DATA 621 – Business Analytics and Data Mining

Homework 3: Critical Thinking Group 2

VP

2020-04-05

Step 1.

Download the classification output data set.

```
df <- read.csv("https://raw.githubusercontent.com/mkivenson/Business-Analytics-Data-Mining/master/Class
kable(head(df,10), booktabs = T)
```

zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	lstat	medv	target
0	19.58	0	0.605	7.929	96.2	2.0459	5	403	14.7	3.70	50.0	1
0	19.58	1	0.871	5.403	100.0	1.3216	5	403	14.7	26.82	13.4	1
0	18.10	0	0.740	6.485	100.0	1.9784	24	666	20.2	18.85	15.4	1
30	4.93	0	0.428	6.393	7.8	7.0355	6	300	16.6	5.19	23.7	0
0	2.46	0	0.488	7.155	92.2	2.7006	3	193	17.8	4.82	37.9	0
0	8.56	0	0.520	6.781	71.3	2.8561	5	384	20.9	7.67	26.5	0
0	18.10	0	0.693	5.453	100.0	1.4896	24	666	20.2	30.59	5.0	1
0	18.10	0	0.693	4.519	100.0	1.6582	24	666	20.2	36.98	7.0	1
0	5.19	0	0.515	6.316	38.1	6.4584	5	224	20.2	5.68	22.2	0
80	3.64	0	0.392	5.876	19.1	9.2203	1	315	16.4	9.25	20.9	0

Data Exploration

Summary

First, we take a look at a summary of the data. A few items of interest are revealed:

- There are no missing values in the dataset
- There are no immediately apparent outliers
- Expected clusters are of similar size (237 and 229). This is a necessary assumption for algorithms such as K-Means clustering.

```
summary(df)
```

```
##        zn             indus            chas              nox
##  Min.   :  0.00   Min.   : 0.460   Min.   :0.00000   Min.   :0.3890
##  1st Qu.:  0.00   1st Qu.: 5.145   1st Qu.:0.00000   1st Qu.:0.4480
##  Median :  0.00   Median : 9.690   Median :0.00000   Median :0.5380
##  Mean   : 11.58   Mean   :11.105   Mean   :0.07082   Mean   :0.5543
##  3rd Qu.: 16.25   3rd Qu.:18.100   3rd Qu.:0.00000   3rd Qu.:0.6240
##  Max.   :100.00   Max.   :27.740   Max.   :1.00000   Max.   :0.8710
##        rm             age              dis               rad
```

