HW9

Diego Valdes

March 16, 2019

```
rm(list=ls()) # clear work space
#dev.off(dev.list()["RStudioGD"]) # clear plots

library(kernlab)
```

```
## Warning: package 'kernlab' was built under R version 3.5.2
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.5.2
```

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

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

```
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 3.5.2
```

```
library(gridExtra)
```

```
## Warning: package 'gridExtra' was built under R version 3.5.2
```

```
library(grid)

# get air quality data
dataAQ = airquality
summary(dataAQ)
```

```
##      Ozone           Solar.R           Wind             Temp
##  Min.   :  1.00   Min.   :  7.0   Min.   : 1.700   Min.   :56.00
##  1st Qu.: 18.00   1st Qu.:115.8   1st Qu.: 7.400   1st Qu.:72.00
##  Median : 31.50   Median :205.0   Median : 9.700   Median :79.00
##  Mean   : 42.13   Mean   :185.9   Mean   : 9.958   Mean   :77.88
##  3rd Qu.: 63.25   3rd Qu.:258.8   3rd Qu.:11.500   3rd Qu.:85.00
##  Max.   :168.00   Max.   :334.0   Max.   :20.700   Max.   :97.00
##  NA's   :37       NA's   :7
##      Month            Day
##  Min.   :5.000   Min.   : 1.0
##  1st Qu.:6.000   1st Qu.: 8.0
##  Median :7.000   Median :16.0
##  Mean   :6.993   Mean   :15.8
##  3rd Qu.:8.000   3rd Qu.:23.0
##  Max.   :9.000   Max.   :31.0
##
```

```
str(dataAQ)
```

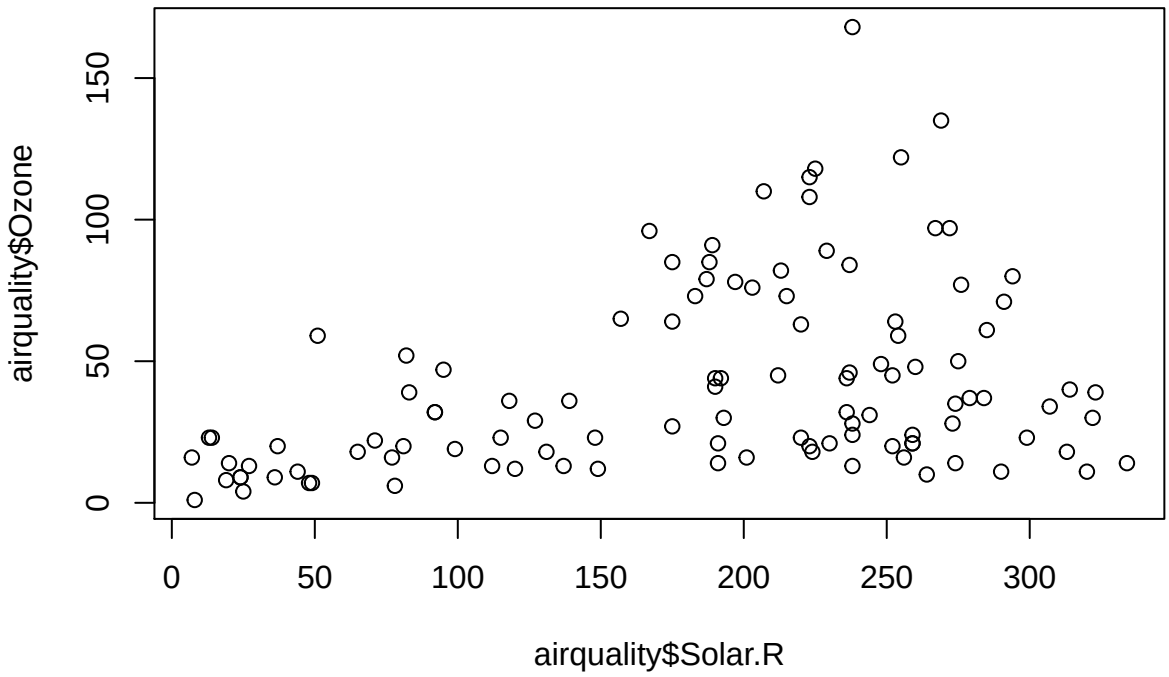
```
## 'data.frame':    153 obs. of  6 variables:
```

