MBA6693 Assignment 2

Xinkai Zhou, 3326722

Due 18/7/2020

Introoduction

In this assignment, I will practice classification problem using logistic regression, LDA and KNN methods, and compare the model using out of sample data, base on confusion matrix.

The data I am using is the famous (Fisher's or Anderson's) iris data set gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris. The species are Iris setosa, versicolor, and virginica.

The dataset "iris" is a data frame with 150 cases (rows) and 5 variables (columns) named Sepal.Length, Sepal.Width, Petal.Length, Petal.Width, and Species.

There are two assumptions I have made for this assignment

1. Use 70% randomly picked observations as training data, train the logistic regression, LDA and KNN(3).
2. The 30% held out data used to test aginst the models. Compare model performance based on confusion matrix for each model, and print out of sample error rate.

Model

First load and prepare the iris data for further analysis.

```r
#### Preparing the Data ####
# clearing the environment
rm(list = ls())

# loading the required library
library(ISLR)
```

```
## Warning: package 'ISLR' was built under R version 3.6.3
```

```r
# loading the iris data
data(iris)
iris_2d <- iris[,c("Petal.Length", "Petal.Width", "Species")]
head(iris_2d,4)
```

```
##   Petal.Length Petal.Width Species
## 1          1.4         0.2  setosa
## 2          1.4         0.2  setosa
## 3          1.3         0.2  setosa
## 4          1.5         0.2  setosa
```

```r
# removing the last 50 rows (virginica rows)
iris_2d <- iris_2d[-c(101:150),]
# changing the factor level
str(iris_2d)
```

```
## 'data.frame':    100 obs. of  3 variables:
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
Factors_Species <- c("setosa","versicolor")
iris_2d$Species <- factor(Factors_Species)

summary(iris_2d)
```

```
##   Petal.Length    Petal.Width          Species
##  Min.   :1.000   Min.   :0.100   setosa    :50
##  1st Qu.:1.500   1st Qu.:0.200   versicolor:50
##  Median :2.450   Median :0.800
##  Mean   :2.861   Mean   :0.786
##  3rd Qu.:4.325   3rd Qu.:1.300
##  Max.   :5.100   Max.   :1.800
```

Logistics Regression

Now I use logistic regression to create a predictive model using 70% of the data as training data. At the final stage, I will compare the regression results with 30% of real data based on confusion matrix.

```
#### Logistic Regression: ####
sample_Number <- round(nrow(iris_2d)*.7)
train_idx <- sample (nrow(iris_2d),sample_Number)
train_data <- iris_2d[train_idx,]
`%notin%` <- Negate(`%in%`)
test_data <- iris_2d[c(1:nrow(iris_2d))%notin%train_idx,]
dim(test_data)[1]/nrow(iris_2d)
```

```
## [1] 0.3
```

```
logistic_fit <- glm(Species ~ Petal.Length
                    + Petal.Width,
                    data = iris_2d, family = binomial,
                    subset = train_idx)
logistic_prob <- predict(logistic_fit, test_data,
                         type = "response")

# Evaluate Logistic Regression Out of Sample
logistic_pred <- rep("setosa", nrow(test_data))
logistic_pred[logistic_prob > 0.5] <- "versicolor"

# Compare model performance based on confusion matrix
table(logistic_pred, test_data$Species)
```

```
##
## logistic_pred setosa versicolor
##    setosa          1          3
##    versicolor     17          9
```

```
mean(logistic_pred==test_data$Species)
```

```
## [1] 0.3333333
```

Linear Discrimination Analysis

Now using the LDA method to predict based of the training set:

```
#### Linear Discrimination Analysis (LDA) ####
library(MASS)

ida_fit <- lda(Species ~ Petal.Length
               + Petal.Width,
               data = iris_2d, family = binomial,
               subset = train_idx)
ida_fit
```

```
## Call:
## lda(Species ~ Petal.Length + Petal.Width, data = iris_2d, family = binomial,
##     subset = train_idx)
##
## Prior probabilities of groups:
##     setosa versicolor
##  0.4571429  0.5428571
##
## Group means:
##            Petal.Length Petal.Width
## setosa         2.728125   0.7531250
## versicolor     2.802632   0.7631579
##
## Coefficients of linear discriminants:
##                    LD1
## Petal.Length  3.322729
## Petal.Width  -8.199514
```

```
# Prediction
ida_pred <- predict(ida_fit, test_data)

# Compare model performance based on confusion matrix
table(ida_pred$class, test_data$Species)
```

```
##
##               setosa versicolor
##   setosa           1          3
##   versicolor      17          9
```

```
mean(ida_pred$class == test_data$Species)
```

```
## [1] 0.3333333
```

k-Nearest Neighbors Method

Finally, I will perform KNN method to create another model and then evaluate it based on the confusion matrix like previous steps.

```
#### KNN(k-Nearest Neighbors) ####
library(class)
train_x <- train_data[,c("Petal.Length","Petal.Width")]
train_y <- train_data[,"Species"]
test_x <- test_data[,c("Petal.Length","Petal.Width")]

set.seed(1)
knn_pred <- knn(train_x, test_x, train_y, k = 3)

# Compare model performance based on confusion matrix
table(knn_pred, test_data$Species)
```

```
##
## knn_pred     setosa versicolor
##   setosa          6          3
##   versicolor     12          9
```

```
mean(knn_pred == test_data$Species)
```

```
## [1] 0.5
```

Conclusion

In the rogistics regression, we can see the test error rate is 1 - 0.46667 = 53.334%, which is worse than random guessing. Also we can see that the linear discrimination analysis' test error rate is 1 - 0.46667 = 53.334%, which is still worse than random guessing and similar to logistic regression. The KNN model produces the test error rate is 1 - 0.5 = 50%, which is like random guessing but better than logistic regression and LDA in our case.

These three models are not performing well due to the relative small sample size. These method is a train method that means more samples will provides better error rate.