Topic 6. Classification

Examples of Classification

- An online banking service must be able to determine whether or not a transaction being performed is fraudulent, on the basis of the IP address, past transaction history, and so forth.
- On the basis of DNA sequence data for a number of patients with or without a given disease, a biologist would like to figure out which DNA mutations are disease causes.
- A patient with a set of symptoms that could possibly be attributed to one of three types of diabetes. Which one does the individual have?

General Setup

- Assume $y \in \{1, \dots, K\}$ is qualitative and $\mathbf{x} \in R^p$.
- A classifier $G: R^p \to \{1, \dots, K\}$.
- A desirable $G(\mathbf{x})$ is to minimize the misclassification error

$$\mathrm{err}(G) = P\big(y \neq G(\mathbf{x})\big) = E\Big(I\big(y \neq G(\mathbf{x})\big)\Big)$$

where $I(\cdot)$ is an indicator function.

Some Concepts

- **Classification function** $h_k(\mathbf{x})$: $R^p \to R$, $k = 1, \ldots, K$
- **Classifier** $G(\mathbf{x})$ is set as

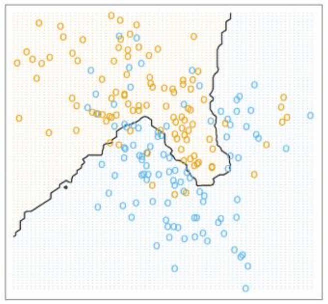
$$G(\mathbf{x}) = \underset{k}{\operatorname{argmax}}\, h_k(\mathbf{x})$$

- Estimate $h_k(\mathbf{x})$ based on the available training data, leading to

$$\hat{G}(\mathbf{x}) = \underset{k}{\operatorname{argmax}}\, \hat{h}_k(\mathbf{x})$$

- **Classification boundary** between classes k and l:

$$\{\mathbf{x}: h_k(\mathbf{x}) = h_l(\mathbf{x})\}$$



A Special Case: Binary Classification

- $K = 2$ and the class labels encoded as $\{0, 1\}$ or $\{-1,1\}$.

- A classifier $G(\mathbf{x})$ is set as

$$G(\mathbf{x}) = I(h(\mathbf{x}) > 0.5) \text{ or } G(\mathbf{x}) = \text{sign}\big(h(\mathbf{x})\big)$$

where $h(\mathbf{x}): R^p \to R$ is the classification function.

- Estimate $h(\mathbf{x})$ from the training data, and then

$$\hat{G}(\mathbf{x}) = I\big(\hat{h}(\mathbf{x}) > 0.5\big) \text{ or } \hat{G}(\mathbf{x}) = \text{sign}\left(\hat{h}(\mathbf{x})\right)$$

- Classification boundary between the two classes is

$$\{\mathbf{x}: h(\mathbf{x}) = 0.5\} \text{ or } \{\mathbf{x}: h(\mathbf{x}) = 0\}$$

Linear Regression for Classification

- Suppose Y has three levels: type 1, type 2 and gestational
- To model it with linear regression, consider the coding

$$Y = \begin{cases} 1 & if\ type\ 1 \\ 2 & if\ type\ 2 \\ 3 & if\ gestational \end{cases}$$

- Fit a regression of Y against the predictors

$$Y = \beta_0 + X^T\beta + \epsilon$$

where the noise assumptions may not hold.

Issues about Coding

- The coding is problematic as the difference between type 1 and type 2 diabetes can be drastically different from the difference between type 2 and gestational diabetes.
- One could choose to code gestational as 1, type 1 as 2, and type 2 as 3, which could lead to a different regression model.
- It is less a problem for binary response, by converting it to a dummy variable.
- Fit a linear regression model to the dummy variable, and predict based on whether the fitted response > 0.5.

Coding with Dummy Variables

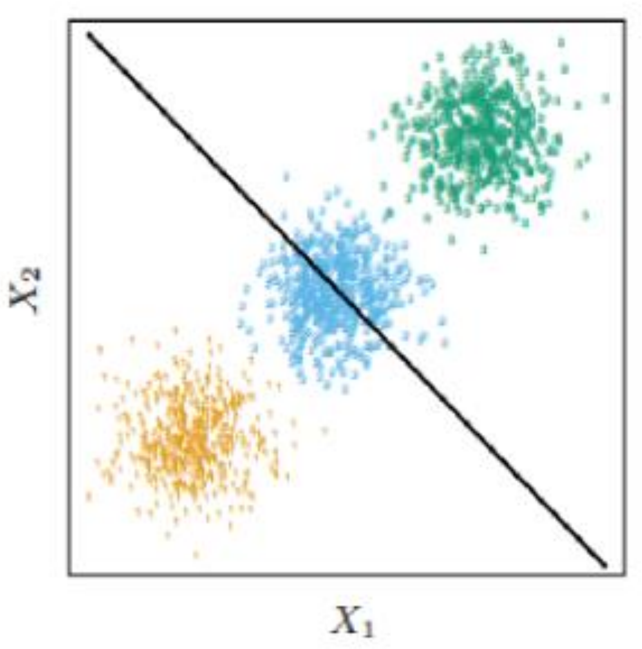
- For qualitative response with more than 2 levels, consider the indicator response matrix

$$y = \begin{cases} 3 \\ 2 \\ 1 \end{cases} \rightarrow \mathbf{Y} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

- Fit a linear regression model $\hat{h}_k(X)$ to each column of $\mathbf{Y}$.
- The final classifier is then $\hat{G}(X) = \underset{k}{\operatorname{argmax}}\, \hat{h}_k(X)$.

Further Issues

- The rationale is to estimate $p_k(X) = P(Y = k|X)$ for $k = 1, \dots, K$.
- Yet the estimated $\hat{h}_k(X)$ can be less than 0 or greater than 1, making the estimator inefficient in estimating $p_k(X)$.
- Masking problem



Bayes Rule

- The optimal classifier, or the **Bayes rule**, is the one minimizing $\mathrm{err}(G)$,

$$G^*(X) = \operatorname*{argmax}_k p_k(X)$$

- Some methods attempt to estimate $p_k(X)$
  - Discriminant analysis, logistic regression, classification tree, deep neural network
- Other methods attempt to estimate $G^*(X)$ directly
  - Support vector machine, Boosting, Bagging

Discriminant Analysis

- $f_k(X)$ = conditional density of X in class $y = k$
- $\pi_k = P(y = k)$ is the class prior
- Bayes theorem

$$p_k(X) = P(y = k|X) = \frac{\pi_k f_k(X)}{\sum_{l=1}^{K} \pi_l f_l(X)}$$

- Comparing $p_k(X)$ can be simplified to comparing $\pi_k f_k(X)$.
- Various assumptions on $f_k(X)$ lead to LDA, QDA, Naïve Bayes,….

Linear Discriminant Analysis (LDA)

- Assume $X|y = k \sim N_p(\mu_k, \Sigma)$ for $k = 1, \dots, K$, then

$$f_k(X) = \frac{1}{(2\pi)^{1/2}|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(X-\mu_k)^T\Sigma^{-1}(X-\mu_k)\right\}$$

- To compare $p_k(X)$, we have

$$\log\frac{p_k(X)}{p_l(X)} > 0 \Leftrightarrow p_k(X) > p_l(X)$$

- Simple algebra yields that

$$\log\frac{p_k(X)}{p_l(X)} = \log\left(\frac{\pi_k}{\pi_l}\right) - \frac{1}{2}(\mu_k+\mu_l)^T\Sigma^{-1}(\mu_k-\mu_l) + X^T\Sigma^{-1}(\mu_k-\mu_l)$$

which is a linear function in X.

LDA (Cont.)

- The quadratic terms of X vanish because of the equal covariance assumption across classes.
- Given a training dataset $(X_i, y_i)_{i=1}^n$, if π_k's are unavailable, $\hat{\pi}_k = n_k/n$, where $n_k = \sum_{i=1}^n I(y_i = k)$.
- Estimate μ_k by centroid in class k,

$$\hat{\mu}_k = \frac{1}{n_k} \sum_{\{i:\, y_i=k\}} X_i$$

- Estimate Σ by pooled within-class covariance matrix,

$$\hat{\Sigma} = \frac{1}{n-K} \sum_{i=1}^{n} (X_i - \hat{\mu}_{y_i})(X_i - \hat{\mu}_{y_i})^T$$

LDA (Cont.)

