Linear Discriminant Analysis

ACM

September 21, 2023

Breast Cancer Data Example

Each record represents follow-up data for one breast cancer case. These are consecutive patients seen by Dr. Wolberg since 1984, and include only those cases exhibiting invasive breast cancer and no evidence of distant metastases at the time of diagnosis.

The first 30 features (actually rows 4-33) are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image. A few of the images can be found at http://www.cs.wisc.edu/~street/images/

This data has two possible learning problems:

1) Predicting field 2, outcome: R = recurrent, N = nonrecurrent
    - Dataset should first be filtered to reflect a particular endpoint; e.g., recurrences before 24 months = positive, nonrecurrence beyond 24 months = negative.
    - 86.3% accuracy estimated accuracy on 2-year recurrence using previous version of this data.
2) Predicting Time To Recur (field 3 in recurrent records)
    - Estimated mean error 13.9 months using Recurrence Surface Approximation.

```
wdbct <- read.csv("wpbc.csv")
head(wdbct[, 1:5])
```

ID	Outcome	Time	radius_M	texture_M
119513	N	31	18.02	27.60
8423	N	61	17.99	10.38
842517	N	116	21.37	17.44
843483	N	123	11.42	20.38
843584	R	27	20.29	14.34
843786	R	77	12.75	15.29

```
X <- matrix(as.numeric(unlist(wdbct[, 4:32])), 198, 29)
K <- as.factor(wdbct[, 2])

library(MASS)
`?`(lda)
```

```
Linear Discriminant Analysis

Description:

     Linear discriminant analysis.
```

```
Usage:

     lda(x, ...)

     ## S3 method for class 'formula'
     lda(formula, data, ..., subset, na.action)

     ## Default S3 method:
     lda(x, grouping, prior = proportions, tol = 1.0e-4,
         method, CV = FALSE, nu, ...)

     ## S3 method for class 'data.frame'
     lda(x, ...)

     ## S3 method for class 'matrix'
     lda(x, grouping, ..., subset, na.action)

Arguments:

 formula: A formula of the form 'groups ~ x1 + x2 + ...'  That is, the
          response is the grouping factor and the right hand side
          specifies the (non-factor) discriminators.

    data: An optional data frame, list or environment from which
          variables specified in 'formula' are preferentially to be
          taken.

       x: (required if no formula is given as the principal argument.)
          a matrix or data frame or Matrix containing the explanatory
          variables.

grouping: (required if no formula principal argument is given.)  a
          factor specifying the class for each observation.

   prior: the prior probabilities of class membership.  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.

     tol: A tolerance to decide if a matrix is singular; it will reject
          variables and linear combinations of unit-variance variables
          whose variance is less than 'tol^2'.

  subset: An index vector specifying the cases to be used in the
          training sample.  (NOTE: If given, this argument must be
          named.)

na.action: A function to specify the action to be taken if 'NA's are
          found.  The default action is for the procedure to fail.  An
          alternative is 'na.omit', which leads to rejection of cases
          with missing values on any required variable.  (NOTE: If
          given, this argument must be named.)
```

```
  method: '"moment"' for standard estimators of the mean and variance,
          '"mle"' for MLEs, '"mve"' to use 'cov.mve', or '"t"' for
          robust estimates based on a t distribution.

      CV: If true, returns results (classes and posterior
          probabilities) for leave-one-out cross-validation. Note that
          if the prior is estimated, the proportions in the whole
          dataset are used.

      nu: degrees of freedom for 'method = "t"'.

     ...: arguments passed to or from other methods.
```

```r
WDB_DF <- data.frame(X = X, K = K)
fit <- lda(K ~ X, data = WDB_DF)
fit  # show results
```

```
Call:
lda(K ~ X, data = WDB_DF)

Prior probabilities of groups:
        N         R
0.7626263 0.2373737

Group means:
        X1       X2       X3        X4        X5        X6        X7         X8
N 17.10596 22.42980 112.7564  932.8278 0.1025366 0.1426256 0.1540870 0.08454682
R 18.39660 21.78191 121.6038 1089.5979 0.1031466 0.1427189 0.1631689 0.09393617
         X9        X10       X11      X12      X13     X14         X15
N 0.1942775 0.06315815 0.5804781 1.286778 4.082457 66.1747 0.006848285
R 0.1878596 0.06125128 0.6768170 1.192715 4.811000 83.2534 0.006484213
         X16        X17        X18        X19         X20      X21      X22
N 0.03129277 0.04145099 0.01530010 0.02079112 0.004033007 20.47113 30.31033
R 0.03089898 0.03849702 0.01445398 0.01979581 0.003838787 22.79106 29.58894
       X23      X24       X25       X26       X27       X28       X29
N 136.6176 1328.222 0.1434491 0.3669328 0.4349827 0.1769083 0.3265298
R 152.3319 1651.496 0.1454362 0.3592191 0.4421553 0.1847830 0.3133617

Coefficients of linear discriminants:
              LD1
X1  -3.022985e+00
X2  -1.218928e-01
X3   2.033923e-01
X4   1.341129e-02
X5   6.489718e+01
X6   6.629209e+00
X7  -1.062637e+01
X8   2.994518e+00
X9   8.507493e+00
X10 -1.933905e+02
X11 -4.775079e+00
X12 -1.263370e+00
X13  1.269176e+00
X14 -2.503344e-02
X15 -6.079053e+00
```

```
X16  5.250138e+01
X17 -5.180766e+01
X18 -7.094485e+01
X19  6.461521e+01
X20  2.919723e+02
X21  1.416718e+00
X22  9.515986e-02
X23 -9.520105e-02
X24 -3.777199e-03
X25  9.506298e+00
X26 -6.768119e+00
X27  8.173294e+00
X28 -8.152277e+00
X29 -1.006818e+01
```

