Jacob Dineen

Unexecuted, Executed, and Plots shown below

UNEXECUTED:

**#ADS IST 687**

**#Jacob Dineen**

**#Homework 9**

**#Due 9/17/2017**

**#################All Calls to Clear Environment and Fetch Packages**

**#CLEAR ENVIRONMENT AND INSTALL INITIAL PACKAGES**

**rm(list = ls(all = TRUE))#Clear Enviroment**

**# install.packages("kernlab")**

**# library("kernlab")**

**# library(e1071)**

**# library(ggplot2)**

**# library(gridExtra)**

**# Packages: kernlab, e1071, gridExtra, ggplot2, caret**

**#specify the packages of interest**

**packages=c("kernlab","e1071","gridExtra","ggplot2", "caret")**

**#use this function to check if each package is on the local machine**

**#if a package is installed, it will be loaded**

**#if any are not, the missing package(s) will be installed and loaded**

**package.check <- lapply(packages, FUN = function(x) {**

**if (!require(x, character.only = TRUE)) {**

**install.packages(x, dependencies = TRUE)**

**library(x, character.only = TRUE)**

**}**

**})**

**#verify they are loaded**

**search()**

**library("kernlab")**

**install.packages("kernlab")**

**########################################################## Step 1: Load the data**

**air <- data.frame(airquality)**

**# find which columns in the dataframe contain NAs.**

**colnames(air)[colSums(is.na(air)) > 0]**

**# find the NAs in column "Ozone" and "Solar" and replace them by the mean value of this column**

**air$Ozone[is.na(air$Ozone)] <- mean(air$Ozone, na.rm=TRUE)**

**air$Solar.R[is.na(air$Solar.R)] <- mean(air$Solar.R, na.rm=TRUE)**

**#Rerun Check on NAs**

**colnames(air)[colSums(is.na(air)) > 0]**

**########################################################## Step 2: Create Train and Test Data Sets**

**nrows <- nrow(air)**

**random.index <- sample(1:nrows)**

**head(random.index)**

**cutPoint <- floor(nrows/3\*2)**

**#Training Data (2/3 of total data sampled)**

**air.trainingdata <- air[random.index[1:cutPoint],]**

**dim(air.trainingdata)**

**#Testing Data (1/3 of total data sampled)**

**air.testingdata <- air[random.index[(cutPoint+1):nrows],]**

**dim(air.testingdata)**

**#root mean squared error function**

**rmse <- function(error)**

**{**

**sqrt(mean(error^2))**

**}**

**########################################################## Step 3: Build a Model using KSVM and Visualize the Results**

**require(kernlab)**

**require(e1071)**

**require(ggplot2)**

**#Using KSVM**

**model.ksvm.train <- ksvm(Ozone ~., data=air.trainingdata) #building the model**

**model.ksvm.predict <- predict(model.ksvm.train, air.testingdata) #testing the model on the testing data**

**air.testingdata$error <- air.testingdata$Ozone - model.ksvm.predict #computing the error between the predicted vs actual**

**head(air.testingdata)**

**rmse(air.testingdata$error) #Computing RMSE. RMSE = 13.35492**

**ksvmgraph <- ggplot(air.testingdata, aes(x=Temp, y=Wind)) + geom\_point(aes(size=error, color=error)) + ggtitle("KSVM") #Plotting via Scatterplot**

**ksvmgraph # Point size and color shade illustrate how big is the error**

**#Using SVM**

**Model.svm.train <- svm(Ozone~.,data = air.trainingdata)**

**model.svm.predict <- predict(Model.svm.train, air.testingdata)**

**air.testingdata$error <- air.testingdata$Ozone - model.svm.predict**

**head(air.testingdata)**

**rmse(air.testingdata$error) #Computing RMSE. RMSE = 13.48514**

**svmgraph <- ggplot(air.testingdata, aes(x=Temp, y=Wind)) + geom\_point(aes(size=error, color=error)) + ggtitle("SVM") #Plotting via Scatterplot**

