Class 3: Classification

MSA 8150: Machine Learning for Analytics

Alireza Aghasi

Institute for Insight, Georgia State University

Introduction

Brief Overview of Maximum Likelihood

- Maximum likelihood (ML) is a statistical estimation technique
- The main goal in ML is estimating the parameters of a statistical model given some sample observations
- Let $x_1, x_2, \cdots x_n$ be samples from a distribution with some unknown parameter θ and joint distribution

$$f(x_1, x_2, \cdots, x_n|\theta)$$

- The maximum likelihood estimate of θ based on the observations $\tilde{x}_1, \tilde{x}_2, \cdots \tilde{x}_n$ is

$$\theta_{ML} = \underset{\theta}{\text{argmax}} \; f(\tilde{x}_1, \tilde{x}_2, \cdots, \tilde{x}_n|\theta)$$

- When $x_1, x_2, \cdots x_n$ are i.i.d samples from a distribution $f(\cdot)$, then

$$f(x_1, x_2, \cdots, x_n|\theta) = f(x_1|\theta)f(x_2|\theta)\cdots f(x_n|\theta)$$

Brief Overview of Maximum Likelihood

Example. We have a normal distribution $\mathcal{N}(\mu, 1)$ and we do not know μ. We take 5 independent samples from this distribution and the values turn out to be

$$\tilde{x}_1 = 2.5377,\ \tilde{x}_2 = 3.8339,\ \tilde{x}_3 = -0.2588,\ \tilde{x}_4 = 2.8622,\ \tilde{x}_5 = 2.3188,$$

what is the ML estimate of μ.

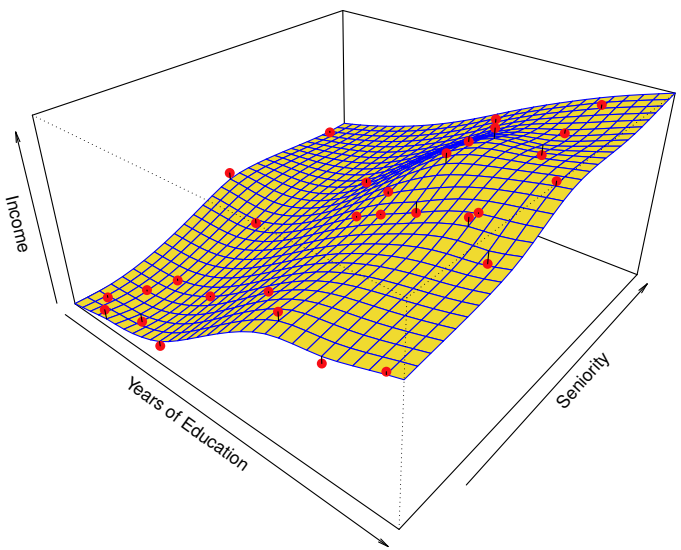
Solution. If we take 5 independent samples $x_1, x_2, \cdots x_5$ from a normal distribution $\mathcal{N}(\mu, 1)$, their joint distribution is

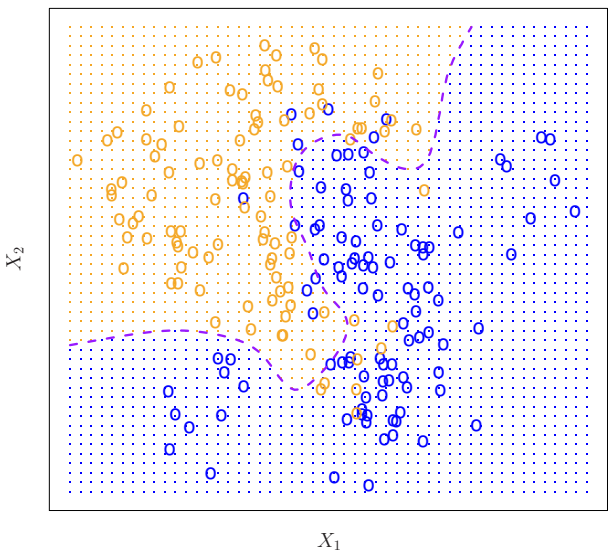
$$f(x_1, x_2, x_3, x_4, x5|\mu) = \prod_{i=1}^{5} \frac{1}{\sqrt{2\pi}} \exp(-\frac{(x_i - \mu)^2}{2}),$$

some basic calculus yields $\mu_{ML} = \frac{\tilde{x}_1 + \tilde{x}_2 + \cdots + \tilde{x}_5}{5} = 2.2587$ (why?)

