Logistic Regression

Concepts:

Maximum likelihood estimation application to LR

Maximum a posterior (MAP) estimation and connection to regularization

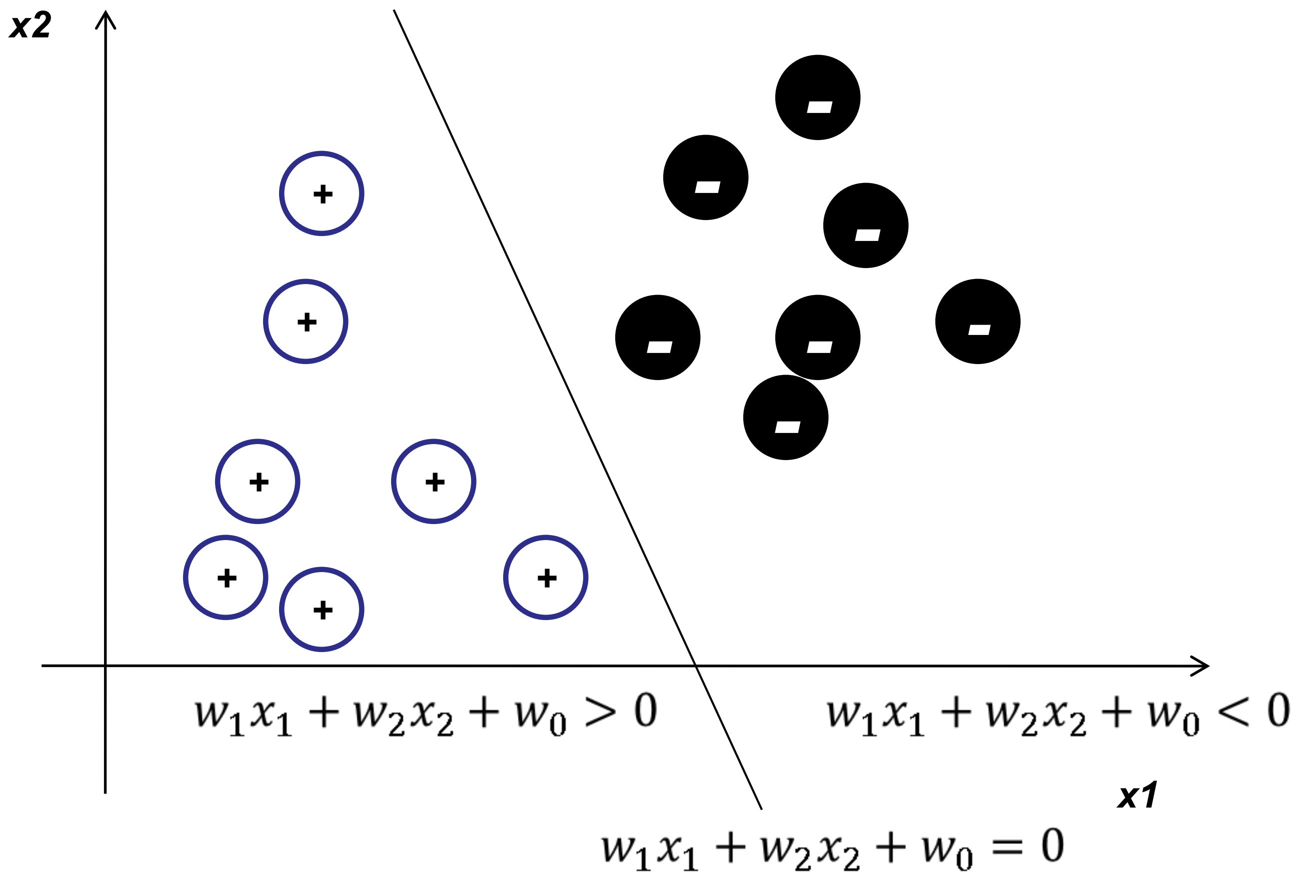
Classification problem

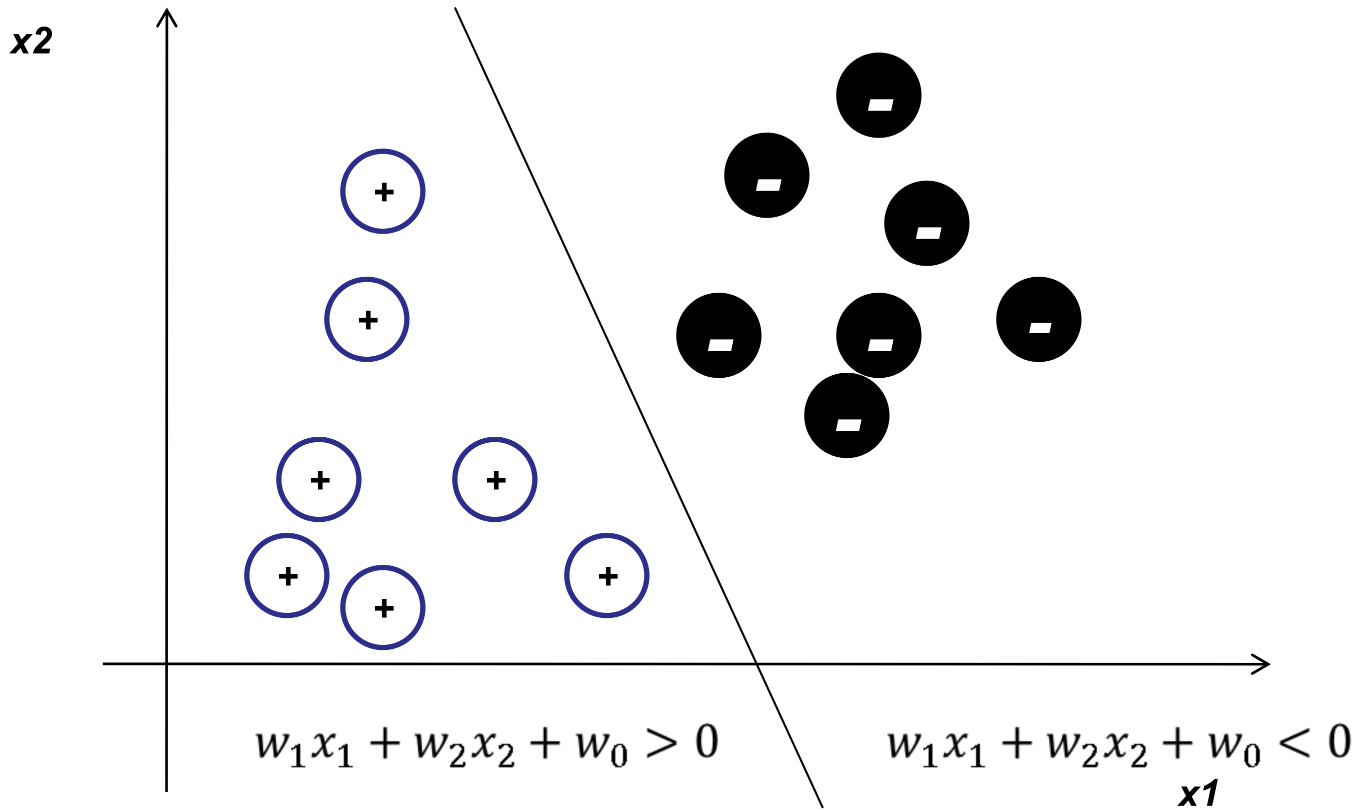
- Given input $\mathbf{x}$, the goal is to predict y, which is a categorical variable
  - $\mathbf{x}$: the feature vector
  - y: the class label

- Example:
  - $\mathbf{x}$: monthly income and bank saving amount;
    y: risky or not risky (binary)
  - $\mathbf{x}$: review text for a product
    y: sentiment positive, negative or neutral (multiclass)