**svmgraph**

**#using LM**

**model.lm.train <- lm(Ozone~.,data = air.trainingdata)**

**model.lm.predict <- predict(model.lm.train, air.testingdata)**

**air.testingdata$error <- air.testingdata$Ozone - model.lm.predict**

**head(air.testingdata)**

**rmse(air.testingdata) #Computing RMSE. RMSE = 84.80184**

**lmgraph <- ggplot(air.testingdata, aes(x=Temp, y=Wind)) + geom\_point(aes(size=error, color=error)) + ggtitle("LM") #Plotting via Scatterplot**

**lmgraph**

**grid.arrange(ksvmgraph,svmgraph,lmgraph, nrow=2)**

**########################################################## Step 4 : Create a goodOzone Variable**

**meanozone <- mean(air$Ozone, na.rm=TRUE)**

**air.trainingdata$goodOzone <- ifelse(air.trainingdata$Ozone<meanozone, 0, 1)**

**air.testingdata$goodOzone <- ifelse(air.testingdata$Ozone<meanozone, 0, 1)**

**# convert "goodOzone" in train data from numeric to factor**

**air.trainingdata$goodOzone <- as.factor(air.trainingdata$goodOzone)**

**# convert "goodOzone" in test data from numeric to factor**

**air.testingdata$goodOzone <- as.factor(air.testingdata$goodOzone)**

**# remove "Ozone" from train data**

**air.trainingdata <- air.trainingdata[,-1]**

**# remove "Ozone" from test data**

**air.testingdata <- air.testingdata[,-1]**

**air.testingdata <- air.testingdata[,-6]**

**str(air.testingdata)**

**str(air.trainingdata)**

**########################################################## Step 5 : See if we can do a better job predicting good and bad days**

**#KSVM**

**model.ksvm.train <-ksvm(goodOzone~., data=air.trainingdata)**

**air.testingdata$predictedGoodOzone <- predict(model.ksvm.train, air.testingdata)**

**head(air.testingdata)**

**results <- table(air.testingdata$predictedGoodOzone, air.testingdata$goodOzone)**

**print(results)**

**percentCorrect <- (results[1,1]+results[2,2])/(results[1,1]+results[1,2]+results[2,1]+results[2,2])\*100**

**print(round(percentCorrect) )**

**compgood1 <- data.frame(air.testingdata[,6], air.testingdata$predictedGoodOzone)**

**colnames(compgood1) <- c("test","Pred")**

**compgood1$correct <- ifelse(compgood1$test==compgood1$Pred,"correct","wrong")**

**Plot\_ksvm <- data.frame(compgood1$correct,air.testingdata$Temp,air.testingdata$Wind,air.testingdata$goodOzone,compgood1$Pred)**

**colnames(Plot\_ksvm) <- c("correct","Temp","Wind","goodOzone","Predict")**

**ksvmgoodbadplot <- ggplot(Plot\_ksvm, aes(x=Temp,y=Wind)) +**

**geom\_point(aes(size=correct,color=goodOzone,shape = Predict))+**

**ggtitle("ksvm - good/bad ozone")**

**ksvmgoodbadplot**

**## 73% correct on our predictions using the KSVM model**

**#SVM**

**Model.svm.train <- svm(goodOzone~., data = air.trainingdata)**

**air.testingdata$predictedGoodOzone <- predict(Model.svm.train, air.testingdata)**

**head(air.testingdata)**

**results <- table(air.testingdata$predictedGoodOzone, air.testingdata$goodOzone)**

**print(results)**

**percentCorrect <- (results[1,1]+results[2,2])/(results[1,1]+results[1,2]+results[2,1]+results[2,2])\*100**

**print(round(percentCorrect) )**

**compgood2 <- data.frame(air.testingdata[,6], air.testingdata$predictedGoodOzone)**

**colnames(compgood2) <- c("test","Pred")**

**compgood2$correct <- ifelse(compgood2$test==compgood2$Pred,"correct","wrong")**

**Plot\_svm <- data.frame(compgood2$correct,air.testingdata$Temp,air.testingdata$Wind,air.testingdata$goodOzone,compgood2$Pred)**