Classification

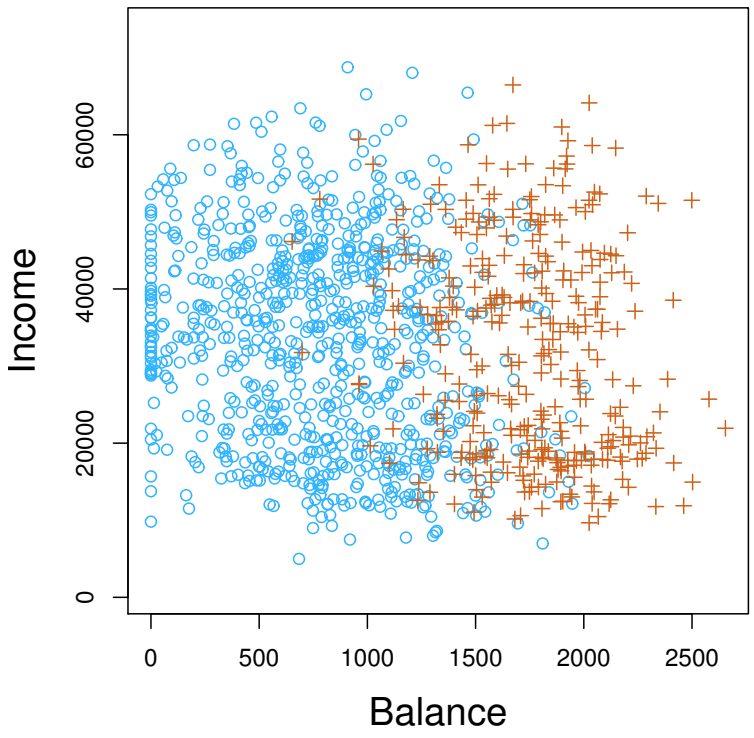
- In many applications, the response is not a quantitative value and instead represents a class, e.g., $y \in \{\text{student, non-student}\}$, $y \in \{\text{while, yellow, green}\}$
- Yet based on the observation of some features, we would like to predict the class (what we refer to as the classification)
- Regression vs classification

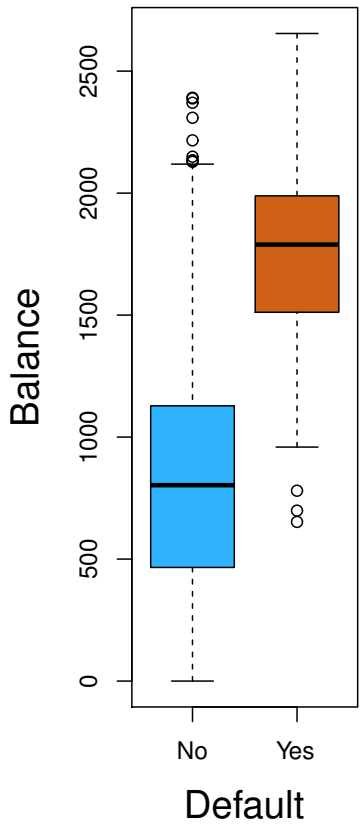


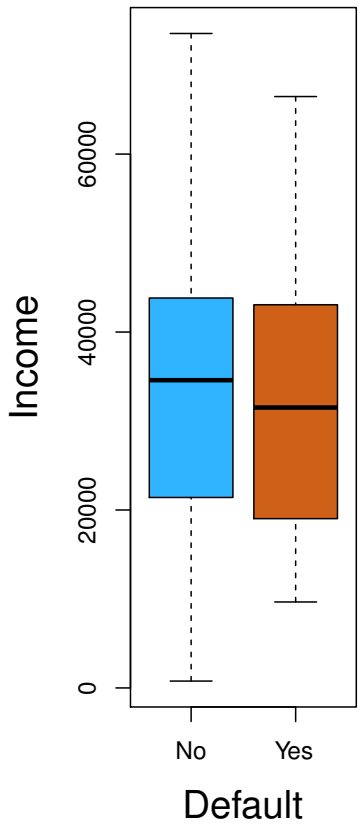


Classification

Example. Predicting default cases on the credit card (unable to pay the credit card), based on the income and current balance







(one immediate observation is probably balance is a more useful feature)

Binary Classification

- In simple regression for a single feature x we fitted a line $y = \beta_0 + \beta_1 x$ to the data
- In binary classification with only one feature, we don't have values any more, but two classes (say class 0 and class 1)
- Can we do the fit in a way that the sign of $\beta_0 + \beta_1 x$ becomes an indicator of the class for us?
- In other words, for a given feature x_t, we make a decision based on the following:

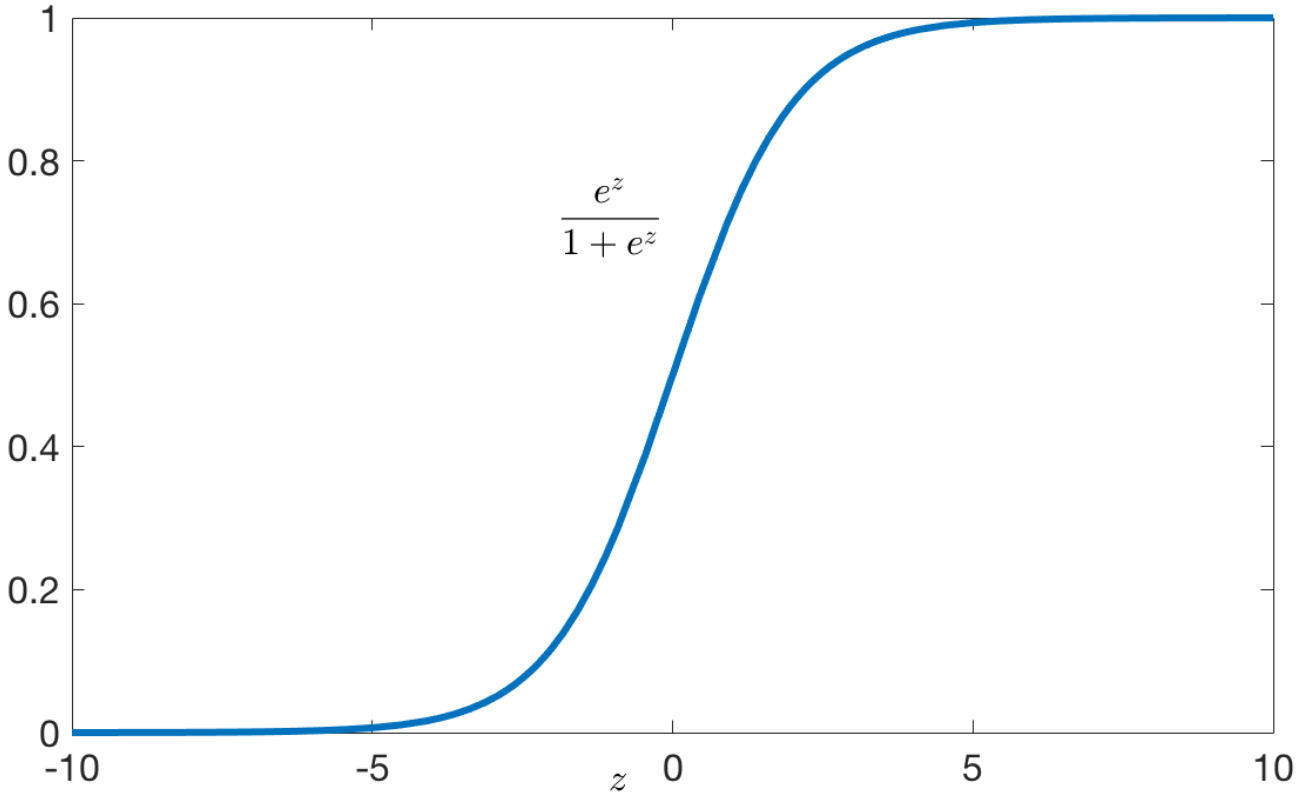
$$y_t = \begin{cases} 1 & \beta_0 + \beta_1 x_t > 0 \\ 0 & \beta_0 + \beta_1 x_t < 0 \end{cases},$$

- A smooth function (called Sigmoid – also inverse Logit) that takes almost binary values $0, 1$ based on the sign of the input z is

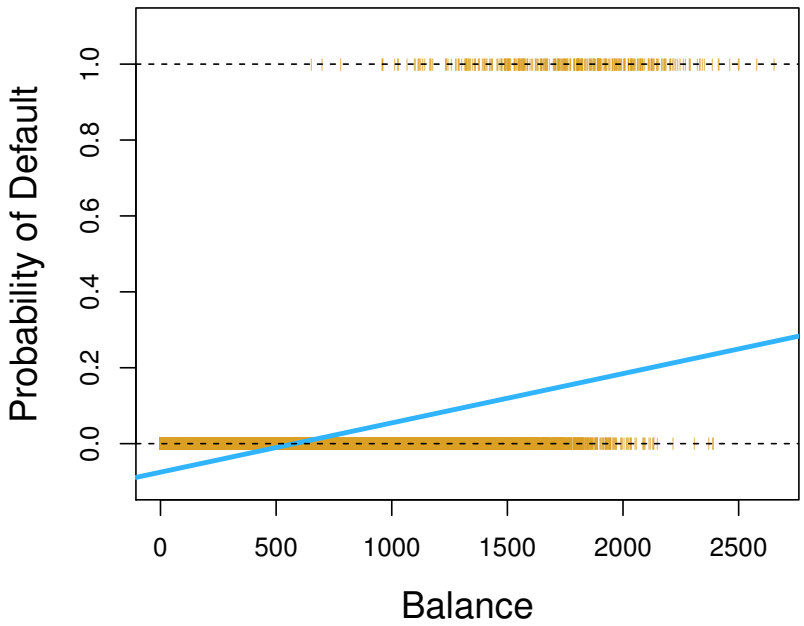
$$\frac{e^z}{1+e^z} \approx \begin{cases} 1 & z >> 0 \\ 0 & z << 0 \end{cases}$$