```
##  $ Ozone  : int  41 36 12 18 NA 28 23 19 8 NA ...
##  $ Solar.R: int  190 118 149 313 NA NA 299 99 19 194 ...
##  $ Wind   : num  7.4 8 12.6 11.5 14.3 14.9 8.6 13.8 20.1 8.6 ...
##  $ Temp   : int  67 72 74 62 56 66 65 59 61 69 ...
##  $ Month  : int  5 5 5 5 5 5 5 5 5 5 ...
##  $ Day    : int  1 2 3 4 5 6 7 8 9 10 ...
```

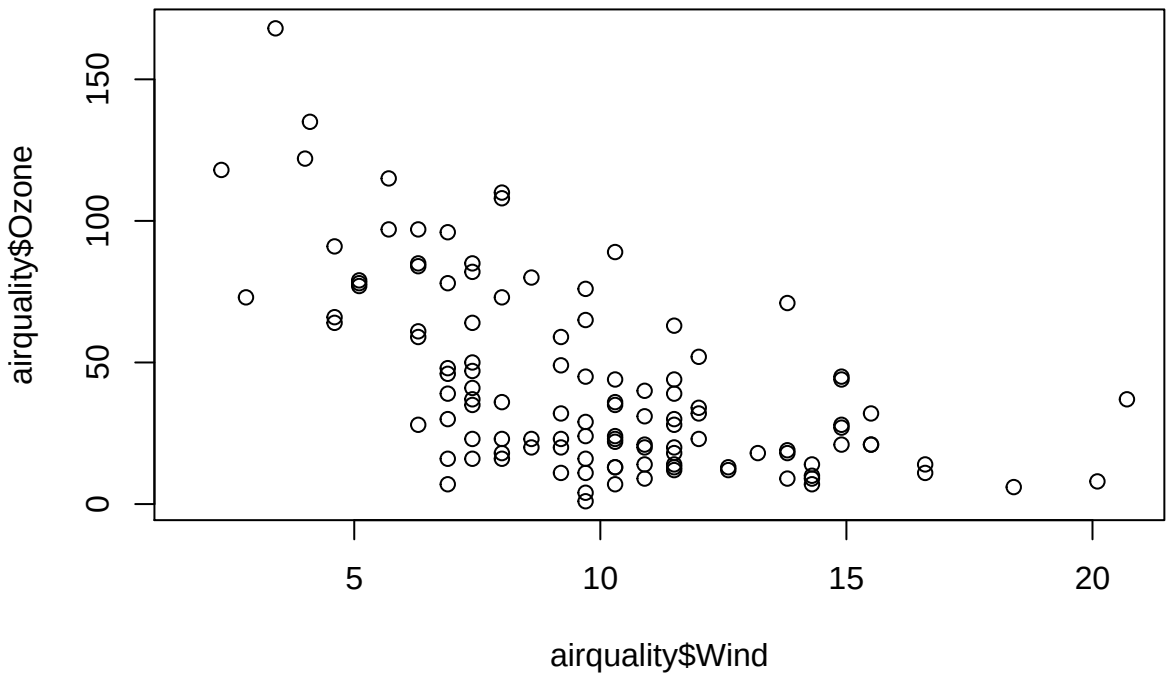
```r
# remove nas
dataAQ = na.omit(dataAQ)
sum(is.na(dataAQ) == TRUE)
```

```
## [1] 0
```

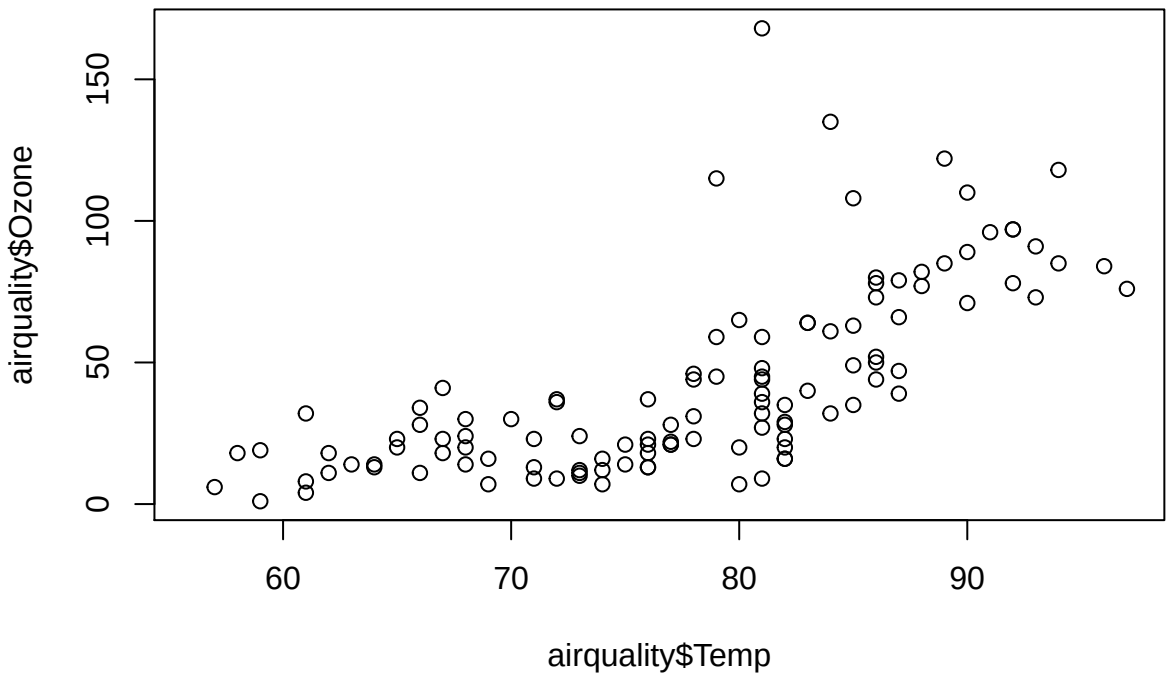
```r
# split trng and test data
trngData = dataAQ[1:77, ]
testData = dataAQ[78:111, ]

# plots to see what variables to start with
plot(airquality$Solar.R, airquality$Ozone)
```



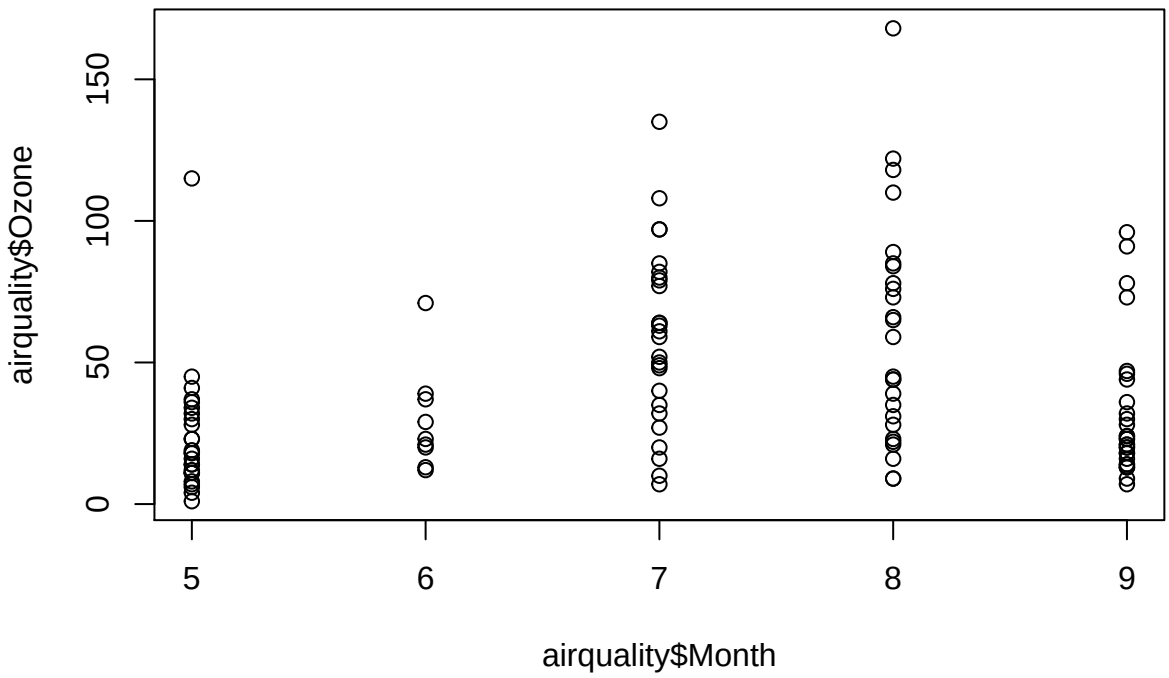

```r
plot(airquality$Wind, airquality$Ozone)
```




```
plot(airquality$Temp, airquality$Ozone)
```




```
plot(airquality$Month, airquality$Ozone)
```

