Filippo Riva
MT 3100 (Brandt)
Statistical Learning

Assignment 9

In this assignment we continue to study the Auto data set. We will stick with a training set of 300 and a test set of 92. We will find some QDA and KNN models.

1. QDA is a close relative of LDA, just some of the assumptions are different. Let's find some QDA models. For QDA notation, see p. 179. Set up your training and test sets as you did in Assignment 8.

a. Find a QDA model for FromUS as a function of cylinders, displacement, and weight. Complete the table:

CODE:

```
 Auto = read.csv(file="Auto.csv",head=TRUE,sep=",",stringsAsFactors=FALSE, na.strings="?")
Auto = na.omit(Auto)
displacement=Auto$displacement
cylinders=Auto$cylinders
horsepower=Auto$horsepower
weight=Auto$weight
mpg=Auto$mpg
FromUS=rep(0,392) #Set up a vector of 392 zeroes
FromUS[Auto$origin==1]=1 #Switch value fro US cars
Auto=data.frame(Auto,FromUS) #Add new variable to Auto
set.seed(0) # Choose a seed
train=sample.int(392, 300) # Randomly choose 300 of the cars.
Auto.train=Auto[train,] # The rows of Auto for the training cars.
Auto.test=Auto[-train,] # The rows of Auto for the test cars.
FromUS.test=Auto$FromUS[-train] # FromUS values for the test cars.
library(MASS)
qdaMod= qda(FromUS~cylinders + displacement + weight, data = Auto.train, family =
binomial)
qdaMod
qdaMod.pred=predict(qdaMod,Auto.test)
qdaMod.class=qdaMod.pred$class
table(qdaMod.class,FromUS.test)

set.seed(3) # Choose a seed
train=sample.int(392, 300) # Randomly choose 300 of the cars.
Auto.train=Auto[train,] # The rows of Auto for the training cars.
Auto.test=Auto[-train,] # The rows of Auto for the test cars.
FromUS.test=Auto$FromUS[-train] # FromUS values for the test cars.

library(MASS)
qdaMod= qda(FromUS~cylinders + displacement + weight, data = Auto.train, family =
binomial)
```

```
qdaMod
qdaMod.pred=predict(qdaMod,Auto.test)
qdaMod.class=qdaMod.pred$class
table(qdaMod.class,FromUS.test)
```

```
set.seed(4) # Choose a seed
train=sample.int(392, 300) # Randomly choose 300 of the cars.
Auto.train=Auto[train,] # The rows of Auto for the training cars.
Auto.test=Auto[-train,] # The rows of Auto for the test cars.
FromUS.test=Auto$FromUS[-train] # FromUS values for the test cars.

library(MASS)
qdaMod= qda(FromUS~cylinders + displacement + weight, data = Auto.train, family =
binomial)
qdaMod
qdaMod.pred=predict(qdaMod,Auto.test)
qdaMod.class=qdaMod.pred$class
table(qdaMod.class,FromUS.test)
```

Seed	Confusion Matrix	Success Rate
0	FromUS.test qdaMod.class 0 1 0 35 6 1 6 45	0.8695 = 86.95%
3	FromUS.test qdaMod.class 0 1 0 32 12 1 2 46	0.8478 = 84.78%
4	FromUS.test qdaMod.class 0 1 0 28 11 1 5 48	0.8260 = 82.60%

QDA Model 1: FromUS~cylinders+displacement+weight

b. Find a QDA model for origin as a function of cylinders, displacement, and weight. Complete the table:

CODE:

```
Auto = read.csv(file="Auto.csv",head=TRUE,sep=",",stringsAsFactors=FALSE, na.strings="?")
Auto = na.omit(Auto)
displacement=Auto$displacement
```

```
cylinders=Auto$cylinders
horsepower=Auto$horsepower
weight=Auto$weight
mpg=Auto$mpg
origin=Auto$origin

#FromUS=rep(0,392) #Set up a vector of 392 zeroes
#FromUS[Auto$origin==1]=1 #Switch value fro US cars
#Auto=data.frame(Auto,FromUS) #Add new variable to Auto

set.seed(0) # Choose a seed
train=sample.int(392, 300) # Randomly choose 300 of the cars.
Auto.train=Auto[train,] # The rows of Auto for the training cars.
Auto.test=Auto[-train,] # The rows of Auto for the test cars.
origin.test=Auto$origin[-train] # FromUS values for the test cars.

library(MASS)
qdaMod = qda(origin~cylinders+displacement+weight)
qdaMod
qdaMod.pred=predict(qdaMod,Auto.test)
qdaMod.class=qdaMod.pred$class
table(qdaMod.class,origin.test)

set.seed(3) # Choose a seed
train=sample.int(392, 300) # Randomly choose 300 of the cars.
Auto.train=Auto[train,] # The rows of Auto for the training cars.
Auto.test=Auto[-train,] # The rows of Auto for the test cars.
origin.test=Auto$origin[-train] # FromUS values for the test cars.

library(MASS)
qdaMod = qda(origin~cylinders+displacement+weight)
qdaMod
qdaMod.pred=predict(qdaMod,Auto.test)
qdaMod.class=qdaMod.pred$class
table(qdaMod.class,origin.test)


set.seed(4) # Choose a seed
train=sample.int(392, 300) # Randomly choose 300 of the cars.
Auto.train=Auto[train,] # The rows of Auto for the training cars.
Auto.test=Auto[-train,] # The rows of Auto for the test cars.
origin.test=Auto$origin[-train] # FromUS values for the test cars.

library(MASS)
qdaMod = qda(origin~cylinders+displacement+weight)
qdaMod
qdaMod.pred=predict(qdaMod,Auto.test)
qdaMod.class=qdaMod.pred$class
```

table(qdaMod.class,origin.test)

Seed	Confusion Matrix	Success Rate
0	origin.test qdaMod2.class 1 2 3 1 46 2 4 2 0 9 2 3 5 8 16	0.7717 = 77.17%
3	origin.test qdaMod2.class 1 2 3 1 47 1 1 2 3 8 4 3 8 8 12	0.7283 = 72.83 %
4	origin.test qdaMod2.class 1 2 3 1 48 1 2 2 3 6 0 3 8 9 15	0.75 = 75%

QDA Model 2: origin~cylinders+displacement+weight

c. Compare your QDA tables with the corresponding LDA tables from Assignment 8. Do you think QDA is an improvement over LDA for this situation?

By comparing the result with the lda model found in assignment 8, we can not show an improvement by using the new model function qda, Most of the success rate are higher in assignment 8 showing a better model created with the function lda.

2. Now we find some KNN models. In R, the setup for KNN is quite different from our previous model types. See details on p. 181-182.

Here we will set up our training set a bit differently. We could go with the approach in Question 1, but this new approach may seem more intuitive. Instead of randomly selecting 300 cars for the training set, we will randomly scramble the entire data set. Then we can train the first 300 cars in this scrambled set and test on the last 92. As before, we will set a seed so we can all see the same results. In part a, we will find a model for FromUS as a function of cylinders, displacement, and weight. The commands below will set it up. Please compare these with the commands the book uses in its first KNN example. They are similar but not identical to the book's setup.

set.seed(0)

scram=sample.int(392, 392) # Randomly scramble integers 1 through 392
newAuto=Auto[scram,] # newAuto = same data set with the rows scrambled

trn=rep(TRUE,300) # Use first 300 rows of newAuto as training set
tst=rep(FALSE,92) # Use last 92 rows of newAuto as test set
train=c(trn,tst) # Put trn and tst together to form train
 # cylinders, displacement, and weight are in columns 2, 3, and 5
train.X=newAuto[train,c(2,3,5)]
test.X=newAuto[!train,c(2,3,5)]

train.FromUS=newAuto[train,10] # Column 10 is FromUS
test.FromUS=newAuto[!train,10]

a. **Find KNN models for FromUS as a function of cylinders, displacement, and weight. Complete the table:**

