705604096_stats101c_hw2

Jade Gregory

2023-10-18

Question 1

```
bcdat <- read.csv("BCNew.csv")
head(bcdat)
```

```
##   X diagnosis radius_mean texture_mean perimeter_mean area_mean smoothness_mean
## 1 1         B       13.11        22.54          87.02     529.4         0.10020
## 2 2         B       15.19        13.21          97.65     711.8         0.07963
## 3 3         B       11.25        14.78          71.38     390.0         0.08306
## 4 4         M       15.08        25.74          98.00     716.6         0.10240
## 5 5         M       18.22        18.87         118.70    1027.0         0.09746
## 6 6         B       11.06        14.83          70.31     378.2         0.07741
##   compactness_mean concavity_mean concave.points_mean symmetry_mean
## 1          0.14830      0.0870500            0.051020        0.1850
## 2          0.06934      0.0339300            0.026570        0.1721
## 3          0.04458      0.0009737            0.002941        0.1773
## 4          0.09769      0.1235000            0.065530        0.1647
## 5          0.11170      0.1130000            0.079500        0.1807
## 6          0.04768      0.0271200            0.007246        0.1535
##   fractal_dimension_mean
## 1                0.07310
## 2                0.05544
## 3                0.06081
## 4                0.06464
## 5                0.05664
## 6                0.06214
```

```
dim(bcdat)
```

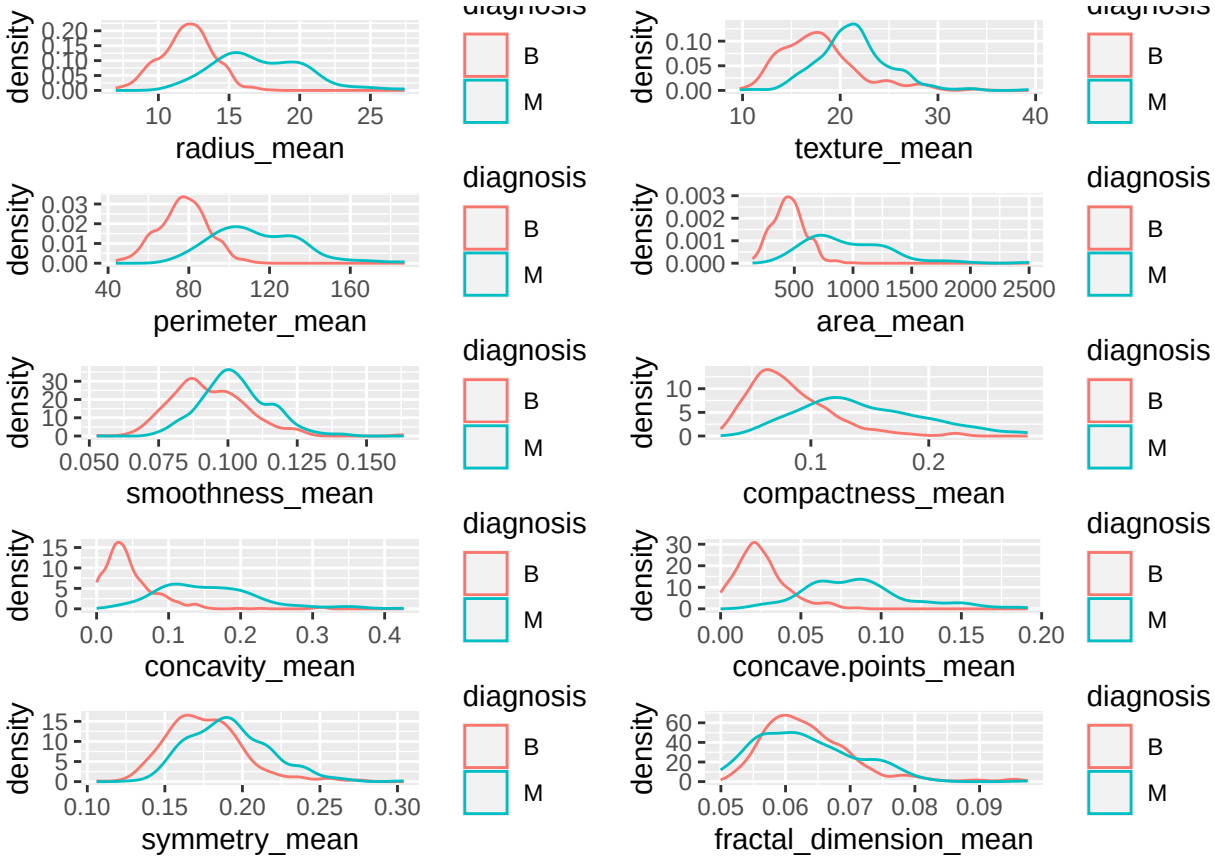
```
## [1] 850  12
```

a) The dimensions of the BCNew data set is 850 rows by 12 columns.

