STAT 435 HW3

Chongyi Xu

April 19, 2018

Question 1

A random variable X has an Exp(λ) distribution if its probability density function is of the form

$$f(x) = \begin{cases} \lambda e^{-\lambda x} & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}$$

where $\lambda > 0$ is a parameter. Furthermore, the mean of an Exp(λ) random varibale is $1/\lambda$.

Now consider a classification problem with $K = 2$ classes and a single feature $X \in \mathbb{R}$. If an observation is in class 1, then $X \sim Exp(\lambda_1)$. And if an observation is in class 2, then $X \sim Exp(\lambda_2)$. Let π_1 denote the probability that an observation is in class 1, and let $\pi_2 = 1 - \pi_1$.

(a) Derive an expression for $Pr(Y = 1|X = x)$.

$$\begin{aligned} Pr(Y = 1|X = x) &= \frac{\pi_1 f_1(x)}{\sum_{i=1}^{K=2} \pi_i f_i(x)} \\ &= \frac{\pi_1 \lambda_1 e^{-\lambda_1 x}}{\pi_1 \lambda_1 e^{-\lambda_1 x} + \pi_2 \lambda_2 e^{-\lambda_2 x}} \end{aligned}$$

(b) Write a simple expression for the Bayes classifier decision boundary.

The classifier decision boundary is the set of x such that $Pr(Y = 1|X = x) = Pr(Y = 2|X = x)$.

$$\begin{aligned} Pr(Y = 1|X = x) &= Pr(Y = 2|X = x) \\ \frac{\pi_1 \lambda_1 e^{-\lambda_1 x}}{\pi_1 \lambda_1 e^{-\lambda_1 x} + \pi_2 \lambda_2 e^{-\lambda_2 x}} &= \frac{\pi_2 \lambda_2 e^{-\lambda_2 x}}{\pi_1 \lambda_1 e^{-\lambda_1 x} + \pi_2 \lambda_2 e^{-\lambda_2 x}} \\ \pi_1 \lambda_1 e^{-\lambda_1 x} &= \pi_2 \lambda_2 e^{-\lambda_2 x} \\ e^{(\lambda_1 - \lambda_2)x} &= \frac{\pi_1 \lambda_1}{\pi_2 \lambda_2} \\ (\lambda_1 - \lambda_2)x &= ln(\pi_1 \lambda_1) - ln(\pi_2 \lambda_2) \\ x &= \frac{ln(\pi_1 \lambda_1) - ln(\pi_2 \lambda_2)}{\lambda_1 - \lambda_2} \end{aligned}$$

(c) Suppose $\lambda_1 = 2, \lambda_2 = 7, \pi_1 = 0.5$. Make a plot of feature space.

```
library(ggplot2)
```

