

STOR566: Introduction to Deep Learning

Lecture 3: Linear regression and classification

Yao Li
UNC Chapel Hill

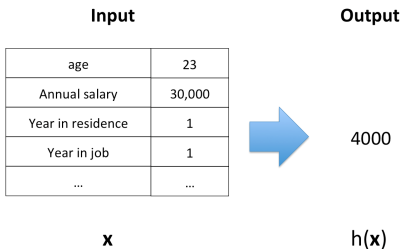
Aug 27, 2024

Materials are from *Learning from data* (Caltech) and *Deep Learning* (UCLA)

Linear Regression

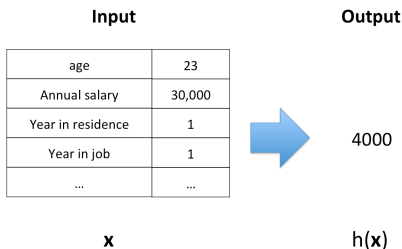
Regression

- Regression: predicting a real number



Regression

- Regression: predicting a real number



Linear Regression: $h(\mathbf{x}) = \sum_{i=0}^d w_i x_i = \mathbf{w}^T \mathbf{x}$

Problem definition

- Training data:

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N)$$

$\mathbf{x}_n \in \mathbb{R}^d$: feature vector for a sample

$y_n \in \mathbb{R}$: observed output (real number)

Problem definition

- Training data:

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N)$$

$\mathbf{x}_n \in \mathbb{R}^d$: feature vector for a sample

$y_n \in \mathbb{R}$: observed output (real number)

- Linear regression: find a function $h(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$ to approximate y

Problem definition

- Training data:

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N)$$

$\mathbf{x}_n \in \mathbb{R}^d$: feature vector for a sample

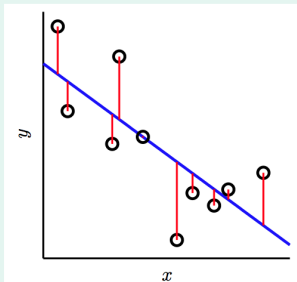
$y_n \in \mathbb{R}$: observed output (real number)

- Linear regression: find a function $h(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$ to approximate y
- Measure the error by $(h(\mathbf{x}) - y)^2$ (square error)

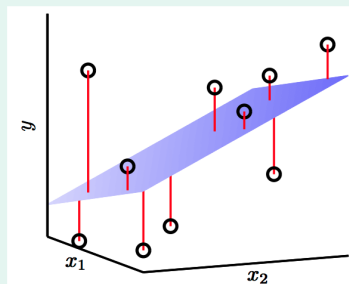
$$\text{Training error : } L_{\text{train}}(h) = \frac{1}{N} \sum_{n=1}^N (h(\mathbf{x}_n) - y_n)^2$$

Illustration

$$\mathbf{x} = (x) \in \mathbb{R}$$



$$\mathbf{x} = (x_1, x_2) \in \mathbb{R}^2$$



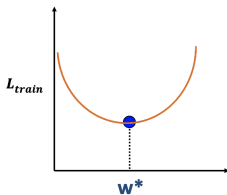
Linear regression: find **linear function** with small **residual**

Minimize L_{train}

$$\min_{\mathbf{w}} f(\mathbf{w}) = \|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2$$

- $\mathbf{X} \in \mathbb{R}^{N \times d}$, $\mathbf{y} \in \mathbb{R}^N$
- The objective function is continuous, differentiable, **convex**
- The optimal \mathbf{w}^* will satisfy:

$$\nabla f(\mathbf{w}^*) = \begin{bmatrix} \frac{\partial f}{\partial w_0}(\mathbf{w}^*) \\ \vdots \\ \frac{\partial f}{\partial w_d}(\mathbf{w}^*) \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$$



Minimizing f

$$f(\mathbf{w}) = \|X\mathbf{w} - \mathbf{y}\|^2 = \mathbf{w}^T X^T X \mathbf{w} - 2\mathbf{w}^T X^T \mathbf{y} + \mathbf{y}^T \mathbf{y}$$

$$\nabla f(\mathbf{w}) = ?$$

Minimizing f

$$\nabla f(\mathbf{w}^*) = 0 \Rightarrow \underbrace{X^T X \mathbf{w}^*}_{= X^T \mathbf{y}}$$