b)

```
g1 <- ggplot(bcdat, aes(radius_mean, color = diagnosis)) + geom_density()
g2 <- ggplot(bcdat, aes(texture_mean, color = diagnosis)) + geom_density()
g3 <- ggplot(bcdat, aes(perimeter_mean, color = diagnosis)) + geom_density()
g4 <- ggplot(bcdat, aes(area_mean, color = diagnosis)) + geom_density()
```

```
g5 <- ggplot(bcdat, aes(smoothness_mean, color = diagnosis)) + geom_density()
g6 <- ggplot(bcdat, aes(compactness_mean, color = diagnosis)) + geom_density()
g7 <- ggplot(bcdat, aes(concavity_mean, color = diagnosis)) + geom_density()
g8 <- ggplot(bcdat, aes(concave.points_mean, color = diagnosis)) + geom_density()
g9 <- ggplot(bcdat, aes(symmetry_mean, color = diagnosis)) + geom_density()
g10 <- ggplot(bcdat, aes(fractal_dimension_mean, color = diagnosis)) + geom_density()
grid.arrange(g1,g2,g3,g4,g5,g6,g7,g8,g9,g10, nrow = 5)
```



Based off of the density graphs, area_mean, concavity_mean, and concave.points_mean are the best three predictors since they have the least amount of overlap between the diagnosis types.

c)

```
library(class)
```

```
set.seed(113355)
test.i <- sample(1:850, 250, replace = F)
X.mat <- bcdat[,c(6, 9, 10)]
head(X.mat)
```

```
##   area_mean concavity_mean concave.points_mean
## 1     529.4      0.0870500            0.051020
## 2     711.8      0.0339300            0.026570
## 3     390.0      0.0009737            0.002941
```

```
## 4     716.6      0.1235000            0.065530
## 5    1027.0      0.1130000            0.079500
## 6     378.2      0.0271200            0.007246
```

```
Xtest <- X.mat[test.i,]
Xtrain <- X.mat[-test.i,]
Ytest <- bcdat$diagnosis[test.i]
Ytrain <- bcdat$diagnosis[-test.i]
```

```
#k = 1
out1 <- knn(Xtrain, Xtest, Ytrain, k = 1)
mean(out1 == Ytest)
```

```
## [1] 0.924
```

```
#k = 3
out3 <- knn(Xtrain, Xtest, Ytrain, k = 3)
mean(out3 == Ytest)
```

```
## [1] 0.896
```

```
#k = 5
out5 <- knn(Xtrain, Xtest, Ytrain, k = 5)
mean(out5 == Ytest)
```

```
## [1] 0.892
```

```
# k = 7
out7 <- knn(Xtrain, Xtest, Ytrain, k = 7)
mean(out7 == Ytest)
```

```
## [1] 0.888
```

```
# k = 9
out9 <- knn(Xtrain, Xtest, Ytrain, k = 9)
mean(out9 == Ytest)
```

```
## [1] 0.884
```

```
# k = 11
out11 <- knn(Xtrain, Xtest, Ytrain, k = 11)
mean(out11 == Ytest)
```

```
## [1] 0.884
```

d)

```r
mean(out1 != Ytest)
```

```
## [1] 0.076
```

```r
mean(out3 != Ytest)
```

```
## [1] 0.104
```

```r
mean(out5 != Ytest)
```

```
## [1] 0.108
```

```r
mean(out7 != Ytest)
```

```
## [1] 0.112
```

```r
mean(out9 != Ytest)
```

```
## [1] 0.116
```

```r
mean(out11 != Ytest)
```

```
## [1] 0.116
```

The best k is k = 1 because it has the lowest misclassification rate.

e)

```r
set.seed(1133355)
stest.i <- sample(1:850, 250, replace = F)
X.mat.scale <- scale(bcdat[,c(6, 9, 10)])
head(X.mat.scale)
```

```
##       area_mean concavity_mean concave.points_mean
## [1,] -0.3255839   -0.009628358          0.07543829
## [2,]  0.2004500   -0.709071866         -0.57323676
## [3,] -0.7276076   -1.143015203         -1.20013012
## [4,]  0.2142930    0.470317348          0.46039842
## [5,]  1.1094734    0.332061383          0.83103197
## [6,] -0.7616383   -0.798740734         -1.08591556
```

```r
sXtest <- X.mat.scale[stest.i,]
sXtrain <- X.mat.scale[-stest.i,]
sYtest <- bcdat$diagnosis[stest.i]
sYtrain <- bcdat$diagnosis[-stest.i]
```

```r
sout1 <- knn(sXtrain, sXtest, sYtrain, k = 1)
mean(sout1 == sYtest)
```

```
## [1] 0.924
```

```r
sout3 <- knn(sXtrain, sXtest, sYtrain, k = 3)
mean(sout3 == sYtest)
```

```
## [1] 0.936
```

```r
sout5 <- knn(sXtrain, sXtest, sYtrain, k = 5)
mean(sout5 == sYtest)
```

```
## [1] 0.924
```

```r
sout7 <- knn(sXtrain, sXtest, sYtrain, k = 7)
mean(sout7 == sYtest)
```

```
## [1] 0.912
```

```r
sout9 <- knn(sXtrain, sXtest, sYtrain, k = 9)
mean(sout9 == sYtest)
```

```
## [1] 0.904
```

```r
sout11 <- knn(sXtrain, sXtest, sYtrain, k = 11)
mean(sout11 == sYtest)
```

```
## [1] 0.908
```

