How can one perform quadratic discriminant analysis in R, step-by-step?

Authored by

stats writer

April 21, 2024

Quadratic discriminant analysis (QDA) is a statistical method used to classify data into different groups based on their characteristics. It is commonly used in machine learning and data analysis. In this description, we will explain the step-by-step process of performing quadratic discriminant analysis in R.

Step 1: Import the data
The first step in performing QDA in R is to import the data that you want to analyze. This can be done by using the "read.csv" function or any other appropriate function based on the format of your data.

Step 2: Pre-process the data
Before performing QDA, it is important to pre-process the data to ensure that it is in the correct format. This may include removing any missing values, converting categorical variables into numerical values, and scaling the data if necessary.

Step 3: Split the data into training and test sets
To evaluate the performance of the QDA model, it is important to split the data into two sets - the training set and the test set. The training set will be used to train the model, while the test set will be used to evaluate its performance.

Step 4: Fit the QDA model
Using the "qda" function in R, fit the QDA model to the training data. This function will estimate the parameters of the model based on the training data.

Step 5: Make predictions
Once the model has been trained, use it to make predictions on the test data using the "predict" function. This will give you the predicted class labels for each observation in the test set.

Step 6: Evaluate the model
To evaluate the performance of the QDA model, compare the predicted class labels with the actual class labels in the test set. This can be done by calculating various metrics such as accuracy, precision, and recall.

Step 7: Refine the model (optional)
If the performance of the QDA model is not satisfactory, you can refine it by adjusting the model parameters or by performing feature selection to improve its accuracy.

In conclusion, quadratic discriminant analysis is a powerful tool for classifying data into different groups. By following these step-by-step instructions, you can easily perform QDA in R and use it to analyze your data.

Quadratic Discriminant Analysis in R (Step-by-Step)

Quadratic discriminant analysis is a method you can use when you have a set of predictor variables and you'd like to classify a response variable into two or more classes. It is considered to be the non-linear equivalent to linear discriminant analysis.

This tutorial provides a step-by-step example of how to perform quadratic discriminant analysis in R.

Step 1: Load Necessary Libraries

First, we'll load the necessary libraries for this example:

```
library(MASS)
library(ggplot2)
```

Step 2: Load the Data

For this example, we'll use the built-in iris dataset in R. The following code shows how to load and view this dataset:

```
#attach iris dataset to make it easy to work with
attach(iris)
```

```
#view structure of dataset
str(iris)

'data.frame': 150 obs. of 5 variables:
$ Sepal.Length: num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
$ Sepal.Width : num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
$ 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 ...
```

We can see that the dataset contains 5 variables and 150 total observations.

For this example we'll build a quadratic discriminant analysis model to classify which species a given flower belongs to.

We'll use the following predictor variables in the model:

Sepal.lengthSepal.WidthPetal.LengthPetal.Width

And we'll use them to predict the response variable

***Species*, which takes on the following three potential classes:**

setosaversicolorvirginica

Step 3: Create Training and Test Samples

Next, we'll split the dataset into a training set to train the model on and a testing set to test the model on:

```
#make this example reproducible
set.seed(1)

#Use 70% of dataset as training set and remaining 30% as testing set
sample <- sample(c(TRUE, FALSE), nrow(iris), replace=TRUE, prob=c(0.7,0.3))
train <- iris
test <- iris
```

Step 4: Fit the QDA Model

Next, we'll use the qda() function from the MASS package to fit the QDA model to our data:

```
#fit QDA model
```

model <- qda(Species~., data=train)

#view model output
model

Call:
qda(Species ~ ., data = train)

Prior probabilities of groups:
setosa versicolor virginica
0.3207547 0.3207547 0.3584906

Group means:
Sepal.Length Sepal.Width Petal.Length Petal.Width
setosa 4.982353 3.411765 1.482353 0.2411765
versicolor 5.994118 2.794118 4.358824 1.3676471
virginica 6.636842 2.973684 5.592105 2.0552632

Here is how to interpret the output of the model:

Prior probabilities of group: These represent the proportions of each Species in the training set. For example, 35.8% of all observations in the training set were of species *virginica*.

Group means: These display the mean values for each

predictor variable for each species.

Step 5: Use the Model to Make Predictions

Once we've fit the model using our training data, we can use it to make predictions on our test data:

#use QDA model to make predictions on test data
predicted <- predict(model, test)

names(predicted)

"class" "posterior" "x"

This returns a list with two variables:

class: The predicted classposterior: The posterior probability that an observation belongs to each class

We can quickly view each of these results for the first six observations in our test dataset:

#view predicted class for first six observations in test set
head(predicted$class)

setosa setosa setosa setosa setosa setosa

Levels: setosa versicolor virginica

#view posterior probabilities for first six observations in test set
head(predicted$posterior)

setosa versicolor virginica
4 1 7.224770e-20 1.642236e-29
6 1 6.209196e-26 8.550911e-38
7 1 1.248337e-21 8.132700e-32
15 1 2.319705e-35 5.094803e-50
17 1 1.396840e-29 9.586504e-43
18 1 7.581165e-25 8.611321e-37

Step 6: Evaluate the Model

We can use the following code to see what percentage of observations the QDA model correctly predicted the Species for:

#find accuracy of model
mean(predicted$class==test$Species)

1

It turns out that the model correctly predicted the

Species for 100% of the observations in our test dataset.

In the real-world an QDA model will rarely predict every class outcome correctly, but this iris dataset is simply built in a way that machine learning algorithms tend to perform very well on it.

You can find the complete R code used in this tutorial here.