IST 687
Homework 9
Due Date: 12/7

Code requires the following packages to run

```
library(arules) #install.packages('arules')
library(arulesViz) #install.packages('arulesViz')
library(kernlab) #install.packages('kernlab')
library(ggplot2) #install.packages('ggplot2')
library(e1071) #install.packages('e1071')
library(gridExtra) #install.packages('gridExtra')
library(caret) #install.packages('caret')
```

Step 1 load the data

```
#load the data and pad the NAs
aq <- airquality #store the data as a new variable
aq$Ozone[which(is.na(aq$Ozone))] <- as.integer(mean(aq$Ozone, na.rm = TRUE)) #Replace NAs to the
mean for Ozone column
aq$Solar.R[which(is.na(aq$Solar.R))] <- as.integer(mean(aq$Solar.R, na.rm = TRUE)) #Replace NA's to th
e mean for Solar.R column
```

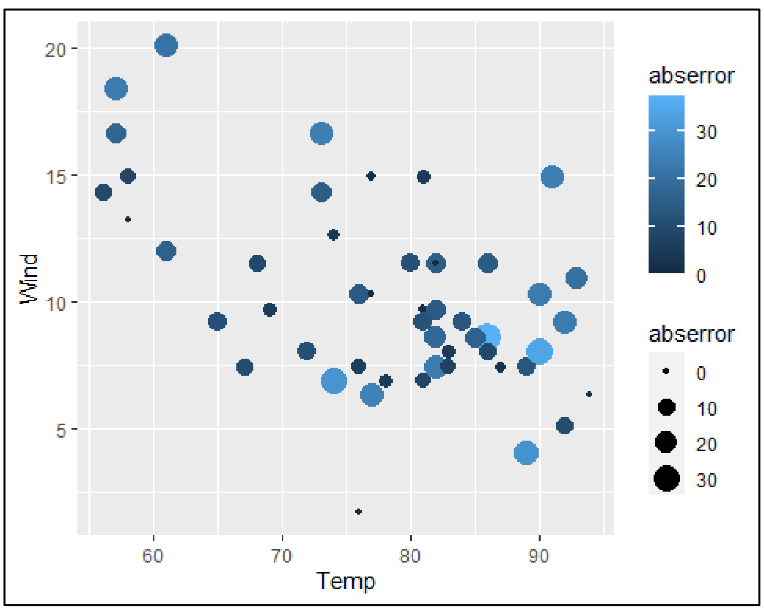
Step 2 create train and test data sets

```
#create list/vector variable random index
randIndex <- sample(1:dim(aq)[1])

#verify the random index
summary(randIndex)
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1      39      77      77     115     153
length(randIndex)
## [1] 153
head(randIndex)
## [1]  78  66 107  20  28 102

#establish the 2/3 cut point
cutPoint2_3 <- floor(2*dim(aq)[1]/3)
cutPoint2_3

## [1] 102



#create the training and test sets
trainData <- aq[randIndex[1:cutPoint2_3],]
trainData <- trainData[,-5:-6]
```

```
testDataKSVM <- aq[randIndex[(cutPoint2_3+1):dim(aq)[1]],]
testDataKSVM <- testDataKSVM[,-5:-6]
testDataSVM <- testDataKSVM
testDataLM <- testDataKSVM
```