f)

```r
mean(sout1 != sYtest)
```

```
## [1] 0.076
```

```r
mean(sout3 != sYtest)
```

```
## [1] 0.064
```

```r
mean(sout5 != sYtest)
```

```
## [1] 0.076
```

```r
mean(sout7 != sYtest)
```

```
## [1] 0.088
```

```r
mean(sout9 != sYtest)
```

```
## [1] 0.096
```

```r
mean(sout11 != sYtest)
```

```
## [1] 0.092
```

The best k is k = 3 because it has the lowest misclassification rate.

Question 2

```r
#non scaled
set.seed(113355)
ntest.i <- sample(1:850, 250, replace = F)
nX.mat <- bcdat[,c(3:12)]
head(nX.mat)
```

```
##   radius_mean texture_mean perimeter_mean area_mean smoothness_mean
## 1       13.11        22.54          87.02     529.4         0.10020
## 2       15.19        13.21          97.65     711.8         0.07963
## 3       11.25        14.78          71.38     390.0         0.08306
## 4       15.08        25.74          98.00     716.6         0.10240
## 5       18.22        18.87         118.70    1027.0         0.09746
## 6       11.06        14.83          70.31     378.2         0.07741
##   compactness_mean concavity_mean concave.points_mean symmetry_mean
## 1          0.14830      0.0870500            0.051020        0.1850
## 2          0.06934      0.0339300            0.026570        0.1721
## 3          0.04458      0.0009737            0.002941        0.1773
## 4          0.09769      0.1235000            0.065530        0.1647
## 5          0.11170      0.1130000            0.079500        0.1807
## 6          0.04768      0.0271200            0.007246        0.1535
##   fractal_dimension_mean
## 1                0.07310
## 2                0.05544
## 3                0.06081
## 4                0.06464
## 5                0.05664
## 6                0.06214
```

```r
nXtest <- nX.mat[ntest.i,]
nXtrain <- nX.mat[-ntest.i,]
nYtest <- bcdat$diagnosis[ntest.i]
nYtrain <- bcdat$diagnosis[-ntest.i]
```

```r
nout1 <- knn(nXtrain, nXtest, nYtrain, k = 1)
mean(nout1 == nYtest)
```

```
## [1] 0.936
```

```r
nout3 <- knn(nXtrain, nXtest, nYtrain, k = 3)
mean(nout3 == nYtest)
```

```
## [1] 0.924
```

```r
nout5 <- knn(nXtrain, nXtest, nYtrain, k = 5)
mean(nout5 == nYtest)
```

```
## [1] 0.888
```

```r
nout7 <- knn(nXtrain, nXtest, nYtrain, k = 7)
mean(nout7 == nYtest)
```

```
## [1] 0.896
```

```r
nout9 <- knn(nXtrain, nXtest, nYtrain, k = 9)
mean(nout9 == nYtest)
```

```
## [1] 0.904
```

```r
nout11 <- knn(nXtrain, nXtest, nYtrain, k = 11)
mean(nout11 == nYtest)
```

```
## [1] 0.892
```

```r
mean(nout1 != nYtest)
```

```
## [1] 0.064
```

```r
mean(nout3 != nYtest)
```

```
## [1] 0.076
```

```r
mean(nout5 != nYtest)
```

```
## [1] 0.112
```

```r
mean(nout7 != nYtest)
```

```
## [1] 0.104
```

```r
mean(nout9 != nYtest)
```

```
## [1] 0.096
```

```r
mean(nout11 != nYtest)
```

```
## [1] 0.108
```

