CUNY School of Professional Studies

SPS.CUNY.EDU

Lecture 07
2020 Spring Data-622
LDA – basic Classifier 1934

Raman Kannan

Instructor Email Address: Raman.Kannan@sps.cuny.edu

Acknowledgements:
Generous support from IBM Power Systems Academic Initiative
IBM PSAI provides computing infrastructure for free

Script for Algorithms

Develop the Intuition
Understand the assumptions
Develop the mathematics
Run the algorithms
Learn to interpret the result/output
Predict using the model
Learn to determine the performance
Distinguish training/testing error
Differentiate between overfitting/underfitting
Techniques to improve performance

What are we doing?

We started with review

– math/statistics/probability

We familiarized ourselves with

– Machine Learning

We took note of many tools

– required for ML

We practiced prediction

– quantitative predictors and response variables

We developed a process, performance metric

– RMSE in the case of regression

Then we encountered dichotomous response

– we transformed into logit

– logistic regression – probabilities

– Instead of RMSE we adoped AUC/Deviance

Are there other strategies to determine

– class (response) variable given predictor variables

This is an age old problem.
Qualitative response variables are all too common.

Fisher's Linear Discriminant

A **dichotomy** /daɪˈkɒtəmi/ is a partition of a whole (or a set) into two parts (subsets). In other words, this couple of parts must be

- jointly exhaustive: everything must belong to one part or the other, and
- mutually exclusive: nothing can belong simultaneously to both parts.

Such a partition is also frequently called a bipartition.

https://en.wikipedia.org/wiki/Dichotomy

The original dichotomous discriminant analysis was developed by Sir Ronald Fisher in 1936.[8] Almost 100 years ago

https://en.wikipedia.org/wiki/Linear_discriminant_analysis

LDA Formulation

If we have n-feature vectors, we can stack them into one matrix as follows;

$$Y = W^T X$$

$$where \quad X_{m\times n} = \begin{bmatrix} x_1^1 & x_1^2 & . & x_1^n \\ . & . & . & . \\ . & . & . & . \\ x_m^1 & x_m^2 & . & x_m^n \end{bmatrix} \quad , \quad Y_{C-1\times n} = \begin{bmatrix} y_1^1 & y_1^2 & . & y_1^n \\ . & . & . & . \\ . & . & . & . \\ y_{C-1}^1 & y_{C-1}^2 & . & y_{C-1}^n \end{bmatrix}$$

$$and \quad W_{m\times C-1} = \left[w_1 \mid \; w_2 \mid \; \ldots \; \mid w_{C-1} \right]$$

If wecompute W, given any new observation – using the above formula we can map it to the Y space...

LDA Space

4 The LDA Space

Linear Discriminant Analysis (LDA) [3], [4] searches for those vectors in the underlying space that best discriminate among classes (rather than those that best describe the data). More formally, given a number of independent features relative to which the data is described, LDA creates a linear combination of these which yields the largest mean differences between the desired classes. Mathematically speaking, for all the samples of all classes, we define two measures: 1) one is called *within-class* scatter matrix, as given by

$$S_w = \sum_{j=1}^{c} \sum_{i=1}^{N_j} (\mathbf{x}_i^j - \mu_j)(\mathbf{x}_i^j - \mu_j)^T,$$

where $\mathbf{x}_i^j$ is the ith sample of class j, μ_j is the mean of class j, c is the number of classes, and N_j the number of samples in class j; and 2) the other is called *between-class* scatter matrix

$$S_b = \sum_{j=1}^{c} (\mu_j - \mu)(\mu_j - \mu)^T,$$

where μ represents the mean of all classes.

Scatter S_W Within CLASS Scatter
And S_B Between CLASS Scatter

$$S_W = \sum_{i=1}^{C} S_i$$

$$\text{where} \quad S_i = \sum_{x \in \omega_i} (x - \mu_i)(x - \mu_i)^T$$

$$\text{and} \quad \mu_i = \frac{1}{N_i} \sum_{x \in \omega_i} x$$

$$S_B = \sum_{i=1}^{C} N_i (\mu_i - \mu)(\mu_i - \mu)^T$$

$$\text{where} \quad \mu = \frac{1}{N} \sum_{\forall x} x = \frac{1}{N} \sum_{\forall x} N_i \mu_i$$

$$\text{and} \quad \mu_i = \frac{1}{N_i} \sum_{x \in \omega_i} x$$

http://www2.ece.ohio-state.edu/~aleix/pami01.pdf

LDA Projection

– We can define the mean vectors for the projected samples **y** as:

$$\tilde{\mu}_i = \frac{1}{N_i} \sum_{y \in \omega_i} y \qquad and \qquad \tilde{\mu} = \frac{1}{N} \sum_{\forall y} y$$