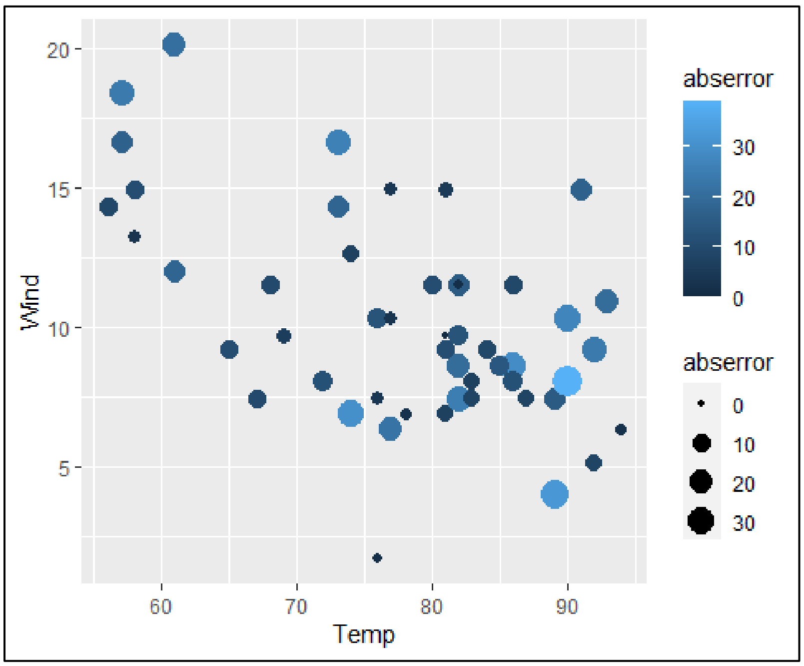
Step 3 build a model using KSVM & Visualize the results

```
#build KSVM model based on the training dataset
modelKSVM <- ksvm(Ozone ~ ., data = trainData)

#test KSVM model on the testing dataset
testDataKSVM$predictedOzone <- round(predict(modelKSVM, testDataKSVM))

#compute KSVM model root mean squared error
testDataKSVM$error <- testDataKSVM$predictedOzone-testDataKSVM$Ozone
testDataKSVM$abserror <- abs(testDataKSVM$error)
testDataKSVM$sqerror <- testDataKSVM$error^2
sqrt(mean(testDataKSVM$sqerror))

## [1] 16.01164

#plot the KSVM model results with a scatter plot
plotKSVM <- ggplot(testDataKSVM, aes(x=Temp, y=Wind, size=abserror, color=abserror))+geom_point()
plotKSVM
```



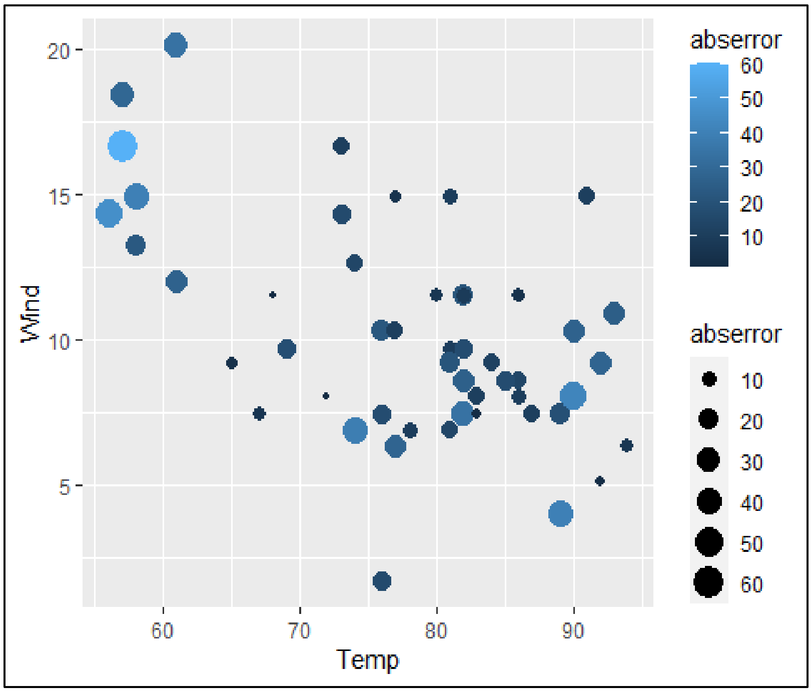

```
#build SVM model based on the training dataset
modelSVM <- svm(Ozone ~ ., data = trainData)
```

```
#test SVM model on the testing dataset
testDataSVM$predictedOzone <- round(predict(modelSVM, testDataSVM))

#compute SVM model root mean squared error
testDataSVM$error <- testDataSVM$predictedOzone-testDataSVM$Ozone
testDataSVM$abserror <- abs(testDataSVM$error)
testDataSVM$sqerror <- testDataSVM$error^2
sqrt(mean(testDataSVM$sqerror))

## [1] 16.06299

#plot the SVM model results with a scatter plot
plotSVM <- ggplot(testDataSVM, aes(x=Temp, y=Wind, size=abserror, color=abserror))+geom_point()
plotSVM
```

