The Mathematics of Machine Learning 2025

Lecture 5

Tim van Erven

University of Amsterdam

Outline

Surrogate Losses for Classification

1. Motivation
2. Surrogate Losses

Log Loss

Logistic Regression

3. Logistic Probability Model
4. Logistic Loss
5. An Extra Reason to Regularize
6. Multiclass Logistic Regression

Surrogate Losses: Motivation

Classification: find $\hat{f}$ with small expected 0/1-loss:

$$\mathrm{EPE}(\hat{f}) = \mathop{\mathbb{E}}_{X,Y}\Big[L\big(Y, \hat{f}(X)\big)\Big],$$

where $L(Y, \hat{Y}) = \begin{cases} 0 & \text{if } \hat{Y} = Y \text{ is correct} \\ 1 & \text{if } \hat{Y} \neq Y \text{ is wrong.} \end{cases}$

Empirical Risk Minimization:

$$\hat{f} = \arg\min_{f \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} L\big(Y_i, f(X_i)\big)$$

Surrogate Losses: Motivation

Classification: find $\hat{f}$ with small expected 0/1-loss:

$$\mathrm{EPE}(\hat{f}) = \mathop{\mathbb{E}}_{X,Y}\Big[L\big(Y, \hat{f}(X)\big)\Big],$$

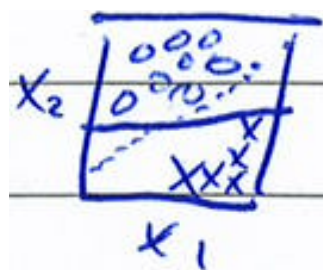
where $L(Y, \hat{Y}) = \begin{cases} 0 & \text{if } \hat{Y} = Y \text{ is correct} \\ 1 & \text{if } \hat{Y} \neq Y \text{ is wrong.} \end{cases}$

Empirical Risk Minimization:

$$\hat{f} = \arg\min_{f \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} L\big(Y_i, f(X_i)\big)$$

Difficulties:

- ERM solution not unique for linearly separable data
- Impossible to compute efficiently for non-separable data



Surrogate Losses I

Surrogate/Proxy Loss L':

Care about loss L, but train classifier with loss L'

Want:

$$L \text{ large} \Leftrightarrow L' \text{ large}$$
$$L \text{ small} \Leftrightarrow L' \text{ small}$$

Surrogate Losses I

Surrogate/Proxy Loss L':

Care about loss L, but train classifier with loss L'

Want:

$$L \text{ large} \Leftrightarrow L' \text{ large}$$
$$L \text{ small} \Leftrightarrow L' \text{ small}$$

Usually: surrogate loss L' **convex in parameters**, so can optimize efficiently

Surrogate Losses I

Surrogate/Proxy Loss L':

Care about loss L, but train classifier with loss L'

Want:

$$L \text{ large} \Leftrightarrow L' \text{ large}$$
$$L \text{ small} \Leftrightarrow L' \text{ small}$$

Usually: surrogate loss L' **convex in parameters**, so can optimize efficiently

Two most important surrogates:

logistic loss, ↑ today

hinge loss ↑ used in support vector machines (after midterm)