Binary Linear Classifier

- We will be begin with the simplest choice: linear classifiers for binary classification problems ($y \in \{0,1\}$)





$$w_1x_1 + w_2x_2 + w_0 = 0$$

w_1, w_2 decides the slope of the decision boundary. Vector (w_1, w_2) has the following properties:

- is perpendicular to the decision boundary and
- points to the positive side

w_0 is called bias or intercept of the decision boundary, changing w_0 moves the decision boundary up and down

There are many ways we can learn the linear decision boundary separating the two classes

Logistic regression: models the target as a Bernoulli random variable and learns conditional distribution $P(y|\mathbf{x})$

Input $\mathbf{x} = [1, x_1, x_2, \ldots, x_d]^T$, target output $y \in \{0,1\}$

Logistic regression assumes:

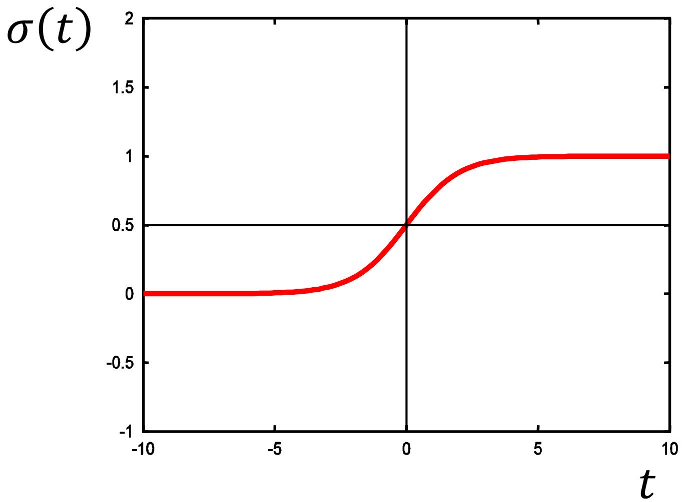
$$P(y = 1|\mathbf{x}; \mathbf{w}) = \sigma\left(\mathbf{w}^{\mathbf{T}}\mathbf{x}\right) = \frac{1}{1 + \exp(-\mathbf{w}^{\mathbf{T}}\mathbf{x})}$$

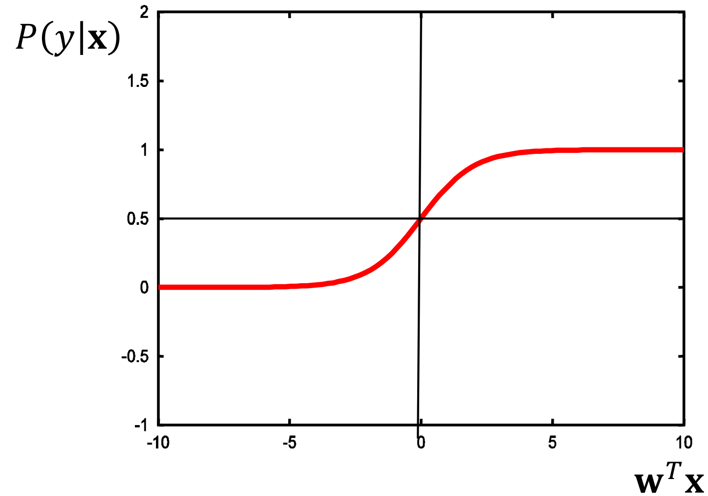
$$P(y = 0|\mathbf{x}; \mathbf{w}) = 1 - \sigma\left(\mathbf{w}^{\mathbf{T}}\mathbf{x}\right) = \frac{\exp(-\mathbf{w}^{\mathbf{T}}\mathbf{x})}{1 + \exp(-\mathbf{w}^{\mathbf{T}}\mathbf{x})}$$

Logistic Regression

- Sigmoid function:

$$\sigma(t) = \frac{1}{1 + \exp(-t)}$$





- In linear regression we use linear function $y = \mathbf{w}^T\mathbf{x} \in (-\infty, \infty)$
- For classification we need $y \in \{0,1\}$
- Use the sigmoid function $\sigma: R \to (0,1)$ to warp the value of $\mathbf{w^T x}$ to a value between 0 and 1, interpreted as $P(y|\mathbf{x})$

How to make prediction with $P(y|\mathbf{x})$?

