MY474: Applied Machine Learning for Social Science

Lecture 3: Linear Discriminant Analysis, Logistic Regression

Blake Miller

15 January 2021

Agenda

Decision-Theoretic Framework for Classification

Decision-Theoretic Framework for Classification

The model:

- Let the class label be a random variable y taking values in $\{1, ..., K\}$.
- For each data point i, $\quad i = 1, \ldots, n$, its label is drawn independently from a population of labels, with $P(y_i = k) = \pi_k$.
- The **class prior** $\pi_k \in (0, 1)$, $\quad \sum_{k=1}^{K} \pi_k = 1$ gives us the probability of observing each class k.
- For each object i, there is a vector of measurements $\mathbf{x_i}$ (the feature vector). $p(x)$ refers to our **probability density function**, our evidence (e.g. how frequently we observe measurements $\mathbf{x}$)
- If $y_i = k$, then $\mathbf{x_i}$ is sampled from the distribution $p_k(\mathbf{x}) = P(\mathbf{x}|y = k)$, known as the **likelihood**.
- Task: assign an object with feature vector $\mathbf{x}$ to one of the K predefined classes

Loss function

- To design a classifier, we use a **loss function** which calculates a penalty for incorrect classifications
- Let the loss function be $\mathcal{L}(\hat{y}, y)$ be the loss associated with assigning class $\hat{y}$ to the feature vector x when its true class is y
- Properties:

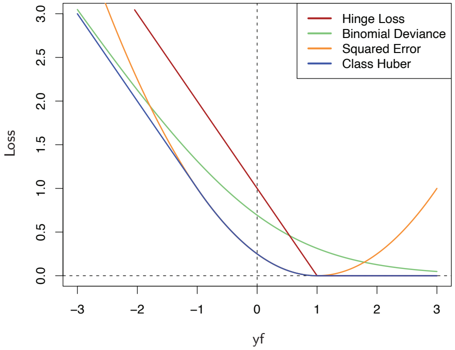
$$\begin{aligned} \mathcal{L}(\hat{y}, y) &= 0 \quad \mathit{if}\, \hat{y} = y \\ \mathcal{L}(\hat{y}, y) &> 0 \quad \mathit{if}\, \hat{y} \neq y \end{aligned}$$

Classification error loss

- For **classification error loss**, all errors are treated equally $\mathcal{L}(\hat{y}, y) = \mathbf{1}(\hat{y} \neq y)$: loss is 0 if the label is correct and 1 if it is incorrect
- In practice the loss is not symmetric:
  - **Because of the task at hand:** e.g. a false negative for a coronavirus test is more serious than a false positive. *Solution:* Choose a loss function that tolerates false positives more than false negatives.
  - **Because of class imbalance:** we want the classifier to identify all classes regardless of how many examples of each class we have. *Solution:* multiply $\mathcal{L}(\hat{y}, y)$ by weights that are inversely proportional to class frequencies π_k:

$$w_k = N/(|y| * n_k)$$

Loss Functions



Loss Function	Task	$\mathcal{L}(y, g(\mathbf{x}))$
ℓ_1 loss	Reg.	$\lvert y_i - g(\mathbf{x_i})\rvert$
ℓ_2 loss, quadratic loss	Reg.	$(y_i - g(\mathbf{x_i}))^2$
Classification loss, zero-one loss	Class.	$I(g(\mathbf{x_i}) \neq y_i)$
Cross-entropy, neg. log-likelihood	Class.	$log(1 + e^{-2yf(x)})$
Binomial deviance	Class.	$log(1 + e^{-yf(x)})$
Hinge loss	Class.	$\max(0, 1 - yg(\mathbf{x_i}))$

The Risk Function

- A classifier $c(\mathbf{x}) : \mathcal{X} \rightarrow \mathcal{Y} = \{1, ..., K\}$ is a decision rule (a function from the feature space $\mathcal{X}$ to the collection of class labels).
- The risk function for classifier c is the expected loss as a function of the unknown class y:

$$\mathcal{R}(c, k) = E[L(k, c(\mathbf{x}))|y = k] = \sum_{l=1}^{K} \mathcal{L}(k, l) P(c(\mathbf{x}) = l|y = k)$$

The Total Risk

The total risk is the total expected loss, with expectation taken over both Y and the vector X :

$$\mathcal{R}(c) = E[\mathcal{R}(c, y)] = \sum_{k=1}^{K} P(Y = k)\mathcal{R}(c, k)$$

For the classification error loss, this is the overall misclassification probability,

$$P(c(\mathbf{x}) \neq y) = \sum_{k=1}^{K} \pi_k P(c(\mathbf{x}) \neq k | y = k)$$

Bayes Rule

- Assume that the class densities $p_k(\mathbf{x})$ and the prior probabilities π_k are known in advance for all $k \in \{1, \dots, K\}$.
- Use the classification error loss $\mathcal{L}(k, l) = \mathbf{1}(k \neq l)$
- The **posterior probability** of class k given $\mathbf{x}$ is

$$
\begin{aligned}
p(k|\mathbf{x}) &= \frac{\pi_k p_k(\mathbf{x})}{\sum_{l=1}^{K} \pi_l p_l(\mathbf{x})} \\
&\propto \pi_k p_k(\mathbf{x})
\end{aligned}
$$

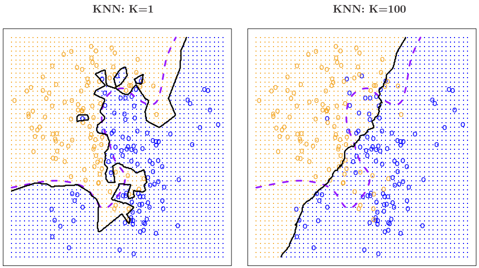
- The Bayes rule: assign observation x to the class with maximum posterior probability given $\mathbf{x}$:

$$c^*(\mathbf{x}) = k \text{ if } p(k|\mathbf{x}) = \max_{1 \leq l \leq K} p(l|\mathbf{x})$$

The Bayes Rule is Optimal

- The value of total risk $\mathcal{R}(c^*)$ for the Bayes rule is known as the Bayes risk
- No classifier can achieve a risk below the Bayes risk (see proof); thus the Bayes rule is a theoretical benchmark for all other procedures.
- In practice, the Bayes rule and the Bayes risk cannot be computed, because we don't know π_k's and $p_k(\mathbf{x})$

Performance Assessment in Practice



Bayes (purple) and KNN (black) decision boundary visualized with simulated data.

