Machine Learning I

Lecture 5: Linear Methods in Classification

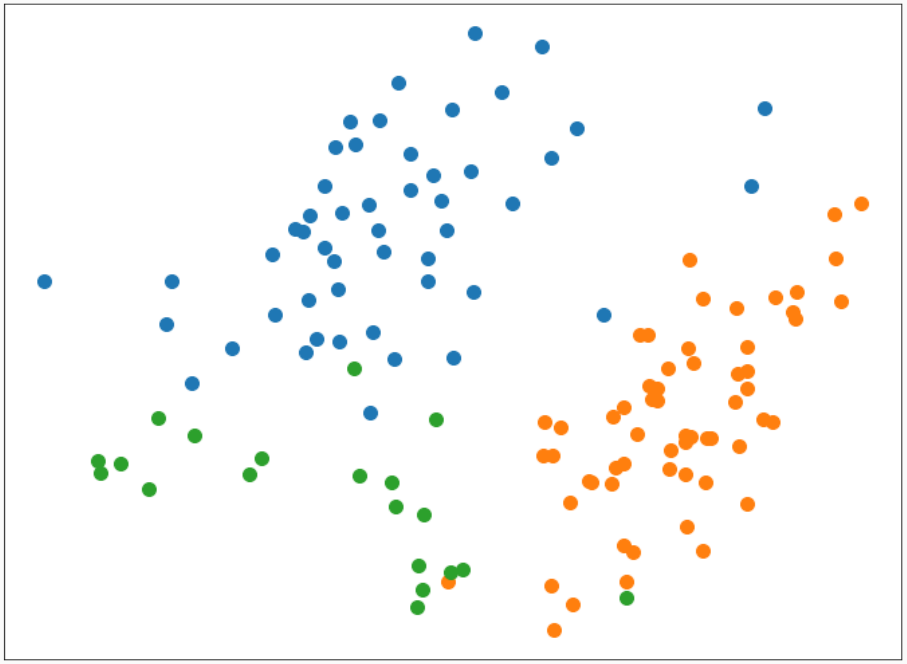
Nathaniel Bade

Northeastern University Department of Mathematics

Table of contents

Multilabel Classification

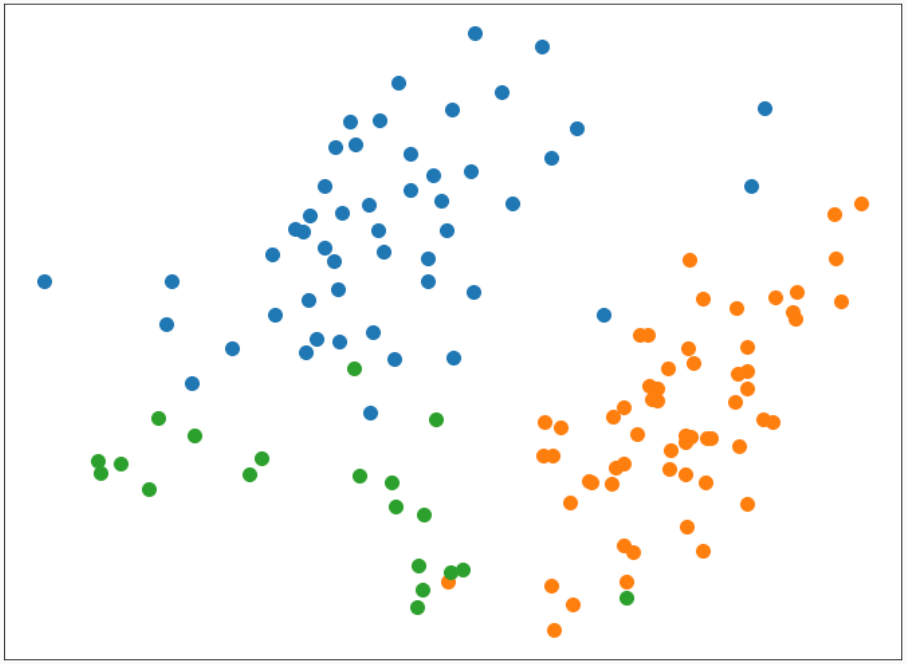
Multilable Classification: Notation



We will now expand our toolbox to include methods of classification for multiple labels. Suppose there are $|\mathcal{G}| = K$ classes, labeled $1, 2, \ldots, K$. Let Y_k be the **indicator** function for k, that is $Y_k = 1$ if $G = k$, but is 0 otherwise.

These variables are collected together into a vector $Y = (Y_1, \ldots, Y_K) \in \mathbb{R}^K$. The N training instances then form an $N \times K$ matrix $\mathbf{Y}$.

Multilable Classification: Notation

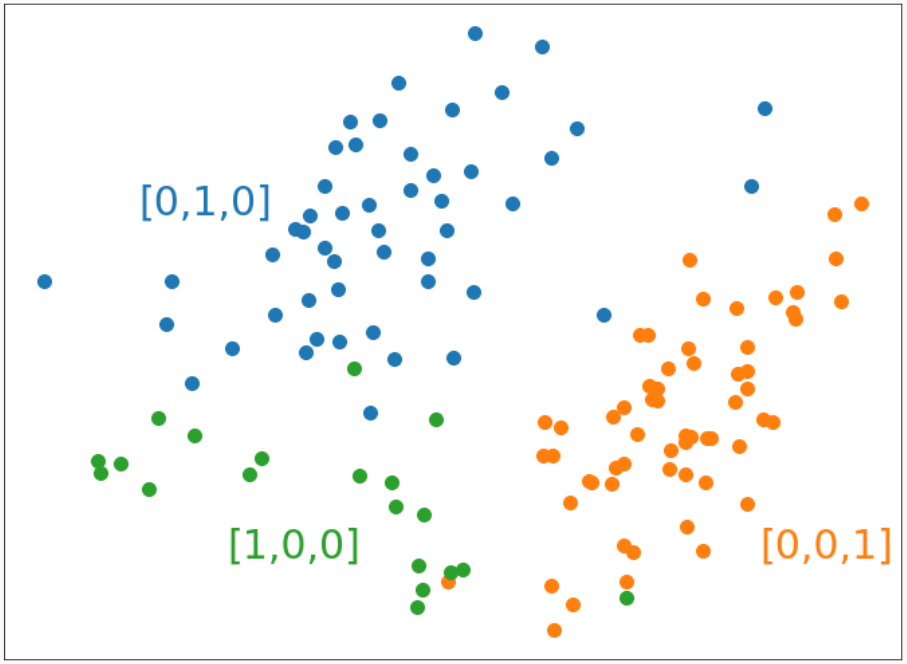


A model $Y = \hat{f}(X)$ will give a probability for each Y_k for each point X in the domain. Letting

$$\hat{f}(X) = (\hat{f}_1(X), \ldots, \hat{f}_K(X)) \quad \text{given} \quad \hat{f}_1(X) + \ldots + \hat{f}_K(X) = 1\,,\ \hat{f}_k(X) \geq 0\,,$$

we have $\hat{f}(X) \in \mathbb{R}^K$, with $\hat{f}_k(X) = \hat{Y}_k = \mathbb{P}(Y = k|X = X)$ be the probability that Y takes the label k given X.

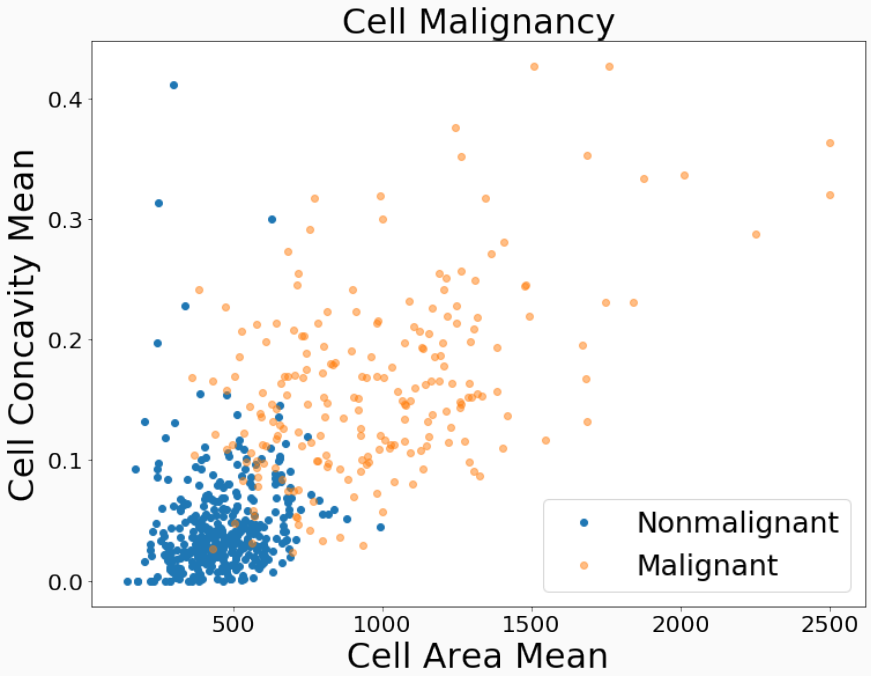
Multilabel Classification



For each datapoint (x_i, y_i), this means that $y_i = k$ is now encoded in a K vector, where the k'th entry is 1 and the other entries are 0. If we encode this bitwise it is known as a **one-hot encoding**. If probability functions $\hat{f}_k$ have been fit, a label $\hat{y}_i$ can be predicted by taking

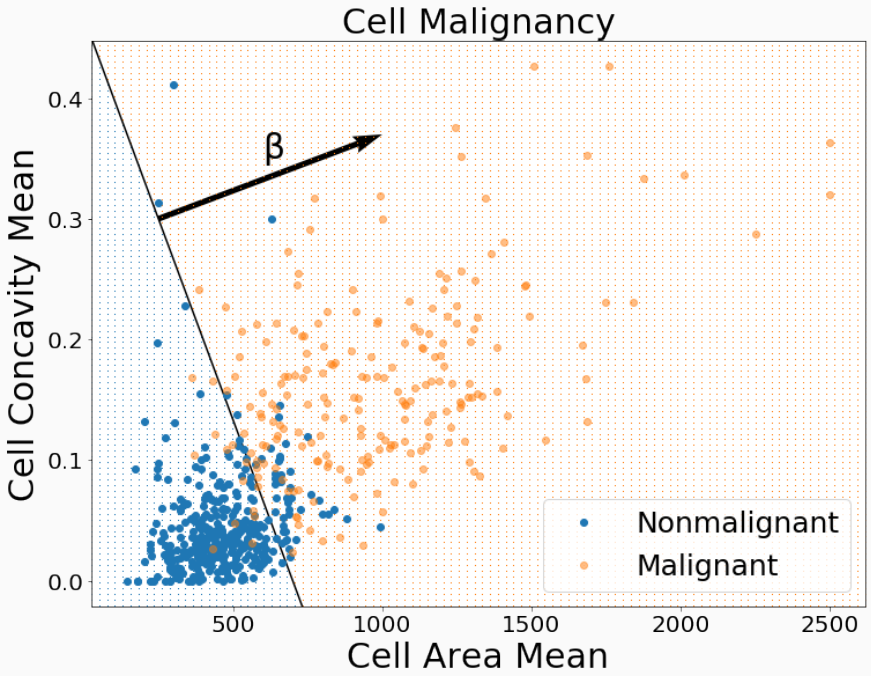
$$\hat{G}(x_i) = \underset{k \in \mathcal{G}}{\text{argmax}}\ \hat{f}_k(x_i)$$

Multilabel Classification



As a first example, in the plot above we are trying to separate cells into malignant and benign by their mean area and concavity. We have two labels, so $Y \in \mathbb{R}^2$, with Y_1 the probability that a cell with features X is malignant and Y_2 the probability it is benign.