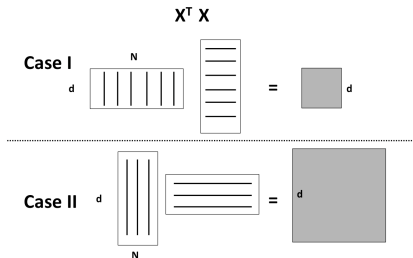
Minimizing f

$$\nabla f(\mathbf{w}^*) = 0 \Rightarrow \underbrace{X^T X \mathbf{w}^*}_{= X^T \mathbf{y}}$$

$$\Rightarrow \mathbf{w}^* = (X^T X)^{-1} X^T \mathbf{y} \quad ??$$

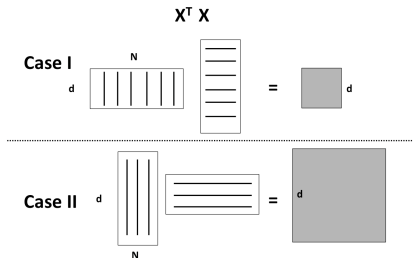
More on Linear Regression Solutions

- Case I: $X^T X$ is invertible \Rightarrow Unique solution
 - Often when $N > d$
 - $\mathbf{w}^* = (X^T X)^{-1} X^T \mathbf{y}$
- Case II: $X^T X$ is non-invertible \Rightarrow Many solutions
 - Often when $d > N$



More on Linear Regression Solutions

- Case I: $X^T X$ is invertible \Rightarrow Unique solution
 - Often when $N > d$
 - $\mathbf{w}^* = (X^T X)^{-1} X^T \mathbf{y}$
- Case II: $X^T X$ is non-invertible \Rightarrow Many solutions
 - Often when $d > N$



pseudo-inverse of $X^T X$

Logistic Regression

Binary Classification

- Input: training data $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n \in \mathbb{R}^d$ and corresponding outputs $y_1, y_2, \dots, y_n \in \{+1, -1\}$
- Training: compute a function f such that $\text{sign}(f(\mathbf{x}_i)) \approx y_i$ for all i
- Prediction: given a testing sample $\tilde{\mathbf{x}}$, predict the output as $\text{sign}(f(\tilde{\mathbf{x}}))$

Logistic Regression

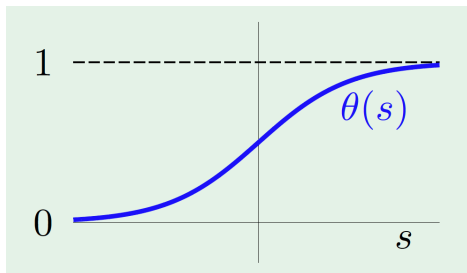
- Assume **linear** scoring function: $s = f(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$

Logistic Regression

- Assume **linear** scoring function: $s = f(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$
- **Logistic hypothesis**:

$$P(y = 1 \mid \mathbf{x}) = \theta(\mathbf{w}^T \mathbf{x}),$$

where $\theta(s) = \frac{e^s}{1+e^s} = \frac{1}{1+e^{-s}}$



Logistic Regression

- Assume **linear** scoring function: $s = f(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$
- Logistic hypothesis:

$$P(y = 1 \mid \mathbf{x}) = \theta(\mathbf{w}^T \mathbf{x}),$$

$$\text{where } \theta(s) = \frac{e^s}{1+e^s} = \frac{1}{1+e^{-s}}$$

- How about $P(y = -1 \mid \mathbf{x})$?