Next, we'll use `CV = TRUE` which will return results (classes and posterior probabilities) for leave-one-out cross-validation.

```
WDB_DF <- data.frame(X=X,K=K)
fit <- lda(K ~ X,
           data=WDB_DF,
           CV=TRUE)
head( fit$posterior) # show results
```

N	R
0.9536968	0.0463032
0.8967891	0.1032109
0.9544604	0.0455396
0.9628922	0.0371078
0.6719942	0.3280058
0.7073578	0.2926422

Now let's fit a logistic regression model to the data:

```
log_fit <- glm(K ~ X,family=binomial, data=WDB_DF)
summary(log_fit) # show results
```

```
##
## Call:
## glm(formula = K ~ X, family = binomial, data = WDB_DF)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  6.157e+00  1.200e+01   0.513   0.6079
## X1          -6.581e+00  3.196e+00  -2.059   0.0395 *
## X2          -2.633e-01  1.673e-01  -1.574   0.1156
## X3           7.142e-01  4.745e-01   1.505   0.1323
## X4           1.689e-02  1.114e-02   1.515   0.1297
## X5           1.175e+02  6.276e+01   1.872   0.0612 .
## X6          -1.110e+01  2.935e+01  -0.378   0.7054
## X7          -2.269e+01  2.017e+01  -1.125   0.2606
## X8          -7.778e+00  3.724e+01  -0.209   0.8345
## X9           1.743e+01  1.875e+01   0.929   0.3527
## X10         -2.225e+02  1.203e+02  -1.850   0.0643 .
```

```
## X11         -6.101e+00  7.926e+00  -0.770   0.4415
## X12         -2.811e+00  1.251e+00  -2.247   0.0246 *
## X13          1.717e+00  1.056e+00   1.626   0.1040
## X14         -3.220e-02  3.338e-02  -0.965   0.3347
## X15          1.129e+02  2.294e+02   0.492   0.6225
## X16          9.849e+01  6.074e+01   1.622   0.1049
## X17         -1.043e+02  5.992e+01  -1.742   0.0816 .
## X18         -1.887e+02  1.277e+02  -1.478   0.1393
## X19          1.571e+02  7.410e+01   2.120   0.0340 *
## X20          4.082e+02  3.723e+02   1.097   0.2729
## X21          2.132e+00  1.155e+00   1.845   0.0650 .
## X22          2.348e-01  1.562e-01   1.503   0.1327
## X23         -1.383e-01  1.085e-01  -1.275   0.2024
## X24         -6.931e-03  5.937e-03  -1.167   0.2431
## X25          1.049e+01  3.146e+01   0.333   0.7389
## X26         -1.108e+01  6.949e+00  -1.595   0.1107
## X27          1.473e+01  6.496e+00   2.268   0.0233 *
## X28         -6.135e+00  1.614e+01  -0.380   0.7038
## X29         -2.016e+01  1.155e+01  -1.745   0.0809 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 217.02  on 197  degrees of freedom
## Residual deviance: 160.00  on 168  degrees of freedom
## AIC: 220
##
## Number of Fisher Scoring iterations: 6
```

We'll now do leave one out CV:

```
pred <- NULL
n <- length(K)
for(i in 1:n){
  bK <- K
  bK[i] <- NA
  b_log_fit <- glm(bK ~ X,family=binomial)
  pred_vals <- predict.glm( b_log_fit, newdata=data.frame(X),
                            type="response")
  pred <- c(pred,pred_vals[i])
}
```