Multilabel Classification

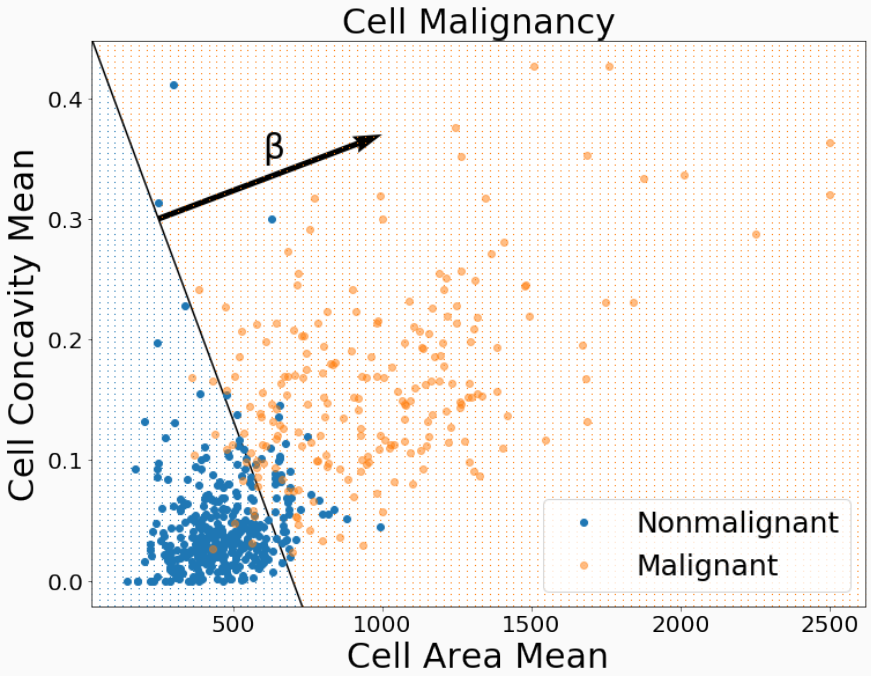


Fitting an affine linear function to Y takes the same form as before:

$$Y = [Y_1, \ldots, Y_k] = X^T \beta \,,$$

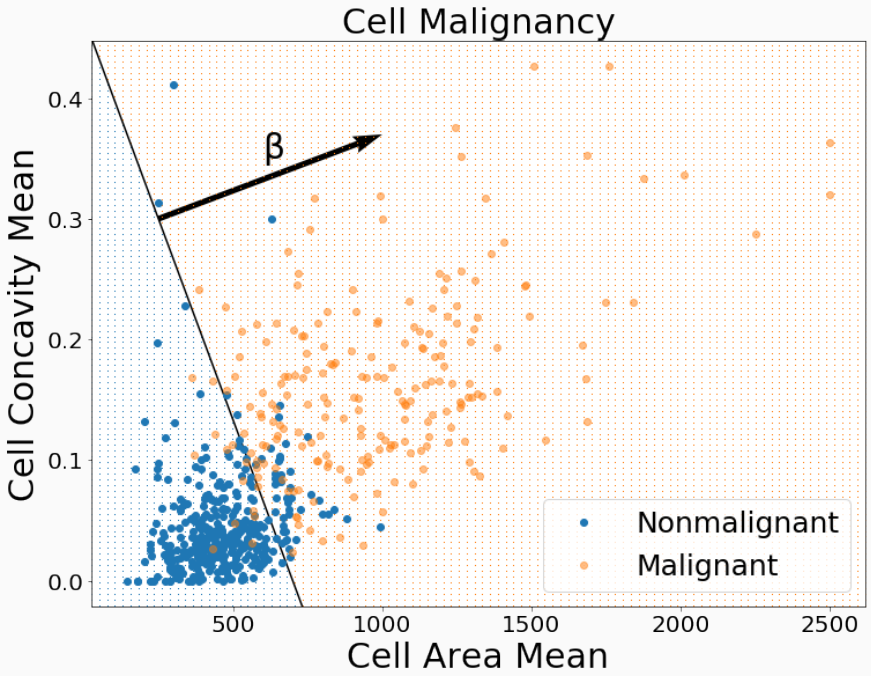
only now $\beta \in \mathbf{p} + \mathbf{1} \times \mathbf{K}$ is a $p+1 \times K$ matrix. Minimize RSS for each of the columns of $\mathbf{Y}$ yields $\beta = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}$, provided we bound $Y_k \in [0, 1]$.

Multilabel Classification



In the cancer classification above, we see a hard line called the **decision boundary**. This is the line where $\hat{Y}_1 = \hat{Y}_2 = .5$, and it separates the portion of the domain for X most probably takes on label k and label j.

Multilabel Classification



Note: The most natural loss function for a categorical variable is simple the 0-1 loss function, that is 1 if $y_i = \text{argmax}_k(\hat{f}_k(x_i))$ and 0 otherwise. However, 0-1 loss is discrete, and as a result is very difficult to optimize using smooth methods. Working instead with the probabilities makes optimization more straightforward.

Regression on Categorical Variables

Multilable Regression

Given a classification problem with K labels, we encode the repose categories $\mathcal{G}$ into a K vector $Y = (Y_1, \ldots, Y_K)$, $Y_i \in \{0, 1\}$. The N training instances form a $N \times K$ matrix $\mathbf{Y}$ of 1's and 0's. Letting X denote the $p + 1$ vector with $X_0 = 1$ as usual, the linear model

$$Y = X^T \beta \,,$$

now has a $p + 1 \times K$ matrix of coefficients β. The loss for each column is a K vector

$$RSS(\beta) = \sum_{i=1}^{N} (y_i - \hat{f}(x_i))^T (y_i - \hat{f}(x_i)) = (\mathbf{Y} - \mathbf{X}\beta)^T (\mathbf{Y} - \mathbf{X}\beta) \,,$$

minimizing all columns simultaneously gives $\hat{\beta} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{Y}$.

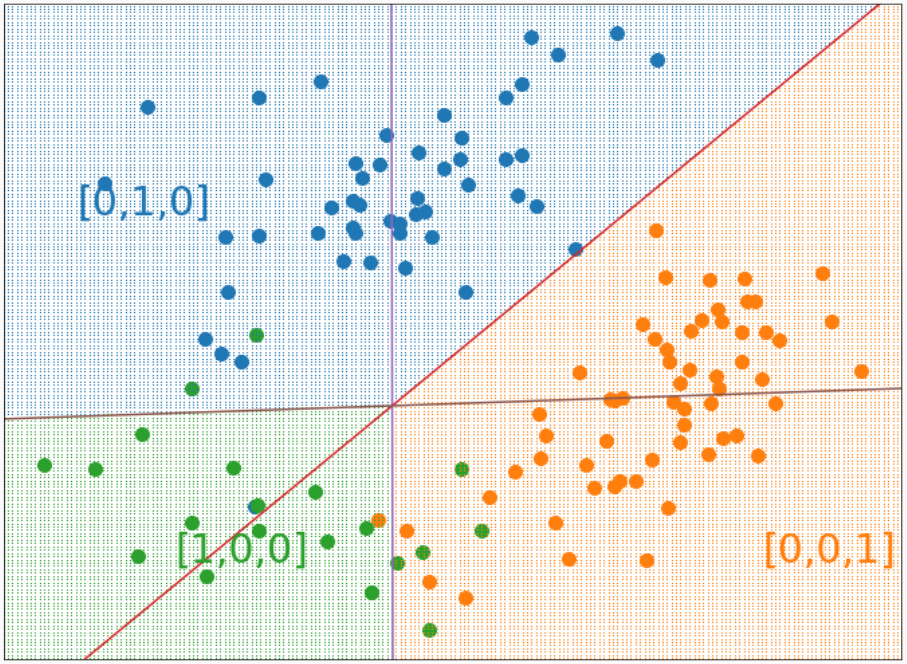
Multilable Regression

With a little bit of work (**exercise**, use centered vectors) one can show that

$$\sum_{k=1}^{K} \hat{f}_k(X) = 1 \quad \forall X\,,$$

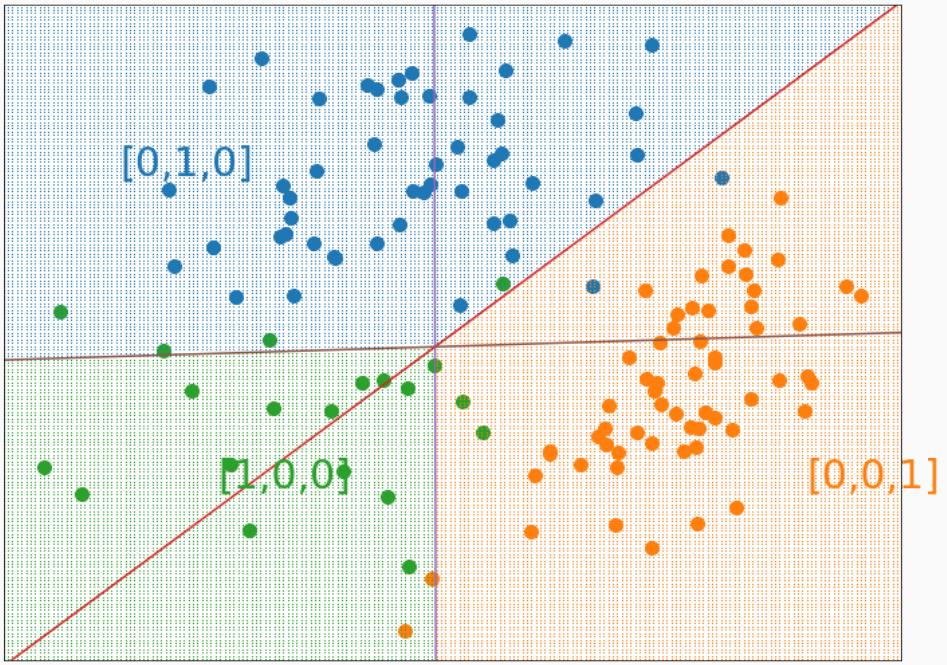
but there is no guarantee that the $\hat{f}_k$ are positive. This isn't necessarily fatal, in fact this kind of linear regression often works well, but we shouldn't understand the $\hat{f}_k$ as providing strict probabilities in any sense.