**colnames(Plot\_svm) <- c("correct","Temp","Wind","goodOzone","Predict")**

**svmgoodbadplot <- ggplot(Plot\_svm, aes(x=Temp,y=Wind)) +**

**geom\_point(aes(size=correct,color=goodOzone,shape = Predict))+**

**ggtitle("svm - good/bad ozone")**

**svmgoodbadplot**

**## 73% correct on our predictions using the SVM model**

**#Naive Bayes**

**model.naivebayes.train <- naiveBayes(goodOzone~., data = air.trainingdata)**

**air.testingdata$predictedGoodOzone <- predict(model.naivebayes.train, air.testingdata)**

**head(air.testingdata)**

**results <- table(air.testingdata$predictedGoodOzone, air.testingdata$goodOzone)**

**print(results)**

**percentCorrect <- (results[1,1]+results[2,2])/(results[1,1]+results[1,2]+results[2,1]+results[2,2])\*100**

**print(round(percentCorrect) )**

**compgood3 <- data.frame(air.testingdata[,6], air.testingdata$predictedGoodOzone)**

**colnames(compgood3) <- c("test","Pred")**

**compgood3$correct <- ifelse(compgood3$test==compgood3$Pred,"correct","wrong")**

**Plot\_nb <- data.frame(compgood3$correct,air.testingdata$Temp,air.testingdata$Wind,air.testingdata$goodOzone,compgood3$Pred)**

**colnames(Plot\_nb) <- c("correct","Temp","Wind","goodOzone","Predict")**

**nbgoodbadplot <- ggplot(Plot\_nb, aes(x=Temp,y=Wind)) +**

**geom\_point(aes(size=correct,color=goodOzone,shape = Predict))+**

**ggtitle("naive bayes - good/bad ozone")**

**nbgoodbadplot**

**## 69% correct on our predictions using the Naive Bayes model**

**grid.arrange(ksvmgoodbadplot,svmgoodbadplot,nbgoodbadplot, nrow=2)**

**########################################################## Step 6 : Which are the best models for this data**

**#Originally when we were using RMSE as our indicator of good vs bad models, the KSVM model had the lowest value, but the SVM model was very close.**

**#When we predicted our training data against actual values, we saw that both the KSVM and SVM models had an accuracy rating of about 73%, while the**

**#Naive Bayes model had a rating of 69% correct. It should be noted that rerunning these models changes the predicted values, and if you rerun them enough**

**#you're likely to find a higher success rating from different models.**

Executed:

#ADS IST 687

> #Jacob Dineen

> #Homework 9

> #Due 9/17/2017

>

> #################All Calls to Clear Environment and Fetch Packages

> #CLEAR ENVIRONMENT AND INSTALL INITIAL PACKAGES

> rm(list = ls(all = TRUE))#Clear Enviroment

>

> # install.packages("kernlab")

> # library("kernlab")

> # library(e1071)

> # library(ggplot2)

> # library(gridExtra)

>

> # Packages: kernlab, e1071, gridExtra, ggplot2, caret

>

> #specify the packages of interest

> packages=c("kernlab","e1071","gridExtra","ggplot2", "caret")

>

> #use this function to check if each package is on the local machine

> #if a package is installed, it will be loaded

> #if any are not, the missing package(s) will be installed and loaded

> package.check <- lapply(packages, FUN = function(x) {

+ if (!require(x, character.only = TRUE)) {

+ install.packages(x, dependencies = TRUE)

+ library(x, character.only = TRUE)

+ }

+ })

Loading required package: caret

Loading required package: lattice

Warning message:

package ‘caret’ was built under R version 3.3.3

>

> #verify they are loaded

> search()

[1] ".GlobalEnv" "package:caret" "package:lattice" "package:gridExtra" "package:ggplot2" "package:e1071" "package:kernlab"

[8] "tools:rstudio" "package:stats" "package:graphics" "package:grDevices" "package:utils" "package:datasets" "package:methods"

[15] "Autoloads" "package:base"

>

> library("kernlab")

> install.packages("kernlab")

Error in install.packages : Updating loaded packages

>

> ########################################################## Step 1: Load the data

> air <- data.frame(airquality)

> # find which columns in the dataframe contain NAs.