- A useful decomposition $\log \frac{p_k(X)}{p_l(X)} = \delta_k(X) - \delta_l(X)$, where

$$\delta_k(X) = \log(\pi_k) - \frac{1}{2}\mu_k^T \Sigma^{-1} \mu_k + X^T \Sigma^{-1} \mu_k$$

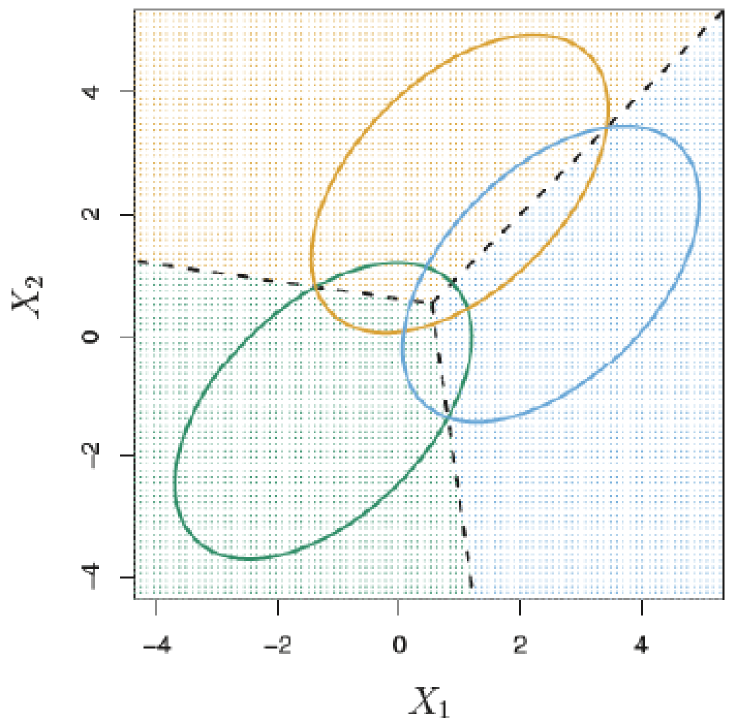
is the **discriminant function**.

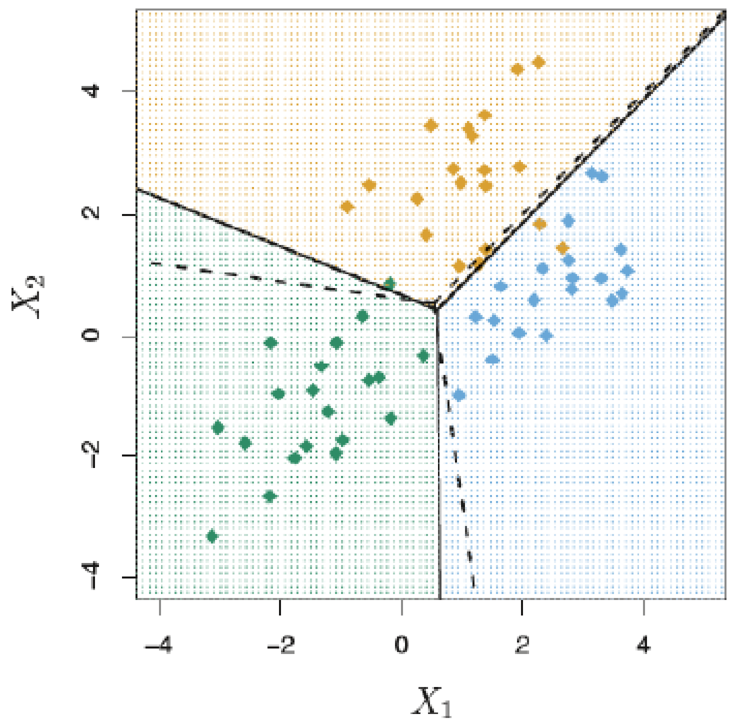
- Classify X to class k with the largest $\delta_k(X)$, since

$$\underset{k}{\operatorname{argmax}}\, p_k(X) = \underset{k}{\operatorname{argmax}}\, \delta_k(X)$$

An Illustrative Example

- Three normal distributions with same covariance and different means; 95% contours of constant density





Dashed: Bayes rule
Solid: LDA

Quadratic Discriminant Analysis (QDA)

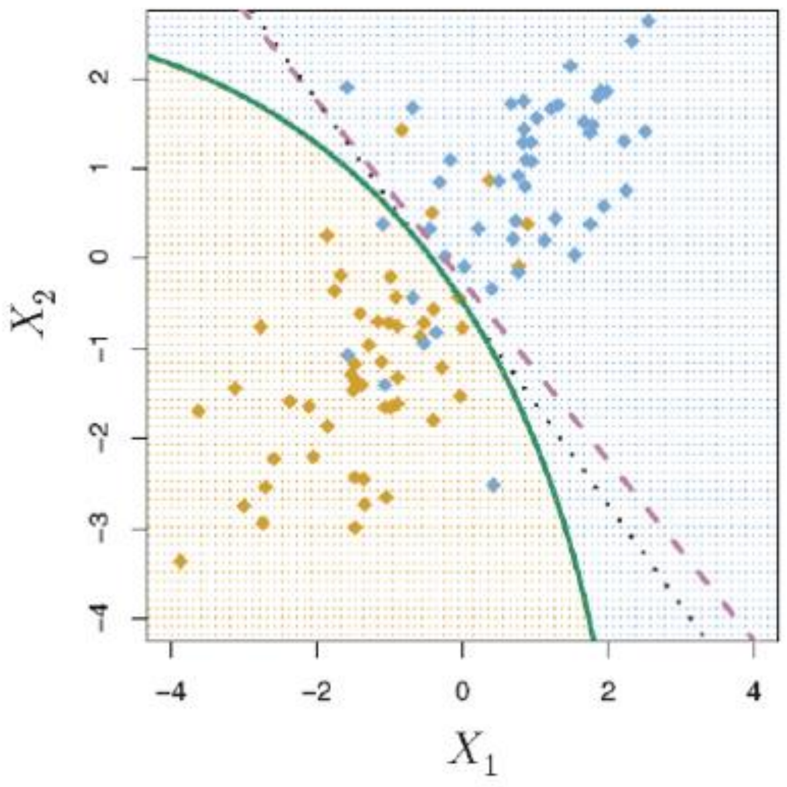
- Assume $X|y = k \sim N_p(\mu_k, \Sigma_k)$ for $k = 1, \ldots, K$, then

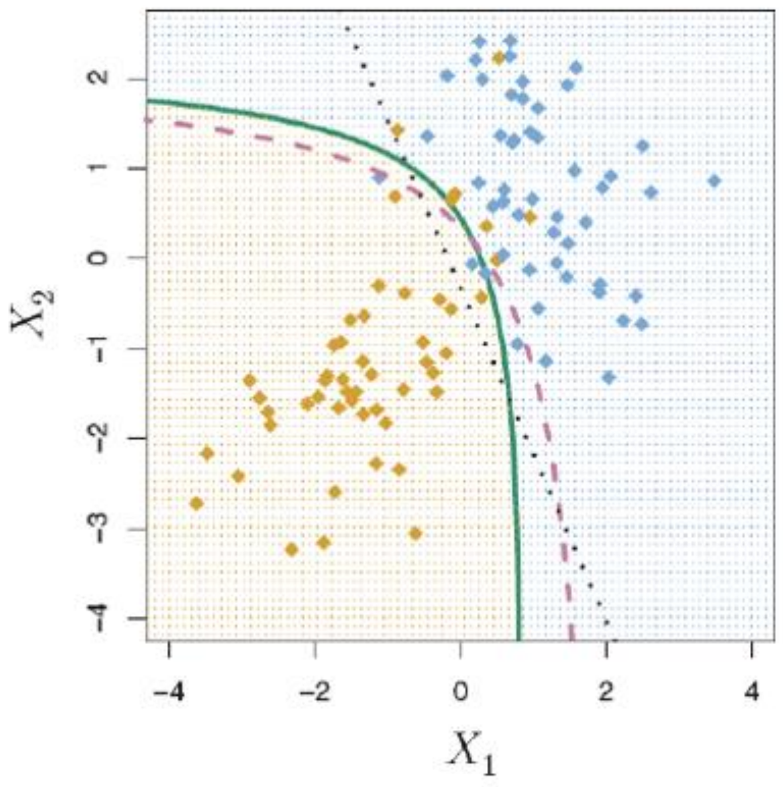
$$\delta_k(X) = \log(\pi_k) - \frac{1}{2} log|\Sigma_k| - -\frac{1}{2}(X - \mu_k)^T \Sigma_k{}^{-1}(X - \mu_k)$$