- Deciding the optimal prediction based on $P(y|\mathbf{x})$ is the topic of decision theory
- In HW0 you have seen decision theory in action, where the goal is the minimize expected loss
- When we have balanced loss, the decision rule reduces to Maximum A-Postier prediction

$$y_{map} = \arg\max_{v \in \{0,1\}} P(y = v|\mathbf{x}; \mathbf{w})$$

Logistic regression learns a linear classifier

- Maximum A Posteriori (MAP) prediction of y:

$$y_{map} = \arg\max_{v \in \{0,1\}} P(y = v | \mathbf{x}; \mathbf{w})$$

- We will predict $y = 1$ if

$$P(y = 1 | \mathbf{x}; \mathbf{w}) \geq P(y = 0 | \mathbf{x}; \mathbf{w}) \Rightarrow$$

$$\frac{1}{1 + \exp(-\mathbf{w}^{\mathbf{T}}\mathbf{x})} \geq \frac{\exp(-\mathbf{w}^{\mathbf{T}}\mathbf{x})}{1 + \exp(-\mathbf{w}^{\mathbf{T}}\mathbf{x})} \Rightarrow$$

Canceling the denominator

$$1 \geq \exp\left(-\mathbf{w}^{\mathbf{T}}\mathbf{x}\right) \Rightarrow$$

Log on both sides

$$0 \geq -\mathbf{w}^{\mathbf{T}}\mathbf{x} \Rightarrow \mathbf{w}^{\mathbf{T}}\mathbf{x} \geq \mathbf{0}$$

- MAP decision boundary is $\mathbf{w}^{\mathbf{T}}\mathbf{x} = 0$, which is linear
- More generally for asymmetric loss matrices, this also leads to comparing $\mathbf{w}^{\mathbf{T}}\mathbf{x}$ with different threshold c

$$\mathbf{w}^{T}\mathbf{x} = c$$

Learning for Logistic Regression

- Given a set of training examples:

$$D = \{(\mathbf{x}_1, y_1) \ldots (\mathbf{x}_N, y_N)\}$$

- We assume examples are identically, independently distributed (I.I.D.) following:

$$P(y = 1|\mathbf{x}; \mathbf{w}) = \frac{1}{1 + \exp(-\mathbf{w}^T \mathbf{x})}$$

$$P(y = 0|\mathbf{x}; \mathbf{w}) = \frac{\exp(-\mathbf{w}^\mathbf{T} \mathbf{x})}{1 + \exp(-\mathbf{w}^\mathbf{T} \mathbf{x})}$$

- Learn $\mathbf{w}$ from the training data using Maximum Likelihood Estimation

Maximum (conditional) Likelihood Estimation

- Data log-likelihood:

$$\log \prod_{i=1}^{N} P(\mathbf{x}_i, y_i; \mathbf{w}) = \sum_i \log P(\mathbf{x}_i, y_i; \mathbf{w})$$

$$= \sum_i \log P(y_i | \mathbf{x}_i; \mathbf{w}) P(\mathbf{x}_i; \mathbf{w})$$

Distribution of $\mathbf{x}$, *has nothing to do with* $\mathbf{w}$, thus we don't care

$$= \sum_i \log P(y_i | \mathbf{x}_i; \mathbf{w}) + C$$

We only care about the mapping from x to y --- in doing so, we are learning a discriminative model