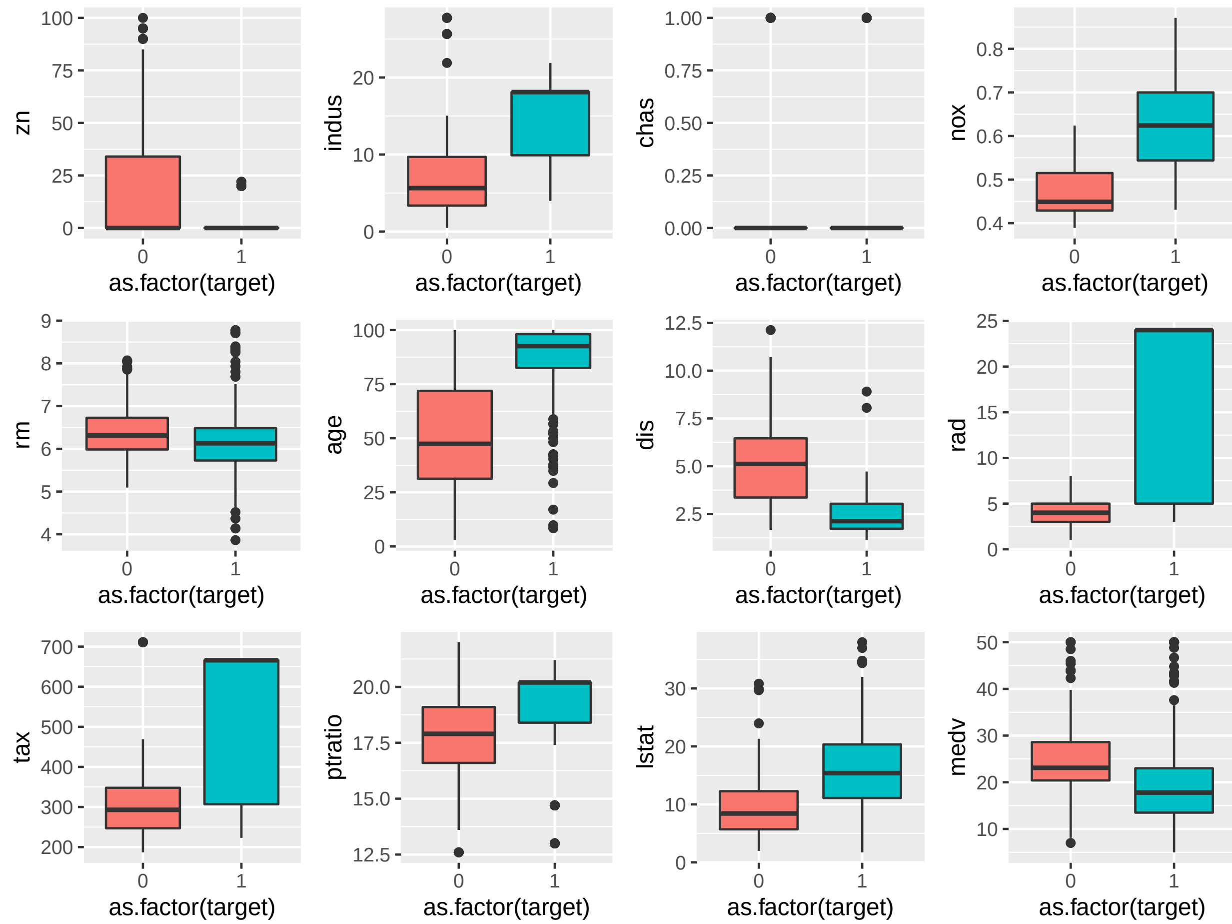
```
##  Min.   :3.863   Min.   :  2.90   Min.   : 1.130   Min.   : 1.00
##  1st Qu.:5.887   1st Qu.: 43.88   1st Qu.: 2.101   1st Qu.: 4.00
##  Median :6.210   Median : 77.15   Median : 3.191   Median : 5.00
##  Mean   :6.291   Mean   : 68.37   Mean   : 3.796   Mean   : 9.53
##  3rd Qu.:6.630   3rd Qu.: 94.10   3rd Qu.: 5.215   3rd Qu.:24.00
##  Max.   :8.780   Max.   :100.00   Max.   :12.127   Max.   :24.00
##       tax           ptratio         lstat             medv
##  Min.   :187.0   Min.   :12.6   Min.   : 1.730   Min.   : 5.00
##  1st Qu.:281.0   1st Qu.:16.9   1st Qu.: 7.043   1st Qu.:17.02
##  Median :334.5   Median :18.9   Median :11.350   Median :21.20
##  Mean   :409.5   Mean   :18.4   Mean   :12.631   Mean   :22.59
##  3rd Qu.:666.0   3rd Qu.:20.2   3rd Qu.:16.930   3rd Qu.:25.00
##  Max.   :711.0   Max.   :22.0   Max.   :37.970   Max.   :50.00
##      target
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.4914
##  3rd Qu.:1.0000
##  Max.   :1.0000
```

Boxplots

Next, we create boxplots of each of the features - color coded by the target variable. These boxplots reveal significant information about the predictor variables

- The `chas` dummy variable has most of its values at 0
- indus, zn, nox, age, dis, rad, tax, ptratio, lstat, and medv seem to have strong affects on the target variable

```
grid.arrange(ggplot(df, aes(x = as.factor(target), y = zn, fill = as.factor(target))) + geom_boxplot() +
             ggplot(df, aes(x = as.factor(target), y = indus, fill = as.factor(target))) + geom_boxplot
             ggplot(df, aes(x = as.factor(target), y = chas, fill = as.factor(target))) + geom_boxplot(
             ggplot(df, aes(x = as.factor(target), y = nox, fill = as.factor(target))) + geom_boxplot()
             ggplot(df, aes(x = as.factor(target), y = rm, fill = as.factor(target))) + geom_boxplot() +
             ggplot(df, aes(x = as.factor(target), y = age, fill = as.factor(target))) + geom_boxplot()
             ggplot(df, aes(x = as.factor(target), y = dis, fill = as.factor(target))) + geom_boxplot()
             ggplot(df, aes(x = as.factor(target), y = rad, fill = as.factor(target))) + geom_boxplot()
             ggplot(df, aes(x = as.factor(target), y = tax, fill = as.factor(target))) + geom_boxplot()
             ggplot(df, aes(x = as.factor(target), y = ptratio, fill = as.factor(target))) + geom_boxpl
             ggplot(df, aes(x = as.factor(target), y = lstat, fill = as.factor(target))) + geom_boxplot
             ggplot(df, aes(x = as.factor(target), y = medv, fill = as.factor(target))) + geom_boxplot(
             ncol=4)
```

