Lecture 10: LDA and Logistic Regression

Wenbin Lu

Department of Statistics
North Carolina State University

Fall 2019

Outline

- Two Popular Linear Models for Classification
  - Linear Discriminant Analysis (LDA)
  - Logistic Regression Models
- Take-home message:
  - Both LDA and Logistic regression models rely on the linear-odd assumption, indirectly or directly. However, they estimate the coefficients in a different manner.

Linear Classifier

Linear methods: The decision boundary is linear.
Common linear classification methods:

- Linear regression methods (covered in Lecture 9)
- Linear log-odds (logit) models
  - Linear logistic models
  - Linear discriminant analysis (LDA)
- separating hyperplanes (introduced later)
  - perceptron model (Rosenblatt 1958)
  - Optimal separating hyperplane (Vapnik 1996) – SVMs

From now on, we assume equal costs (by default).

Odds, Logit, and Linear Odds Models Linear

Some terminologies

- Call the term $\frac{\Pr(Y=1|\mathbf{X}=\mathbf{x})}{\Pr(Y=0|\mathbf{X}=\mathbf{x})}$ is called odds
- Call $\log \frac{\Pr(Y=1|\mathbf{X}=\mathbf{x})}{\Pr(Y=0|\mathbf{X}=\mathbf{x})}$ log of the odds, or logit function

Linear odds models assume: the logit is linear in $\mathbf{x}$, i.e.,

$$\log \frac{\Pr(Y=1|\mathbf{X}=\mathbf{x})}{\Pr(Y=0|\mathbf{X}=\mathbf{x})} = \beta_0 + \beta_1^T \mathbf{x}.$$

Examples: LDA, Logistic regression

Classifier Based on Linear Odd Models

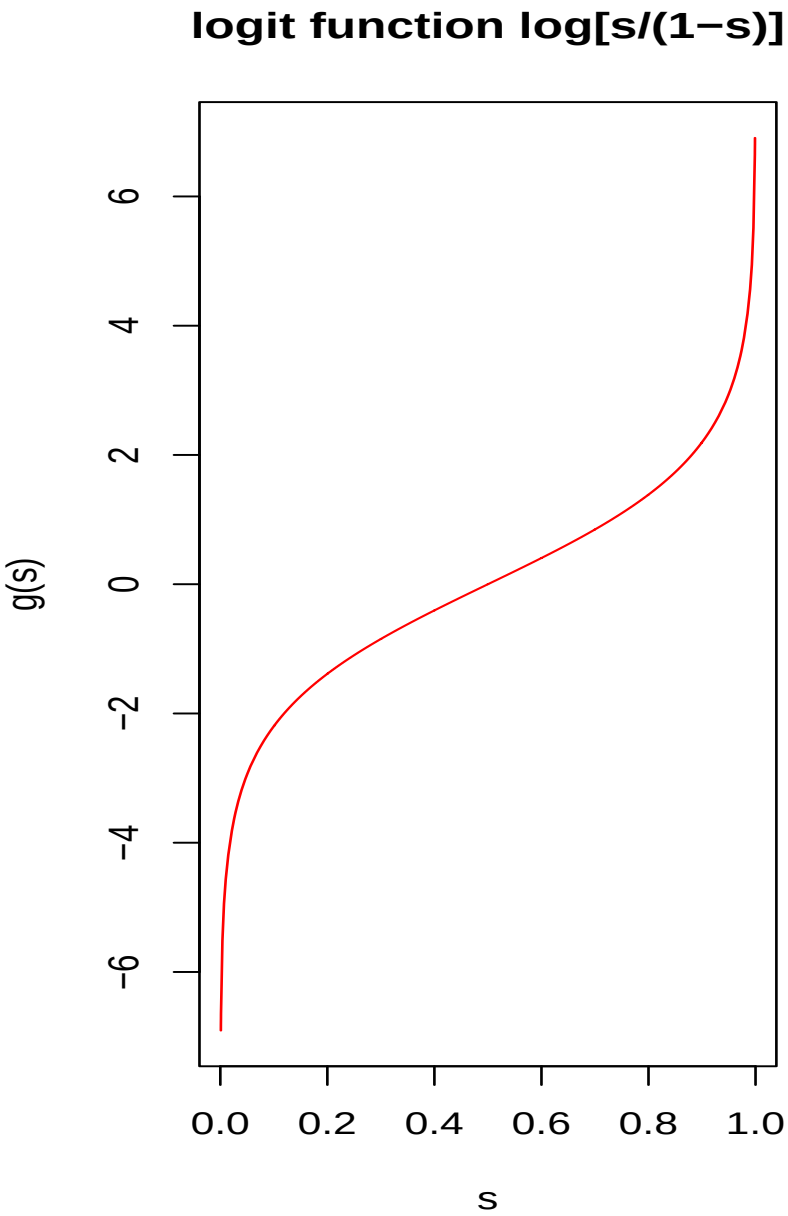
From the linear odds, we can obtain posterior class probabilities

