LDA ON DIABETES DATASET

Srimanta Singha

2023-07-03

Before using the LDA algorithm to classify the dataset, we first study the dataset DIABETES.

Description on data:

This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective is to predict based on diagnostic measurements whether a patient has diabetes or not. There are 8 features (explanatory variables) in the dataset to predict a patient has diabetes or not.

The variables in our dataset are:

• Pregnancies: Number of times pregnant

• Glucose: Plasma glucose concentration in 2 hours in an oral glucose tolerance test

• Blood Pressure: Diastolic blood pressure (mm Hg)

• Skin Thickness: Triceps skin fold thickness (mm)

• Insulin: 2-Hour serum insulin (mu U/ml)

• BMI: Body mass index (weight in kg/(height in m)^2)

• Age: Age (years)

• Outcome: Class variable (0 or 1)

Here, the outcome is our response variable which is categorical variable having two classes, namely 0 and 1, where 0 : patient has no diabetes, 1: patient has diabetes.

Since the output variable is categorical and has some labels, we will use classification technique under supervised machine learning.

Here we will use LDA algorithm to classify the data set.

Aim :

To build the LDA model and classify the dataset and predict the new patient status on given features.

Data Structure:

We will use R code to analyse the data using LDA. Then structure of the data structure given as follows:

```
data=read.csv("C:/Users/Srimanta/Dataset/diabetes.csv") # loading the dataset.
head(data)
```

```
##   Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI
## 1           6     148            72            35       0 33.6
## 2           1      85            66            29       0 26.6
## 3           8     183            64             0       0 23.3
## 4           1      89            66            23      94 28.1
## 5           0     137            40            35     168 43.1
## 6           5     116            74             0       0 25.6
##   DiabetesPedigreeFunction Age Outcome
## 1                    0.627  50       1
## 2                    0.351  31       0
## 3                    0.672  32       1
## 4                    0.167  21       0
## 5                    2.288  33       1
## 6                    0.201  30       0
```

Data processing:

We will replace 0 by NA to identify missing values on the features which values can never 0 generally but here takes 0 in the given dataset. First exclude the features which may takes 0(here Pregnancies) and Outcome(response) from the original data set.

```
data_origin=data # copying original dataset.
data1=data[,-c(1,9)]  #exclude Pregnancies and Outcome.
data1[data1==0]=NA  #REPLACE 0 BY NA
colSums(is.na(data1)) # Identify columns containing missing values
```

```
##                 Glucose           BloodPressure            SkinThickness
##                       5                      35                      227
##                 Insulin                     BMI DiabetesPedigreeFunction
##                     374                      11                        0
##                     Age
##                       0
```

Clearly, Age and diabetes pedigree function also does not contain missing values and hence among the 8 features 3 features does not missing values. Thus the input matrix containing missing values is given by as

```
data2=data1[,-c(6,7)] # input matrix containing missing data.
```

Now we will use multiple imputation technique to impute missing value.

```
 library(mice)
```

```
##
## Attaching package: 'mice'
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

```
imputed =mice(data2,method="pmm") # multiple imputation algorithm applying.
```

```
##
##  iter imp variable
##   1   1  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   1   2  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   1   3  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   1   4  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   1   5  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   2   1  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   2   2  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   2   3  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   2   4  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   2   5  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   3   1  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   3   2  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   3   3  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   3   4  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   3   5  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   4   1  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   4   2  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   4   3  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   4   4  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   4   5  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   5   1  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   5   2  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   5   3  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   5   4  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   5   5  Glucose  BloodPressure  SkinThickness  Insulin  BMI
```

```
completed_data=complete(imputed)    # after imputation getting original dataset #data2.
data_new=cbind(completed_data,data[,c(1,7,8,9)]) # the new input matrix without #missing data.
head(data_new)
```

```
##   Glucose BloodPressure SkinThickness Insulin  BMI Pregnancies
## 1     148            72            35     127 33.6           6
## 2      85            66            29      36 26.6           1
## 3     183            64            33      90 23.3           8
## 4      89            66            23      94 28.1           1
## 5     137            40            35     168 43.1           0
## 6     116            74            19     110 25.6           5
##   DiabetesPedigreeFunction Age Outcome
## 1                    0.627  50       1
## 2                    0.351  31       0
## 3                    0.672  32       1
## 4                    0.167  21       0
## 5                    2.288  33       1
## 6                    0.201  30       0
```

LDA classification:

We will apply LDA on the new dataset data_new.

Required library:

```
library("MASS")
```

Scalling the dataset:

One of the key assumptions of linear discriminant analysis is that each of the predictor variables have the same variance. An easy way to assure that this assumption is met is to scale each variable such that it has a mean of 0 and a standard deviation of 1.