> colnames(air)[colSums(is.na(air)) > 0]

[1] "Ozone" "Solar.R"

> # find the NAs in column "Ozone" and "Solar" and replace them by the mean value of this column

> air$Ozone[is.na(air$Ozone)] <- mean(air$Ozone, na.rm=TRUE)

> air$Solar.R[is.na(air$Solar.R)] <- mean(air$Solar.R, na.rm=TRUE)

> #Rerun Check on NAs

> colnames(air)[colSums(is.na(air)) > 0]

character(0)

>

> ########################################################## Step 2: Create Train and Test Data Sets

> nrows <- nrow(air)

> random.index <- sample(1:nrows)

> head(random.index)

[1] 99 53 19 31 135 8

> cutPoint <- floor(nrows/3\*2)

>

> #Training Data (2/3 of total data sampled)

> air.trainingdata <- air[random.index[1:cutPoint],]

> dim(air.trainingdata)

[1] 102 6

> #Testing Data (1/3 of total data sampled)

> air.testingdata <- air[random.index[(cutPoint+1):nrows],]

> dim(air.testingdata)

[1] 51 6

>

> #root mean squared error function

> rmse <- function(error)

+ {

+ sqrt(mean(error^2))

+ }

>

> ########################################################## Step 3: Build a Model using KSVM and Visualize the Results

> require(kernlab)

> require(e1071)

> require(ggplot2)

>

> #Using KSVM

> model.ksvm.train <- ksvm(Ozone ~., data=air.trainingdata) #building the model

> model.ksvm.predict <- predict(model.ksvm.train, air.testingdata) #testing the model on the testing data

> air.testingdata$error <- air.testingdata$Ozone - model.ksvm.predict #computing the error between the predicted vs actual

> head(air.testingdata)

Ozone Solar.R Wind Temp Month Day error

88 52.00000 82 12.0 86 7 27 7.933790

146 36.00000 139 10.3 81 9 23 -1.460064

137 9.00000 24 10.9 71 9 14 -1.580628

144 13.00000 238 12.6 64 9 21 -2.379152

71 85.00000 175 7.4 89 7 10 18.709533

39 42.12931 273 6.9 87 6 8 -18.235465

> rmse(air.testingdata$error) #Computing RMSE. RMSE = 13.35492

[1] 18.31736

> ksvmgraph <- ggplot(air.testingdata, aes(x=Temp, y=Wind)) + geom\_point(aes(size=error, color=error)) + ggtitle("KSVM") #Plotting via Scatterplot

> ksvmgraph # Point size and color shade illustrate how big is the error

>

> #Using SVM

> Model.svm.train <- svm(Ozone~.,data = air.trainingdata)

> model.svm.predict <- predict(Model.svm.train, air.testingdata)

> air.testingdata$error <- air.testingdata$Ozone - model.svm.predict

> head(air.testingdata)

Ozone Solar.R Wind Temp Month Day error

88 52.00000 82 12.0 86 7 27 10.609910

146 36.00000 139 10.3 81 9 23 1.380479

137 9.00000 24 10.9 71 9 14 -5.577574

144 13.00000 238 12.6 64 9 21 -7.872460

71 85.00000 175 7.4 89 7 10 17.918509

39 42.12931 273 6.9 87 6 8 -16.861711

> rmse(air.testingdata$error) #Computing RMSE. RMSE = 13.48514

[1] 18.36703

> svmgraph <- ggplot(air.testingdata, aes(x=Temp, y=Wind)) + geom\_point(aes(size=error, color=error)) + ggtitle("SVM") #Plotting via Scatterplot

> svmgraph

>

> #using LM

> model.lm.train <- lm(Ozone~.,data = air.trainingdata)

> model.lm.predict <- predict(model.lm.train, air.testingdata)

> air.testingdata$error <- air.testingdata$Ozone - model.lm.predict

> head(air.testingdata)

Ozone Solar.R Wind Temp Month Day error

88 52.00000 82 12.0 86 7 27 3.422741

146 36.00000 139 10.3 81 9 23 -7.358831

137 9.00000 24 10.9 71 9 14 -8.932043

144 13.00000 238 12.6 64 9 21 -2.617270

71 85.00000 175 7.4 89 7 10 22.424975

39 42.12931 273 6.9 87 6 8 -24.002370

> rmse(air.testingdata) #Computing RMSE. RMSE = 84.80184

[1] 83.06031

> lmgraph <- ggplot(air.testingdata, aes(x=Temp, y=Wind)) + geom\_point(aes(size=error, color=error)) + ggtitle("LM") #Plotting via Scatterplot