The best k is k = 1 because it has the lowest misclassification rate.

```r
#scaled
set.seed(113355)
ptest.i <- sample(1:850, 250, replace = F)
pX.mat.scale <- scale(bcdat[,c(6, 9, 10)])
head(pX.mat.scale)
```

```
##       area_mean concavity_mean concave.points_mean
## [1,] -0.3255839   -0.009628358          0.07543829
## [2,]  0.2004500   -0.709071866         -0.57323676
## [3,] -0.7276076   -1.143015203         -1.20013012
## [4,]  0.2142930    0.470317348          0.46039842
## [5,]  1.1094734    0.332061383          0.83103197
## [6,] -0.7616383   -0.798740734         -1.08591556
```

```r
pXtest <- pX.mat.scale[ptest.i,]
pXtrain <- pX.mat.scale[-ptest.i,]
pYtest <- bcdat$diagnosis[ptest.i]
pYtrain <- bcdat$diagnosis[-ptest.i]
```

```r
pout1 <- knn(pXtrain, pXtest, pYtrain, k = 1)
mean(pout1 == pYtest)
```

```
## [1] 0.972
```

```r
pout3 <- knn(pXtrain, pXtest, pYtrain, k = 3)
mean(pout3 == pYtest)
```

```
## [1] 0.952
```

```r
pout5 <- knn(pXtrain, pXtest, pYtrain, k = 5)
mean(pout5 == pYtest)
```

```
## [1] 0.94
```

```r
pout7 <- knn(pXtrain, pXtest, pYtrain, k = 7)
mean(pout7 == pYtest)
```

```
## [1] 0.924
```

```r
pout9 <- knn(pXtrain, pXtest, pYtrain, k = 9)
mean(pout9 == pYtest)
```

```
## [1] 0.924
```

```r
pout11 <- knn(pXtrain, pXtest, pYtrain, k = 11)
mean(pout11 == pYtest)
```

```
## [1] 0.916
```

```r
mean(pout1 != pYtest)
```

```
## [1] 0.028
```

```r
mean(pout3 != pYtest)
```

```
## [1] 0.048
```

```r
mean(pout5 != pYtest)
```

```
## [1] 0.06
```

```r
mean(pout7 != pYtest)
```

```
## [1] 0.076
```

```r
mean(pout9 != pYtest)
```

```
## [1] 0.076
```

```r
mean(pout11 != pYtest)
```

```
## [1] 0.084
```

The best k is k = 1 because it has the lowest misclassification rate.

For the unscaled model of all variables, I get the same misclassification rates as I did in part 1. I believe that our scaled model provides more accurate statistics of our data as it helps to balance the impact of our predictor variables onto our dependent variable, which can improve the quality of models.

Question 3

a)

```r
testdat <- bcdat[test.i,]
traindat <- bcdat[-test.i,]
```

```r
traindat$diagnosis <- as.factor(traindat$diagnosis)
lr.model1 <- glm(diagnosis ~ . - X - diagnosis, data = traindat, family = binomial())
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
summary(lr.model1)
```

```
##
## Call:
## glm(formula = diagnosis ~ . - X - diagnosis, family = binomial(),
##     data = traindat)
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)            -10.87106   12.23768  -0.888 0.374365
## radius_mean              0.75211    3.53664   0.213 0.831591
## texture_mean             0.49477    0.08112   6.099 1.07e-09 ***
## perimeter_mean          -0.61504    0.47995  -1.281 0.200027
## area_mean                0.05141    0.01574   3.266 0.001093 **
## smoothness_mean         96.06579   29.00032   3.313 0.000924 ***
## compactness_mean         8.36864   18.88190   0.443 0.657614
## concavity_mean          21.75315    8.54036   2.547 0.010862 *
## concave.points_mean     65.87544   28.99394   2.272 0.023084 *
## symmetry_mean           22.45558   10.68940   2.101 0.035664 *
## fractal_dimension_mean -77.51313   91.74116  -0.845 0.398160
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 800.68  on 599  degrees of freedom
## Residual deviance: 148.72  on 589  degrees of freedom
## AIC: 170.72
##
## Number of Fisher Scoring iterations: 9
```

```r
# training data confusion matrix
trainprob <- predict(lr.model1, data = traindat, type = "response")
predlogit <- rep('M', length(trainprob))
predlogit[trainprob <= 0.5] <- 'B'
table(predlogit, traindat$diagnosis)
```

```
##
## predlogit   B   M
##         B 352  14
##         M  16 218
```

```r
mean(predlogit == traindat$diagnosis)
```

```
## [1] 0.95
```

```r
mean(predlogit != traindat$diagnosis)
```

```
## [1] 0.05
```

The misclassification rate is 5%

```r
# testing data confusion matrix
testprob <- predict(lr.model1, data = traindat, newdata = testdat, type = "response")
predlogit2 <- rep("M", length(testprob))
predlogit2[testprob <= 0.5] <- "B"
table(predlogit2, testdat$diagnosis)
```

```
##
## predlogit2   B   M
##          B 151   8
##          M   6  85
```

```r
mean(predlogit2 == testdat$diagnosis)
```

```
## [1] 0.944
```

```r
mean(predlogit2 != testdat$diagnosis)
```

```
## [1] 0.056
```