Surrogate Losses II

For $0/1$ loss with $Y \in \{-1, +1\}$, predict correctly if and only if

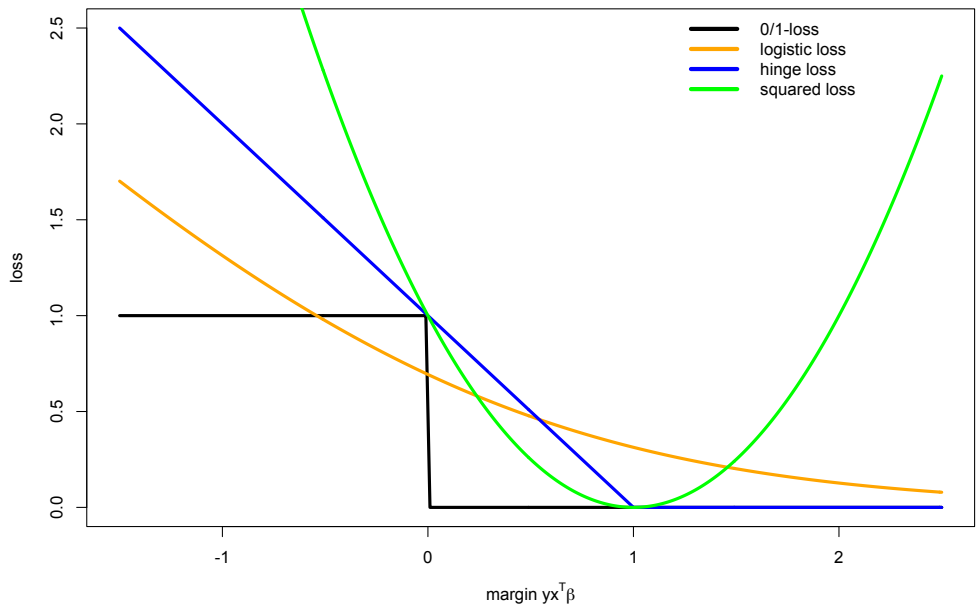
$$\text{sign}(X^\intercal \hat{\beta}) = Y \quad \Leftrightarrow \quad YX^\intercal \hat{\beta} > 0$$

Surrogate Losses II

For $0/1$ loss with $Y \in \{-1, +1\}$, predict correctly if and only if

$$\text{sign}(X^\intercal \hat{\beta}) = Y \quad \Leftrightarrow \quad YX^\intercal \hat{\beta} > 0$$

Most surrogate losses are therefore decreasing as a function of the **margin** $YX^\intercal \hat{\beta}$:

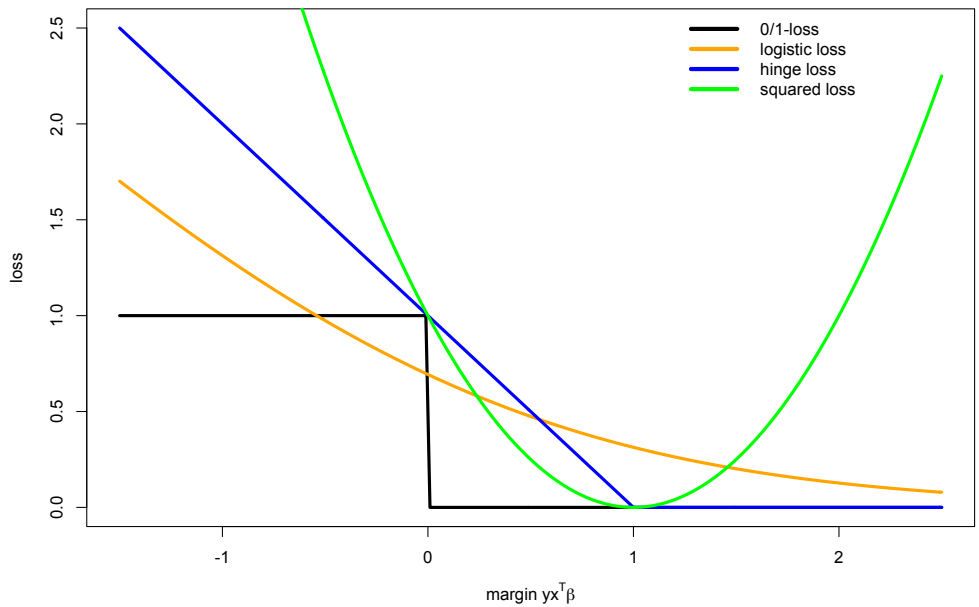


Surrogate Losses II

For 0/1 loss with $Y \in \{-1, +1\}$, predict correctly if and only if

$$\text{sign}(X^\intercal \hat{\beta}) = Y \quad \Leftrightarrow \quad YX^\intercal \hat{\beta} > 0$$

Most surrogate losses are therefore decreasing as a function of the **margin** $YX^\intercal \hat{\beta}$:



Exception: squared error $(Y - X^\intercal \hat{\beta})^2 = (1 - YX^\intercal \hat{\beta})^2$