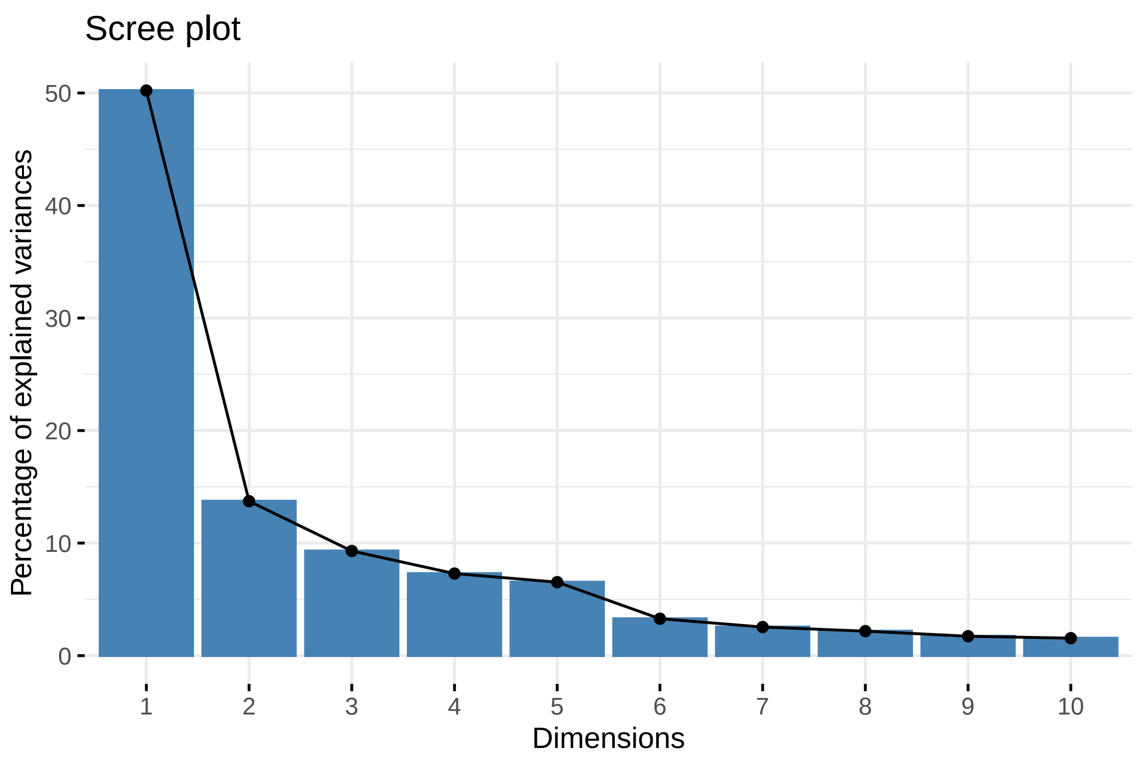


PCA Component Visualization

PCA can be used for classification, but for now, it will be used to visualize the clusters. First, the number of components will be selected based on the variances explained by each component.

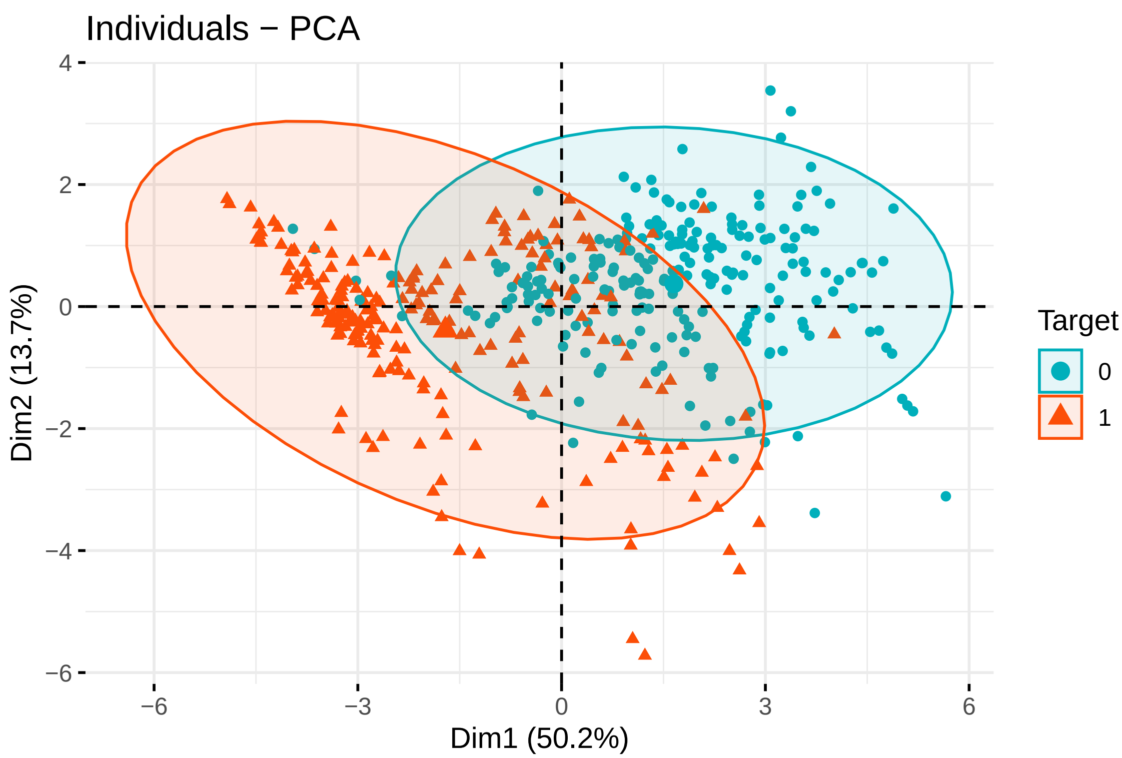
Taking a look at the plot of percentages explained by each principal component, it seems like most of the variance can be explained by 2 principal components.

```
df.pca <- prcomp(df[1:12], center = TRUE, scale. = TRUE)
fviz_eig(df.pca)
```