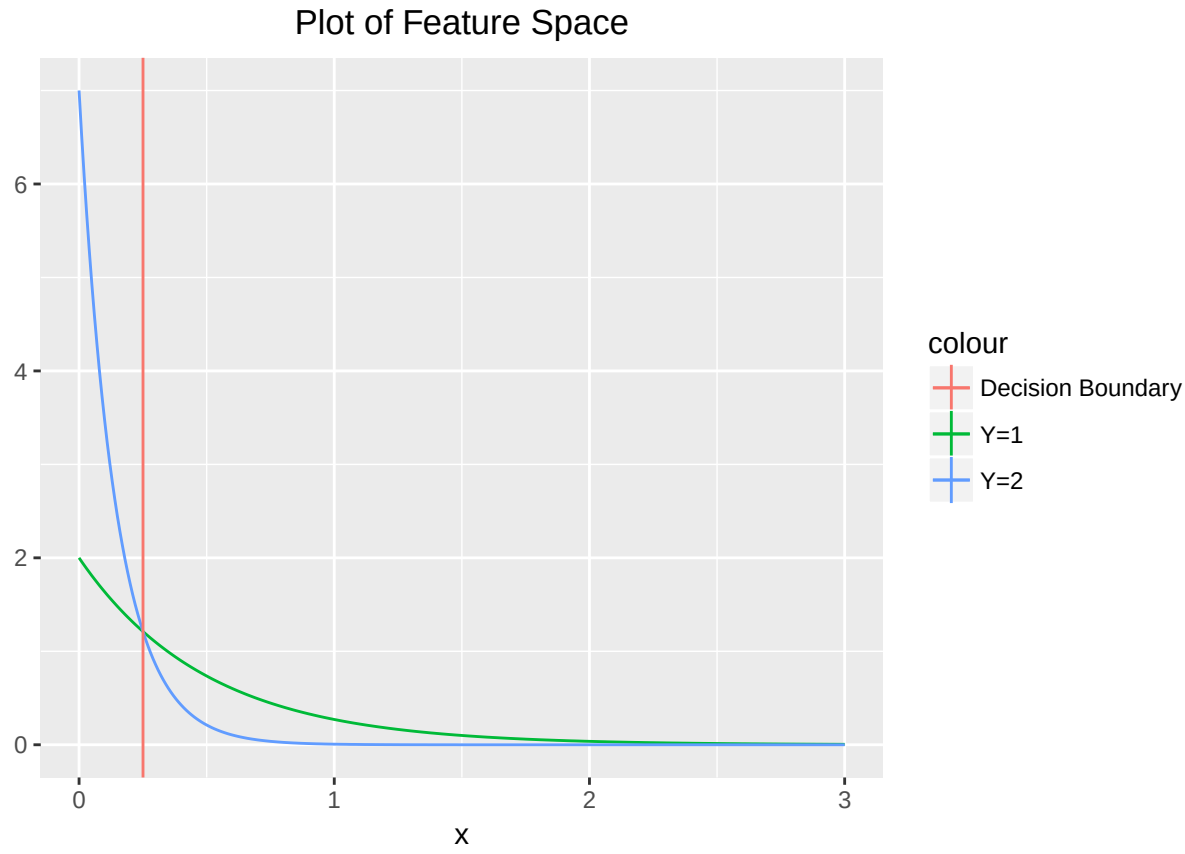
```
## Warning: package 'ggplot2' was built under R version 3.4.4
```

```
lambda_1 <- 2
lambda_2 <- 7
pi_1 <- 0.5
pi_2 <- 1 - pi_1

xx <- seq(from=0, to=3, by=0.01)
boundary <- (log(pi_1*lambda_1) - log(pi_2*lambda_2))/(lambda_1-lambda_2)
```

```r
x1 <- dexp(xx, rate=lambda_1)
x2 <- dexp(xx, rate=lambda_2)

ggplot() + geom_line(aes(xx, x1, col='Y=1')) + geom_line(aes(xx, x2, col='Y=2')) + geom_vline(aes(xinte
```



(d) Now suppose that we observe n independent training observations,

$$(x_1, y_1), \cdots, (x_n, y_n)$$

Provide simple estimators for $\lambda_1, \lambda_2, \pi_1, \pi_2$ in terms pf the training observations.

Let $\bar{x_k^1}$ denote as the mean of x_ks where $y_k = 1$ for $k = 1, \cdots, n$. Similarly, denote $\bar{x_k^2}$ as the mean of x_ks where $y_k = 2$ for $k = 1, \cdots, n$. Let $I(y_i = 1)$ be the indicator function that $I(y_i = 1) = 1$ if $y_i = 1$, otherwise, $I(y_i = 1) = 0$ for $i = 1, \cdots, n$. Similarly, let $I(y_i = 2)$ also be the indicator function for checking if $y = 2$

$$\begin{aligned}
\hat{\lambda_1} &= \bar{x_k^1} \\
\hat{\lambda_2} &= \bar{x_k^2} \\
\hat{\pi_1} &= \frac{\sum_{i=1}^{n} I(y_i = 1)}{n} \\
\hat{\pi_2} &= \frac{\sum_{i=1}^{n} I(y_i = 2)}{n}
\end{aligned}$$

(e) Given a test observation $X = x_0$, provide an estimate of

$$P(Y = 1 | X = x_0)$$

Using the simple estimators that I have derived in part (d).

