Learn Linear Discriminant Analysis with R: A Step-by-Step Tutorial

Authored by

Mohammed looti

November 6, 2025

Linear Discriminant Analysis (LDA) is a foundational statistical technique used extensively in machine learning for both supervised classification and effective dimensionality reduction. Its primary goal is to find linear combinations of features that best separate two or more classes of objects. Unlike Principal Component Analysis (PCA), which focuses on maximizing variance, LDA specifically seeks to maximize the distance between class means while minimizing the scatter within each class.

This comprehensive tutorial provides a rigorous, step-by-step walkthrough demonstrating the implementation of **Linear Discriminant Analysis** using the powerful R statistical computing environment. We will cover data preparation, model fitting, interpretation of the linear discriminants, and final visualization to ensure a deep understanding of this classification method.

Step 1: Load Essential Libraries

To successfully execute the Linear Discriminant Analysis, we must first ensure that the necessary statistical packages are installed and loaded into the current R session. The **MASS** package is absolutely critical for this task, as it contains the primary function, `lda()`, used for model fitting. Furthermore, we will load the ggplot2 package, which is the industry standard in R for generating high-quality, informative data visualizations later in the process.

Before proceeding, verify that both the MASS and ggplot2 libraries are available on your system. If they are not installed, you can typically install them using `install.packages("MASS")` and `install.packages("ggplot2")`. Once installed, execute the following R commands to activate them:

library(MASS)
library(ggplot2)

Step 2: Access and Examine the Dataset

For practical demonstration purposes, we will utilize the renowned built-in iris dataset in R. This dataset is a classic benchmark in classification tasks and consists of 150 observations of iris flowers, categorized into three distinct species (Setosa, Versicolor, and Virginica). The data provides four key physical measurements for each flower.

The following R commands load the data and provide a structural summary, allowing us to inspect the variable types and ensure data integrity before modeling:

 # Attach the iris dataset to make variable names accessible
attach(iris)

```
# View the structure and variable types 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 summary confirms that the dataset contains 150 observations across five variables. Our objective is to build an LDA model capable of predicting the flower's species based on its four continuous measurements. In the context of our classification problem, the variables are clearly defined as follows:

The **Predictor Variables** (or features) used to inform the model are:

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

The **Response Variable** (or target class) is *Species*, which is a categorical variable with three distinct levels:

setosa
versicolor
virginica

Step 3: Standardize the Input Data

Standardizing the data is a crucial preprocessing step for Linear Discriminant Analysis. A core assumption of LDA is that the variance and covariance structure of the predictor variables must be equal across all classes (homoscedasticity). Even if this assumption is slightly violated, standardizing the data ensures that all features contribute equally to the distance calculations used by the discriminant functions, preventing variables with larger scales from dominating the model.

We standardize the data by scaling each continuous predictor variable so that it has a mean of 0 and a standard deviation of 1. The `scale()` function in R efficiently performs this Z-score normalization:

 # Scale the first four columns (the predictor variables)
iris <- scale(iris)

We can quickly confirm the success of the scaling operation by calculating the mean and standard deviation for the first four columns using the `apply()` function. The results below confirm that the means are approximately zero (due to minor floating-point errors) and the standard deviations are exactly 1, verifying the standardization required for optimal LDA performance.

 # Calculate the mean of each scaled predictor variable
apply(iris, 2, mean)

Sepal.Length Sepal.Width Petal.Length Petal.Width
-4.484318e-16 2.034094e-16 -2.895326e-17 -3.663049e-17

# Calculate the standard deviation of each scaled predictor variable
apply(iris, 2, sd)

Sepal.Length Sepal.Width Petal.Length Petal.Width
1 1 1 1

Step 4: Divide Data into Training and Test Samples

A critical step in any robust machine learning workflow is splitting the dataset into training and testing subsets. The **training set** is used to fit the model and derive the discriminant functions, while the **testing set** is reserved to evaluate the model's performance on unseen data, thereby assessing its generalization ability.

For this demonstration, we allocate 70% of the observations to the training set and the remaining 30% to the testing set. We utilize `set.seed(1)` to ensure that the random sampling process is fully reproducible, meaning the same subsets will be generated every time the code is run.

 # Ensure the random sampling is reproducible
set.seed(1)

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

Step 5: Fit the Linear Discriminant Analysis Model

With the data standardized and split, we are ready to fit the LDA model. We employ the `lda()` function from the MASS package. The formula `Species~.` is a shorthand instruction in R that tells the function to use all other variables in the dataset (the four physical measurements) to predict the *Species* variable, utilizing only the observations contained within our training data.

 # Fit the LDA model using the training data
model <- lda(Species~., data=train)

Display the detailed model output
model

Call:
lda(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 -1.0397484 0.8131654 -1.2891006 -1.2570316
versicolor 0.1820921 -0.6038909 0.3403524 0.2208153
virginica 0.9582674 -0.1919146 1.0389776 1.1229172

Coefficients of linear discriminants:
LD1 LD2
Sepal.Length 0.7922820 0.5294210
Sepal.Width 0.5710586 0.7130743
Petal.Length -4.0762061 -2.7305131
Petal.Width -2.0602181 2.6326229

Proportion of trace:
LD1 LD2
0.9921 0.0079

The model output is highly informative and provides the necessary parameters for interpreting the discriminant functions:

Prior probabilities of group: These represent the proportion of observations in the training set

belonging to each species. For example, 35.8% belong to the *virginica* species. In Bayesian classification, these prior probabilities are used to bias the final classification towards more frequently observed classes.

Group means: This table displays the mean value for each scaled predictor variable, calculated separately for each species. These group means define the exact centroid (center) of each class in the feature space.