Using these two principal components, a scatterplot of the clusters can be created. Having two principal components makes it easier to distinguish between the two clusters, though there is some overlap.

```
fviz_pca_ind(df.pca,
             col.ind = as.factor(df$target), # Color by the quality of representation
             palette = c("#00AFBB",  "#FC4E07"),
             addEllipses = TRUE,
             legend.title = "Target",
             labels = FALSE
             )
```

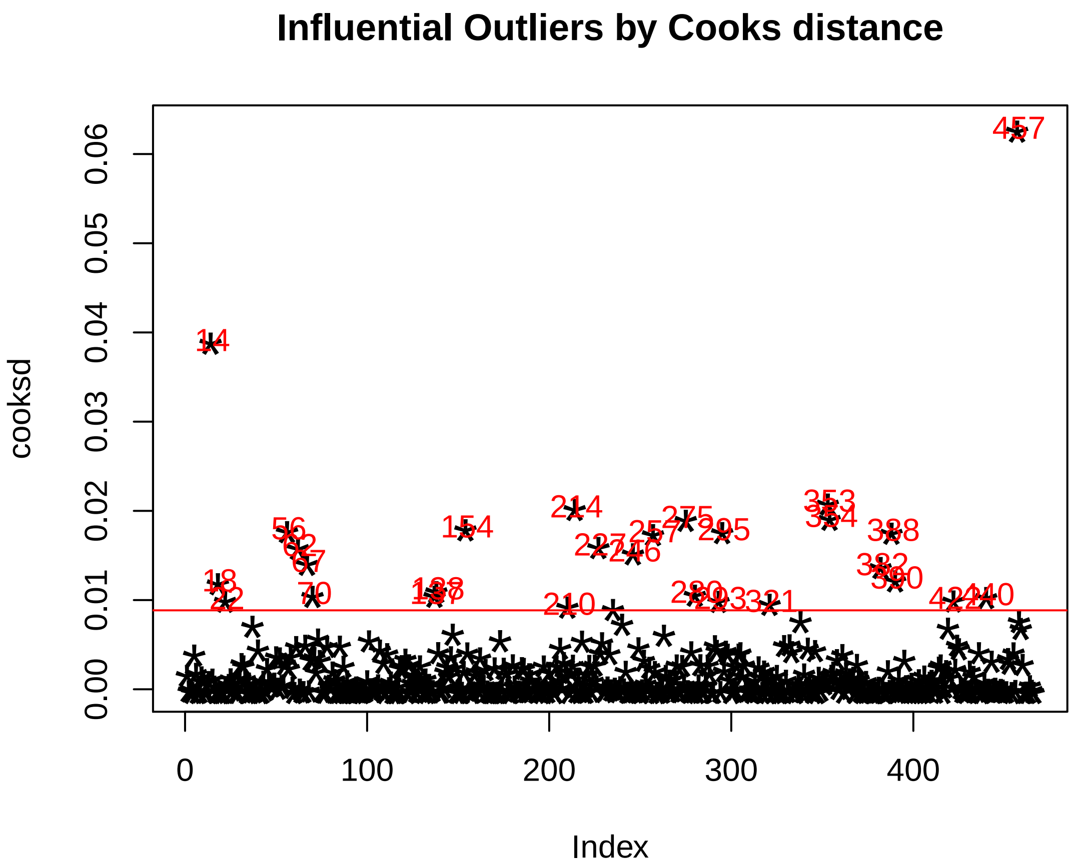


Data Preparation

Since the dataset does not have any missing values and there are no outliers that particulary stand out, data preparation will be limited. However, we will locate and address any influential outliers using Cooks Distance. Outliers that have a Cooks distance outside the acceptable threshold of 4 / (N - k - 1) where N is the number of observations and k is the number of predictive variables, will be removed.

