STAT 1361 - Homework 4

Gordon Lu

2/18/2021

Exercise 2:

ISLR Conceptual Exercise 2

a)

If X is uniformly distributed, then (0.65-0.55)/(1-0) = 10%.

b)

For two features, $10\% \times 10\% = 1\%$.

c)

On average, $0.10^{100} \times 100 = 10^{-98}\%$

d)

When there are a large number of dimensions, the percentage of observations that can be used to predict with KNN becomes very small. This means that for a set sample size, more features leads to fewer neighbors.

e)

$$p = 1, l = 0.10$$
$$p = 2, l = \sqrt{(0.10)} \times 0.32$$
$$p = 3, l = 0.10^{1/3} \times 0.46$$
$$...$$
$$p = N, l = 0.10^{1/N}$$

When p = 1, length = 0.1

When p = 2, length = $0.1^{1/2} = 0.316$

When p = 100, length = $0.1^{1/100} = 0.977$.

When the number of features is high (i.e. p=100), to use on average 10% of the training observations would mean that we would need to include almost the entire range of each individual feature.

2b)

In exercise 4, we were observing KNN. To argue against the statemnt, "non-parametric approaches often perform poorly when p is large," we can make the claim that if a hypercube's dimensions increase as p increases, the classfier should be able to predict samples efficient, regardless of the size of p. Generally for non-parametric appraoches, if there are a lot of features, but few samples, it will be difficult to extrapolate predictions based on nearby samples. However, if there are a lot of samples, a test observation should be able to extrapolate from nearby previously observed samples, which is not the case with few samples. Thus, the generalization is not necessarily always true.

2c)

In high-dimensional space, you are often forced to **overfit**.

2d)

There are two valid answers to this question. Generally, more data is not a bad thing. This follows from the idea of distributions and the Central Limit Theorem; the more data you have, the better you can observe the underlying distribution of the data. As such, you can create a model that fits the data even better. On the other hand, the more data that enters the model, the higher the probability is to introduce random noise into the dataset. In turn, this can mess up models, damage inferential procedures and accuracies. Generally, sticking to the first approach is a good rule of thumb, but it always has the potential to negatively impact the modeling.

Exercise 3:

ISLR Conceptual Exercise 5

5a)

With a linear Bayes decision boundary, we expect QDA to be better than LDA for the training set. QDA is generally more flexible than LDA, so it will likely have a closer/better fit than LDA. On the test set, LDA is expected to be better than QDA, since there is the risk of overfitting the linearity on the decision boundary.

5b)

Since the study has a non-linear decision boundary, QDA is expected to be better than LDA on both the training and testing sets, since there is a lower risk of overfitting to data with the new assumption.

5c)

As the sample size increases, we expect the test prediction accuracy of QDA to improve relative to LDA, since the variance of data is less of a concern, so overfitting in turn is also less of a concern. QDA is generally more flexible and has more variance than LDA.

5d)

False. When the sample size is small, QDA may lead to overfitting, leading to a lower test accuracy.

ISLR Conceptual Exercise 8

8) In KNN, with K = 1, the training error rate is 0%. Since the average of training error and testing error is 18%, (0% + x%)/2 = 18%, where x is the testing error percentage. We can find the test error to be 36%, which is higher than the logistic regression testing error. As such, we can explain this KNN situation as overfitting. Thus, we want to use logistic regression.