The misclassification rate is 5.6%

```r
scaleXtest <- scale(testdat[c(3:12)])
scaleYtest <- testdat$diagnosis
scaleXtrain <- scale(traindat[c(3:12)])
scaleYtrain <- traindat$diagnosis
sctest <- data.frame(scaleYtest, scaleXtest)
sctrain <- data.frame(scaleXtrain, scaleYtrain)
head(sctest)
```

```
##     scaleYtest radius_mean texture_mean perimeter_mean  area_mean
## 466          M   0.3951596    0.2373059      0.4760827  0.2863063
## 120          M   1.0074155    1.9406585      0.9553683  1.0171100
## 61           B  -0.6990204    0.5025821     -0.6887131 -0.7001001
## 85           M   1.5317732    0.2722106      1.3818885  1.5721055
## 557          B  -0.5535339   -1.8383642     -0.5862603 -0.5664601
## 426          B  -1.3212785    0.6677980     -1.3069475 -1.1305380
##     smoothness_mean compactness_mean concavity_mean concave.points_mean
## 466       0.4542047        0.9714590     1.21904175           1.1708298
## 120       0.1144027        0.1282559     0.24041403           0.4978501
## 61        0.8893169       -0.2025537    -0.53330911          -0.6872040
## 85       -0.5589852       -0.8645538    -0.04907903           0.3318635
## 557      -0.5942086       -0.9867516    -0.81362311          -0.8105648
## 426      -1.2178695       -0.8323865    -0.65419011          -1.1727995
##     symmetry_mean fractal_dimension_mean
## 466    0.17718830           -0.08644780
## 120   -0.05213858           -0.09035922
## 61     0.12910363            0.85359897
## 85    -0.95095198           -1.76575464
## 557   -0.76231213           -1.05517837
## 426   -1.97182646            0.48722846
```

```r
myglm <- glm(scaleYtrain ~ scaleXtrain, data = sctrain, family = binomial())
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
summary(myglm)
```

```
##
## Call:
## glm(formula = scaleYtrain ~ scaleXtrain, family = binomial(),
##     data = sctrain)
##
## Coefficients:
##                                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)                         0.9670     0.5298   1.825 0.067948 .
## scaleXtrainradius_mean              2.6960    12.6776   0.213 0.831591
## scaleXtraintexture_mean             2.0635     0.3383   6.099 1.07e-09 ***
## scaleXtrainperimeter_mean         -15.1550    11.8263  -1.281 0.200027
## scaleXtrainarea_mean               18.5757     5.6884   3.266 0.001093 **
## scaleXtrainsmoothness_mean          1.3333     0.4025   3.313 0.000924 ***
## scaleXtraincompactness_mean         0.4364     0.9847   0.443 0.657614
## scaleXtrainconcavity_mean           1.6963     0.6660   2.547 0.010862 *
## scaleXtrainconcave.points_mean      2.5434     1.1194   2.272 0.023084 *
## scaleXtrainsymmetry_mean            0.5917     0.2817   2.101 0.035664 *
## scaleXtrainfractal_dimension_mean  -0.5578     0.6602  -0.845 0.398160
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 800.68  on 599  degrees of freedom
## Residual deviance: 148.72  on 589  degrees of freedom
## AIC: 170.72
##
## Number of Fisher Scoring iterations: 9
```

```r
# training data confusion matrix
trainprob3 <- predict(myglm, data = sctrain, type = "response")
predlogit3 <- rep('M', length(trainprob3))
predlogit3[trainprob3 <= 0.5] <- 'B'
table(predlogit3, sctrain$scaleYtrain)
```

```
##
## predlogit3   B   M
##          B 352  14
##          M  16 218
```

```r
mean(predlogit3 == sctrain$scaleYtrain)
```

```
## [1] 0.95
```

```r
mean(predlogit3 != sctrain$scaleYtrain)
```

```
## [1] 0.05
```

The misclassification rate is 5%

```r
# testing data confusion matrix
testprob4 <- predict(myglm, data = sctrain, newdata = sctest, type = "response")
```

```
## Warning: 'newdata' had 250 rows but variables found have 600 rows
```

```r
predlogit4 <- rep('M', length(testprob4))
predlogit4[testprob4 <= 0.5] <- 'B'
table(predlogit4, sctest$scaleYtest)
```

```
## Error in table(predlogit4, sctest$scaleYtest): all arguments must have the same length
```

```r
mean(predlogit4 == sctest$scaleYtest)
```

```
## Warning in predlogit4 == sctest$scaleYtest: longer object length is not a
## multiple of shorter object length
```

```
## [1] 0.5266667
```

```r
mean(predlogit4 != sctest$scaleYtest)
```

```
## Warning in predlogit4 != sctest$scaleYtest: longer object length is not a
## multiple of shorter object length
```

```
## [1] 0.4733333
```

The misclassification rate is 47.3%

c) Our confusion matrices for the testing data and training data for the scaled and unscaled data sets are listed above. Our misclassification rate for the unscaled training data is 5%. Our misclassification rate for the unscaled testing data is 5.6%. Our misclassification rate for the scaled training data is 5% as well. Our misclassification rate for the scaled training data is 47.3%. We can see that the misclassification rates for the unscaled and scaled training data are the same.

Question 4

We can see that our unscaled glm model of all predictors has the lowest misclassification rate among all of the glm and knn models we made. Therefore, I would argue that it is our "hero" model in this instance.