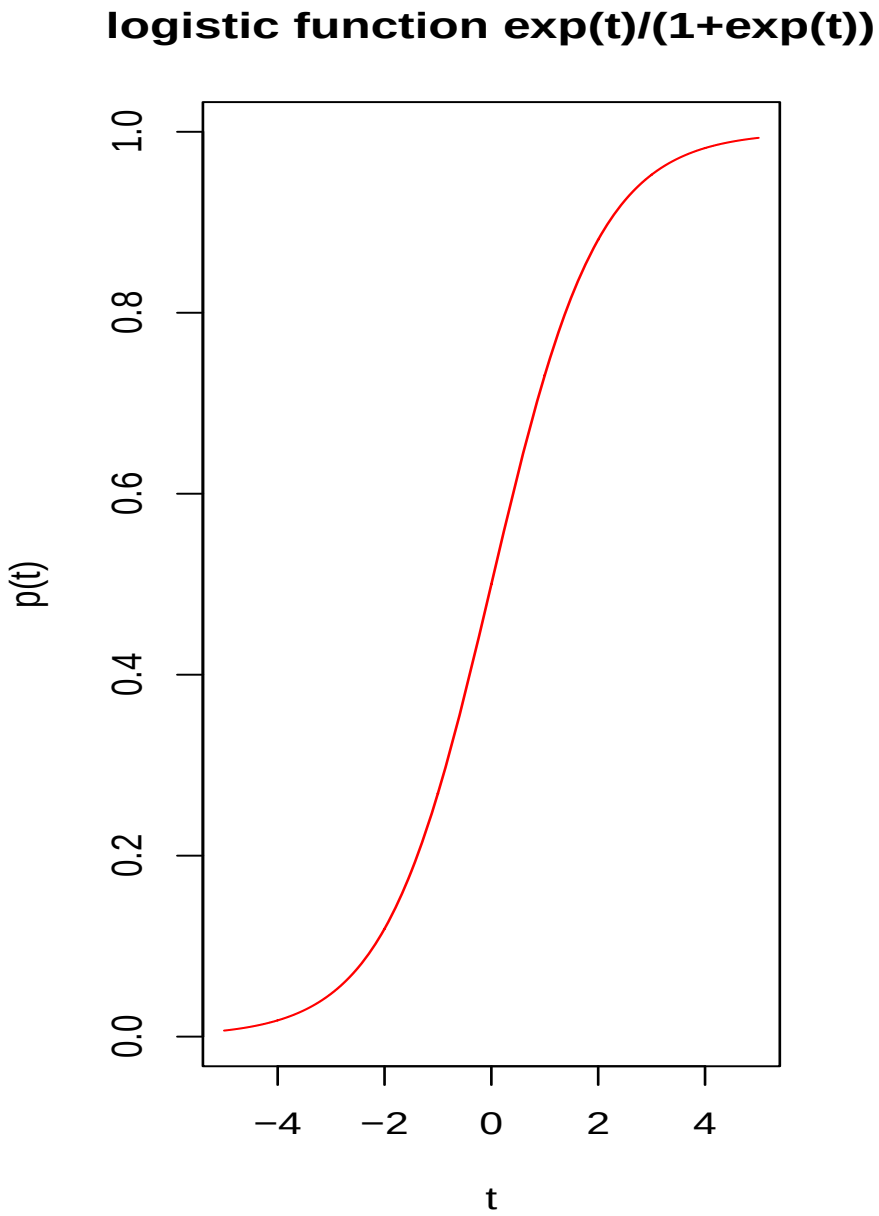
$$\begin{aligned} \Pr(Y=1|\mathbf{x}) &= \frac{\exp\left(\beta_0+\beta_1^T\mathbf{x}\right)}{1+\exp\left(\beta_0+\beta_1^T\mathbf{x}\right)} \\ \Pr(Y=0|\mathbf{x}) &= \frac{1}{1+\exp\left(\beta_0+\beta_1^T\mathbf{x}\right)} \end{aligned}$$

Assuming equal costs, the decision boundary is given by

$$\{\mathbf{x}|P(Y=1|\mathbf{x})=0.5\}=\{\mathbf{x}|\beta_0+\beta_1^T\mathbf{x}=0\},$$

which can be interpreted as "zero log-odds"


logit function log[s/(1−s)]
g(s)
6
4
2
0
−2
−4
−6
0.0
0.2
0.4
0.6
0.8
1.0
s


logistic function exp(t)/(1+exp(t))
p(t)
1.0
0.8
0.6
0.4
0.2
0.0
−4
−2
0
2
4
t

Linear Discriminant Analysis (LDA)

LDA assumes

- Assume each class density is multivariate Gaussian, i.e.,

$$\mathbf{X}|Y = j \sim N(\mu_j, \Sigma_j), \quad j = 0, 1.$$

- Equal covariance assumption

$$\Sigma_j = \Sigma, \quad j = 0, 1.$$

In other words,

- both classes are from Gaussian and they have the same covariance matrix.

Linear Discriminant Function

Under the mixture Gaussian assumption, the log-odd is

$$\begin{aligned} & \log \frac{\Pr(Y=1|\mathbf{X}=\mathbf{x})}{\Pr(Y=0|\mathbf{X}=\mathbf{x})} \\ = \quad & \log \frac{\pi_1}{\pi_0} - \frac{1}{2}(\mu_1+\mu_0)^T\Sigma^{-1}(\mu_1-\mu_0) + \mathbf{x}^T\Sigma^{-1}(\mu_1-\mu_0) \end{aligned}$$

Under equal costs, the LDA classifies to "1" if and only if

$$\left[\log \frac{\pi_1}{\pi_0} - \frac{1}{2}(\mu_1+\mu_0)^T\Sigma^{-1}(\mu_1-\mu_0)\right] + \mathbf{x}^T\Sigma^{-1}(\mu_1-\mu_0) > 0.$$

It has a linear boundary $\{\mathbf{x} : \beta_0 + \mathbf{x}^T\boldsymbol{\beta}_1 = 0\}$, with

$$\begin{aligned} \beta_0 &= \log \frac{\pi_1}{\pi_0} - \frac{1}{2}(\mu_1+\mu_0)^T\Sigma^{-1}(\mu_1-\mu_0), \\ \boldsymbol{\beta}_1 &= \Sigma^{-1}(\mu_1-\mu_0). \end{aligned}$$

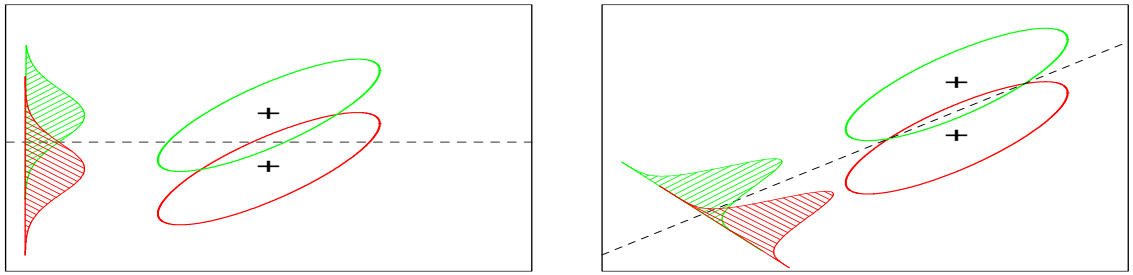


Figure 4.9: *Although the line joining the centroids defines the direction of greatest centroid spread, the projected data overlap because of the covariance (left panel). The discriminant direction minimizes this overlap for Gaussian data (right panel).*

