Discriminative Classification

Classification 2: Logistic Regression and Practical Aspects of Classification

Siva Balakrishnan
Data Mining: 36-462/36-662

January 22, 2018

ISL 4.3 (Logistic Regression), K&J Chapter 11 (Practical Aspects)

Recap: Setup

$x_i \in \mathbb{R}^p$

$y_i \in \mathbb{R}$ - regression

$y_i \in \{0, \dots, K-1\}$ → classification

- Supervised Learning $\{(x_1, y_1), \dots, (x_n, y_n)\} \sim \mathcal{P}_{xy}$.
- Hypothetical – if we knew $\mathcal{P}_{xy}$. What is the best way to predict y from x?

Regression	Classification
→ use sq. loss	→ use 0/1 loss
→ use conditional expectation	→ pick most likely label:
$f(x) = \mathbb{E}[y \mid X = x]$	$f(x) = \underset{j \in \{0, \dots, K-1\}}{\arg\max} \mathbb{P}(Y = j \mid X = x)$ ↳ Bayes classifier

- How good are these predictors?

Regression	Classification
Unpredictable error	Bayes error / Bayes risk
$\mathbb{E}\left[(y - f(x))^2\right]$	↳ Binary case $= \mathbb{E}\left[\min\{f(x), 1 - f(x)\}\right]$

Recap: Loss Function in Classification

- Usually we use 0/1 loss. Most classification problems are not naturally symmetric.
- Most generally, can specify a $(K \times K)$ matrix of losses and calculate the Bayes classifier.

Confusion Matrix: True Label

Guess of Label

$$
\begin{array}{c|cccc}
 & 0 & \cdots & & K-1 \\
\hline
0 & L_{00} & L_{01} & \cdots & \\
\vdots & & L_{11} & & \\
 & & & \ddots & \\
K-1 & & & & L_{(K-1)(K-1)}
\end{array}
$$

can be 1 / can be asymmetric

0

- Important practical knob to be aware of, and to think carefully about.

Recap: Classification v/s Regression

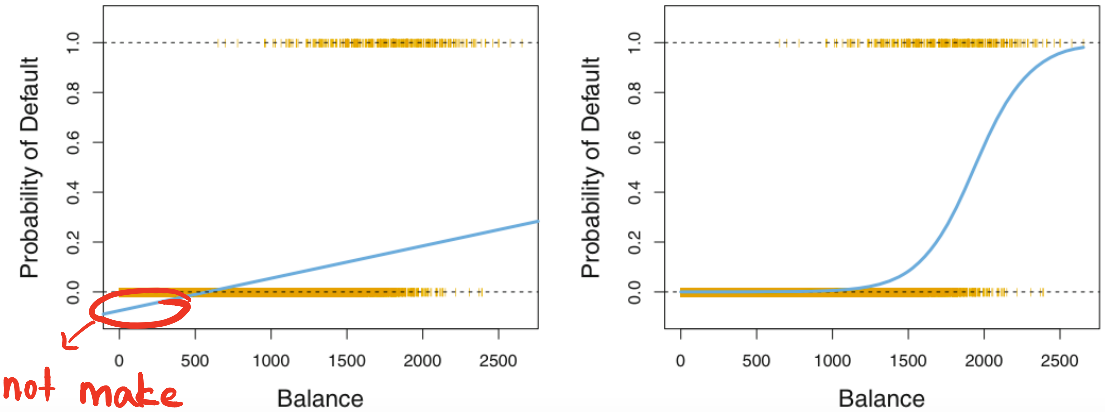
- Binary Classification is closely related to regression. If we encode, $y \in \{0, 1\}$ then:

$$\mathbb{E}[y|X = x] = \mathbb{P}(y = 1|X = x).$$

  So to classify well in the binary case, we need to know whether the regression function is above $1/2$ or below $1/2$.

- Using *squared loss* and fitting a linear model (for instance) is still a bad idea.

Recreate in HW1



does not make sense

- The relationship between classification and regression completely breaks down in the multi-class setting.

