Stat 577: Discriminant Analysis

T Shapla

11/08/2023

## Example: Iris data

In this example, we will use iris data for predicting iris species based on the predictor variables Sepal.Length, Sepal.Width, Petal.Length, and Petal.Width, all measured in centimeter. The grouping variable in this data is “Species”, which has three levels: setosa, versicolor, and virginica. Since all numeric variables have the same scale, we do not need to standardize the data. However, for presentation purpose we will standardize the data by subtracting the variable mean and dividing by the standard deviation.

## Checking the Assumptions

First thing first. We check the following two assumptions for iris data:

1. Multivariate normality of numeric variables
2. Homogeneity of covariance matrices

library(mvnormtest)  
mshapiro.test(t(iris[,1:4]))

Shapiro-Wilk normality test  
  
data: Z  
W = 0.97935, p-value = 0.02342

library(car)  
library(carData)  
library(heplots)  
boxM(iris[,1:4], iris[,5])

Box's M-test for Homogeneity of Covariance Matrices  
  
data: iris[, 1:4]  
Chi-Sq (approx.) = 140.94, df = 20, p-value < 2.2e-16

From the output we note that both assumptions are violated for the data at 5% level of significance (p-value<0.05). To make the data work, below are couple of approaches that we can undertake:

1. Perform both tests at 1% significance level. Thus, the normality assumption is being satisfied (p-value>0.01). Since the the data shows evidence of heterogeneity of covariance matrices at 1% level of significance (p-value < 2.2e-16), we can apply quadratic discriminant analysis instead of linear discriminant analysis for the data.
2. We attempt to data transformation by taking log or applying BoxCox transformation. If both assumptions are satisfied, then we can go ahead and perform linear discriminant analysis.

For the sake of presentation, we assume that both assumptions are met and continue with developing linear discriminant functions. Later in this presentation, we will learn how to find the quadratic discriminant functions, which is more suitable for this situation.

## Splitting the data

First we split the data into training set (80%) and test set (20%) by assigning observations randomly into these two groups.

We have a total of 150 observations in iris data. Thus, 80% of 150 will be 120 observations, which will comprise the training set.

set.seed(123)# for reproducibility  
data("iris")  
dim(iris) #to see number of rows and columns

[1] 150 5

class(iris)

[1] "data.frame"

# Split the data into training set (80%) and test set (20%)  
index=sample(1:nrow(iris), floor(0.8\*nrow(iris)), replace=F)  
head(index,3)

[1] 14 50 118

train.data=iris[index,]  
head(train.data,3)

Sepal.Length Sepal.Width Petal.Length Petal.Width Species  
14 4.3 3.0 1.1 0.1 setosa  
50 5.0 3.3 1.4 0.2 setosa  
118 7.7 3.8 6.7 2.2 virginica

test.data=iris[-index,]

## Standardizing the data

Since all the predictor variables are measured on the same scale, data standardization is not required. However, it is recommended to standardize the variables to give them equal weight in the analysis.

Below we standardize/normalize all variables except the Species variable.

#install. packages("tidyverse")  
library(tidyverse) # to use select() function  
library(dplyr)  
train.trs=data.frame(scale(select(train.data, where(is.numeric))))#scaling all numeric variables  
  
test.trs=data.frame(scale(select(test.data, where(is.numeric))))

Since fifth column “Species” was dropped from train.trs and test.trs data, we can easily add it back by using the following code:

train.trs$Species=train.data$Species  
test.trs$Species=test.data$Species  
head(train.trs,3)

Sepal.Length Sepal.Width Petal.Length Petal.Width Species  
14 -1.830156 -0.1347287 -1.511194 -1.455808 setosa  
50 -1.004522 0.5484026 -1.343904 -1.326690 setosa  
118 2.180068 1.6869548 1.611569 1.255675 virginica

head(test.trs,3)

Sepal.Length Sepal.Width Petal.Length Petal.Width Species  
1 -0.9388007 1.0472052 -1.294350 -1.244756 setosa  
2 -1.2032517 -0.1163561 -1.294350 -1.244756 setosa  
3 -1.4677026 0.3490684 -1.354552 -1.244756 setosa

## Moving forward with the LDA

To perform linear discriminant analysis, we use lda() function from MASS package. In order to install the package, first remove the # sign from the first line in R chunk below and execute it. Once the package has been installed, comment this command line to suppress it from future execution.

In the lda() function, the first input is formula of the form group~x1+x2+…That is, the response is the grouping factor and the right hand side specifies the (non-factor) discriminators/predictors. To include all predictors in the model, we write group~. instead of typing each predictor with a plus sign between them.

The prior probabilities of class membership is specified by the option prior=c(). If unspecified, the class proportions for the training set are used. If present, the probabilities should be specified in the order of the factor levels.