Parameter Estimation in LDA

In practice, $\pi_1, \pi_0, \mu_0, \mu_1, \Sigma$ are unknown

- We estimate the parameters from the training data, using MLE or the moment estimator
  - $\hat{\pi}_j = n_j/n$, where n_k is the size size of class j.
  - $\hat{\mu}_j = \sum_{Y_i=j} \mathbf{x}_i/n_j$ for $j = 0, 1$.
  - The sample covariance matrix is

$$S_j = \frac{1}{n_j - 1} \sum_{Y_i=j} (\mathbf{x}_i - \hat{\mu}_j)(\mathbf{x}_i - \hat{\mu}_j)^T$$

  - (Unbiased) pooled sample covariance is a weighted average

$$\begin{aligned} \hat{\Sigma} &= \frac{n_0 - 1}{(n_0 - 1) + (n_1 - 1)} S_0 + \frac{n_1 - 1}{(n_0 - 1) + (n_1 - 1)} S_1 \\ &= \sum_{j=0}^{1} \sum_{Y_i=j} (\mathbf{x}_i - \hat{\mu}_j)(\mathbf{x}_i - \hat{\mu}_j)^T/(n - 2) \end{aligned}$$

R code for LDA Fitting (I)

There are two ways to call the function "lda". The first way is to use a formula and an optional data frame.

```
library(MASS)
lda(formula, data,subset)
```

Arguments:

- *formula*: the form "groups $\sim x_1 + x_2 + \ldots$", where the response is the grouping factor and the right hand side specifies the (non-factor) discriminators.
- *data*: data frame from which variables specified
- *subset*: An index vector specifying the cases to be used in the training sample.

Output:

- an object of class "lda" with multiple components

R code for LDA Fitting (II)

The second way is to use a matrix and group factor as the first two arguments.

```
library(MASS)
lda(x, grouping, prior = proportions, CV = FALSE)
```

Arguments:

- *x*: a matrix or data frame or Matrix containing predictors.
- *grouping*: 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.

Output:

- If CV = TRUE, the return value is a list with components "class" (the maximum a posteriori probability (MAP) classification, a factor) and "posterior" (posterior probabilities for the classes).

R code for LDA Prediction

We use the "predict" or "predict.lda" function to classify multivariate observations with lda

```
predict(object, newdata, ...)
```

Arguments:

- *object*: object of class "lda"
- *newdata*: data frame of cases to be classified or, if "object" has a formula, a data frame with columns of the same names as the variables used.

Output:

- a list with the components "class" (the MAP classification, a factor) and "posterior" (posterior probabilities for the classes)

Fisher's Iris Data (Three-Classification Problems)

Fisher (1936) "The use of multiple measurements in taxonomic problems".

- Three species: Iris setosa, Iris versicolor, Iris verginica
- Four features: the length and the width of the sepals and petals, in centimeters.
- 50 samples from each species

In the following analysis, we randomly select 50% of the data points as the training set, and the rest as the test set.

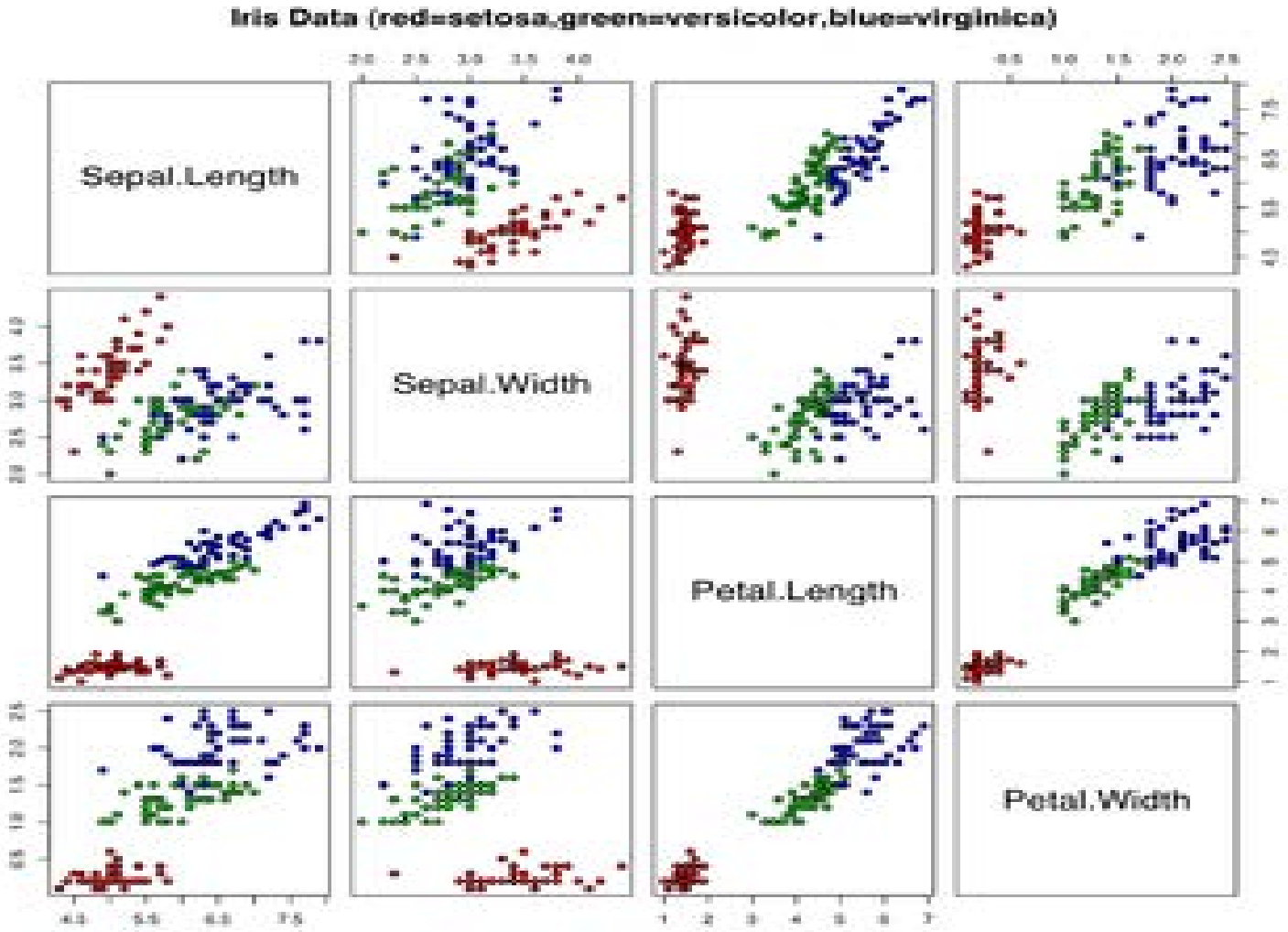

Iris Data (red=setosa,green=versicolor,blue=virginica)
Sepal.Length
Sepal.Width
Petal.Length
Petal.Width