   CODE:
   seed (0) K = 2

```
Auto = read.csv(file="Auto.csv",head=TRUE,stringsAsFactors=FALSE, na.strings="?")
Auto = na.omit(Auto)
dim(Auto)
library(class)
displacement=Auto$displacement
cylinders=Auto$cylinders
horsepower=Auto$horsepower
weight=Auto$weight
mpg=Auto$mpg

origin=Auto$origin
FromUS=rep(0,392) #Set up a vector of 392 zeroes
FromUS[Auto$origin==1]=1 #Switch value fro US cars
Auto=data.frame(Auto,FromUS) #Add new variable to Auto
set.seed(0)
scram=sample.int(392, 392) # Randomly scramble integers 1 through 392
newAuto=Auto[scram,] # newAuto = same data set with the rows scrambled
trn=rep(TRUE,300) # Use first 300 rows of newAuto as training set
tst=rep(FALSE,92) # Use last 92 rows of newAuto as test set
train=c(trn,tst) # Put trn and tst together to form train
# cylinders, displacement, and weight are in columns 2, 3, and 5
train.X=newAuto[train,c(2,3,5)]
test.X=newAuto[!train,c(2,3,5)]
train.FromUS=newAuto[train,10] # Column 10 is FromUS
test.FromUS=newAuto[!train,10]
knnMod=knn(train.X, test.X, train.FromUS, k = 2)
knnMod
table(knnMod,test.FromUS)
```

Seed (0) K = 10

```
Auto = read.csv(file="Auto.csv",head=TRUE,stringsAsFactors=FALSE, na.strings="?")
Auto = na.omit(Auto)
dim(Auto)
library(class)
displacement=Auto$displacement
cylinders=Auto$cylinders
horsepower=Auto$horsepower
weight=Auto$weight
mpg=Auto$mpg

origin=Auto$origin
FromUS=rep(0,392) #Set up a vector of 392 zeroes
FromUS[Auto$origin==1]=1 #Switch value fro US cars
Auto=data.frame(Auto,FromUS) #Add new variable to Auto
set.seed(0)
scram=sample.int(392, 392) # Randomly scramble integers 1 through 392
newAuto=Auto[scram,] # newAuto = same data set with the rows scrambled
trn=rep(TRUE,300) # Use first 300 rows of newAuto as training set
tst=rep(FALSE,92) # Use last 92 rows of newAuto as test set
train=c(trn,tst) # Put trn and tst together to form train
# cylinders, displacement, and weight are in columns 2, 3, and 5
train.X=newAuto[train,c(2,3,5)]
test.X=newAuto[!train,c(2,3,5)]
train.FromUS=newAuto[train,10] # Column 10 is FromUS
test.FromUS=newAuto[!train,10]
knnMod=knn(train.X, test.X, train.FromUS, k = 10)
knnMod
table(knnMod,test.FromUS)
```

Seed (3) K = 2

```
Auto = read.csv(file="Auto.csv",head=TRUE,stringsAsFactors=FALSE, na.strings="?")
Auto = na.omit(Auto)
dim(Auto)
library(class)
displacement=Auto$displacement
cylinders=Auto$cylinders
horsepower=Auto$horsepower
weight=Auto$weight
mpg=Auto$mpg

origin=Auto$origin
FromUS=rep(0,392) #Set up a vector of 392 zeroes
FromUS[Auto$origin==1]=1 #Switch value fro US cars
Auto=data.frame(Auto,FromUS) #Add new variable to Auto
set.seed(3)
```

```
scram=sample.int(392, 392) # Randomly scramble integers 1 through 392
newAuto=Auto[scram,] # newAuto = same data set with the rows scrambled
trn=rep(TRUE,300) # Use first 300 rows of newAuto as training set
tst=rep(FALSE,92) # Use last 92 rows of newAuto as test set
train=c(trn,tst) # Put trn and tst together to form train
# cylinders, displacement, and weight are in columns 2, 3, and 5
train.X=newAuto[train,c(2,3,5)]
test.X=newAuto[!train,c(2,3,5)]
train.FromUS=newAuto[train,10] # Column 10 is FromUS
test.FromUS=newAuto[!train,10]
knnMod=knn(train.X, test.X, train.FromUS, k = 2)
knnMod
table(knnMod,test.FromUS)
```