> lmgraph

>

> grid.arrange(ksvmgraph,svmgraph,lmgraph, nrow=2)

>

> ########################################################## Step 4 : Create a goodOzone Variable

> meanozone <- mean(air$Ozone, na.rm=TRUE)

> air.trainingdata$goodOzone <- ifelse(air.trainingdata$Ozone<meanozone, 0, 1)

> air.testingdata$goodOzone <- ifelse(air.testingdata$Ozone<meanozone, 0, 1)

>

> # convert "goodOzone" in train data from numeric to factor

> air.trainingdata$goodOzone <- as.factor(air.trainingdata$goodOzone)

> # convert "goodOzone" in test data from numeric to factor

> air.testingdata$goodOzone <- as.factor(air.testingdata$goodOzone)

>

>

> # remove "Ozone" from train data

> air.trainingdata <- air.trainingdata[,-1]

> # remove "Ozone" from test data

> air.testingdata <- air.testingdata[,-1]

> air.testingdata <- air.testingdata[,-6]

>

> str(air.testingdata)

'data.frame': 51 obs. of 6 variables:

$ Solar.R : num 82 139 24 238 175 ...

$ Wind : num 12 10.3 10.9 12.6 7.4 6.9 10.3 11.5 14.3 6.9 ...

$ Temp : int 86 81 71 64 89 87 77 71 56 86 ...

$ Month : int 7 9 9 9 7 6 8 9 5 8 ...

$ Day : int 27 23 14 21 10 8 16 15 5 4 ...

$ goodOzone: Factor w/ 2 levels "0","1": 2 1 1 1 2 2 1 1 2 2 ...

> str(air.trainingdata)

'data.frame': 102 obs. of 6 variables:

$ Solar.R : num 255 59 322 279 259 99 260 223 47 150 ...

$ Wind : num 4 1.7 11.5 7.4 15.5 13.8 6.9 11.5 10.3 6.3 ...

$ Temp : int 89 76 68 76 76 59 81 68 73 77 ...

$ Month : int 8 6 5 5 9 5 7 9 6 6 ...

$ Day : int 7 22 19 31 12 8 16 30 27 21 ...

$ goodOzone: Factor w/ 2 levels "0","1": 2 2 1 1 1 1 2 1 2 2 ...

>

> ########################################################## Step 5 : See if we can do a better job predicting good and bad days

>

> #KSVM

> model.ksvm.train <-ksvm(goodOzone~., data=air.trainingdata)

>

> air.testingdata$predictedGoodOzone <- predict(model.ksvm.train, air.testingdata)

> head(air.testingdata)

Solar.R Wind Temp Month Day goodOzone predictedGoodOzone

88 82 12.0 86 7 27 1 1

146 139 10.3 81 9 23 0 0

137 24 10.9 71 9 14 0 0

144 238 12.6 64 9 21 0 0

71 175 7.4 89 7 10 1 1

39 273 6.9 87 6 8 1 1

> results <- table(air.testingdata$predictedGoodOzone, air.testingdata$goodOzone)

> print(results)

0 1

0 21 5

1 3 22

>

> percentCorrect <- (results[1,1]+results[2,2])/(results[1,1]+results[1,2]+results[2,1]+results[2,2])\*100

> print(round(percentCorrect) )

[1] 84

>

> compgood1 <- data.frame(air.testingdata[,6], air.testingdata$predictedGoodOzone)

> colnames(compgood1) <- c("test","Pred")

> compgood1$correct <- ifelse(compgood1$test==compgood1$Pred,"correct","wrong")

> Plot\_ksvm <- data.frame(compgood1$correct,air.testingdata$Temp,air.testingdata$Wind,air.testingdata$goodOzone,compgood1$Pred)

> colnames(Plot\_ksvm) <- c("correct","Temp","Wind","goodOzone","Predict")

>

>

> ksvmgoodbadplot <- ggplot(Plot\_ksvm, aes(x=Temp,y=Wind)) +

+ geom\_point(aes(size=correct,color=goodOzone,shape = Predict))+

+ ggtitle("ksvm - good/bad ozone")

>

> ksvmgoodbadplot

Warning message:

Using size for a discrete variable is not advised.