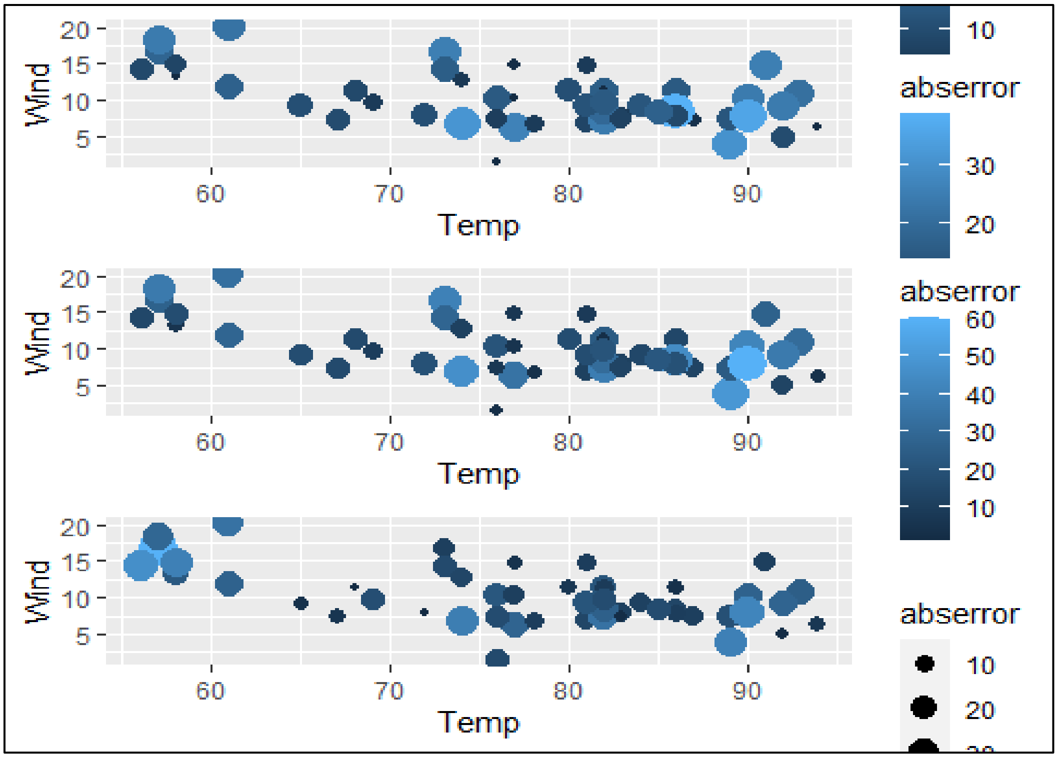



```
#build LM model based on the training dataset
modelLM <- lm(formula=Ozone ~ Solar.R + Wind + Temp, data=trainData)

#test LM model on the testing dataset
testDataLM$predictedOzone <- round(predict(modelLM, testDataLM))

#compute LM model root mean squared error
testDataLM$error <- testDataLM$predictedOzone-testDataLM$Ozone
testDataLM$abserror <- abs(testDataLM$error)
```

```
testDataLM$sqerror <- testDataLM$error^2
sqrt(mean(testDataLM$sqerror))

## [1] 22.44601

#plot the LM  model results with a scatter plot
plotLM <- ggplot(testDataLM, aes(x=Temp, y=Wind, size=abserror, color=abserror))+geom_point()
plotLM
```




```
#show the three plots on the same view
grid.arrange(plotKSVM, plotSVM, plotLM)
```