We'll now do quadratic discriminant analysis

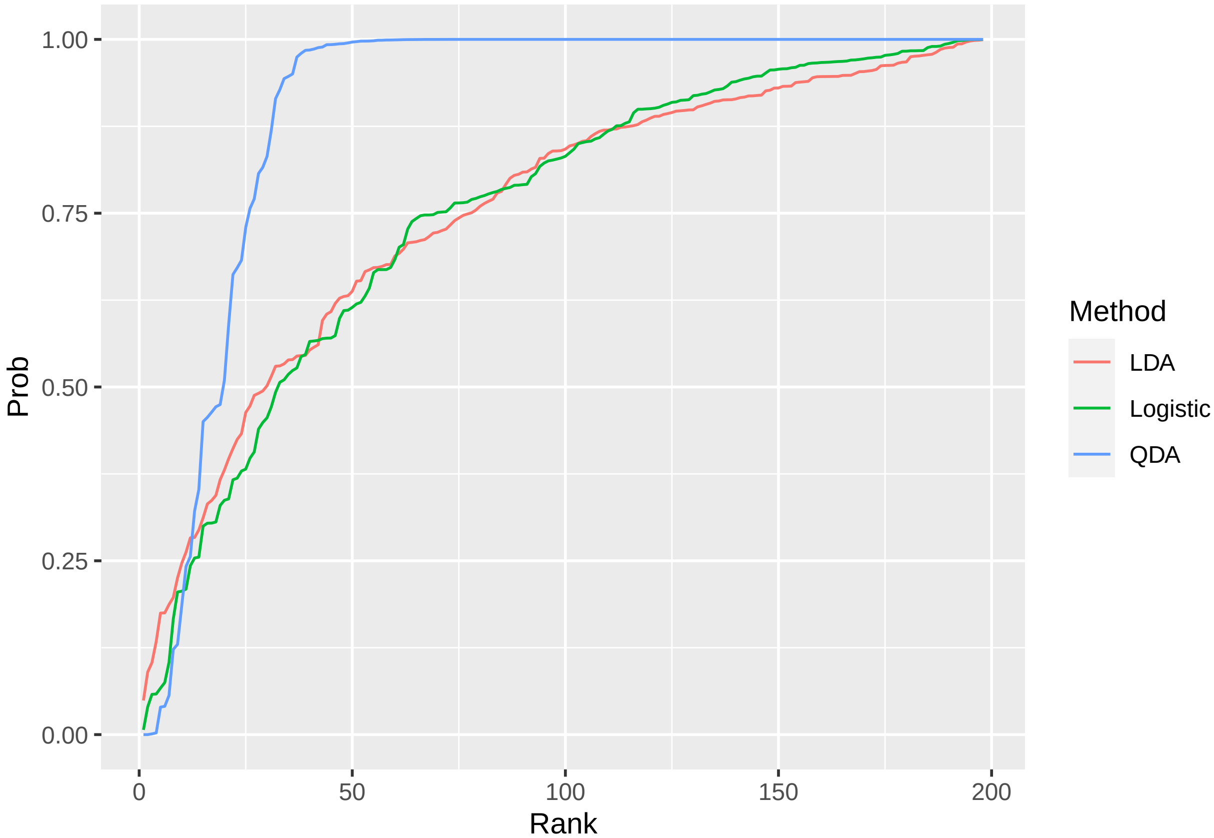
```
q_fit <- qda(K ~ X,
             data=WDB_DF,
             CV=TRUE)
head( q_fit$posterior) # show results
```