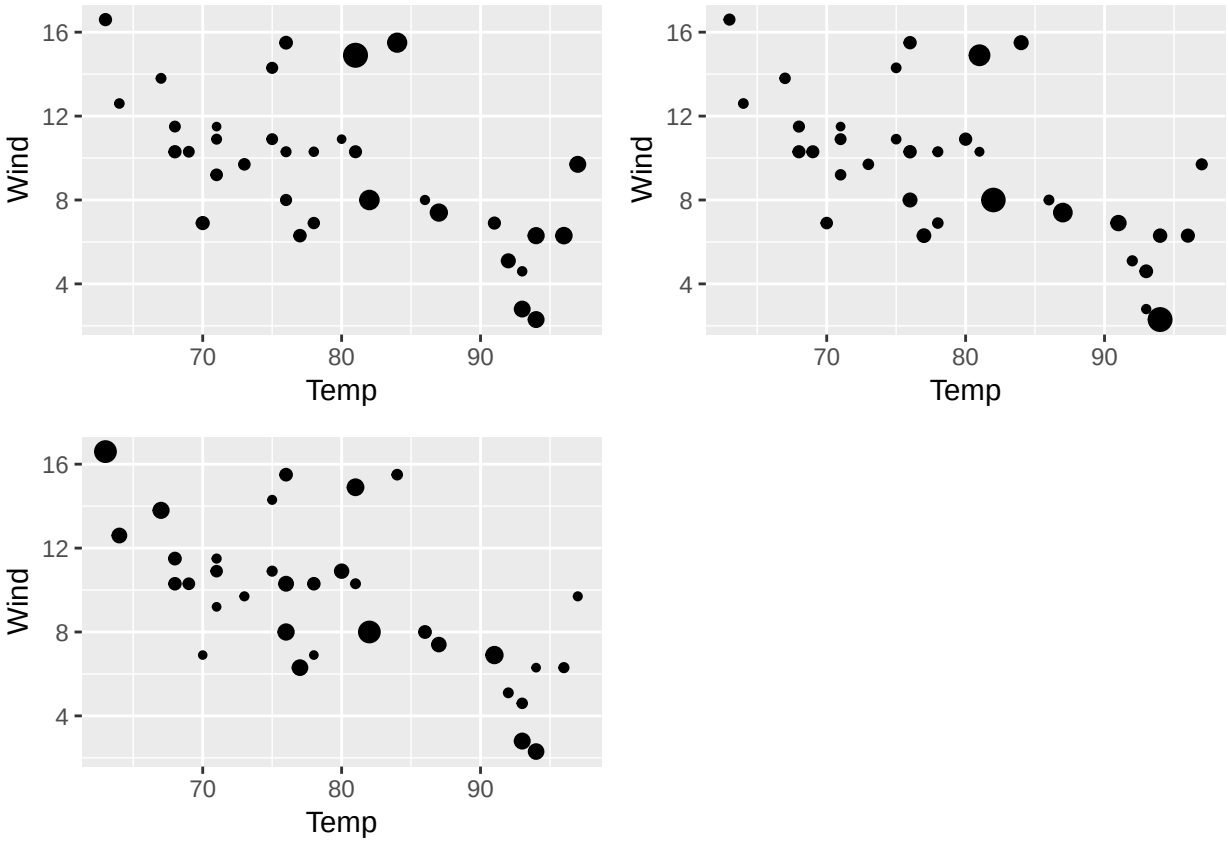



```
# ksvm model
model.ksvm = ksvm(Ozone ~ Wind + Temp + Month, data = trngData,
                  kernel = "rbfdot", kpar = "automatic", C = 5,
                  cross = 3, prob.model = TRUE)
model.ksvm
```

```
## 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 =  0.339222129231843
##
## Number of Support Vectors : 61
##
## Objective Function Value : -83.6021
## Training error : 0.206437
## Cross validation error : 436.6313
## Laplace distr. width : 41.4453
```

```
result.ksvm = predict(model.ksvm, testData[ , 3:5])

# transform for plotting
P.Ozone = testData$Ozone - result.ksvm
P.Ozone = scale(P.Ozone) # z score
P.Ozone = sqrt(P.Ozone*P.Ozone) + 1 # get rid of negatives and ad 1 for graphing
```

```r
resultsdf = cbind(testData$Wind, testData$Temp, P.Ozone)

result.ksvn.df = as.data.frame(resultsdf) # assign to new df