- Maximum (conditional) likelihood objective for learning $\mathbf{w}$:

$$\mathbf{w}_{MLE} = \text{argmax}_{\mathrm{w}} \sum_i \log P(y_i | \mathbf{x}_i; \mathbf{w})$$

Computing Log-likelihood

$$\boxed{P(y_i|\mathbf{x}_i;\mathbf{w}) = P(y=1|\mathbf{x}_i;\mathbf{w})^{y_i}\, P(y=0|\mathbf{x}_i;\mathbf{w})^{(1-y_i)}}$$

$$l(\mathbf{w}) = \sum_i \log P(y_i|\mathbf{x}_i;\mathbf{w})$$

$$= \sum_i [y_i \log P(y=1|\mathbf{x}_i;\mathbf{w}) + (1-y_i)\log P(y=0|\mathbf{x}_i;\mathbf{w})]$$

$$= \sum_i y_i \log \frac{P(y=1|\mathbf{x}_i;\mathbf{w})}{P(y=0|\mathbf{x}_i;\mathbf{w})} + \log P(y=0|\mathbf{x}_i;\mathbf{w})$$

$$= \sum_i y_i \mathbf{w}^{\mathbf{T}}\mathbf{x}_i + \log(1-\sigma(\mathbf{w}^T\mathbf{x}_i))$$

Gradient

$$\sigma'(t) = \sigma(t)\big(1 - \sigma(t)\big)$$

$$l(\mathbf{w}) = \sum_i y_i \mathbf{w}^{\mathbf{T}} \mathbf{x}_i + \log(1 - \sigma(\mathbf{w}^T \mathbf{x}_i))$$

$$\nabla l(\mathbf{w}) = \sum_i \left[y_i \mathbf{x}_i + \frac{\nabla\big(1 - \sigma(\mathbf{w}^T \mathbf{x}_i)\big)}{1 - \sigma(\mathbf{w}^T \mathbf{x}_i)} \right]$$

$$\sum_i \left[y_i \mathbf{x}_i - \frac{\sigma(\mathbf{w}^T \mathbf{x}_i)\, \cancel{(1 - \sigma(\mathbf{w}^T \mathbf{x}_i))}\mathbf{x}_i}{\cancel{1 - \sigma(\mathbf{w}^T \mathbf{x}_i)}} \right]$$

$$= \sum_i [y_i \mathbf{x}_i - \sigma(\mathbf{w}^T \mathbf{x}_i)\mathbf{x}_i]$$

$$= \sum_i [y_i - \sigma(\mathbf{w}^T \mathbf{x}_i)]\, \mathbf{x}_i$$

Batch Gradient <u>Ascent</u> for LR

Given : training examples $(\boldsymbol{x}_i\ ,\ y_i),\ i = 1,\ldots,N$

Let $\boldsymbol{w} \leftarrow \boldsymbol{w}_0$ // e.g., (0,0,0,...,0) *// Random initialization*

Repeat until convergence

$\boldsymbol{d} \leftarrow (0,0,0,\ldots,0)$ *// d: gradient vector*

For $i = 1$ to N do *// for loop can be efficiently implemented via matrix multiplication*

$\widehat{y}_i \leftarrow \sigma(\boldsymbol{w}^T \boldsymbol{x}_{\boldsymbol{i}})$

$\boldsymbol{d} = \boldsymbol{d} + (y_i - \widehat{y}_i) \cdot \boldsymbol{x}_i$

$\boldsymbol{w} \leftarrow \boldsymbol{w} + \gamma\, \boldsymbol{d}$ *// Update along gradient direction to increase likilihood*

γ: learning rate

Stochastic Gradient Ascent for LR

Given : training examples $(\boldsymbol{x}_i\ ,\ y_i),\ i = 1,\ldots,N$

Let $\boldsymbol{w} \leftarrow \boldsymbol{w}_0$ // e.g., (0,0,0,...,0)

