HW3

Yifei Chen

November 20, 2019

Honor Code:"The codes and results derived by using these codes constitutemy own work. I have consulted the following resources regarding this assignment: Chao Cheng

1

```
load('C:/Users/yifeichen3/OneDrive/academic/2019 Fall/STA 141A/lung.RData')
test_index = sample(nrow(lung), nrow(lung)*.2)
test_data = lung[test_index, ]
# sort the test_data by index
test_data = test_data[order(as.numeric(row.names(test_data))),]
train_data = lung[-test_index, ]
```

2

```
# (a) Report the class-speci???c means of the predictor variables for the training data.
smoke_years = lung[, 2]
second_hand_years = lung[, 3]
biopsy = as.factor(lung[, 1])
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked _by_ '.GlobalEnv':
##
##     biopsy
```

```
# Use linear discriminant analysis to the training data.
lung.lda = lda(biopsy ~ smoke_years + second_hand_years, train_data)
lung.lda
```

```
## Call:
## lda(biopsy ~ smoke_years + second_hand_years, data = train_data)
##
## Prior probabilities of groups:
##          0          1
## 0.93263473 0.06736527
##
## Group means:
```

```
##   smoke_years second_hand_years
## 0    3.976885          6.608954
## 1    9.988795          5.099338
##
## Coefficients of linear discriminants:
##                          LD1
## smoke_years        0.4113629
## second_hand_years -0.1598333
```

```
# Predict for test data.
lung.lda.pred = predict(lung.lda, test_data)
# (b) Compute theconfusion matrixfor the test data, and the misclassification error rate.
# Create the confusion matrix by tabulating true classes against predicted classes.
lung.lda.conf = table(true = test_data$biopsy, predicted = lung.lda.pred$class)
```

3

```
## (a) Fit a logistic regression model to the training data, using the variablessmoke_yearsandsecond_ha
# use family = binomial with the glm function for a two-class logistic rgression
lung.glm = glm(biopsy ~ smoke_years + second_hand_years, train_data, family = binomial)
# (i) Obtain the estimates and their standard errors for the model parameters.
# overall summary
summary(lung.glm)
```

```
##
## Call:
## glm(formula = biopsy ~ smoke_years + second_hand_years, family = binomial,
##     data = train_data)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -0.99758  -0.14053  -0.02726  -0.00458   2.79308
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -12.1621     2.1623  -5.625 1.86e-08 ***
## smoke_years         1.5683     0.2603   6.025 1.69e-09 ***
## second_hand_years  -0.2397     0.1147  -2.089   0.0367 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 329.68  on 667  degrees of freedom
## Residual deviance: 109.73  on 665  degrees of freedom
## AIC: 115.73
##
## Number of Fisher Scoring iterations: 9
```

```
# (ii) Compute the confusion matrix for the test data, and the misclassi???cation error rate.
# Predict for test data. Use type = "response" to get class probabilities.
```

```r
lung.glm.pred.prob = predict(lung.glm, test_data, type = "response")
lung.glm.pred = (lung.glm.pred.prob > 0.5) + 0
# Create the confusion matrix by tabulating true classes against predicted classes.
lung.glm.conf = table(true = test_data$biopsy, predicted = lung.glm.pred)
lung.glm.conf
```

```
##     predicted
## true   0   1
##    0 158   0
##    1   1   8
```

```r
# (iii) Which is the most relevant predictor for the purpose ofclassification? Justify.
## (b) Fit a logistic regression model to the training data, using the variablesmoke_yearsas a one-dimer
lung.glm2 = glm(biopsy ~ smoke_years, train_data, family = binomial)
# (i) Obtain the estimates and their standard errors for the model parameters.
summary(lung.glm2)
```

```
##
## Call:
## glm(formula = biopsy ~ smoke_years, family = binomial, data = train_data)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -0.79997  -0.14438  -0.02313  -0.00346   2.90507
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -14.5550     2.0553  -7.082 1.42e-12 ***
## smoke_years   1.6982     0.2672   6.355 2.09e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 329.68  on 667  degrees of freedom
## Residual deviance: 114.36  on 666  degrees of freedom
## AIC: 118.36
##
## Number of Fisher Scoring iterations: 9
```

```r
# (ii) Compute the confusion matrix for the test data, and the misclassification error rate.
lung.glm.pred.prob2 = predict(lung.glm2, test_data, type = "response")
lung.glm.pred2 = (lung.glm.pred.prob2 > 0.5) + 0
# Create the confusion matrix by tabulating true classes against predicted classes.
lung.glm.conf2 = table(true = test_data$biopsy, predicted = lung.glm.pred2)
lung.glm.conf2
```

```
##     predicted
## true   0   1
##    0 158   0
##    1   1   8
```

```
# (iii) Compare the results with those in 3(a).  Does your result in 3(b)(ii) support theanswer to 3(a)
```