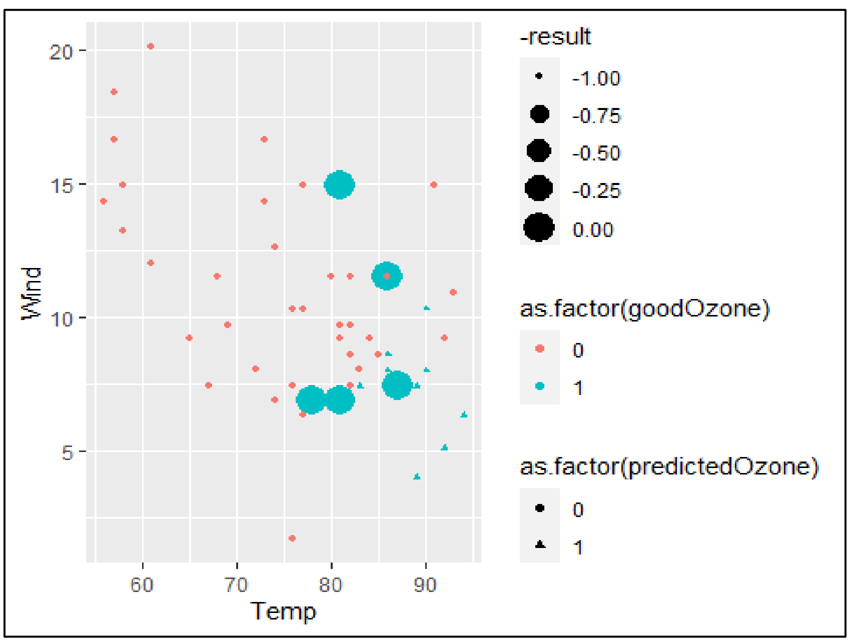
Step 4 Create a good ozone variable

```
#store new data frame for this exercise
aqnew <- airquality
aqnew$Ozone[which(is.na(aqnew$Ozone))] <- as.integer(mean(aqnew$Ozone, na.rm = TRUE))
aqnew$Solar.R[which(is.na(aqnew$Solar.R))] <- as.integer(mean(aqnew$Solar.R, na.rm = TRUE))

#create function for good or bad and apply it
mean(aqnew$Ozone)

## [1] 42.09804

GoodorBad <- function(number){
if(number>=42.09804)
return('1')
return('0')}
aqnew$goodOzone <- as.integer(sapply(aqnew$Ozone, FUN = GoodorBad))




#create the training and test sets
trainDatanew <- aqnew[randIndex[1:cutPoint2_3],]
trainDatanew <- trainDatanew[,-5:-6]
```

```
testDatanewKSVM <- aqnew[randIndex[(cutPoint2_3+1):dim(aqnew)[1]],]
testDatanewKSVM <- testDatanewKSVM[,-5:-6]
testDatanewSVM <- testDatanewKSVM
testDatanewNB <- testDatanewKSVM
```