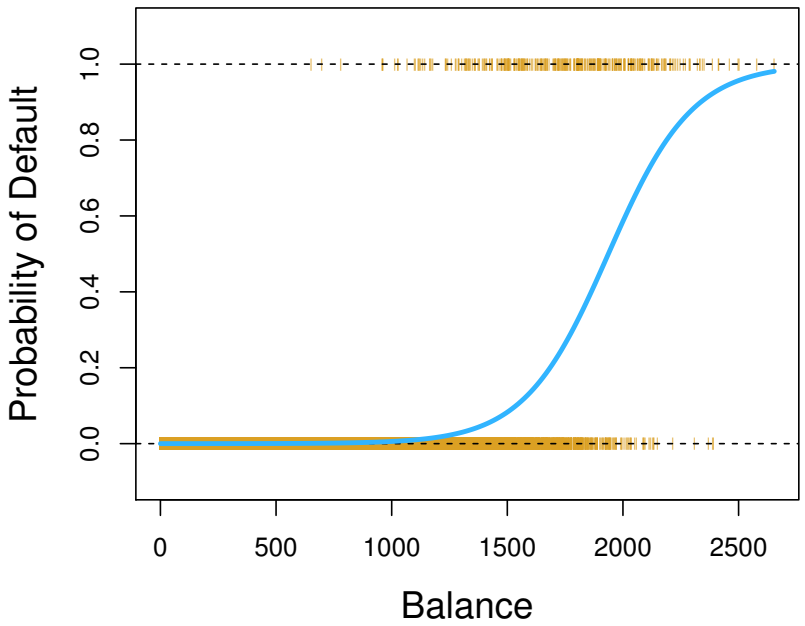
Binary Classification

– When we have a smooth approximation of the sign function, learning the parameters β_0 and β_1 is numerically easier



Binary Classification





Trying to treat the classification problem as a regression problem does not produce reasonable results!

How Does Binary Classification Work?

- We somehow learn β_0 and β_1 from the training data (will be explained soon)
- We are given a test point x_t, for which we evaluate $\beta_0 + \beta_1 x_t$
- We pass this quantity to our smooth sign approximation

$$p(x_t) = \frac{e^{\beta_0 + \beta_1 x_t}}{1 + e^{\beta_0 + \beta_1 x_t}}$$

- If $p(x_t)$ was closer to 1 our prediction of the class for x_t is class one (e.g., $p(x_t) = 0.7$) and if $p(x_t)$ was closer to 0 our prediction of the class for x_t is class zero (e.g., $p(x_t) = 0.3$)
- Now that $p(\cdot)$ generates some value between zero and one for us, one immediate interpretation for it is being the probability of label 1

$$p(x_t) = \mathbb{P}(y = 1 | x_t) = 1 - \mathbb{P}(y = 0 | x_t)$$

so if $p(x_t) = 0.7$, then the test label is 1 with probability 0.7, and 0 with probability 0.3

How to Do the Training for the Simple Logistic Regression?

- We observe samples $(x_1, y_1), \cdots (x_n, y_n)$, where $y_i \in \{0, 1\}$
- We want to determine β_0 and β_1 such that the probability of assigning the right labels is maximized

$$\arg\max_{\beta_0, \beta_1} \quad \mathbb{P}(Y_1 = y_1, \cdots Y_n = y_n | X_1 = x_1, \cdots, X_n = x_n, \beta_0, \beta_1)$$

- Basically, we want to find the ML estimates for β_0 and β_1

– Since our samples are independent, we get

$$\begin{aligned}\mathbb{P}(Y_1 = y_1, \cdots Y_n = y_n | x_1, \cdots, x_n, \beta_0, \beta_1) &= \prod_{i=1}^{n} \mathbb{P}(Y_i = y_i | x_i, \beta_0, \beta_1) \\ &= \prod_{i:y_i=1} p(x_i) \prod_{i:y_i=0} (1 - p(x_i)) \\ &= \prod_{i=1}^{n} p(x_i)^{y_i} (1 - p(x_i))^{1-y_i}\end{aligned}$$

where the first equality is thanks to

$$p(x_i) = \mathbb{P}(Y = 1 | x_i) = 1 - \mathbb{P}(y = 0 | x_i)$$

– So we ultimately want to find β_0 and β_1 that maximize

$$\prod_{i=1}^{n} p(x_i)^{y_i} (1-p(x_i))^{1-y_i} = \prod_{i=1}^{n} \left(\frac{e^{\beta_0+\beta_1 x_i}}{1 + e^{\beta_0+\beta_1 x_i}} \right)^{y_i} \left(1 - \frac{e^{\beta_0+\beta_1 x_i}}{1 + e^{\beta_0+\beta_1 x_i}} \right)^{1-y_i}$$

Some Notes on The Logistic Regression

- In logistic regression, we end up with a more complex cost function to optimize

$$\prod_{i=1}^{n} p(x_i)^{y_i}(1-p(x_i))^{1-y_i} = \prod_{i=1}^{n} \left(\frac{e^{\beta_0+\beta_1 x_i}}{1+e^{\beta_0+\beta_1 x_i}}\right)^{y_i} \left(1-\frac{e^{\beta_0+\beta_1 x_i}}{1+e^{\beta_0+\beta_1 x_i}}\right)^{1-y_i}$$

- Generally speaking, a closed-form solution for the maximizer is not available and often maximization techniques such as gradient ascent (or gradient descent on the negative log-likelihood) are considered

What Happens for More than One Feature?