Illustration 1

```
Iris <- data.frame(rbind(iris3[,,1], iris3[,,2],
   iris3[,,3]), Sp = rep(c("s","c","v"), rep(50,3)))
train <- sample(1:150, 75)
table(Iris$Sp[train])
z <- lda(Sp ~ ., Iris, prior = c(1,1,1)/3, subset = train)
```

Training Error and Test Error

```
#training error rate
ytrain <- predict(z, Iris[train, ])$class
table(ytrain, Iris$Sp[train])
train_err <- mean(ytrain!=Iris$Sp[train])
#test error rate
ytest <- predict(z, Iris[-train, ])$class
table(ytest, Iris$Sp[-train])
test_err <- mean(ytest!=Iris$Sp[-train])
```

Illustration 2

```
tr <- sample(1:50, 25)
train <- rbind(iris3[tr,,1], iris3[tr,,2], iris3[tr,,3])
test <- rbind(iris3[-tr,,1], iris3[-tr,,2], iris3[-tr,,3])
cl <- factor(c(rep("s",25), rep("c",25), rep("v",25)))
z <- lda(train, cl)
ytrain <- predict(z, train)$class
table(ytrain, cl)
train_err <- mean(ytrain!=cl)
ytest <- predict(z, test)$class
table(ytest, cl)
test_err <- mean(ytest!=cl)
```

Logistic Regression

Model assumption: the log-odd is linear in $\mathbf{x}$.

$$\log \frac{\Pr(Y=1|\mathbf{X}=\mathbf{x})}{\Pr(Y=0|\mathbf{X}=\mathbf{x})} = \beta_0 + \boldsymbol{\beta}_1^T \mathbf{x}.$$

Define

$$p(\mathbf{x};\boldsymbol{\beta}) = \frac{\exp(\beta_0 + \boldsymbol{\beta}_1^T \mathbf{x})}{1+\exp(\beta_0 + \boldsymbol{\beta}_l^T \mathbf{x})}$$

Write $\boldsymbol{\beta} = (\beta_0, \boldsymbol{\beta}_1)$. The class probabilities can be calculated as

$$\begin{aligned} p_1(\mathbf{x}) &= \Pr(Y=1|\mathbf{x}) = p(\mathbf{x};\boldsymbol{\beta}), \\ p_0(\mathbf{x}) &= \Pr(Y=0|\mathbf{x}) = 1 - p(\mathbf{x};\boldsymbol{\beta}). \end{aligned}$$

The classification boundary is given by: $\{\mathbf{x} : \beta_0 + \boldsymbol{\beta}_1^T \mathbf{x} = 0\}$.

Model Interpretation

- Denote $\mu = E(Y|\mathbf{X}) = P(Y = 1|\mathbf{X})$
- By assuming

$$g(\mu) = \log[\mu/(1-\mu)] = \beta_0 + \beta_1^T \mathbf{X},$$

  the logit g connects μ with the linear predictor $\beta_0 + \beta_1^T \mathbf{X}$.
- We call g the link function
- $Var(Y|\mathbf{X}) = \mu(1-\mu)$.

Interpretation of β_j

In logistic regression,

$$\text{odds}_{X_j} = \frac{\text{odds}(\ldots, X_j = x+1, \ldots)}{\text{odds}(\ldots, X_j = x, \ldots)} = e^{\beta_j}.$$

- If $X_j = 0$ or 1, then odds for group with $X_j = 1$ are e^{β_j} higher than for group with $X_j = 0$, with other parameters fixed.

For rare diseases,

- when incidence is rare, $Pr(Y = 0) \approx 1$, then odds $\approx Pr(Y = 1)$

$$e^{\beta_j} = \text{odds}_{X_j} \approx \frac{\Pr(\ldots, X_j = x+1, \ldots)}{\Pr(\ldots, X_j = x, \ldots)}.$$

- in a cancer study, an log-OR of 5 means that smokers are $e^5 \approx 150$ times more likely to develop the cancer

Maximum Likelihood Estimate (MLE) for Logistic Models

The joint conditional likelihood of y_i given $\mathbf{x}_i$ is

$$l(\beta) = \sum_{i=1}^{n} \log p_{y_i}(\mathbf{x}; \beta),$$

where

$$p_y(\beta; \mathbf{x}) = p(\mathbf{x}; \beta)^y [1 - p(\mathbf{x}; \beta)]^{1-y}.$$

In details,

$$\begin{aligned} l(\beta) &= \sum_{i=1}^{n} \{y_i \log p(\mathbf{x}_i; \beta) + (1 - y_i) \log[1 - p(\mathbf{x}_i, \beta)]\} \\ &= \sum_{i=1}^{n} \{y_i(\beta_{10} + \beta_1^T \mathbf{x}_i) - \log[1 + \exp(\beta_{10} + \beta_1^T \mathbf{x}_i)]. \end{aligned}$$

.

Score Equations

For simplicity, now assume $\mathbf{x}_i$ has 1 in its first component.

$$\frac{\partial l(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = \sum_{i=1}^{n} \mathbf{x}_i [y_i - p(\mathbf{x}_i; \boldsymbol{\beta})] = 0,$$

Totally (d+1) nonlinear equations. The first equation

$$\sum_{i=1}^{n} y_i = \sum_{i=1}^{n} p(\mathbf{x}_i; \boldsymbol{\beta}),$$

expected number of 1's = *observed* number in sample.

$$\begin{aligned} \mathbf{y} &= [y_1, \cdots, y_n]^T \\ \mathbf{p} &= [p(\mathbf{x}_1; \boldsymbol{\beta}^{\text{old}}), \cdots, p(\mathbf{x}_n; \boldsymbol{\beta}^{\text{old}})]^T. \\ W &= \text{diag}\left\{ p(\mathbf{x}_i; \boldsymbol{\beta}^{\text{old}})(1 - p(\mathbf{x}_i; \boldsymbol{\beta}^{\text{old}})) \right\}. \end{aligned}$$

Newton-Raphson Algorithm

The second-derivative (Hessian) matrix

$$\frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T} = -\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^T p(\mathbf{x}_i; \beta)[1 - p(\mathbf{x}_i; \beta)].$$