  - The quadratic term of X now is necessary.
  - $\hat{\pi}_k = n_k / n$
  - μ_k is estimated by the centroid in each class k.
  - Σ_k is estimated by sample covariance matrix in each class

$$\hat{\Sigma}_k = \frac{1}{n_k - 1} \sum_{\{i:\, y_i = k\}} (X_i - \hat{\mu}_k)(X_i - \hat{\mu}_k)^T$$

LDA and QDA





Purple: Bayes rule
Black: LDA
Green: QDA

Naïve Bayes

- Recall the Bayes theorem

$$P(y = k|X) = \frac{\pi_k f_k(X)}{\sum_{l=1}^{K} \pi_l f_l(X)}$$

- To use this rule in practice, we need estimates for π_k and $f_k(X)$, $k = 1, \dots, K$. Estimating the priors is straightforward, but estimating the density functions are challenging.

- In LDA and QDA, strong normality assumption has been made to simplify the task.

Naïve Bayes

- The Naïve Bayes classifier does not assume the density functions belong to a particular family of distributions. It is based on another assumption: **Within each class, the predictors are independent**.
- This means that for each class

$$f_k(X) = f_{k1}(x_1) \times f_{k2}(x_2) \times \cdots \times f_{kp}(x_p)$$

- By assuming the independence between predictors, we eliminate the need to figure out their associations, thus simplifying the estimation task.
- We can estimate the density function of each predictor in different ways, such as assuming normality or using non-parametric methods.

Performance of Naïve Bayes

- The independence assumption in Naïve Bayes is made for convenience and may not be true in most scenarios, but it often leads to pretty decent results, especially when n is not large enough relative to p.

- This assumption introduces some bias, but reduces variance, leading to a classifier that works quite well in practice as a result of the bias-variance trade-off.

Binary Logistic Regression

- In binary classification with $y \in \{0,1\}$, one natural choice is to model $p(X) = P(y = 1|X)$.
- Linear regression assumes $p(X) = \beta_0 + X^T\beta$, but the fitted value can be less than 0 or larger than 1.
- Linear logistic regression assumes

$$p(X) = \frac{e^{\beta_0 + X^T\beta}}{1 + e^{\beta_0 + X^T\beta}}$$

  - The fitted value is guaranteed to be in [0, 1].
  - $\frac{e^z}{1+e^z}$ is the logistic function, which maps R onto [0, 1].

Logit and Odds Ratio

- An equivalent form,

$$\text{logit}(p(X)) = \log \frac{p(X)}{1 - p(X)} = \beta_0 + X^T\beta$$

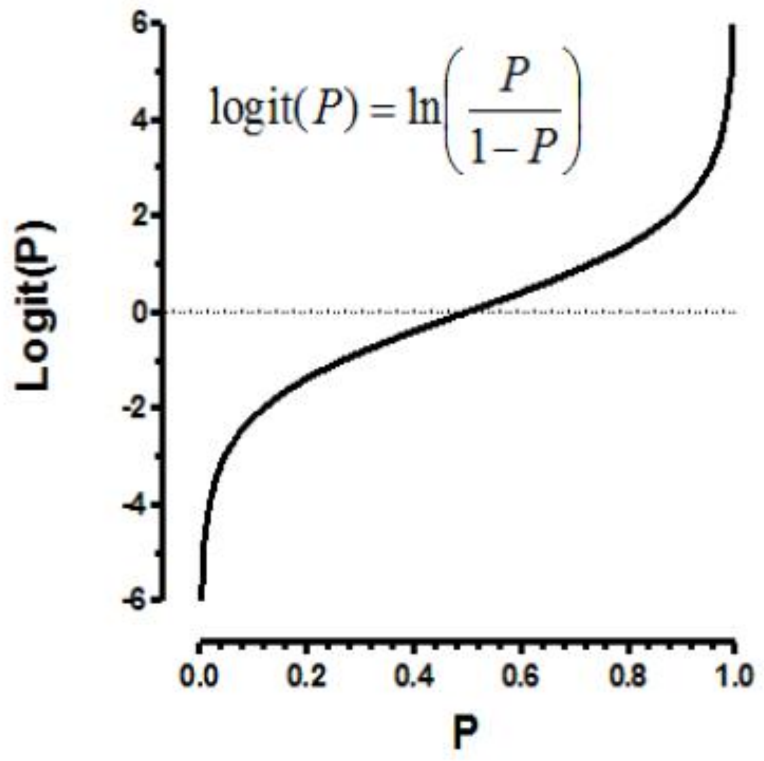
where $\text{odds}(X) = \frac{p(X)}{1-p(X)} = e^{\beta_0 + X^T\beta}$ is called the odds, and log-odds is also known as the **logit** function.

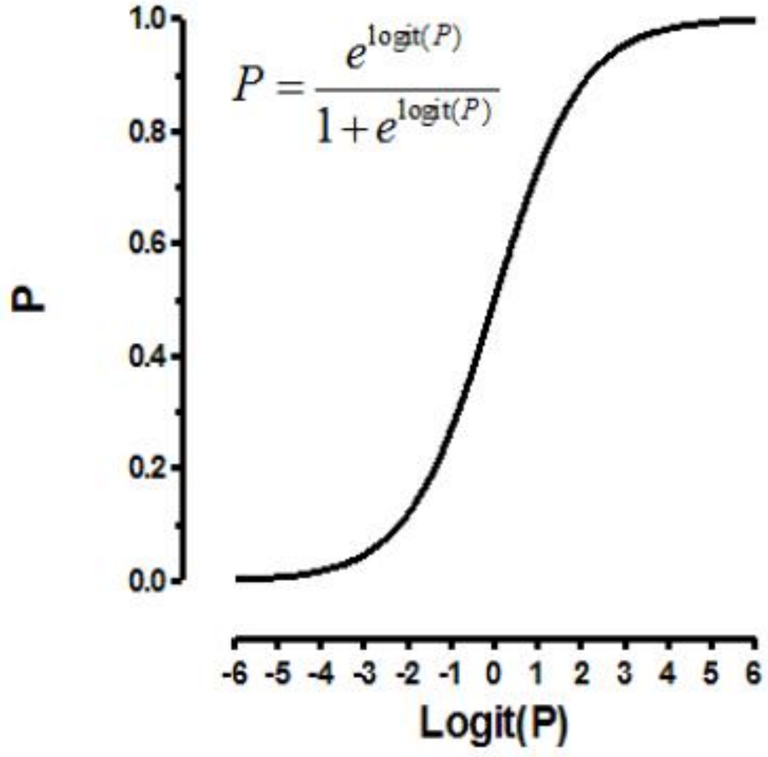
- For each predictor X_j, the odds ratio

$$OR_j = \frac{\text{odds}\left(X_1, \dots, X_j + 1, \dots, X_p\right)}{\text{odds}\left(X_1, \dots, X_j, \dots, X_p\right)} = e^{\beta_j}$$