$$\begin{aligned} Pr(Y=1|X=x_0) &= \frac{\hat{\pi_1}\hat{\lambda_1}e^{-\hat{\lambda_1}x_0}}{\hat{\pi_1}\hat{\lambda_1}e^{-\hat{\lambda_1}x_0}+\hat{\pi_2}\hat{\lambda_2}e^{-\hat{\lambda_2}x_0}} \\ &= \frac{\frac{\sum_{i=1}^{n} I(y_i=1)}{n}\bar{x_k^1}\cdot e^{-\bar{x_k^1}x_0}}{\frac{\sum_{i=1}^{n} I(y_i=1)}{n}\bar{x_k^1}\cdot e^{-\bar{x_k^1}x_0}+\frac{\sum_{i=1}^{n} I(y_i=2)}{n}\bar{x_k^2}\cdot e^{-\bar{x_k^2}x_0}} \end{aligned}$$

Question 2

We collect some data for students in a statistics class, with predictor X_1 =number of lectures attended, X_2 =average number of hours studied per week and response Y = receive an A. We fit a logistic regression model, and get coefficient estimates $\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_2$

(a) Write out an expression for the probability taht a student gets an A, as a function of the number of lectures she attended, and the average number of hours she studied per week.

$$Pr(Y=1|X_1,X_2) = \frac{e^{\hat{\beta}_0+\hat{\beta}_1X_1+\hat{\beta}_2X_2}}{1+e^{\hat{\beta}_0+\hat{\beta}_1X_1+\hat{\beta}_2X_2}}$$

(b) Write out an expression for the minimum number of hours a student should study per week in order to have at least an 80% chance of getting an A.

$$\begin{aligned} log(\frac{p(x)}{1-p(x)}) &= \hat{\beta}_0+\hat{\beta}_1X_1+\hat{\beta}_2X_2 \\ log(\frac{0.8}{0.2}) &= \hat{\beta}_0+\hat{\beta}_1X_1+\hat{\beta}_2X_2 \\ log(4) &= \hat{\beta}_0+\hat{\beta}_1X_1+\hat{\beta}_2X_2 \\ X_2 &= \frac{log(4)-\hat{\beta}_0-\hat{\beta}_1X_1}{\hat{\beta}_2} \end{aligned}$$

So the minimum number of hours a student should study per week in order to have at least an 80% chance of getting an A is $\frac{0.8-\hat{\beta}_0-\hat{\beta}_1X_1}{\hat{\beta}_2}$

(c) Based on a student's value of X_1 and X_2, her predicted probability of getting an A in this course is 60%. If she increases her studying by one hour per week, then what will be her predicted probability of getting an A in this course?

$$\begin{aligned} Pr(Y=1|X_1,X_2) &= \frac{e^{\hat{\beta}_0+\hat{\beta}_1X_1+\hat{\beta}_2X_2}}{1+e^{\hat{\beta}_0+\hat{\beta}_1X_1+\hat{\beta}_2X_2}} = 0.6 \\ Pr(Y=1|X_1,X_2+1) &= \frac{e^{\hat{\beta}_0+\hat{\beta}_1X_1+\hat{\beta}_2X_2+\hat{\beta}_2}}{1+e^{\hat{\beta}_0+\hat{\beta}_1X_1+\hat{\beta}_2X_2+\hat{\beta}_2}} \\ \Rightarrow \frac{Pr(Y=1|X_1,X_2+1)}{0.6} &= \frac{\frac{e^{\hat{\beta}_0+\hat{\beta}_1X_1+\hat{\beta}_2X_2+\hat{\beta}_2}}{1+e^{\hat{\beta}_0+\hat{\beta}_1X_1+\hat{\beta}_2X_2+\hat{\beta}_2}}}{\frac{e^{\hat{\beta}_0+\hat{\beta}_1X_1+\hat{\beta}_2X_2}}{1+e^{\hat{\beta}_0+\hat{\beta}_1X_1+\hat{\beta}_2X_2}}} \\ &= e^{\hat{\beta}_2}(\frac{1+e^{\hat{\beta}_0+\hat{\beta}_1X_1+\hat{\beta}_2X_2}}{1+e^{\hat{\beta}_0+\hat{\beta}_1X_1+\hat{\beta}_2X_2+\hat{\beta}_2}}) \\ Pr(Y=1|X_1,X_2+1) &= 0.6e^{\hat{\beta}_2}(\frac{1+e^{\hat{\beta}_0+\hat{\beta}_1X_1+\hat{\beta}_2X_2}}{1+e^{\hat{\beta}_0+\hat{\beta}_1X_1+\hat{\beta}_2X_2+\hat{\beta}_2}}) \end{aligned}$$

Question 3

When the number of features p is large, there tends to be a deterioration in the performance of K-nearest neighbors (KNN) and other approaches that perform prediction using only observations that are near the test observation for whicch a prediction must be made. This phenomenon is known as the curse of dimensionality. We will now investigate this curse.