Multilabel Classification: Example



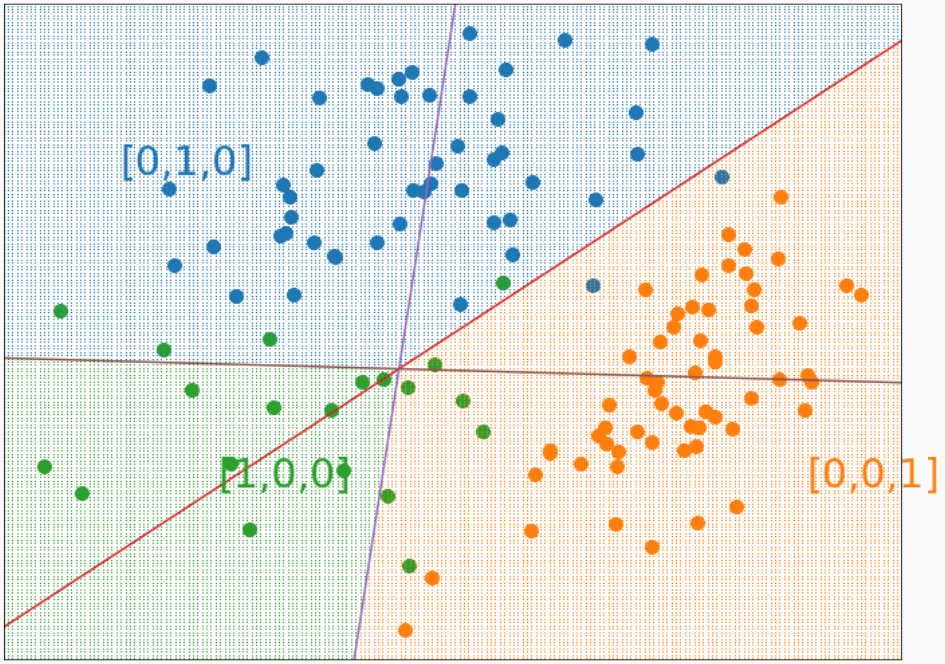
Also, while a linear decision boundary may not provide the best fit they have low variance relative to other classes. The regions above are defined by the regions in which $\hat{f}_k(x)$ is the largest for each k. The lines are the intersection lines $\hat{f}_k = \hat{f}_j$.

Multilabel Classification: Example



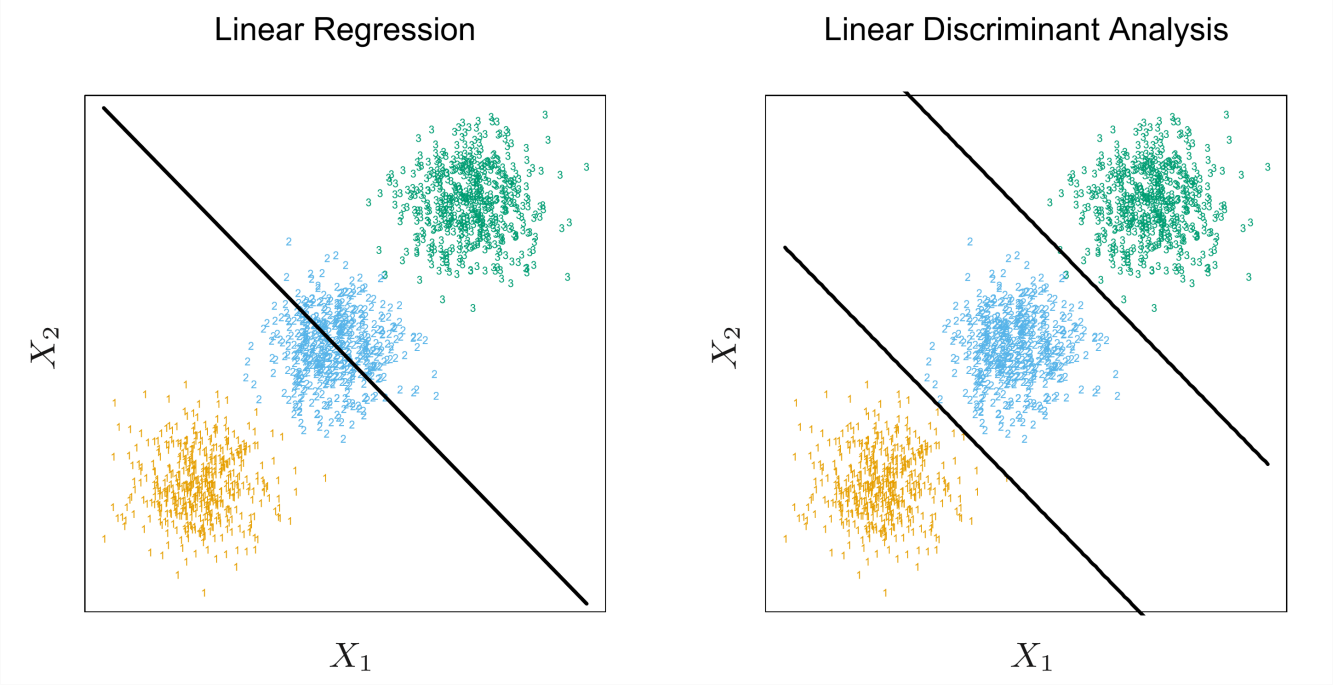
Also, while a linear decision boundary may not provide the best fit they have low variance relative to other classes. The regions above are defined by the regions in which $\hat{f}_k(x)$ is the largest for each k. The lines are the intersection lines $\hat{f}_k = \hat{f}_j$.

Multilabel Classification: Example



Also, while a linear decision boundary may not provide the best fit they have low variance relative to other classes. The regions above are defined by the regions in which $\hat{f}_k(x)$ is the largest for each k. The lines are the intersection lines $\hat{f}_k = \hat{f}_j$.

Multilabel Classification: Masking



The biggest problem with linear decision boundaries comes from masking. In the admittedly extreme example on the left, the three clusters are clearly distinct but the linear regressor misses the middle class completely. On the right we see the classes being successfully fit with a quadratic regression.

Multilabel Classification: Quadratic Fitting

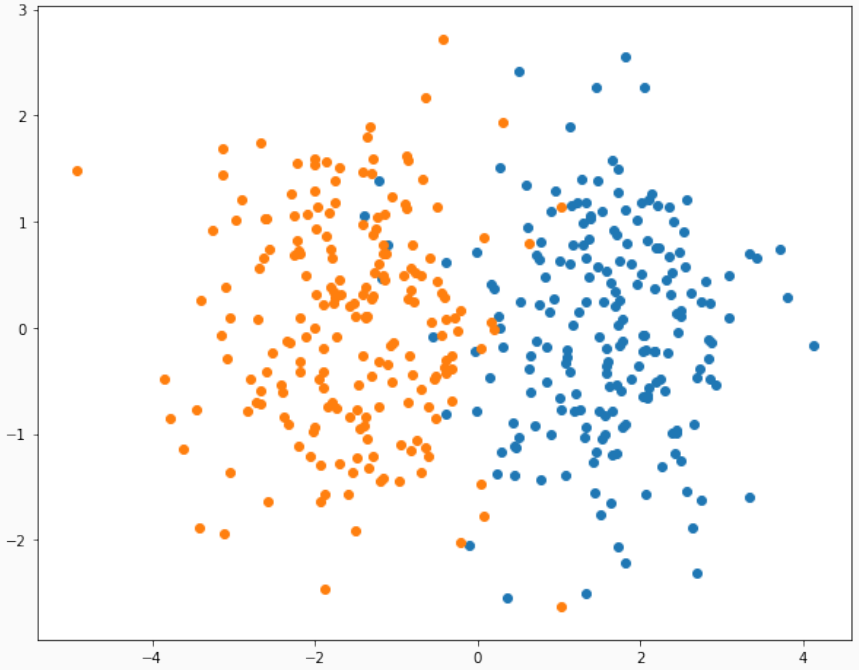
A quick note about quadratic regression. While we will say more about nonlinear fitting later, quadratic regression can be though of a linear regression on the constructed features $X_i X_j$, for all i, j. If we're willing to add $p(p+1)/2$ new variables the quadratic regression

$$Y = \beta_0 + \sum_i X_i \beta_i + \sum_{i<j} X_i X_j \alpha_{ij}$$

can be done with a linear fit.

Linear Discriminant Analysis

Class Conditional Density

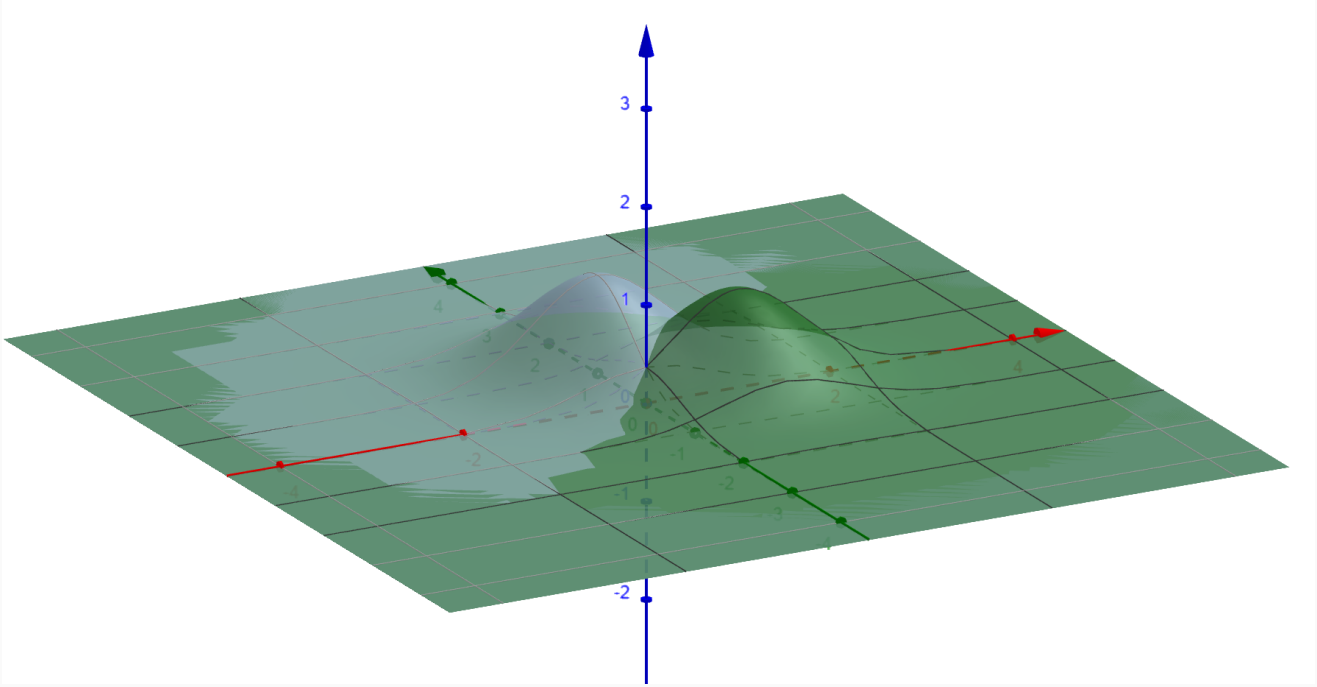


Consider a mixture model like the one show above, where each of the labels is distributed as a multivariate normal distribution. The **class conditional density**

$$f_k(x) = \mathbb{P}(X = x | G = k)\,,$$

is the multivariate normal distribution for the k'th label.

Class Conditional Density



Consider a mixture model like the one show above, where each of the labels is distributed as a multivariate normal distribution. The **class conditional density**

$$f_k(x) = \mathbb{P}(X = x | G = k)\,,$$

is the multivariate normal distribution for the k'th label. Above are the normal f_k viewed as functions.

Bayes Theorem and Class Densities

We are interested in estimating the conditional probabilities $P(G = k|X = x)$. Out first step of course should be to total the number of labels for each class in our training set, giving us a (**prior**) probability π_k for each label. Since the class conditional density

$$f_k(x) = \mathbb{P}(X = x|G = k)\,,$$

is the distribution of x's for the label k, Bayes theorem gives the probability for a label k at a point x:

$$\mathbb{P}(G = k|X = x) = \frac{\mathbb{P}(G = k)\mathbb{P}(X = x|G = k)}{\mathbb{P}(X = x)} = \frac{\pi_k\, f_k(x)}{\sum_{j=1}^{K} \pi_j}\, f_j(x)\,.$$

So knowing $f_k(x)$ and estimating the label proportions is almost good enough to know $\mathbb{P}(G|X)$.