Logistic Regression

- Assume **linear** scoring function: $s = f(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$
- Logistic hypothesis:

$$P(y = 1 \mid \mathbf{x}) = \theta(\mathbf{w}^T \mathbf{x}),$$

$$\text{where } \theta(s) = \frac{e^s}{1+e^s} = \frac{1}{1+e^{-s}}$$

- How about $P(y = -1 \mid \mathbf{x})$?

$$P(y = -1 \mid \mathbf{x}) = 1 - \frac{1}{1+e^{-\mathbf{w}^T \mathbf{x}}} = \frac{1}{1+e^{\mathbf{w}^T \mathbf{x}}} = \theta(-\mathbf{w}^T \mathbf{x})$$

Logistic Regression

- Assume **linear** scoring function: $s = f(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$
- Logistic hypothesis:

$$P(y = 1 \mid \mathbf{x}) = \theta(\mathbf{w}^T \mathbf{x}),$$

$$\text{where } \theta(s) = \frac{e^s}{1+e^s} = \frac{1}{1+e^{-s}}$$

- How about $P(y = -1 \mid \mathbf{x})$?

$$P(y = -1 \mid \mathbf{x}) = 1 - \frac{1}{1+e^{-\mathbf{w}^T \mathbf{x}}} = \frac{1}{1+e^{\mathbf{w}^T \mathbf{x}}} = \theta(-\mathbf{w}^T \mathbf{x})$$

- Therefore, $P(y \mid \mathbf{x}) = \theta(y\mathbf{w}^T \mathbf{x})$

Maximizing the likelihood

- Likelihood of $\mathcal{D} = (\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)$:

$$\prod_{n=1}^N P(y_n \mid \mathbf{x}_n) = \prod_{n=1}^N \theta(y_n \mathbf{w}^T \mathbf{x}_n)$$

Maximizing the likelihood

- Likelihood of $\mathcal{D} = (\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)$:

$$\prod_{n=1}^N P(y_n \mid \mathbf{x}_n) = \prod_{n=1}^N \theta(y_n \mathbf{w}^T \mathbf{x}_n)$$

- Find \mathbf{w} to maximize the likelihood!

$$\begin{aligned} & \max_{\mathbf{w}} \prod_{n=1}^N \theta(y_n \mathbf{w}^T \mathbf{x}_n) \\ \Leftrightarrow & \max_{\mathbf{w}} \log(\prod_{n=1}^N \theta(y_n \mathbf{w}^T \mathbf{x}_n)) \\ \Leftrightarrow & \min_{\mathbf{w}} - \sum_{n=1}^N \log(\theta(y_n \mathbf{w}^T \mathbf{x}_n)) \\ \Leftrightarrow & \min_{\mathbf{w}} \sum_{n=1}^N \log(1 + e^{-y_n \mathbf{w}^T \mathbf{x}_n}) \end{aligned}$$

Empirical Risk Minimization (linear)

- Linear classification/regression:

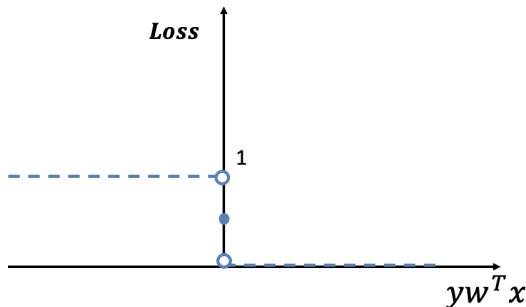
$$\min_{\mathbf{w}} \frac{1}{N} \sum_{n=1}^N \text{loss}(\underbrace{\mathbf{w}^T \mathbf{x}_n}_{\hat{y}_n: \text{the predicted score}}, y_n)$$

- Linear regression: $\text{loss}(h(\mathbf{x}_n), y_n) = (\mathbf{w}^T \mathbf{x}_n - y_n)^2$
- Logistic regression: $\text{loss}(h(\mathbf{x}_n), y_n) = \log(1 + e^{-y_n \mathbf{w}^T \mathbf{x}_n})$
- Hinge loss (SVM): $\text{loss}(h(\mathbf{x}_n), y_n) = \max(0, 1 - y_n \mathbf{w}^T \mathbf{x}_n)$