- In case of multiple features, only minor modification is required
- We still try to maximize $\prod_{i=1}^{n} p(x_i)^{y_i}(1-p(x_i))^{1-y_i}$, but now we have

$$p(x_t) = \frac{e^{\beta_0+\beta_1 x_1+\beta_2 x_2+\cdots+\beta_p x_p}}{1+e^{\beta_0+\beta_1 x_1+\beta_2 x_2+\cdots+\beta_p x_p}}$$

- We run the maximization to estimate $\beta_0, \beta_1, \cdots, \beta_p$
- In practice you never have to do the maximization and most software such as R, Python and Matlab have packages to do that numerically

What Happens for More than Two Classes?

- Example, based on some features such as city, year of education and number of publications, classify the students of a class into undergrads, Masters, and PhDs
- Recall our method of classification in the binary case, we evaluated $p(x_t)$ which was technically $\mathbb{P}(Y = 1|x_t)$ and if it was closer to 1 then our class prediction was 1, if it was small, then $\mathbb{P}(Y = 0|x_t) = 1 - \mathbb{P}(Y = 1|x_t)$ would be large and our prediction is class zero
- One way of interpreting this is evaluating $\mathbb{P}(Y = k|x_t)$ for $k = 0, 1$ and the k that produces the largest value for $\mathbb{P}(Y = k|x_t)$ is our predicted label
- Now for K labels, we evaluate $\mathbb{P}(Y = k|x_t)$ for $k = 1, 2, \cdots, K$ and the k that produces the largest value for $\mathbb{P}(Y = k|x_t)$ is our predicted label

What Happens for More than Two Classes?

- For K labels, we evaluate $\mathbb{P}(Y = k|x_t)$ for $k = 1, 2, \cdots, K$ and the k that produces the largest value for $\mathbb{P}(Y = k|x_t)$ is our predicted label
- When we have $K > 2$ labels (e.g., $y \in$ {while, yellow, green}) and p features $x_1, x_2, \cdots, x_p$, we fit K models parametrized by

$$\begin{aligned} \text{Label 1:} \quad & \{\beta_0^{(1)}, \beta_1^{(1)}, \cdots, \beta_p^{(1)}\} \\ \text{Label 2:} \quad & \{\beta_0^{(2)}, \beta_1^{(2)}, \cdots, \beta_p^{(2)}\} \\ & \vdots \\ \text{Label K:} \quad & \{\beta_0^{(K)}, \beta_1^{(K)}, \cdots, \beta_p^{(K)}\} \end{aligned}$$

- For this problem we consider the following form:

$$p_k(\boldsymbol{x}) = \mathbb{P}(Y = k|\boldsymbol{x}) = \frac{e^{\beta_0^{(k)} + \cdots + \beta_p^{(k)} x_p}}{e^{\beta_0^{(1)} + \cdots + \beta_p^{(1)} x_p} + \ldots + e^{\beta_0^{(K)} + \cdots + \beta_p^{(K)} x_p}}$$

- What is the sum of all $\mathbb{P}(Y = k|\boldsymbol{x})$ for a fixed $\boldsymbol{x}$?

Some R Simulations

– Let's preform some basic classification tasks in R!

Linear and Quadratic Discriminant Analysis

Linear Discriminant Analysis (LDA)

– Probabilistically, suppose that our y can take K distinct values. By the Bayes' theorem we have

$$\mathbb{P}(Y = \ell | \boldsymbol{X} = \boldsymbol{x}) = \frac{\mathbb{P}(Y = \ell)\mathbb{P}(\boldsymbol{X} = \boldsymbol{x} | Y = \ell)}{\sum_{k=1}^{K} \mathbb{P}(Y = k)\mathbb{P}(\boldsymbol{X} = \boldsymbol{x} | Y = k)}$$
$$= \frac{\pi_\ell f_\ell(\boldsymbol{x})}{\sum_{k=1}^{K} \pi_k f_k(\boldsymbol{x})}$$

– Let's see why this equality holds, knowing the Bayes' equality

$$\mathbb{P}(A|B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)}$$

and $A = A_1 \cup A_2 \cdots \cup A_K$, where $A_i \cap A_j = \emptyset$.

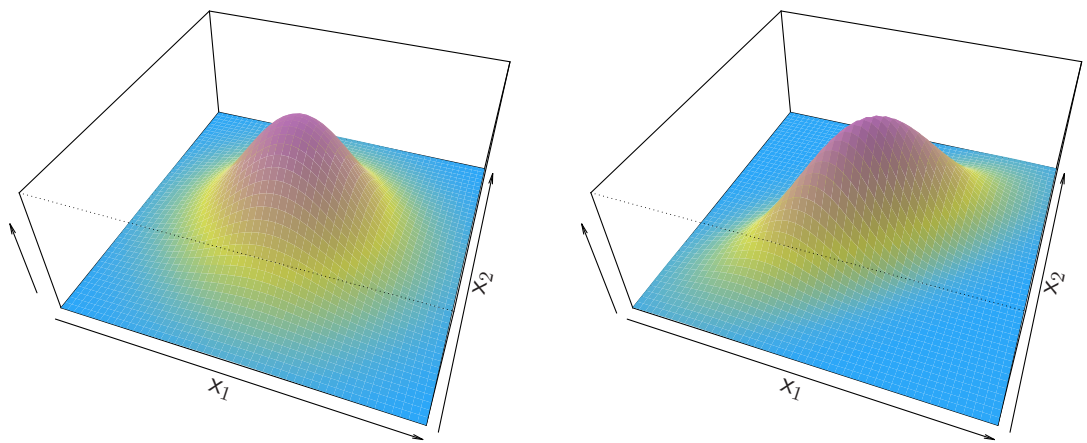
Little Introduction about Multivariate Normal

