homework3

Sunny Lee

2/14/2021

1)

```
library("MASS")
data("Boston")
```

2)

```
?Boston
```

```
## starting httpd help server ... done
```

In this dataset, crim refers to the per capita crime rate by town.

3)

```
crim01 <- rep(0, length(Boston$crim))
crim01[Boston$crim > median(Boston$crim)] <- 1
Boston <- data.frame(Boston, crim01)
```

4)

```
cor(Boston)
```

```
##                crim          zn       indus         chas         nox
## crim     1.00000000 -0.20046922  0.40658341 -0.055891582  0.42097171
## zn      -0.20046922  1.00000000 -0.53382819 -0.042696719 -0.51660371
## indus    0.40658341 -0.53382819  1.00000000  0.062938027  0.76365145
## chas    -0.05589158 -0.04269672  0.06293803  1.000000000  0.09120281
## nox      0.42097171 -0.51660371  0.76365145  0.091202807  1.00000000
## rm      -0.21924670  0.31199059 -0.39167585  0.091251225 -0.30218819
## age      0.35273425 -0.56953734  0.64477851  0.086517774  0.73147010
## dis     -0.37967009  0.66440822 -0.70802699 -0.099175780 -0.76923011
## rad      0.62550515 -0.31194783  0.59512927 -0.007368241  0.61144056
## tax      0.58276431 -0.31456332  0.72076018 -0.035586518  0.66802320
## ptratio  0.28994558 -0.39167855  0.38324756 -0.121515174  0.18893268
## black   -0.38506394  0.17552032 -0.35697654  0.048788485 -0.38005064
## lstat    0.45562148 -0.41299457  0.60379972 -0.053929298  0.59087892
## medv    -0.38830461  0.36044534 -0.48372516  0.175260177 -0.42732077
## crim01   0.40939545 -0.43615103  0.60326017  0.070096774  0.72323480
##                rm         age         dis          rad         tax    ptratio
## crim    -0.21924670  0.35273425 -0.37967009  0.625505145  0.58276431  0.2899456
## zn       0.31199059 -0.56953734  0.66440822 -0.311947826 -0.31456332 -0.3916785
## indus   -0.39167585  0.64477851 -0.70802699  0.595129275  0.72076018  0.3832476
## chas     0.09125123  0.08651777 -0.09917578 -0.007368241 -0.03558652 -0.1215152
## nox     -0.30218819  0.73147010 -0.76923011  0.611440563  0.66802320  0.1889327
## rm       1.00000000 -0.24026493  0.20524621 -0.209846668 -0.29204783 -0.3555015
## age     -0.24026493  1.00000000 -0.74788054  0.456022452  0.50645559  0.2615150
```

```
## dis      0.20524621 -0.74788054  1.00000000 -0.494587930 -0.53443158 -0.2324705
## rad     -0.20984667  0.45602245 -0.49458793  1.000000000  0.91022819  0.4647412
## tax     -0.29204783  0.50645559 -0.53443158  0.910228189  1.00000000  0.4608530
## ptratio -0.35550149  0.26151501 -0.23247054  0.464741179  0.46085304  1.0000000
## black    0.12806864 -0.27353398  0.29151167 -0.444412816 -0.44180801 -0.1773833
## lstat   -0.61380827  0.60233853 -0.49699583  0.488676335  0.54399341  0.3740443
## medv     0.69535995 -0.37695457  0.24992873 -0.381626231 -0.46853593 -0.5077867
## crim01  -0.15637178  0.61393992 -0.61634164  0.619786249  0.60874128  0.2535684
##               black      lstat       medv      crim01
## crim    -0.38506394  0.4556215 -0.3883046  0.40939545
## zn       0.17552032 -0.4129946  0.3604453 -0.43615103
## indus   -0.35697654  0.6037997 -0.4837252  0.60326017
## chas     0.04878848 -0.0539293  0.1752602  0.07009677
## nox     -0.38005064  0.5908789 -0.4273208  0.72323480
## rm       0.12806864 -0.6138083  0.6953599 -0.15637178
## age     -0.27353398  0.6023385 -0.3769546  0.61393992
## dis      0.29151167 -0.4969958  0.2499287 -0.61634164
## rad     -0.44441282  0.4886763 -0.3816262  0.61978625
## tax     -0.44180801  0.5439934 -0.4685359  0.60874128
## ptratio -0.17738330  0.3740443 -0.5077867  0.25356836
## black    1.00000000 -0.3660869  0.3334608 -0.35121093
## lstat   -0.36608690  1.0000000 -0.7376627  0.45326273
## medv     0.33346082 -0.7376627  1.0000000 -0.26301673
## crim01  -0.35121093  0.4532627 -0.2630167  1.00000000
```

```
tail(sort(abs(cor(Boston)[, 15])), 6)
```

```
##       tax       age       dis       rad       nox    crim01
## 0.6087413 0.6139399 0.6163416 0.6197862 0.7232348 1.0000000
```

Ignoring the crim01 variable, the five strongest predictors are tax, age, dis, rad, nox

5)

```
set.seed(5492)
v <- sort(sample(1:nrow(Boston),100)) # this creates a random selection of 100 numbers from 1 to n
Boston.test<-Boston[v,]
Boston.train<-Boston[-v,]
```

6)

```
logreg.model <- glm(crim01~tax+age+dis+rad+nox, data = Boston.train, family = binomial)

logreg.prob <- predict(logreg.model, Boston.test, type = "response")
logreg.predict <- rep(0, length(logreg.prob))
logreg.predict[logreg.prob > .5] <- 1

table(logreg.predict, Boston.test$crim01)
```

```
##
## logreg.predict  0  1
##              0 46 14
##              1  2 38
```

```
mean(logreg.predict == Boston.test$crim01)
```

```
## [1] 0.84
```

```
mean(logreg.predict != Boston.test$crim01)
```

```
## [1] 0.16
```

We find that the test error rate is 16 percent, and our test accuracy of our logisitc regression model is 84 percent

7)

```
lda.model <- lda(crim01~tax+age+dis+rad+nox, data = Boston.train)
lda.pred = predict(lda.model, Boston.test)

lda.class <- lda.pred$class

table(lda.class, Boston.test$crim01)
```

```
##
## lda.class  0  1
##         0 48 17
##         1  0 35
```

```
mean(lda.class == Boston.test$crim01)
```

```
## [1] 0.83
```

```
mean(lda.class != Boston.test$crim01)
```

```
## [1] 0.17
```

We find that the test error rate is 17 percent, and our test accuracy of our lda model is 83 percent.

8)

```
library(class)
set.seed(5492)
#v <- sort(sample(1:nrow(std.boston),100)) # this creates a random selection of 100 numbers from 1 to n
#std.boston.test<-Boston[v,]
#std.boston.train<-Boston[-v,]

train.X <- cbind(Boston.train$tax, Boston.train$age, Boston.train$dis, Boston.train$rad, Boston.train$n
test.X <- cbind(Boston.test$tax, Boston.test$age, Boston.test$dis, Boston.test$rad, Boston.test$nox)
train.Y <- Boston.train$crim01

knn.pred <- knn(train.X, test.X, train.Y, k = 1)
table(knn.pred, Boston.test$crim01)
```

```
##
## knn.pred  0  1
##        0 45  5
##        1  3 47
```

```
mean(knn.pred == Boston.test$crim01)
```

```
## [1] 0.92
```

```
mean(knn.pred != Boston.test$crim01)
```

```
## [1] 0.08
```

We find that hte test error rate is 8 percent and our test accuracy of our knn model is 92 percent.