Cooks Distance

```
mod <- lm(target ~ ., data=df)
cooksd <- cooks.distance(mod)
plot(cooksd, pch="*", cex=2, main="Influential Outliers by Cooks distance")
abline(h = 4 / (nrow(df) - ncol(df) - 1), col="red")  # add cutoff line
text(x=1:length(cooksd)+1, y=cooksd, labels=ifelse(cooksd>4*mean(cooksd, na.rm=T),names(cooksd),""), col
```



We remove the influencial outliers. Removing these outliers also makes the two primary components (visualized in the previous step) explain more of the variance in the data.

```
influential <- as.numeric(names(cooksd)[(cooksd > 4 / (nrow(df) - ncol(df) - 1))])
df <- df[-influential, ]
```

Building logistic regression

We will build a logistic classifer using generlized linear regresson with binomaial distribution.

Lets evaluate the distribution of target class label and check whether the dataset is imbalanced or not.

```
table(df$target)
```

```
##
##   0   1
## 224 213
```

we see that both label 0 and label 1 is balanced and have nearly equal number of datapoints.

Now lets split the given data set into 80% of training data and 20% testing data. And build logistic classifer with the training set

Model 1: All Variable

```
split = sample.split(df$target, SplitRatio = 0.8)
training_set = subset(df, split == TRUE)
test_set = subset(df, split == FALSE)

model1 <- glm(target ~ ., data = training_set, family = "binomial")
summary(model1)
```

```
##
## Call:
## glm(formula = target ~ ., family = "binomial", data = training_set)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7051  -0.0705   0.0000   0.0004   3.7534
##
## Coefficients:
##              Estimate Std. Error z value   Pr(>|z|)
## (Intercept) -34.340164   9.241005  -3.716   0.000202 ***
## zn           -0.083633   0.052158  -1.603   0.108835
## indus         0.298232   0.143593   2.077   0.037809 *
## chas         -2.908559   2.433155  -1.195   0.231936
## nox          43.983373   9.833343   4.473 0.00000772 ***
## rm           -0.758046   1.054874  -0.719   0.472379
## age           0.079291   0.021868   3.626   0.000288 ***
## dis           0.509342   0.318314   1.600   0.109571
## rad           1.118164   0.297577   3.758   0.000172 ***
## tax          -0.028988   0.009174  -3.160   0.001578 **
## ptratio       0.361584   0.193082   1.873   0.061110 .
## lstat        -0.057079   0.091460  -0.624   0.532571
## medv          0.116462   0.095802   1.216   0.224118
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 483.58  on 348  degrees of freedom
## Residual deviance: 103.21  on 336  degrees of freedom
## AIC: 129.21
##
## Number of Fisher Scoring iterations: 10
```

If I drop all non signifigant variables I am left with the following variables:nox, age, dis, rad, tax, pratio
Therefore I am going to build a model with thoses variables.
Here is the summary for that model (model2)

```
model2 <- glm(target~nox+ age+dis+ rad+tax+ptratio  , data =training_set, family="binomial" )
summary(model2)
```

```
##
## Call:
## glm(formula = target ~ nox + age + dis + rad + tax + ptratio,
```

```
##     family = "binomial", data = training_set)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3454  -0.0980  -0.0013   0.0016   3.3810
##
## Coefficients:
##               Estimate Std. Error z value    Pr(>|z|)
## (Intercept) -36.651554   7.130384  -5.140 0.000000274 ***
## nox          53.556380  10.383757   5.158 0.000000250 ***
## age           0.064890   0.016821   3.858    0.000115 ***
## dis           0.229638   0.214132   1.072    0.283534
## rad           0.863527   0.221784   3.894 0.000098790 ***
## tax          -0.021919   0.006311  -3.473    0.000515 ***
## ptratio       0.249334   0.132656   1.880    0.060169 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 483.58  on 348  degrees of freedom
## Residual deviance: 117.40  on 342  degrees of freedom
## AIC: 131.4
##
## Number of Fisher Scoring iterations: 9
```

If I drop all non signifigant variables I am left with the following variables:nox, age, pratio Therefore I am going to build a model with thoses variables.
Here is the summary for that model (model2)

```
model3 <- glm(target~nox+ age+ rad+tax+ptratio  , data =training_set, family="binomial" )
summary(model3)
```

```
##
## Call:
## glm(formula = target ~ nox + age + rad + tax + ptratio, family = "binomial",
##     data = training_set)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3507  -0.0869  -0.0011   0.0018   3.3315
##
## Coefficients:
##               Estimate Std. Error z value    Pr(>|z|)
## (Intercept) -33.951956   6.501157  -5.222 0.000000177 ***
## nox          51.290775  10.203197   5.027 0.000000498 ***
## age           0.061621   0.016223   3.799    0.000146 ***
## rad           0.865372   0.221142   3.913 0.000091084 ***
## tax          -0.022993   0.006246  -3.681    0.000232 ***
## ptratio       0.244325   0.130986   1.865    0.062142 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 483.58  on 348  degrees of freedom
## Residual deviance: 118.53  on 343  degrees of freedom
## AIC: 130.53
##
## Number of Fisher Scoring iterations: 9
```

```
model4 <- glm(target~nox+ age+ rad+tax  , data =training_set, family="binomial" )
summary(model4)
```