Coefficients of linear discriminants: These are the weights assigned to the predictor variables that form the discriminant functions (LD1 and LD2). These functions are constructed to maximize the separation between the class means relative to the within-class variance. Note the large negative weight of Petal.Length on LD1, indicating its strong importance in separation.

Proportion of trace: This metric indicates the percentage of the total between-class separation explained by each linear discriminant function. Here, LD1 accounts for approximately 99.21% of the total discriminating power, confirming that it is by far the most significant dimension for separating the three species.

Step 6: Generate Predictions and Evaluate Accuracy

Once the LDA model has been fitted to the training data, we use the `predict()` function to generate classifications for the observations in our reserved, unseen test dataset. This step is crucial for evaluating the model's accuracy and generalization capabilities.

 # Use the LDA model to generate predictions on the test data
predicted <- predict(model, test)

names(predicted)

"class" "posterior" "x"

The prediction output is returned as a list containing three vital components:

class: The final predicted categorical classification (e.g., setosa, versicolor, virginica) for each observation in the test set.

posterior: The posterior probability that an observation belongs to each of the possible classes, given the observed feature values.

x: The scores for the linear discriminants (LD1 and LD2) calculated for each observation in the test set.

Reviewing the results for the first six observations in the test set provides a tangible example of the model's output:

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

setosa setosa setosa setosa setosa setosa
Levels: setosa versicolor virginica

# View the posterior probabilities for the initial six observations
head(predicted$posterior)

setosa versicolor virginica
4 1 2.425563e-17 1.341984e-35
6 1 1.400976e-21 4.482684e-40
7 1 3.345770e-19 1.511748e-37
15 1 6.389105e-31 7.361660e-53
17 1 1.193282e-25 2.238696e-45
18 1 6.445594e-22 4.894053e-41

# View the linear discriminant scores (LD1 and LD2) for the initial six observations
head(predicted$x)

LD1 LD2
4 7.150360 -0.7177382
6 7.961538 1.4839408
7 7.504033 0.2731178
15 10.170378 1.9859027
17 8.885168 2.1026494
18 8.113443 0.7563902

Finally, we calculate the overall model accuracy by comparing the predicted class labels (`predicted$class`) against the actual species labels (`test$Species`) in the test set.

 # Calculate the proportion of correctly classified observations
mean(predicted$class==test$Species)

1

The result of `1` indicates that the model achieved a remarkable **100% accuracy**, correctly predicting the species for every observation in the test dataset. While such perfect separation is

uncommon in complex, real-world applications, the **iris** dataset is known for its highly separable classes, making it an excellent demonstration of the power of LDA when the assumptions of the model are largely met.

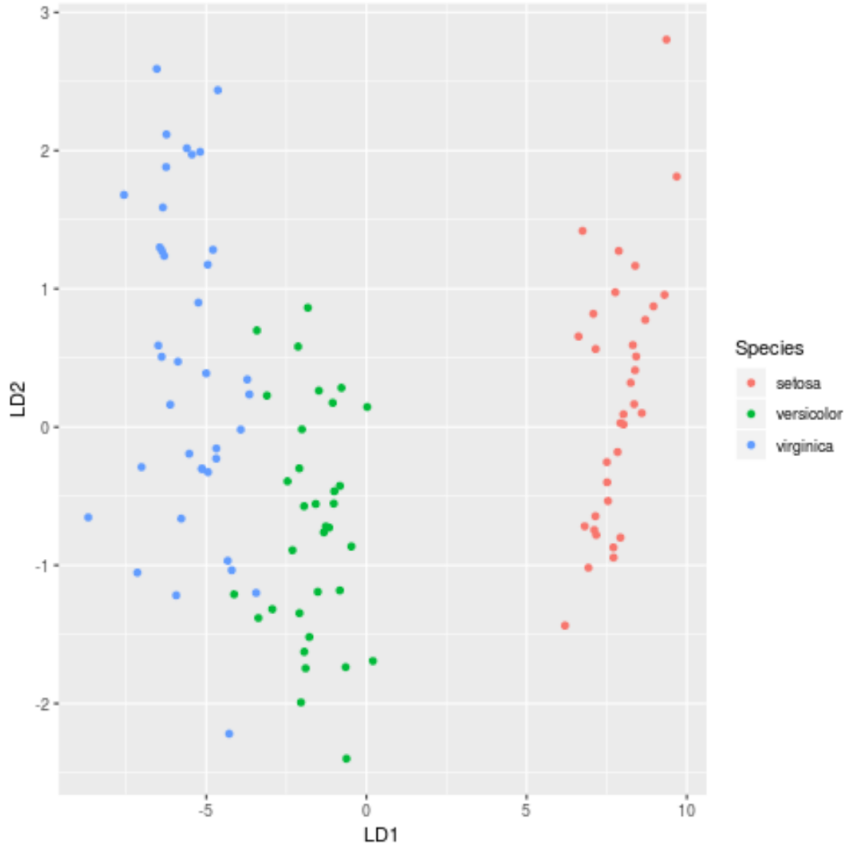
Step 7: Visualize the Linear Discriminants

The final and most intuitive step is visualizing the results using the calculated linear discriminant scores (LD1 and LD2). Since LD1 captured 99.21% of the variance, plotting the data points projected onto the LD1 and LD2 axes provides a definitive visual confirmation of the class separation achieved by the model.

We begin by combining the training data with the calculated discriminant scores and then use the **ggplot2** package to generate a scatter plot, coloring the points based on their actual species affiliation. This visualization transforms the four-dimensional feature space into a highly separable two-dimensional plot.

```
# Combine the training data with the calculated discriminant scores
lda_plot <- cbind(train, predict(model)$x)

# Create a scatter plot using LD1 and LD2, colored by Species
ggplot(lda_plot, aes(LD1, LD2)) +
geom_point(aes(color = Species))
```



The resulting plot visually confirms the exceptional separation power of the LDA model. The three species groups form distinct, non-overlapping clusters based on their coordinates along the LD axes. This graphic validation directly corresponds to the high classification accuracy observed in the previous step, demonstrating how Linear Discriminant Analysis effectively projects multi-dimensional data onto a lower-dimensional space optimized for classification.

The complete R code used for this Linear Discriminant Analysis tutorial is available for review here.