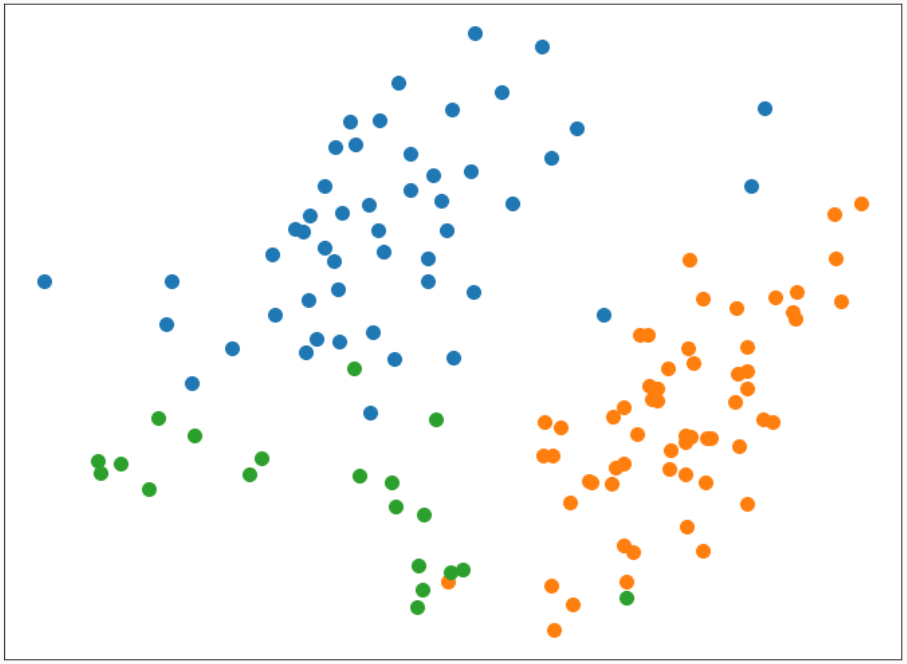
Bayes Theorem and Class Densities

Many techniques are based on models for the class densities:

- If we assume the class density function are Gaussian, this leads to linear and quadratic class boundaries. This process is known as Linear or Quadratic discriminant analysis in reference to the class boundary shape.
- We will also see that we can generate linear class boundaries without the Gaussian assumption by fitting linear functions to the probabilities directly, while forcing the result to be a real probability vector. This is known as logistic regression.
- Finally, we can generate class densities with nonlinear boundaries by using non-parametric density estimates (ie sampled or bootstrapped densities).

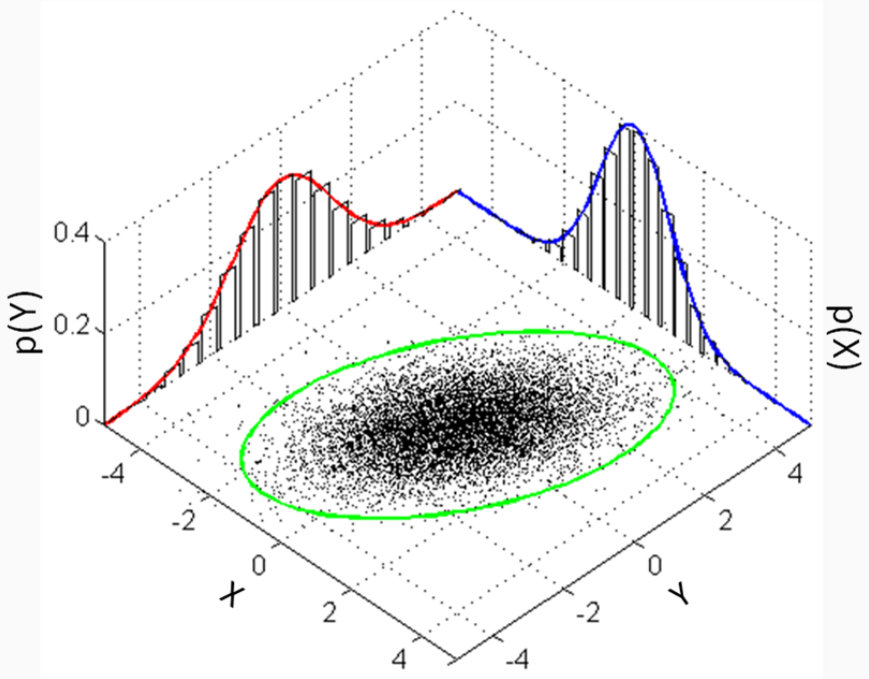
We will show the first point now, that if the class conditional densities are Gaussian, the decision boundaries must be linear or quadratic.

Gaussian Densities



Given a multilabel classification problem, we may try to model each of the labels directly by guess their underlying probability distribution and using maximum likelihood, or other methods.

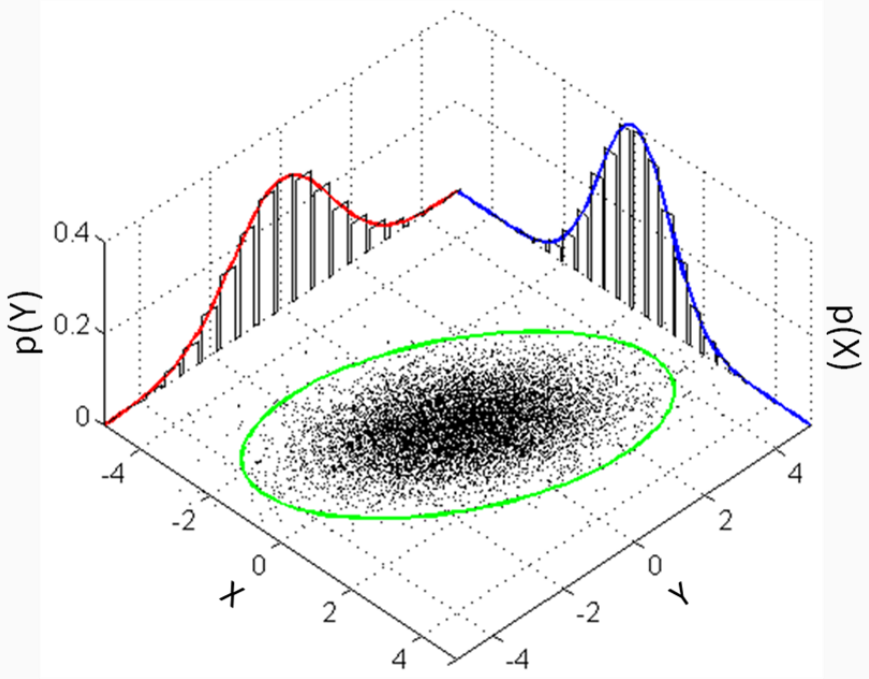
Gaussian Densities



Suppose we model each density $f_k(x)$ as a multivariate Gaussian with covariance matrix $\mathbf{\Sigma}_k$

$$f_k(x) = \left[(2\pi)^p |\mathbf{\Sigma}_k|\right]^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(x-\mu_k)^T \mathbf{\Sigma}_k^{-1}(x-\mu_k)\right).$$

Gaussian Densities

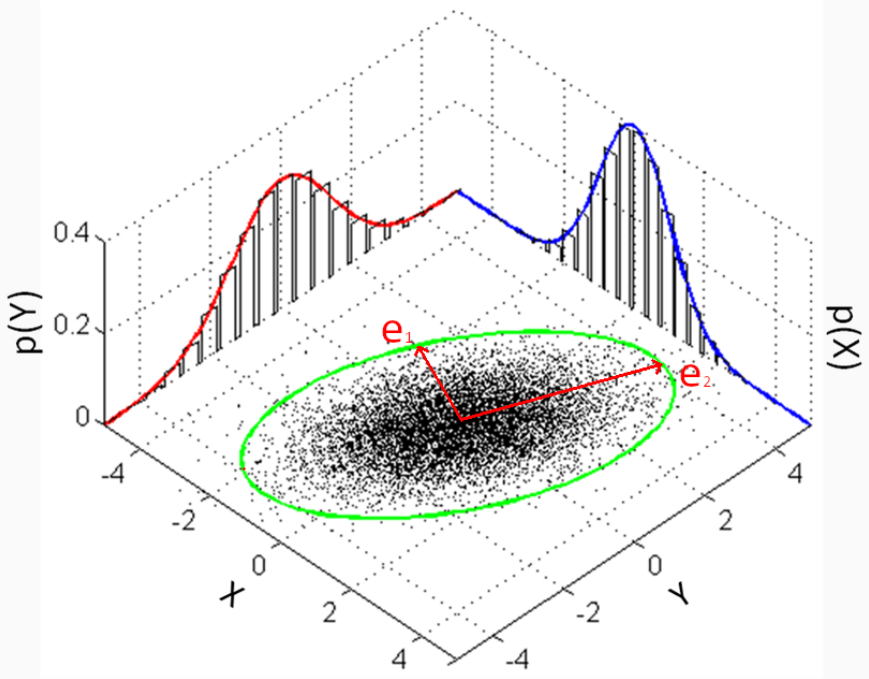


$$f_k(x) = \left[(2\pi)^p |\mathbf{\Sigma}_k|\right]^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(x-\mu_k)^T \mathbf{\Sigma}_k^{-1}(x-\mu_k)\right)$$

Above are the 3σ ellipse and marginal distributions for

$$\mu_k = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \mathbf{\Sigma} = \begin{pmatrix} 1 & 3/5 \\ 3/5 & 2 \end{pmatrix}.$$

Gaussian Densities



The eigenvectors e_i of $\mathbf{\Sigma}$ define the axes off the ellipsoid, the axis length is $\sqrt{\lambda_i}$. This is a direct result of the eigen-decomposition

$$\mathbf{\Sigma} = UDU^T$$

which always exists because $\mathbf{\Sigma}$ is real and symmetric.

Gaussian Densities

Suppose we model each density $f_k(x)$ as a multivariate Gaussian with covariance matrix $\mathbf{\Sigma}_k$

$$f_k(x) = \left[(2\pi)^p |\mathbf{\Sigma}_k|\right]^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(x-\mu_k)^T \mathbf{\Sigma}_k^{-1}(x-\mu_k)\right).$$

We would like to characterize when $\mathbb{P}(G = k|X = x)$ is greater than $\mathbb{P}(G = j|X = x)$, for each k and j. Given the exponential, it's natural to compare the ratio of the logs. Using Bayes Theomre, k is more likely when

$$\log \frac{\mathbb{P}(G = k|X = x)}{\mathbb{P}(G = j|X = x)} = \log \frac{\pi_k f_k}{\pi_j f_j} > 0\,.$$

Linear Discriminant Functions

If the covaraince matrices are the same for each k, that is $\mathbf{\Sigma}_k = \mathbf{\Sigma}$ for all k, then

$$f_k(x) = \left[(2\pi)^p |\mathbf{\Sigma}|\right]^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(x-\mu_k)^T \mathbf{\Sigma}^{-1}(x-\mu_k)\right).$$

and both the constant and quadratic terms cancel

$$\begin{aligned}
\log \frac{f_k \pi_k}{f_j \pi_j} &= \log \frac{\pi_k}{\pi_j} - \frac{1}{2}(x-\mu_k)^T \mathbf{\Sigma}^{-1}(x-\mu_k) + \frac{1}{2}(x-\mu_j)^T \mathbf{\Sigma}^{-1}(x-\mu_j) \\
&= \log \frac{\pi_k}{\pi_j} - \frac{1}{2}(\mu_k+\mu_j)^T \mathbf{\Sigma}^{-1}(\mu_k-\mu_j) + x^T \mathbf{\Sigma}^{-1}(\mu_k-\mu_j) \\
&\geq 0\,.
\end{aligned}$$

This is a linear expression in x and so leads to linear decision boundaries.