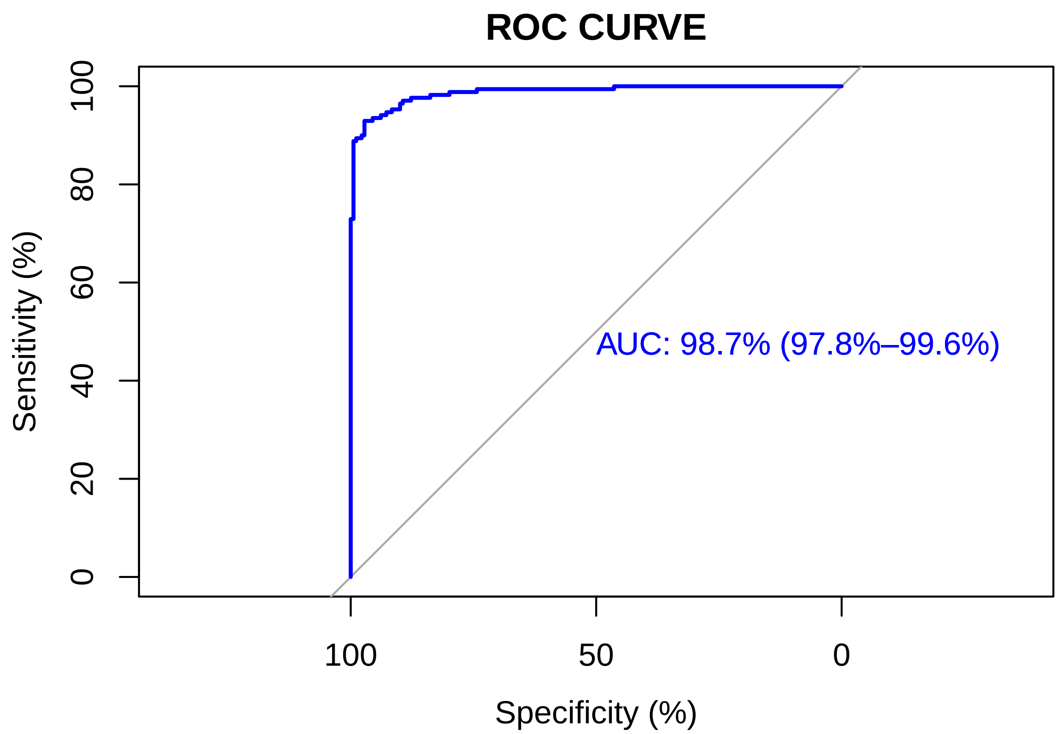
```
##
## Call:
## glm(formula = target ~ nox + age + rad + tax, family = "binomial",
##     data = training_set)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1111  -0.1212  -0.0032   0.0067   3.1645
##
## Coefficients:
##               Estimate Std. Error z value      Pr(>|z|)
## (Intercept) -27.268584   4.702604  -5.799 0.00000000669 ***
## nox          48.251002   9.762751   4.942 0.00000077184 ***
## age           0.054496   0.014934   3.649      0.000263 ***
## rad           0.716235   0.181813   3.939 0.00008168161 ***
## tax          -0.020462   0.005923  -3.455      0.000551 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 483.58  on 348  degrees of freedom
## Residual deviance: 122.12  on 344  degrees of freedom
## AIC: 132.12
##
## Number of Fisher Scoring iterations: 9
```

#Select Models: I am going to select the model based on area under the ROC curve (A/K/A AUC) and AIC.

```
roc(target~model1$fitted.values, data = training_set,plot = TRUE, main = "ROC CURVE", col= "blue",
    percent=TRUE,
    ci = TRUE, # compute AUC (of AUC by default)
    print.auc = TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```




```
##
## Call:
## roc.formula(formula = target ~ model1$fitted.values, data = training_set,     plot = TRUE, main = "R
##
## Data: model1$fitted.values in 179 controls (target 0) < 170 cases (target 1).
## Area under the curve: 98.72%
## 95% CI: 97.83%-99.61% (DeLong)
```