(a) Suppose that we have a set of observations, each with measurements on $p = 1$ feature, X. We assume that X is uniformly distributed on $[0, 1]$. Asccociated with each observation's response value. Suppose that we wish to predict a test observation's response using only observations that are within 10% of the range of X closet to that test observation. On average, what fraction of available observations will use to make the prediction?

Since $X \sim Uni[0, 1]$, 10% of the range of X will be $(1 - 0) * 10\% = 0.1$, which means that we will look for the observations in the interval of $[X - 0.05, X + 0.05]$. Therefore, the fraction of observations that will be used for making the prediction is

$$\frac{(X + 0.05) - (X - 0.05)}{1 - 0} = \frac{0.1}{1} = 0.1$$

(b) we have a set of observations, each with measurements on $p = 2$ features, X_1 and X_2. We assume that (X_1, X_2) are uniformly distributed on $[0, 1] \times [0, 1]$. We wish to predict a test observation's response using only observations that are within 10% of the range of X_1 and within 10% of the range of X_2 closest to that test observation. On average, what fraction of the available observations will we use to make the prediction?

From part(a), we have concluded that the fraction of observations for single feature that will be used for making the prediction is 0.1. In this part, there are two faetures and our observation is a pair of features. Therefore, the fraction of the available observations will be used is $0.1 \cdot 0.1 = 0.01$

(c) Now suppse that we have a set of observations on $p = 100$ features... What fraction of the available observations will we use to make the prediction?

Similar to part(b), but this time we have $p = 100$ observations. Thus, the fraction will be $0.1^{100} = 1 \cdot 10^{-100}$

(d) Using your anwers to parts(a)-(c), argue that a drawback of KNN when p is large is that there are very few training observations "near" any given test observation.

As we can see in the previous parts, if we only use observations that are close to the test observation, such as the 10% in the previous parts, the fraction of observations that we will use for our prediction will be extremely small as $p \to +\infty$. KNN is an example of such an algorithm that only uses observations that are near the test observation. And as p becomes large, very few training observations will be available to use.

(e) Now suppose that we wish to make a prediction for a test observation by creating a $p-$dimensional hypercube centered around the test observation that contains, on average, 10% of the training observations. For $p = 1, 2, 100$, what is the length of each side of the hypercube?

In order to have an average of 10% of the training observations,

- $p = 1$

$p = 1$ the hypercube will just be a line segment, and in order to get 10% of the training observations, the length of the line will be 0.1

- $p = 2$

When $p = 2$, the hypercube will be a 2D square, and the length will be $\sqrt{0.1} = \frac{\sqrt{10}}{10}$

- $p = 100$

When $p = 100$, the hypercube will be a 100-dimensional hypercube. And in order to let the hypercube contains 10% of the training observations, the length will be $(0.1)^{\frac{1}{100}} \approx 0.98$

Question 4

Pick a data set of your choice. It can be chosen from the `ISLR` package, or it can be another data set that you choose. Choose a binary qualitative variable in your data set to be the response, Y.

```
library(ISLR)
```

```
## Warning: package 'ISLR' was built under R version 3.4.4
```

```
library(MASS)
```

```
## Warning: package 'MASS' was built under R version 3.4.4
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:MASS':
##
##     select
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
dat <- OJ # Purchase is a binary qualitative variable
dat <- dat %>%
  dplyr::select(Purchase, WeekofPurchase, StoreID, DiscCH, DiscMM, LoyalCH, SalePriceCH, SalePriceMM, Sp
dat$StoreID <- as.factor(dat$StoreID)
dat$SpecialCH <- as.factor(dat$SpecialCH)
dat$SpecialMM <- as.factor(dat$SpecialMM)
```

(a) Describe the data. What are the values of n and p? What are you trying to predict?