Recap: Generative v/s Discriminative

- In the binary case, the Bayes classifier is:

$$f_{\text{Bayes}}(x) = \mathbb{I}(\underbrace{\mathbb{P}(y=1|X=x)} \geq 1/2).$$

- Suggests two different approaches to classification:

Discriminative	Generative
1. Model $\mathbb{P}(y=1 \mid X=x)$	1. $\dfrac{\mathbb{P}(X=x \mid y=1) \times \mathbb{P}(y=1)}{\mathbb{P}(X=x)}$ compare to $\dfrac{\mathbb{P}(X=x \mid y=0)\mathbb{P}(y=0)}{\mathbb{P}(X=x)}$
2. "Assumption - light", (ie) not positing model for $X \mid y$.	2. Generate samples since we have modelled x.
	3. In some cases, more natural to model how data came about

Discriminative Classifiers

How should we think about modeling $\mathbb{P}(Y = k|X = x)$ directly?

If we only had a few x values, we could directly look at Y conditional on each one:

	Default=No	Default=Yes
Student=No	6850	206
Student=Yes	2817	127

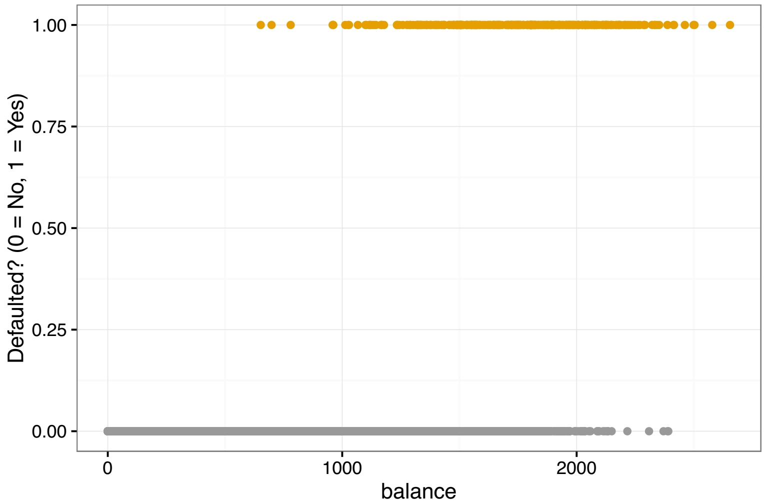
$$P(\text{Default}|\text{Student}) = \frac{127}{2817+127}$$

$$P(\text{Default}|\text{Not Student}) = \frac{206}{206+6850}$$

what is best classifier?

→ Always predict "no default"

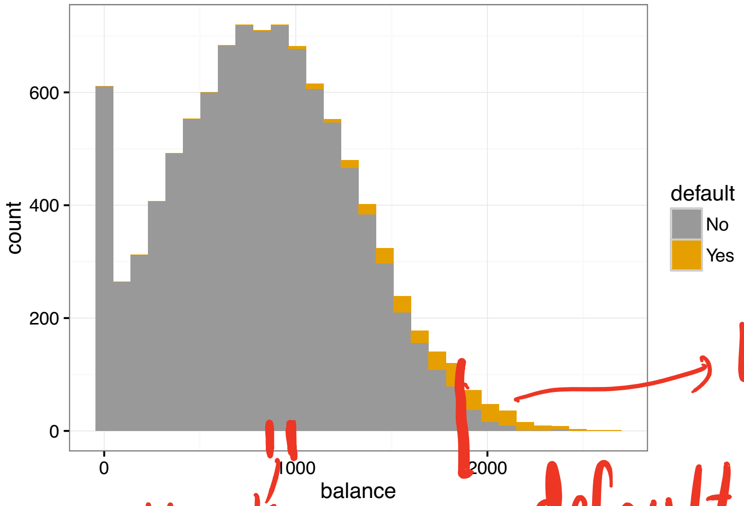
Discriminative Classifiers



If we have many values of x, we can't directly look at $P(Y = k|X = x)$ for each x. We need some way to pool information from *similar* points.