Seed (3) K =10

```
Auto = read.csv(file="Auto.csv",head=TRUE,stringsAsFactors=FALSE, na.strings="?")
Auto = na.omit(Auto)
dim(Auto)
library(class)
displacement=Auto$displacement
cylinders=Auto$cylinders
horsepower=Auto$horsepower
weight=Auto$weight
mpg=Auto$mpg

origin=Auto$origin
FromUS=rep(0,392) #Set up a vector of 392 zeroes
FromUS[Auto$origin==1]=1 #Switch value fro US cars
Auto=data.frame(Auto,FromUS) #Add new variable to Auto
set.seed(3)
scram=sample.int(392, 392) # Randomly scramble integers 1 through 392
newAuto=Auto[scram,] # newAuto = same data set with the rows scrambled
trn=rep(TRUE,300) # Use first 300 rows of newAuto as training set
tst=rep(FALSE,92) # Use last 92 rows of newAuto as test set
train=c(trn,tst) # Put trn and tst together to form train
# cylinders, displacement, and weight are in columns 2, 3, and 5
train.X=newAuto[train,c(2,3,5)]
test.X=newAuto[!train,c(2,3,5)]
train.FromUS=newAuto[train,10] # Column 10 is FromUS
test.FromUS=newAuto[!train,10]
knnMod=knn(train.X, test.X, train.FromUS, k = 10)
knnMod
table(knnMod,test.FromUS)
```

Seed	K	Confusion Matrix	Success Rate
0	2	test.FromUS knnMod 0 1 0 33 9 1 8 42	0.8152 = 81.52%
0	10	test.FromUS knnMod 0 1 0 32 8 1 9 43	0.8152 = 81.52 %
3	2	test.FromUS knnMod 0 1 0 27 12 1 7 46	0.7934 = 79.34%
3	10	test.FromUS knnMod 0 1 0 31 14 1 3 44	0.8152= 81.52%

KNN Model 1: FromUS~cylinders+displacement+weight

b. Repeat part a with standardized data (see page 183 for details). This seems like a good idea because our predictors are very different in scale. Cylinders are mostly 4, 6, and 8, whereas weight is measured in thousands of pounds. Thus, when KNN looks for a test point's nearest neighbors, differences in weight will be much more significant than differences in number of cylinders. The solution is to scale the data set so that all variables have the same distribution of values. These commands set up the scaled data set:

```
set.seed(0)
scram=sample.int(392, 392) # Randomly scramble 1 through 392
newAuto=Auto[scram,] # Scramble the data set

# Define the "scaled" newAuto set.
# Remove 9th column (name) and 10th column (FromUS)
newAutoS=scale(newAuto[,-c(9,10)])

trn=rep(TRUE,300)  # Use first 300 rows of newAuto as training set
tst=rep(FALSE,92) # Use last 92 rows of newAuto as test set.
train=c(trn,tst) # Put trn and tst together to form train.
```

```
 train.X=newAutoS[train,c(2,3,5)]
test.X=newAutoS[!train,c(2,3,5)]

# NewAuto (without the S), since newAutoS does not have FromUS in it!
train.FromUS=newAuto[train,10]
test.FromUS=newAuto[!train,10]
```

CODE:

Seed (0) K=2

```
Auto = read.csv(file="Auto.csv",head=TRUE,stringsAsFactors=FALSE, na.strings="?")
Auto = na.omit(Auto)
dim(Auto)
library(class)
displacement=Auto$displacement
cylinders=Auto$cylinders
horsepower=Auto$horsepower
weight=Auto$weight
mpg=Auto$mpg
origin=Auto$origin
FromUS=rep(0,392) #Set up a vector of 392 zeroes
FromUS[Auto$origin==1]=1 #Switch value fro US cars
Auto=data.frame(Auto,FromUS) #Add new variable to Auto

set.seed(0)
scram=sample.int(392, 392) # Randomly scramble 1 through 392
newAuto=Auto[scram,] # Scramble the data set
# Define the "scaled" newAuto set.
# Remove 9th column (name) and 10th column (FromUS)
newAutoS=scale(newAuto[,-c(9,10)])
trn=rep(TRUE,300)  # Use first 300 rows of newAuto as training set
tst=rep(FALSE,92) # Use last 92 rows of newAuto as test set.
train=c(trn,tst) # Put trn and tst together to form train.
train.X=newAutoS[train,c(2,3,5)]
test.X=newAutoS[!train,c(2,3,5)]
# NewAuto (without the S), since newAutoS does not have FromUS in it!
train.FromUS=newAuto[train,10]
test.FromUS=newAuto[!train,10]
knnMod2=knn(train.X, test.X, train.FromUS, k = 2)
knnMod2
table(knnMod2,test.FromUS)
```