1. Choose an initial value β^0
2. Update β by

$$\beta^{\text{new}} = \beta^{\text{old}} - \left[\frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T}\right]^{-1}_{\beta^{\text{old}}} \frac{\partial l(\beta)}{\partial \beta}_{\beta^{\text{old}}}$$

Using matrix notations, we have

$$\frac{\partial l(\beta)}{\partial \beta} = \mathbf{X}^T(\mathbf{y} - \mathbf{p}), \quad \frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T} = -\mathbf{X}^T W \mathbf{X}.$$

Iteratively Re-weighted Least Squares (IRLS)

Newton-Raphson step

$$\begin{aligned}\beta^{\text{new}} &= \beta^{\text{old}} + (\mathbf{X}^T W \mathbf{X})^{-1} \mathbf{X}^T (\mathbf{y} - \mathbf{p}) \\ &= (\mathbf{X}^T W \mathbf{X})^{-1} \mathbf{X}^T W \left(\mathbf{X}\beta^{\text{old}} + W^{-1}(\mathbf{y} - \mathbf{p}) \right) \\ &= (\mathbf{X}^T W \mathbf{X})^{-1} \mathbf{X}^T W \mathbf{z},\end{aligned}$$

where we defined the *adjusted response*

$$\mathbf{z} = \mathbf{X}\beta^{\text{old}} + W^{-1}(\mathbf{y} - \mathbf{p})$$

- Repeatedly solve weighted least squares till convergence.

$$\beta^{\text{new}} = \arg\min_{\beta} (\mathbf{z} - \mathbf{X}\beta)^T W (\mathbf{z} - \mathbf{X}\beta),$$

  Weight W, response $\mathbf{z}$, and $\mathbf{p}$ change in each iteration.
- The algorithm can be generalized to $K \geq 3$ case.

Quadratic Approximations

$\widehat{\beta}$ satisfies a self-consistency relationship: it solves a weighted least square fit with response

$$z_i = \mathbf{x}_i^T \widehat{\beta} + \frac{(y_i - \hat{p}_i)}{\hat{p}_i(1 - \hat{p}_i)}$$

and the weight $w_i = \hat{p}_i(1 - \hat{p}_i)$.

- $\beta = 0$ seems to be a good starting value
- Typically the algorithm converges, but it is never guaranteed.
- The weighted residual sum-of-squared is **Pearson chi-square** statistic

$$\sum_{i=1}^{n} \frac{(y_i - \hat{p}_i)^2}{\hat{p}_i(1 - \hat{p}_i)},$$

a quadratic approximation to the deviance.

Statistical Inferences

Using the weighted least squares formulation, we have

- Asymptotic likelihood theory says: if the model is correct, $\widehat{\beta}$ is consistent
- Using central limit theorem, the distribution of $\widehat{\beta}$ converges to $N(\beta, (\mathbf{X}^T W \mathbf{X})^{-1})$
- Model building is costly due to iterations, popular shortcuts:
  - For inclusion of a term, use *Rao score test*.
  - For exclusion of a term, use *Wald test*.

  Neither of these two algorithms require iterative fitting, and are based on the maximum likelihood fit of the current model.

R Code for Logistic Regression

```
logist <- glm(formula, family, data, subset, ...)
predict(logist)
```

Arguments:

- *formula*: an object of class "formula"
- *family*: a description of the error distribution and link function to be used in the model.
- *data*: an optional data frame, list or environment containing the variables in the model.
- *subset*: an optional vector specifying a subset of observations to be used in the fitting process.

Output:

- returns an object of class inheriting from "glm" and "lm"

R Code for Logistic Prediction

```
logist <- glm(y~x, data, family=binomial(link="logit"))
predict(logist, newdata)
summary(logist)
anova(logist)
```

- The function "predict" gives the predicted values, newdata is a data frame
- The function "summary" is used to obtain or print a summary of the results
- The function "anova" produces an analysis of variance table.

Relationship between LDA and Least Squares (LS)

For two-class problems, both LDA and least squares fit a linear boundary $\beta_0 + \boldsymbol{\beta}^T \mathbf{x}$. Their solutions have the following relationship:

- The least square regression coefficient $\hat{\boldsymbol{\beta}}$ is proportional to the LDA direction, i.e.,

$$\hat{\boldsymbol{\beta}} \propto \hat{\Sigma}^{-1}(\hat{\boldsymbol{\mu}}_1 - \hat{\boldsymbol{\mu}}_0),$$

  where $\hat{\Sigma}$ is the pooled sample covariance matrix, and $\hat{\boldsymbol{\mu}}_k$ is the sample mean of points from class $k, k = 0, 1$. In other words, their slope coefficients are identical, up to a scalar multiple. (Exercise 4.2 in textbook)
- The LS intercept $\hat{\beta}_0$ is generally different from that of LDA, unless $n_1 = n_0$.

In general, they have different decision rules (unless $n_1 = n_0$).

Common Feature of LDA and Logistic Regression Models

For both LDA and logistic regression, the logit has a linear form

$$\log \frac{\Pr(Y=1|\mathbf{x})}{\Pr(Y=0|\mathbf{x})} = \beta_0 + \boldsymbol{\beta}_1^T \mathbf{x}.$$

Or equivalently, for both estimators, their posterior class probability can be expressed in the form of

$$\Pr(Y=1|\mathbf{x}) = \frac{\exp\left(\beta_0 + \boldsymbol{\beta}_1^T \mathbf{x}\right)}{1+\exp\left(\beta_0 + \boldsymbol{\beta}_1^T \mathbf{x}\right)}.$$

They have exactly same forms. Are they same estimators?

Major Differences of LDA and Logistsics Regresion

Main differences of two estimators include

- Where is the linear-logit from?
- What assumptions are made on the data distribution? (Difference on data distribution)
- How to estimate the linear coefficients? (Difference in parameter estimation)
- Any assumption on the marginal density of $\mathbf{X}$? (flexibility of model)

Where is Linear-logit from?