This should remind you of regression.

How should we model $P(Y = 1|X = x)$ for a continuous X?

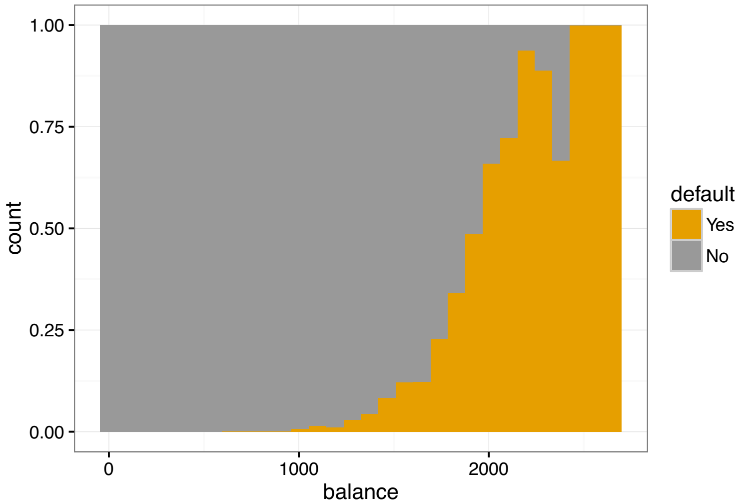


here default is more likely.

balance b/w 900-1000 < default / no default

We can start to see that most individuals do not default, and that large balance seems to be related to default.

This is a conditional density plot. We look at the *probability of default* within each bin.