4

```
# Use knn() from package class for k-nearest neighbors.
library(class)
# Use the kNN classifier with k = 20
lung.knn.20 = knn(
  train = train_data[c("smoke_years", "second_hand_years")], # training data for features used in class
  test = test_data[c("smoke_years", "second_hand_years")], # test data data for features used in classi
  cl = train_data$biopsy, # vector of class labels for training data
  k = 20)
lung.knn.conf.20 = table(true = test_data$biopsy, predicted = lung.knn.20)
lung.knn.conf.20
```

```
##     predicted
## true   0   1
##    0 158   0
##    1   3   6
```

```
# Use the kNN classifier with k = 50
lung.knn.50 = knn(
  train = train_data[c("smoke_years", "second_hand_years")], # training data for features used in class
  test = test_data[c("smoke_years", "second_hand_years")], # test data data for features used in classi
  cl = train_data$biopsy, # vector of class labels for training data
  k = 50)
lung.knn.conf.50 = table(true = test_data$biopsy, predicted = lung.knn.50)
lung.knn.conf.50
```

```
##     predicted
## true   0   1
##    0 158   0
##    1   8   1
```

R Appendix

```
knitr::opts_chunk$set(echo = TRUE)
load('C:/Users/yifeichen3/OneDrive/academic/2019 Fall/STA 141A/lung.RData')
test_index = sample(nrow(lung), nrow(lung)*.2)
test_data = lung[test_index, ]
# sort the test_data by index
test_data = test_data[order(as.numeric(row.names(test_data))),]
train_data = lung[-test_index, ]
# (a) Report the class-speci???c means of the predictor variables for the training data.
smoke_years = lung[, 2]
second_hand_years = lung[, 3]
biopsy = as.factor(lung[, 1])
library(MASS)
```

```
# Use linear discriminant analysis to the training data.
lung.lda = lda(biopsy ~ smoke_years + second_hand_years, train_data)
lung.lda
# Predict for test data.
lung.lda.pred = predict(lung.lda, test_data)
# (b) Compute theconfusion matrixfor the test data, and the misclassification error rate.
# Create the confusion matrix by tabulating true classes against predicted classes.
lung.lda.conf = table(true = test_data$biopsy, predicted = lung.lda.pred$class)
## (a) Fit a logistic regression model to the training data, using the variablessmoke_yearsandsecond_han
# use family = binomial with the glm function for a two-class logistic rgression
lung.glm = glm(biopsy ~ smoke_years + second_hand_years, train_data, family = binomial)
# (i) Obtain the estimates and their standard errors for the model parameters.
# overall summary
summary(lung.glm)
# (ii) Compute the confusion matrix for the test data, and the misclassi???cation error rate.
# Predict for test data. Use type = "response" to get class probabilities.
lung.glm.pred.prob = predict(lung.glm, test_data, type = "response")
lung.glm.pred = (lung.glm.pred.prob > 0.5) + 0
# Create the confusion matrix by tabulating true classes against predicted classes.
lung.glm.conf = table(true = test_data$biopsy, predicted = lung.glm.pred)
lung.glm.conf
# (iii) Which is the most relevant predictor for the purpose ofclassification? Justify.
## (b) Fit a logistic regression model to the training data, using the variablesmoke_yearsas a one-dimen
lung.glm2 = glm(biopsy ~ smoke_years, train_data, family = binomial)
# (i) Obtain the estimates and their standard errors for the model parameters.
summary(lung.glm2)
# (ii) Compute the confusion matrix for the test data, and the misclassification error rate.
lung.glm.pred.prob2 = predict(lung.glm2, test_data, type = "response")
lung.glm.pred2 = (lung.glm.pred.prob2 > 0.5) + 0
# Create the confusion matrix by tabulating true classes against predicted classes.
lung.glm.conf2 = table(true = test_data$biopsy, predicted = lung.glm.pred2)
lung.glm.conf2
# (iii) Compare the results with those in 3(a).  Does your result in 3(b)(ii) support theanswer to 3(a)
# Use knn() from package class for k-nearest neighbors.
library(class)
# Use the kNN classifier with k = 20
lung.knn.20 = knn(
  train = train_data[c("smoke_years", "second_hand_years")], # training data for features used in class
  test = test_data[c("smoke_years", "second_hand_years")], # test data data for features used in classi
  cl = train_data$biopsy, # vector of class labels for training data
  k = 20)
lung.knn.conf.20 = table(true = test_data$biopsy, predicted = lung.knn.20)
lung.knn.conf.20
# Use the kNN classifier with k = 50
lung.knn.50 = knn(
  train = train_data[c("smoke_years", "second_hand_years")], # training data for features used in class
  test = test_data[c("smoke_years", "second_hand_years")], # test data data for features used in classi
  cl = train_data$biopsy, # vector of class labels for training data
  k = 50)
lung.knn.conf.50 = table(true = test_data$biopsy, predicted = lung.knn.50)
lung.knn.conf.50
```