y=Wind)) + geom_point(aes(shape=Label,color=Ozone, size=Correct_

# Create NB (Naive Bayes)
svm_cat2 <- naiveBayes(OzoneCategorical~., data=traindata_cat)
svm_cat2
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##         0         1
## 0.6351351 0.3648649
##
## Conditional probabilities:
##    Ozone
## Y       [,1]      [,2]
##   0 21.10638  9.974389
##   1 73.59259 25.837152
##
##    Solar.R
## Y       [,1]     [,2]
```

```
##   0 150.2979 98.28440
##   1 211.6296 50.10923
##
##    Wind
## Y         [,1]     [,2]
##   0 11.080851 2.860422
##   1  7.822222 3.275942
##
##    Temp
## Y        [,1]     [,2]
##   0 73.29787 7.700691
##   1 86.48148 5.109346
##
##    Month
## Y        [,1]      [,2]
##   0 6.893617 1.5774779
##   1 7.740741 0.9443187
##
##    Day
## Y        [,1]      [,2]
##   0 14.87234  7.603252
##   1 17.44444 10.591482
```

```
# Remove extra columns from testdata_cat
testdata_cat <- testdata_cat[ , -length(testdata_cat)+1:-length(testdata_cat)] # revert testdata_cat ba

# Predict how accurate we are w/test data when cost=5
svmPred_cat2 <- predict(svm_cat2, testdata_cat)
svmPred_cat2 <- data.frame(svmPred_cat2)
CompTable5 <- data.frame(testdata_cat$OzoneCategorical, svmPred_cat2)
c <- table(CompTable5)

# Percent Ozone correctly predicted
goodozone_per3 <- 100*(c[1,1]/(c[1,1] + c[1,2]))
message("Percent of Good Ozone predicted = ", goodozone_per3, "%")
```

```
## Percent of Good Ozone predicted = 100%
```

```
# Create Good/Bad day labels
testdata_cat$Label <- rep(0,nrow(testdata_cat)) # input a Label column filled with 0s
i <- 1 # Create a while loop that basis Good/Bad Days on Ozone Categorical field
while (i < nrow(testdata_cat)+1) {
  if (testdata_cat$Ozone[i] < meanOzone)
  {
    testdata_cat$Label[i] <- "Good Day"
  } else {testdata_cat$Label[i] <- "Bad Day"}
  i <- i+1
}

index <- which(CompTable5[1] == CompTable5[2])
testdata_cat$Correct_Pred[index] <- "Correct"
napoints <- which(is.na(testdata_cat$Correct_Pred))
testdata_cat$Correct_Pred[napoints] <- "Wrong"
```

