

Linear Models for Supervised Learning

Weinan Zhang

Shanghai Jiao Tong University

<http://wnzhang.net>

<http://wnzhang.net/teaching/cs420/index.html>

Discriminative Model and Generative Model

- Discriminative model

- modeling the dependence of unobserved variables on observed ones
- also called conditional models.
- Deterministic: $y = f_{\theta}(x)$
- Probabilistic: $p_{\theta}(y|x)$

- Generative model

- modeling the joint probabilistic distribution of data
- given some hidden parameters or variables

$$p_{\theta}(x, y)$$

- then do the conditional inference

$$p_{\theta}(y|x) = \frac{p_{\theta}(x, y)}{p_{\theta}(x)} = \frac{p_{\theta}(x, y)}{\sum_{y'} p_{\theta}(x, y')}$$

Discriminative Model and Generative Model

- Discriminative model
 - modeling the dependence of unobserved variables on observed ones
 - also called conditional models.
 - Deterministic: $y = f_{\theta}(x)$
 - Probabilistic: $p_{\theta}(y|x)$
- Directly model the dependence for label prediction
- Easy to define dependence specific features and models
- Practically yielding higher prediction performance
- Linear regression, logistic regression, k nearest neighbor, SVMs, (multi-layer) perceptrons, decision trees, random forest etc.

Discriminative Model and Generative Model

- Generative model

- modeling the joint probabilistic distribution of data
- given some hidden parameters or variables

$$p_{\theta}(x, y)$$

- then do the conditional inference

$$p_{\theta}(y|x) = \frac{p_{\theta}(x, y)}{p_{\theta}(x)} = \frac{p_{\theta}(x, y)}{\sum_{y'} p_{\theta}(x, y')}$$

- Recover the data distribution [essence of data science]
- Benefit from hidden variables modeling
- Naive Bayes, Hidden Markov Model, Mixture Gaussian, Markov Random Fields, Latent Dirichlet Allocation etc.

Linear Regression

Linear Discriminative Models

- Discriminative model
 - modeling the dependence of unobserved variables on observed ones
 - also called conditional models.
 - **Deterministic:** $y = f_{\theta}(x)$
 - Probabilistic: $p_{\theta}(y|x)$
- Focus of this course
 - Linear regression model
 - Linear classification model

Linear Discriminative Models

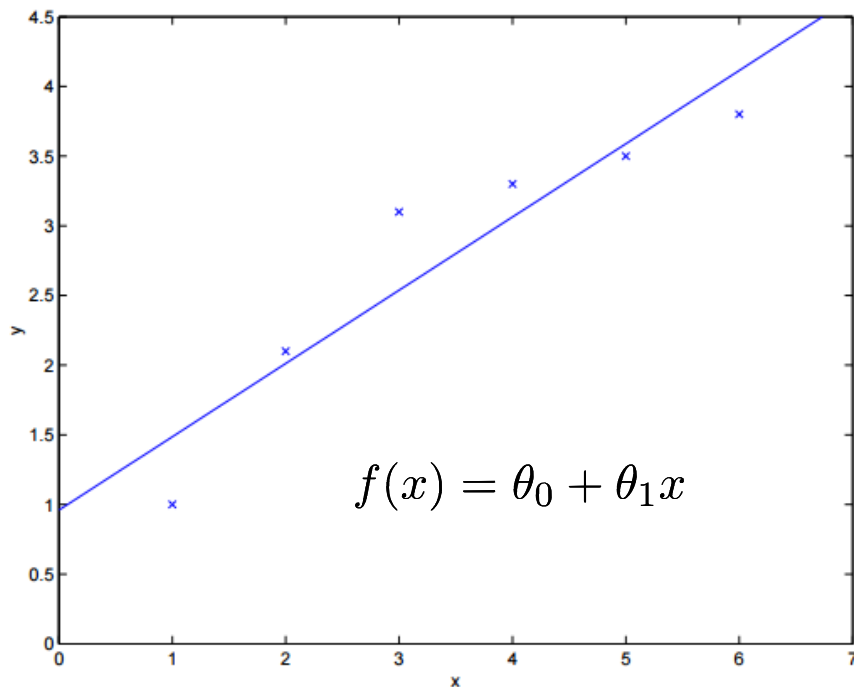
- Discriminative model
 - modeling the dependence of unobserved variables on observed ones
 - also called conditional models.
 - **Deterministic:** $y = f_{\theta}(x)$
 - Probabilistic: $p_{\theta}(y|x)$
- Linear regression model