```
model1$aic
```

```
## [1] 129.2068
```

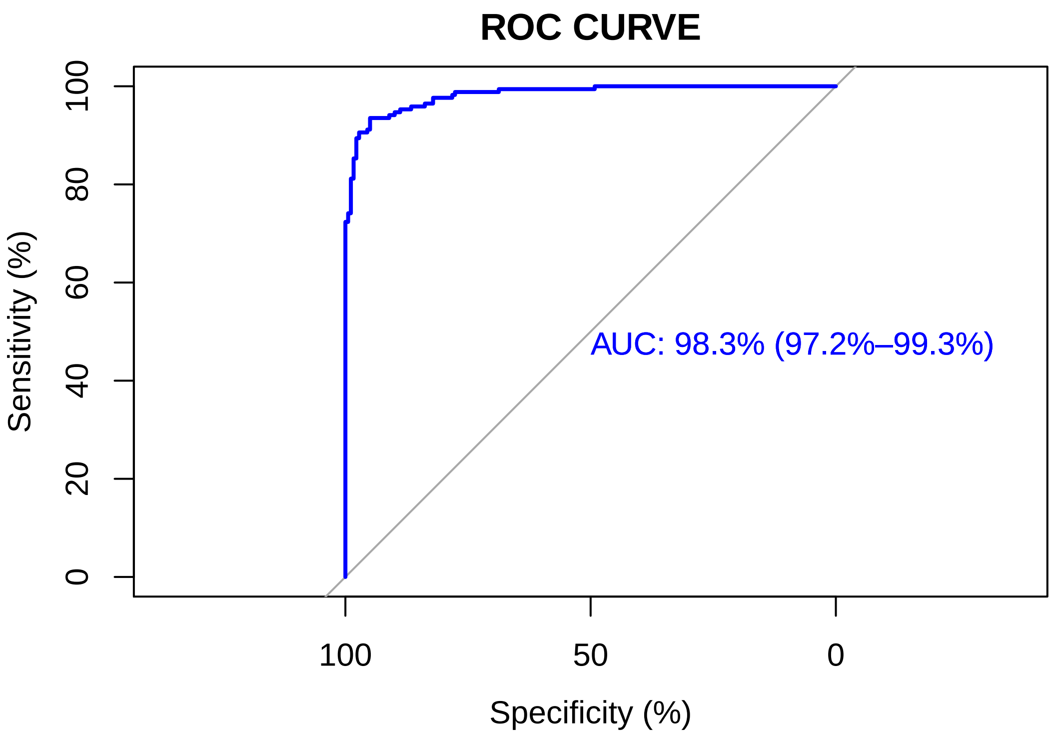
The AIC for model1 is 129.2068247

Model2 Variables in Model 2: `nox + age+dis+ rad + tax + ptratio`

```
roc(target~model2$fitted.values, data = training_set, plot = TRUE, main = "ROC CURVE", col= "blue",perc
    ci = TRUE, # compute AUC (of AUC by default)
    print.auc = TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```




```
##
## Call:
## roc.formula(formula = target ~ model2$fitted.values, data = training_set,     plot = TRUE, main = "R
##
## Data: model2$fitted.values in 179 controls (target 0) < 170 cases (target 1).
## Area under the curve: 98.27%
## 95% CI: 97.23%-99.3% (DeLong)
```

```
model2$aic
```

```
## [1] 131.3969
```