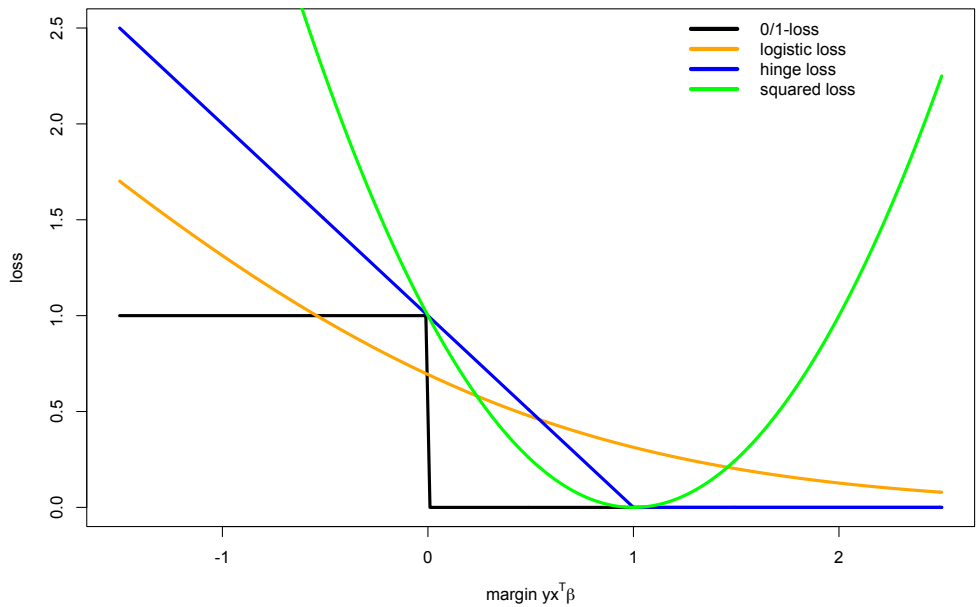
- Increases for $YX^\intercal \hat{\beta} > 1$.

Surrogate Losses II

For 0/1 loss with $Y \in \{-1, +1\}$, predict correctly if and only if

$$\text{sign}(X^\intercal \hat{\beta}) = Y \quad \Leftrightarrow \quad YX^\intercal \hat{\beta} > 0$$

Most surrogate losses are therefore decreasing as a function of the **margin** $YX^\intercal \hat{\beta}$:



Exception: squared error $(Y - X^\intercal \hat{\beta})^2 = (1 - YX^\intercal \hat{\beta})^2$

- Increases for $YX^\intercal \hat{\beta} > 1$.
- This is undesirable, so squared error not a natural choice for classification!

Outline

Surrogate Losses for Classification

1. Motivation
2. Surrogate Losses

Log Loss

Logistic Regression

3. Logistic Probability Model
4. Logistic Loss
5. An Extra Reason to Regularize
6. Multiclass Logistic Regression

Log(arithmic) Loss

For predictions that are probability (densities) p for Y:

$$L(Y, p) = -\log p(Y)$$

Small loss $\Leftrightarrow$ p assigns large probability (density) to Y

Log(arithmic) Loss

For predictions that are probability (densities) p for Y:

$$L(Y, p) = -\log p(Y)$$

Small loss $\Leftrightarrow$ p assigns large probability (density) to Y

Maximum likelihood = ERM for log loss:

$$\text{ML:} \qquad \arg\max_{\beta} \prod_{i=1}^{N} p_\beta(Y_i \mid X_i) = \arg\min_{\beta} \sum_{i=1}^{N} -\log p_\beta(Y_i \mid X_i)$$

$$= \arg\min_{\beta} \sum_{i=1}^{N} L\big(Y_i, p_\beta(\cdot \mid X_i)\big)$$

Log(arithmic) Loss

For predictions that are probability (densities) p for Y:

$$L(Y, p) = -\log p(Y)$$

Small loss $\Leftrightarrow$ p assigns large probability (density) to Y

Maximum likelihood = ERM for log loss:

$$\text{ML:} \qquad \arg\max_\beta \prod_{i=1}^N p_\beta(Y_i \mid X_i) = \arg\min_\beta \sum_{i=1}^N -\log p_\beta(Y_i \mid X_i)$$

$$= \arg\min_\beta \sum_{i=1}^N L\big(Y_i, p_\beta(\cdot \mid X_i)\big)$$

Bayes-optimal predictor for log loss is true conditional probability:

$$f_{\text{B}}(X) = \arg\min_p \mathbb{E}[L(Y, p) \mid X] = p^*(\cdot \mid X)$$

Outline

Surrogate Losses for Classification

1. Motivation
2. Surrogate Losses

Log Loss

Logistic Regression

3. Logistic Probability Model
4. Logistic Loss
5. An Extra Reason to Regularize
6. Multiclass Logistic Regression

Logistic Probability Model (for two classes)

Plug-in estimator for $P^*(Y \mid X)$ directly

- So not using Bayes' rule, like LDA or Naive Bayes
- This is called a **discriminative model** as opposed to a **generative model** when using Bayes' rule

How can we model the **class probabilities** $P^*(Y \mid X)$
with a **linear model** $f_\beta(X) = X^\intercal \beta$?