```r
# Plot the results using a scatter plot
g6 <- ggplot(testdata_cat, aes(x=Temp, y=Wind)) + geom_point(aes(shape=Label,color=Ozone, size=Correct_
modelnames2 <- c("KSVM", "SVM", "NB")
RMSE <- c(goodozone_per, goodozone_per2, goodozone_per3)
data.frame(modelnames2, RMSE)
```

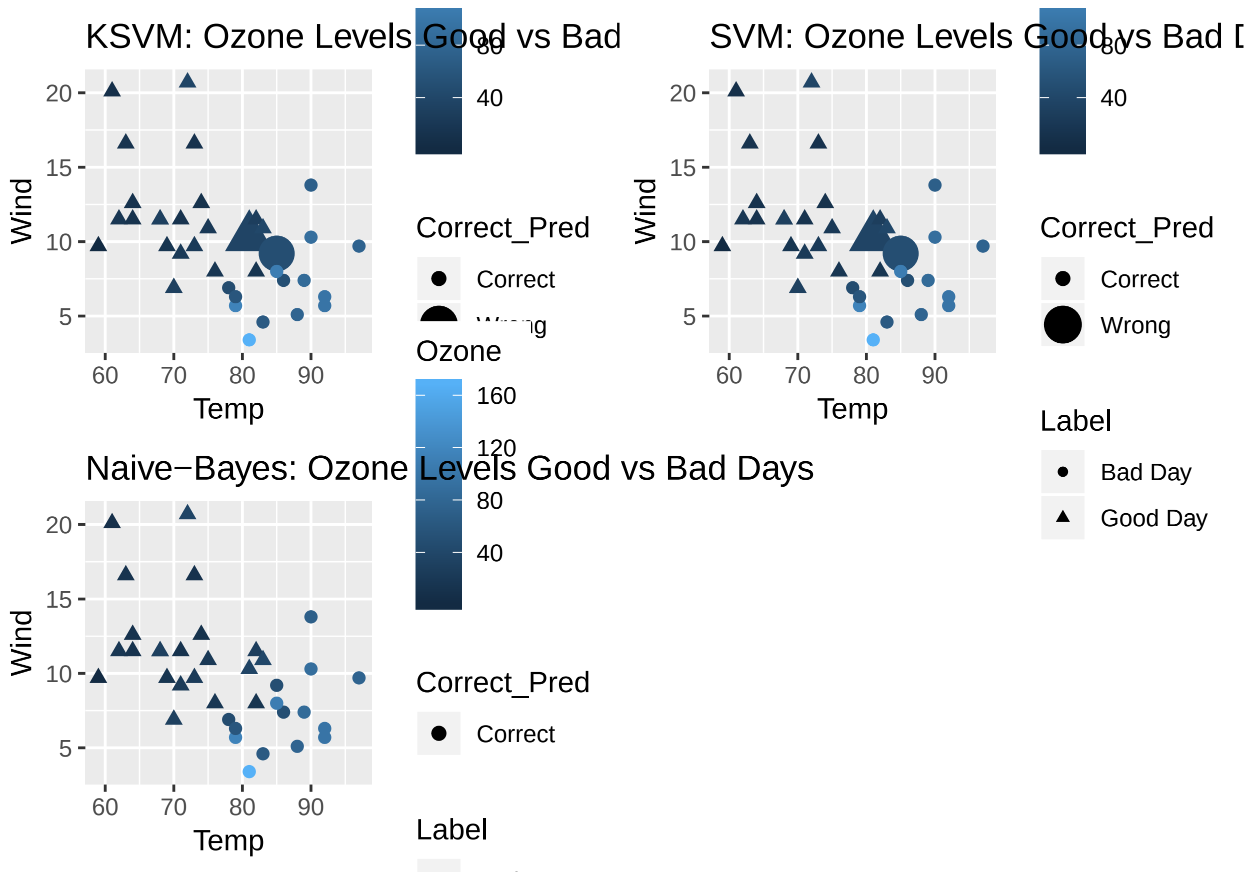
```
##   modelnames2      RMSE
## 1        KSVM  95.45455
## 2         SVM  95.45455
## 3          NB 100.00000
```

```r
## Group all plots together
library("gridExtra")
grid.arrange(g4, g5, g6, nrow = 2)
```

```
## Warning: Using size for a discrete variable is not advised.

## Warning: Using size for a discrete variable is not advised.

## Warning: Using size for a discrete variable is not advised.
```



Step 6: Conclusions

When using the airquality dataset I found that all 3 models generated very similar results, with there being on average (ran multiple simulations) a 2% range between all 3 models. A more in depth dive into how these algorithms are arriving at their outputs is needed to decide which one is outperforming the others. For the purpose of this exercise all 3 models prove effective in predicting ozone concentration levels.

When comparing a kernel support vector machine (KSVM), a SVM, and a Naive Bayes(NB) supervised learning model on their ability to predict the percent of “Good” Ozone days in a subset of the airquality dataset there really isn’t much difference either. All three models generally (ran model several times) produced around the same result of ~92%-100% accuracy.
However, based on the entire dataset and not just the subset of “Good” days the Naive Bayes model does prove to consistenly predict ~5% better. There appears to be 2 data points that hug very closely to the classifying line/separator that KSVM and SVM consistently get wrong whereas NB gets them right a majority of the time. With this in mind, a NB supervised learning model would be the method of choice for trying to predict “Good” and “Bad” Ozone level days in NY.