The data set I will use is the `Orange Juice Data` from `ISLR` pacakge. It originally contains $n = 1070$ observations and $p = 18 - 1 = 17$ features (the one get substract off is the response Y). However, after reading the description and documentation, I decided to reduce the size of the feature since there are some similar variables such as `SalePriceCH` (sale price for CH), `SalePriceMM`(sale price for MM) and `PriceDiff` (sale price of MM less sale price of CH). Obvisouly we could get `PriceDiff` from `SalePriceCH` and `SalePriceMM`.

After reconstructing my data, I finally get a data with $n = 1070$ observations and

```
ncol(dat)
```

```
## [1] 10
```

$p = 10$ features.

For this question, I will predict the response Y, as a binary qualitative variable indicating whether the customer purchased `CH` for Citrus Hill or `MM` for Minute Maid Orange Juice. The features I will use are

- `WeekofPurchase`: Week of purchase
- `StoreID`: The ID of the store
- `DiscCH`: Discount offered for CH
- `DiscMM`: Discount offered for MM

- `LoyalCH`: Customer brand loyalty for CH
- `SalePriceCH`: Sale price for CH
- `SalePriceMM`: Sale price for MM
- `SpecialCH`: Indicator of special on CH
- `SpecialMM`: Indicator of special on MM

(b) Split the data into a training set and a test set. Perform LDA on the training set in order to predict Y using the features. What is the training error of the model obtained? What is the test error?

```
# Make the first 800 observations to be training set.
training_dat <- dat[1:800,]
test_dat <- dat[801:1070,]

# LDA
lda.fit <- lda(data=training_dat, Purchase~.)
lda.fit
```

```
## Call:
## lda(Purchase ~ ., data = training_dat)
##
## Prior probabilities of groups:
##      CH      MM
## 0.62125 0.37875
##
## Group means:
##    WeekofPurchase  StoreID2  StoreID3   StoreID4  StoreID7     DiscCH
## CH       256.7746 0.1931590 0.1388330 0.17706237 0.3843058 0.06503018
## MM       251.8845 0.2673267 0.3333333 0.06930693 0.1914191 0.02521452
##       DiscMM   LoyalCH SalePriceCH SalePriceMM SpecialCH1 SpecialMM1
## CH 0.0950503 0.7339171    1.813239    2.012093 0.16700201  0.1086519
## MM 0.1641584 0.3079179    1.843366    1.901815 0.07920792  0.2310231
##
## Coefficients of linear discriminants:
##                            LD1
## WeekofPurchase -6.014426e-05
## StoreID2        1.672081e-01
## StoreID3        1.388415e-01
## StoreID4       -1.215048e-02
## StoreID7       -1.774807e-01
## DiscCH         -1.328106e-01
## DiscMM         -6.411588e-01
## LoyalCH        -3.915603e+00
## SalePriceCH     1.605996e+00
## SalePriceMM    -1.720532e+00
## SpecialCH1      6.746731e-02
## SpecialMM1      3.439313e-01
```

```
# training error
lda.train_err <- mean(predict(lda.fit, newdata=training_dat)$class!=training_dat$Purchase)

print('Linear Discriminant Analysis:')
```

```
## [1] "Linear Discriminant Analysis:"
```

```r
print(paste('The training error is ', lda.train_err))
```

```
## [1] "The training error is  0.15125"
```

```r
# testing error
lda.test_err <- mean(predict(lda.fit, newdata=test_dat)$class!=test_dat$Purchase)
print(paste('The test error is ', lda.test_err))
```

```
## [1] "The test error is  0.214814814814815"
```