- The true risk cannot be calculated, but is useful theoretically and pedagogically (see figure above)
- In practice, we assess performance with:
  - **Apparent error rate**, or **training error**: error rate of our decision rule when applied to the training data (same data used to construct the classifier).
  - **True error rate**, or **test error**: the error rate of our decision rule when applied to new independent data (the test set).

Discriminant Analysis

Discriminant Analysis

Class densities are unknown but we believe/assume that the data come from a **parametric model**

$$p_k(\mathbf{x}) = p_k(\mathbf{x}; \boldsymbol{\theta_k}), \quad k = 1, \ldots, K$$

The form of p_k is assumed known, and the distribution is determined by the vector of unknown parameters $\theta_k \in \theta$. Plug-in classifier: plug in the estimated parameters into the Bayes rule:

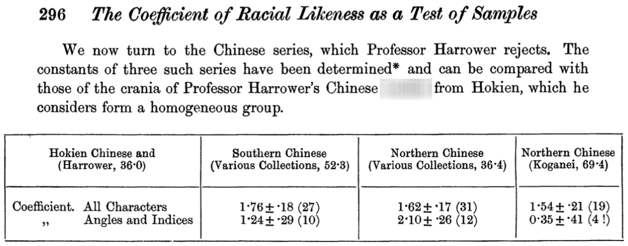
$$\hat{p}(k|\mathbf{x}) \propto \pi_k p_k(\mathbf{x}; \hat{\boldsymbol{\theta}}_{\boldsymbol{k}})$$

Assuming normal densities: $\theta_k = (\boldsymbol{\mu_k}, \boldsymbol{\Sigma_k})$,

$$\frac{1}{(2\pi|\boldsymbol{\Sigma_k}|)^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu_k})' \boldsymbol{\Sigma_k}^{-1} (\mathbf{x} - \boldsymbol{\mu_k})\right\}$$

leads to linear or quadratic discriminant analysis

The Racist History of Discriminant Analysis



296 *The Coefficient of Racial Likeness as a Test of Samples*

We now turn to the Chinese series, which Professor Harrower rejects. The constants of three such series have been determined* and can be compared with those of the crania of Professor Harrower's Chinese [redacted] from Hokien, which he considers form a homogeneous group.

		Hokien Chinese and (Harrower, 36·0)	Southern Chinese (Various Collections, 52·3)	Northern Chinese (Various Collections, 36·4)	Northern Chinese (Koganei, 69·4)
Coefficient.	All Characters		1·76 ± ·18 (27)	1·62 ± ·17 (31)	1·54 ± ·21 (19)
,,	Angles and Indices		1·24 ± ·29 (10)	2·10 ± ·26 (12)	0·35 ± ·41 (4 !)

In this Biometrika paper, Pearson used discriminant functions to determine if "a mixture of races existed in the district where [an anthropologist] was collecting [human skulls]." Note: I have blurred a racial slur.

- British statistician Karl Pearson first introduced an early discriminant method with his "coefficient of racial likeness"
- Ronald Fisher developed discriminant analysis as a criterion comparing between and within group variance, with applications to the fields of biology and eugenics in mind.
- In case you were wondering if people are still doing this... look no further than Stanford University.

Discriminant Analysis

- We will derive a parametric classifier for normal distributions:
  - Assume population distributions are normal
  - Estimate parameters
  - Plug into the Bayes rule
- This classifier is called **quadratic discriminant analysis**
- With an additional assumption of equal covariances, this classifier is called **linear discriminant analysis**

The Normal Populations Case: a Univariate Example

- Let $\mathbf{x}$ be univariate $p = 1$, with binary outcome $K = 2$
- $p_1(x) \sim N(\mu_1, \sigma_1^2), \quad p_2(x) \sim N(\mu_2, \sigma_2^2)$
- The normal density is

$$p_k(x) = \frac{1}{\sqrt{2\pi\sigma_k^2}} exp\left\{-\frac{1}{2\sigma_k^2}(x - \mu_k)^2\right\}$$

- Compute the posterior probability: for $k = 1, 2$,

$$P(Y = k|x) = \frac{\pi_k p_k(x)}{\pi_1 p_1(x) + \pi_2 p_2(x)} \propto \pi_k p_k(x)$$

- Assuming for simplicity $\pi_1 = \pi_2 = 0.5$, the Bayes rule says that x should be classified to the population with higher $p_k(x)$.

The Normal Populations Case: a Univariate Example

- Take logs:

$$p(Y=1|x) > p(Y=2|x) \Leftrightarrow log(p(Y=1|x)) > log(p(Y=2|x))$$

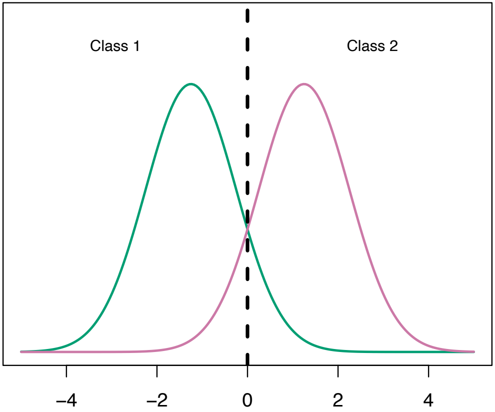
- The rule becomes: assign class 1 if

$$x^2\left(\frac{1}{\sigma_2^2}-\frac{1}{\sigma_1^2}\right)-2x\left(\frac{\mu_2}{\sigma_2^2}-\frac{\mu_1}{\sigma_1^2}\right)+\left(\frac{\mu_2^2}{\sigma_2^2}-\frac{\mu_1^2}{\sigma_1^2}\right)+2log\left(\frac{\sigma_2}{\sigma_1}\right)>0$$

  otherwise assign class 2.
- This is a quadratic function in x
- If we assume $\sigma_1 = \sigma_2$, after a bit of algebra, this reduces to:

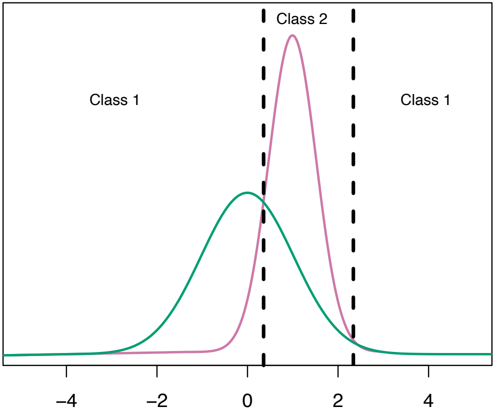
$$\frac{\mu_1^2-\mu_2^2}{2(\mu_1-\mu_2)}=\frac{\mu 1+\mu 2}{2}$$

Example (Equal Variances, Linear Rule)



- $\sigma_1 = \sigma_2 = 1, \mu_1 = -1.25, \mu_2 = 1.25.$
- Bayes decision boundary $x = \frac{\mu 1 + \mu 2}{2} = \frac{-1.25 + 1.25}{2} = 0$
- Class 1 if $x < 0$, otherwise class 2

Example (Unequal Variances, Quadratic Rule)



- $\sigma_1 = 1, \sigma_2 = 1/2, \mu_1 = 0, \mu2 = 1.$
- x has two solutions when plugging these values into the quadratic equation above.
- Bayes decision rule: class 1 if $x < .381$ or $x > 2.286$, otherwise class 2.

The Multivariate Normal Case

- Suppose that we have K multivariate normal populations $N(\boldsymbol{\mu_k}, \boldsymbol{\Sigma_k})$,

$$p_k(\mathbf{x}) = \frac{1}{(2\pi)^{p/2}|\boldsymbol{\Sigma_k}|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu_k})'\boldsymbol{\Sigma_k}^{-1}(\mathbf{x}-\boldsymbol{\mu_k})\right\}$$

- Picking class k with the largest posterior probability is equivalent to picking the class with the largest **discriminant function**:

$$\delta_k(x) = \frac{1}{2}log|\boldsymbol{\Sigma_k}| - \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu_k})'\boldsymbol{\Sigma_k}^{-1}(\mathbf{x}-\boldsymbol{\mu_k}) + log\pi_k$$

- In general, $\delta_k(\mathbf{x})$ is a quadratic function of $\mathbf{x}$, and the boundaries between classes are determined by quadratic curves

Linear Discriminant Analysis (LDA)

- Assume all $\boldsymbol{\Sigma_k} = \Sigma$ (equal covariances)
- The discriminant function simplifies to linear:

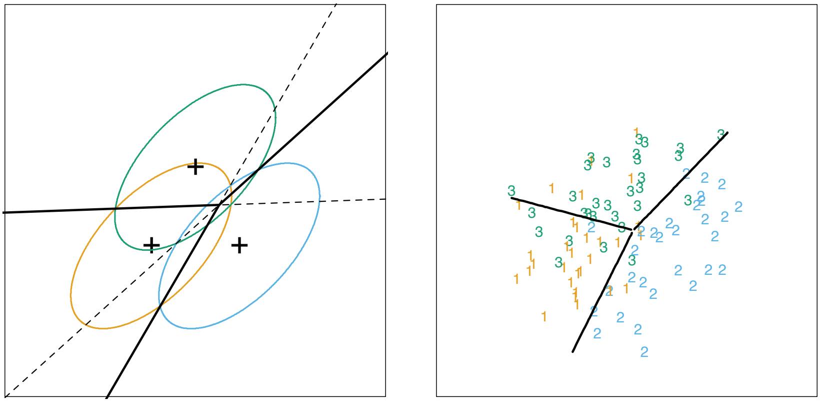
$$\delta_k(x) = x'\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_k} - 21\boldsymbol{\mu_k}'\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_k} + log \pi_k.$$

- To compare any two classes k and l, only need log-ratio of posterior probabilities:

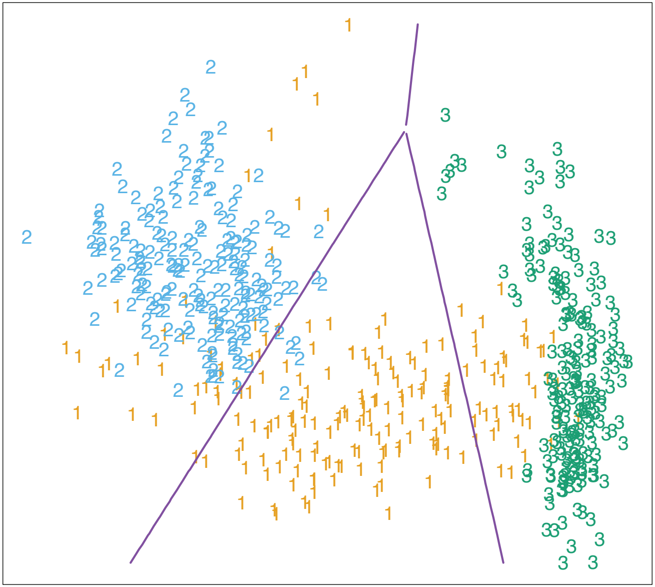
$$log \frac{p(k|x)}{p(l|x)} = log \frac{\pi_k}{\pi_l} - \frac{1}{2}(\boldsymbol{\mu_k} + \boldsymbol{\mu_l})'\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu_k} - \boldsymbol{\mu_l}) + x'\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu_k} - \boldsymbol{\mu_l}).$$