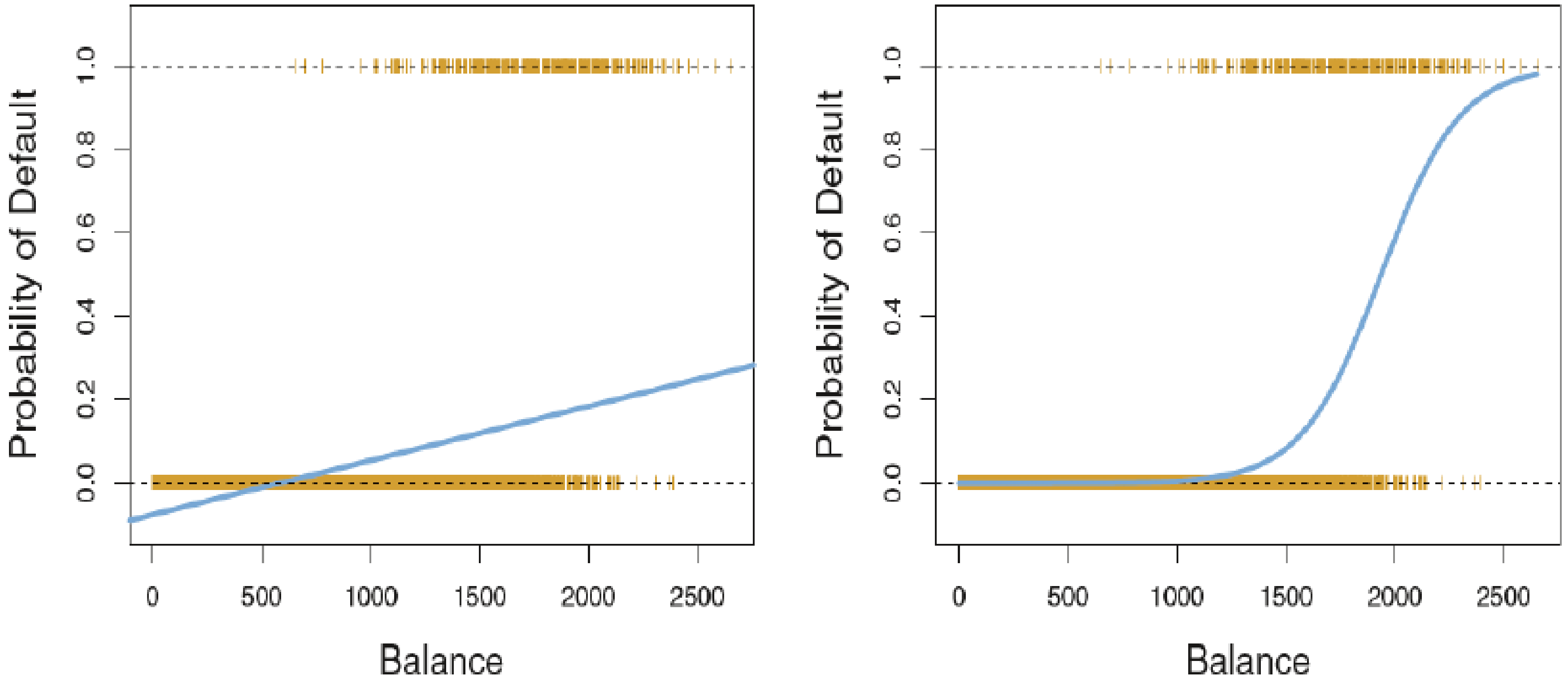
indicating the odds multiple with e^{β_j} per 1-unit increase in X_j.

Logit Function





Comparison b/w Linear and Logistic Regression



Fitting Binary Logistic Regression

- Let $\tilde{\beta} = (\beta_0, \beta)$, then its log-likelihood is

$$l(\tilde{\beta}) = \log\left(\prod_{i=1}^{n} p(\mathbf{x}_i)^{y_i}(1 - p(\mathbf{x}_i))^{1-y_i}\right)$$

$$= \sum_{i=1}^{n}\left(y_i \log p(\mathbf{x}_i) + (1 - y_i)\log\left(1 - p(\mathbf{x}_i)\right)\right)$$

$$= \sum_{i=1}^{n}\left(y_i \log \frac{p(\mathbf{x}_i)}{1 - p(\mathbf{x}_i)} + \log\left(1 - p(\mathbf{x}_i)\right)\right)$$

- Estimate the coefficients by maximizing $l(\tilde{\beta})$.

MLE for Logistic Regression

- To maximize $l(\tilde{\beta})$, simple algebra yields that

$$l(\tilde{\beta}) = \sum_{i=1}^{n} \left(y_i(\tilde{\mathbf{x}}_i{}^T\tilde{\beta}) - \log\left(1 + \exp(\tilde{\mathbf{x}}_i{}^T\tilde{\beta})\right) \right)$$

$$l'(\tilde{\beta}) = \sum_{i=1}^{n} (y_i - p_i)\tilde{\mathbf{x}}_i = \widetilde{\mathbf{X}}^T(\mathbf{y} - \mathbf{p})$$

$$l''(\tilde{\beta}) = -\sum_{i=1}^{n} p_i(1 - p_i)\tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i{}^T = -\widetilde{\mathbf{X}}^T\mathbf{W}\widetilde{\mathbf{X}}$$

where $\mathbf{p} = (p_1, \dots, p_n)^T$, $p_i = p(\mathbf{x}_i)$, $\widetilde{\mathbf{X}} = (\tilde{\mathbf{x}}_1, \dots, \tilde{\mathbf{x}}_n)^T$, $\tilde{\mathbf{x}}_i = (1, \mathbf{x}_i^T)^T$, and $\mathbf{W} = \mathrm{diag}\{p_1(1-p_1), \dots, p_n(1-p_n)\}$.

Iteratively Reweighted Least Square (IRLS)

- Newton's method updates β iteratively,

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

$$= (\tilde{\mathbf{X}}^T\mathbf{W}\tilde{\mathbf{X}})^{-1}\tilde{\mathbf{X}}^T\mathbf{W}\left(\tilde{\mathbf{X}}\tilde{\beta}^{old} + \mathbf{W}^{-1}(\mathbf{y}-\mathbf{p})\right)$$

$$= (\tilde{\mathbf{X}}^T\mathbf{W}\tilde{\mathbf{X}})^{-1}\tilde{\mathbf{X}}^T\mathbf{W}\mathbf{y}^{\text{new}}$$

- In the last two lines we have re-expressed the Newton-Raphson step as a weighted least squares step, with the new response

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

- This is also to iteratively refit the linear regression model with new weights and responses, and thus the name IRLS.

Multiclass Logistic Regression

- Multiclass logistic regression assumes

$$\log \frac{p_1(X)}{p_K(X)} = \tilde{\mathbf{x}}^T \tilde{\beta}_1$$

$$\log \frac{p_2(X)}{p_K(X)} = \tilde{\mathbf{x}}^T \tilde{\beta}_2$$

$$\log \frac{p_{K-1}(X)}{p_K(X)} = \tilde{\mathbf{x}}^T \tilde{\beta}_{K-1}$$

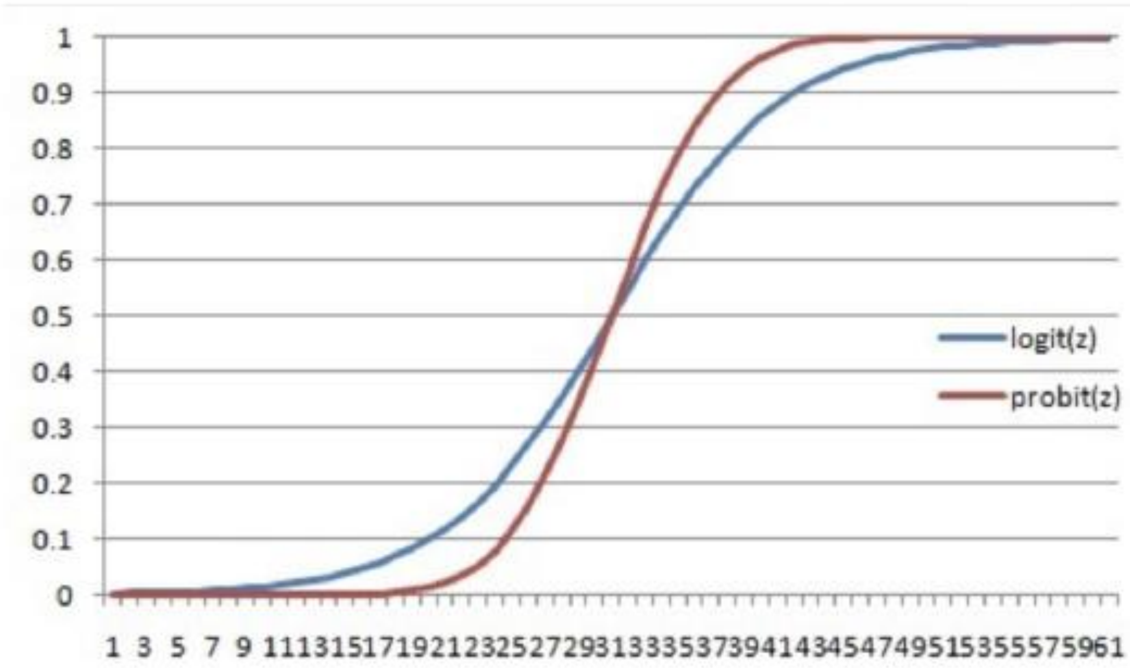
- Coefficients are also estimated by MLE based on the likelihood function of multinomial distribution.