Exercise 4:

ISLR Applied Exercise 11

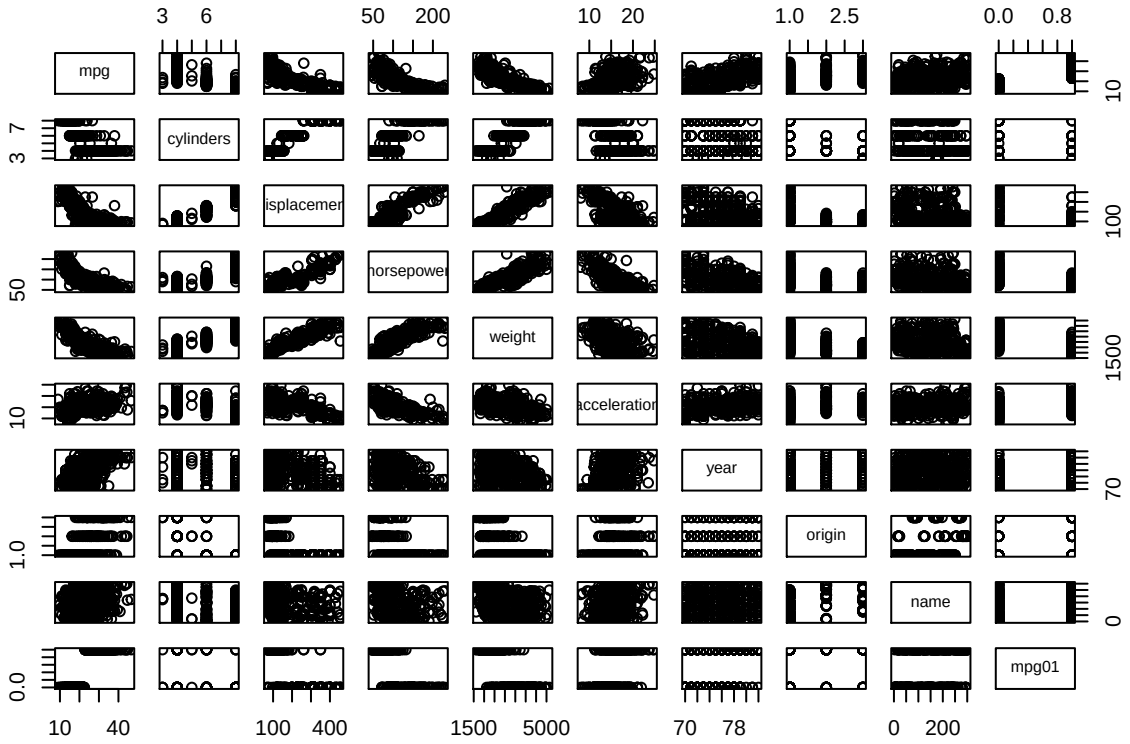
11a)

```
auto <- read.csv(file = 'C:/Users/gordo/Desktop/Auto.csv')
mpg01 <- rep(0, length(auto$mpg))
mpg01[auto$mpg > median(auto$mpg)] <- 1
auto <- data.frame(auto, mpg01)
auto
```

```
cor(auto[,-9])
```

```
##                     mpg  cylinders displacement horsepower     weight
## mpg           1.0000000 -0.7776175   -0.8051269 -0.7784268 -0.8322442
## cylinders    -0.7776175  1.0000000    0.9508233  0.8429834  0.8975273
## displacement -0.8051269  0.9508233    1.0000000  0.8972570  0.9329944
## horsepower   -0.7784268  0.8429834    0.8972570  1.0000000  0.8645377
## weight       -0.8322442  0.8975273    0.9329944  0.8645377  1.0000000
## acceleration  0.4233285 -0.5046834   -0.5438005 -0.6891955 -0.4168392
## year          0.5805410 -0.3456474   -0.3698552 -0.4163615 -0.3091199
## origin        0.5652088 -0.5689316   -0.6145351 -0.4551715 -0.5850054
## mpg01         0.8369392 -0.7591939   -0.7534766 -0.6670526 -0.7577566
##              acceleration       year     origin      mpg01
## mpg             0.4233285  0.5805410  0.5652088  0.8369392
## cylinders      -0.5046834 -0.3456474 -0.5689316 -0.7591939
## displacement   -0.5438005 -0.3698552 -0.6145351 -0.7534766
## horsepower     -0.6891955 -0.4163615 -0.4551715 -0.6670526
## weight         -0.4168392 -0.3091199 -0.5850054 -0.7577566
## acceleration    1.0000000  0.2903161  0.2127458  0.3468215
## year            0.2903161  1.0000000  0.1815277  0.4299042
## origin          0.2127458  0.1815277  1.0000000  0.5136984
## mpg01           0.3468215  0.4299042  0.5136984  1.0000000
```

```
pairs(auto)
```