Seed (0) K=10

```
Auto = read.csv(file="Auto.csv",head=TRUE,stringsAsFactors=FALSE, na.strings="?")
Auto = na.omit(Auto)
dim(Auto)
```

```
library(class)
displacement=Auto$displacement
cylinders=Auto$cylinders
horsepower=Auto$horsepower
weight=Auto$weight
mpg=Auto$mpg
origin=Auto$origin
FromUS=rep(0,392) #Set up a vector of 392 zeroes
FromUS[Auto$origin==1]=1 #Switch value fro US cars
Auto=data.frame(Auto,FromUS) #Add new variable to Auto

set.seed(0)
scram=sample.int(392, 392) # Randomly scramble 1 through 392
newAuto=Auto[scram,] # Scramble the data set
# Define the "scaled" newAuto set.
# Remove 9th column (name) and 10th column (FromUS)
newAutoS=scale(newAuto[,-c(9,10)])
trn=rep(TRUE,300)  # Use first 300 rows of newAuto as training set
tst=rep(FALSE,92) # Use last 92 rows of newAuto as test set.
train=c(trn,tst) # Put trn and tst together to form train.
train.X=newAutoS[train,c(2,3,5)]
test.X=newAutoS[!train,c(2,3,5)]
# NewAuto (without the S), since newAutoS does not have FromUS in it!
train.FromUS=newAuto[train,10]
test.FromUS=newAuto[!train,10]
knnMod2=knn(train.X, test.X, train.FromUS, k = 10)
knnMod2
table(knnMod2,test.FromUS)
```

Seed (3) K=2

```
Auto = read.csv(file="Auto.csv",head=TRUE,stringsAsFactors=FALSE, na.strings="?")
Auto = na.omit(Auto)
dim(Auto)
library(class)
displacement=Auto$displacement
cylinders=Auto$cylinders
horsepower=Auto$horsepower
weight=Auto$weight
mpg=Auto$mpg
origin=Auto$origin
FromUS=rep(0,392) #Set up a vector of 392 zeroes
FromUS[Auto$origin==1]=1 #Switch value fro US cars
Auto=data.frame(Auto,FromUS) #Add new variable to Auto

set.seed(3)
scram=sample.int(392, 392) # Randomly scramble 1 through 392
newAuto=Auto[scram,] # Scramble the data set
```

```
# Define the "scaled" newAuto set.
# Remove 9th column (name) and 10th column (FromUS)
newAutoS=scale(newAuto[,-c(9,10)])
trn=rep(TRUE,300)  # Use first 300 rows of newAuto as training set
tst=rep(FALSE,92) # Use last 92 rows of newAuto as test set.
train=c(trn,tst) # Put trn and tst together to form train.
train.X=newAutoS[train,c(2,3,5)]
test.X=newAutoS[!train,c(2,3,5)]
# NewAuto (without the S), since newAutoS does not have FromUS in it!
train.FromUS=newAuto[train,10]
test.FromUS=newAuto[!train,10]
knnMod2=knn(train.X, test.X, train.FromUS, k = 2)
knnMod2
table(knnMod2,test.FromUS)
```

Seed (3) K=10

```
Auto = read.csv(file="Auto.csv",head=TRUE,stringsAsFactors=FALSE, na.strings="?")
Auto = na.omit(Auto)
dim(Auto)
library(class)
displacement=Auto$displacement
cylinders=Auto$cylinders
horsepower=Auto$horsepower
weight=Auto$weight
mpg=Auto$mpg
origin=Auto$origin
FromUS=rep(0,392) #Set up a vector of 392 zeroes
FromUS[Auto$origin==1]=1 #Switch value fro US cars
Auto=data.frame(Auto,FromUS) #Add new variable to Auto

set.seed(3)
scram=sample.int(392, 392) # Randomly scramble 1 through 392
newAuto=Auto[scram,] # Scramble the data set
# Define the "scaled" newAuto set.
# Remove 9th column (name) and 10th column (FromUS)
newAutoS=scale(newAuto[,-c(9,10)])
trn=rep(TRUE,300)  # Use first 300 rows of newAuto as training set
tst=rep(FALSE,92) # Use last 92 rows of newAuto as test set.
train=c(trn,tst) # Put trn and tst together to form train.
train.X=newAutoS[train,c(2,3,5)]
test.X=newAutoS[!train,c(2,3,5)]
# NewAuto (without the S), since newAutoS does not have FromUS in it!
train.FromUS=newAuto[train,10]
test.FromUS=newAuto[!train,10]
knnMod2=knn(train.X, test.X, train.FromUS, k = 10)
knnMod2
table(knnMod2,test.FromUS)
```