Question 5

```r
boston <- read.csv("boston.csv")
dim(boston)
```

```
## [1] 506  14
```

```r
.7*506
```

```
## [1] 354.2
```

```r
506 - 354
```

```
## [1] 152
```

```r
crim_med <- median(boston$crim)
for(i in 1:length(boston$crim)){
if(boston$crim[i] >= crim_med){
  boston$med_crim[i] <- 1
} else {
  boston$med_crim[i] <- 0
}
}
```

```r
crim_med
```

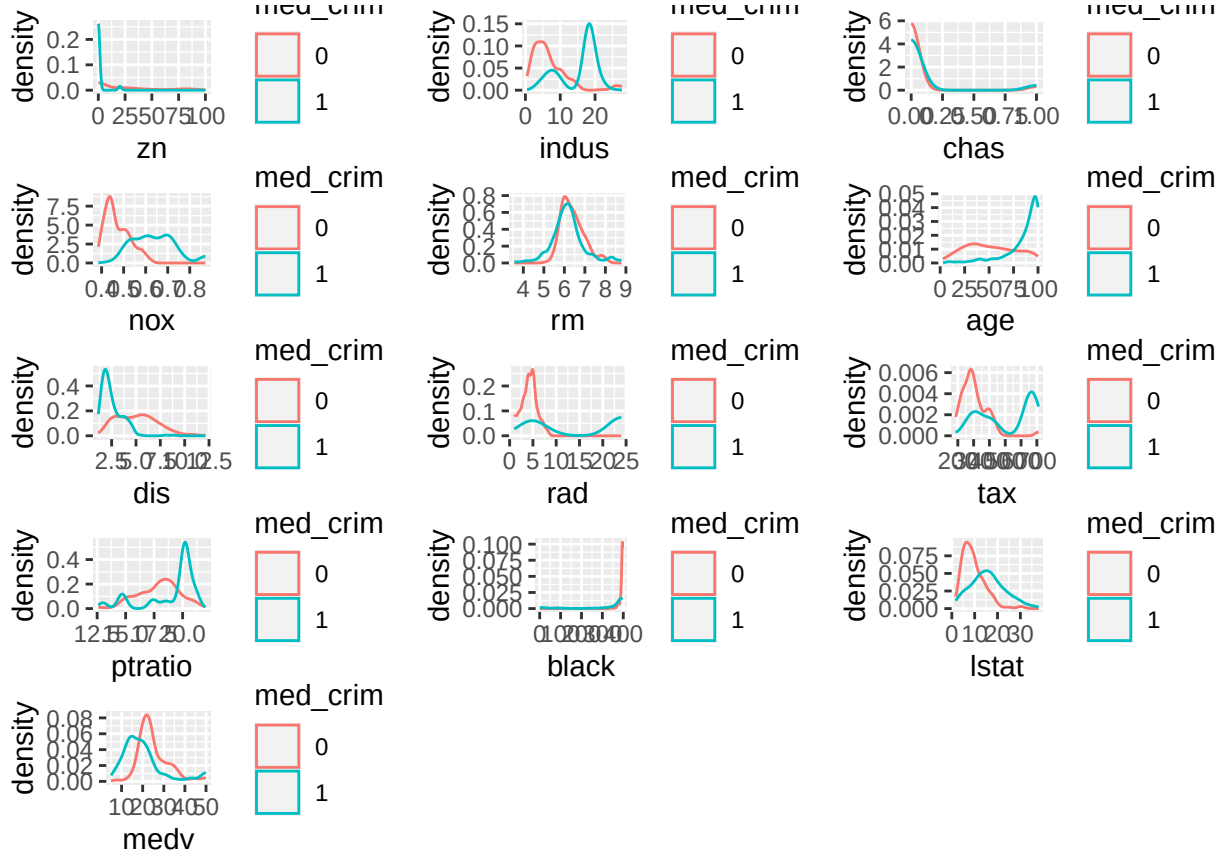
```
## [1] 0.25651
```

```r
boston$med_crim <- as.factor(boston$med_crim)
head(boston)
```

```
##      crim   zn indus chas   nox    rm  age    dis rad tax ptratio  black lstat
## 1 0.00632 18.0  2.31    0 0.538 6.575 65.2 4.0900   1 296    15.3 396.90  4.98
## 2 0.02731  0.0  7.07    0 0.469 6.421 78.9 4.9671   2 242    17.8 396.90  9.14
## 3 0.06905  0.0  2.18    0 0.458 7.147 54.2 6.0622   3 222    18.7 396.90  5.33
## 4 0.02985  0.0  2.18    0 0.458 6.430 58.7 6.0622   3 222    18.7 394.12  5.21
## 5 0.08829 12.5  7.87    0 0.524 6.012 66.6 5.5605   5 311    15.2 395.60 12.43
## 6 0.14455 12.5  7.87    0 0.524 6.172 96.1 5.9505   5 311    15.2 396.90 19.15
##   medv med_crim
## 1 24.0        0
## 2 21.6        0
## 3 36.2        0
## 4 28.7        0
## 5 22.9        0
## 6 27.1        0
```

```r
gg1 <- ggplot(boston, aes(zn, color = med_crim)) + geom_density()
gg2 <- ggplot(boston, aes(indus, color = med_crim)) + geom_density()
gg3 <- ggplot(boston, aes(chas, color = med_crim)) + geom_density()
gg4 <- ggplot(boston, aes(nox, color = med_crim)) + geom_density()
gg5 <- ggplot(boston, aes(rm, color = med_crim)) + geom_density()
```

```
gg6 <- ggplot(boston, aes(age, color = med_crim)) + geom_density()
gg7 <- ggplot(boston, aes(dis, color = med_crim)) + geom_density()
gg8 <- ggplot(boston, aes(rad, color = med_crim)) + geom_density()
gg9 <- ggplot(boston, aes(tax, color = med_crim)) + geom_density()
gg10 <- ggplot(boston, aes(ptratio, color = med_crim)) + geom_density()
gg11 <- ggplot(boston, aes(black, color = med_crim)) + geom_density()
gg12 <- ggplot(boston, aes(lstat, color = med_crim)) + geom_density()
gg13 <- ggplot(boston, aes(medv, color = med_crim)) + geom_density()
grid.arrange(gg1,gg2,gg3,gg4,gg5,gg6,gg7,gg8,gg9,gg10,gg11,gg12,gg13, nrow = 5)
```