$$y = f_{\theta}(x) = \theta_0 + \sum_{j=1}^d \theta_j x_j = \theta^{\top} x$$

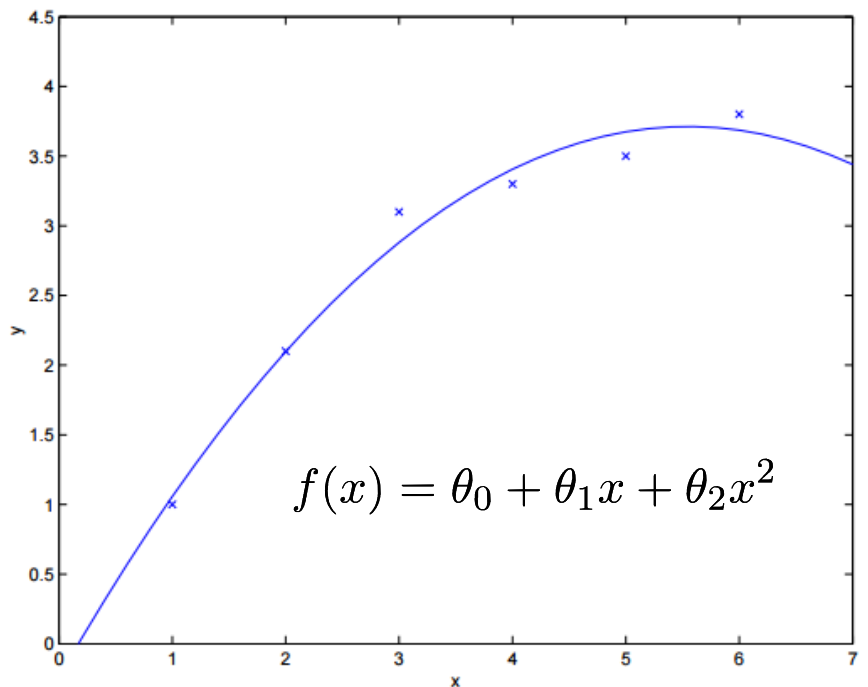
$$x = (1, x_1, x_2, \dots, x_d)$$

Linear Regression

- One-dimensional linear & quadratic regression



Linear Regression

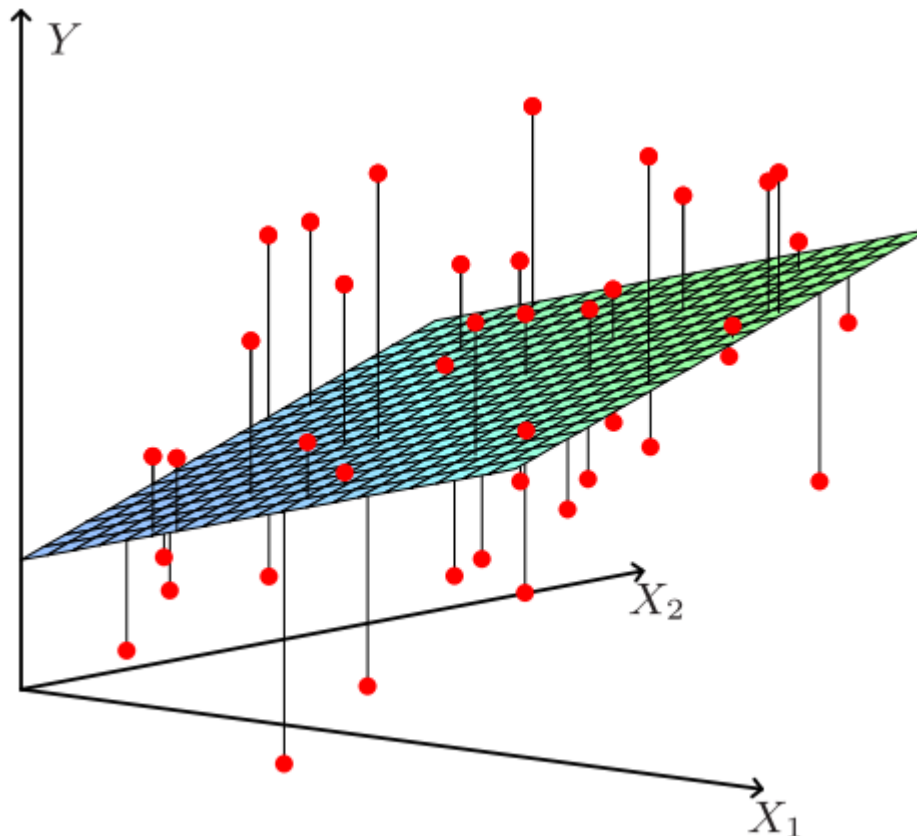


Quadratic Regression
(A kind of generalized
linear model)

Linear Regression

- Two-dimensional linear regression

$$f(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$



Learning Objective

- Make the prediction close to the corresponding label

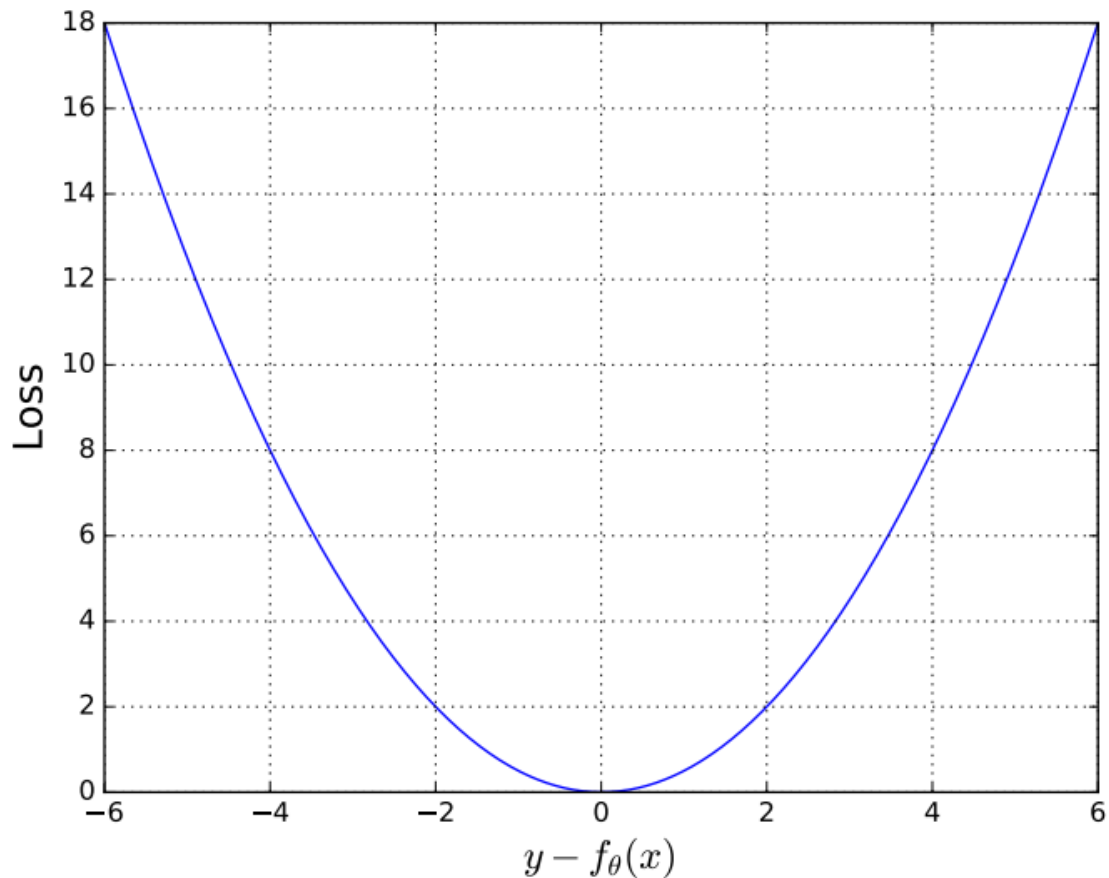
$$\min_{\theta} \frac{1}{N} \sum_{i=1}^N \mathcal{L}(y_i, f_{\theta}(x_i))$$

- Loss function $\mathcal{L}(y_i, f_{\theta}(x_i))$ measures the error between the label and prediction
- The definition of loss function depends on the data and task
- Most popular loss function: squared loss

$$\mathcal{L}(y_i, f_{\theta}(x_i)) = (y_i - f_{\theta}(x_i))^2$$

Squared Loss

$$\mathcal{L}(y_i, f_{\theta}(x_i)) = \frac{1}{2}(y_i - f_{\theta}(x_i))^2$$

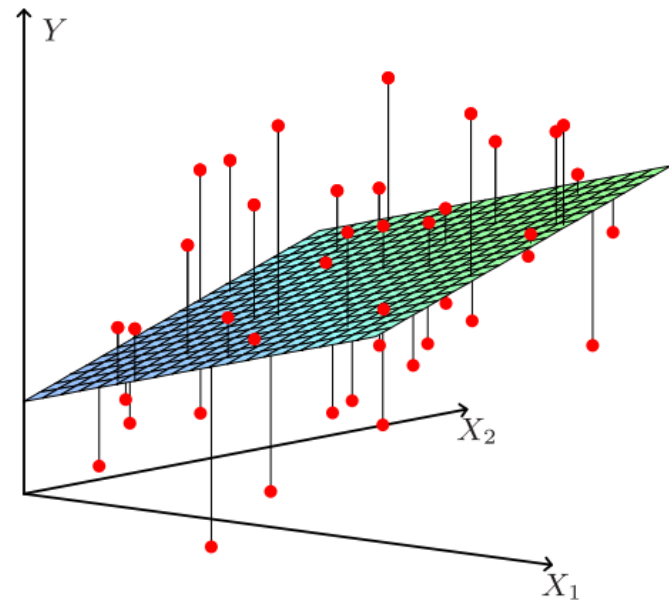
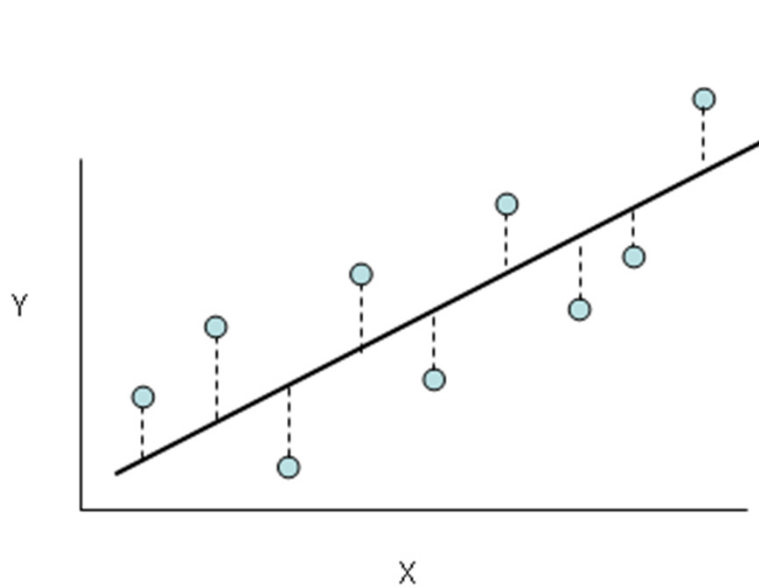