```
data_new[,1:8]=scale(data[,1:8])
apply(data_new[,1:8],2,mean)  # finding mean of #each column of data_new[,1:8]
```

```
##                 Glucose           BloodPressure           SkinThickness
##           -6.901102e-17           -3.640265e-18            1.177826e-17
##                 Insulin                     BMI             Pregnancies
##            4.668542e-17           -4.414552e-17           -1.971323e-16
## DiabetesPedigreeFunction                     Age
##            6.894834e-17            1.987660e-16
```

```
apply(data_new[,1:8],2,sd)  ## finding standard deviation of each column of data_new[,1:8]
```

```
##                 Glucose           BloodPressure           SkinThickness
##                       1                       1                       1
##                 Insulin                     BMI             Pregnancies
##                       1                       1                       1
## DiabetesPedigreeFunction                     Age
##                       1                       1
```

Splitting the dataset:

```
set.seed(1)
sample=sample(c(TRUE, FALSE),
nrow(data_new), replace=TRUE, prob=c(0.7,0.3))
train=data_new[sample, ]
test=data_new[!sample, ]
```

MODEL FITTING ON TRAIN DATA:

```
model=lda(Outcome ~., data=train) # LDA MODEL BUILDING
model
```

```
## Call:
## lda(Outcome ~ ., data = train)
##
## Prior probabilities of groups:
##         0         1
## 0.6537037 0.3462963
##
## Group means:
##      Glucose BloodPressure SkinThickness     Insulin         BMI Pregnancies
## 0 -0.1868942    -0.3264806   -0.06018653 -0.08086646 -0.08785931  -0.2174835
## 1  0.4221085     0.6828575    0.09041970  0.01062072  0.19301049   0.3864703
##   DiabetesPedigreeFunction        Age
## 0               -0.1049029 -0.1269541
## 1                0.2534785  0.3469294
##
## Coefficients of linear discriminants:
##                                 LD1
## Glucose                   0.465284518
## BloodPressure             0.922083776
## SkinThickness            -0.153507597
## Insulin                  -0.015112601
## BMI                      -0.117623914
## Pregnancies               0.428177615
## DiabetesPedigreeFunction  0.198717084
## Age                       0.003418632
```

PREDICTION ON TEST DATA :

```
predicted <- predict(model, test) # Use LDA to make prediction on test data
names(predicted) # Gives the variables name in the data frame predicted.
```

```
## [1] "class"     "posterior" "x"
```

```
head(predicted$class) # Gives the predicted class of first six row of test data.
```

```
## [1] 0 0 0 1 0 0
## Levels: 0 1
```

```
head(predicted$posterior) # Gives the posterior probabilities of first six rows.
```

```
##            0          1
## 4  0.9723282 0.02767178
## 6  0.8410173 0.15898267
## 7  0.9579671 0.04203287
## 15 0.3588041 0.64119587
## 17 0.7677680 0.23223205
## 18 0.7807664 0.21923358
```

```
head(predicted$x) # Gives the predicted values of linear discriminant of first six rows.
```

```
##           LD1
## 4  -1.7363617
## 6  -0.4638310
## 7  -1.4454120
## 15  1.0458795
## 17 -0.1479157
## 18 -0.1979091
```

ACCURACY:

```
mean(predicted$class==test$Outcome)
```

```
## [1] 0.745614
```

The accuracy is 74.56%.

#NEW DATA POINT PREDICTION:

```
Pregnancies=5
Glucose=170
BloodPressure=60
SkinThickness=28
Insulin=140
BMI=30.5
DiabetesPedigreeFunction=0.693
Age=40
new=data.frame(Pregnancies=5,Glucose=170,BloodPressure=60,SkinThickness=28,Insulin=140,BMI=30.5,DiabetesPedigreeFunction=0.6
93,Age=40)
predicted=predict(model,new)
predicted$class
```

```
## [1] 1
## Levels: 0 1
```

This implies our new data point belongs to the class 1 that is , the patient may have the diabetes.

```
                    # THANK YOU
```