Binary Classification Loss

- Linear regression: $\text{loss}(h(\mathbf{x}_n), y_n) = (\mathbf{w}^T \mathbf{x}_n - y_n)^2$
- Logistic regression: $\text{loss}(h(\mathbf{x}_n), y_n) = \log(1 + e^{-y_n \mathbf{w}^T \mathbf{x}_n})$
- Hinge loss (SVM): $\text{loss}(h(\mathbf{x}_n), y_n) = \max(0, 1 - y_n \mathbf{w}^T \mathbf{x}_n)$



Empirical Risk Minimization (general)

- Assume $f_W(\mathbf{x})$ is the decision function to be learned (W is the parameters of the function)
- General empirical risk minimization:

$$\min_W \frac{1}{N} \sum_{n=1}^N \text{loss}(f_W(\mathbf{x}_n), y_n)$$

- Example: Neural network ($f_W(\cdot)$ is the network)

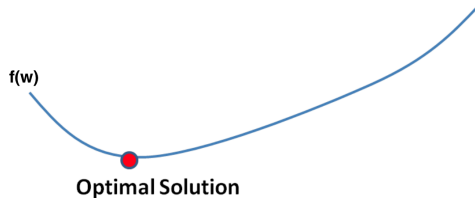
Gradient descent and SGD

Optimization

- Goal: find the minimizer of a function

$$\min_{\mathbf{w}} f(\mathbf{w})$$

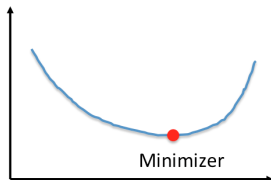
For now we assume f is twice differentiable



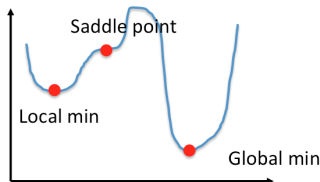
Convex vs Nonconvex

- Convex function:
 - $\nabla f(\mathbf{x}) = 0 \Leftrightarrow$ Global minimum
 - A function is convex if $\nabla^2 f(\mathbf{x})$ is positive definite
 - Example: linear regression, logistic regression, ...
- Non-convex function:
 - $\nabla f(\mathbf{x}) = 0 \Leftrightarrow$ Global min, local min, or saddle point
most algorithms only converge to gradient=0
 - Example: neural network, ...

Convex



Non-Convex



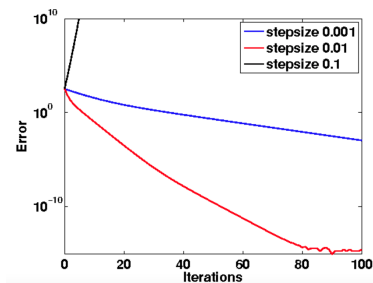
Gradient Descent

- Gradient descent: repeatedly do

$$\mathbf{w}^{t+1} \leftarrow \mathbf{w}^t - \alpha \nabla f(\mathbf{w}^t)$$

$\alpha > 0$ is the **step size**

- Step size too large \Rightarrow diverge; too small \Rightarrow slow convergence



Why gradient descent?

- At each iteration, form an approximation function of $f(\cdot)$:

$$f(\mathbf{w}^t + \mathbf{d}) \approx g(\mathbf{d}) := f(\mathbf{w}^t) + \nabla f(\mathbf{w}^t)^T \mathbf{d} + \frac{1}{2\alpha} \|\mathbf{d}\|^2$$

- Update solution by $\mathbf{w}^{t+1} \leftarrow \mathbf{w}^t + \mathbf{d}^*$

- $\mathbf{d}^* = \arg \min_{\mathbf{d}} g(\mathbf{d})$

$$\nabla g(\mathbf{d}^*) = 0 \Rightarrow \nabla f(\mathbf{w}^t) + \frac{1}{\alpha} \mathbf{d}^* = 0 \Rightarrow \mathbf{d}^* = -\alpha \nabla f(\mathbf{w}^t)$$