Seed	k	Confusion Matrix	Success Rate
0	2	test.FromUS knnMod2 0 1 0 34 3 1 7 48	0.8913 = 89.13%
0	10	test.FromUS knnMod2 0 1 0 35 5 1 6 46	0.8804 = 88.04%
3	2	test.FromUS knnMod2 0 1 0 29 9 1 5 49	0.8478 = 84.78%
3	10	test.FromUS knnMod2 0 1 0 29 10 1 5 48	0.8369 = 83.69%

KNN Model 2: FromUS~cylinders+displacement+weight (standardized)

c. Does it appear that using standardized data improved the model? Explain.

Yes, the success rate of every seeds has improved from the previous exercise, increasing their numbers.

d. Find KNN models for origin as a function of cylinders, displacement, and weight using standardized data values. Complete the table:

CODE:

Seed 0, K=2

```
Auto = read.csv(file="Auto.csv",head=TRUE,stringsAsFactors=FALSE, na.strings="?")
Auto = na.omit(Auto)
dim(Auto)
library(class)
displacement=Auto$displacement
cylinders=Auto$cylinders
horsepower=Auto$horsepower
weight=Auto$weight
mpg=Auto$mpg
origin=Auto$origin

origin=rep(0,392) #Set up a vector of 392 zeroes
origin[Auto$origin==1]=1 #Switch value fro US cars
Auto=data.frame(Auto,origin) #Add new variable to Auto

set.seed(0)
scram=sample.int(392, 392) # Randomly scramble 1 through 392
newAuto=Auto[scram,] # Scramble the data set
# Define the "scaled" newAuto set.
# Remove 8th column(origin), 9th column (name) and 10th column (origin)
newAutoS=scale(newAuto[,-c(8,9,10)])
trn=rep(TRUE,300)  # Use first 300 rows of newAuto as training set
tst=rep(FALSE,92) # Use last 92 rows of newAuto as test set.
train=c(trn,tst) # Put trn and tst together to form train.
train.X=newAutoS[train,c(2,3,5)]
test.X=newAutoS[!train,c(2,3,5)]
# NewAuto (without the S), since newAutoS does not have FromUS in it!
train.origin=newAuto[train,8]
test.origin=newAuto[!train,8]
knnMod3=knn(train.X, test.X, train.origin, k = 2)
knnMod3
table(knnMod3,test.origin)
```

Seed 0, K=10

```
Auto = read.csv(file="Auto.csv",head=TRUE,stringsAsFactors=FALSE, na.strings="?")
Auto = na.omit(Auto)
dim(Auto)
```

```
library(class)
displacement=Auto$displacement
cylinders=Auto$cylinders
horsepower=Auto$horsepower
weight=Auto$weight
mpg=Auto$mpg
origin=Auto$origin

origin=rep(0,392) #Set up a vector of 392 zeroes
origin[Auto$origin==1]=1 #Switch value fro US cars
Auto=data.frame(Auto,origin) #Add new variable to Auto

set.seed(0)
scram=sample.int(392, 392) # Randomly scramble 1 through 392
newAuto=Auto[scram,] # Scramble the data set
# Define the "scaled" newAuto set.
# Remove 8th column(origin), 9th column (name) and 10th column (origin)
newAutoS=scale(newAuto[,-c(8,9,10)])
trn=rep(TRUE,300)  # Use first 300 rows of newAuto as training set
tst=rep(FALSE,92) # Use last 92 rows of newAuto as test set.
train=c(trn,tst) # Put trn and tst together to form train.
train.X=newAutoS[train,c(2,3,5)]
test.X=newAutoS[!train,c(2,3,5)]
# NewAuto (without the S), since newAutoS does not have FromUS in it!
train.origin=newAuto[train,8]
test.origin=newAuto[!train,8]
knnMod3=knn(train.X, test.X, train.origin, k = 10)
knnMod3
table(knnMod3,test.origin)
```