Probit Model

- In binary classification, probit model assumes

$$p(\mathbf{x}) = \Phi(\beta_0 + \mathbf{x}^T \beta)$$

where Φ is the cumulative distribution function of standard normal distribution.



Logistic Regression and LDA

- In binary classification, LDA also leads to

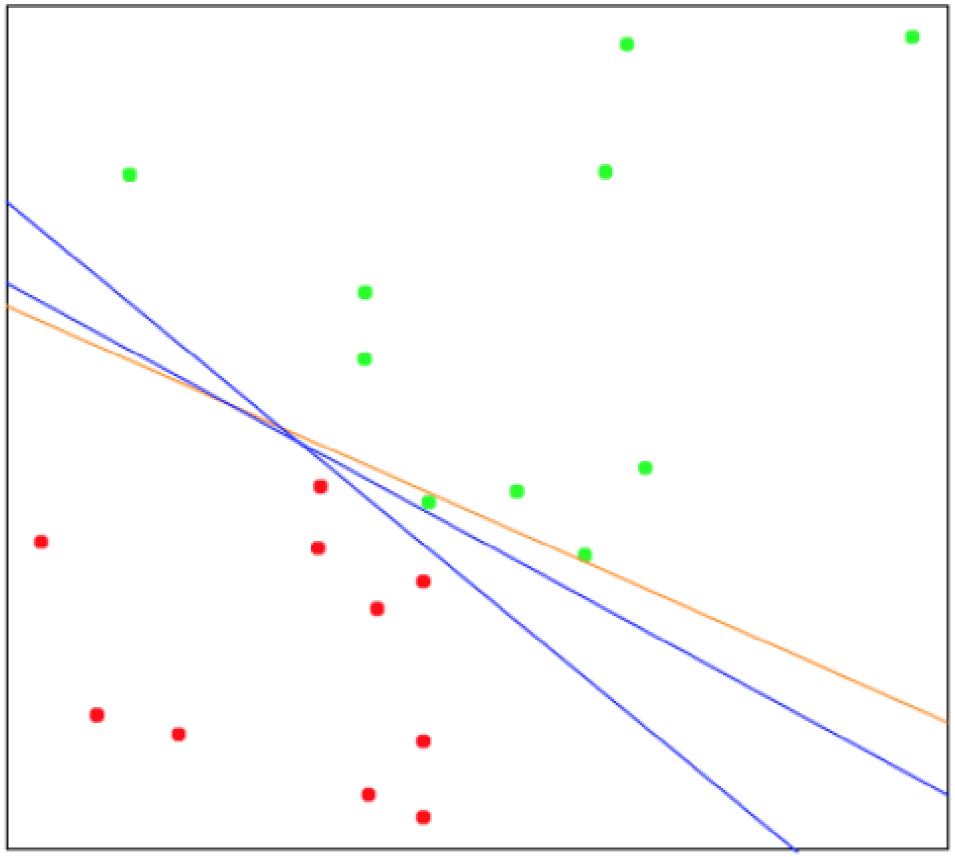
$$\operatorname{logit}(p(\mathbf{x})) = \log \frac{p(\mathbf{x})}{1 - p(\mathbf{x})} = c_0 + \mathbf{x}^T c_1$$

where $c_0 = \log\left(\frac{\pi}{1-\pi}\right) - \frac{1}{2}(\mu_1 + \mu_0)^T \Sigma^{-1} (\mu_1 + \mu_0)$ and $c_1 = \Sigma^{-1}(\mu_1 - \mu_0)$.

- Thus logistic regression and LDA differ only in their fitting procedures
  - Logistic regression estimates the coefficients by MLE.
  - LDA estimates them based on the estimated mean and covariance matrix of a normal distribution.

Separating Hyperplane

- This method constructs linear classification boundaries that try to separate the data into different classes as well as possible.



Geometric View of Classification

- **Feature space**
  - Space formed by the predictors
  - Also referred to as the state space, input space
  - p-dimensional (p predictors), n-points (n observations)

- **Training data**
  - Inputs determine the location in the feature space

$$x_1 = \begin{pmatrix} x_{11} \\ \vdots \\ x_{1p} \end{pmatrix}, \ldots, x_n = \begin{pmatrix} x_{n1} \\ \vdots \\ x_{np} \end{pmatrix}$$

  - Outputs $y_1, \ldots, y_n$ determine the **color** (i.e., **classes**)

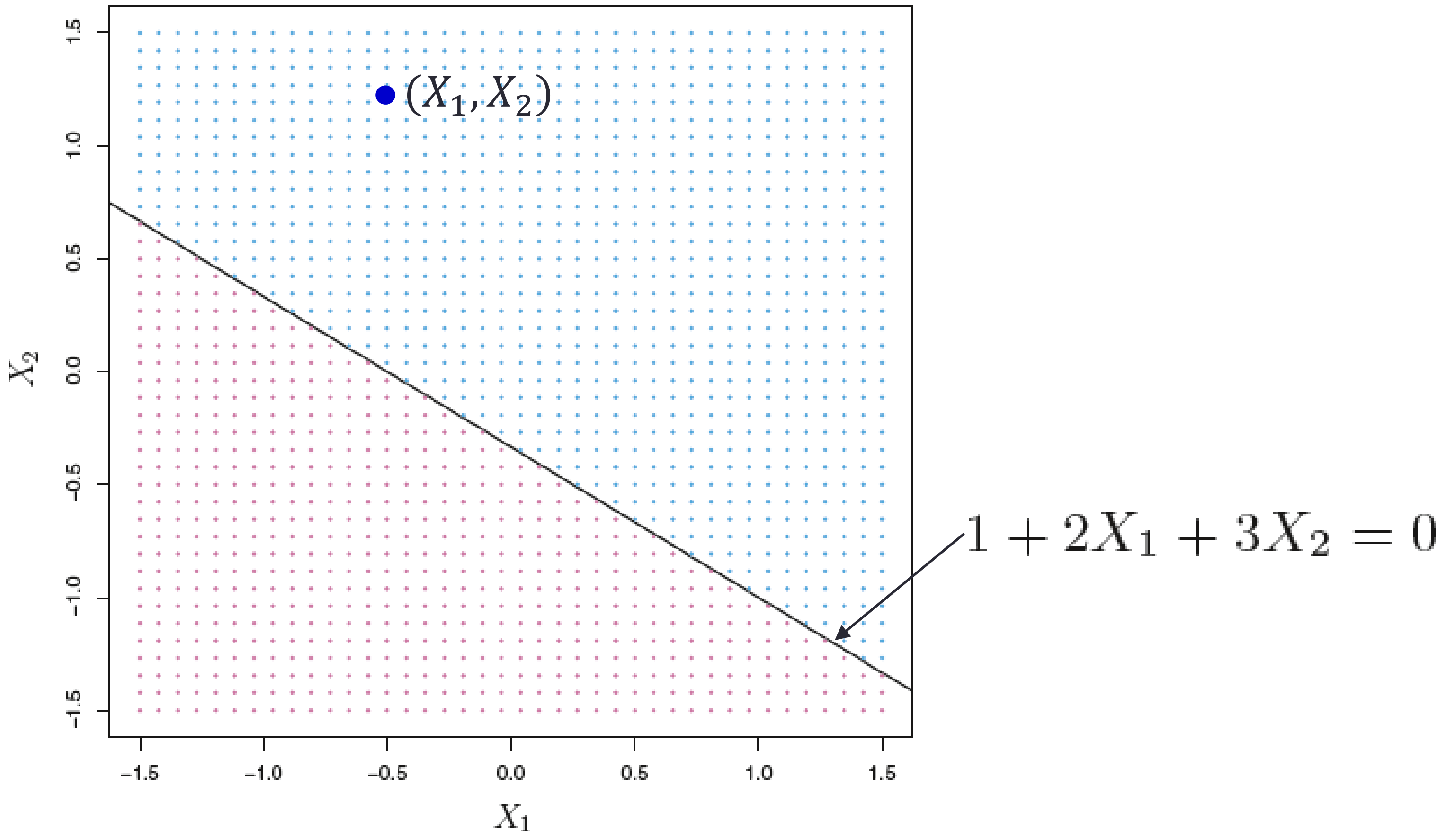
- **Classification**: Find the hyperplane such that a test point

$$x^* = \begin{pmatrix} x_1^* & \ldots & x_p^* \end{pmatrix}^T$$

is assigned the correct class.