Logistic Probability Model (for two classes)

Plug-in estimator for $P^*(Y \mid X)$ directly

- So not using Bayes' rule, like LDA or Naive Bayes
- This is called a **discriminative model** as opposed to a **generative model** when using Bayes' rule

How can we model the **class probabilities** $P^*(Y \mid X)$
with a **linear model** $f_\beta(X) = X^\intercal \beta$?

Bad Ideas:

1. $P_\beta(Y = +1 \mid X) = f_\beta(X)$
2. $\log P_\beta(Y = +1 \mid X) = f_\beta(X) \quad \Leftrightarrow \quad P_\beta(Y = +1 \mid X) = e^{f_\beta(X)}$

- Probabilities would go outside of range $[0, 1]$

Logistic Probability Model (for two classes)

Plug-in estimator for $P^*(Y \mid X)$ directly

- So not using Bayes' rule, like LDA or Naive Bayes
- This is called a **discriminative model** as opposed to a **generative model** when using Bayes' rule

How can we model the **class probabilities** $P^*(Y \mid X)$
with a **linear model** $f_\beta(X) = X^\intercal \beta$?

Bad Ideas:

1. $P_\beta(Y = +1 \mid X) = f_\beta(X)$
2. $\log P_\beta(Y = +1 \mid X) = f_\beta(X) \quad \Leftrightarrow \quad P_\beta(Y = +1 \mid X) = e^{f_\beta(X)}$

- Probabilities would go outside of range $[0, 1]$

Logistic Regression: Linear log-odds:

$$\log \frac{P_\beta(Y = +1 \mid X)}{P_\beta(Y = -1 \mid X)} = f_\beta(X)$$

Logistic Probability Model (for two classes)

Plug-in estimator for $P^*(Y \mid X)$ directly

- So not using Bayes' rule, like LDA or Naive Bayes
- This is called a **discriminative model** as opposed to a **generative model** when using Bayes' rule

How can we model the **class probabilities** $P^*(Y \mid X)$
with a **linear model** $f_\beta(X) = X^\intercal \beta$?

Bad Ideas:

1. $P_\beta(Y = +1 \mid X) = f_\beta(X)$
2. $\log P_\beta(Y = +1 \mid X) = f_\beta(X) \quad \Leftrightarrow \quad P_\beta(Y = +1 \mid X) = e^{f_\beta(X)}$

- Probabilities would go outside of range $[0, 1]$

Logistic Regression: Linear log-odds:

$$\log \frac{P_\beta(Y = +1 \mid X)}{P_\beta(Y = -1 \mid X)} = f_\beta(X) \quad \Leftrightarrow \quad \begin{cases} P_\beta(Y = +1 \mid X) = \frac{e^{f_\beta(X)}}{1+e^{f_\beta(X)}} = \frac{1}{1+e^{-f_\beta(X)}} \\ P_\beta(Y = -1 \mid X) = \frac{1}{1+e^{f_\beta(X)}} \end{cases}$$

Logistic Probability Model (for two classes)

Plug-in estimator for $P^*(Y \mid X)$ directly

- So not using Bayes' rule, like LDA or Naive Bayes
- This is called a **discriminative model** as opposed to a **generative model** when using Bayes' rule

How can we model the **class probabilities** $P^*(Y \mid X)$ with a **linear model** $f_\beta(X) = X^\intercal \beta$?