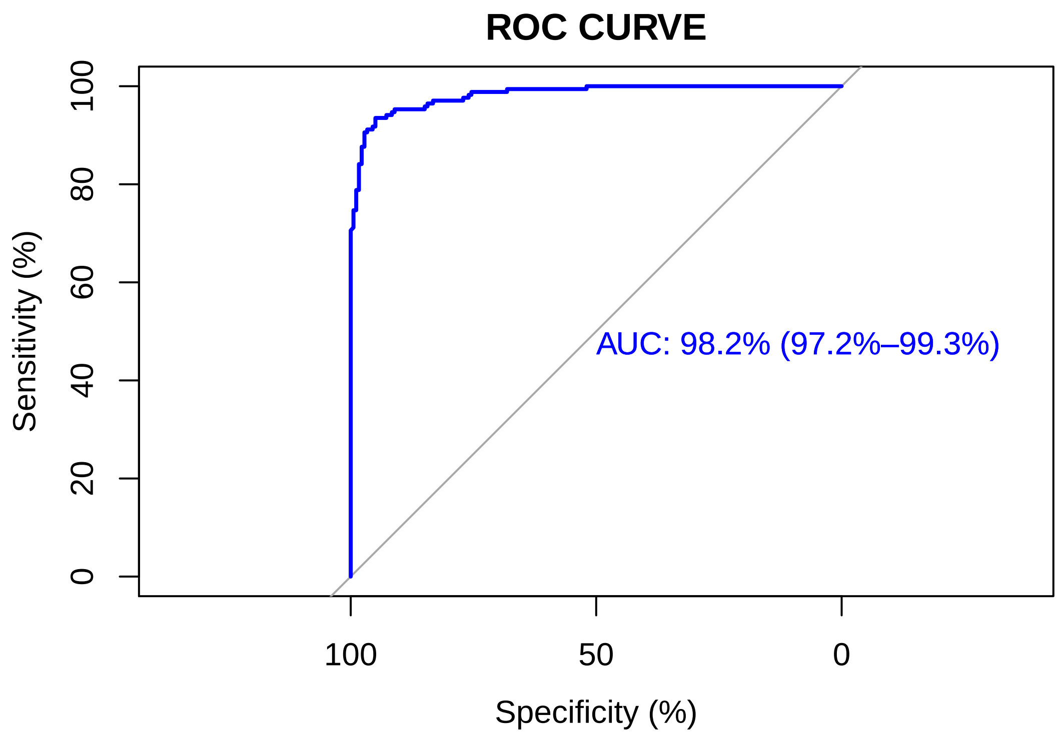
The AIC for model2 is 131.3969212

Model3 Variables: `nox+ age+ rad+tax+ptratio`,

```
roc(target~model3$fitted.values, data = training_set, plot = TRUE, main = "ROC CURVE", col= "blue",perc
    ci = TRUE, # compute AUC (of AUC by default)
    print.auc = TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```




```
##
## Call:
## roc.formula(formula = target ~ model3$fitted.values, data = training_set,     plot = TRUE, main = "R
##
## Data: model3$fitted.values in 179 controls (target 0) < 170 cases (target 1).
## Area under the curve: 98.23%
## 95% CI: 97.19%-99.28% (DeLong)
```

```
model3$aic
```

```
## [1] 130.532
```

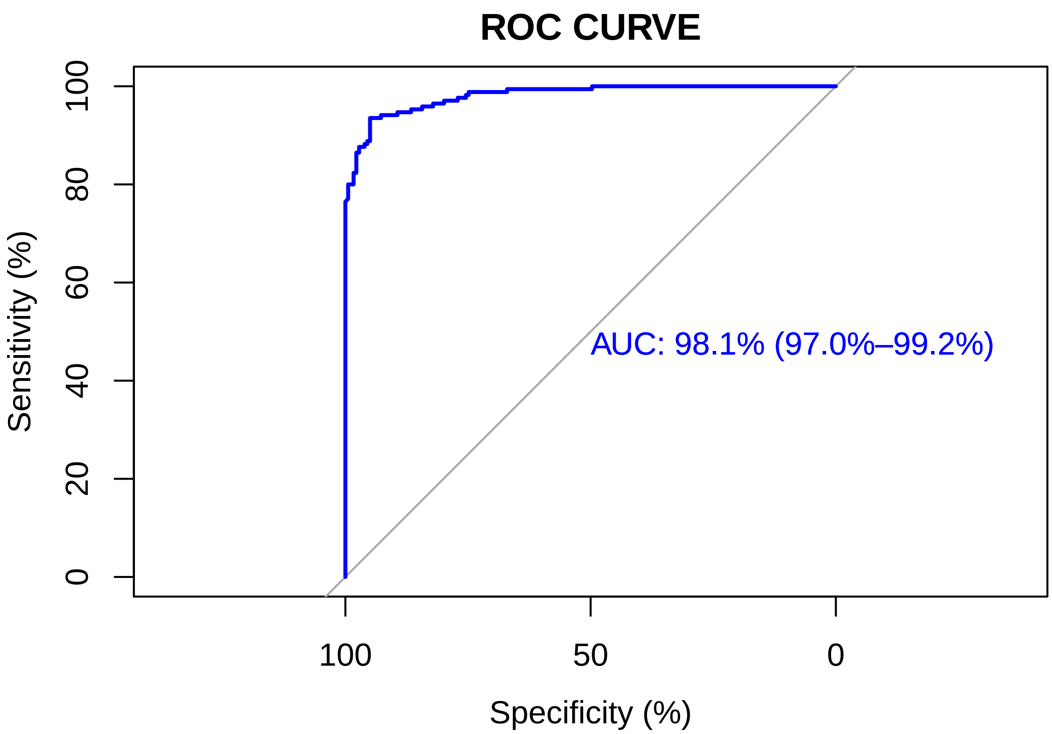
The AIC for model3 is 130.5320461

Model4 Variables: `nox+ age+ rad+tax`,

```
roc(target~model4$fitted.values, data = training_set, plot = TRUE, main = "ROC CURVE", col= "blue",perc
    ci = TRUE, # compute AUC (of AUC by default)
    print.auc = TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```




```
##
## Call:
## roc.formula(formula = target ~ model4$fitted.values, data = training_set,     plot = TRUE, main = "R
##
## Data: model4$fitted.values in 179 controls (target 0) < 170 cases (target 1).
## Area under the curve: 98.12%
## 95% CI: 97.04%-99.19% (DeLong)
```

```
model4$aic
```

```
## [1] 132.118
```

The AIC for model4 is 132.1180309

Based the fact that the area under the curve for model 2 and model 3 are virtually identical and the AIC for model 2 is about 1/2 the AIC for model 1 I am going to select model2.