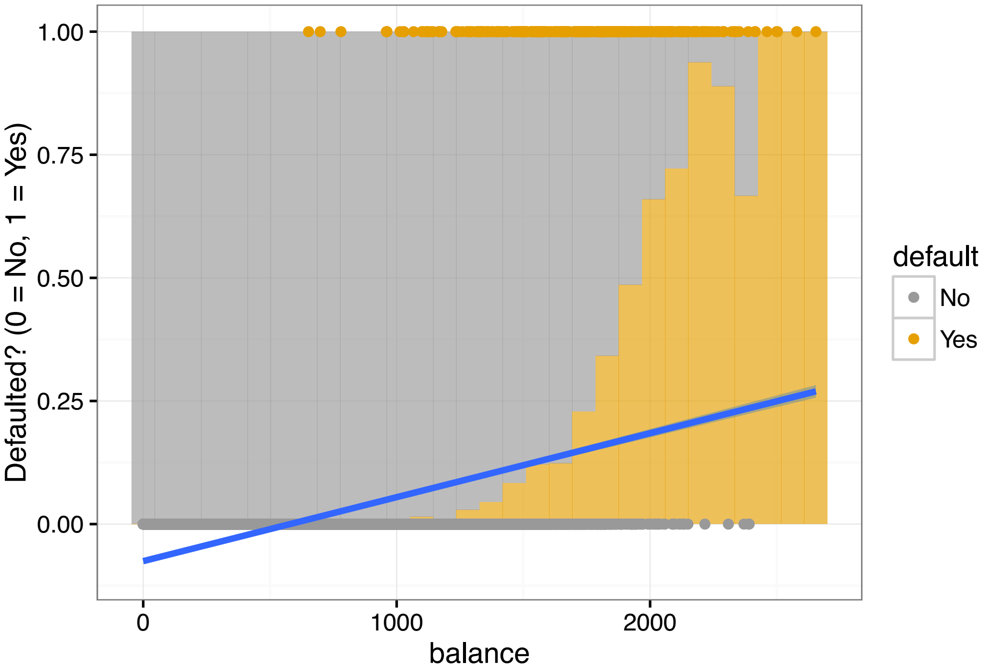


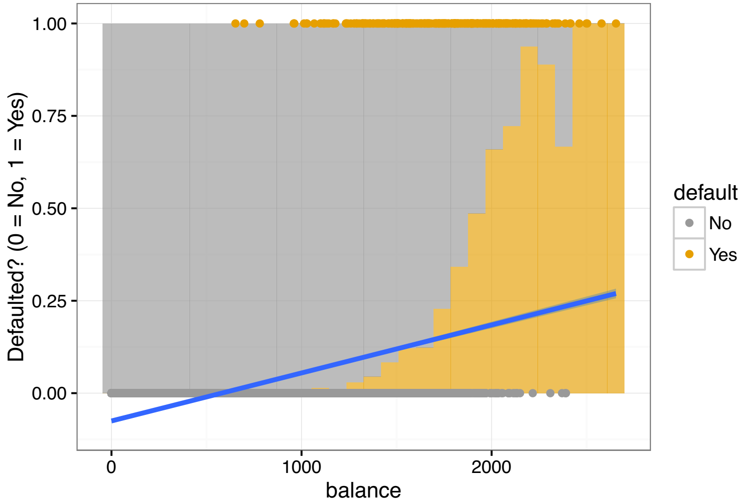
This is a (binned) plot of $P(Y = 1|X = x)$! This is clearly a natural way to think about classification. If I know which bin you are in (X), I can look at how likely you are to default.

How should I build a model of this?

We could try a linear model:

$$Y = \beta_0 + \beta_1 x$$





Are you happy?

- Predictions outside $[0, 1]$ don't really make sense.
- Interpretations of β are strange.
- Least squares doesn't really make sense for estimation.

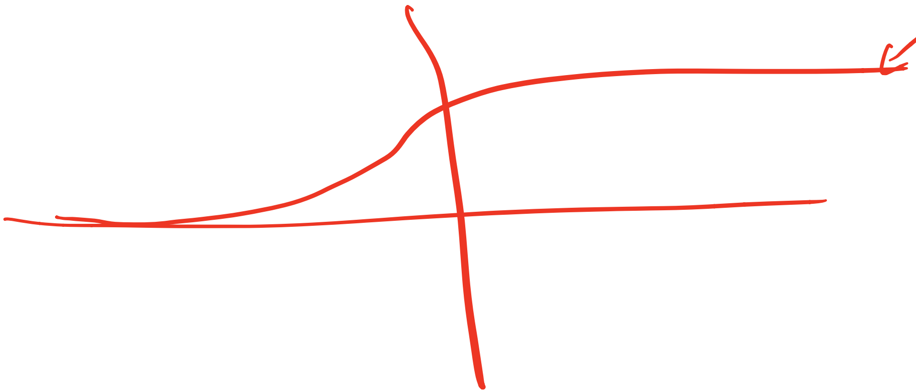
Linear regression doesn't work very well for estimating $P(Y = 1|X = x)$, since

- It doesn't make sense to extrapolate outside of $[0, 1]$.
- Least squares is an odd way to approximate probabilities.

We still like the idea of forming linear functions of our data, $\beta_0 + x_1\beta_1$ (who doesn't?).

We want a way to squish that linear function back into $[0, 1]$:

$$P(Y = 1|X = x) = \text{squish}(\beta_0 + \beta_1 x_1)$$



squish fn.

Logistic regression

In logistic regression, we model

$$\log\left\{\frac{\mathrm{P}(Y=1|X=x)}{\mathrm{P}(Y=0|X=x)}\right\} = \beta_0 + \beta^T x$$

log-odds

for some unknown $\beta_0 \in \mathbb{R}$, $\beta \in \mathbb{R}^p$, which we will estimate directly