displacement, horsepower, weight, and acceleration seem to be highly correlated with mpg01.

```
train = (auto$year%%2 == 0)
auto.train = auto[train, ]
auto.test = auto[!train, ]
mpg01.test = mpg01[!train]
```

```
library(MASS)
fit.lda <- lda(mpg01 ~ cylinders + weight + displacement
               + horsepower, data = auto, subset = train)
fit.lda
```

```
## Call:
## lda(mpg01 ~ cylinders + weight + displacement + horsepower, data = auto,
##     subset = train)
##
## Prior probabilities of groups:
##         0         1
## 0.4571429 0.5428571
##
## Group means:
##   cylinders   weight displacement horsepower
## 0  6.812500 3604.823     271.7396  133.14583
## 1  4.070175 2314.763     111.6623   77.92105
##
```

```
## Coefficients of linear discriminants:
##                        LD1
## cylinders    -0.6741402638
## weight       -0.0011465750
## displacement  0.0004481325
## horsepower    0.0059035377
```

```
pred.lda <- predict(fit.lda, auto.test)
table(pred.lda$class, mpg01.test)
```

```
##    mpg01.test
##      0  1
##   0 86  9
##   1 14 73
```

```
mean(pred.lda$class != mpg01.test)
```

```
## [1] 0.1263736
```

The test error is: 12.6374%.

11e)

```
fit.qda <- qda(mpg01 ~ cylinders + weight + displacement
               + horsepower, data = auto, subset = train)
fit.qda
```

```
## Call:
## qda(mpg01 ~ cylinders + weight + displacement + horsepower, data = auto,
##     subset = train)
##
## Prior probabilities of groups:
##         0         1
## 0.4571429 0.5428571
##
## Group means:
##   cylinders   weight displacement horsepower
## 0  6.812500 3604.823     271.7396  133.14583
## 1  4.070175 2314.763     111.6623   77.92105
```

```
pred.qda <- predict(fit.qda, auto.test)
table(pred.qda$class, mpg01.test)
```

```
##    mpg01.test
##      0  1
##   0 89 13
##   1 11 69
```

```r
mean(pred.qda$class != mpg01.test)
```

```
## [1] 0.1318681
```

The test error is: 13.1868%.

11f)

```r
fit.glm <- glm(mpg01 ~ cylinders + weight + displacement
               + horsepower, data = auto, family = binomial, subset = train)
fit.glm
```

```
##
## Call:  glm(formula = mpg01 ~ cylinders + weight + displacement + horsepower,
##     family = binomial, data = auto, subset = train)
##
## Coefficients:
##  (Intercept)     cylinders        weight  displacement    horsepower
##    17.658730     -1.028032     -0.002922      0.002462     -0.050611
##
## Degrees of Freedom: 209 Total (i.e. Null);  205 Residual
## Null Deviance:       289.6
## Residual Deviance: 83.24     AIC: 93.24
```

```r
probs <- predict(fit.glm, auto.test, type = "response")
pred.glm <- rep(0, length(probs))
pred.glm[probs > 0.5] <- 1
table(pred.glm, mpg01.test)
```

```
##         mpg01.test
## pred.glm  0  1
##        0 89 11
##        1 11 71
```

```r
mean(pred.glm != mpg01.test)
```

```
## [1] 0.1208791
```

The test error is: 12.0879%.