Hyperplane

- In a p-dimensional space, a *hyperplane* is a flat affine subspace of dimension $p - 1$. For example, in a two-dimensional space, a hyperplane is a line.



Review of Vector Algebra

- A hyperplane L is defined by the linear equation:

$$L = \{\mathbf{x}: f(\mathbf{x}) = \beta_0 + \mathbf{x}^T\beta = 0\}$$

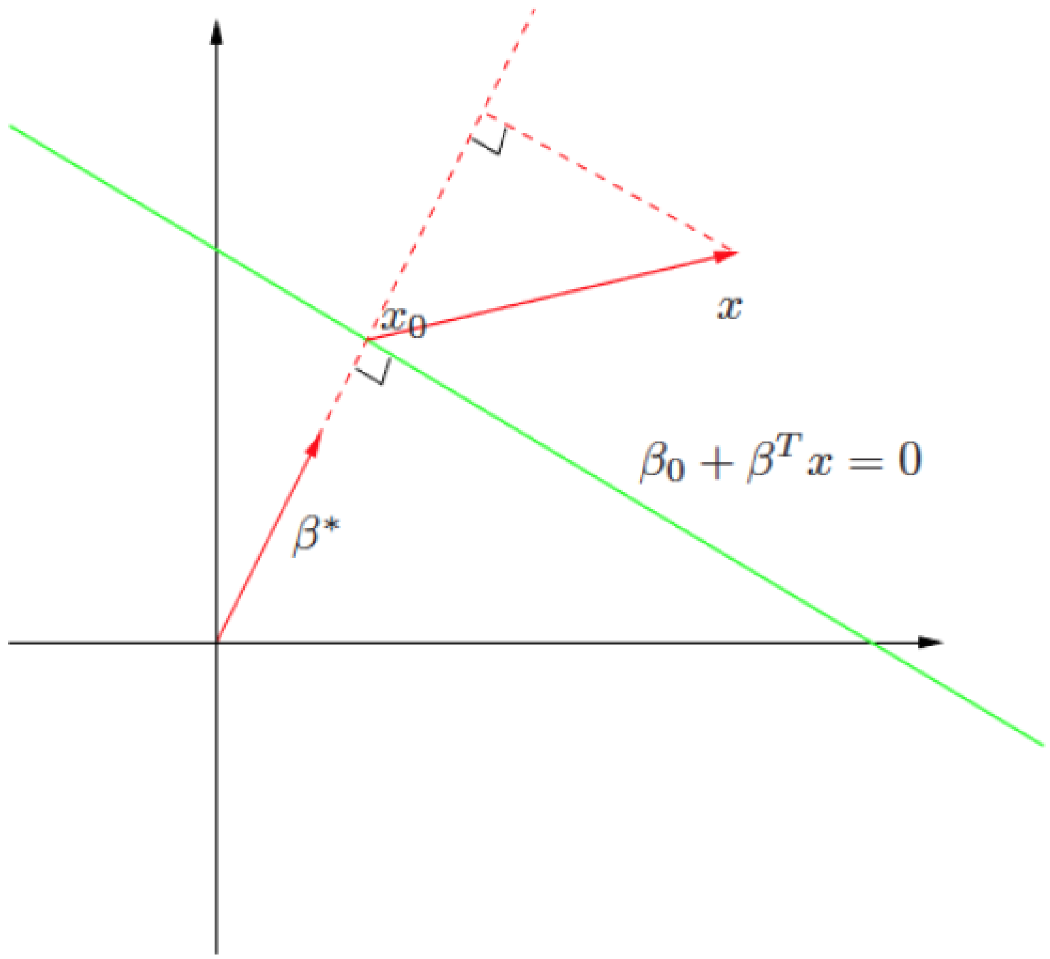
- For any two points $\mathbf{x}_1$ and $\mathbf{x}_2$ lying in L, $(\mathbf{x}_1 - \mathbf{x}_2)^T\beta = 0$, and hence $\beta^* = \beta/||\beta||$ is the vector normal to the surface of L.

- For any point $\mathbf{x}_0$ in L, ${\mathbf{x}_0}^T\beta = -\beta_0$.

- The signed distance of any point $\mathbf{x}$ to L is given by

$$(\mathbf{x} - \mathbf{x}_0)^T\beta^* = \frac{1}{||\beta||}(\mathbf{x}^T\beta + \beta_0)$$

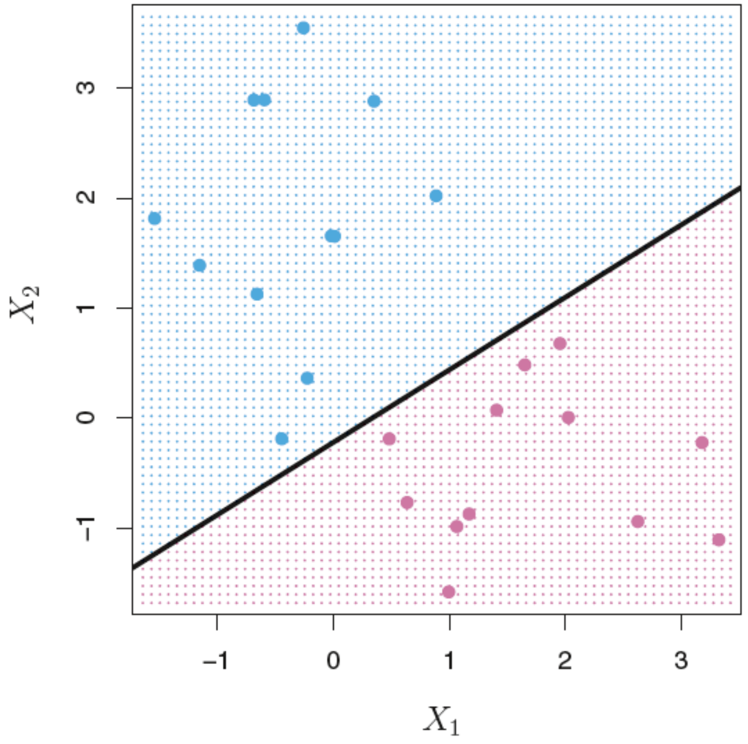
Geometry of A Hyperplane



Separating Hyperplane

- **Separating hyperplane**: separates the training observations perfectly according to their class labels
- Blue: class 1 ($y = 1$)
  Purple: class –1 ($y = -1$)

- $f(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p$
- class 1: $f(x) > 0$
  class –1: $f(x) < 0$



- Property: $\boldsymbol{y_i f(x_i) > 0}$, for all training points $x_1, x_2, \ldots, x_n$