- Penalty much more on larger distances
- Accept small distance (error)
 - Observation noise etc.
 - Generalization

Least Square Linear Regression

- Objective function to minimize

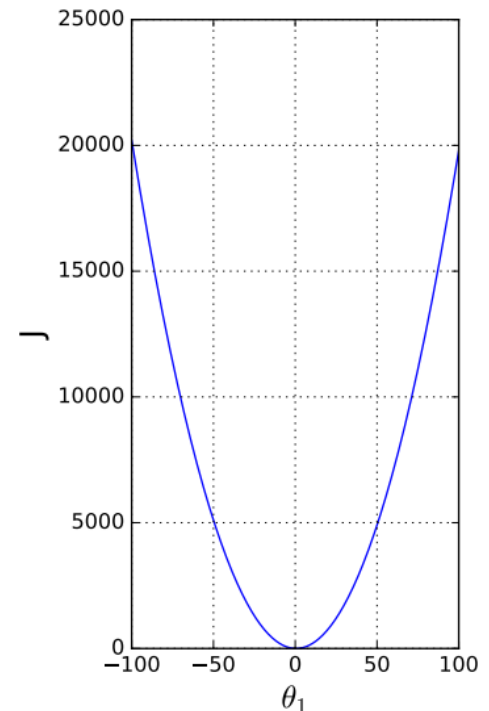
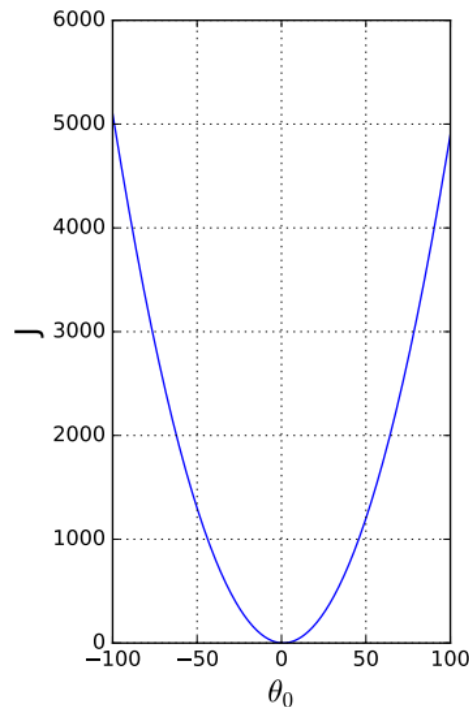
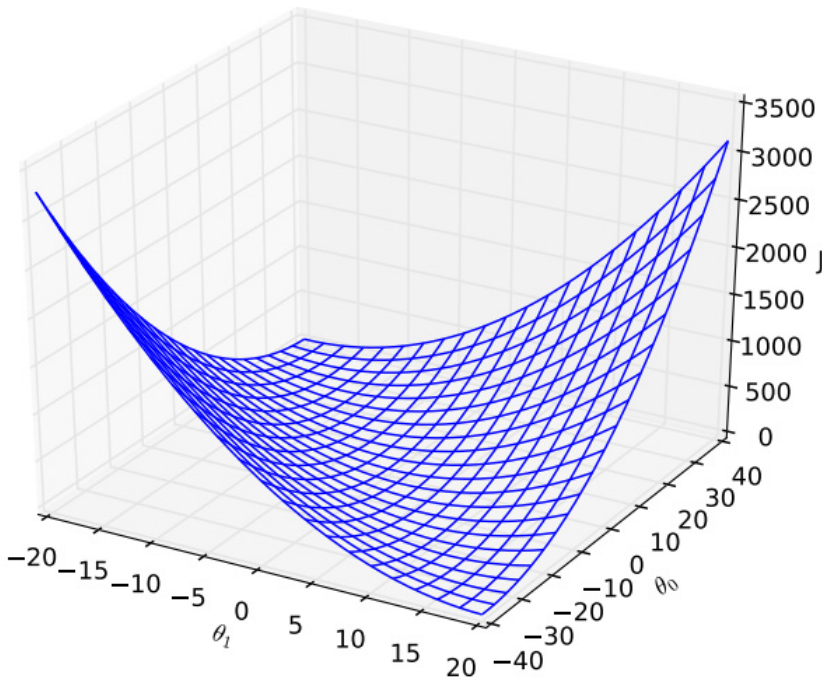
$$J_{\theta} = \frac{1}{2N} \sum_{i=1}^N (y_i - f_{\theta}(x_i))^2 \quad \min_{\theta} J_{\theta}$$



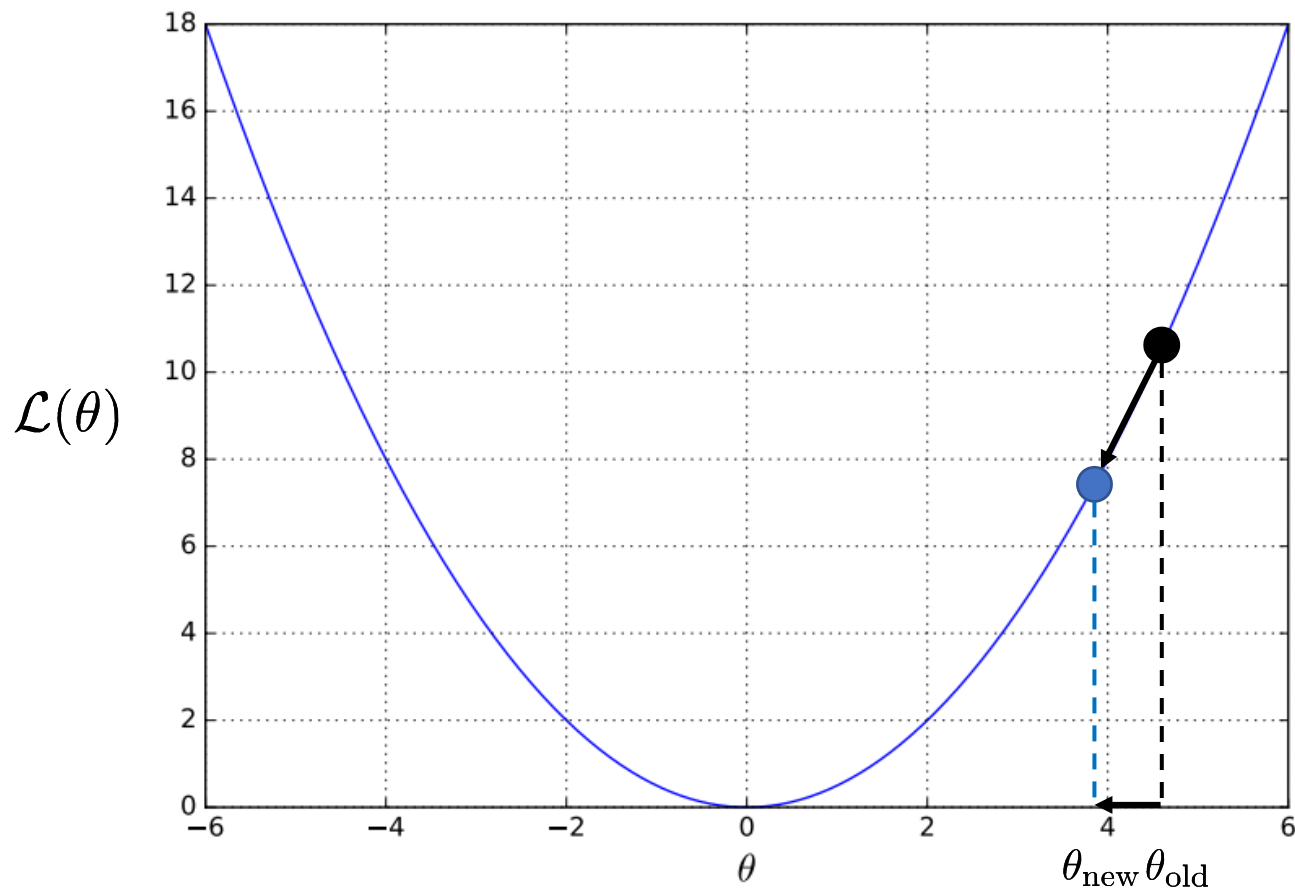
Minimize the Objective Function

- Let $N=1$ for a simple case, for $(x,y)=(2,1)$

$$J(\theta) = \frac{1}{2}(y - \theta_0 - \theta_1 x)^2 = \frac{1}{2}(1 - \theta_0 - 2\theta_1)^2$$