Repeat until convergence

Randomly shuffle examples

For $i = 1$ to N do

$$\widehat{y}_i \leftarrow \sigma(\boldsymbol{w}^T \boldsymbol{x}_i)$$

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \gamma\,(y_i - \widehat{y}_i)\boldsymbol{x}_i$$

Compare to linear regression update rule:

$$\mathbf{w} \leftarrow \mathbf{w} + \gamma(\boldsymbol{y_i} - \mathbf{w}^T\mathbf{x}_i)\mathbf{x}_i$$

- Stochastic gradient ascent performs updates for each example
- Learning shows more fluctuations than batch gradient ascent
- Shuffling the examples in each round (epoch) helps to make it more robust

Soft-max Logistic Regression for $K > 2$ classes

- For $K > 2$ classes, we learn K weight vectors: $\boldsymbol{W} = \{\mathbf{w}_1, \ldots, \mathbf{w}_K\}$ one per class
- Define the probability using the soft-max function

$$P(y = k | \mathbf{x}, \boldsymbol{W}) = \hat{y}_k = \frac{\exp\left(\mathbf{w}_k^T \mathbf{x}\right)}{\sum_{j=1}^{K} \exp\left(\mathbf{w}_j^T \mathbf{x}\right)}$$

- With MLE, we can arrive at the following gradient:

Graident w.r.t. $\mathbf{w}_k$

$$\nabla l(\mathbf{w}_k) = \sum_{i=1}^{N} (y_{ik} - \hat{y}_{ik})\mathbf{x}_i$$

where $[y_{i1}, y_{i2}, \ldots, y_{iK}]^T \in \{0,1\}^K$ is one-hot encoding of y_i

One-hot encoding quick example: $y \in \{1, 2, 3, 4\}$,
one hot encoding for $y = 2$: $[0, 1, 0, 0]^T$ for $y = 4$: $[0, 0, 0, 1]^T$

Logistic regression overfits

- Consider the gradient:

$$\sum_i [y_i - P(y=1|\mathbf{x}_i;\mathbf{w})]\,\mathbf{x}_i$$

If we have a binary feature x_p that only takes value 1 for positive examples, i.e., this feature perfectly classify the examples

What will happen to $\frac{\partial l}{\partial x_p}$? Will it ever be zero?

What will happen to w_p?

Logistic regression overfits

- Consider the gradient:

$$\sum_i [y_i - P(y = 1|\mathbf{x}_i; \mathbf{w})]\, \mathbf{x}_i$$

If we have a binary feature x_p that only takes value 1 for positive examples, i.e., this feature perfectly classify the examples

What will happen to $\frac{\partial l}{\partial x_p}$? Will it ever be zero?

What will happen to w_p?

It will keep increasing because $y_i - P(y = 1|\mathbf{x}_i; \mathbf{w})$ never fully reaches zero for positive examples

In general, when data is linearly separable, LR overfits as it will always try to increase $|\mathbf{w}^T\mathbf{x}|$ to increase likelihood – so we should regularize the weights

Regularization as MAP Estimation

- So far we have introduced MLE for logistic regression
- We will now introduce another paradigm for estimating model parameters
  - The Bayesian paradigm

Bayesian vs. Frequentist

- Two different views for parameter estimation
- Frequentist: a parameter θ is a deterministic unknown value
- Bayesian: a parameter is a random variable with a distribution
  - Use priors to express our belief/preference about the parameter before observing any data
  - After observing the data, update our belief by computing the posterior distribution of the parameter

Prior distribution of θ

$$p(\theta|D) = \frac{p(\theta)p(D|\theta)}{p(D)} = \frac{p(\theta)p(D|\theta)}{\int p(D|\theta)p(\theta)d\theta}$$

Posterior distribution of θ

Maximum A Posteriori (MAP) estimation as a penalty method

$$\hat{\theta}_{MAP} = \underset{\theta}{\operatorname{argmax}}\, p(\theta|D)$$
$$= \underset{\theta}{\operatorname{argmax}} \frac{p(D|\theta)p(\theta)}{p(D)}$$
$$= \underset{\theta}{\operatorname{argmax}}\, p(D|\theta)p(\theta)$$
$$= \underset{\theta}{\operatorname{argmax}}\ \log p(D|\theta) + \log p(\theta)$$