- Boundaries between classes are linear (hyperplanes in p dimensions)

LDA Illustration: 3 Classes, Equal Covariance



LDA Illustration: 3 Classes, Unequal Covariance



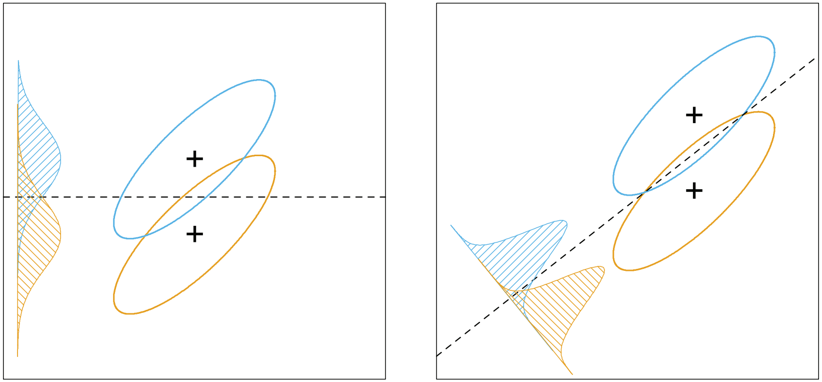
LDA Discriminant Directions

- The discriminant direction vector is $\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu_k} - \boldsymbol{\mu_l})$, generally not the same as the direction $\boldsymbol{\mu_k} - \boldsymbol{\mu_l}$
- If we further assume that all π_k are equal, the decision rule is equivalent to assigning class k with the smallest Mahalanobis distance from x to $\boldsymbol{\mu_k}$:

$$d_M(x, \boldsymbol{\mu_k})^2 = (x - \boldsymbol{\mu_k})'\boldsymbol{\Sigma}^{-1}(x - \boldsymbol{\mu_k})$$

- Note that using straight Euclidean distance $(x - \boldsymbol{\mu_k})'(x - \boldsymbol{\mu_k})$ is not the same (and generally much worse)

LDA Illustration: Discriminant Directions



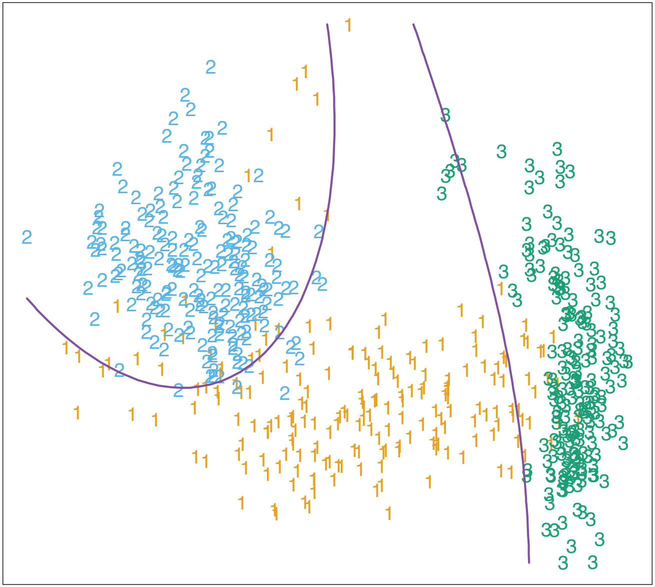
Quadratic Discriminant Analysis (QDA)

- The general case: normal populations, covariances $\boldsymbol{\Sigma_k}$ are **not assumed to be equal**.
- Then we have **quadratic discriminant functions**

$$\delta_k(x) = -\frac{1}{2} log|\boldsymbol{\Sigma_k}| - \frac{1}{2}(x - \boldsymbol{\mu_k})'\boldsymbol{\Sigma_k}^{-1}(x - \boldsymbol{\mu_k}) + log \pi_k.$$

- The boundary between each pair of classes k and l is described by a quadratic function

Quadratic Discriminant Analysis: Illustration



LDA and QDA in Practice

- In practice, we **estimate all parameters from training data**
- If priors are not assumed equal or known, they can be estimated by
- Class means are estimated by sample means,

$$\boldsymbol{\mu_k} = \bar{x}_k = \frac{1}{n_k - 1} \sum_{i:y_i=k} \mathbf{x_i}$$

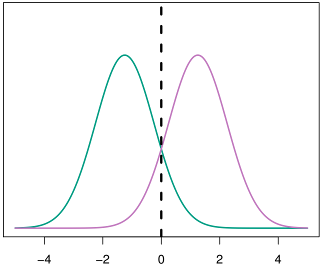
- QDA: each covariance matrix is estimated individually,

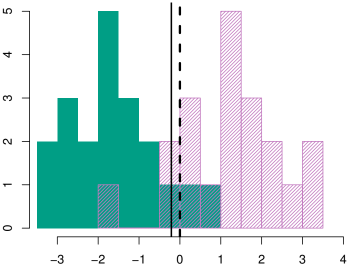
$$\hat{\boldsymbol{\Sigma}}_{\boldsymbol{k}} = \frac{1}{n_k - 1} \sum_{i:y_i=k} (\mathbf{x_i} - \hat{\mu}_k)(\mathbf{x_i} - \hat{\mu}_k)'$$

- LDA: estimate the common Σ by the **pooled covariance**

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{n - K} \sum_{k=1}^{K} \sum_{i:y_i=k} (\mathbf{x_i} - \hat{\mu}_k)(\mathbf{x_i} - \hat{\mu}_k)'$$