Linear Discriminant Functions

The equation

$$0 \geq \log \frac{\pi_k}{\pi_j} - \frac{1}{2}(\mu_k + \mu_j)^T \mathbf{\Sigma}^{-1}(\mu_k - \mu_j) + x^T \mathbf{\Sigma}^{-1}(\mu_k - \mu_j)$$

can be rewritten as

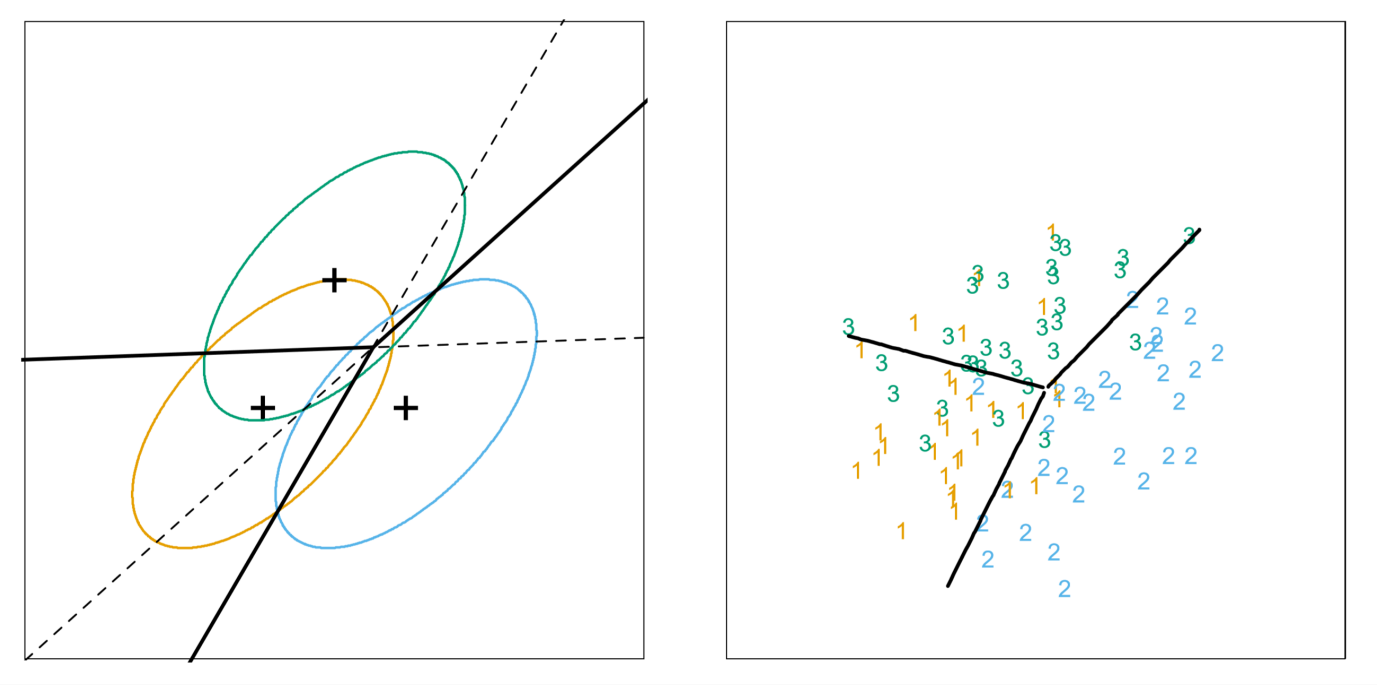
$$\delta_k(x) \geq \delta_k(x)\,,$$

where

$$\delta_k(x) = \log(\pi_k) - \frac{1}{2}\mu_k^T \mathbf{\Sigma}^{-1}\mu_k + x^T \mathbf{\Sigma}^{-1}\mu_k\,.$$

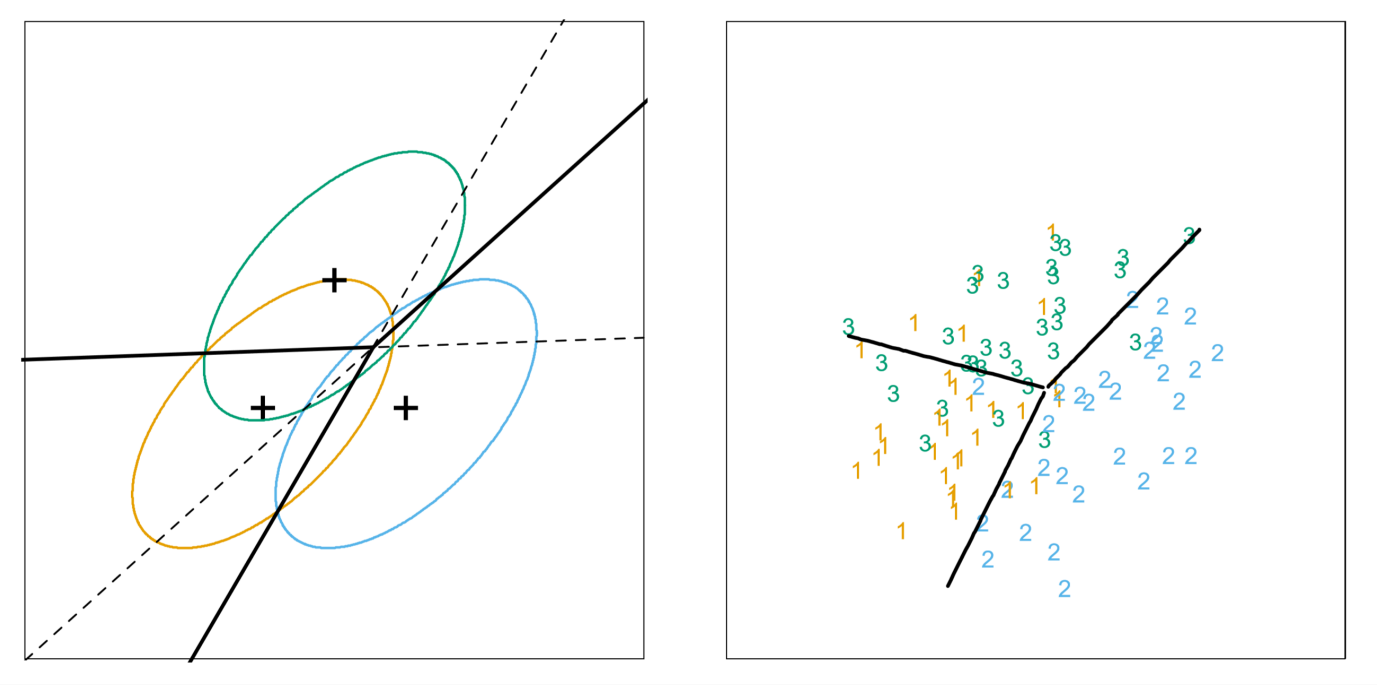
The label on x is given by $\text{argmax}_k \delta_k(x)$. The linear decision boundaries are $\delta_k(x) = \delta_j(x)$.

Linear Discriminant Functions



The label on x is given by $\text{argmax}_k \delta_k(x)$. The linear decision boundaries are $\delta_k(x) = \delta_j(x)$. Above, we see three Gaussian distributions 95% density contours, as well as the decision boundaries.

Linear Discriminant Functions



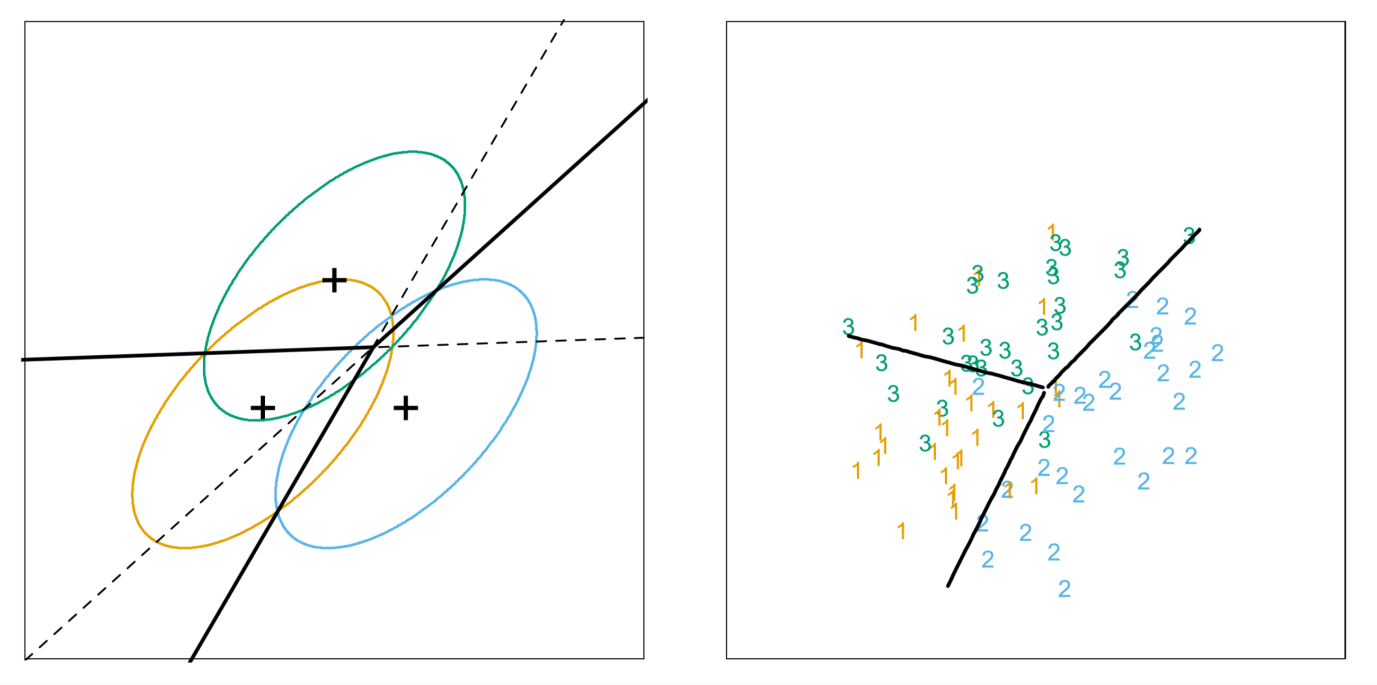
In practice, we need to estimate the parameters of the Gaussian distribution as follows: For each label k,

$\hat{\pi}_k = N_k/N$, where N_k is the number of observations of k.

$\hat{\mu}_k = \frac{1}{N_k} \sum_{y_i=k} x_i$ is the mean of k observations.

$\hat{\mathbf{\Sigma}}_k = \mathbf{Var}(x_i)$ for $y_i = k$, is the sample covariance matrix.