11g)

```r
library(class)
training <- cbind(auto$cylinders, auto$weight, auto$displacement, auto$horsepower)[train,]
testing <- cbind(auto$cylinders, auto$weight, auto$displacement, auto$horsepower)[!train,]
train.mpg01 <- mpg01[train]
set.seed(1)
pred.knn <- knn(training, testing, train.mpg01, k = 1)
table(pred.knn, mpg01.test)
```

```
##          mpg01.test
## pred.knn  0  1
##        0 83 11
##        1 17 71
```

```
mean(pred.knn != mpg01.test)
```

```
## [1] 0.1538462
```

```
pred.knn <- knn(training, testing, train.mpg01, k = 5)
table(pred.knn, mpg01.test)
```

```
##          mpg01.test
## pred.knn  0  1
##        0 82  9
##        1 18 73
```

```
mean(pred.knn != mpg01.test)
```

```
## [1] 0.1483516
```

```
pred.knn <- knn(training, testing, train.mpg01, k = 25)
table(pred.knn, mpg01.test)
```

```
##          mpg01.test
## pred.knn  0  1
##        0 83  9
##        1 17 73
```

```
mean(pred.knn != mpg01.test)
```

```
## [1] 0.1428571
```

When K = 1, the test error is 15.3846%.

When K = 5, the test error is 14.8352%.

When K = 25, the test error is 14.2857%.

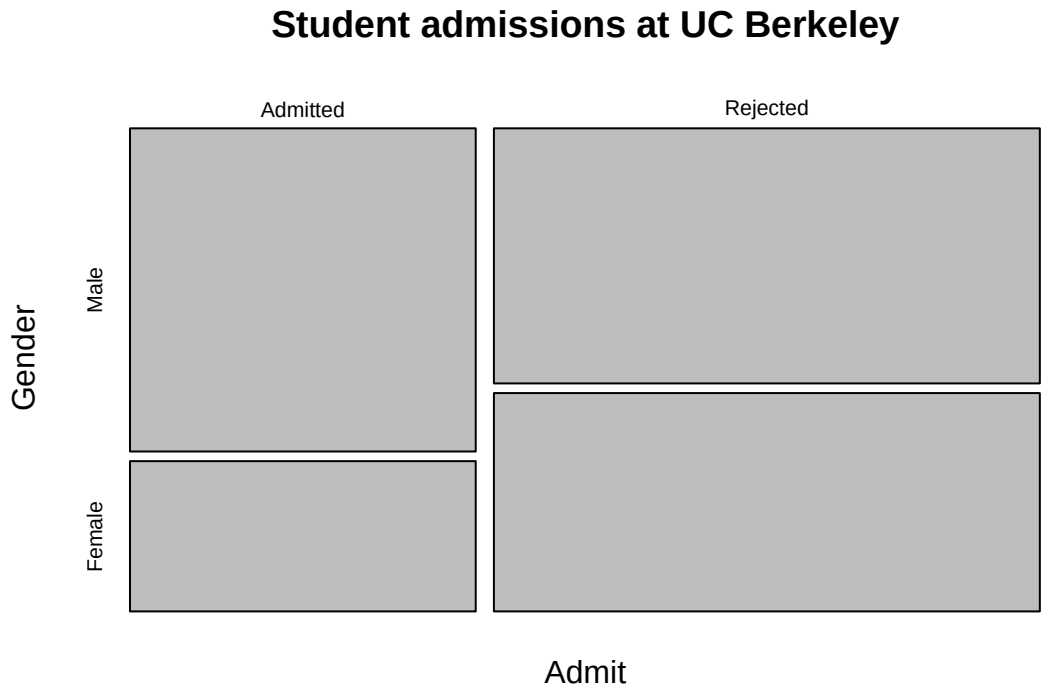
Thus, K = 25 performs the best among K = 1, 5, and 25 on this dataset.