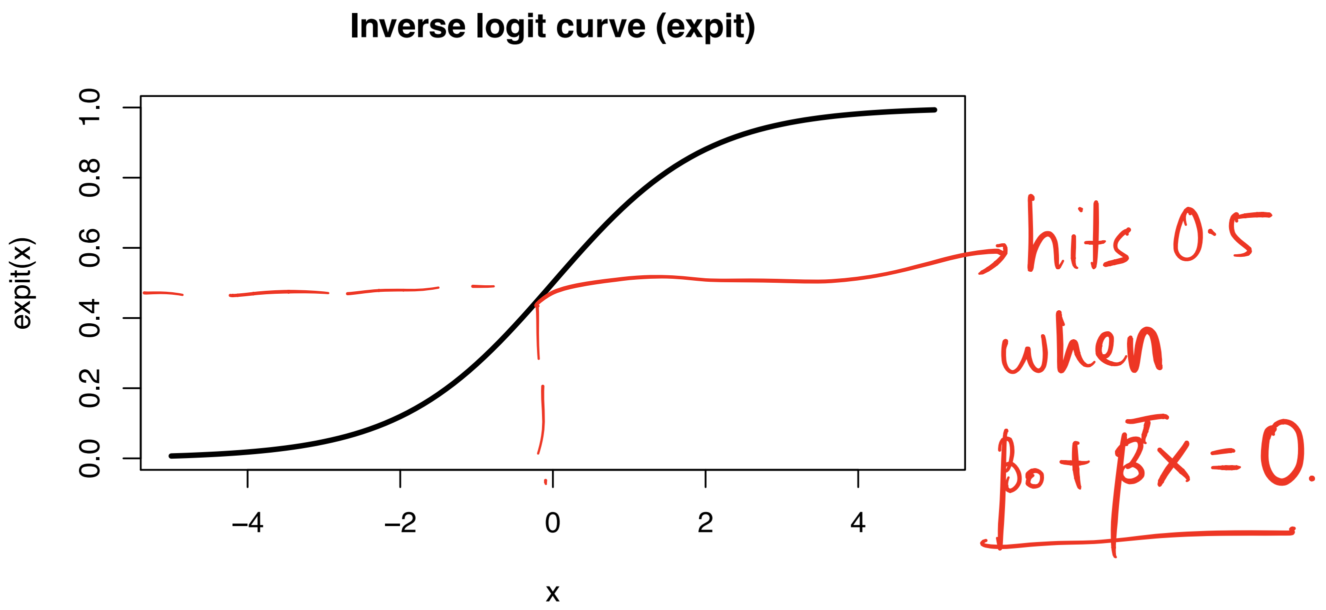
Note that $P(Y=0|X=x) = 1 - P(Y=1|X=x)$, and

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta^T x \quad \Leftrightarrow \quad \frac{p}{1-p} = \exp\{\beta_0 + \beta^T x\}$$

$$\Leftrightarrow \quad p = \frac{\exp\{\beta_0 + \beta^T x\}}{1 + \exp\{\beta_0 + \beta^T x\}}$$

So our model is equivalent to

$$\mathrm{P}(Y=1|X=x) = \frac{\exp(\beta_0 + \beta^T x)}{1 + \exp(\beta_0 + \beta^T x)}$$