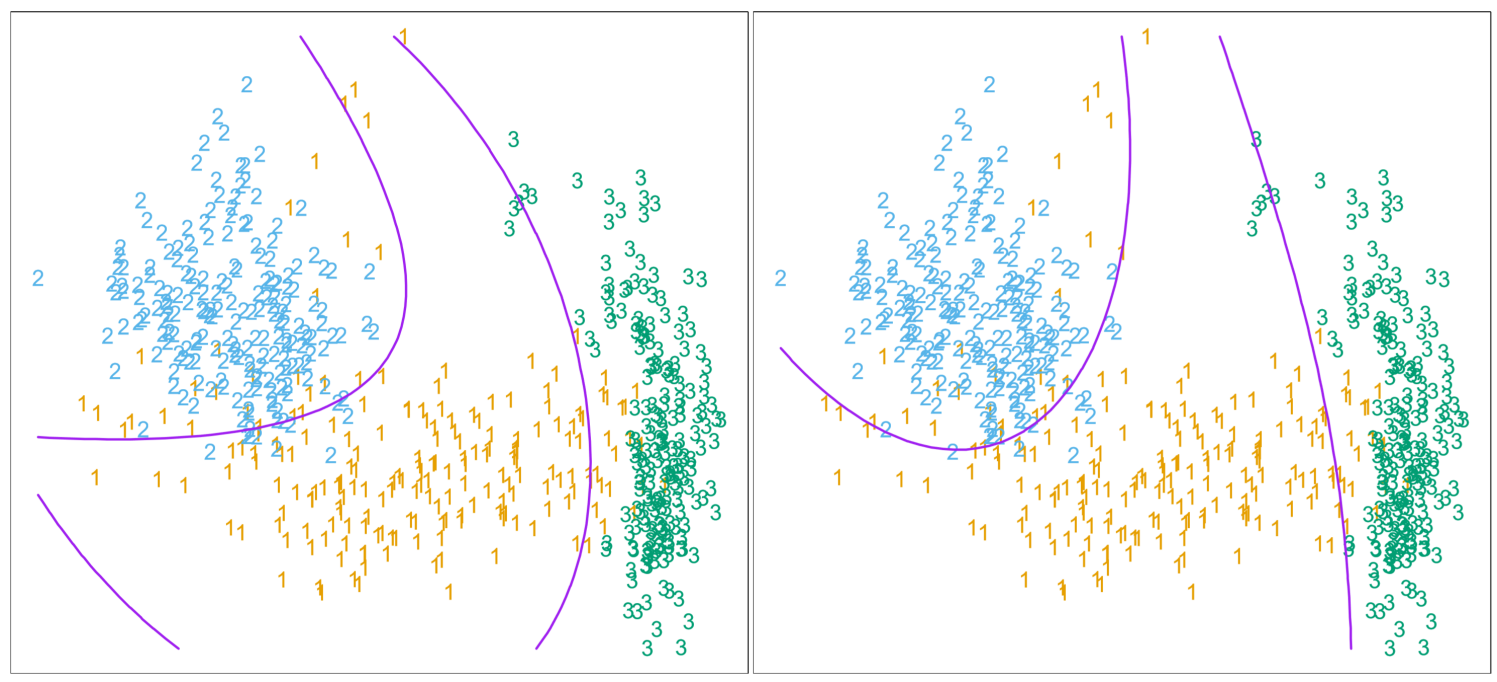
Linear Discriminant Functions



For binary classification it can be shown that the coefficient vector β from least squares is proportional to the coefficient vector of linear discriminant analysis (LDA). But the origin β_0 might be different.

For more labels, the correspondence between LDA and regression can be made by changing the loss function. This again shows the versatility of regression, since LDA can be shown to avoid masking problems.

Quadratic Discriminant Functions

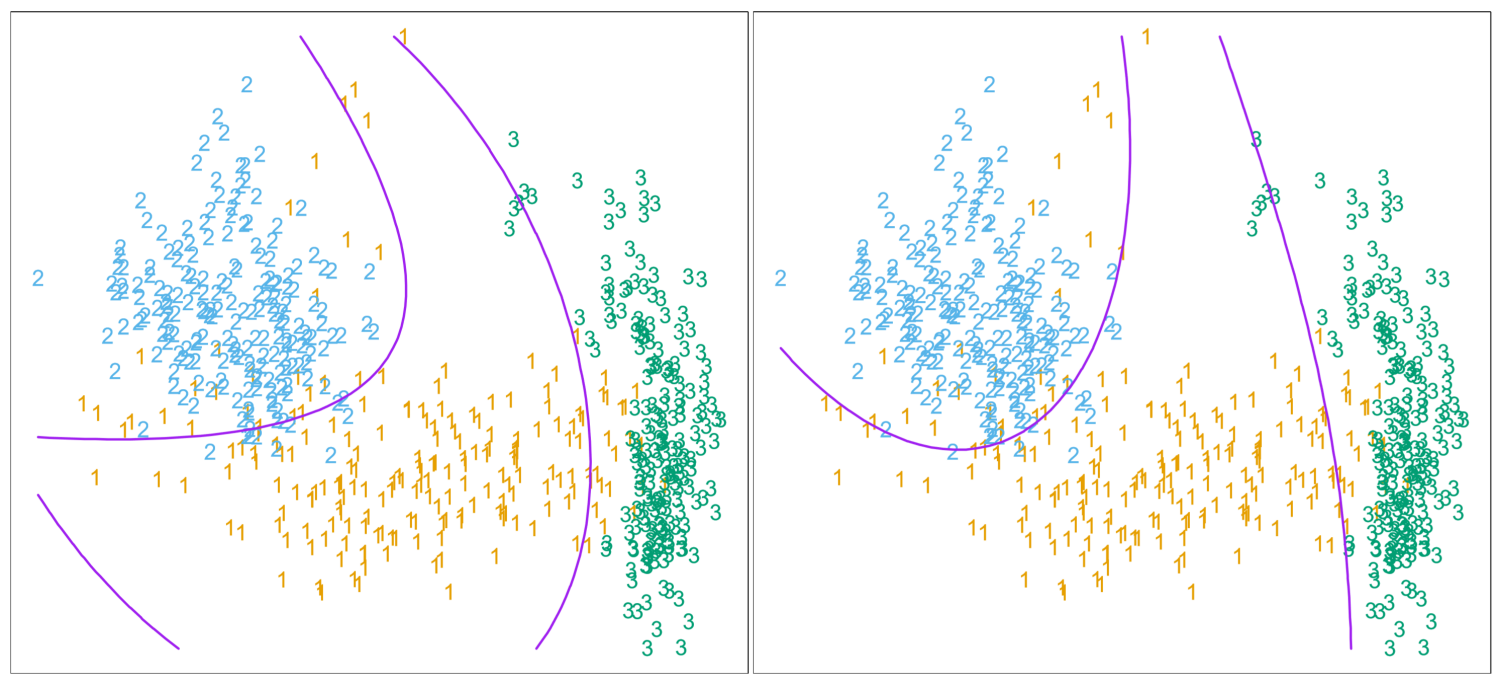


If we remove the requirement that the $\mathbf{\Sigma}_k$ are all equal, the discriminant functions become quadratic

$$\delta_k(x) = -\frac{1}{2}\log|\mathbf{\Sigma}_k| - \frac{1}{2}(x-\mu_k)^T\mathbf{\Sigma}_k^{-1}(x-\mu_k) + \log\pi_k\,.$$

This is known as **quadratic discriminant analysis** (QDA).

Quadratic Discriminant Functions



Here, the right plot uses LDA on X_1, X_2, X_1X_2, X_1^2 and X_2^2 while the right plot uses QDA by directly fitting the coefficients using gradient decent. Both techniques work very well on a wide variety of labeling tasks.

Here, we see that if we assume that the class conditional densities are Gaussian, the decision boundaries must be linear or quadratic.

Logistic Regression

Logistic Regression

Logistic regression rises from two places:

- The desire to define a meaningful, smooth probability density functions for discrete data.
- The desire to define a linear model for the conditional probabilities $\mathbb{P}(G = j|X = x)$, while holding constant the fact that they sum to 1.

Taking our hint from the linear discriminant, we fix the probabilities relative to $G = K$ as linear functions

$$\log \frac{\mathbb{P}(G = j|X = x)}{\mathbb{P}(G = K|X = x)} = \beta_{j,0} + x^T \beta_j \quad \forall j = 1, \ldots, K-1 \,.$$

Logistic Regression

We can solve

$$\log \frac{\mathbb{P}(G = j|X = x)}{\mathbb{P}(G = K|X = x)} = \beta_{j,0} + x^T \beta_j \quad \forall j = 1, \ldots, K-1 \,.$$

for

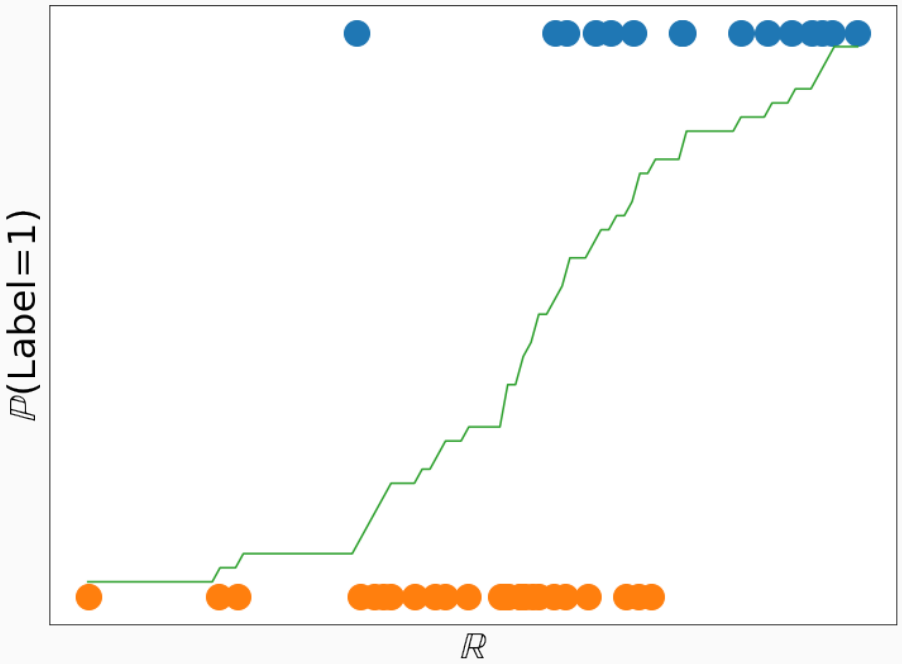
$$\mathbb{P}(G = j|X = x) = \frac{\exp(\beta_{j,0} + x^T \beta_j)}{1 + \sum_{\ell=1}^{K-1} \exp(\beta_{\ell,0} + x^T \beta_\ell)} \,, \quad \forall j = 1, \ldots, K-1 \,.$$

and

$$\mathbb{P}(G = K|X = x) = \frac{1}{1 + \sum_{\ell=1}^{K-1} \exp(\beta_{\ell,0} + x^T \beta_\ell)} \,.$$