From our density graphs, the top five predictors in order of best predictor to worst are the variables zn, black, rad, age, and dis.

```
# using training data to fit knn models
set.seed(113355)
ror <- sample(1:506, 354, replace = F)
# best 3 predictors
best3bos <- boston[,c(2, 12, 9)]
head(best3bos)
```

```
##      zn  black rad
## 1 18.0 396.90   1
## 2  0.0 396.90   2
## 3  0.0 396.90   3
## 4  0.0 394.12   3
```

```
## 5 12.5 395.60   5
## 6 12.5 396.90   5
```

```
# choose k as sqrt(n)
myk <- 23
best3Xtest <- best3bos[ror,]
best3Xtrain <- best3bos[-ror,]
best3Ytest <- boston$med_crim[ror]
best3Ytrain <- boston$med_crim[-ror]
```

```
#k = 23
bos3out <- knn(best3Xtrain, best3Xtest, best3Ytrain, k = myk)
mean(bos3out == best3Ytest)
```

```
## [1] 0.8022599
```

```
mean(bos3out != best3Ytest)
```

```
## [1] 0.1977401
```

The knn model for our best 3 predictors has a misclassification rate of 19.774%

```
# using training data to fit knn models
set.seed(113355)
ror <- sample(1:506, 354, replace = F)
#best 4 predictors
best4bos <- boston[,c(2, 12, 9, 7)]
head(best4bos)
```

```
##     zn  black rad  age
## 1 18.0 396.90   1 65.2
## 2  0.0 396.90   2 78.9
## 3  0.0 396.90   3 54.2
## 4  0.0 394.12   3 58.7
## 5 12.5 395.60   5 66.6
## 6 12.5 396.90   5 96.1
```

```
best4Xtest <- best4bos[ror,]
best4Xtrain <- best4bos[-ror,]
best4Ytest <- boston$med_crim[ror]
best4Ytrain <- boston$med_crim[-ror]
```

```
# k = 23
bos4out <- knn(best4Xtrain, best4Xtest, best4Ytrain, k = myk)
mean(bos4out == best4Ytest)
```

```
## [1] 0.7853107
```

```
mean(bos4out != best4Ytest)
```

```
## [1] 0.2146893
```

The knn model for our best 4 predictors has a misclassification rate of 21.4%

```
# using training data to fit knn models
set.seed(113355)
ror <- sample(1:506, 354, replace = F)
#best 5 predictors
best5bos <- boston[,c(2, 12, 9, 7, 8)]
head(best5bos)
```

```
##     zn  black rad  age    dis
## 1 18.0 396.90   1 65.2 4.0900
## 2  0.0 396.90   2 78.9 4.9671
## 3  0.0 396.90   3 54.2 6.0622
## 4  0.0 394.12   3 58.7 6.0622
## 5 12.5 395.60   5 66.6 5.5605
## 6 12.5 396.90   5 96.1 5.9505
```

```
best5Xtest <- best5bos[ror,]
best5Xtrain <- best5bos[-ror,]
best5Ytest <- boston$med_crim[ror]
best5Ytrain <- boston$med_crim[-ror]
```

```
# k = 23
bos5out <- knn(best5Xtrain, best5Xtest, best5Ytrain, k = myk)
mean(bos5out == best5Ytest)
```

```
## [1] 0.7824859
```

```
mean(bos5out != best5Ytest)
```

```
## [1] 0.2175141
```