N	R
1.0000000	0.0000000
1.0000000	0.0000000
1.0000000	0.0000000
1.0000000	0.0000000

N	R
1.0000000	0.0000000
0.9991475	0.0008525

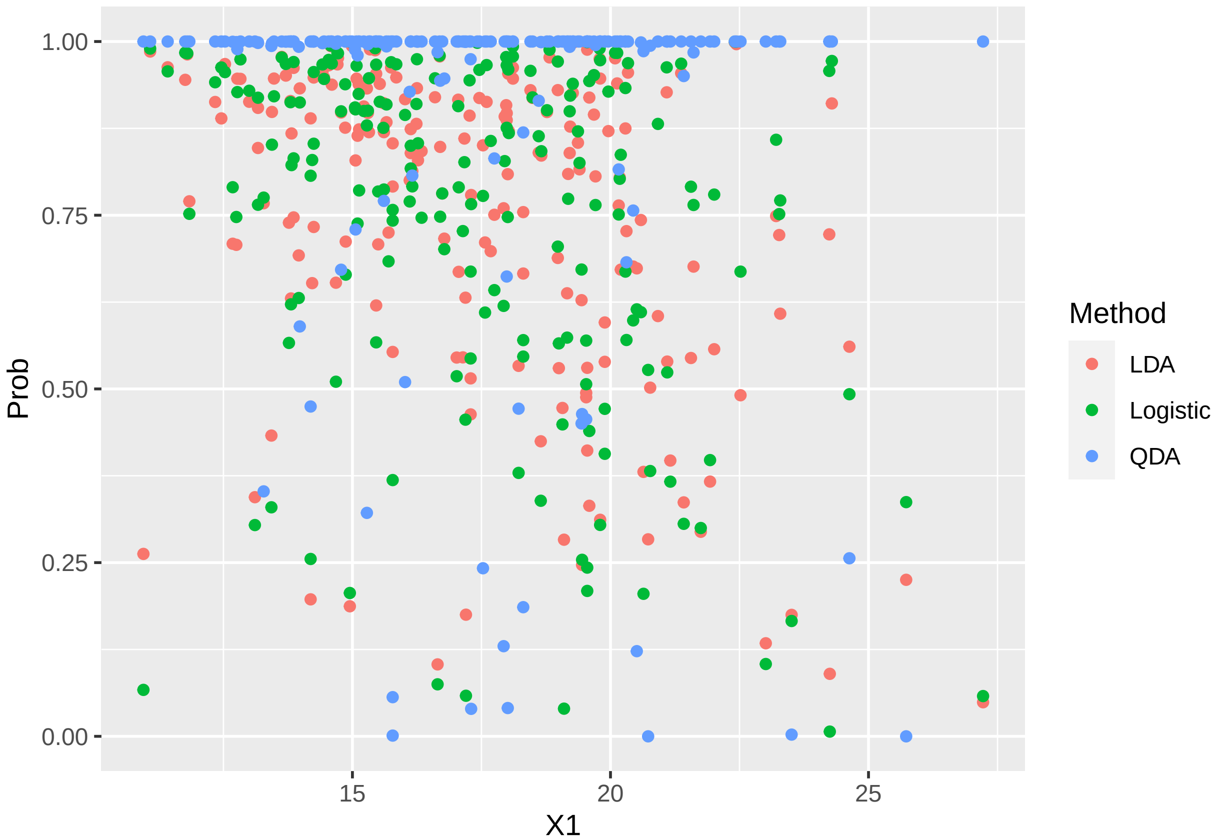
```
CV_prob <- data.frame( Rank = rep(1:198, 3),
                       Prob = c(sort(fit$posterior[,1]), sort( 1-pred),
                                sort(q_fit$posterior[,1])),
                       Method = rep(c("LDA", "Logistic", "QDA"), each = 198) )

ggplot(data = CV_prob, aes(x = Rank, y= Prob, color = Method)) + geom_line()
```




```
CV_prob <- data.frame( X1 = rep( WDB_DF$X.1, 3),
                       Prob = c(fit$posterior[,1], 1-pred,
                                q_fit$posterior[,1]),
                       Method = rep(c("LDA", "Logistic", "QDA"), each = 198) )

ggplot(data = CV_prob, aes(x = X1, y= Prob, color = Method)) + geom_point()
```



Assess the accuracy of the prediction percent correct for each category of K.

```
ct <- table(WDB_DF$K, fit$class)
ct
```

First for LDA:

/	N	R
N	135	16
R	34	13

```
prop.table(ct)
```

/	N	R
N	0.6818182	0.0808081
R	0.1717172	0.0656566

```
log_pred <- 1*I(pred>=0.5)
lt <- table(K,log_pred)
lt
```

Second for logistic:

K/log_pred	0	1
N	132	19
R	34	13

```
prop.table(lt)
```

K/log_pred	0	1
N	0.6666667	0.0959596
R	0.1717172	0.0656566

```
qt <- table(WDB_DF$K, q_fit$class)
qt
```

Third for QDA:

/	N	R
N	137	14
R	42	5

```
prop.table(qt)
```

/	N	R
N	0.6919192	0.0707071
R	0.2121212	0.0252525

Total percent correct for all three methods:

```
LDA_acc <- 1-sum(diag(prop.table(ct)))
LOG_acc <- 1-sum(diag(prop.table(lt)))
QDA_acc <- 1-sum(diag(prop.table(qt)))
```

Let's take it home:

```
res <- matrix(c(LDA_acc, LOG_acc, QDA_acc))
rownames(res) <- c("LDA", "Logistic Reg", "QDA")
colnames(res) <- "N-fold EPE"
res
```

	N-fold EPE
LDA	0.2525253
Logistic Reg	0.2676768
QDA	0.2828283