For LDA, the linear logit is due to the equal-covariance Gaussian assumption on data

$$\begin{aligned}\log\frac{\Pr(Y=1|\mathbf{x})}{\Pr(Y=0|\mathbf{x})} &= \log\frac{\pi_1}{\pi_0}-\frac{1}{2}(\boldsymbol{\mu}_1+\boldsymbol{\mu}_0)^T\Sigma^{-1}(\boldsymbol{\mu}_1+\boldsymbol{\mu}_0)\\ &\quad +\mathbf{x}^T\Sigma^{-1}(\boldsymbol{\mu}_1-\boldsymbol{\mu}_0)\\ &= \beta_0+\boldsymbol{\beta}_1^T\mathbf{x}\end{aligned}$$

For Logistic model, the linear logit is due to construction

$$\log\frac{\Pr(Y=1|\mathbf{x})}{\Pr(Y=0|\mathbf{x})}=\beta_0+\boldsymbol{\beta}_1^T\mathbf{x}$$

Difference in Marginal Density Assumption

The assumptions on $\Pr(\mathbf{X})$:

- The logistic model leaves the marginal density of $\mathbf{X}$ arbitrary and unspecified.
- The LDA model assumes a Gaussian density

$$\Pr(\mathbf{X}) = \sum_{j=0}^{1} \pi_j \phi(\mathbf{X}; \boldsymbol{\mu}_j, \Sigma)$$

Conclusion: The logistic model makes less assumptions about the data, and hence is more general.

Difference in Parameter Estimation

Logistic regression

- Maximizing the conditional likelihood, the multinomial likelihood with probabilities $\Pr(Y = k|\mathbf{X})$
- The marginal density Pr(X) is totally ignored (fully nonparametric using the empirical distribution function which places $1/n$ at each observation)

LDA

- Maximizing the full log-likelihood based on the joint density

$$\Pr(\mathbf{X}, Y = j) = \phi(\mathbf{X}; \mu_j, \Sigma)\pi_j,$$

  Standard MLE theory leads to estimators $\hat{\boldsymbol{\mu}}_j, \hat{\Sigma}, \hat{\pi}_j$

- Marginal density does play a role

More Comments

- LDA is easier to compute than logistic regression.
- If the true $f_k(\boldsymbol{x})$'s are Gaussian, LDA is better.
  - Logistic regression may lose efficiency around 30% asymptotically in error rate (by Efron 1975)
- Robustness?
  - LDA uses all the points to estimate the covariance matrix; more information but not robust against outliers
  - Logistic regression down-weights points far from decision boundary (recall the weight is $p_i(1-p_i)$; more robust and safer
- In practice, these two methods often give similar results (for approximately normal distributed data)

Two-dimensional Linear Example

Consider the following scenarios for a two-class problem:

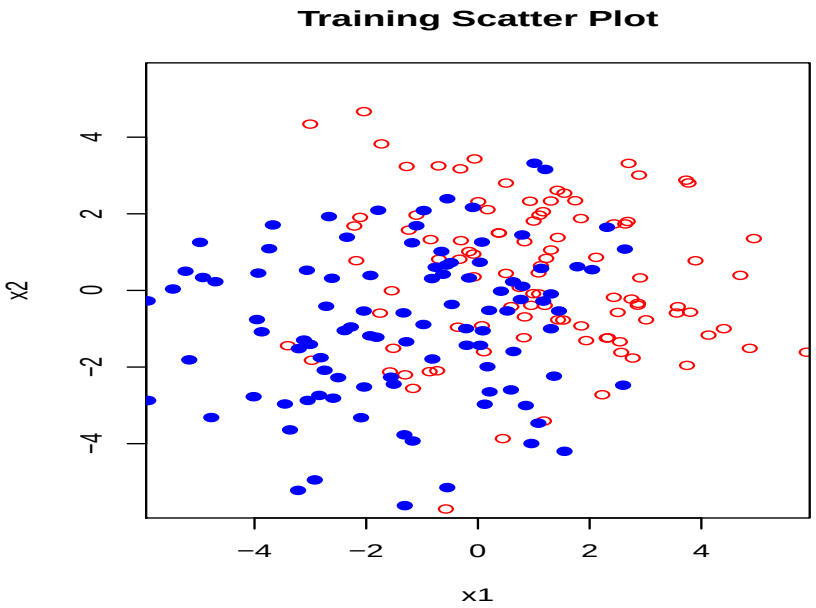
- $\pi_1 = \pi_0 = 0.5$.
- Class 1: $\mathbf{X} \sim N_2((1,1)^T, 4\mathbf{I})$
- Class 2: $\mathbf{X} \sim N_2((-1,-1)^T, 4\mathbf{I})$