The knn model for our best 5 predictors has a misclassification rate of 21.7%

```
best3 <- data.frame(best3Xtrain, best3Ytrain)
head(best3)
```

```
##      zn  black rad best3Ytrain
## 4   0.0 394.12   3           0
## 9  12.5 396.90   5           0
## 10 12.5 390.50   5           0
## 13  0.0 395.62   4           1
## 14  0.0 386.85   4           1
## 18  0.0 396.90   4           1
```

```
# glm for best 3 predictors
bosglm3 <- glm(best3Ytrain ~ zn + black + rad, data = best3, family = binomial())
summary(bosglm3)
```

```
##
## Call:
## glm(formula = best3Ytrain ~ zn + black + rad, family = binomial(),
##     data = best3)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.313352   2.380780   0.132  0.89529
## zn          -0.043139   0.019938  -2.164  0.03049 *
## black       -0.006814   0.005950  -1.145  0.25214
## rad          0.374370   0.135816   2.756  0.00584 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 210.61  on 151  degrees of freedom
## Residual deviance: 113.82  on 148  degrees of freedom
## AIC: 121.82
##
## Number of Fisher Scoring iterations: 8
```

```
bos3prob <- predict(bosglm3, data = best3, type = "response")
bos3pred <- rep("1", length(bos3prob))
bos3pred[bos3prob <= 0.5] <- "0"
table(bos3pred, best3$best3Ytrain)
```

```
##
## bos3pred  0  1
##        0 77 25
##        1  1 49
```

```
mean(bos3pred == best3$best3Ytrain)
```

```
## [1] 0.8289474
```

```
mean(bos3pred != best3$best3Ytrain)
```

```
## [1] 0.1710526
```

The glm for the best 3 predictors has a misclassification rate of 17.1%

```
best4 <- data.frame(best4Xtrain, best4Ytrain)
head(best4)
```

```
##      zn  black rad  age best4Ytrain
## 4   0.0 394.12   3 58.7           0
## 9  12.5 396.90   5 82.9           0
## 10 12.5 390.50   5 39.0           0
## 13  0.0 395.62   4 56.5           1
## 14  0.0 386.85   4 29.3           1
## 18  0.0 396.90   4 91.7           1
```

```r
# glm for best 4 predictors
bosglm4 <- glm(best4Ytrain ~ zn + black + rad + age, data = best4, family = binomial())
summary(bosglm4)
```

```
##
## Call:
## glm(formula = best4Ytrain ~ zn + black + rad + age, family = binomial(),
##     data = best4)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.999663   2.571331  -1.167  0.24338
## zn          -0.023529   0.021402  -1.099  0.27161
## black       -0.003660   0.005634  -0.650  0.51585
## rad          0.377240   0.143963   2.620  0.00878 **
## age          0.029064   0.011051   2.630  0.00854 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 210.61  on 151  degrees of freedom
## Residual deviance: 105.72  on 147  degrees of freedom
## AIC: 115.72
##
## Number of Fisher Scoring iterations: 8
```

```r
bos4prob <- predict(bosglm4, data = best4, type = "response")
bos4pred <- rep("1", length(bos4prob))
bos4pred[bos4prob <= 0.5] <- "0"
table(bos4pred, best4$best4Ytrain)
```

```
##
## bos4pred  0  1
##        0 70 21
##        1  8 53
```

```r
mean(bos4pred == best4$best4Ytrain)
```

```
## [1] 0.8092105
```

```r
mean(bos4pred != best4$best4Ytrain)
```

```
## [1] 0.1907895
```

The glm for our best 4 predictors has a misclassification rate of 19.07%

```r
best5 <- data.frame(best5Xtrain, best5Ytrain)
head(best5)
```

```
##      zn  black rad  age    dis best5Ytrain
## 4   0.0 394.12   3 58.7 6.0622           0
## 9  12.5 396.90   5 82.9 6.2267           0
## 10 12.5 390.50   5 39.0 5.4509           0
## 13  0.0 395.62   4 56.5 4.4986           1
## 14  0.0 386.85   4 29.3 4.4986           1
## 18  0.0 396.90   4 91.7 3.9769           1
```

```
# glm for best 5 predictors
bosglm5 <- glm(best5Ytrain ~ zn + black + rad + age + dis, data = best5, family = binomial())
summary(bosglm5)
```

```
##
## Call:
## glm(formula = best5Ytrain ~ zn + black + rad + age + dis, family = binomial(),
##     data = best5)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.705647   2.796916  -0.610  0.54197
## zn          -0.014395   0.021799  -0.660  0.50901
## black       -0.003586   0.005754  -0.623  0.53309
## rad          0.387408   0.149651   2.589  0.00963 **
## age          0.021204   0.012590   1.684  0.09213 .
## dis         -0.237681   0.193829  -1.226  0.22011
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 210.61  on 151  degrees of freedom
## Residual deviance: 104.15  on 146  degrees of freedom
## AIC: 116.15
##
## Number of Fisher Scoring iterations: 8
```

```
bos5prob <- predict(bosglm5, data = best5, type = "response")
bos5pred <- rep("1", length(bos5prob))
bos5pred[bos5prob <= 0.5] <- "0"
table(bos5pred, best5$best5Ytrain)
```

```
##
## bos5pred  0  1
##        0 67 19
##        1 11 55
```

```
mean(bos5pred == best5$best5Ytrain)
```

```
## [1] 0.8026316
```

```r
mean(bos5pred != best5$best5Ytrain)
```

```
## [1] 0.1973684
```

The glm for our best 5 predictors has a misclassification rate of 19.7%

From the given models and their misclassification rates, I believe our hero model is the glm model for the best three predictors in our model. This is because it has the lowest misclassification rate. All of our findings from our models had misclassification rates above 10%, even some in the 20% area. But, the glm model for the best three predictors had the lowest of all six models.