- Recall the normal distribution for a random variable x:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{(x-\mu)^2}{2\sigma^2})$$

- Similar to the scalar case, we can define a distribution for the random vector $\boldsymbol{x} = [x_1, \cdots, x_p]^T$ as

$$f(\boldsymbol{x}) = \frac{1}{(2\pi)^{p/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right)$$



Linear Discriminant Analysis (LDA)

- Recall

$$\mathbb{P}(Y = \ell | \boldsymbol{X} = \boldsymbol{x}) = \frac{\mathbb{P}(Y = \ell)\mathbb{P}(\boldsymbol{X} = \boldsymbol{x} | Y = \ell)}{\sum_{k=1}^{K} \mathbb{P}(Y = k)\mathbb{P}(\boldsymbol{X} = \boldsymbol{x} | Y = k)} = \frac{\pi_\ell f_\ell(\boldsymbol{x})}{\sum_{k=1}^{K} \pi_k f_k(\boldsymbol{x})}$$

- The purpose of LDA is learning a model for $\mathbb{P}(Y = \ell | \boldsymbol{X} = \boldsymbol{x})$
- In the formulation above, $f_\ell(\boldsymbol{x})$ is in a sense the distribution we consider for the data points in class ℓ, and π_ℓ is the probability that we pick some random sample and it belongs to class ℓ
- In LDA, we assume that all $f_\ell(\boldsymbol{x})$ have a multivariate normal distribution with similar covariances and different means, i.e.

$$f_\ell(\boldsymbol{x}) = \frac{1}{(2\pi)^{p/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_\ell)^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_\ell)\right)$$

- Unlike logistic regression, which involved a rather complicated maximization for learning, in LDA we have closed form expressions for π_ℓ, $\boldsymbol{\mu}_\ell$ and $\boldsymbol{\Sigma}$ and classifying new test points becomes very easy

Linear Discriminant Analysis (LDA)

– Given a training set $(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_n, y_n)$ where the responses y can take K distinct class values $1, 2, \cdots, K$, we can easily learn the LDA model by calculating π_ℓ, $\boldsymbol{\mu}_\ell$ and $\boldsymbol{\Sigma}$ via (considering c_ℓ to be the index of samples in class ℓ)

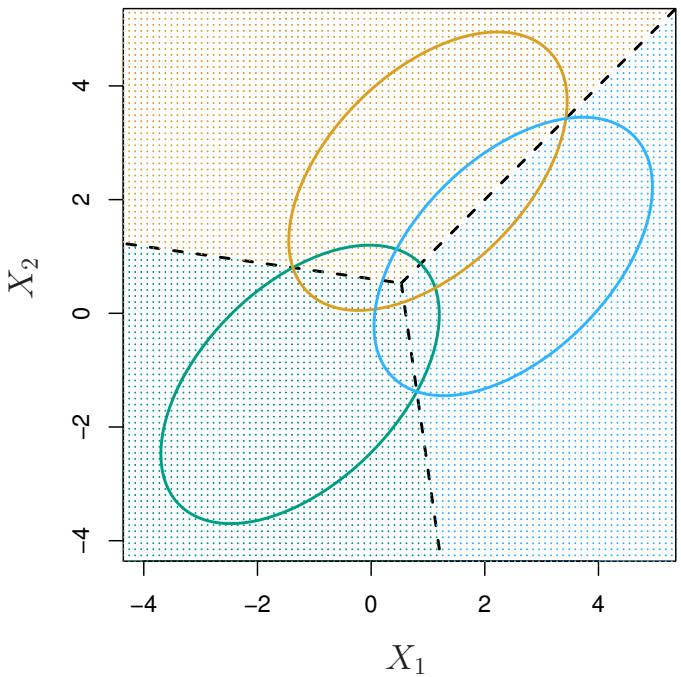
$$\hat{\pi}_\ell = \frac{\#\text{ of elements in } c_\ell}{n}$$