- \mathbf{d}^* will decrease $f(\cdot)$ if α (step size) is sufficiently small

Illustration of gradient descent

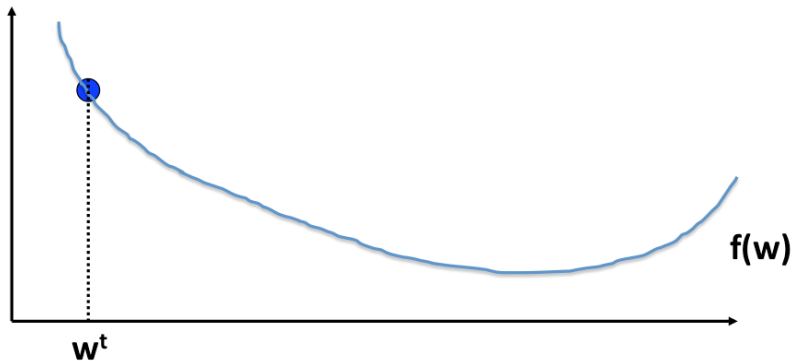
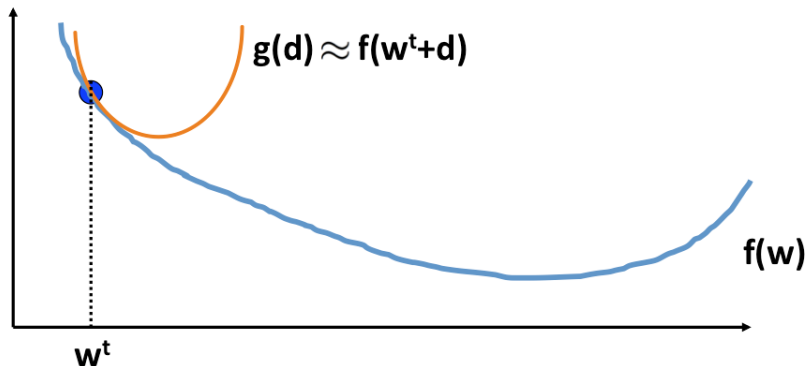


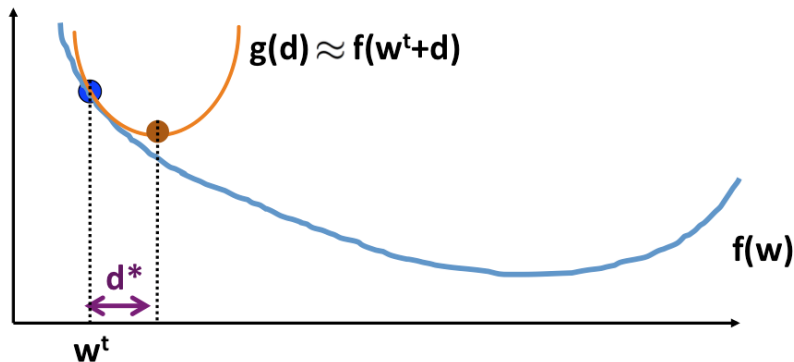
Illustration of gradient descent



Form a quadratic approximation

$$f(\mathbf{w}^t + \mathbf{d}) \approx g(\mathbf{d}) = f(\mathbf{w}^t) + \nabla f(\mathbf{w}^t)^T \mathbf{d} + \frac{1}{2\alpha} \|\mathbf{d}\|^2$$

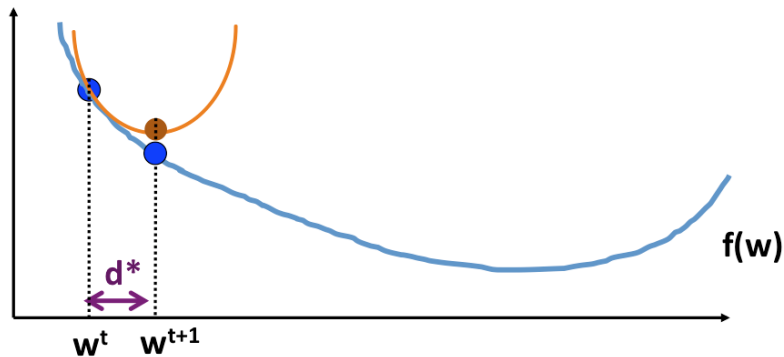
Illustration of gradient descent



Minimize $g(\mathbf{d})$:

$$\nabla g(\mathbf{d}^*) = 0 \Rightarrow \nabla f(\mathbf{w}^t) + \frac{1}{\alpha} \mathbf{d}^* = 0 \Rightarrow \mathbf{d}^* = -\alpha \nabla f(\mathbf{w}^t)$$