Prediction

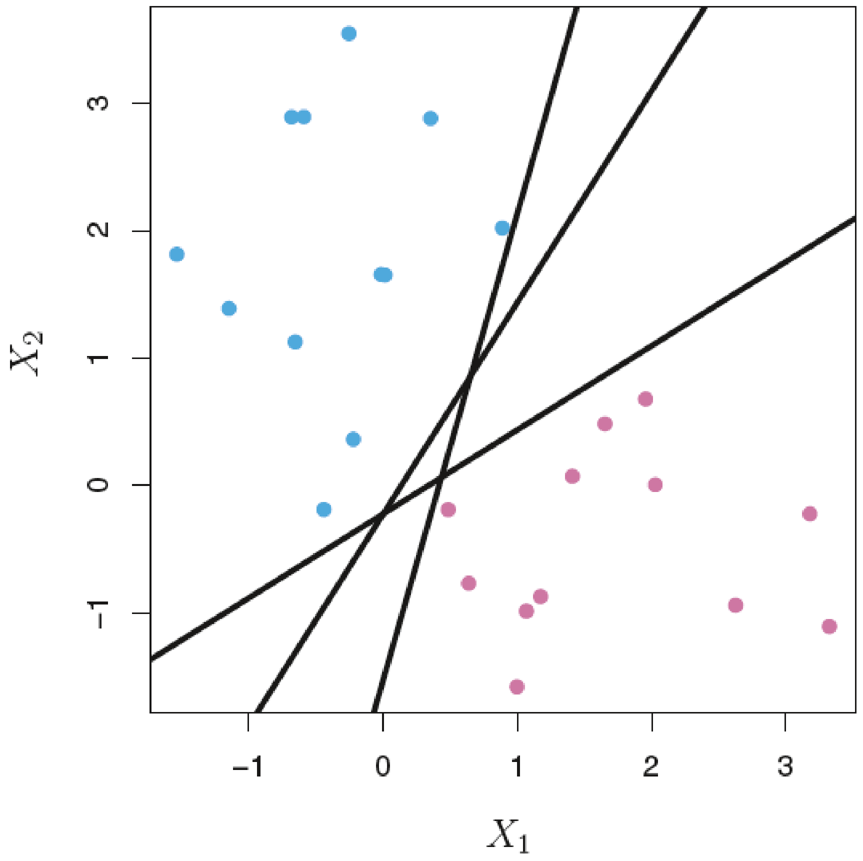
- Given a test point x^*, we will assign it to
  class 1 ($y^* = 1$), if $f(x^*) > 0$
  class –1 ($y^* = -1$), if $f(x^*) < 0$

- If $\boldsymbol{y^* f(x^*)}$ is far from zero, that means the test point lies far from the hyperplane, and so we can be confident about our class assignment for it.

- If $\boldsymbol{y^* f(x^*)}$ is close to zero, that means the test point is located near the hyperplane, and so we are less certain about the class assignment for it.

How to Do It Right?

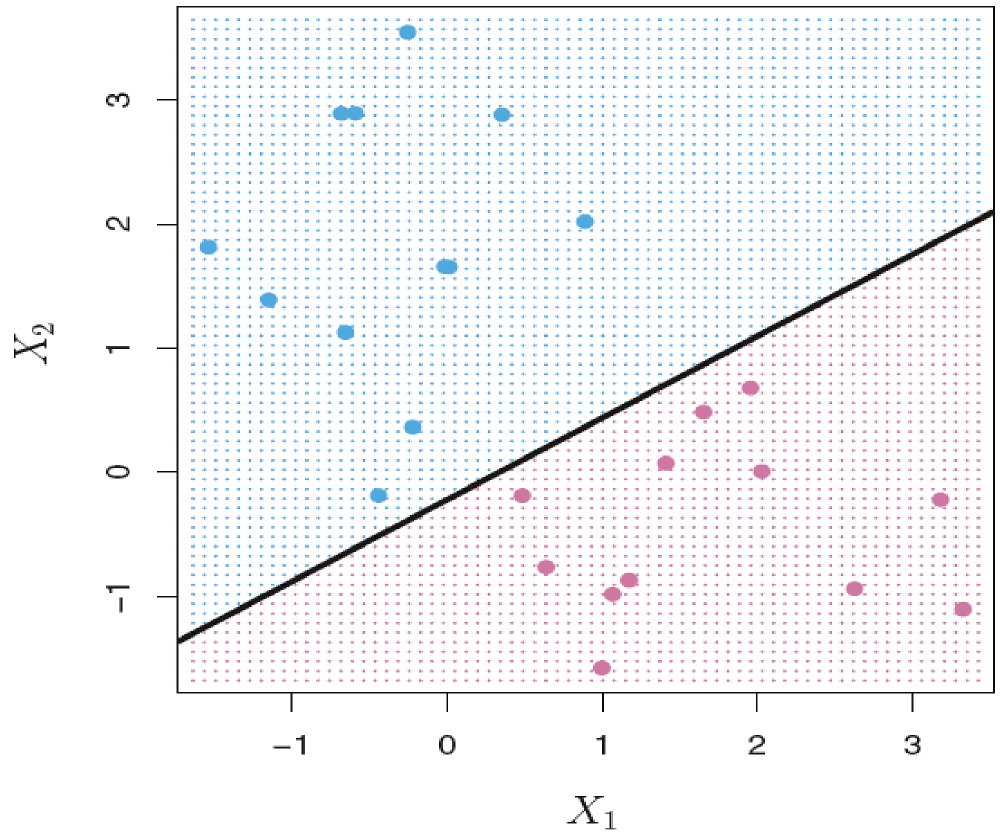
- There may be an infinite number of hyperplanes that separates the training observations perfectly.
- We need to decide which hyperplane to use.



Margin

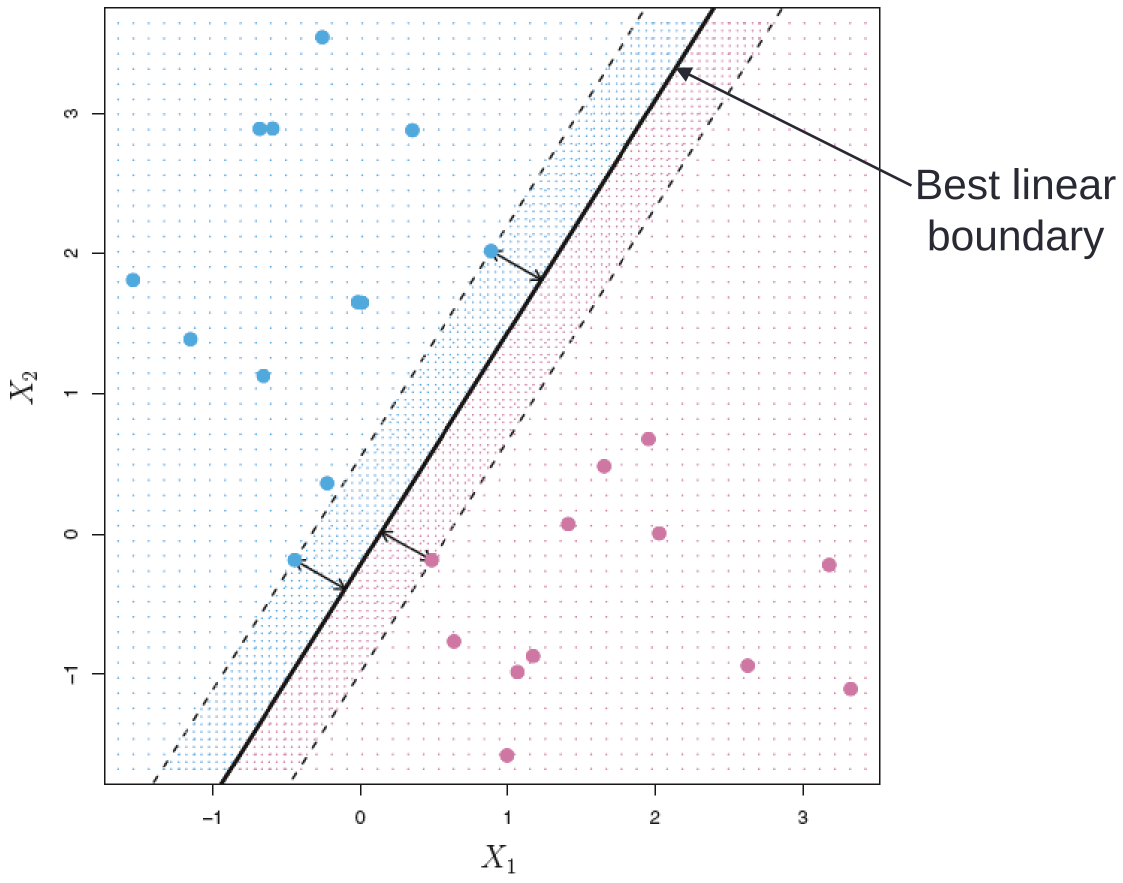
- Suppose we have a separating hyperplane
- Find perpendicular distance from every point to the hyperplane
- Margin: The smallest of such distances, M

i.e., minimum distance from the observations to the hyperplane



Maximal Margin Classifier

- Maximize min-distance (max margin)
- Represent the mid-line of the widest “slab” inserted between the two classes



Optimal Separating Hyperplane

- Consider binary classification with $y_i \in \{1, -1\}$.
- Suppose the two classes can be linearly separated.
- Optimal separating hyperplane separates the two classes and maximizes the distance to the closest point from either class.
- Tend to have better classification performance on test data

$$\max_{\beta,\beta_0} \quad C$$

$$\text{subject to } \frac{1}{\|\beta\|} y_i({\mathbf{x}_i}^T\beta + \beta_0) \geq C,\ i = 1, \dots, n$$

- That is, every point is at least C away from the decision boundary $\beta_0 + \mathbf{x}^T\beta = 0$.