Gradient Learning Methods



$$\theta_{\text{new}} \leftarrow \theta_{\text{old}} - \eta \frac{\partial \mathcal{L}(\theta)}{\partial \theta}$$

Batch Gradient Descent

$$J(\theta) = \frac{1}{2N} \sum_{i=1}^N (y_i - f_{\theta}(x_i))^2 \quad \min_{\theta} J(\theta)$$

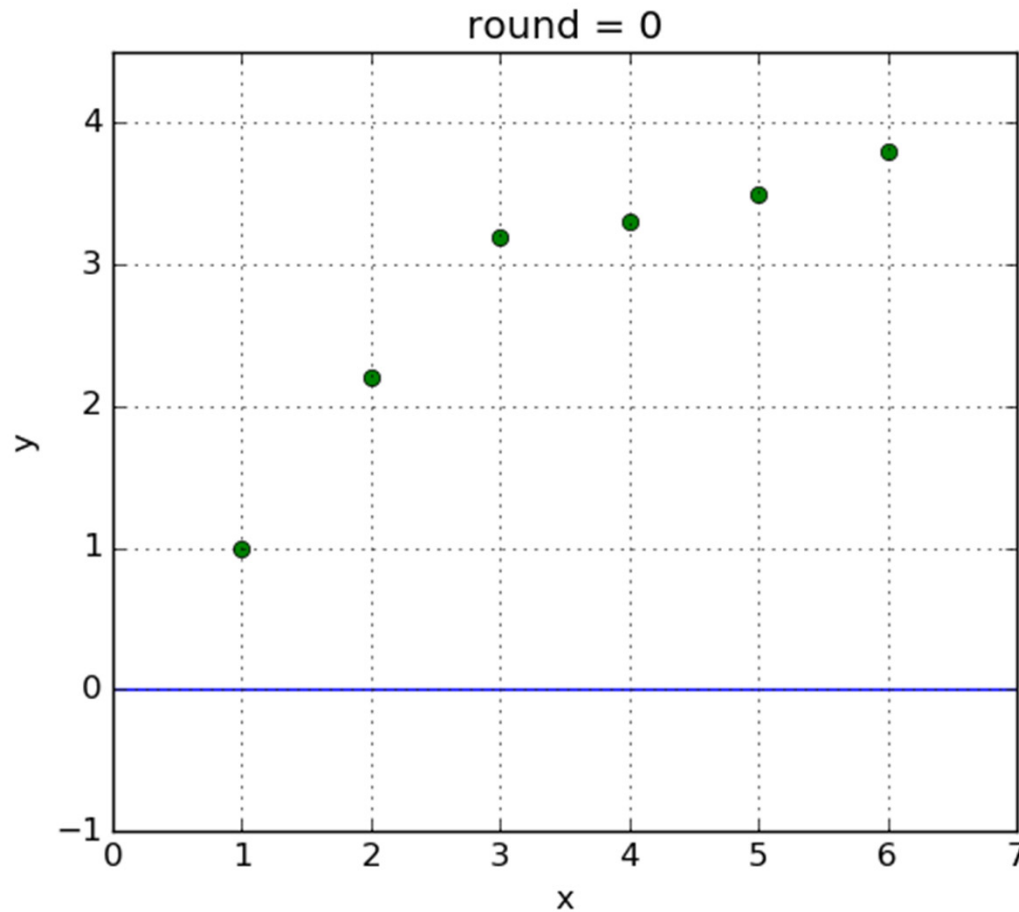
- Update $\theta_{\text{new}} \leftarrow \theta_{\text{old}} - \eta \frac{\partial J(\theta)}{\partial \theta}$ for the whole batch

$$\frac{\partial J(\theta)}{\partial \theta} = -\frac{1}{N} \sum_{i=1}^N (y_i - f_{\theta}(x_i)) \frac{\partial f_{\theta}(x_i)}{\partial \theta}$$

$$= -\frac{1}{N} \sum_{i=1}^N (y_i - f_{\theta}(x_i)) x_i$$

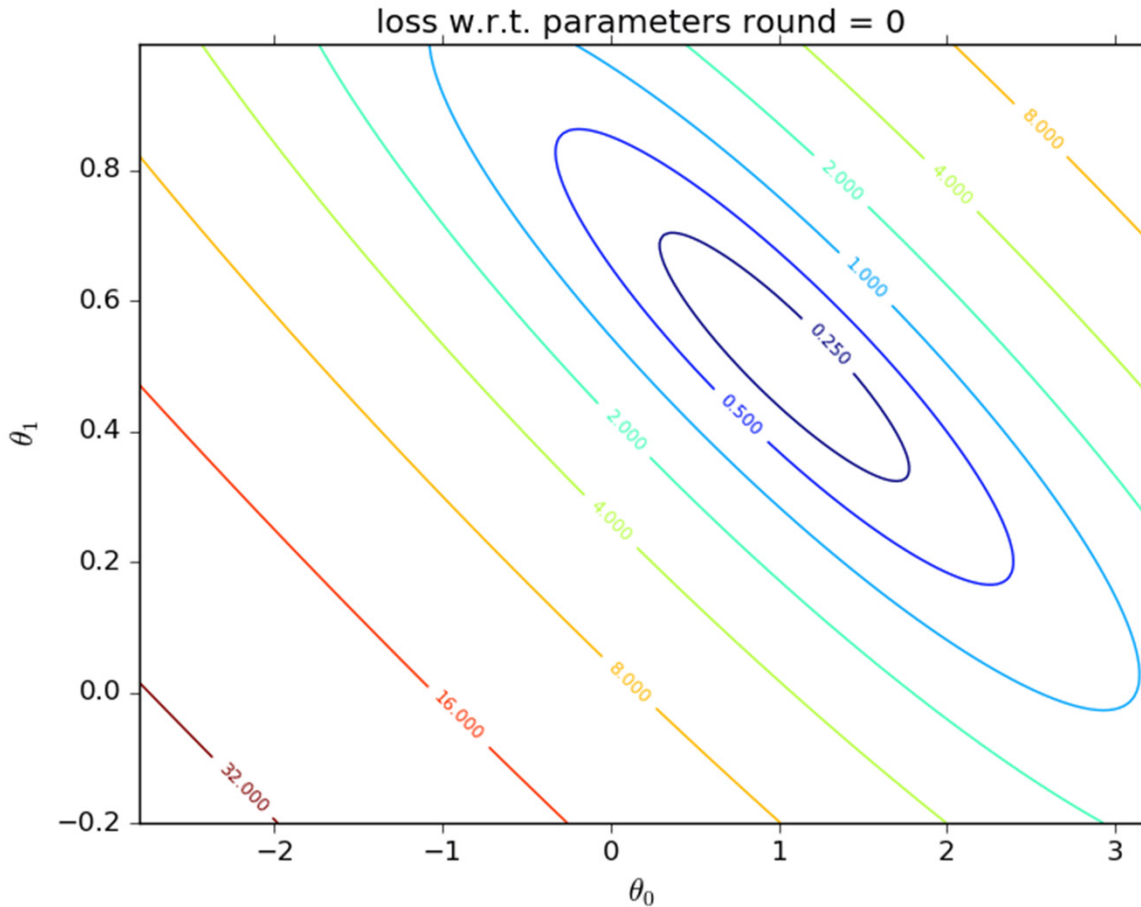
$$\theta_{\text{new}} = \theta_{\text{old}} + \eta \frac{1}{N} \sum_{i=1}^N (y_i - f_{\theta}(x_i)) x_i$$