colnames(result.ksvn.df) = c("Wind", "Temp", "P.Ozone.ksvm")

k = ggplot(result.ksvn.df, aes(Temp, Wind)) + geom_point(size = P.Ozone) # plot

# svm model
model.svm = svm(Ozone ~ Wind + Temp + Month, data = trngData)
result.svm = predict(model.svm, testData[ , 3:5])

# transform and add to df for plot
P.Ozone.svm = testData$Ozone - result.svm
P.Ozone.svm = scale(P.Ozone.svm)
P.Ozone.svm = sqrt(P.Ozone.svm*P.Ozone.svm) + 1
result.ksvn.df$P.Ozone.svm = P.Ozone.svm

s = ggplot(result.ksvn.df, aes(Temp, Wind)) + geom_point(size = P.Ozone.svm) # plot

# lm model
model.lm = lm(Ozone ~ Wind + Temp + Month, data = trngData)
result.lm = predict(model.lm, testData[ , 3:5])

# transform and add to df for plot
P.Ozone.lm = testData$Ozone - result.lm
P.Ozone.lm = scale(P.Ozone.lm)
P.Ozone.lm = sqrt(P.Ozone.lm*P.Ozone.lm) + 1
result.ksvn.df$P.Ozone.lm = P.Ozone.lm

l = ggplot(result.ksvn.df, aes(Temp, Wind)) + geom_point(size = P.Ozone.lm)

# plot on same grid
grid.arrange(k, s, l, ncol = 2)
```




```
# good ozone variable for trng and test data
goodOzone = mean(trngData$Ozone)
trngData$GoodOzone = ifelse(trngData$Ozone >= goodOzone, 1, 0)

goodOzone = mean(testData$Ozone)
testData$GoodOzone = ifelse(testData$Ozone >= goodOzone, 1, 0)

# ksvm model
gOzoneM.ksvm = ksvm(GoodOzone ~ Wind + Temp + Month, data = trngData,
                    kernel = "rbfdot", kpar = "automatic",
                    C = 50, cross = 3, prob.model = TRUE, type = 'C-svc')
gOzoneM.ksvm
```

```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 50
##
## Gaussian Radial Basis kernel function.
##  Hyperparameter : sigma =  0.611298483044244
##
## Number of Support Vectors : 30
##
## Objective Function Value : -538.7725
## Training error : 0.025974
## Cross validation error : 0.154872
```