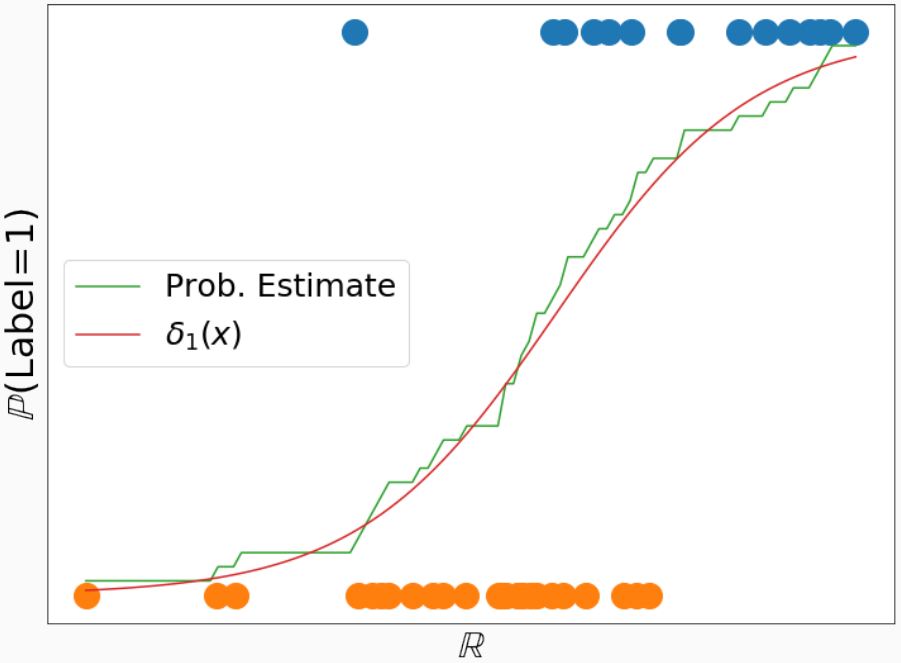
These clearly sum to 1.

One Variable Examples



In one variable, this gives a meaningful smoothing of the density of labels along a continuous axis.

One Variable Examples

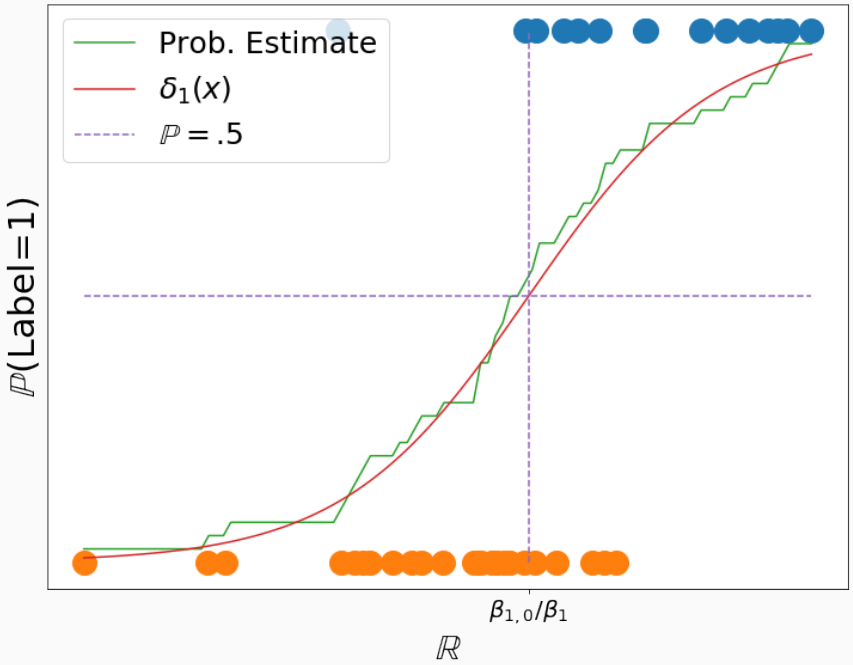


In one variable, this gives a meaningful smoothing of the density of labels along a continuous axis.

$$\delta_1 = \frac{\exp(\beta_{1,0} + x^T\beta_1)}{1 + \exp(\beta_{1,0} + x^T\beta_1)}.$$

In terms of these parameters, $\delta_1 = .5$ when $x = -\beta_{1,0}/\beta_1$

One Variable Examples

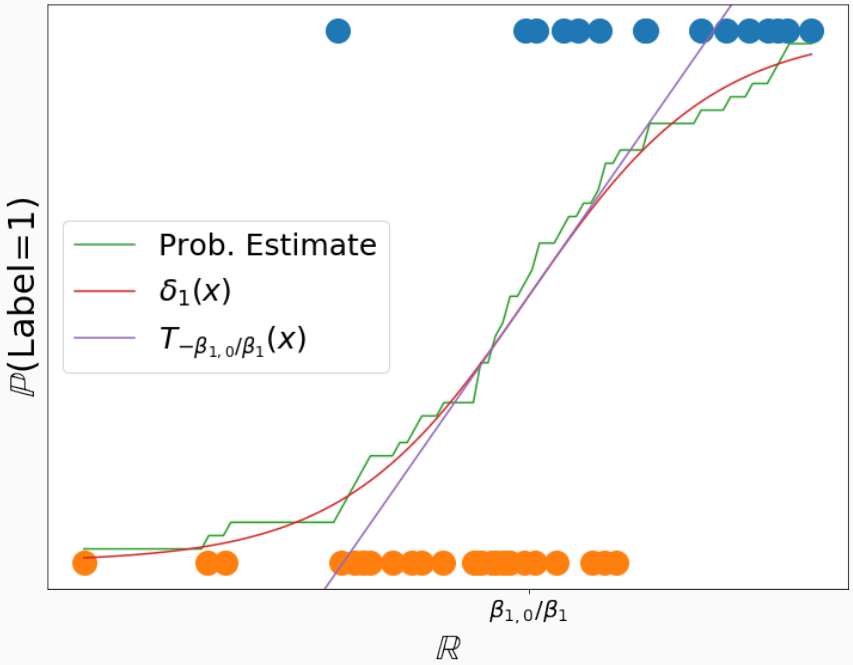


In one variable, this gives a meaningful smoothing of the density of labels along a continuous axis.

$$\delta_1 = \frac{\exp(\beta_{1,0} + x^T\beta_1)}{1 + \exp(\beta_{1,0} + x^T\beta_1)}.$$

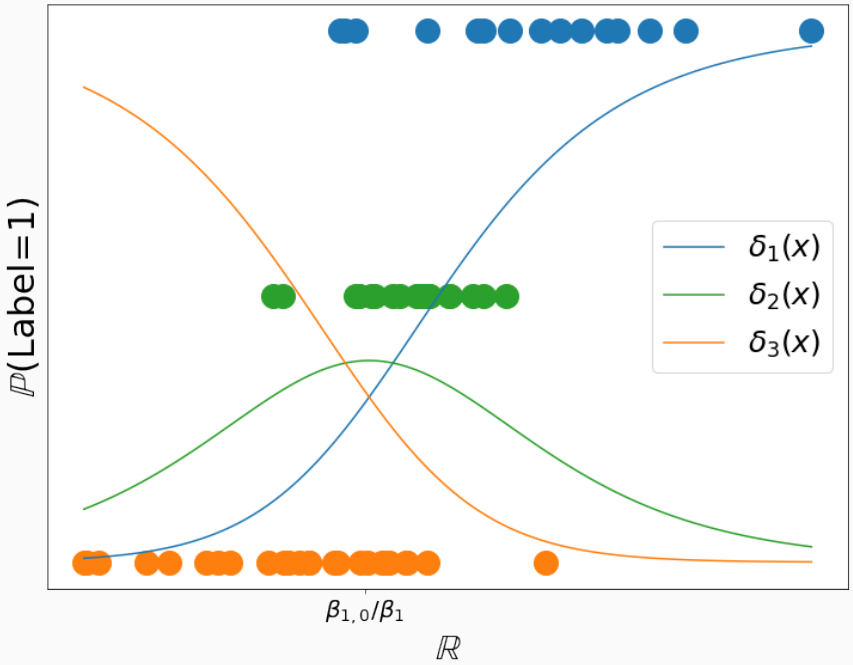
In terms of these parameters, $\delta_1 = .5$ when $x = -\beta_{1,0}/\beta_1$

One Variable Examples



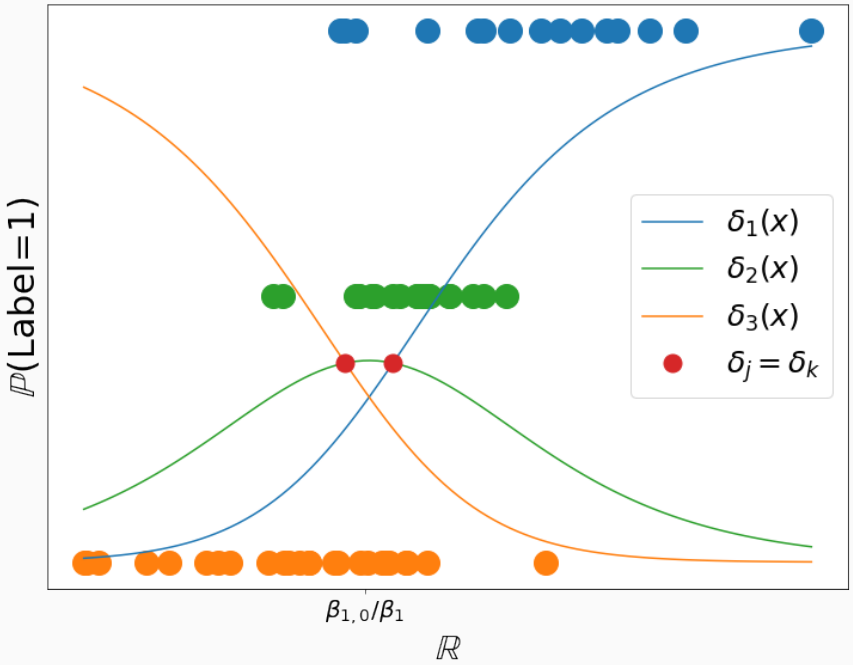
In terms of these parameters, $\delta_1 = .5$ when $x = -\beta_{1,0}/\beta_1$. Furthermore, this is an inflection point and the slope of $\delta_1(x)$ point is $\beta_1/4$.

One Variable Examples



For multiple labels, the dependence on the parameters is less explicit but can still be worked out. Notice that the $\delta_2(x)$ is defined explicitly in terms of δ_1 and δ_2.

One Variable Examples



For multiple labels, the dependence on the parameters is less explicit but can still be worked out. Notice that the $\delta_2(x)$ is defined explicitly in terms of δ_1 and δ_2. For more variables, the intersection points will become our linear decision boundaries $\delta_j(x) = \delta_k(x)$.

Fitting Logistic Regression

Logistic regression is usually fit using maximum likelihood. For N observations of data (x_i, y_i), the likelihood is

$$L(\beta) = \prod_{i=1}^{N} \mathbb{P}(G = y_i | X = x_i; \beta) \, .$$

To maximize $L(\beta)$ is it enough to maximize $\ell(\beta) = \log L(\beta)$.

Logistic Regression vs LDA

So what then is the difference between logistic regression and linear discriminant analysis? In both cases we arrived at the same formula for the conditional probabilities: Taking K as the reference variable,

$$\frac{\mathbb{P}(G = k|X = x)}{\mathbb{P}(G = K|X = x)} = \beta_{k0} + \beta_k^T x_k \, .$$

The difference lies in the fitting itself: In Logistic regression we make no assumptions about the distribution on X, directly fitting the conditional likelihood $\mathbf{P}(G = k|X)$.