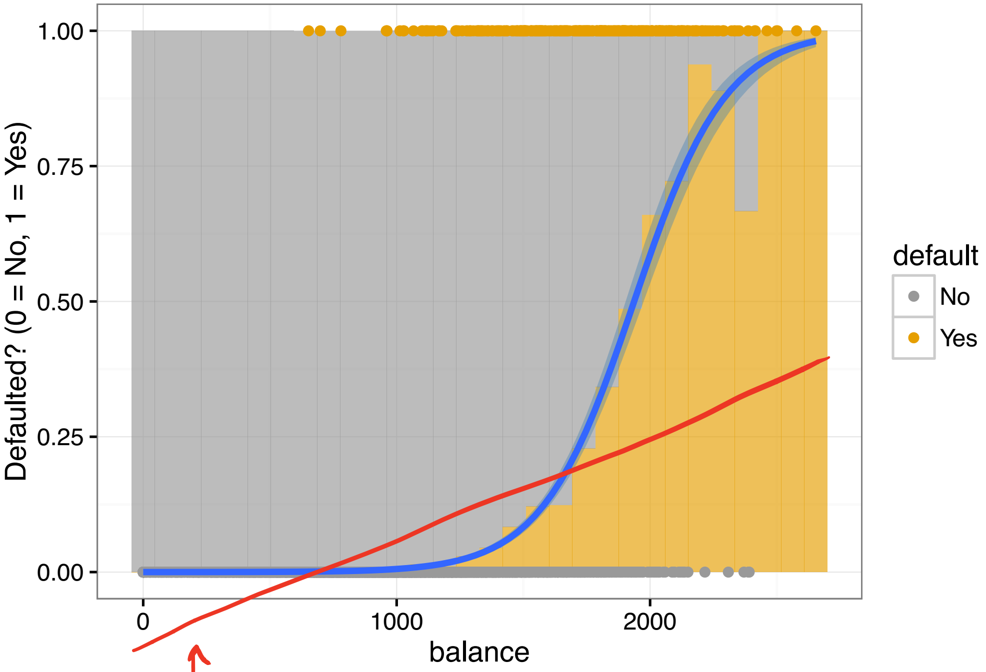
The function

$$\text{logit}^{-1}(z) = \text{expit}(z) = \frac{e^z}{1+e^z} = \frac{1}{1+e^{-z}}$$

is our desired "squishing" function, transforming real numbers into $[0, 1]$.

Logistic regression and the Default data

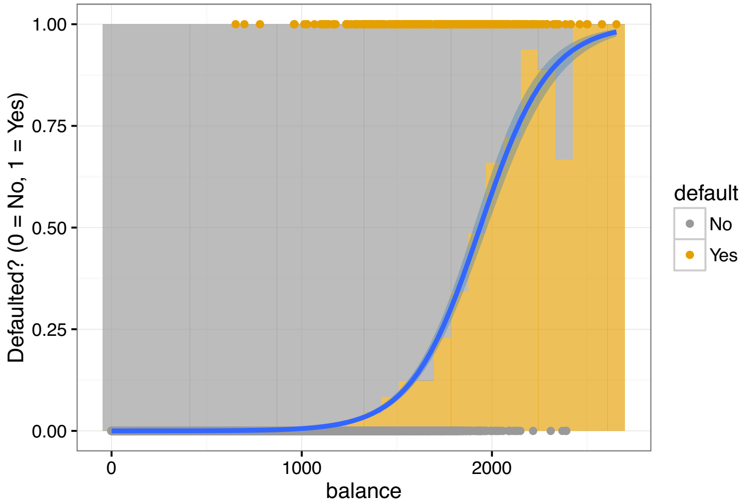
The logistic fit gives a much more reasonable estimate of $P(Y = 1|X = x)$!



contrast with

linear regression fit

Logistic regression and the Default data



$$\log \frac{P(\text{Default}|\text{Balance})}{1 - P(\text{Default}|\text{Balance})} = \beta_0 + \beta_1 \cdot \text{Balance}$$

Logistic regression: Estimated probabilities

Once we have estimated $\widehat{\beta_0}, \widehat{\beta}_1$, we can estimate conditonal probabilities:

$$\mathrm{P}(Y = 1 | X = x) = \frac{\exp(\widehat{\beta}_0 + \widehat{\beta}_1 x)}{1 + \exp(\widehat{\beta}_0 + \widehat{\beta}_1 x)}$$

For a balance of \$1000 or \$2000,

$$\widehat{p}(1000) = \frac{\exp(\widehat{\beta}_0 + \widehat{\beta}_1 \cdot 1000)}{1 + \exp(\widehat{\beta}_0 + \widehat{\beta}_1 \cdot 1000)} = 0.00576$$

$$\widehat{p}(2000) = \frac{\exp(\widehat{\beta}_0 + \widehat{\beta}_1 \cdot 2000)}{1 + \exp(\widehat{\beta}_0 + \widehat{\beta}_1 \cdot 2000)} = 0.586$$

Interpretation of logistic regression coefficients

We start to see that the coefficients are *interpretable*.

We have modeled

$$\log \frac{P(Y=1|X=x)}{P(Y=0|X=x)} = \beta_0 + \beta_1 x_1$$

The left side, $\log \frac{P(Y=1|X=x)}{P(Y=0|X=x)}$, is called the *log-odds* that $Y = 1$.