LDA and QDA in Practice: Illustration





Linear Regression

Review: Ordinary Least Squares (OLS) Model

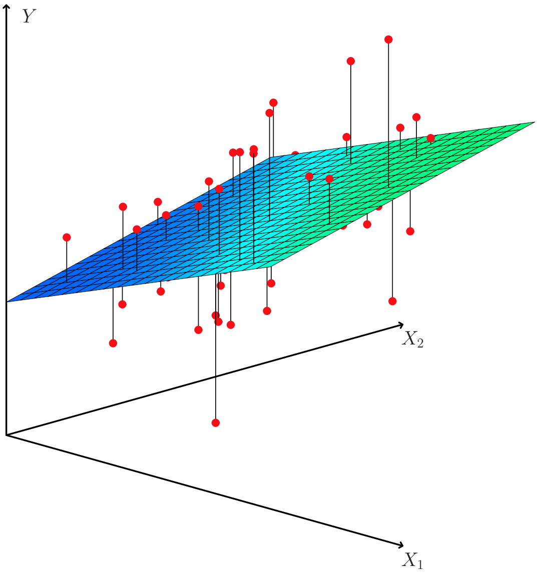
Assume the linear model

$$\mathbf{y} = \boldsymbol{X}\beta + \boldsymbol{\varepsilon}, \quad \boldsymbol{\varepsilon} \sim \mathcal{N}_n(0, \sigma^2 \mathbf{I}_n)$$

- $\mathbf{y}$ is the $n \times 1$ random response vector (observed)
- $\mathbf{X}$ is the fixed $n \times p$ matrix of predictor variables (observed data)
- β is the $p \times 1$ vector of fixed and unknown parameters (not observed)
- ε is the $n \times 1$ vector of error terms (not observed)

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} X_{11} & X_{12} & \cdots & X_{1p} \\ X_{21} & X_{22} & \cdots & X_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ X_{n1} & X_{n2} & \cdots & X_{np} \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_p \end{bmatrix}$$

Least Squares Estimation



Least Squares Estimation

- Find β to minimize **residual sum of squares**, the **loss function** most commonly used to for OLS:

$$\text{RSS} = \boldsymbol{\varepsilon}'\boldsymbol{\varepsilon} = (\mathbf{y} - \mathbf{X}\beta)'(\mathbf{y} - \mathbf{X}\beta)$$

- Leads to the normal equations

$$(\mathbf{X}'\mathbf{X})\hat{\beta} = \mathbf{X}'\mathbf{y}$$

- Assuming $\mathbf{X}$ has rank p, the unique solution is

$$\hat{\beta} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$$

Interpreting regression coefficients

- The ideal scenario is when the predictors are uncorrelated - a **balanced design**
  - Each coefficient can be estimated and tested separately.
  - Interpretations such as “a unit change in X_j is associated with a β_j change in Y, while all the other variables stay fixed,” are possible.
- Correlations amongst predictors cause problems:
  - The variance of all coefficients tends to increase, sometimes dramatically
  - Interpretations become hazardous — when X_j changes, everything else changes.
  - Recall the advertising data from the book. What if there is a fixed advertising budget? If more is spend on radio, less can be spent on TV, newspaper
- **Claims of causality** should be avoided for observational data.

Linear Probability Model

Suppose for the censorship classification task, we code

$$y = \begin{cases} 1 & \text{if censored} \\ 0 & \text{if } \neg\text{censored} \end{cases}$$

- Can we simply perform a linear regression of y on x and classify as Yes if $\hat{y} > 0.5$?
  - In this case of a binary outcome, linear regression does a good job as a classifier, and is equivalent to **linear discriminant analysis** which we discuss later.
  - Since in the population $E(Y|X = x) = Pr(Y = 1|X = x)$, we might think that regression is perfect for this task.
  - However, **linear regression** might produce probabilities less than zero or bigger than one. **Logistic regression** is more appropriate.

Linear probability model

$$\mathbf{y} = \boldsymbol{X}\beta + \boldsymbol{\varepsilon}, \quad \boldsymbol{\varepsilon} \sim \mathcal{N}_n(0, \sigma^2 \mathbf{I}_n)$$

$$y_i = \begin{cases} 1 & \text{if censored} \\ 0 & \text{if } \neg\text{censored} \end{cases}$$

- Since in this case, y is a discrete random variable, we can express $E[y]$ as a weighted sum:

$$\begin{aligned} E[y] &= 1 \times P(y = 1) + 0 \times P(y = 0) \\ &= P(y = 1) \end{aligned}$$

- It then follows that, $\mathbf{X}\beta = P(y = 1)$
- β_i represents the change in $P(X)$ from a one unit increase in X, holding all else constant.

Problems with the linear probability model

- Our output space is not bound between 0 and 1, a problem when interpreting $\hat{y}$ as probabilities.

$$-\infty < \beta_0 + \beta_1 \times x_1 + \cdots + \beta_p \times x_p < \infty$$

- Assuming $E(\varepsilon) = 0$, the model has non-constant errors:

$$\varepsilon = y - \mathbf{X}\beta = y - p \qquad \varepsilon_i = \begin{cases} 1 - p_i & \text{if } y_i = 0 \\ -p_i & \text{if } y_i = 1 \end{cases}$$

$$\begin{aligned} \sigma_i^2 = E[\varepsilon_i^2] &= (1 - p_i)^2 \times p_i + (-p_i)^2 \times (1 - p_i) \\ &= p_i(1 - p_i) \end{aligned}$$

Because the error term depends on $\mathbf{x_i}$, the model is heteroskedastic and should not be used (without correcting standard errors) for inference.