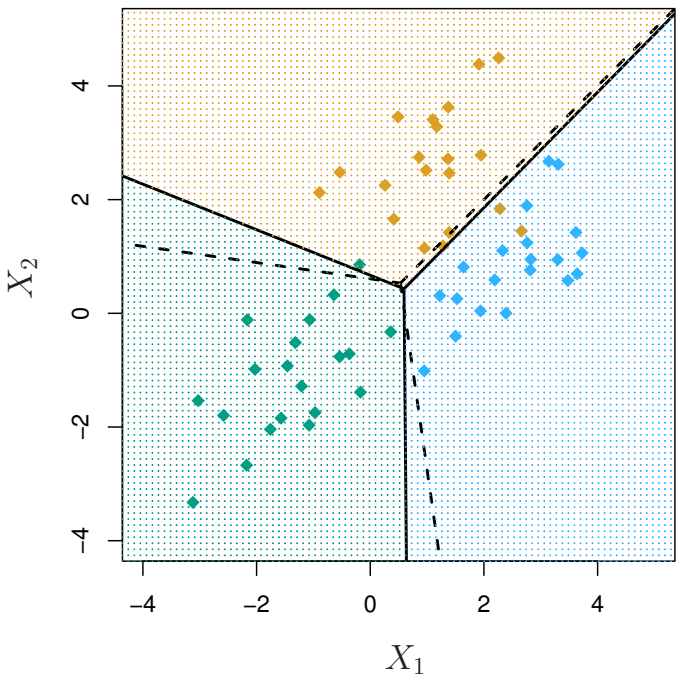
$$\hat{\boldsymbol{\mu}}_\ell = \frac{1}{\#\text{ of elements in } c_\ell} \sum_{i \in c_\ell} \boldsymbol{x}_i$$

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{N-K} \sum_{k=1}^{K} \sum_{i \in c_\ell} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_\ell)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_\ell)^\top$$

– After this point for a new test point $\boldsymbol{x}_t$ we have all that is needed to calculate $\mathbb{P}(Y = \ell | \boldsymbol{X} = \boldsymbol{x}_t)$ for $\ell = 1, \cdots, K$ and pick as the label the one that is largest

Linear Discriminant Analysis (LDA)





Linear Discriminant Analysis (LDA)

- In practice to assign a label to a given test point $\boldsymbol{x}_t$ we do not need to calculate

$$\mathbb{P}(Y = \ell | \boldsymbol{X} = \boldsymbol{x}_t) = \frac{\pi_\ell f_\ell(\boldsymbol{x}_t)}{\sum_{k=1}^{K} \pi_k f_k(\boldsymbol{x}_t)}$$

  and only comparing $\pi_\ell f_\ell(\boldsymbol{x}_t)$ is enough

- This reduces to evaluate

$$\delta_\ell = \boldsymbol{x}_t^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_\ell - \frac{1}{2} \boldsymbol{\mu}_\ell^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_\ell + \log \pi_\ell$$

  and pick as the class ℓ corresponding to the largest δ_ℓ

- You can find the decision boundary between class i and j by finding the points for which $\delta_i = \delta_j$

- [see the sample Matlab code]

Quadratic Discriminant Analysis (QDA)

- Recall

$$\mathbb{P}(Y = \ell | \boldsymbol{X} = \boldsymbol{x}_t) = \frac{\pi_\ell f_\ell(\boldsymbol{x}_t)}{\sum_{k=1}^{K} \pi_k f_k(\boldsymbol{x}_t)}$$

- The purpose of QDA is learning a model for $\mathbb{P}(Y = \ell | \boldsymbol{X} = \boldsymbol{x})$ in a more flexible way compared to LDA
- In QDA, we assume that all $f_\ell(\boldsymbol{x})$ have a multivariate normal distribution with similar covariances and different means, i.e.

$$f_\ell(\boldsymbol{x}) = \frac{1}{(2\pi)^{p/2} |\boldsymbol{\Sigma}_\ell|^{1/2}} \exp\left(-\frac{1}{2} (\boldsymbol{x} - \boldsymbol{\mu}_\ell)^\top \boldsymbol{\Sigma}_\ell^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_\ell) \right)$$

- The main difference between LDA and QDA is in LDA we consider a single $\boldsymbol{\Sigma}$ for all classes, but in QDA we allow more flexibility by having a different covariance matrix for each class
- Similar to LDA, QDA can be learned easily and we can obtain closed form expressions for π_ℓ, $\boldsymbol{\mu}_\ell$ and $\boldsymbol{\Sigma}_\ell$

Quadratic Discriminant Analysis (QDA)

– Given a training set $(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_n, y_n)$ where the responses y can take K distinct class values $1, 2, \cdots, K$, we can easily learn the QDA model by calculating π_ℓ, $\boldsymbol{\mu}_\ell$ and $\boldsymbol{\Sigma}_\ell$ via (considering c_ℓ to be the index of samples in class ℓ)

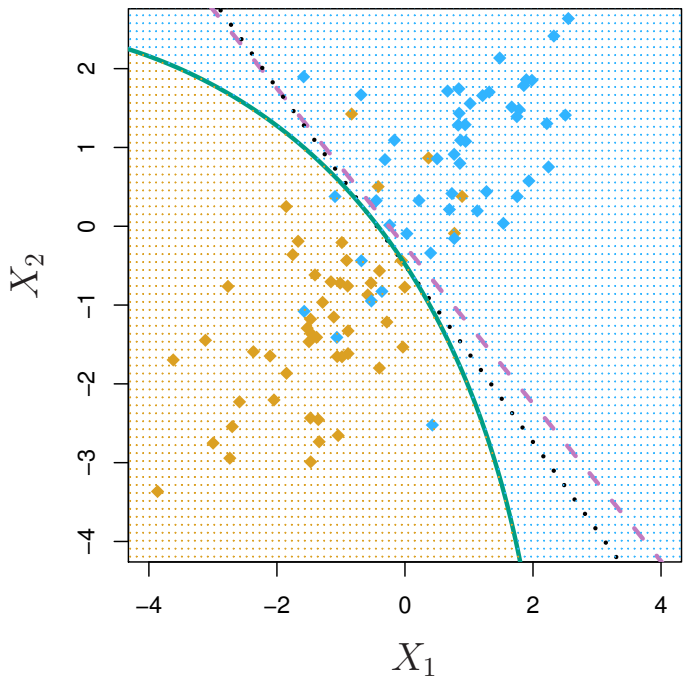
$$\hat{\pi}_\ell = \frac{\#\text{ of elements in } c_\ell}{n}$$