>

>

> ## 73% correct on our predictions using the KSVM model

>

> #SVM

> Model.svm.train <- svm(goodOzone~., data = air.trainingdata)

> air.testingdata$predictedGoodOzone <- predict(Model.svm.train, air.testingdata)

> head(air.testingdata)

Solar.R Wind Temp Month Day goodOzone predictedGoodOzone

88 82 12.0 86 7 27 1 1

146 139 10.3 81 9 23 0 0

137 24 10.9 71 9 14 0 0

144 238 12.6 64 9 21 0 0

71 175 7.4 89 7 10 1 1

39 273 6.9 87 6 8 1 1

> results <- table(air.testingdata$predictedGoodOzone, air.testingdata$goodOzone)

> print(results)

0 1

0 21 5

1 3 22

> percentCorrect <- (results[1,1]+results[2,2])/(results[1,1]+results[1,2]+results[2,1]+results[2,2])\*100

> print(round(percentCorrect) )

[1] 84

>

> compgood2 <- data.frame(air.testingdata[,6], air.testingdata$predictedGoodOzone)

> colnames(compgood2) <- c("test","Pred")

> compgood2$correct <- ifelse(compgood2$test==compgood2$Pred,"correct","wrong")

> Plot\_svm <- data.frame(compgood2$correct,air.testingdata$Temp,air.testingdata$Wind,air.testingdata$goodOzone,compgood2$Pred)

> colnames(Plot\_svm) <- c("correct","Temp","Wind","goodOzone","Predict")

>

> svmgoodbadplot <- ggplot(Plot\_svm, aes(x=Temp,y=Wind)) +

+ geom\_point(aes(size=correct,color=goodOzone,shape = Predict))+

+ ggtitle("svm - good/bad ozone")

>

> svmgoodbadplot

Warning message:

Using size for a discrete variable is not advised.

>

>

> ## 73% correct on our predictions using the SVM model

>

> #Naive Bayes

> model.naivebayes.train <- naiveBayes(goodOzone~., data = air.trainingdata)

> air.testingdata$predictedGoodOzone <- predict(model.naivebayes.train, air.testingdata)

> head(air.testingdata)

Solar.R Wind Temp Month Day goodOzone predictedGoodOzone

88 82 12.0 86 7 27 1 1

146 139 10.3 81 9 23 0 1

137 24 10.9 71 9 14 0 0

144 238 12.6 64 9 21 0 0

71 175 7.4 89 7 10 1 1

39 273 6.9 87 6 8 1 1

> results <- table(air.testingdata$predictedGoodOzone, air.testingdata$goodOzone)

> print(results)

0 1

0 19 5

1 5 22

> percentCorrect <- (results[1,1]+results[2,2])/(results[1,1]+results[1,2]+results[2,1]+results[2,2])\*100

> print(round(percentCorrect) )

[1] 80

>

> compgood3 <- data.frame(air.testingdata[,6], air.testingdata$predictedGoodOzone)

> colnames(compgood3) <- c("test","Pred")

> compgood3$correct <- ifelse(compgood3$test==compgood3$Pred,"correct","wrong")

> Plot\_nb <- data.frame(compgood3$correct,air.testingdata$Temp,air.testingdata$Wind,air.testingdata$goodOzone,compgood3$Pred)

> colnames(Plot\_nb) <- c("correct","Temp","Wind","goodOzone","Predict")

>

> nbgoodbadplot <- ggplot(Plot\_nb, aes(x=Temp,y=Wind)) +

+ geom\_point(aes(size=correct,color=goodOzone,shape = Predict))+

+ ggtitle("naive bayes - good/bad ozone")

>

> nbgoodbadplot

Warning message:

Using size for a discrete variable is not advised.