```
## Probability model included.
```

```
result.ksvm.GO = predict(gOzoneM.ksvm, testData[ , 3:5], type = 'response')
result.ksvm.GO
```

```
##  [1] 1 1 1 1 1 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0
```

```
# place in df for plotting
testData$gOzoneResult.k = result.ksvm.GO
testData$Correct.k = ifelse(testData$gOzoneResult.k == testData$GoodOzone, 1, 3)
k = ggplot(testData, aes(Temp, Wind)) +
  geom_point(shape = testData$gOzoneResult.k, size = testData$Correct.k, color = testData$GoodOzone+1)

# svm model
gOzoneM.svm = svm(GoodOzone ~ Wind + Temp + Month, data = trngData)
gOzoneM.svm
```

```
##
## Call:
## svm(formula = GoodOzone ~ Wind + Temp + Month, data = trngData)
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  radial
##        cost:  1
##       gamma:  0.3333333
##     epsilon:  0.1
##
##
## Number of Support Vectors:  48
```

```
# place in df for modeling
result.svm.GO = predict(gOzoneM.svm, testData[ , 3:5])
result.svm.GO
```

```
##         118         120         121         122         123         124
## 0.95417959 0.94216821 0.71475024 0.88037438 0.95221679 0.71542481
##         125         126         127         128         129         130
## 0.66721126 0.58381056 0.63666337 0.68927128 0.17937094 0.26514843
##         131         132         133         134         135         136
## 0.22299089 0.10696623 0.11794995 0.11922824 0.06423368 0.43597561
##         137         138         139         140         141         142
## 0.06331964 0.05510289 0.43336222 0.12491469 0.15809133 0.08061359
##         143         144         145         146         147         148
## 0.52157441 0.15341856 0.11191139 0.34452557 0.07539362 0.25892969
##         149         151         152         153
## 0.21385211 0.04905646 0.29131140 0.07554367
```

```
# get % for prediction
accuracy = .5
testData$gOzoneResult.s = ifelse(result.svm.GO > accuracy, 1, 0)
testData$Correct.s = ifelse(testData$gOzoneResult.s == testData$GoodOzone, 1, 3)
s = ggplot(testData, aes(Temp, Wind)) +
  geom_point(shape = testData$gOzoneResult.s, size = testData$Correct.s, color = testData$GoodOzone+1)

# naive bayes model
```

```r
gOzoneM.nb = naiveBayes(GoodOzone ~ Wind + Temp + Month, data = trngData)