Logistic Regression

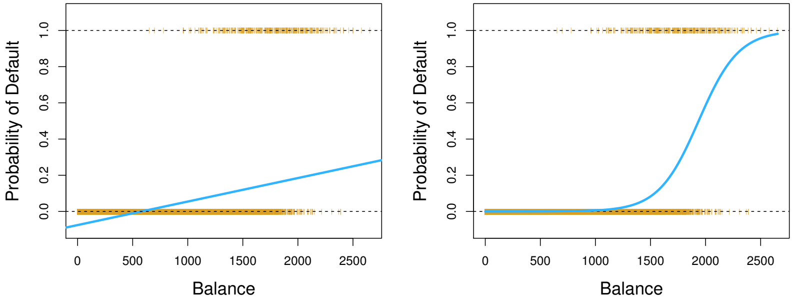
The Logistic Link Function

- To avoid nonsensical probabilities, we need a **link function** $\theta(\cdot)$ connecting value of the outcome to the linear model.
- For logit, we need to take the linear model oucome and smoothly restricts it to $[0, 1]$
- $\theta^{-1}(\cdot)$ should be a function where $F(-\infty)$ approaches 0 and $F(\infty)$ approaches 1
- One such function is the **logistic link function** (also known as the sigmoid function) which maps our linear model output to the mean of a **Bernoulli distribution** ($y \in \{0, 1\}$)

$$h(\mathbf{x}) = \theta^{-1}(\beta'\mathbf{x}); \qquad \theta^{-1}(s) = \frac{e^s}{1 + e^s}$$

- $e \approx 2.71828$ is a mathematical constant called Euler's number.

Linear versus Logistic Regression



The orange marks indicate the response Y, either 0 or 1. Linear regression does not estimate $Pr(Y = 1|X)$ well. Logistic regression seems better suited to the task.

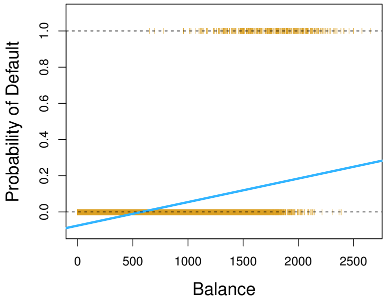
Logistic Regression

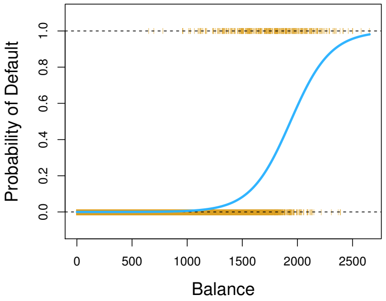
Let's write $P(X) = Pr(Y = 1|X)$ for short and consider using `balance` to predict `default`. Logistic regression uses the form

$$P(X) = \frac{e^{\boldsymbol{\beta}'\mathbf{x}}}{1 + e^{\boldsymbol{\beta}'\mathbf{x}}}$$

- No matter what values $\beta_1, \ldots, \beta_p$, or X take, $P(X)$ will have values between 0 and 1. Why?
- This monotone transformation is called the log odds, sigmoid, or logit transformation of $P(X)$.
- Other options are possible besides logit (e.g. normal CDF/probit, complementary log-log transformations)

Linear versus Logistic Regression





Logistic regression ensures that our estimate for p(X) lies between 0 and 1.

Odds ratios in Logistic Regression

- Imagine an event with $p = 0.8$ probability of success. The odds ratio is $\frac{p}{1-p} = 4$, a $4:1$ odds of success.
- For logistic regression, we define the odds as follows:

$$p = \frac{e^{\boldsymbol{\beta}'\mathbf{x}}}{1 + e^{\boldsymbol{\beta}'\mathbf{x}}}$$

$$\begin{aligned} 1 - p &= 1 - \frac{e^{\boldsymbol{\beta}'\mathbf{x}}}{1 + e^{\boldsymbol{\beta}'\mathbf{x}}} \\ &= \frac{1 + e^{\boldsymbol{\beta}'\mathbf{x}}}{1 + e^{\boldsymbol{\beta}'\mathbf{x}}} - \frac{e^{\boldsymbol{\beta}'\mathbf{x}}}{1 + e^{\boldsymbol{\beta}'\mathbf{x}}} \\ &= \frac{1}{1 + e^{\boldsymbol{\beta}'\mathbf{x}}} \end{aligned}$$

It then follows that the odds ratio is:

$$\frac{p}{1-p} = e^{\boldsymbol{\beta}'\mathbf{x}}$$

Log odds

- We take the log of the previous expression to get the linear combination of the predictors and the parameters $\boldsymbol{\beta}$:

$$log\left(\frac{p}{1-p}\right) = \boldsymbol{\beta}'\mathbf{x}$$

- Interpreting logit coefficients is more tricky
- Increasing x_i by one unit changes the log odds by β_i or multiplies the odds by e^{β_i}, holding all other parameters constant.
- Because of non-linear relationship between x_i and $P(x_i)$, we cannot interpret β_i as a a change in $P(x_i)$ associated with a one unit increase in x_i

Logistic Regression vs. LDA

- Both models produce a linear decision boundary between classes.
- The logistic classifier specifies $p(y|\mathbf{x})$ but does not make distributional assumptions about $p(\mathbf{x})$.
- LDA models the full joint distribution $p(y, \mathbf{x})$
- LDA assumes $p(\mathbf{x})$ is a mixture of normal distributions:

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k p_k(\mathbf{x}); \quad p_k \sim N(\boldsymbol{\mu_k}, \boldsymbol{\Sigma})$$

- If LDA's assumptions about $p(\mathbf{x})$ are not reasonable, logistic regression may perform better.

Logistic regression with more than two classes

So far we have discussed logistic regression with two classes. It is easily generalized to more than two classes. One version (used in the `R` package `glmnet`) has the symmetric form

$$P(y = k|\mathbf{x}) = \frac{e^{\boldsymbol{\beta_k}'\mathbf{x}}}{\sum_{\ell=1}^{K} e^{\boldsymbol{\beta_\ell}'\mathbf{x}}}$$

- A linear function for each class, only K - 1 linear functions are needed.
- Multiclass logistic regression is also referred to as multinomial regression as it models the mean of a multinomial random variable.
- This multinomial transformation is often called **the softmax** in computer science.