Illustration of gradient descent



Update w :

$$w^{t+1} = w^t + d^* = w^t - \alpha \nabla f(w^t)$$

Illustration of gradient descent

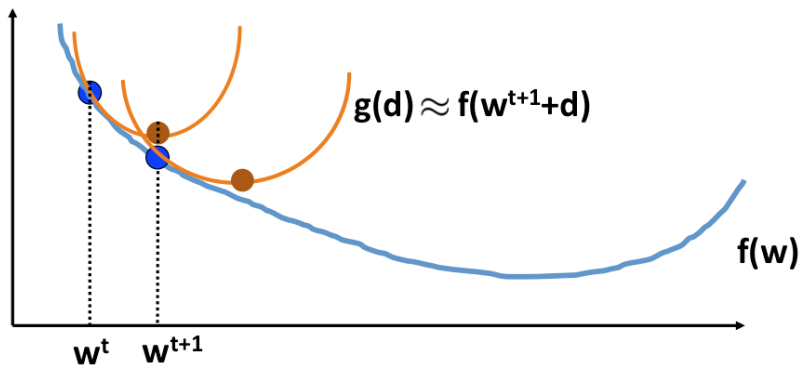
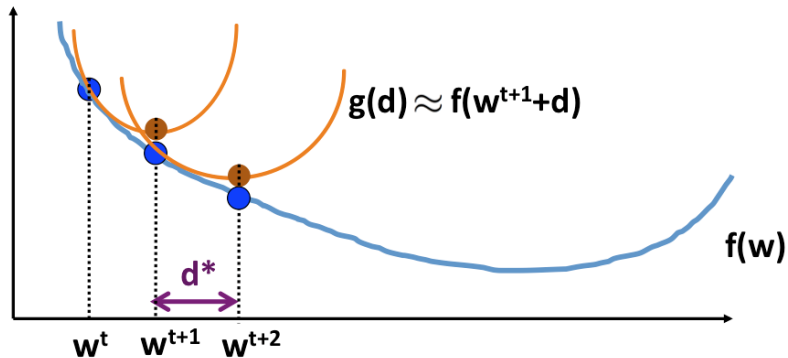


Illustration of gradient descent



Convergence

- Let L be a constant such that $\nabla^2 f(\mathbf{x}) \preceq LI$ for all \mathbf{x}
- **Theorem:** gradient descent converges if $\alpha < \frac{2}{L}$
- Optimal choice: $\alpha < \frac{1}{L}$
- In practice, we do not know $L \dots$

need to tune step size when running gradient descent

Applying to Logistic regression

gradient descent for logistic regression

- Initialize the weights \mathbf{w}_0
- For $t = 1, 2, \dots$
 - Compute the gradient

$$\nabla f(\mathbf{w}) = -\frac{1}{N} \sum_{n=1}^N \frac{y_n \mathbf{x}_n}{1 + e^{y_n \mathbf{w}^T \mathbf{x}_n}}$$

- Update the weights: $\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla f(\mathbf{w})$
- Return the final weights \mathbf{w}

Applying to Logistic regression

gradient descent for logistic regression

- Initialize the weights \mathbf{w}_0
- For $t = 1, 2, \dots$
 - Compute the gradient

$$\nabla f(\mathbf{w}) = -\frac{1}{N} \sum_{n=1}^N \frac{y_n \mathbf{x}_n}{1 + e^{y_n \mathbf{w}^T \mathbf{x}_n}}$$

- Update the weights: $\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla f(\mathbf{w})$
- Return the final weights \mathbf{w}

When to stop?

- Fixed number of iterations, or
- Stop when $\|\nabla f(\mathbf{w})\| < \epsilon$

Conclusions

- Linear regression:
 - Square loss \Rightarrow solving a linear system
 - Closed form solution
- Logistic regression:
 - A classification model based on a probability assumption
- Gradient descent: an iterative solver

Questions?