Learning Quadratic Discriminant Analysis (QDA) with R: A Step-by-Step Guide

Authored by

Mohammed looti

November 6, 2025

Quadratic Discriminant Analysis (QDA) stands as a sophisticated statistical method essential for classification tasks. Its primary function is to predict a categorical response variable utilizing a collection of continuous or discrete predictor variables. A core assumption of QDA is that observations within each specified class are derived from a Gaussian distribution. Crucially, QDA distinguishes itself from related techniques by permitting each class to possess its own unique covariance matrix, thereby allowing for distinct variances and correlations across groups.

This flexibility is the reason QDA is classified as a non-linear classifier. When the true separation between classes is curved or complex--meaning the decision boundaries are non-linear--QDA is often preferred over its more restrictive counterpart, Linear Discriminant Analysis (LDA). By accommodating different covariance structures, QDA offers superior performance in modeling intricate data patterns where LDA's simplifying assumptions would lead to suboptimal classification.

This comprehensive guide provides a practical, step-by-step methodology for implementing and interpreting a Quadratic Discriminant Analysis model using the powerful R programming language. We will walk through data preparation, model fitting, prediction generation, and rigorous performance evaluation.

Step 1: Setting Up the R Environment and Loading Essential Libraries

Before initiating any statistical analysis in R, the required packages must be loaded into the active environment. For executing QDA, the central function is housed within the **MASS** package, which is fundamental to many advanced statistical computations in R. We will ensure this package is installed and loaded. Additionally, while not mandatory for the model fitting process itself, we include **ggplot2** in case visualization of the data or results is required later in the workflow.

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

Users must verify that these packages are correctly installed on their system prior to attempting to load them using the `library()` function. If installation is necessary, the standard R command `install.packages("MASS")` should be executed.

Step 2: Acquiring and Inspecting the Dataset

To provide a clear, practical demonstration of QDA implementation, we will utilize the widely recognized **iris** dataset, which is conveniently built into the R environment. This dataset serves as a canonical benchmark for various classification algorithms. The dataset comprises 150 observations of flowers, detailing four distinct physical measurements (our predictor variables) and

classifying them across three unique species (our categorical response variable).

The following R code snippet loads the dataset and provides a structured overview, confirming the nature of the variables, their data types, and the total count of observations available for analysis:

 # Attach the iris dataset to the R search path for easy reference
attach(iris)

# View the internal structure of the 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 ...

The structural output verifies that we are working with 150 complete observations and five variables. Our objective in modeling is to utilize the four numeric features--Sepal Length, Sepal Width, Petal Length, and Petal Width--to accurately categorize the nominal factor variable, *Species*.

The predictor variables used in the QDA model are:

Sepal.Length
Sepal.Width
Petal.Length
Petal.Width

These four measurements are used to classify the response variable, *Species*, which encompasses three distinct biological classes:

setosa
versicolor
virginica

Step 3: Partitioning the Data for Training and Testing

In rigorous machine learning practice, it is paramount to evaluate a model's generalizability on data it has never encountered. To achieve this, we must partition the **iris** dataset into two distinct subsets: a training set, which will be used exclusively to fit the QDA model parameters, and a

testing set, which is reserved solely for assessing the model's predictive accuracy and performance validation.

For this demonstration, we will allocate 70% of the total observations to the training set and reserve the remaining 30% for testing. To ensure that the random sampling process yields identical results every time the script is run--a crucial aspect of reproducible research--we initialize the random number generator using the `set.seed(1)` function before performing the split.

 # Set a seed to ensure reproducibility of the random sample
set.seed(1)

# Create a logical vector (sample) where TRUE = Training (70%) and FALSE = Testing (30%)
sample <- sample(c(TRUE, FALSE), nrow(iris), replace=TRUE, prob=c(0.7,0.3))
train <- iris
test <- iris

This segregation prevents data leakage and provides an unbiased estimate of how well the resulting QDA classifier will perform when deployed against new, real-world observations.