Exercise 5)

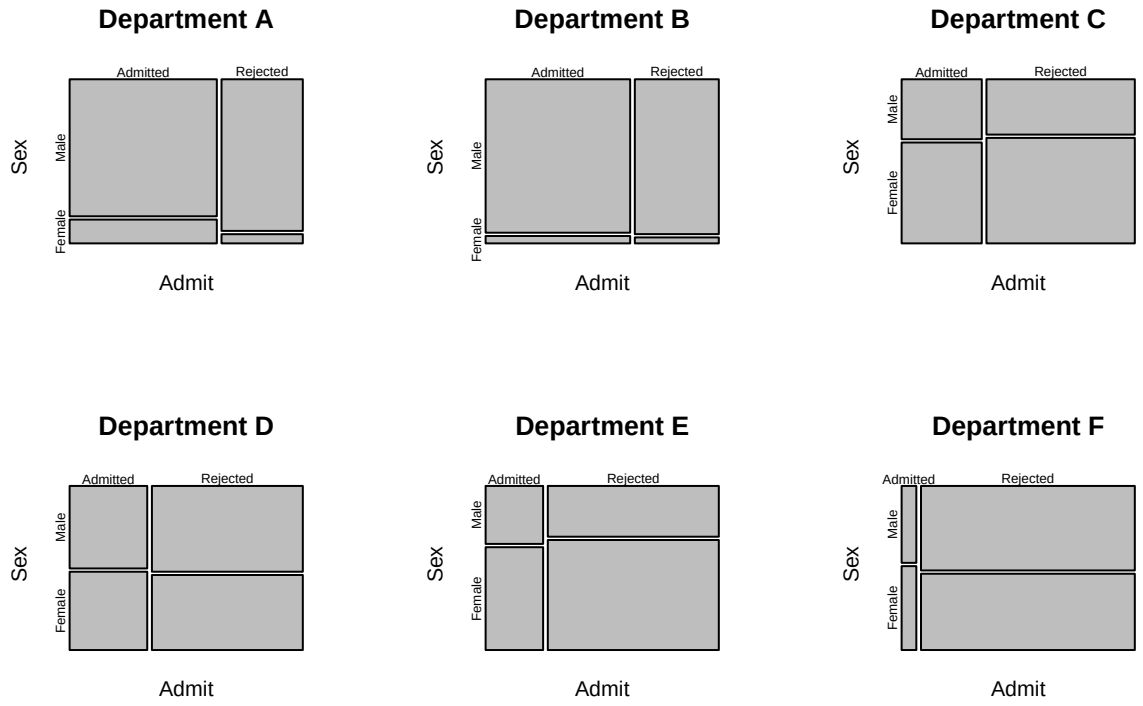
5a)

```
example(UCBAdmissions)
```

```
##
## UCBAdm> require(graphics)
##
## UCBAdm> ## Data aggregated over departments
## UCBAdm> apply(UCBAdmissions, c(1, 2), sum)
##           Gender
## Admit      Male Female
##   Admitted 1198    557
##   Rejected 1493   1278
##
## UCBAdm> mosaicplot(apply(UCBAdmissions, c(1, 2), sum),
## UCBAdm+            main = "Student admissions at UC Berkeley")
```




```
##
## UCBAdm> ## Data for individual departments
## UCBAdm> opar <- par(mfrow = c(2, 3), oma = c(0, 0, 2, 0))
##
## UCBAdm> for(i in 1:6)
## UCBAdm+   mosaicplot(UCBAdmissions[,,i],
## UCBAdm+     xlab = "Admit", ylab = "Sex",
## UCBAdm+     main = paste("Department", LETTERS[i]))
```




```
##
## UCBAdm> mtext(expression(bold("Student admissions at UC Berkeley")),
## UCBAdm+       outer = TRUE, cex = 1.5)
##
## UCBAdm> par(opar)
```

The first plot implies that males make up a significantly higher proportion of the admitted students compares to females. Meanwhile, males and females make up about the same proportion of rejected (50/50 split). Therefore, yes, there seems to be a bias towards males being admitted more than females. The overall percentage of men that were accepted is 44.5188%, while the percentage for woman is 30.3542%.