>

> ## 69% correct on our predictions using the Naive Bayes model

>

> grid.arrange(ksvmgoodbadplot,svmgoodbadplot,nbgoodbadplot, nrow=2)

Warning messages:

1: Using size for a discrete variable is not advised.

2: Using size for a discrete variable is not advised.

3: Using size for a discrete variable is not advised.

>

> ########################################################## Step 6 : Which are the best models for this data

> #Originally when we were using RMSE as our indicator of good vs bad models, the KSVM model had the lowest value, but the SVM model was very close.

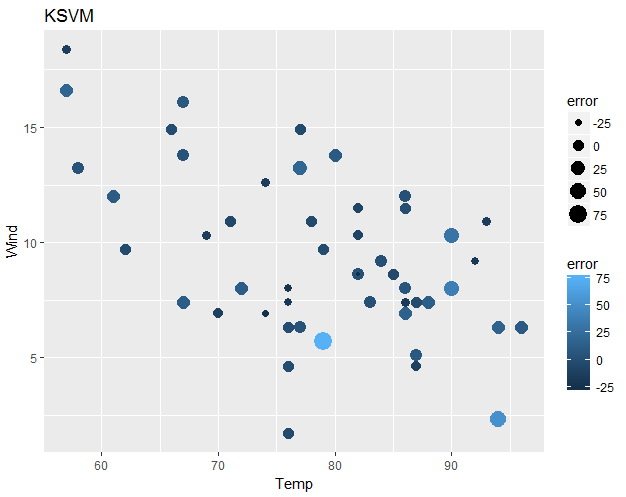
> #When we predicted our training data against actual values, we saw that both the KSVM and SVM models had an accuracy rating of about 73%, while the

> #Naive Bayes model had a rating of 69% correct. It should be noted that rerunning these models changes the predicted values, and if you rerun them enough

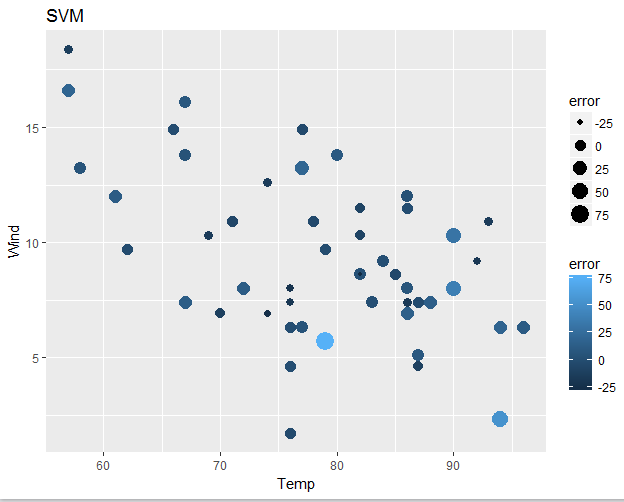
> #you're likely to find a higher success rating from different models.