Penalty term /Regularization
Different prior lead to different regularization terms

MAP for Logistic Regression

$$\underset{\mathbf{w}}{\operatorname{argmax}} P(\mathbf{w}|\boldsymbol{D}) = \underset{\mathbf{w}}{\operatorname{argmax}} P(\boldsymbol{D}|\mathbf{w})\, P(\mathbf{w})$$
$$= \underset{\mathbf{w}}{\operatorname{argmax}} \log P(\boldsymbol{D}|\mathbf{w}) + \log P(\mathbf{w})$$

- $\log P(D|\mathbf{w})$: the log-likelihood of $\mathbf{w}$

$$\sum_i \log P(y_i|\mathbf{x}_i, \mathbf{w})$$

- $P(\mathbf{w})$: a prior distribution. A common prior is:

$$\mathbf{w} \sim N(0, \sigma^2\mathbf{I})$$

This is natural as large weights often associate with overfitting, it is natural to assume a prior that prefers simpler hypothesis (i.e, zero mean)

Logistic Regression: MAP

$$\underset{\mathbf{w}}{\operatorname{argmax}} \log P(D|\mathbf{w}) + \log P(\mathbf{w})$$

$$= \underset{\mathbf{w}}{\operatorname{argmax}}\, l(\mathbf{w}) + \log N(\mathbf{w}; 0, \sigma^2 \mathbf{I})$$

$$= \underset{\mathbf{w}}{\operatorname{argmax}}\, l(\mathbf{w}) + \sum_j \log\left(\frac{1}{\sqrt{2\pi}\sigma} \exp \frac{-w_j^2}{2\sigma^2}\right)$$

$$= \underset{\mathbf{w}}{\operatorname{argmax}}\, l(\mathbf{w}) + \sum_j \frac{-w_j^2}{2\sigma^2}$$

$\boxed{\lambda = \frac{1}{\sigma^2}\text{, regularization parameter}}$

$$= \underset{\mathbf{w}}{\operatorname{argmax}}\, l(\mathbf{w}) - \frac{\lambda}{2} \sum_j w_j^2$$

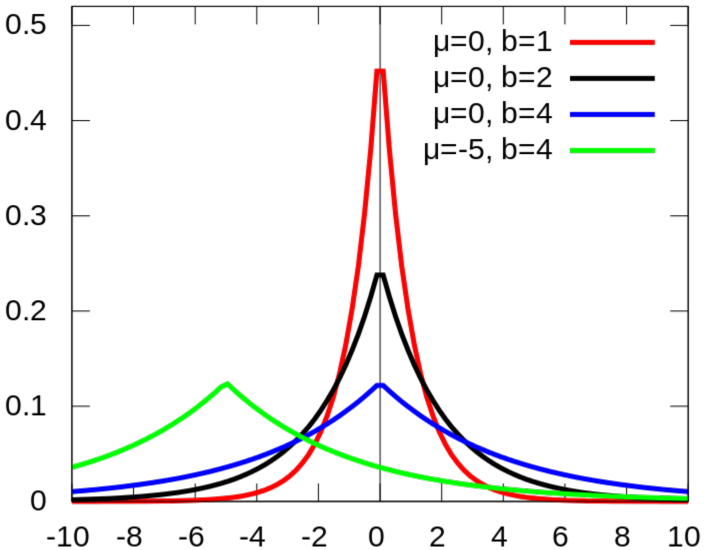
L2 - Regularization

Old gradient:

$$\boxed{\nabla l(\mathbf{w}) = \sum_{i=1}^{N} (y_i - \hat{y}_i)\mathbf{x}_i} \Rightarrow \boxed{\nabla l(\mathbf{w}) = \sum_{i=1}^{N} (y_i - \hat{y}_i)\mathbf{x}_i - \lambda \mathbf{w}}$$