Learning Linear Model - Curve



$$f(x) = \theta_0 + \theta_1 x$$

Learning Linear Model - Weights



Stochastic Gradient Descent

$$J^{(i)}(\theta) = \frac{1}{2}(y_i - f_{\theta}(x_i))^2 \quad \min_{\theta} \frac{1}{N} \sum_i J^{(i)}(\theta)$$

- Update $\theta_{\text{new}} = \theta_{\text{old}} - \eta \frac{\partial J^{(i)}(\theta)}{\partial \theta}$ for every single instance

$$\frac{\partial J^{(i)}(\theta)}{\partial \theta} = -(y_i - f_{\theta}(x_i)) \frac{\partial f_{\theta}(x_i)}{\partial \theta}$$

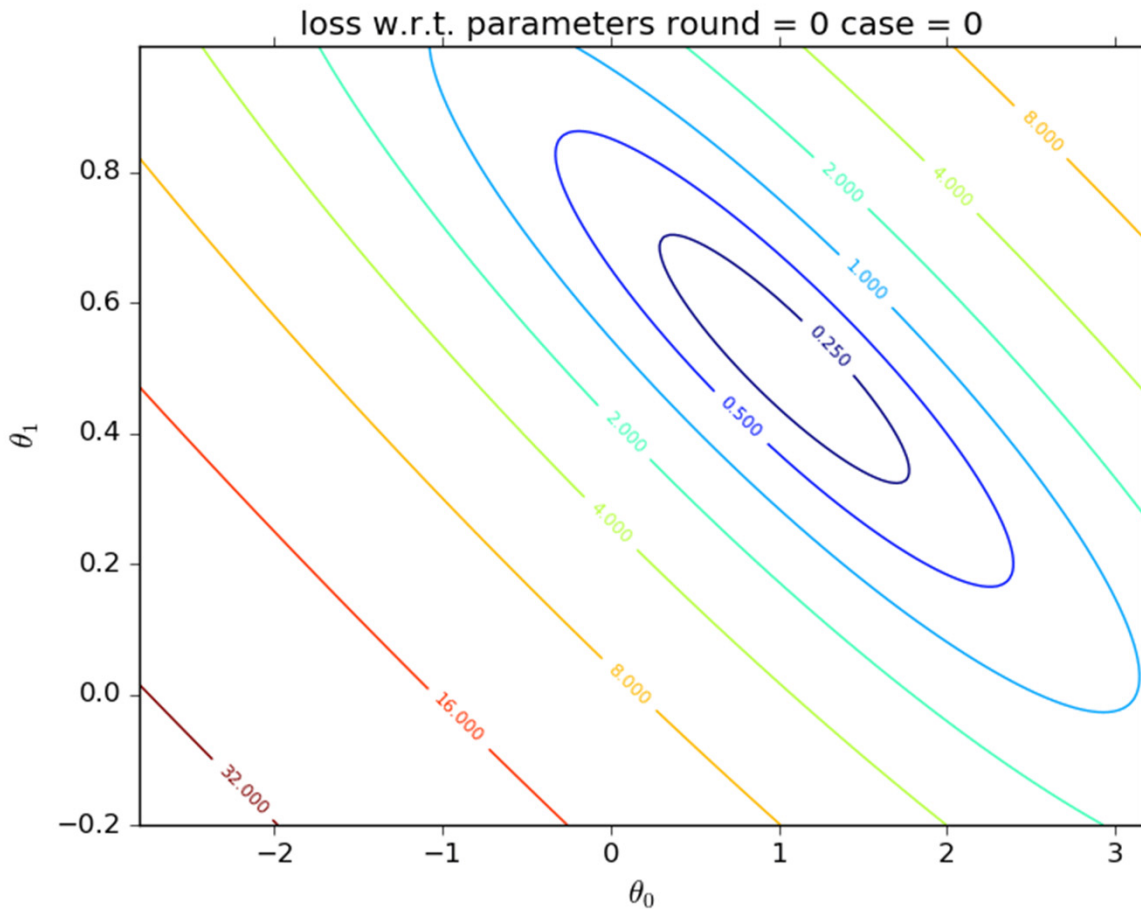
$$= -(y_i - f_{\theta}(x_i))x_i$$

$$\theta_{\text{new}} = \theta_{\text{old}} + \eta(y_i - f_{\theta}(x_i))x_i$$

- Compare with BGD

- Faster learning
- Uncertainty or fluctuation in learning

Linear Classification Model



Mini-Batch Gradient Descent

- A combination of batch GD and stochastic GD
- Split the whole dataset into K mini-batches

$$\{1, 2, 3, \dots, K\}$$

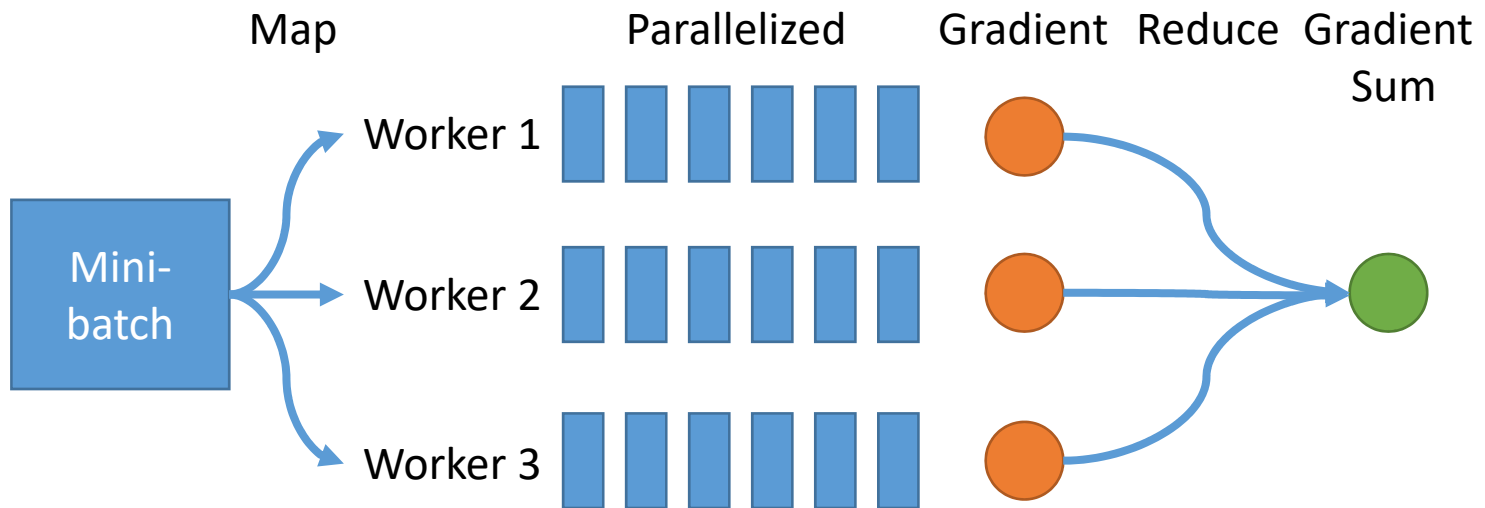
- For each mini-batch k , perform one-step BGD towards minimizing

$$J^{(k)}(\theta) = \frac{1}{2N_k} \sum_{i=1}^{N_k} (y_i - f_{\theta}(x_i))^2$$

- Update $\theta_{\text{new}} = \theta_{\text{old}} - \eta \frac{\partial J^{(k)}(\theta)}{\partial \theta}$ for each mini-batch