This means that the odds that $Y = 1$,

$$\frac{P(Y=1|X=x)}{P(Y=0|X=x)} = \exp\left\{\beta_0 + \beta_1 x_1\right\}$$

Increasing x_1 by one unit increases the estimated odds that $Y = 1$ by $e^{\hat{\beta}_1}$.

Multiple variables

We can extend this idea to multiple variables, just like linear regression.

For variables $x_1, \dots, x_p$, we model

$$\log \frac{P(Y=1|X=x)}{P(Y=0|X=x)} = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$$
$$= x^T \beta$$

→ we will often use this notation

→ hide β_0, imagine new covariate

$$x_0 = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}$$

Classification by logistic regression

Suppose that we fit a logistic regression, estimating $\widehat{\beta}_0, \widehat{\beta}$. How do we classify?

Recall that our optimal classifier chooses

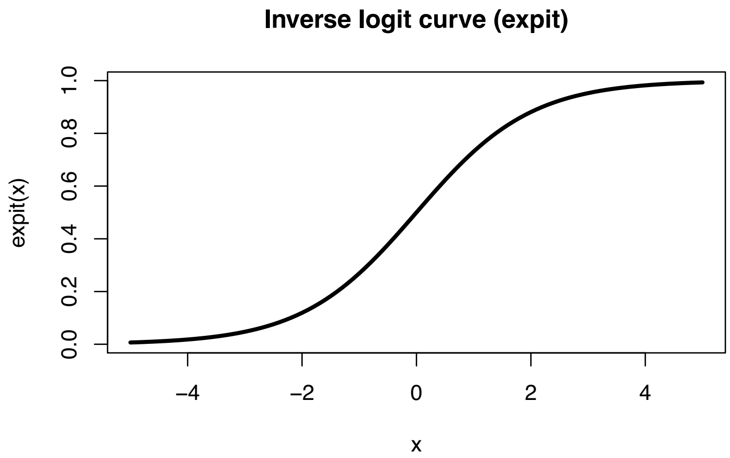
$$\underset{k}{\operatorname{argmax}}\, P(Y = k | X = x)$$

to minimize 0-1 loss.

Logistic regression gives us an estimate of $P(Y = k|X = x)$! We can just pick the category with the biggest value.

$$\widehat{f}(x) = \begin{cases} 1 & \text{if } \frac{\exp(x^T\widehat{\beta})}{1+\exp(x^T\widehat{\beta})} > 0.5 \\ 0 & \text{otherwise} \end{cases}$$

Classification by logistic regression



Remember that logit and expit are monotonically increasing! This gives us a much simpler rule!

$$\frac{\exp(x^T\widehat{\beta})}{1+\exp(x^T\widehat{\beta})} > 0.5 \quad \Leftrightarrow \quad x^T \hat{\beta} \geq 0.$$

Classification by logistic regression

This gives our final decision rule

$$\widehat{f}(x) = \begin{cases} 1 & \text{if } x^T\widehat{\beta} > 0 \\ 0 & \text{otherwise} \end{cases}$$

Therefore the decision boundary between classes 1 and 0 is the set of all $x \in \mathbb{R}^p$ such that

$$x^T\hat{\beta} = 0.$$

This is a point in $\mathbb{R}^1$ or a line in $\mathbb{R}^2$.

Linear classifier!

Decision boundary is linear.

Estimating logistic regression coefficients

To actually estimate the $\widehat{\beta}$, we just use maximum likelihood!

Suppose that we are given an i.i.d. sample (x_i, y_i), $i = 1, \dots n$. Here y_i denotes the class $\in \{0, 1\}$ of the ith observation. Then

$$\mathcal{L}(\beta) = \prod_{i=1}^{n} \mathrm{P}(C = y_i | X = x_i)$$

the likelihood of these n observations, so the log likelihood is

$$\ell(\beta) = \sum_{i=1}^{n} \log \mathrm{P}(C = y_i | X = x_i)$$

We just plug in our logistic model for these probabilities and optimize.