Impact of prior

- Instead of using Gaussian prior, one can also consider other priors
- Laplace prior: $w_i \sim Laplace(0, b)$



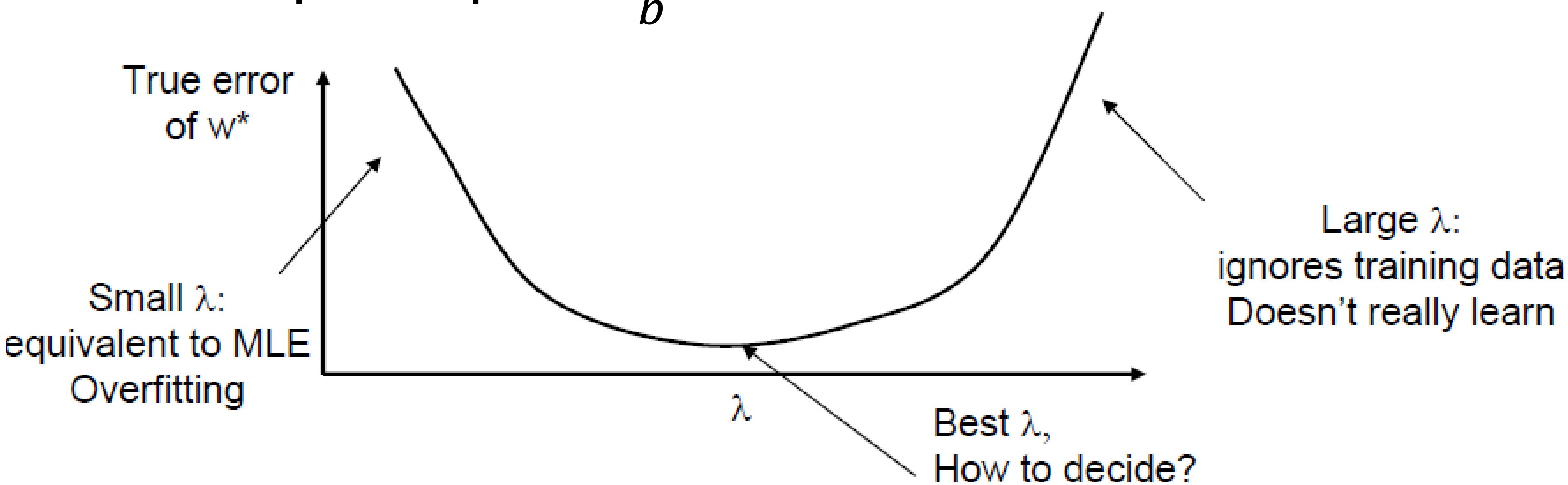
$$p(w_i) = \frac{1}{2b} \exp - \frac{|w_i|}{b}$$

- Lead to L1 regularization: $-\frac{1}{b} \sum_i |w_i|$

Impact of λ

- λ is inversely proportional to the variance of our prior belief
  - Gaussian prior: $\lambda = \frac{1}{\sigma^2}$
  - Laplace prior: $\frac{1}{b}$

At very low Lambda, we overfit as t



- Selecting appropriate λ is a model selection problem

Summary of Logistic Regression

- A popular discriminative classifier
- Learns conditional probability distribution $P(y \mid \mathbf{x})$
  - Defined by a logistic function
  - Produces a **linear** decision boundary
  - Nonlinear classifier by using basis functions
- Maximum likelihood estimation (MLE)
  - Gradient ascent bears interesting similarity with perceptron
  - Overfits for linearly separable case, regularization can help
  - Multi-class logistic regression: use the soft-max function
- Maximum posterior estimation (MAP)
  - Gaussian prior on the weights $= L_2$ regularization
  - Laplace prior = L_1 regularization
  - Overfitting controlled by the variance on the prior