For LDA, we assume that the X for each class are Gaussian. Mathematically, we are maximizing the full log likelihood of the joint distribution $\mathbf{P}(G = k, X)$.

Logistic Regression vs LDA

What role do these additional assumptions play?

Additional model assumptions of LDA restrict the model, raising the bias and lowering the variance compared to LR.

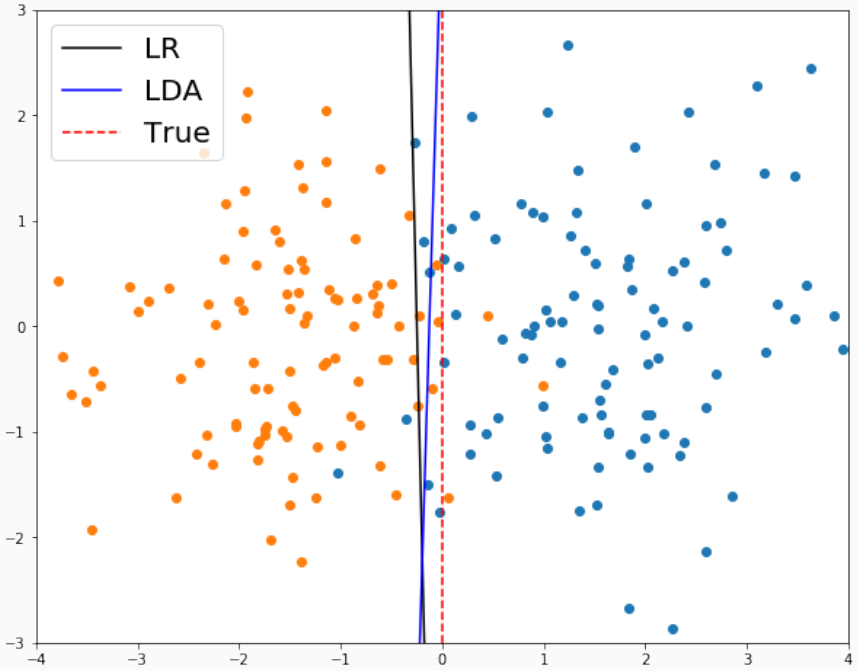
If the data within each label are normally distributed, LR can require up to 30% more data to be as efficient as LDA.

In LDA, unlabeled data can help us better compute the $\hat{f}_k$, improving our fit.

On the other hand, LR downplays outliers since all points are treated equally. LDA is more sensitive, and an outlier in X can pull morph the whole distribution.

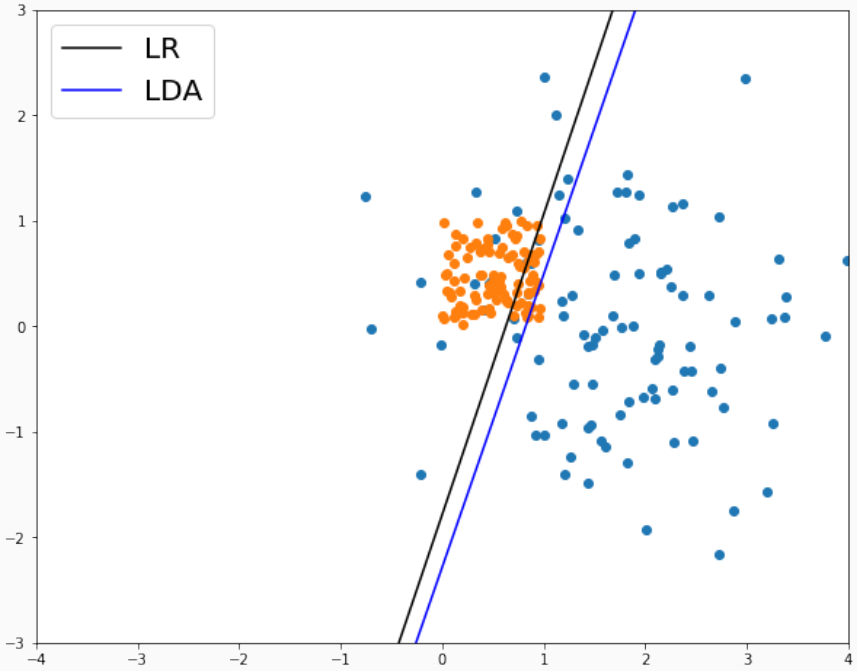
LR is agnostic about the distribution of the X's, and as such will better be able to fit non-normal data.

Logistic Regression vs LDA



However, these difference should always be understood as generalization. When the X distribution is Gaussian LDA tends to do better, but for enough data the fits are similar.

Logistic Regression vs LDA



On the other hand, when X distribution is not Gaussian LDA tends to make a worse fit, even in the face a a mountain of evidence.

Fitting Logistic Regression with Newtons Method

Newtons Method

Newtons Method is an iterative method of finding zeros of differential functions. In one variable, we try to find a zero of $f(x)$ by successive approximations of $f(x)$ by it's Taylor polynomial. Starting with x_0, we try to find an improved guess $x + \delta$ by Taylor expanding $f(x + \delta)$ around x:

$$f(x + \delta) \approx f(x) + \delta f'(x) = 0\,.$$

Solving for $\delta = -f(x)/f'(x)$, we have an "improved" guess $x_{new} = x - f(x)/f'(x)$ for the location of the zero. We then iterate until we have arrived at a zero. It can be proved that under reasonable assumptions on $f(x)$ we always will.

Multivariate Newtons Method

For higher dimensional functions, we proceed exactly as before: Let $f(x)$ be differentiable. Expanding $f(x+\delta)$ around x, we find

$$f(x+\delta) \approx f(x) + J(x)\delta = 0\,,$$

where $J_{ij} = \frac{\partial f_i}{\partial x_j}$ is the Jacobean matrix (gradient if $f(x) \in \mathbb{R}$). If $J(x)$ is invertible, we can solve for

$$x_{new} = x - J^{-1}(x)f(x)\,.$$

Optimization via Newtons Method

For a single valued multivariate function $f(x)$, we can use Newtons method to perform optimization. Since optimization is just finding $\nabla f(x) = 0$, we write

$$\nabla f(x + \delta) \approx \nabla f(x) + H_f(x)\delta = 0$$

where $H_f(x) = (\frac{\partial f}{\partial x_i \partial x_j})_{ij}$ is the Hessian matrix. At each iteration then we update x_{old} to

$$x_{new} = x_{old} - H_f^{-1}(x)\nabla f(x)\,.$$

Note, Newtons Method converges much faster than gradient decent, but requires computing higher derivatives which might not exist, or may be hard to compute. As a result, it is often not used.

Fitting Logistic Regression

We will discuss fitting the binary label case. Let y_i take probabilities $k \in \{0, 1\}$. Under the logistic assumption we set

$$\delta_1(x) = \frac{\exp(x^T\beta)}{1 + \exp(x^T\beta)}, \quad \delta_0(x) = \frac{1}{1 + \exp(x^T\beta)},$$

where we have again absorbed β_0 into β. Then

$$\begin{aligned}
\ell(\beta) &= \sum_{i=1}^{N} \log \mathbb{P}(G = y_i | X = x_i; \beta) \\
&= \sum_{i=1}^{N} y_i \log(\delta_1(x_i)) + (1 - y_i) \log(1 - \delta_0(x_i)) \\
&= \sum_{i=1}^{N} y_i x_i^T \beta - \log(1 + e^{x_i^T \beta}).
\end{aligned}$$

Fitting Logistic Regression

Denoting by $\mathbf{d}$ the vector whose i'th coordinate is $\delta_1(x_i)$ fitted to β^{old}, the gradient of

$$\ell(\beta) = \sum_{i=1}^{N} y_i x_i^T \beta - \log(1 + e^{x_i^T \beta}$$

is

$$\nabla \ell(\beta) = \mathbf{X}^T(\mathbf{y} - \mathbf{d}) .$$

Letting $\mathbf{W}$ be the $N \times N$ diagonal matrix with i'th entry $\delta_1(x_i)$ fitted to β^{old}, the Hessian is

$$H = \frac{\partial^2 \ell(\beta)}{\partial \beta \partial \beta^T} = -\mathbf{X}^T \mathbf{W} \mathbf{X} .$$

Then

$$\beta^{new} = \beta^{old} + (\mathbf{X}^T \mathbf{W} \mathbf{X})^{-1} \mathbf{X}^T (\mathbf{y} - \mathbf{p}) .$$

Fitting Logistic Regression

Logistic regression is a powerful tool since Newtons Method can be used directly. In addition, it is an explanatory tool, since in general it's coefficients have readily available, explicit interpretations. As a result, Newtons Method is often used in data analysis and situation in which one would like to actually explain outcomes. But it is a power tool in machine learning, and a corner stone of the analysis of categorical variables.

Extra: Bayes Classifier

Multilabel Classification: Loss functions

For multiple labels, the 0-1 loss function can be generalized. For any $K \times K$ matrix L that is 0 on the diagonal and positive off diagonal,

$$L = \begin{pmatrix} 0 & 1 \\ 2 & 0 \end{pmatrix} \qquad \text{(for example)}$$

we can define the loss function $\ell(j, k) = L_{jk}$ where L_{jk} is the "price" for misclassifying k as j. The expected prediction error is

$$EPE = E[\ell(Y, \hat{G})] \,.$$

Of course, the most common error is to set all off diagonal entries to 1, punishing all misclassifacations equally.

Multilabel Classification: Bayes Classifier

The Bayes optimal predictor can be found by conditionalizing:

$$EPE = E_{\mathcal{T}}[\ell(Y, \hat{G})] = E_X E_{Y|X}[\ell(Y, \hat{G}(X))]$$
$$= E_X \left[\sum_{k=1}^{K} \ell[k, \hat{G}(X)] \cdot \mathbb{P}(\, k|X\,) \right] .$$

As before, it's clearly sufficient to minimize EPE pointwise, so

$$\hat{G}(x) = \underset{g \in \mathcal{G}}{\text{argmin}}\ E_X \left[\sum_{k=1}^{K} \ell[k, g] \cdot \mathbb{P}(\, k|X = x\,) \right] .$$

With 0-1 loss, this is more simply

$$\hat{G}(x) = \underset{g \in \mathcal{G}}{\text{argmin}}\ \mathbb{P}(g|X = x)\,.$$

Multilabel Classification: Bayes Classifier

In words, (for 0-1 loss) at each point x, the Bayes classifier returns the value g that is most probable at that point

$$\hat{G}(x) = \underset{g \in \mathcal{G}}{\operatorname{argmin}} \, \mathbb{P}(g|X = x) \, .$$

For more general loss, the Bayes classifier returns the value g that yields the smallest total potential for loss.

The error rate of the Bayes classifier is called the Bayes rate, and is the theoretical minimum of loss.

Multilabel Classification: Methods

Our goal of course is to get as close to the Bayes classifier as possible. We will first consider regression on each of the functions $\hat{f}_k$, modeling them as linear and functions.

The regression approach is a member of a class of methods that model discriminant functions δ_i. If we allow ourselves to make assumptions about the background distribution, we will see that we can derive quite good algorithms for 2d visualization by comparing probabilities.

Finally, we will look at using logistic regression to smoothly interpolate between discrete probabilities.

References:

Projection of Normal distribution taken from Wikimedia Foundation.

This lecture covers material from Chapter 4 of Elements of Statistical Learning II.