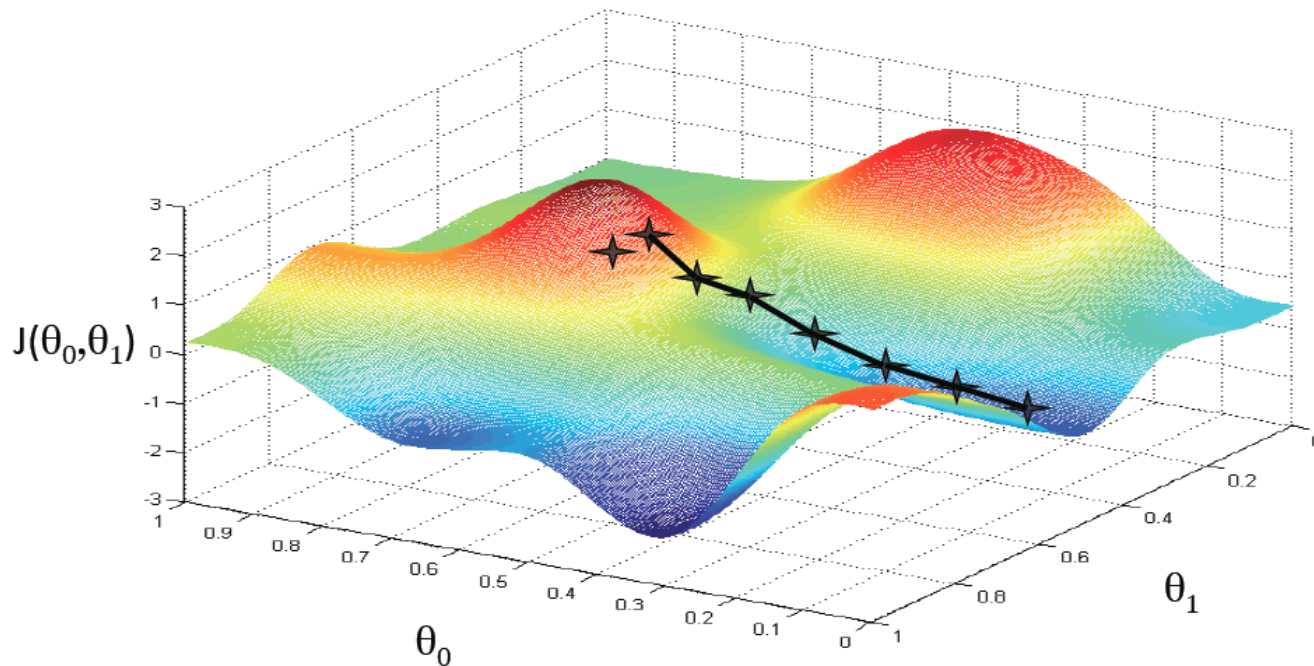
Mini-Batch Gradient Descent

- Good learning stability (BGD)
- Good convergence rate (SGD)
- Easy to be parallelized
 - Parallelization within a mini-batch



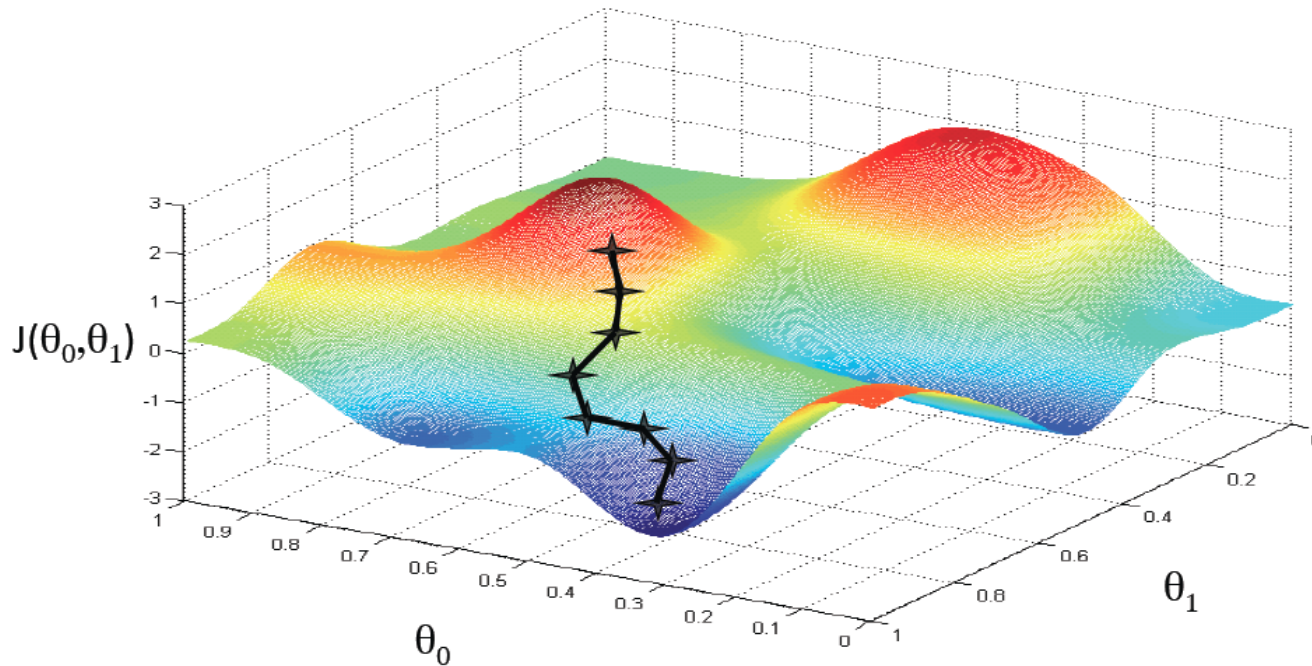
Basic Search Procedure

- Choose an initial value for θ
- Update θ iteratively with the data
- Until we reach a minimum

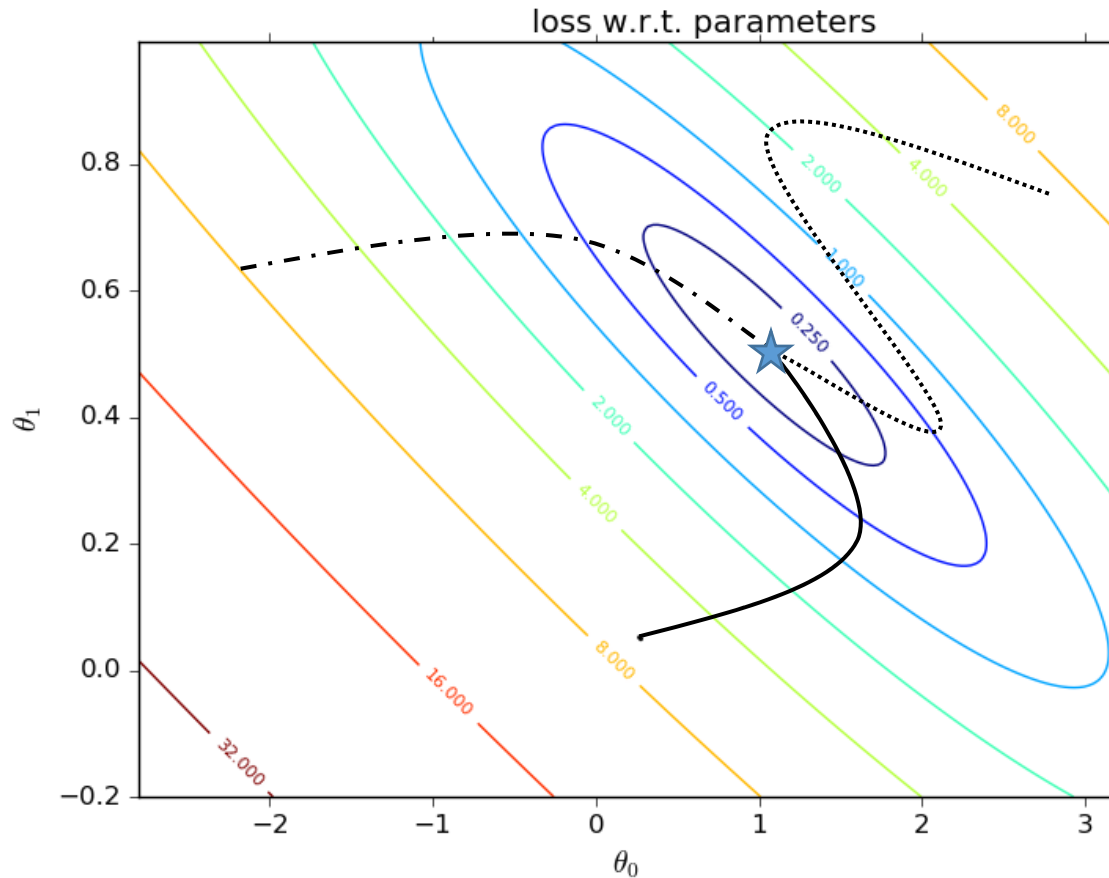


Basic Search Procedure

- Choose a new initial value for θ
- Update θ iteratively with the data
- Until we reach a minimum