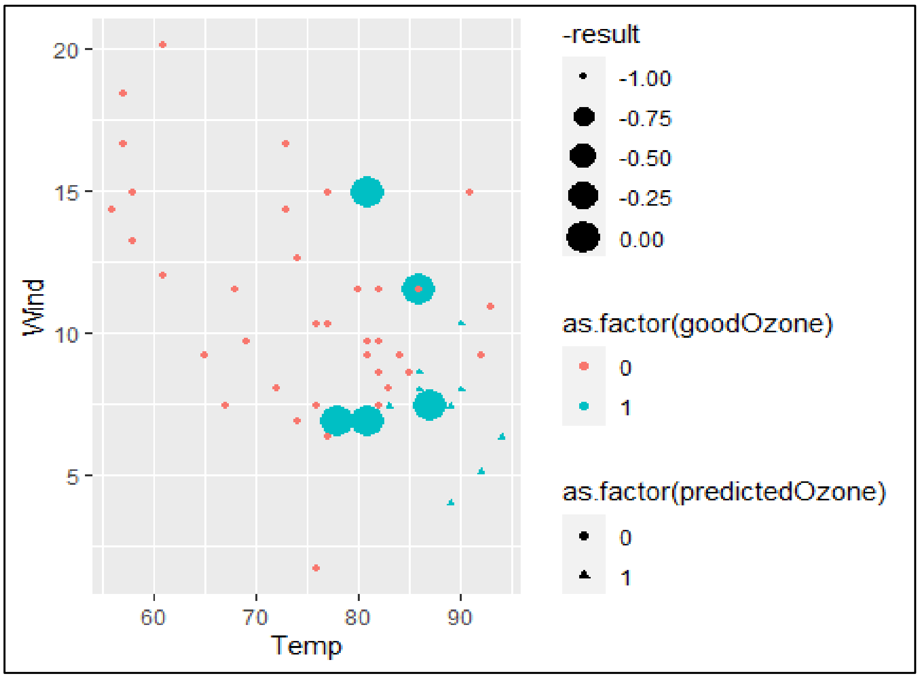
Step 5 see if we can do a better job predicting good and bad days

```
#build KSVM model based on the training dataset
modelKSVMnew <- ksvm(goodOzone ~ ., data = trainDatanew)

#test KSVM model on the testing dataset
testDatanewKSVM$predictedOzone <- round(predict(modelKSVMnew, testDatanewKSVM))

#compute KSVM model percent correct
testDatanewKSVM$result <- as.numeric(testDatanewKSVM$goodOzone == testDatanewKSVM$predicte
dOzone)
(sum(as.numeric(testDatanewKSVM$goodOzone == testDatanewKSVM$predictedOzone))/length(testD
atanewKSVM$Ozone)) *100

## [1] 90.19608

#plot the KSVM model results with a scatter plot
plotKSVMnew <- ggplot(testDatanewKSVM, aes(x=Temp, y=Wind, shape=as.factor(predictedOzone), col
or=as.factor(goodOzone), size=-result))+geom_point()
plotKSVMnew
```




```
#build SVM model based on the training dataset
modelSVMnew <- svm(goodOzone ~ ., data = trainDatanew)
```

```
#test SVM model on the testing dataset
testDatanewSVM$predictedOzone <- round(predict(modelSVMnew, testDatanewSVM))

#compute SVM model percent correct
testDatanewSVM$result <- as.numeric(testDatanewSVM$goodOzone == testDatanewSVM$predictedOz
one)
(sum(as.numeric(testDatanewSVM$goodOzone == testDatanewSVM$predictedOzone))/length(testData
newSVM$Ozone)) *100

## [1] 90.19608

#plot the SVM model results with a scatter plot
plotSVMnew <- ggplot(testDatanewSVM, aes(x=Temp, y=Wind, shape=as.factor(predictedOzone), color
=as.factor(goodOzone), size=-result))+geom_point()
plotSVMnew
```

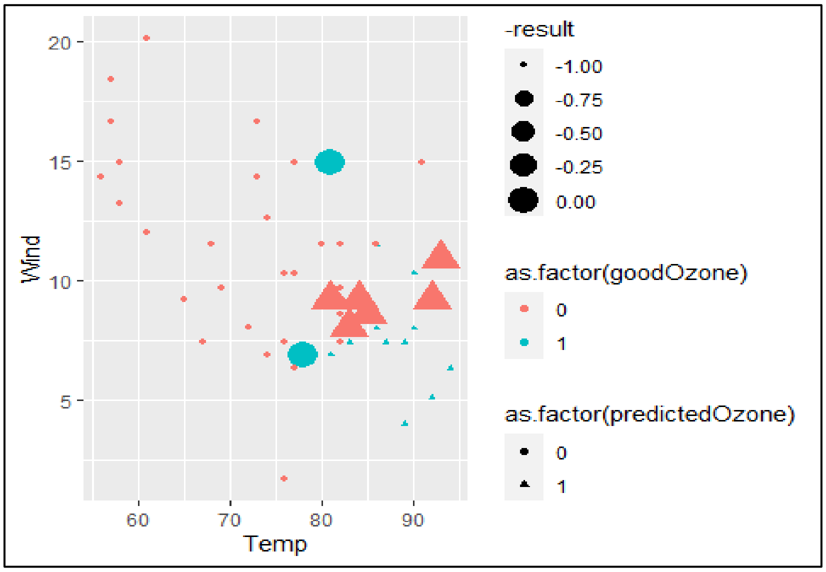



```
#build NB model based on the training dataset
modelNBnew <-naiveBayes(as.factor(goodOzone) ~ Temp + Solar.R + Wind, data = trainDatanew)

#test NB model on the testing dataset
testDatanewNB$predictedOzone <- predict(modelNBnew, testDatanewNB)



#compute NB model percent correct
testDatanewNB$result <- as.numeric(testDatanewNB$goodOzone == testDatanewNB$predictedOzone)
```