#install.packages("MASS")  
lda.iris=MASS::lda(Species~., data=train.trs)  
#you may assign equal probability by using prior=c(1,1,1)/3  
lda.iris

Call:  
lda(Species ~ ., data = train.trs)  
  
Prior probabilities of groups:  
 setosa versicolor virginica   
 0.3333333 0.2916667 0.3750000   
  
Group means:  
 Sepal.Length Sepal.Width Petal.Length Petal.Width  
setosa -1.02221394 0.8159624 -1.3048690 -1.2556746  
versicolor 0.09070745 -0.6617157 0.2573093 0.1305016  
virginica 0.83808437 -0.2106321 0.9597541 1.0146539  
  
Coefficients of linear discriminants:  
 LD1 LD2  
Sepal.Length 0.7020488 -0.02792819  
Sepal.Width 0.6317110 -0.91672010  
Petal.Length -3.9071448 1.55497721  
Petal.Width -2.0567420 -2.03874581  
  
Proportion of trace:  
 LD1 LD2   
0.9924 0.0076

Note: In the output above, we see that two linear discriminant functions are produced. In general, for linear discriminant analysis (LDA) in R, when you have predictors and groups, you can have a maximum of discriminant functions.

You can extract various output objects produced by lda() function just by typing $ sign to the next of lda object name. For example, to get the estimated prior probabilities and the group sample mean of each variable, type

lda.iris$prior

setosa versicolor virginica   
 0.3333333 0.2916667 0.3750000

round(lda.iris$means, digits=4)

Sepal.Length Sepal.Width Petal.Length Petal.Width  
setosa -1.0222 0.8160 -1.3049 -1.2557  
versicolor 0.0907 -0.6617 0.2573 0.1305  
virginica 0.8381 -0.2106 0.9598 1.0147

When you run LDA using the lda() function, one of the results is the “Proportion of trace,” which is a measure of how well the discriminant functions separate the classes or groups in your data. In general, a higher proportion of trace suggests that the LDA model is effective in capturing the variability in the data and discriminating among the classes. Researchers often examine this value to assess the quality of the LDA model and its ability to distinguish between the groups in the data. The proportion of trace for LD1 is found to be 0.9924, which is excellent.

## Predicting observations in the test data

Below we predict the membership of observations in the test data (test.trs) using the linear discriminant models based on the train data.

predictions=predict(lda.iris, test.trs, prior = lda.iris$prior)

The predict() function produces the following information for each observation in the test data:

1. predicted class (in this case, Species) given by ‘class’ variable
2. the posterior probability of belonging in each class given by ‘posterior’ variable
3. The linear discriminant scores calculated based on each linear discriminant function given by ‘x’

We can extract these information as shown below:

head(predictions$class, 3)

[1] setosa setosa setosa  
Levels: setosa versicolor virginica

head(predictions$posterior, 3)

setosa versicolor virginica  
1 1 1.259171e-19 2.296068e-37  
2 1 1.491695e-15 2.520235e-32  
3 1 4.059051e-17 3.397057e-34

head(predictions$x,3)

LD1 LD2  
1 7.619802 -0.4087192  
2 6.699110 0.6653265  
3 7.042686 0.1524349

## Model Accuracy

To evaluate efficiency of the linear discriminant model developed above, we find the model efficacy by calculating the proportion of correctly assigning each subject into a class/category in the test data.

accuracy=mean(predictions$class==test.trs$Species)  
accuracy

[1] 0.9333333

The model efficacy is found to be about 93%, which is reasonably a high percentage. Note that, by default, the probability cutoff used to decide group-membership is 0.5.

## Prediction of group memebership for new observations

Suppose we wish to use our model for group membership of three new subjects, in this case, flowers based on the measurements on four variables. For these three new observations, the species are unknown. Note that in order to use the lda model developed above, the input variables should be given in normalized scale.

After executing the discriminant analysis, we see that the model assigned these observations to setosa, setosa, and virginica group, respectively with posterior probability of 1 or very close 1.

Sepal.Length=c(-0.6751142, -1.4160932, 2.165305)  
Sepal.Width=c(1.5205823, -0.2243482, 1.769858)  
Petal.Length=c(-1.341845, -1.396989, 1.525659)  
Petal.Width=c(-1.382448, -1.508892, 1.146421)  
  
new.obs=data.frame(Sepal.Length, Sepal.Width, Petal.Length, Petal.Width)  
pred.new.obs=predict(lda.iris, new.obs) #predict(model.object, newdata)  
pred.new.obs

$class  
[1] setosa setosa virginica  
Levels: setosa versicolor virginica  
  
$posterior  
 setosa versicolor virginica  
1 1.000000e+00 1.725369e-23 1.150795e-42  
2 1.000000e+00 3.122534e-18 1.982721e-36  
3 1.726241e-38 2.769345e-05 9.999723e-01  
  
$x  
 LD1 LD2  
1 8.572727 -0.643172  
2 7.425750 1.149175  
3 -5.680674 -1.647833

Now, suppose we have collected 3 observations with measurements on four variables given all in centimeter unit. Therefore, we do not need to normalize the data. We predict the membership of these 3 subjects using the model based on the original un-transformed ‘train.data’.