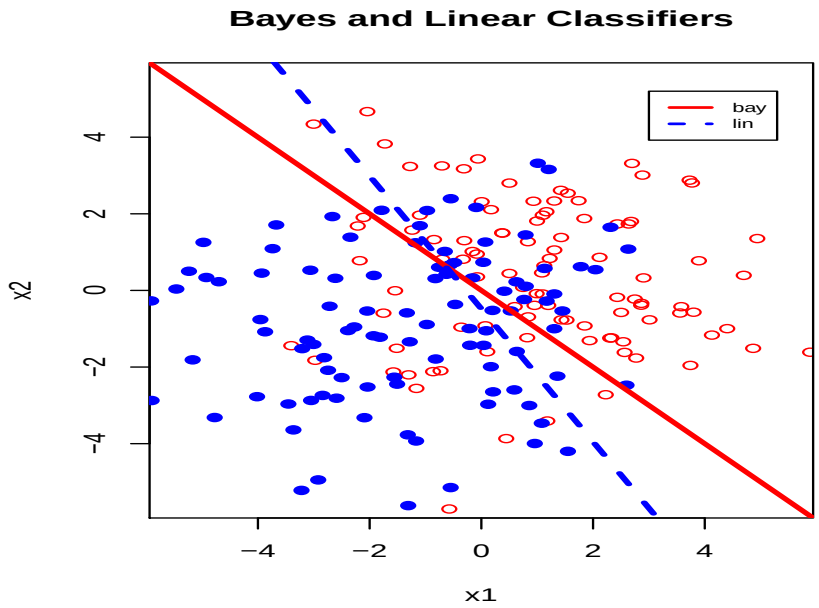
The Bayes boundary is

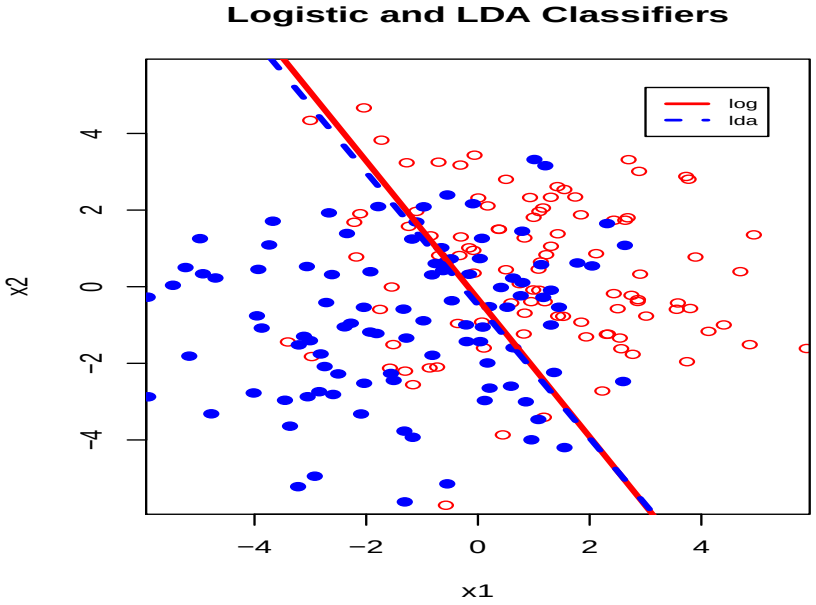
$$\{\mathbf{x} : x_1 - x_2 = 0\}.$$

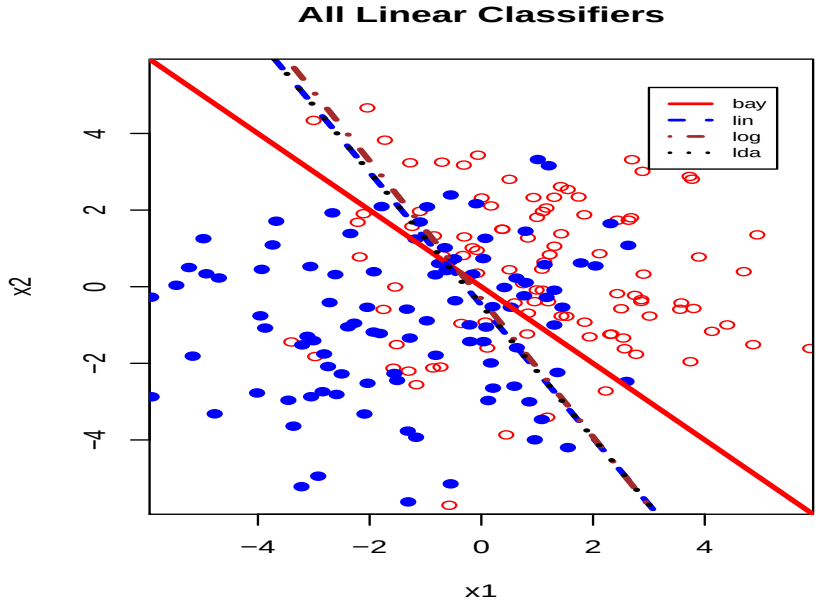
We generate

- the "training set" of $n = 200, 400$ to fit the classifier
- the "testing set" of $n' = 2000$ " to evaluate its prediction performance.


Training Scatter Plot
x2
x1
−4
−2
0
2
4


Bayes and Linear Classifiers
bay
lin
x2
x1
−4
−2
0
2
4


Logistic and LDA Classifiers
log
lda
x2
x1
−4
−2
0
2
4


All Linear Classifiers
bay
lin
log
lda
x2
x1
−4
−2
0
2
4

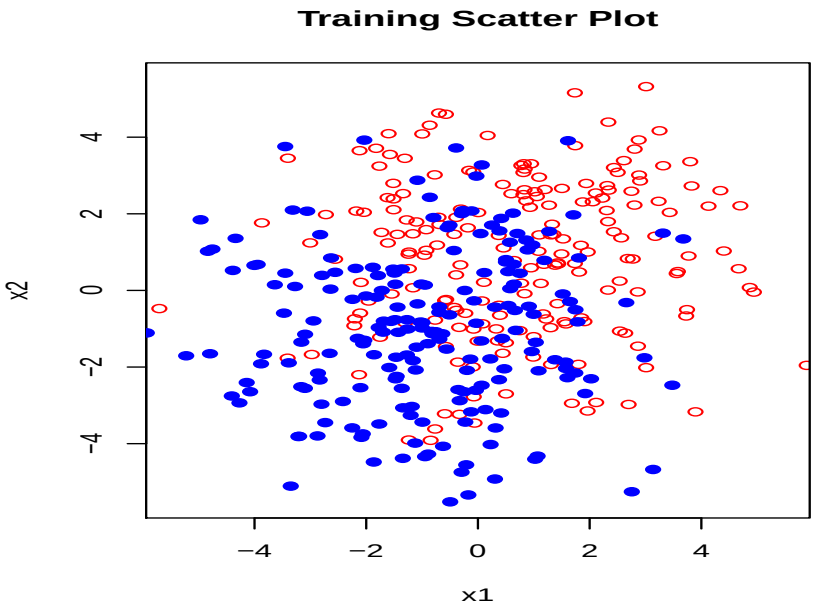
Performance of Various Classifiers (n=200)

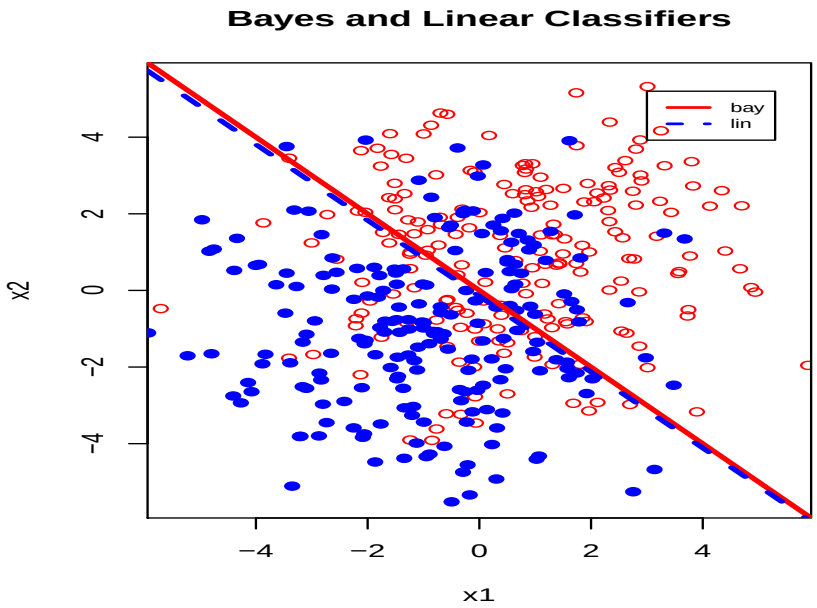
	Bayes	Linear	Logistic	LDA
Train Error	0.250	0.255	0.255	0.255
Test Error	0.240	0.247	0.242	0.247