Logistic Regression: Linear log-odds:

$$\log \frac{P_\beta(Y=+1 \mid X)}{P_\beta(Y=-1 \mid X)} = f_\beta(X) \quad \Leftrightarrow \quad \begin{cases} P_\beta(Y=+1 \mid X) = \frac{e^{f_\beta(X)}}{1+e^{f_\beta(X)}} = \frac{1}{1+e^{-f_\beta(X)}} \\ P_\beta(Y=-1 \mid X) = \frac{1}{1+e^{f_\beta(X)}} \end{cases}$$

$$P_\beta(Y \mid X) = \frac{1}{1+e^{-Yf_\beta(X)}}$$

$$P_{\hat{\beta}}(Y=+1 \mid X) > \tfrac{1}{2} \Leftrightarrow f_{\hat{\beta}}(X) > 0$$

- Decision boundary: $P_{\hat{\beta}}(Y=+1 \mid X) = \frac{1}{2} \Leftrightarrow f_{\hat{\beta}}(X) = 0$ is linear

Logistic Loss

Logistic loss = log loss for logistic regression

$$L\big(Y, f_\beta(X)\big) = -\log P_\beta(Y \mid X) = \log\left(1 + e^{-Yf_\beta(X)}\right)$$

Logistic Loss

Logistic loss = log loss for logistic regression

$$L\big(Y, f_\beta(X)\big) = -\log P_\beta(Y \mid X) = \log\left(1 + e^{-Yf_\beta(X)}\right)$$

Training:

With maximum likelihood = ERM for logistic loss

$$\min_\beta \quad \sum_{i=1}^{N} \log\left(1 + e^{-Y_i f_\beta(X_i)}\right)$$

No formula for minimum, but can find it by convex optimization

Logistic Loss

Logistic loss = log loss for logistic regression

$$L\big(Y, f_\beta(X)\big) = -\log P_\beta(Y \mid X) = \log\left(1 + e^{-Yf_\beta(X)}\right)$$

Training:

With maximum likelihood = ERM for logistic loss

$$\min_{\beta_0,\beta} \quad \sum_{i=1}^{N} \log\left(1 + e^{-Y_i(\beta_0 + X_i^\intercal \beta)}\right)$$

No formula for minimum, but can find it by convex optimization

Logistic Loss

Logistic loss = log loss for logistic regression

$$L\big(Y, f_\beta(X)\big) = -\log P_\beta(Y \mid X) = \log\left(1 + e^{-Yf_\beta(X)}\right)$$

Training:

With maximum likelihood = ERM for logistic loss

$$\min_{\beta_0,\beta} \quad \sum_{i=1}^{N} \log\left(1 + e^{-Y_i(\beta_0 + X_i^\intercal \beta)}\right)$$

No formula for minimum, but can find it by convex optimization

Important Extensions:

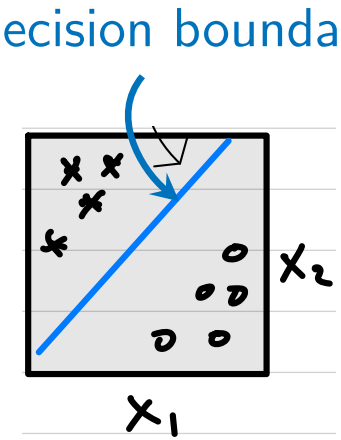
$$\min_{\beta_0,\beta} \quad \sum_{i=1}^{N} \log\left(1 + e^{-Y_i(\beta_0 + X_i^\intercal \beta)}\right) + \lambda\,\mathrm{pen}(\beta)$$

ℓ_2-penalty:	$\mathrm{pen}(\beta) = \|\beta\|_2$	(like ridge regression)
ℓ_1-penalty:	$\mathrm{pen}(\beta) = \|\beta\|_1$	(like Lasso)

- Considerations for penalization similar as for squared error

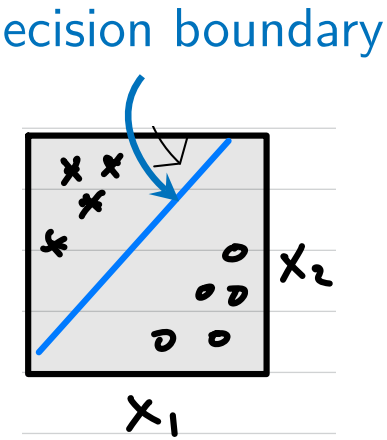
An Extra Reason to Regularize

Consider linearly separable data:



An Extra Reason to Regularize

Consider linearly separable data:

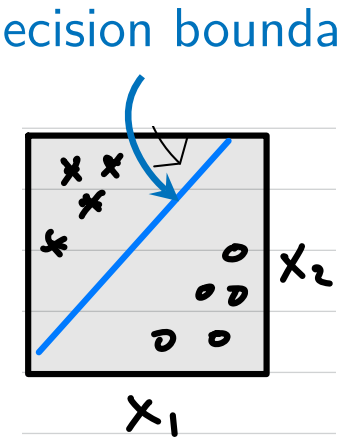


Let $\hat{f}(X) = \hat{\beta}_0 + X^\intercal \hat{\beta}$.
Suppose we **scale** both $\hat{\beta}_0$ and $\hat{\beta}$ by any $\alpha > 0$. Then

- $\hat{f}(X) \to \alpha \hat{f}(X)$ for any X

An Extra Reason to Regularize

Consider linearly separable data:

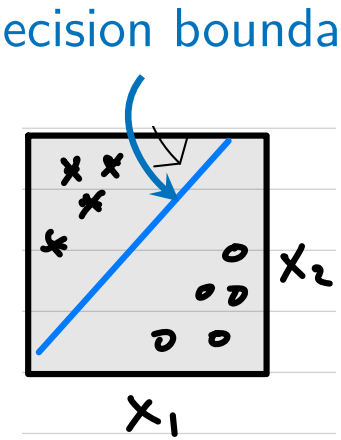


Let $\hat{f}(X) = \hat{\beta}_0 + X^\intercal \hat{\beta}$.
Suppose we **scale** both $\hat{\beta}_0$ and $\hat{\beta}$ by any $\alpha > 0$. Then

- $\hat{f}(X) \to \alpha \hat{f}(X)$ for any X
- Does not change sign $\left(\hat{f}(X)\right)$, so **decision boundary stays the same**

An Extra Reason to Regularize

Consider linearly separable data:

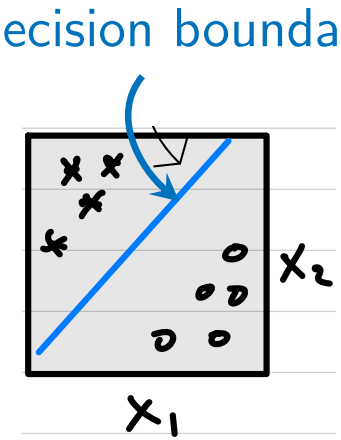


Let $\hat{f}(X) = \hat{\beta}_0 + X^\intercal \hat{\beta}$.
Suppose we **scale** both $\hat{\beta}_0$ and $\hat{\beta}$ by any $\alpha > 0$. Then

- $\hat{f}(X) \to \alpha \hat{f}(X)$ for any X
- Does not change sign $\left(\hat{f}(X)\right)$, so **decision boundary stays the same**
- But, for linearly separable data, will **increase the margin** $Y_i \hat{f}(X_i)$ for all training data points (X_i, Y_i): $Y_i \hat{f}(X_i) \to \alpha Y_i \hat{f}(X_i)$

An Extra Reason to Regularize

Consider linearly separable data:



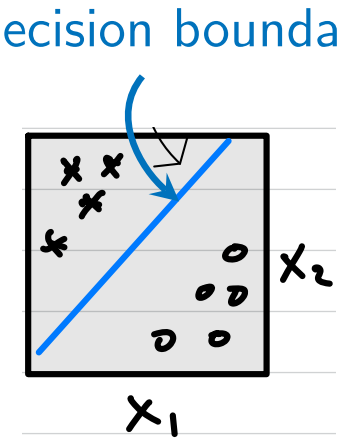
Let $\hat{f}(X) = \hat{\beta}_0 + X^\intercal \hat{\beta}$.
Suppose we **scale** both $\hat{\beta}_0$ and $\hat{\beta}$ by any $\alpha > 0$. Then

- $\hat{f}(X) \to \alpha \hat{f}(X)$ for any X
- Does not change sign $\left(\hat{f}(X)\right)$, so **decision boundary stays the same**
- But, for linearly separable data, will **increase the margin** $Y_i \hat{f}(X_i)$ for all training data points (X_i, Y_i): $Y_i \hat{f}(X_i) \to \alpha Y_i \hat{f}(X_i)$
- So to minimize logistic loss for linearly separable data, take α as large as possible:

$$\left\| \alpha \begin{pmatrix} \hat{\beta}_0 \\ \hat{\beta} \end{pmatrix} \right\| \to \infty$$