Maximum Likelihood Estimation (MLE)

We use maximum likelihood to estimate the parameters.

$$L(\beta; y_i, \mathbf{x_i}) = \prod_{i:y_i=1} P(\mathbf{x_i}) \prod_{i:y_i=0} (1 - P(\mathbf{x_i}))$$

$$P(\mathbf{x_i}) = \theta^{-1}(\mathbf{x_i}'\beta) = \frac{e^{\beta'\mathbf{x}}}{1 + e^{\beta'\mathbf{x}}} = \frac{1}{1 + e^{-\beta'\mathbf{x}}}$$

- Joint probability of the observed zeros and ones in the data.
- Goal is to choose β to make the likelihood of the observed data as high as possible (because we **DID** observe these data!)

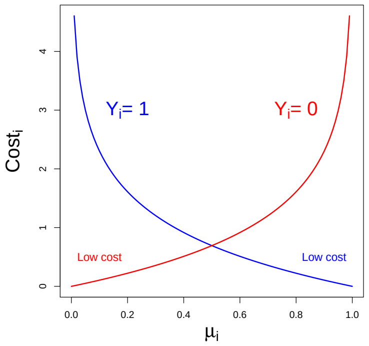
Maximum Likelihood Estimation of β

- To estimate β, we usually maximize the log-likelihood.
- *ln*(.) is a monotonically increasing function, so estimates of β will be the same.
- We maximize the log likelihood for computational reasons:
  1. **Computational complexity**: Summing is less computationally expensive than multiplication
  2. **Numerical stability**: Likelihoods become very small, affecting floating point precision
- Conditional log-likelihood of y given $\mathbf{x}$ after some algebra:

$$\ell(\beta; y_i, \mathbf{x_i}) = -\sum_{i=1} y_i \log P(\mathbf{x_i}) + (1 - y_i) \log(1 - P(\mathbf{x_i}))$$

- In a machine learning context, as we learn by minimizing a **loss/cost function**, we minimize the **negative log-likelihood**, which is equivalent to maximizing the **log-likelihood**

Logit Loss Function, Visualized



Each observation contributes to the loss/cost (Note: $0 < \mu_i < 1$)

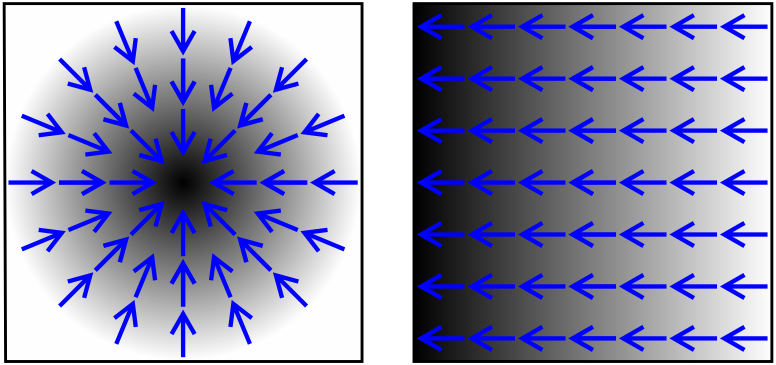
$$\ell(\boldsymbol{\beta}; y_i, \mathbf{x_i}) = \begin{cases} -\log(\mu_i) & \text{if } y_i = 1 \\ -\log(1-\mu_i) & \text{if } y_i = 0 \end{cases}$$

Optimization Algorithms for Parameter Estimation

- There are many algorithms for maximizing the log-likelihood (or minimizing the negative log-likelihood):
- The most popular methods of fitting logistic regression models Newton Raphson and iteratively reweighted least squares (IWLS). Read more on how these work in ESL 4.4.1.
- In `R` the `glm` function uses IWLS.
- The most common optimization algorithms used in machine learning are **gradient descent** and **stochastic gradient descent**

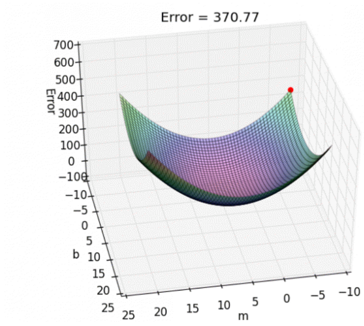
Gradient Descent

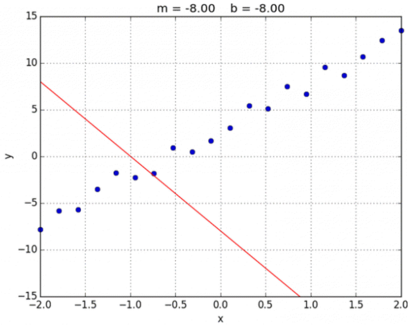
Gradient Descent



The gradient, represented by the blue arrows, denote the direction of greatest change of a scalar function. The values of the function are represented in greyscale and increase in value from white (low) to dark (high). Source: Wikimedia Commons

Gradient Descent visualized with Simple Linear Regression





Source: Alykhan Tejani

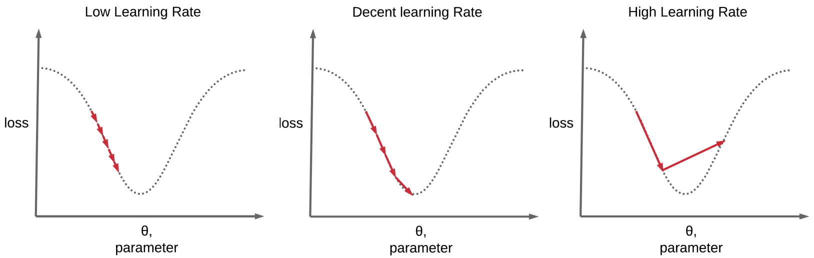
Gradient Descent

Our goal is to estimate β by minimizing the loss function $-ln(\ell(\beta; y_i, \mathbf{x_i}))$:

$$\hat{\beta} = \arg\min_{\beta} - ln(\ell(\beta))$$

- A **loss function** measures how far away predictions $\hat{y}$ are from the true values y.
- The **gradient** $\nabla f(\cdot)$ is the vector of parital derivates where the direction represents the greatest rate of increase of the function.
- For a convex function like a loss function, moving in the negative direction of the gradient will take you toward the minimum of that function.
- In other words, the gradient of the loss gives the direction of the prediction error and we can move in the opposite direction to minimize that error.
- How much we move is a parameter λ called the learning rate

Learning Rate



- λ too small: will approach the minimum very slowly
- λ too big: will jump around the minimum and converge very slowly (if at all)
- We want to choose a λ that minimizes the time to convergence and gives us the right answer.