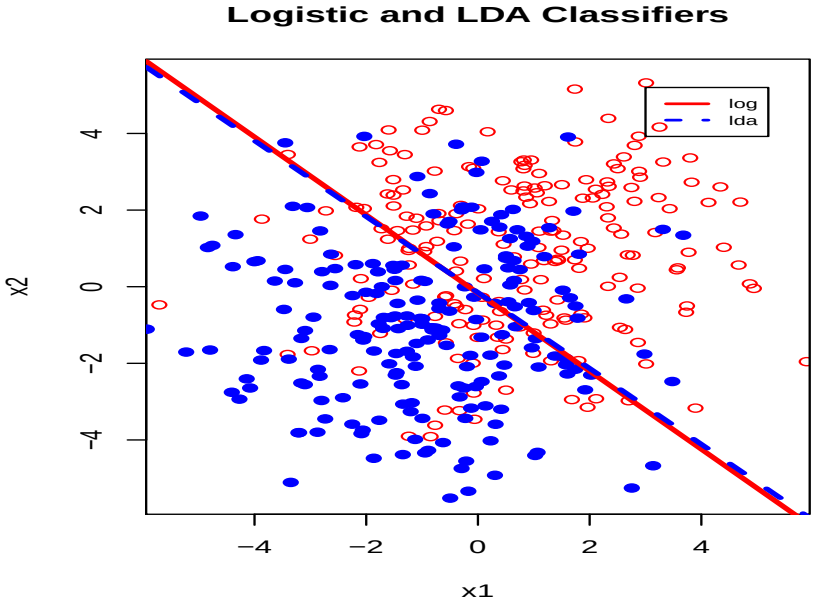
Fitted classification boundaries:

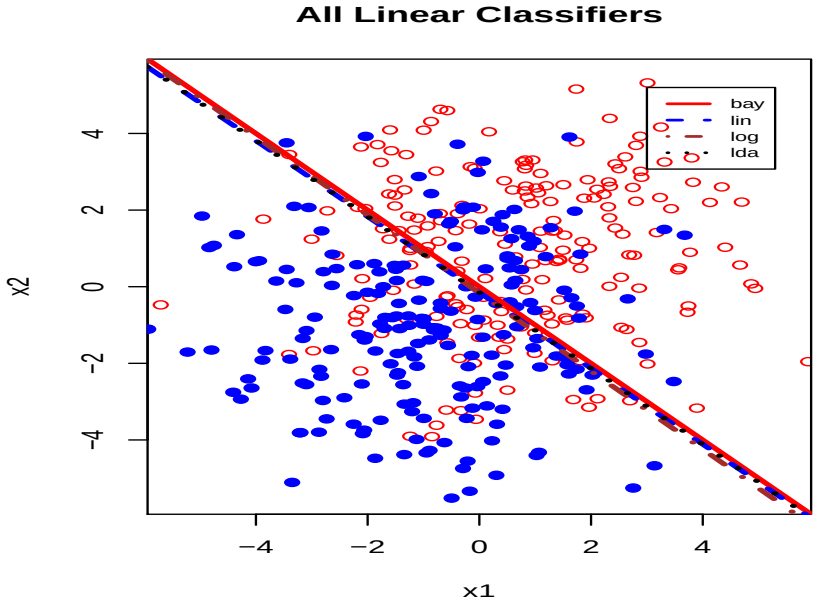
Linear & LDA: $x_2 = -0.48 - 1.73x_1$

Logistic: $x_2 = -0.32 - 1.80x_1$.


Training Scatter Plot
x2
x1
−4
−2
0
2
4


Bayes and Linear Classifiers
bay
lin
x2
x1
−4
−2
0
2
4


Logistic and LDA Classifiers
log
lda
x2
x1
−4
−2
0
2
4


All Linear Classifiers
bay
lin
log
lda
x2
x1
−4
−2
0
2
4

Fisher's Reduced Rank LDA (I)

- Don't need Gaussian distributions for $\mathbf{X}$.
- Find the linear combination $Z = a^T \mathbf{X}$ such that the between-class variance is maximized relative to the within-class variance, i.e. maximizing the Rayleigh quotient

$$\max_a \frac{a^T B a}{a^T W a},$$

  which is equivalent to $\max_a a^T B a$ subject to $a^T W a = 1$.
- Here, B is the between-class covariance of $\mathbf{X}$ (i.e. the covariance of class means μ_k's), while W is the within-class variance of $\mathbf{X}$ (i.e. the common covariance matrix Σ).
- Define $a = \Sigma^{-1/2} b$. Then, it is equivalent to solve $\max_b b^T \Sigma^{-1/2} B \Sigma^{-1/2} b$ subject to $b^T b = 1$. This is a standard eigenvalue problem.

Fisher's Reduced Rank LDA (II)

- b corresponds to the eigenvector of the largest eigenvalue of $\Sigma^{-1/2}B\Sigma^{-1/2}$.
- Note that $\Sigma^{-1/2}B\Sigma^{-1/2}$ and $\Sigma^{-1}B$ have the same eigenvalues.
- For binary classification, $B \propto (\mu_1 - \mu_0)(\mu_1 - \mu_0)^T$.
- Matrix $\Sigma^{-1/2}(\mu_1 - \mu_0)(\mu_1 - \mu_0)^T\Sigma^{-1/2}$ only has one non-zero (positive) eigenvalue and its corresponding eigenvector is given by

$$b \propto \Sigma^{-1/2}(\mu_1 - \mu_0).$$

- Therefore, $a = \Sigma^{-1/2}b \propto \Sigma^{-1}(\mu_1 - \mu_0) = \beta_1$ in LDA based on Gaussian distributions.
- Consider the plane $\mathbf{x}^T\beta_1 + c = 0$. Find c such that μ_1 and μ_0 have the same distance to the plane. Then,

$$c = -\frac{1}{2}(\mu_1 + \mu_0)^T\Sigma^{-1}(\mu_1 - \mu_0) = \beta_0,$$

when $\pi_1 = \pi_0$.