An Extra Reason to Regularize

Consider linearly separable data:



Let $\hat{f}(X) = \hat{\beta}_0 + X^T\hat{\beta}$.
Suppose we **scale** both $\hat{\beta}_0$ and $\hat{\beta}$ by any $\alpha > 0$. Then

- So to minimize logistic loss for linearly separable data, take α as large as possible:

$$\left\| \alpha \begin{pmatrix} \hat{\beta}_0 \\ \hat{\beta} \end{pmatrix} \right\| \to \infty$$

- Your optimizer: **"Warning, optimization did not converge!"**
- Penalization avoids this problem, because $\text{pen}(\alpha\hat{\beta}) > \text{pen}(\hat{\beta})$, so penalty pushes back against too large α.

Multiclass Logistic Regression: ML Version

(Assume implicit intercept again.)

Machine Learning Version:

Parameter vector for each class: $\beta_1, \ldots, \beta_K \in \mathbb{R}^p$,
$\beta = (\beta_1, \ldots, \beta_K) \in \mathbb{R}^{p \times K}$

$$P_\beta(Y = k \mid X) = \frac{e^{X^\intercal \beta_k}}{\sum_{j=1}^{K} e^{X^\intercal \beta_j}}$$

Multiclass Logistic Regression: ML Version

(Assume implicit intercept again.)

Machine Learning Version:

Parameter vector for each class: $\beta_1, \ldots, \beta_K \in \mathbb{R}^p$,
$\beta = (\beta_1, \ldots, \beta_K) \in \mathbb{R}^{p \times K}$

$$P_\beta(Y = k \mid X) = \frac{e^{X^\intercal \beta_k}}{\sum_{j=1}^K e^{X^\intercal \beta_j}}$$

- Not **identifiable**: multiple β map to the same probability distribution:

$$\frac{e^{X^\intercal \beta_k}}{\sum_{j=1}^K e^{X^\intercal \beta_j}} = \frac{e^{X^\intercal (\beta_k - v)}}{\sum_{j=1}^K e^{X^\intercal (\beta_j - v)}} \qquad \text{for any } v \in \mathbb{R}^p$$

Multiclass Logistic Regression: ML Version

(Assume implicit intercept again.)

Machine Learning Version:

Parameter vector for each class: $\beta_1, \ldots, \beta_K \in \mathbb{R}^p$,
$\beta = (\beta_1, \ldots, \beta_K) \in \mathbb{R}^{p \times K}$

$$P_\beta(Y = k \mid X) = \frac{e^{X^\intercal \beta_k}}{\sum_{j=1}^K e^{X^\intercal \beta_j}}$$

- Not **identifiable**: multiple β map to the same probability distribution:

$$\frac{e^{X^\intercal \beta_k}}{\sum_{j=1}^K e^{X^\intercal \beta_j}} = \frac{e^{X^\intercal (\beta_k - v)}}{\sum_{j=1}^K e^{X^\intercal (\beta_j - v)}} \qquad \text{for any } v \in \mathbb{R}^p$$

- But identifiability does not matter for machine learning, because only cares about predictions P_β, not about β.
- Nice symmetry: treats all classes the same
- Does **not** simplify to binary logistic regression for $K = 2$

Multiclass Logistic Regression: Stats Version

(Assume implicit intercept again.)

Statistical Version:

$\beta = (\beta_1, \ldots, \beta_{K-1}) \in \mathbb{R}^{p \times (K-1)}$

$$P_\beta(Y = k \mid X) = \begin{cases} \frac{e^{X^\intercal \beta_k}}{1 + \sum_{j=1}^{K-1} e^{X^\intercal \beta_j}} & \text{for } k \leq K-1 \\ \frac{1}{1 + \sum_{j=1}^{K-1} e^{X^\intercal \beta_j}} & \text{for } k = K \end{cases}$$

- Coincides with ML version if we fix $\beta_K = 0$
- Parameters β **identifiable**: one-to-one mapping $P_\beta \leftrightarrow \beta$
  Important because statistics wants to learn **true parameters** β^*!
- Breaks symmetry between classes for $K > 2$: class K treated differently
- Reduces to binary logistic regression for $K = 2$, for which it is therefore still symmetrical