– While the scatter matrices for the projected samples **y** will be:

$$\tilde{S}_W = \sum_{i=1}^{C} \tilde{S}_i = \sum_{i=1}^{C} \sum_{y \in \omega_i} (y - \tilde{\mu}_i)(y - \tilde{\mu}_i)^T$$

$$\tilde{S}_B = \sum_{i=1}^{C} N_i (\tilde{\mu}_i - \tilde{\mu})(\tilde{\mu}_i - \tilde{\mu})^T$$

LDA like PCA reduces dimension but preserves class semantics
Also assumes a Gaussian (mean and variance) and all features are I.i.d
Appears to work well if there is sufficient data otherwise performs < PCA
Uses PCA

LDA Implementation -1

```
pi_lda <- function(y){
pi_est ←  table(y) / length(y)
    return(as.matrix(pi_est))
    }
```

```
mu_lda <- function(X, y){
  data_est <- as.data.frame(cbind(X,y))
    data_est$X <- as.numeric(as.character(data_est$X))
      mu <- aggregate(data = data_est, X ~ y, FUN = "mean")
        colnames(mu) <- c("y", "X")
          return(mu)
          }
```

```
var_lda <- function(X, y, mu){
  n <- length(X)
    K <- length(unique(y))
      k <- unique(y)
        var_est <- 0

  for (i in 1:K){
      var_est <- sum((X[y == k[i]] - mu$X[k[i] == mu$y])^2) + var_est
        }

  var_est <- (1 / (n - K)) * var_est
    return(var_est)
    }
```

LDA Implementation -1.1

```
discriminant_lda <- function(X, pi, mu, var){
  K <- length(unique(y))
    k <- unique(y)
 disc <- matrix(nrow = length(X), ncol = K)
    colnames(disc) <- k
  for (i in 1:K){
      disc[ ,i] <- X * (mu$X[i] / var) - ((mu$X[i]^2) / (2 * var)) + log(pi[i])
        }

  disc <- as.data.frame(disc)
    disc$predict <- apply(disc, 1, FUN = "which.max")
      return(disc)    }
```

```
X <- iris[ ,1]
y <- as.character(iris[ ,5])

pi_est <- pi_lda(y)
mu_est <- mu_lda(X, y)
var_est <- var_lda(X, y, mu_est)
discriminant_est <- discriminant_lda(X, pi_est, mu_est, var_est)

table(discriminant_est$predict, iris$Species)}
```

LDA Performance

```
> discriminant_est <- discriminant_lda(X, pi_est, mu_est, var_est)
>
> table(discriminant_est$predict, iris$Species)

   setosa versicolor virginica
  1     45          6         1
  2      5         30        12
  3      0         14        37
>
```

```
> discriminant_est <- discriminant_lda(X, pi_est, mu_est, var_est)
> table(discriminant_est$predict, iris$Species)

   setosa versicolor virginica
  1     45          6         1
  2      5         30        12
  3      0         14        37
> LDA_ConfMat<-table(discriminant_est$predict, iris$Species)
> print(paste("Accuracy is::",as.numeric(sum(diag(LDA_ConfMat))/sum(LDA_ConfMat)),sep= " "))
[1] "Accuracy is:: 0.746666666666667"
> print(paste("Error is::",1-as.numeric(sum(diag(LDA_ConfMat))/sum(LDA_ConfMat)),sep= " "))
[1] "Error is:: 0.253333333333333"
```

Performance when more than 2 Classes

Table 19.1 Confusion matrix for two classes.

	Predicted class		
True Class	Positive	Negative	Total
Positive	tp : true positive	fn : false negative	p
Negative	fp : false positive	tn : true negative	n
Total	p'	n'	N

As noted in Alpaydin's book most of these measures are not meaningful when there Are more than 2 classes. We however computer Accuracy and Error.

Table 19.2 Performance measures used in two-class problems.

Name	Formula
error	$(fp + fn)/N$
accuracy	$(tp + tn)/N = 1-$error
tp-rate	tp/p
fp-rate	fp/n
precision	tp/p'
recall	$tp/p =$ tp-rate
sensitivity	$tp/p =$ tp-rate
specificity	$tn/n = 1-$ fp-rate

LDA Let us clear the Smoke and Mirrors

Let us determine W and determine training error using Scatter matrices in R

```
head(iris)
table(iris$Species)
se<-iris[iris$Species=='setosa',]
ve<-iris[iris$Species=='versicolor',]
vi<-iris[iris$Species=='virginica',]
se<-iris[iris$Species=='setosa',]
semean<-apply(se[,1:4],2,mean)
vimean<-apply(vi[,1:4],2,mean)
vemean<-apply(ve[,1:4],2,mean)
allclassmean<-apply(iris[,1:4],2,mean)
(semean+vemean+vimean)/3 == allclassmean # confirm above numbers
S1<-cov(se[,1:4])
S2<-cov(vi[,1:4])
S3<-cov(ve[,1:4])
SW<-S1+S2+S3
N1<-nrow(se)
N2<-nrow(vi)
N3<-nrow(ve)
```