Unique Minimum for Convex Objective



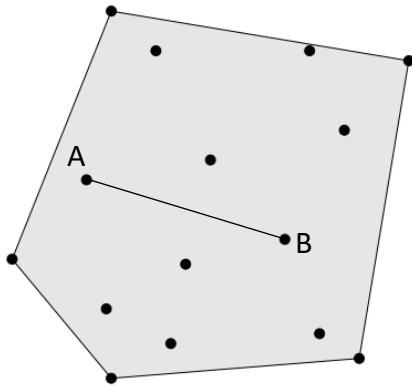
- Different initial parameters and different learning algorithm lead to the same optimum

Convex Set

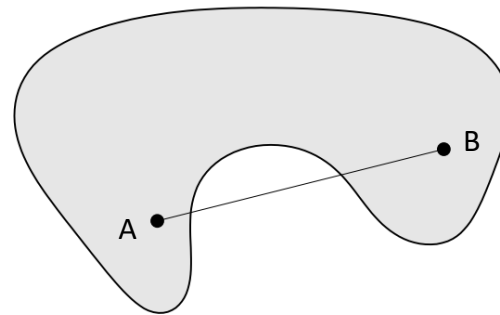
- A convex set S is a set of points such that, given any two points A, B in that set, the line AB joining them lies entirely within S .

$$tx_1 + (1 - t)x_2 \in S$$

for all $x_1, x_2 \in S, 0 \leq t \leq 1$

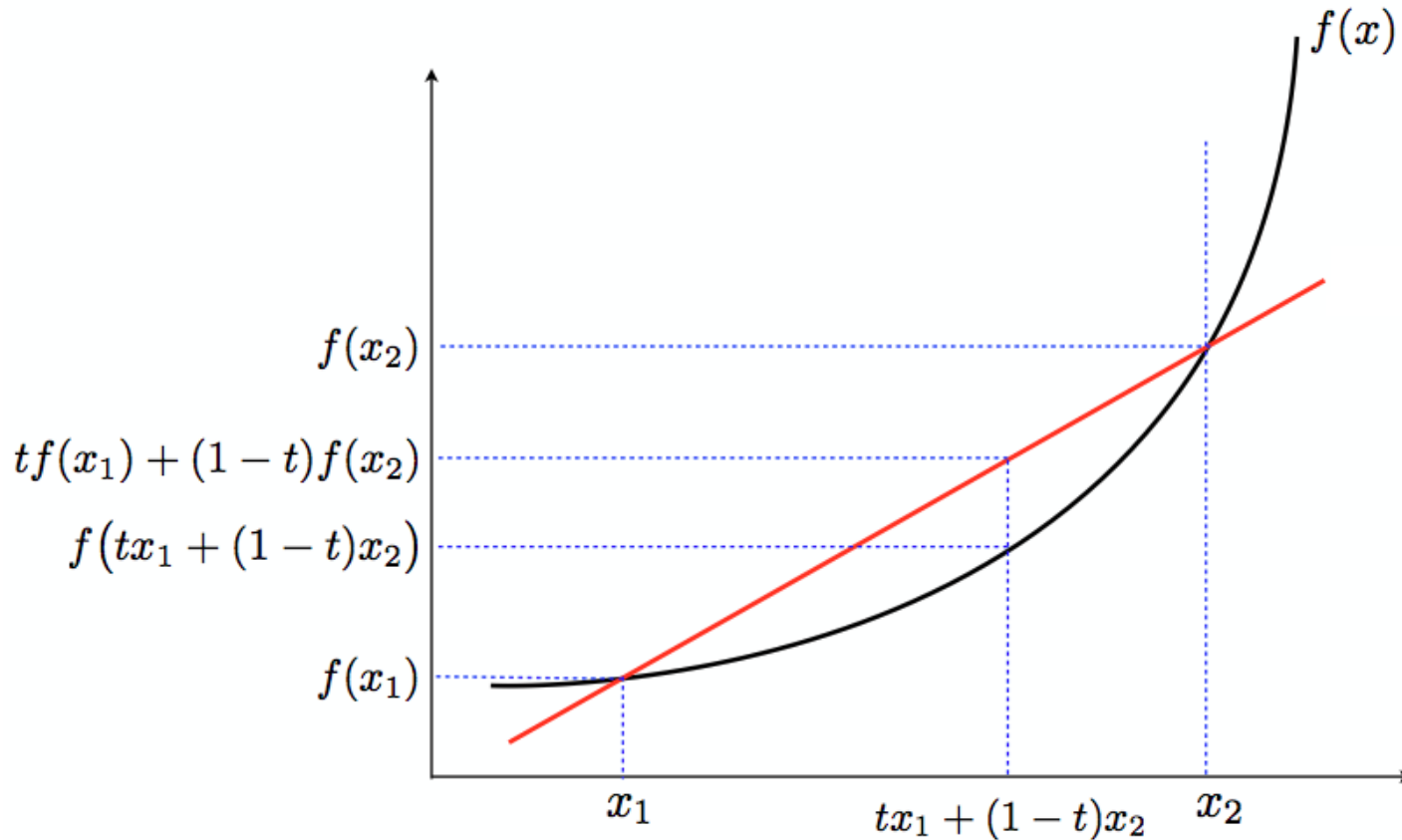


Convex set



Non-convex set

Convex Function



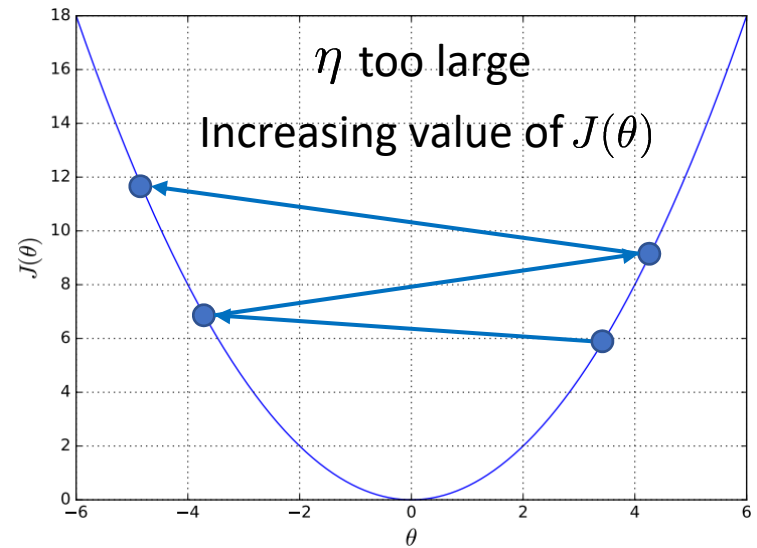
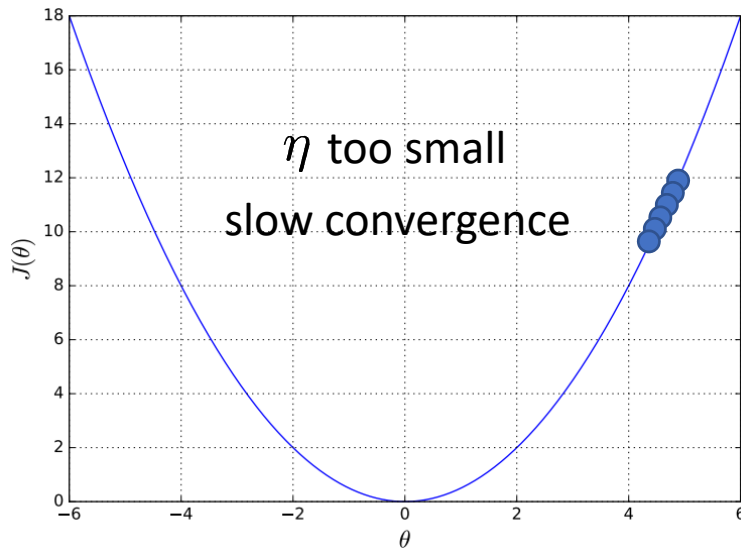
$f : \mathbb{R}^n \rightarrow \mathbb{R}$ is convex if **dom** f is a convex set and