5b)

The 6 plots show that none of the departments are inherently biased. By comparing the admitted box splits to the rejected box splits, this is evident. If there were a serious imbalance when comparing admitted to rejected, then there would be bias similar to (a). However, the six plots do not show such a trend.

5c)

The general idea of Simpson's Paradox is that a specific trend appears when there is a lot of data aggregated, but then completely disappears when the data is split into their respective subgroups.

5d)

This can be explained by claiming females applied to extremely competitive departments that inherently have low acceptance rates, while men typically applied to less competitive departments with higher acceptance rates. Thus, more men ended up getting into their desired departments anways, increasing the number of overall mean accepted compared to females.

5e)

```
data(UCBAdmissions)
Adm <- as.integer(UCBAdmissions)[(1:(6*2))*2-1]
Rej <- as.integer(UCBAdmissions)[(1:(6*2))*2]
Dept <- gl(6,2,6*2,labels=c("A","B","C","D","E","F"))
Sex <- gl(2,1,6*2,labels=c("Male","Female"))
Ratio <- Adm/(Rej+Adm)
berk <- data.frame(Adm,Rej,Sex,Dept,Ratio)
LogReg.gender <- glm(cbind(Adm,Rej)~Sex,data=berk,family=binomial("logit"))
summary(LogReg.gender)
```

```
##
## Call:
## glm(formula = cbind(Adm, Rej) ~ Sex, family = binomial("logit"),
##     data = berk)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -16.7915   -4.7613   -0.4365    5.1025   11.2022
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.22013    0.03879  -5.675 1.38e-08 ***
## SexFemale   -0.61035    0.06389  -9.553  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 877.06  on 11  degrees of freedom
## Residual deviance: 783.61  on 10  degrees of freedom
## AIC: 856.55
##
## Number of Fisher Scoring iterations: 4
```

We can say the admission rate of females is statistically significant with p-vale $<$ 2e-16. Thus, there is in fact a bias against females.

2f)

```
LogReg.gender <- glm(cbind(Adm,Rej)~Sex+Dept,data=berk,family=binomial("logit"))
summary(LogReg.gender)
```

```
##
## Call:
## glm(formula = cbind(Adm, Rej) ~ Sex + Dept, family = binomial("logit"),
##     data = berk)
##
## Deviance Residuals:
##       1        2        3        4        5        6        7        8
## -1.2487   3.7189  -0.0560   0.2706   1.2533  -0.9243   0.0826  -0.0858
##       9       10       11       12
##  1.2205  -0.8509  -0.2076   0.2052
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.58205    0.06899   8.436   <2e-16 ***
## SexFemale    0.09987    0.08085   1.235    0.217
## DeptB       -0.04340    0.10984  -0.395    0.693
## DeptC       -1.26260    0.10663 -11.841   <2e-16 ***
## DeptD       -1.29461    0.10582 -12.234   <2e-16 ***
## DeptE       -1.73931    0.12611 -13.792   <2e-16 ***
## DeptF       -3.30648    0.16998 -19.452   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 877.056  on 11  degrees of freedom
## Residual deviance:  20.204  on  5  degrees of freedom
## AIC: 103.14
##
## Number of Fisher Scoring iterations: 4
```

Once departments are taken into consideration, we can see that the female acceptance rate is no longer statistically significant, so there is no bias against females. The coefficient for females also dropped from -0.6104 to 0.0999, which is closer to 0 compared to (e), making it less of a factor.

Overall, we showed Simpson's Paradox, which is the idea that aggregated data showing specific trend, but subgroups of the data not showing such trends. Thus, though the data may seem biased, it may actually be balanced, and further inference of the data can lead to a high-level reaosning about this paradox. One can discover the root of the biases by either plotting the subgroups as in (a) and (b), or by fitting a model to the data, and performing hypothesis testing on the predictors to determine whether they are statistically significant.