(c) Perform QDA on the training set in order to predict Y using the features. What is the training error? What is the test error?

```r
qda.fit <- qda(data=training_dat, Purchase~.)
qda.fit
```

```
## Call:
## qda(Purchase ~ ., data = training_dat)
##
## Prior probabilities of groups:
##      CH      MM
## 0.62125 0.37875
##
## Group means:
##    WeekofPurchase  StoreID2  StoreID3   StoreID4  StoreID7     DiscCH
## CH       256.7746 0.1931590 0.1388330 0.17706237 0.3843058 0.06503018
## MM       251.8845 0.2673267 0.3333333 0.06930693 0.1914191 0.02521452
##       DiscMM   LoyalCH SalePriceCH SalePriceMM SpecialCH1 SpecialMM1
## CH 0.0950503 0.7339171    1.813239    2.012093 0.16700201  0.1086519
## MM 0.1641584 0.3079179    1.843366    1.901815 0.07920792  0.2310231
```

```r
# training error
qda.train_err <- mean(predict(qda.fit, newdata=training_dat)$class!=training_dat$Purchase)

print('Quadratic Discriminant Analysis:')
```

```
## [1] "Quadratic Discriminant Analysis:"
```

```r
print(paste('The training error is ', qda.train_err))
```

```
## [1] "The training error is  0.17375"
```

```r
# testing error
qda.test_err <- mean(predict(qda.fit, newdata=test_dat)$class!=test_dat$Purchase)
print(paste('The test error is ', qda.test_err))
```

```
## [1] "The test error is  0.207407407407407"
```

(d) Perform logistic regression on the training set in order to predict Y using the features. What is the training error? What is the test error?

```r
glm.fit <- glm(data=training_dat, Purchase~., family=binomial)
glm.fit
```

```
##
## Call:  glm(formula = Purchase ~ ., family = binomial, data = training_dat)
##
## Coefficients:
##    (Intercept)  WeekofPurchase        StoreID2        StoreID3
```

```
##       3.359863       -0.002423        0.280512        0.198016
##       StoreID4        StoreID7          DiscCH          DiscMM
##      -0.061283       -0.501361       -0.494721       -0.846622
##        LoyalCH     SalePriceCH     SalePriceMM      SpecialCH1
##      -6.551745        3.430285       -3.082239        0.045924
##     SpecialMM1
##       0.454821
##
## Degrees of Freedom: 799 Total (i.e. Null);  787 Residual
## Null Deviance:       1062
## Residual Deviance: 585.5     AIC: 611.5
```

```
# training error
train_prediction <- predict(glm.fit, newdata=training_dat,type='response') > 0.5
glm.train_err <- mean(train_prediction!=(training_dat$Purchase=='MM'))

print('Logistic Regression:')
```

```
## [1] "Logistic Regression:"
```

```
print(paste('The training error is ', glm.train_err))
```

```
## [1] "The training error is  0.15125"
```

```
# testing error
test_prediction <- predict(glm.fit, newdata=test_dat,type='response') > 0.5
glm.test_err <- mean(test_prediction!=(test_dat$Purchase=='MM'))
print(paste('The test error is ', glm.test_err))
```

```
## [1] "The test error is  0.207407407407407"
```

(e) Perform KNN on the training set in order to predict Y using the features. What is the training error? What is the test error?

```
library(class)
knn.train <- training_dat[,-1]
knn.test <- test_dat[,-1]
knn.label <- training_dat[,1]
knn.train_prediction <- knn(knn.train, knn.train, knn.label,k=5)
knn.train_err <- mean(knn.train_prediction != knn.label)

print('KNN Classification:')
```

```
## [1] "KNN Classification:"
```

```
print(paste('The training error is ', knn.train_err))
```

```
## [1] "The training error is  0.18375"
```

```
knn.test_prediction <- knn(knn.train, knn.test, knn.label, k=5)
knn.test_err <- mean(knn.test_prediction != test_dat$Purchase)
print(paste('The test error is ', knn.test_err))
```

```
## [1] "The test error is  0.262962962962963"
```

(f) Comment on your results.

```
data.frame(row.names=c('Train Error', 'Test Error'),
           LDA = c(lda.train_err, lda.test_err),
           QDA = c(qda.train_err, qda.test_err),
```

```
           LOGISTIC = c(glm.train_err, glm.test_err),
           KNN = c(knn.train_err, knn.test_err))
```

```
##                   LDA       QDA  LOGISTIC      KNN
## Train Error 0.1512500 0.1737500 0.1512500 0.183750
## Test Error  0.2148148 0.2074074 0.2074074 0.262963
```

From the table above, we could see that for this data set, using the chosen features to predict response of which brand of orange juice the customer will buy has different training error and test error for different classifier. Overall, logistic regression model gives the best training error as LDA does and has the best test error as QDA does. KNN with parameter $k = 5$ perhaps is not a good model for this set up.