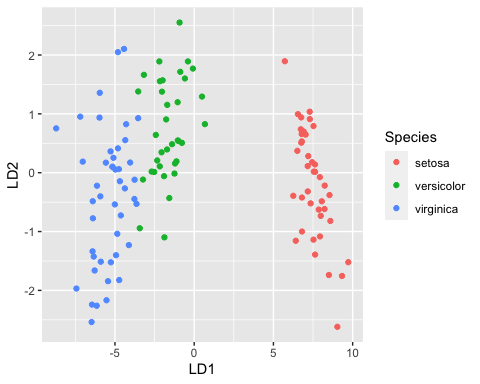
lda.untr=MASS::lda(Species~., train.data)#lda based on un-transformed data  
  
Sepal.Length=c(5.1, 4.9, 6.7)  
Sepal.Width=c(3.5, 3, 3.3)  
Petal.Length=c(1.4,1.4,5.7)  
Petal.Width=c(0.2,0.2, 2.5)  
new.untr=data.frame(Sepal.Length, Sepal.Width, Petal.Length, Petal.Width)  
  
pred.new.untr=predict(lda.untr, new.untr)  
pred.new.untr

$class  
[1] setosa setosa virginica  
Levels: setosa versicolor virginica  
  
$posterior  
 setosa versicolor virginica  
1 1.000000e+00 4.817861e-21 1.881837e-39  
2 1.000000e+00 4.028384e-17 1.309429e-34  
3 7.831529e-43 7.870796e-07 9.999992e-01  
  
$x  
 LD1 LD2  
1 7.991192 -0.2804213  
2 7.106346 0.7699005  
3 -6.448260 -2.2415540

## Plotting of linear discriminants

Below we plot the linear discriminants, obtained by computing LD1 and LD2 for each of the training observations.

#install.packages("tidyverse")  
library(tidyverse)  
library(ggplot2)  
  
lda.data=cbind(train.trs, predict(lda.iris)$x)  
ggplot(lda.data, mapping=aes(x=LD1, y=LD2))+geom\_point(mapping=aes(color=Species))



Above graph shows how well the discriminant functions have placed the flower species into three different categories.

## Finding Confusion matrix

#install.packages("caret")#to use for confusionMatrix()  
#caret is short for classification and regression training  
library(caret)  
confusion\_matrix <- confusionMatrix(predictions$class, test.data$Species)  
print(confusion\_matrix)

Confusion Matrix and Statistics  
  
 Reference  
Prediction setosa versicolor virginica  
 setosa 10 0 0  
 versicolor 0 13 0  
 virginica 0 2 5  
  
Overall Statistics  
   
 Accuracy : 0.9333   
 95% CI : (0.7793, 0.9918)  
 No Information Rate : 0.5   
 P-Value [Acc > NIR] : 4.34e-07   
   
 Kappa : 0.8947   
   
 Mcnemar's Test P-Value : NA   
  
Statistics by Class:  
  
 Class: setosa Class: versicolor Class: virginica  
Sensitivity 1.0000 0.8667 1.0000  
Specificity 1.0000 1.0000 0.9200  
Pos Pred Value 1.0000 1.0000 0.7143  
Neg Pred Value 1.0000 0.8824 1.0000  
Prevalence 0.3333 0.5000 0.1667  
Detection Rate 0.3333 0.4333 0.1667  
Detection Prevalence 0.3333 0.4333 0.2333  
Balanced Accuracy 1.0000 0.9333 0.9600

## Quadratic Discriminant Analysis

Below we perform the quadratic discriminant analysis for the iris data. In this case, we use qda() function from MASS package. All other coding are same as before. The model accuracy is found to be 0.9666667, which implies the quadratic discriminant function is performing better than the linear discriminant function in predicting group membership of iris flowers.

qda.iris=MASS::qda(Species~., data=train.trs)  
qda.iris

Call:  
qda(Species ~ ., data = train.trs)  
  
Prior probabilities of groups:  
 setosa versicolor virginica   
 0.3333333 0.2916667 0.3750000   
  
Group means:  
 Sepal.Length Sepal.Width Petal.Length Petal.Width  
setosa -1.02221394 0.8159624 -1.3048690 -1.2556746  
versicolor 0.09070745 -0.6617157 0.2573093 0.1305016  
virginica 0.83808437 -0.2106321 0.9597541 1.0146539

predict.qda=predict(qda.iris, test.trs, prior = qda.iris$prior)  
mean(predict.qda$class==test.trs$Species)

[1] 0.9666667