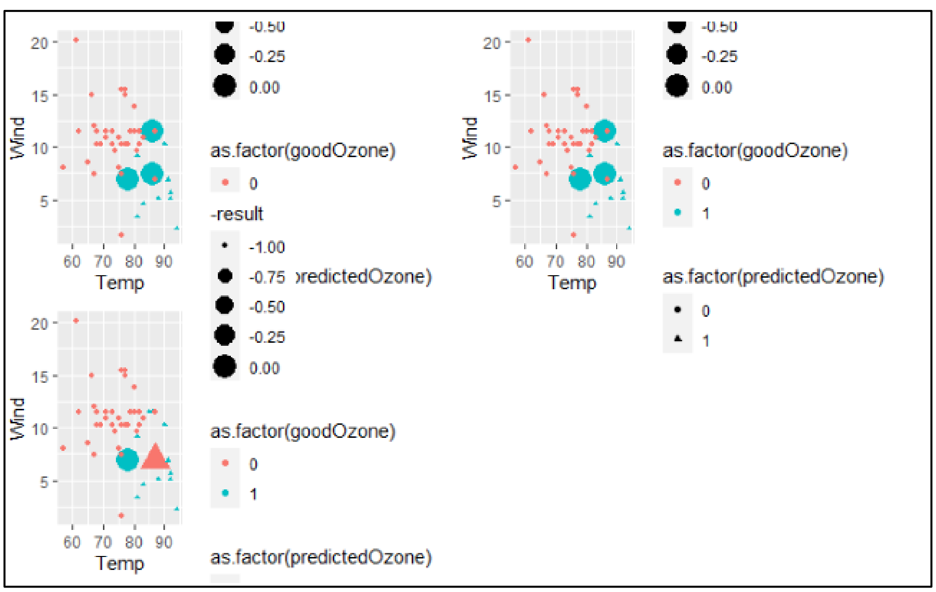
```r
(sum(as.numeric(testDatanewNB$goodOzone == testDatanewNB$predictedOzone))/length(testDatanewNB$Ozone)) *100
```

```
## [1] 84.31373
```

```r
#plot the NB model results with a scatter plot
plotNBnew <- ggplot(testDatanewNB, aes(x=Temp, y=Wind, shape=as.factor(predictedOzone), color=as.factor(goodOzone), size=-result))+geom_point()
plotNBnew
```




```r
#show the three plots on the same view
grid.arrange(plotKSVMnew, plotSVMnew, plotNBnew, nrow=2)
```



Step 6: which are the best models for this data?

Overall, the accuracy of the machine learning models were better than that of the regression model. Between the machine learning models, the margin of error was not that much different, and the result can vary with some amount of randomness. It depends on the situation which one might be better.

In the case of predicting a continuous ozone value, I would argue that the support vector machine (SVM) was the best because it produced the highest accuracy. On the other hand, in the case of the good versus bad ozone, I would say that the support vector machine (SVM) was also the best because it produced the highest accuracy.

In any case, I think that more exploration should be done to be able to say which model is best, but I don’t think you can go wrong with any of these as they are producing accuracies in the 90’s, which is very good considering the relatively small sample of data that is being used to train the models.