Seed 3, K=2

```
Auto = read.csv(file="Auto.csv",head=TRUE,stringsAsFactors=FALSE, na.strings="?")
Auto = na.omit(Auto)
dim(Auto)
library(class)
displacement=Auto$displacement
cylinders=Auto$cylinders
horsepower=Auto$horsepower
weight=Auto$weight
mpg=Auto$mpg
origin=Auto$origin

origin=rep(0,392) #Set up a vector of 392 zeroes
origin[Auto$origin==1]=1 #Switch value fro US cars
Auto=data.frame(Auto,origin) #Add new variable to Auto

set.seed(3)
```

```
scram=sample.int(392, 392) # Randomly scramble 1 through 392
newAuto=Auto[scram,] # Scramble the data set
# Define the "scaled" newAuto set.
# Remove 8th column(origin), 9th column (name) and 10th column (origin)
newAutoS=scale(newAuto[,-c(8,9,10)])
trn=rep(TRUE,300)  # Use first 300 rows of newAuto as training set
tst=rep(FALSE,92) # Use last 92 rows of newAuto as test set.
train=c(trn,tst) # Put trn and tst together to form train.
train.X=newAutoS[train,c(2,3,5)]
test.X=newAutoS[!train,c(2,3,5)]
# NewAuto (without the S), since newAutoS does not have FromUS in it!
train.origin=newAuto[train,8]
test.origin=newAuto[!train,8]
knnMod3=knn(train.X, test.X, train.origin, k = 2)
knnMod3
table(knnMod3,test.origin)
```

Seed 3, K=10

```
Auto = read.csv(file="Auto.csv",head=TRUE,stringsAsFactors=FALSE, na.strings="?")
Auto = na.omit(Auto)
dim(Auto)
library(class)
displacement=Auto$displacement
cylinders=Auto$cylinders
horsepower=Auto$horsepower
weight=Auto$weight
mpg=Auto$mpg
origin=Auto$origin

origin=rep(0,392) #Set up a vector of 392 zeroes
origin[Auto$origin==1]=1 #Switch value fro US cars
Auto=data.frame(Auto,origin) #Add new variable to Auto

set.seed(3)
scram=sample.int(392, 392) # Randomly scramble 1 through 392
newAuto=Auto[scram,] # Scramble the data set
# Define the "scaled" newAuto set.
# Remove 8th column(origin), 9th column (name) and 10th column (origin)
newAutoS=scale(newAuto[,-c(8,9,10)])
trn=rep(TRUE,300)  # Use first 300 rows of newAuto as training set
tst=rep(FALSE,92) # Use last 92 rows of newAuto as test set.
train=c(trn,tst) # Put trn and tst together to form train.
train.X=newAutoS[train,c(2,3,5)]
test.X=newAutoS[!train,c(2,3,5)]
# NewAuto (without the S), since newAutoS does not have FromUS in it!
train.origin=newAuto[train,8]
test.origin=newAuto[!train,8]
```

```
knnMod3=knn(train.X, test.X, train.origin, k = 10)
knnMod3
table(knnMod3,test.origin)
```

Seed	K	Confusion matrix	Success Rate
0	2	test.origin knnMod3 1 2 3 1 49 4 4 2 0 11 5 3 2 4 13	0.7934 = 79.34%
0	10	test.origin knnMod3 1 2 3 1 48 7 4 2 1 4 4 3 2 8 14	0.7173 = 71.73%
3	2	test.origin knnMod3 1 2 3 1 50 3 2 2 2 11 9 3 6 3 6	0.7282 = 72.82%
3	10	test.origin knnMod3 1 2 3 1 51 6 1 2 3 4 3 3 4 7 13	0.6304 = 63.04%

KNN Model 3: origin~cylinders+displacement+weight (standardized)