Estimation

- For any solution of the optimization problem, any positively scaled multiple is a solution as well.
- Set $||\beta|| = 1/C$ and the optimization is equivalent to

$$\min_{\beta,\beta_0} \quad \frac{1}{2}||\beta||^2$$

subject to $y_i(\mathbf{x}_i{}^T\beta + \beta_0) \geq 1,\ i = 1, \ldots, n$

- The Lagrange function is

$$L_p = \min_{\beta,\beta_0} \frac{1}{2}||\beta||^2 - \sum_{i=1}^{n} \alpha_i(y_i(\mathbf{x}_i{}^T\beta + \beta_0) - 1)$$

subject to $\alpha_i \geq 0$

Estimation (Cont.)

- Setting derivative to zero, we have

$$\beta = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \quad \text{and} \quad 0 = \sum_{i=1}^{n} \alpha_i y_i$$

- Substitute into L_p, the Wolfe dual form is

$$L_D = \max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i\, y_j \mathbf{x}_i{}^T \mathbf{x}_j$$

subject to $\alpha_i \geq 0$ and $\sum_{i=1}^{n} \alpha_i y_i = 0$

- This can be solved by quadratic programming.

Linear and Quadratic Programming

- Linear programming (LP)

$$\min_{z} \mathbf{c}^T \mathbf{z}$$

$$\text{subject to } A\mathbf{z} \leq \mathbf{b} \text{ and } \mathbf{z} \geq 0$$

- Quadratic programming (QP)

$$\min_{z} \frac{1}{2} \mathbf{z}^T Q\mathbf{z} + \mathbf{c}^T \mathbf{z}$$

$$\text{subject to } A\mathbf{z} \leq \mathbf{b}$$

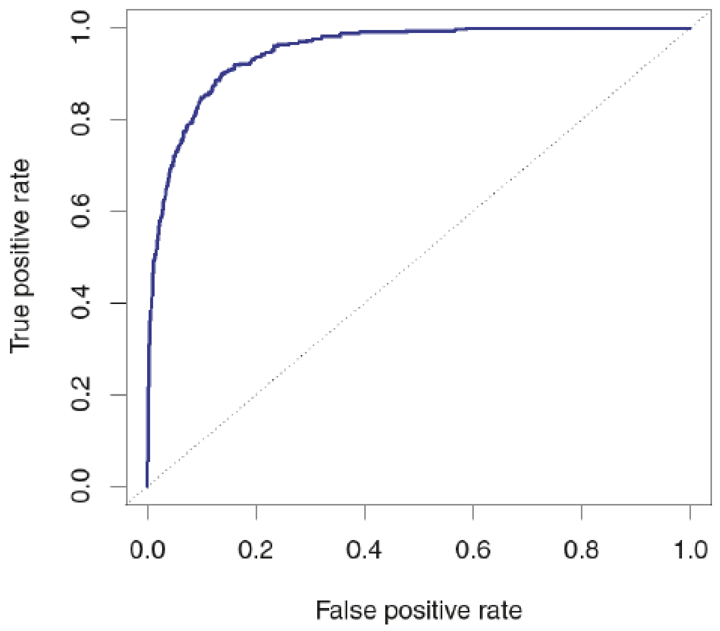
- Implementation is available in most software.

Evaluating Classification Model

- Misclassification error

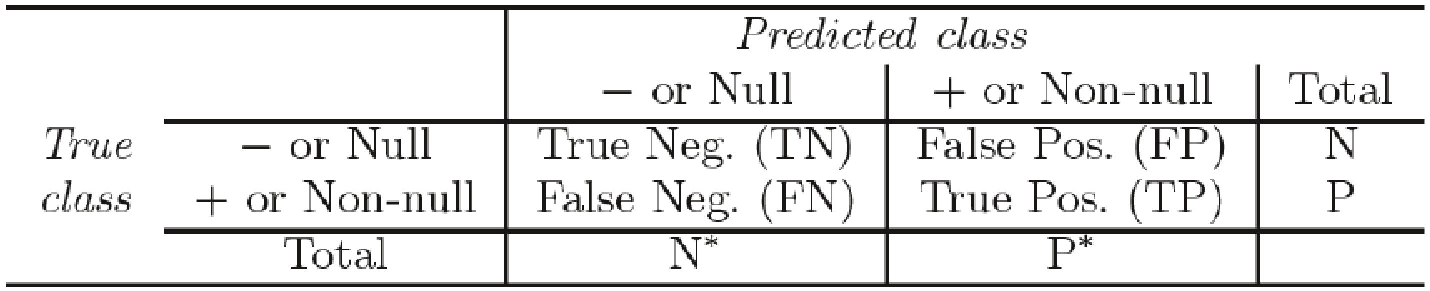
$$\text{test error} = \frac{1}{|\text{test set}|} \sum_{i \in \text{test set}} I\left(\hat{G}(\mathbf{x}_i) \neq y_i\right)$$

- Receiver operating characteristics (ROC) curve



Other Performance Measures

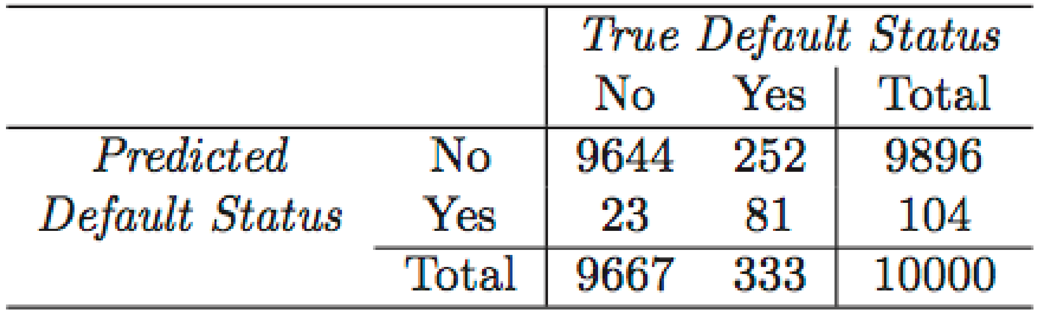
- A few popularly-used terms
  - True positive rate or **Sensitivity** or **Recall** $= TP/P$
  - False positive rate or 1$-$**Specificity** $= FP/N$
  - Positive predictive value or **Precision** $= TP/P^*$
  - F1-score $= 2 \times \frac{\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}}$
  - Area under the ROC curve (AUC)
  - Youden index is $J = \text{Sensitivity} + \text{Specificity} - 1$



		Predicted class		
		$-$ or Null	$+$ or Non-null	Total
True class	$-$ or Null	True Neg. (TN)	False Pos. (FP)	N
	$+$ or Non-null	False Neg. (FN)	True Pos. (TP)	P
	Total	N*	P*	

Example

- Response: Whether a person will default his/her credit card payment (Yes/No = 1/0)



		True Default Status		
		No	Yes	Total
Predicted Default Status	No	9644	252	9896
	Yes	23	81	104
	Total	9667	333	10000

- Misclassification error rate: fraction of people that are incorrectly classified (2.75%)
- Sensitivity (true positive rate): fraction of defaulters that are correctly identified (24.3%)
- Specificity: fraction of non-defaulters that are correctly identified as non-defaulters (99.76%)
- False positive rate (1– Specificity): fraction of non-defaulters that are incorrectly classified as defaulters (0.24%)