LDA Let us clear the Smoke and Mirrors

Let us determine W and determine training error using Scatter matrices in R

```
SB1<-N1*(semean-allclassmean)*(semean-allclassmean)
SB2<-N2*(vimean-allclassmean)*(vimean-allclassmean)
SB3<-N3*(vemean-allclassmean)*(vemean-allclassmean)
SB<-SB1+SB2+SB3
invSW<-solve(SW)
#invSWSB<-invSW%*%SB
#DOES NOT WORK SWSBeigendecomp<-eigen(invSWSB)
invSWSB<-invSW*SB
SWSBeigendecomp<-eigen(invSWSB)
lambdas<-SWSBeigendecomp$vectors
                    [,which.max(SWSBeigendecomp$values)]
as.matrix(iris[,1:4])%*% lambdas->iris.projections
rounded.iris.projections<-round(iris.projections)
```

We computed the eigen values and used it to project data to Species.

LDA Let us clear the Smoke and Mirrors

```
iris.lda<-cbind(iris,classp=iris.projections,spectral=rounded.iris.projections)
iris.lda$predicted <-ifelse(iris.lda$spectral < (1-4.76),'virginica',
        ifelse(iris.lda$spectral>(1-3),'setosa','versicolor'))
iris.lda.confMat<-table(iris.lda$predicted,iris.lda$Species)
iris.lda.confMat
print(paste("Accuracy is::",
as.numeric(sum(diag(iris.lda.confMat))/sum(iris.lda.confMat)),sep= " "))
print(paste("Error is::",
1-as.numeric(sum(diag(iris.lda.confMat))/sum(iris.lda.confMat)),sep= " "))
```

```
> iris.lda.confMat<-table(iris.lda$predicted,iris.lda$Species)
> iris.lda.confMat

             setosa versicolor virginica
  setosa         50          0         0
  versicolor      0         25         0
  virginica       0         25        50
> print(paste("Accuracy is::",as.numeric(sum(diag(iris.lda.confMat))/sum(iris.lda.confMat)),se$
[1] "Accuracy is:: 0.833333333333333"
> print(paste("Error is::",1-as.numeric(sum(diag(iris.lda.confMat))/sum(iris.lda.confMat)),sep$
[1] "Error is:: 0.166666666666667"
```

We got better performance except that I presonnally set the threshold.

QDA

LDAhandles any type of data even if IV is not normally distributed.
But LDA does assume all predictor variables have the same variance
And the derivation relies on univariate Gaussian.

QDA allows multivariate scenario and uses multivariate gaussian.

References:
R.A. Fisher, ªThe Statistical Utilization of Multiple Measurements,º Annals of Eugenics, vol. 8, pp. 376-386, 1938
https://datascienceplus.com/how-to-perform-logistic-regression-lda-qda-in-r/
http://www2.ece.ohio-state.edu/~aleix/pami01.pdf
http://www.sci.utah.edu/~shireen/pdfs/tutorials/Elhabian_LDA09.pdf
https://rstudio-pubs-static.s3.amazonaws.com/336635_7611ceab3e324623b9a7bea8de2b3818.html

Well Known Package

```
require(MASS)
names(iris)
names(iris.lda)
mass_lda<-MASS::lda(Species~Sepal.Length+Sepal.Width+Petal.Length+Petal.Width,data=iris.lda)
mass_lda.predict<-predict(mass_lda,iris.lda[,1:4])
summary(mass_lda.predict)
str(mass_lda.predict)
mass_lda.predict
mass.lda.confmat <-  table(mass_lda.predict$class,iris.lda$Species)
mass.lda.confmat
print(paste("Accuracy is::",as.numeric(sum(diag(mass.lda.confmat))/sum(mass.lda.confmat)),sep= " "))
print(paste("Error is::",1-as.numeric(sum(diag(mass.lda.confmat))/sum(mass.lda.confmat)),sep= " "))
history(25)
```

```
> mass.lda.confmat <-  table(mass_lda.predict$class,iris.lda$Species)
> mass.lda.confmat

             setosa versicolor virginica
  setosa         50          0         0
  versicolor      0         48         1
  virginica       0          2        49
>
> print(paste("Accuracy is::",as.numeric(sum(diag(mass.lda.confmat))/sum(mass.lda.confmat)),sep= " "))
[1] "Accuracy is:: 0.98"
> print(paste("Error is::",1-as.numeric(sum(diag(mass.lda.confmat))/sum(mass.lda.confmat)),sep= " "))
[1] "Error is:: 0.02"
```

And the code for Visualizing is

```
> require(klaR)
Loading required package: klaR
Warning message:
package 'klaR' was built under R version 3.5.3
> partimat(Species~Sepal.Length+Sepal.Width+Petal.Length+Petal.Width,data=iris.lda,method="lda")
> 
```

require(klaR)

partimat(Species~Sepal.Length+Sepal.Width+Petal.Length+Petal.Width,data=iris.lda,method="lda")

Visualize DecisionBoundaries