(Batch) Gradient Descent

Algorithm 1: (Batch) Gradient Descent

```
1 Set tolerance parameter $\epsilon$ to positive real value;
2 Set learning rate parameter $\lambda$ to positive real value;
3 Initialize $\hat{\boldsymbol{\beta}}$ to a random non-zero vector;
4 while $\|\nabla\mathcal{L}(\hat{\mathbf{y}}, \mathbf{y})\| > \epsilon$ do
5     $\hat{\boldsymbol{\beta}} \leftarrow \hat{\boldsymbol{\beta}} - \lambda\nabla\mathcal{L}(\hat{\mathbf{y}}, \mathbf{y})$;
6 end
```

Stochastic Gradient Descent

- With large training data, gradient descent can become very computationally expensive
- Must sum over all examples of the training data at each step in the iteration

$$ln(\ell(\beta; y_i, \mathbf{x_i})) = \sum_{i=1}^{n} \left[y_i \mathbf{x_i}'\beta - log(1 + e^{\mathbf{x_i}'\beta}) \right]$$

- Instead of considering the full batch gradient (with all training data), we compute a stochastic version of the gradient with a single randomly-selected training observation.
- Very efficient in practice, cuts down training time significantly while performing similarly (and sometimes better) than batch gradient descent.

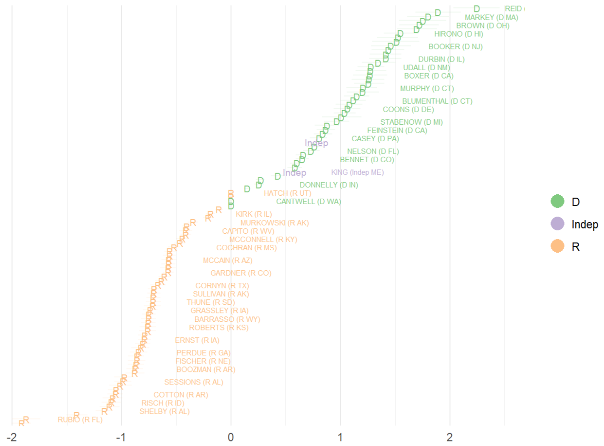
Stochastic Gradient Descent

Algorithm 2: Stochastic Gradient Descent

```
1  Set learning rate parameter $\lambda$ to positive real value;
2  Initialize $\hat{\beta}$ to a random non-zero vector;
3  for $e \leftarrow 0$ to $E$ do
4      Shuffle $\mathcal{D}$ to prevent cycles;
5      foreach $(x_i, y_i) \in \mathcal{D}$ do
6          Compute $\hat{y} \leftarrow h(x_i)$;
7          Compute the loss $\mathcal{L}_i(\hat{y}_i, y_i)$;
8          Compute the gradient of the loss $\nabla \mathcal{L}_i(\hat{y}_i, y_i)$;
9          Update parameters $\hat{\beta} \leftarrow \hat{\beta} - \lambda \nabla \mathcal{L}_i(\hat{y}_i, y_i)$;
10     end
11 end
```

Social Science Applications

Applications: Item Response Theory (IRT)



An extension of logistic regression can allow us to estimate ideal points for members of congress on certain bills. (Source: idealstan package)

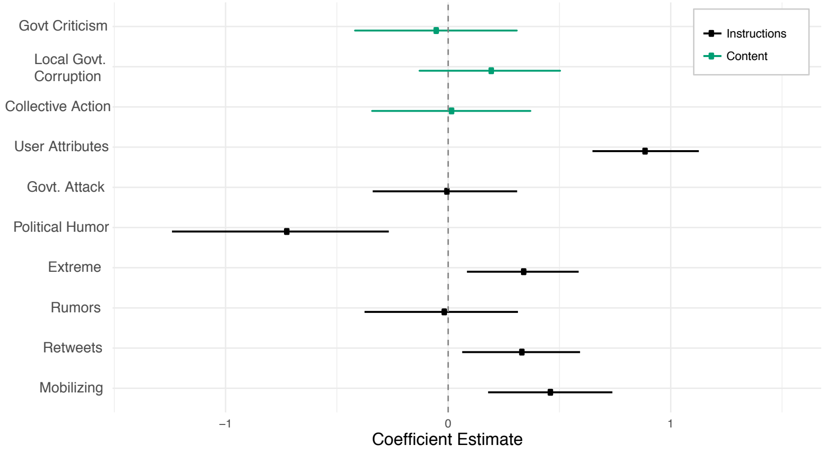
Example: Leaked Censorship Logs

Internet Management Office demands we eliminate all content about the Lufeng incident. Because there is [official] news pubished on this incident, we asked [MANAGER] to seek clarification from the Internet Management Office. [MANAGER] says to only deal with negative content. The current decision is to secret content claiming that someone was beaten to death or inflammatory content; report big users. Content similar to official media should be allowed.

2011-12-11

□ Directive □ Content □ Instructions □ Directive source □ Report user

Example: Leaked Censorship Logs



What kind of content and users are more likely to be reported back to the government?