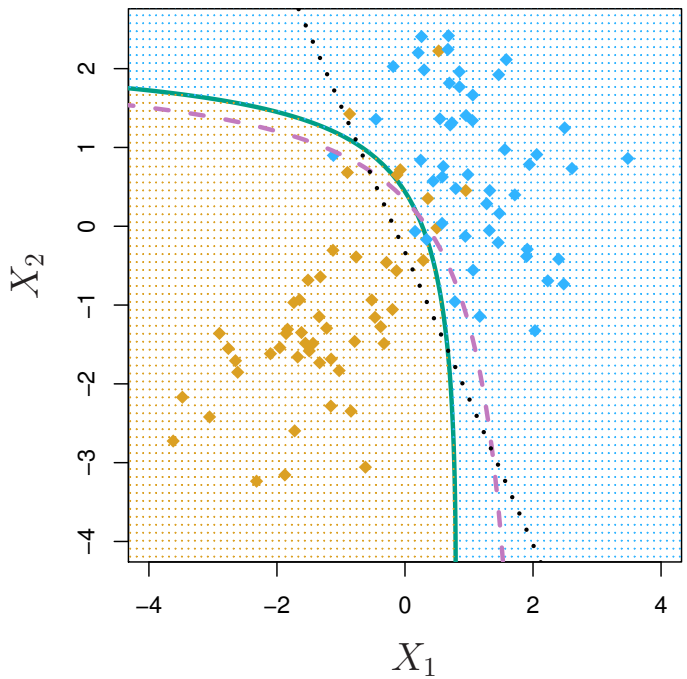
$$\hat{\boldsymbol{\mu}}_\ell = \frac{1}{\#\text{ of elements in } c_\ell} \sum_{i \in c_\ell} \boldsymbol{x}_i$$

$$\hat{\boldsymbol{\Sigma}}_\ell = \frac{1}{\#\text{ of elements in } c_\ell - 1} \sum_{i \in c_\ell} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_\ell)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_\ell)^\top$$

– After this point for a new test point $\boldsymbol{x}_t$ we have all that is needed to calculate $\mathbb{P}(Y = \ell | \boldsymbol{X} = \boldsymbol{x}_t)$ for $\ell = 1, \cdots, K$ and pick as the label the one that is largest

Quadratic Discriminant Analysis (QDA)





Quadratic Discriminant Analysis (QDA)

- In practice to assign a label to a given test point $\boldsymbol{x}_t$ we do not need to calculate

$$\mathbb{P}(Y = \ell | \boldsymbol{X} = \boldsymbol{x}_t) = \frac{\pi_\ell f_\ell(\boldsymbol{x}_t)}{\sum_{k=1}^{K} \pi_k f_k(\boldsymbol{x}_t)}$$

  and only comparing $\pi_\ell f_\ell(\boldsymbol{x}_t)$ is enough

- This reduces to evaluate

$$\delta_\ell = -\frac{1}{2} \log \det(\boldsymbol{\Sigma}_\ell) - \frac{1}{2} (\boldsymbol{x}_t - \boldsymbol{\mu}_\ell)^\top \boldsymbol{\Sigma}_\ell^{-1} (\boldsymbol{x}_t - \boldsymbol{\mu}_\ell) + \log \pi_\ell$$

  and pick as the class the ℓ corresponding to the largest δ_ℓ

- [see the sample Matlab code]

Some R Simulations

– Let's preform some basic classification tasks in R!

Summary

- Logistic regression is very popular for classification, especially for binary classification
- LDA is especially useful when $K > 2$, the number of training samples is small, or the classes are well separated, and Gaussian assumptions are reasonable.
- QDA presents more flexibility in shaping the partitions compared to LDA
- Logistic regression can also fit quadratic boundaries like QDA, by explicitly including quadratic terms in the model

Questions?

References

- https://www.alsharif.info/iom530, 2013.
- J. Friedman, T. Hastie, and R. Tibshirani.
  The elements of statistical learning.
  Springer series in statistics, 2nd edition, 2009.
- G. James, D. Witten, T. Hastie, and R. Tibshirani.
  https://lagunita.stanford.edu/c4x/HumanitiesScience/StatLearning/asset/classification.pdf, 2013.
- G. James, D. Witten, T. Hastie, and R. Tibshirani.
  An introduction to statistical learning: with applications in R, volume 112.
  Springer, 2013.