**Plots:**

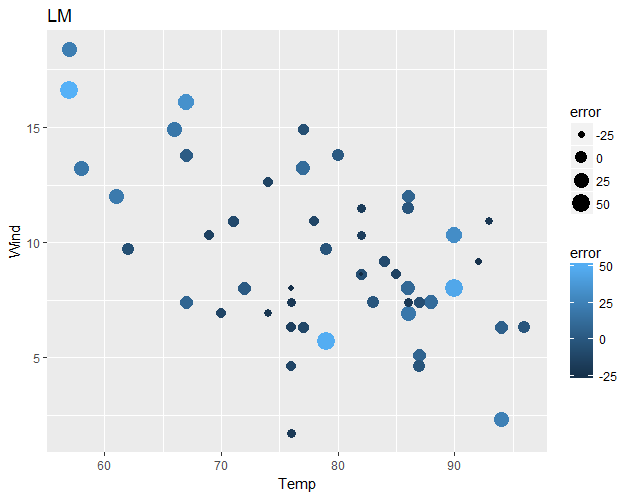
**Step 3 KSVM Graph:**



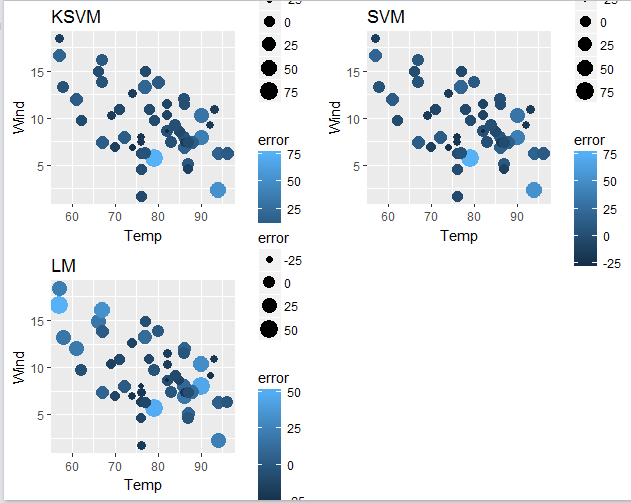
**Step 3 SVM Graph:**

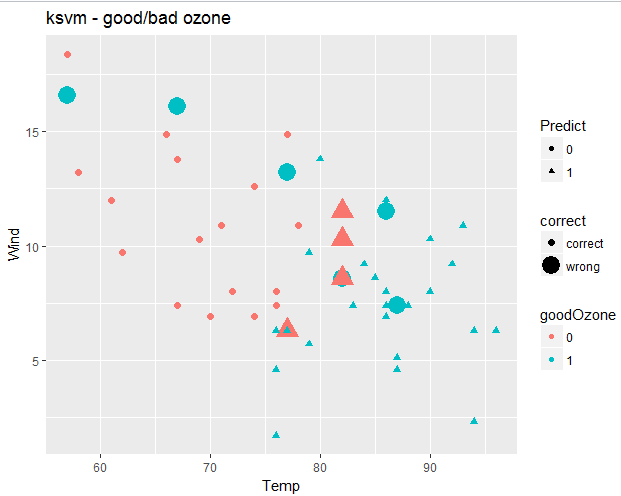


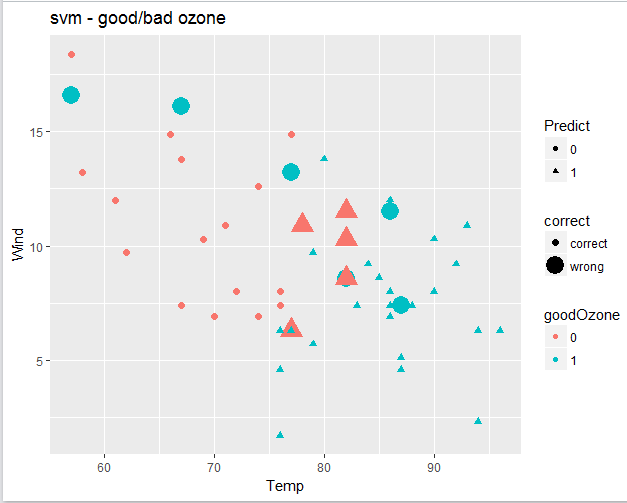
**Step 3 LM Graph:**

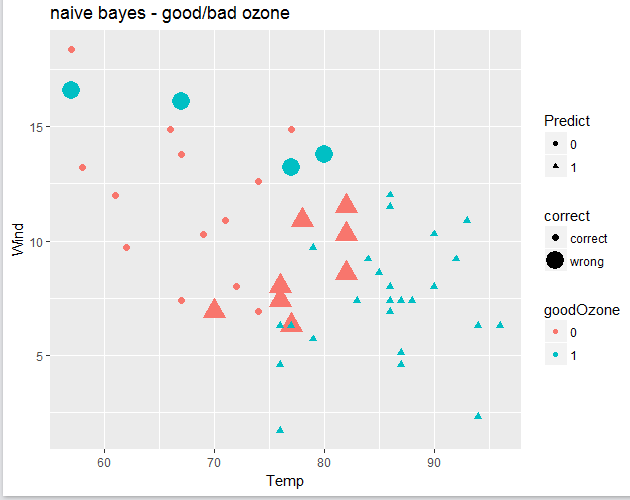


**Step 3 Grid Arrange:**









**Grid Arrange Step 5:**