conditional

The log likelihood can be written as

$y_i \in \{0, 1\}$

$$\ell(\beta) = \sum_{i=1}^{n} \log \mathrm{P}(C = y_i | X = x_i)$$

$$= \sum_{i=1, y_i=1}^{n} \log \frac{\exp\{\beta^T x_i\}}{1+\exp\{\beta^T x_i\}} + \sum_{i: y_i=0}^{n} \log \left[\frac{1}{1+\exp\{\beta^T x_i\}} \right]$$

$$=$$

$$= \sum_{i=1}^{n} \left\{ y_i \cdot (\beta^T x_i) - \log\left(1 + \exp(\beta^T x_i)\right) \right\}$$

The coefficients are estimated by maximizing the likelihood,

$$\widehat{\beta} = \underset{\beta \in \mathbb{R}^p}{\operatorname{argmax}} \sum_{i=1}^{n} \left\{ y_i \cdot (\beta^T x_i) - \log\left(1 + \exp(\beta^T x_i)\right) \right\}$$

The 0/1 loss

- A natural question – why don't we just minimize the 0/1 loss on the training data?
- For the logistic model:

find hyperplane that makes fewest mistakes

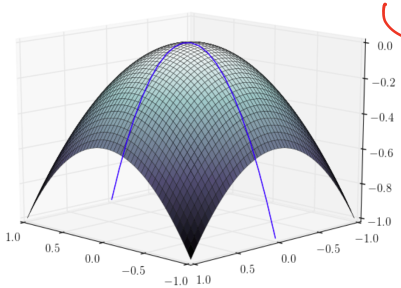
$$\widehat{\beta} = \underset{\cancel{\beta_0 \in \mathbb{R}}, \beta \in \mathbb{R}^p}{\operatorname{argmin}} \sum_{i=1}^{n} \mathbb{I}((2y_i - 1) \cdot (\beta^T x_i) < 0)$$

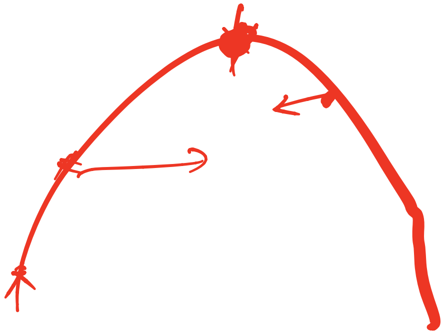
- Minimizing this function is generally computationally hard.

Convexity, loss minimization

(will not be on any HW/exam)

The maximum likelihood problem in logistic regression is an example of a *concave, maximization problem*.





- Cannot solve in closed form (unlike linear regression)
- Can solve using iterative schemes (like gradient ascent)

Multinomial Logistic Regression

We can generalize logistic regression to K classes, leveraging the same ideas.

$\in \mathbb{R}^p$

We now have *vectors* $\beta_1, \dots, \beta_K$, and define

$$\mathbb{P}(Y = k | X = x_i) = \frac{e^{x_i^T \beta_k}}{\sum_{j=1}^{K} e^{x_i^T \beta_j}}$$

It turns out that the β_k are not uniquely identifiable, you can eliminate one of them.

These probabilities are given by the *softmax* function, which we will see again in neural nets.

Logistic Regression with Linearly Separable Data

(ie) perfect linear classifier

- If the data is linearly separable (?) then weights go to ∞ and can overfit!
- **1D Example:**

$\rightarrow \beta_0 = -0.5 \quad \beta_1 = 1$

$\rightarrow$ what if I double βs?

- **2D Example:**

- Always regularize!

all perfect on training data.

Regularization

- Will return to this in more detail.
- **LASSO Logistic:**

$$\arg\max_{\beta} \sum_{i=1}^{n} y_i(x_i^T\beta) - \log(1+\exp(\beta^T x_i)) - \lambda\|\beta\|_1$$

- **Ridge Logistic:**

$$-\lambda\|\beta\|_2^2$$

- **Elastic Net Logistic:**

$$-(\lambda_1\|\beta\|_1 + \lambda_2\|\beta\|_2^2)$$