Step 4: Fitting the Quadratic Discriminant Analysis Model

With the data successfully partitioned, we can now proceed to fit the QDA model using the `qda()` function provided by the **MASS** package. The model syntax employs a standard R formula interface: `Species~.`. This notation explicitly directs R to model the `Species` variable as the outcome, utilizing the dot operator (`.`) to signify the inclusion of all remaining variables within the `train` dataset as predictors.

 # Fit the QDA model to the training data
model <- qda(Species~., data=train)

Display the model summary 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

The model output provides two primary sections critical for interpreting the QDA structure. First, the **Prior probabilities of groups** indicate the proportional frequency of each class within the training data used for fitting. These priors are crucial as they influence the final classification decision, acting as a baseline probability before considering the specific feature measurements of a new observation. In our case, *virginica* has the highest prior probability at approximately 35.8%.

Second, the **Group means** section displays the calculated average value for each of the four predictor variables, computed separately for every species. These means serve as the central coordinates or centroids for each class's Gaussian distribution. These statistics define the location of the input features for each group, which the QDA classifier uses, alongside the unique covariance matrices (not explicitly shown in this summary), to define the decision boundaries.

Step 5: Generating Predictions on Unseen Test Data

Once the QDA model has been successfully trained, our next objective is to utilize it to generate classifications for the reserved, unseen test dataset. The standard `predict()` function in R is used for this purpose, applying the fitted model parameters to the new observations.

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

names(predicted)

"class" "posterior" "x"

The resulting object, stored here as `predicted`, is a list that contains several pieces of essential classification information. The two components most relevant for evaluation are: **class**, which provides the final classification label (the species determined to have the highest likelihood); and **posterior**, which gives the posterior probability--a quantitative measure of how likely the observation is to belong to each respective class, based on the QDA calculations.

To gain a detailed understanding of the model's output, we inspect the predicted class labels and their corresponding probabilities for the initial few observations in the test set:

 # View the predicted class labels for the first six observations
head(predicted$class)

setosa setosa setosa setosa setosa setosa
Levels: setosa versicolor virginica

# View the corresponding posterior probabilities
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

A review of the posterior probabilities confirms the high confidence of the model in its classifications for these initial observations. For example, the probability assigned to the 'setosa' class is virtually 1 in every case, meaning the model is overwhelmingly certain that these particular flowers belong to that species.

Step 6: Assessing Model Performance and Accuracy

The final and most crucial stage of the classification workflow involves evaluating the effectiveness of the QDA model using the reserved test data. We quantify the model's performance by calculating the classification accuracy, which is defined as the proportion of test observations for which the model's predicted class label perfectly matches the actual, known species.

 # Calculate the accuracy of the model on the test set
mean(predicted$class==test$Species)

1

The resulting value of `1` indicates an exceptional level of performance: the Quadratic Discriminant Analysis model successfully achieved **100% accuracy**, correctly classifying every single observation in our test dataset. This perfect result is frequently observed when applying powerful, well-suited classification methods to the highly separable **iris** dataset, which is known for its clear class boundaries.

It is important to contextualize this result. While 100% accuracy is possible with canonical, clean datasets, practitioners should note that such flawless outcomes are exceedingly rare in real-world scenarios involving noisy, highly dimensional, or poorly balanced data. This specific example

serves primarily to illustrate the precise methodology required for fitting and rigorously testing a QDA model within the R environment.

Conclusion and Further Resources

We have successfully navigated the entire workflow for implementing Quadratic Discriminant Analysis in R, covering all necessary steps from initial data loading and environmental setup to comprehensive performance evaluation. QDA remains an invaluable tool within the statistical classification toolkit, particularly favored in situations where the underlying data distributions suggest complex, non-linear separation boundaries between the classes.

The capacity of QDA to utilize distinct covariance structures for each group allows it to model data heterogeneity effectively, often leading to superior predictive performance compared to LDA when assumptions of equal variance are violated. Mastering this technique provides a robust method for analyzing multivariate data.

For those interested in adapting this methodology for different analytical challenges or exploring the underlying R code further, the complete script utilized throughout this tutorial is available for review and modification via the provided external link.

Complete R Script