gOzoneM.nb
```

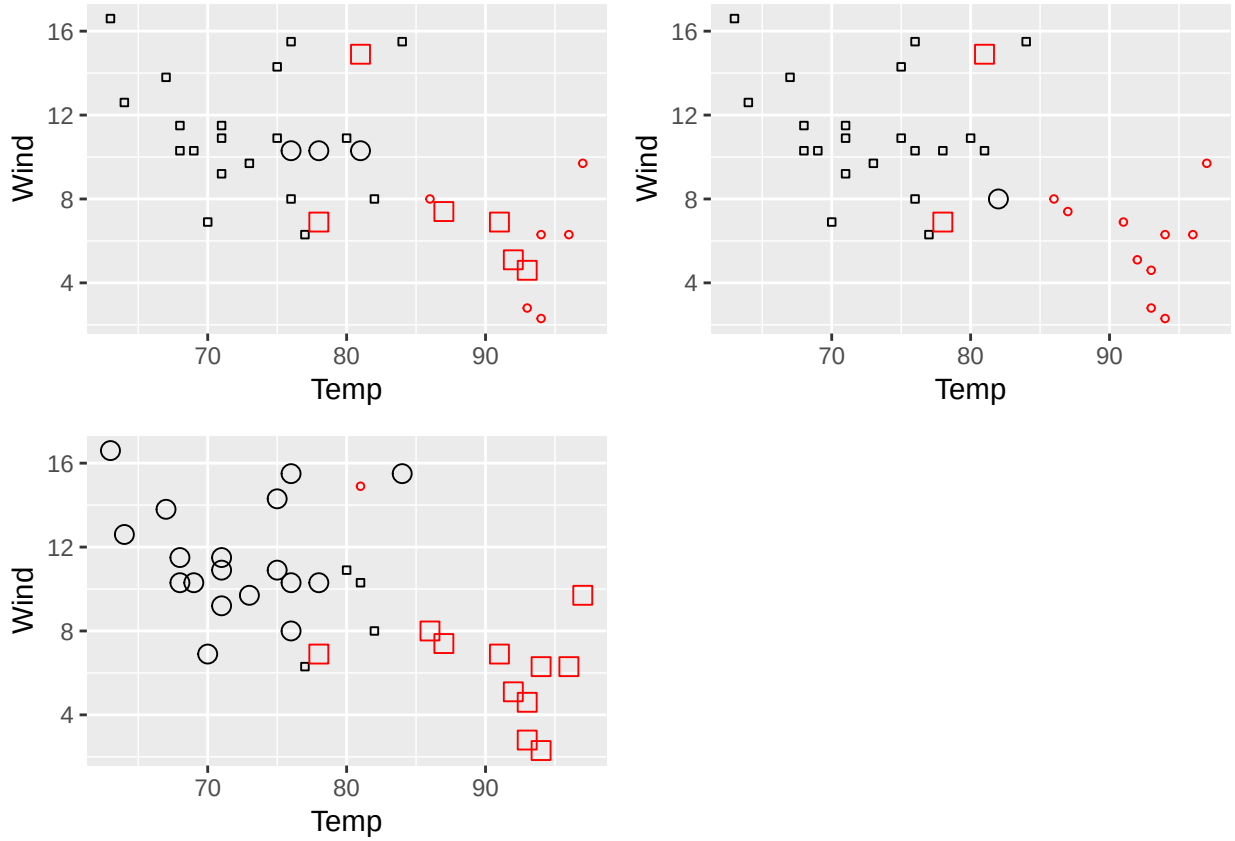
```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##         0         1
## 0.6103896 0.3896104
##
## Conditional probabilities:
##    Wind
## Y        [,1]     [,2]
##   0 11.461702 3.292188
##   1  7.993333 2.900646
##
##    Temp
## Y       [,1]     [,2]
##   0 72.14894 8.275103
##   1 84.90000 4.096677
##
##    Month
## Y       [,1]      [,2]
##   0 6.085106 1.1947310
##   1 7.133333 0.7760792
```

```r
result.nb.GO = predict(gOzoneM.nb, testData[ , 3:5], type = "raw")
result.nb.GO
```

```
##                0            1
##  [1,] 0.03258766 9.674123e-01
##  [2,] 0.16930076 8.306992e-01
##  [3,] 0.01167916 9.883208e-01
##  [4,] 0.04676231 9.532377e-01
##  [5,] 0.02789550 9.721045e-01
##  [6,] 0.03658208 9.634179e-01
##  [7,] 0.02555926 9.744407e-01
##  [8,] 0.01845215 9.815478e-01
##  [9,] 0.02650847 9.734915e-01
## [10,] 0.04249265 9.575073e-01
## [11,] 0.66800375 3.319963e-01
## [12,] 0.46429476 5.357052e-01
## [13,] 0.62867734 3.713227e-01
## [14,] 0.92072926 7.927074e-02
## [15,] 0.96602969 3.397031e-02
## [16,] 0.78154415 2.184558e-01
## [17,] 0.98114823 1.885177e-02
## [18,] 0.43364001 5.663600e-01
## [19,] 0.99521872 4.781278e-03
## [20,] 0.99624691 3.753091e-03
## [21,] 0.35062034 6.493797e-01
```

```
## [22,] 0.99996312 3.687756e-05
## [23,] 0.83336597 1.666340e-01
## [24,] 0.99956574 4.342564e-04
## [25,] 0.12935503 8.706450e-01
## [26,] 0.99999733 2.667708e-06
## [27,] 0.99099814 9.001859e-03
## [28,] 0.31847682 6.815232e-01
## [29,] 0.99890678 1.093220e-03
## [30,] 0.99999988 1.164515e-07
## [31,] 0.99180294 8.197058e-03
## [32,] 0.98123417 1.876583e-02
## [33,] 0.69060001 3.094000e-01
## [34,] 0.99973037 2.696300e-04
```

```r
# get % for prediction
testData$gOzoneResult.n = ifelse(result.nb.GO[,1] > accuracy, 1, 0)
testData$Correct.n = ifelse(testData$gOzoneResult.n == testData$GoodOzone, 1, 3)
n = ggplot(testData, aes(Temp, Wind)) +
  geom_point(shape = testData$gOzoneResult.n, size = testData$Correct.n, color = testData$GoodOzone+1)

# plot
grid.arrange(k, s, n, ncol = 2)
```




```r
# For me, SVM was the better model.  It had the least incorrect predictions.  Of course,
# that could be b/c I completely screwed this up.
```