$$f(tx_1 + (1-t)x_2) \leq tf(x_1) + (1-t)f(x_2)$$

for all $x_1, x_2 \in \mathbf{dom} f, 0 \leq t \leq 1$

Choosing Learning Rate

$$\theta_{\text{new}} = \theta_{\text{old}} - \eta \frac{\partial J(\theta)}{\partial \theta}$$



- The initial point may be too far away from the optimal solution, which takes much time to converge
- To see if gradient descent is working, print out $J(\theta)$ for each or every several iterations. If $J(\theta)$ does not drop properly, adjust η
- May overshoot the minimum
- May fail to converge
- May even diverge

Algebra Perspective

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}^{(1)} \\ \mathbf{x}^{(2)} \\ \vdots \\ \mathbf{x}^{(n)} \end{bmatrix} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & x_3^{(1)} & \dots & x_d^{(1)} \\ x_1^{(2)} & x_2^{(2)} & x_3^{(2)} & \dots & x_d^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_1^{(n)} & x_2^{(n)} & x_3^{(n)} & \dots & x_d^{(n)} \end{bmatrix} \quad \boldsymbol{\theta} = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_d \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

- Prediction $\hat{\mathbf{y}} = \mathbf{X}\boldsymbol{\theta} = \begin{bmatrix} \mathbf{x}^{(1)}\boldsymbol{\theta} \\ \mathbf{x}^{(2)}\boldsymbol{\theta} \\ \vdots \\ \mathbf{x}^{(n)}\boldsymbol{\theta} \end{bmatrix}$

- Objective $J(\boldsymbol{\theta}) = \frac{1}{2}(\mathbf{y} - \hat{\mathbf{y}})^\top (\mathbf{y} - \hat{\mathbf{y}}) = \frac{1}{2}(\mathbf{y} - \mathbf{X}\boldsymbol{\theta})^\top (\mathbf{y} - \mathbf{X}\boldsymbol{\theta})$

Matrix Form

- Objective

$$J(\boldsymbol{\theta}) = \frac{1}{2}(\mathbf{y} - \mathbf{X}\boldsymbol{\theta})^\top (\mathbf{y} - \mathbf{X}\boldsymbol{\theta}) \quad \min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$

- Gradient

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = -\mathbf{X}^\top (\mathbf{y} - \mathbf{X}\boldsymbol{\theta})$$

- Solution $\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \mathbf{0} \Rightarrow \mathbf{X}^\top (\mathbf{y} - \mathbf{X}\boldsymbol{\theta}) = \mathbf{0}$

$$\Rightarrow \mathbf{X}^\top \mathbf{y} = \mathbf{X}^\top \mathbf{X} \boldsymbol{\theta}$$

$$\Rightarrow \hat{\boldsymbol{\theta}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$$

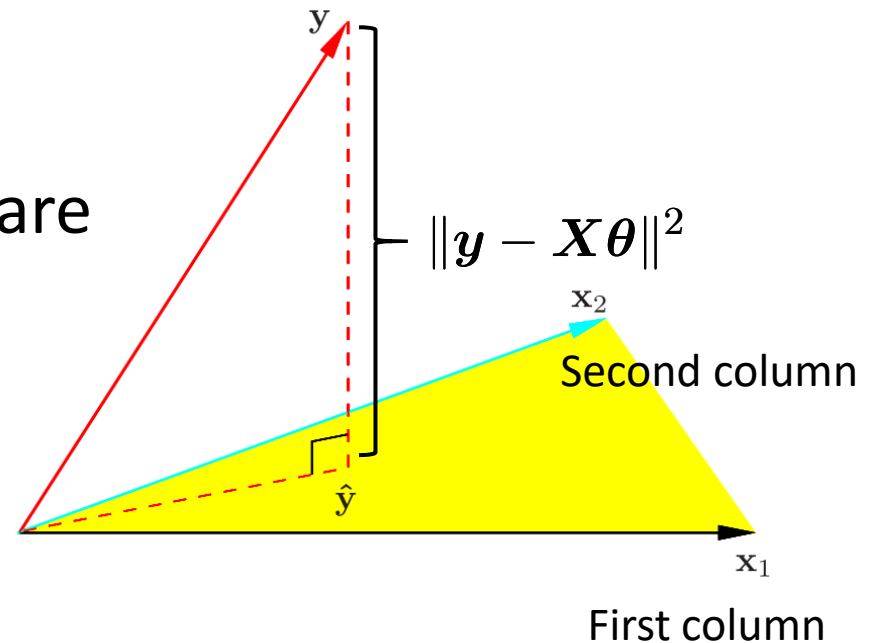
Matrix Form

- Then the predicted values are

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

$$= \mathbf{H} \mathbf{y}$$

H : hat matrix



- Geometrical Explanation

- The column vectors $[\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_d]$ form a subspace of \mathbb{R}^n
- H is a least square projection

$$\mathbf{X} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & x_3^{(1)} & \dots & x_d^{(1)} \\ x_1^{(2)} & x_2^{(2)} & x_3^{(2)} & \dots & x_d^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_1^{(n)} & x_2^{(n)} & x_3^{(n)} & \dots & x_d^{(n)} \end{bmatrix} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_d] \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

$\mathbf{X}^\top \mathbf{X}$ Might be Singular

- When some column vectors are not independent
 - For example, $\mathbf{x}_2 = 3\mathbf{x}_1$
- then $\mathbf{X}^\top \mathbf{X}$ is singular, thus $\hat{\boldsymbol{\theta}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$ cannot be directly calculated.

- Solution: regularization

$$J(\boldsymbol{\theta}) = \frac{1}{2}(\mathbf{y} - \mathbf{X}\boldsymbol{\theta})^\top (\mathbf{y} - \mathbf{X}\boldsymbol{\theta}) + \frac{\lambda}{2} \|\boldsymbol{\theta}\|_2^2$$

Matrix Form with Regularization

- Objective

$$J(\boldsymbol{\theta}) = \frac{1}{2}(\mathbf{y} - \mathbf{X}\boldsymbol{\theta})^\top (\mathbf{y} - \mathbf{X}\boldsymbol{\theta}) + \frac{\lambda}{2}\|\boldsymbol{\theta}\|_2^2 \quad \min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$

- Gradient

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = -\mathbf{X}^\top (\mathbf{y} - \mathbf{X}\boldsymbol{\theta}) + \lambda \boldsymbol{\theta}$$

- Solution

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \mathbf{0} \rightarrow -\mathbf{X}^\top (\mathbf{y} - \mathbf{X}\boldsymbol{\theta}) + \lambda \boldsymbol{\theta} = \mathbf{0}$$

$$\rightarrow \mathbf{X}^\top \mathbf{y} = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I}) \boldsymbol{\theta}$$

$$\rightarrow \hat{\boldsymbol{\theta}} = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{y}$$

Linear Discriminative Models

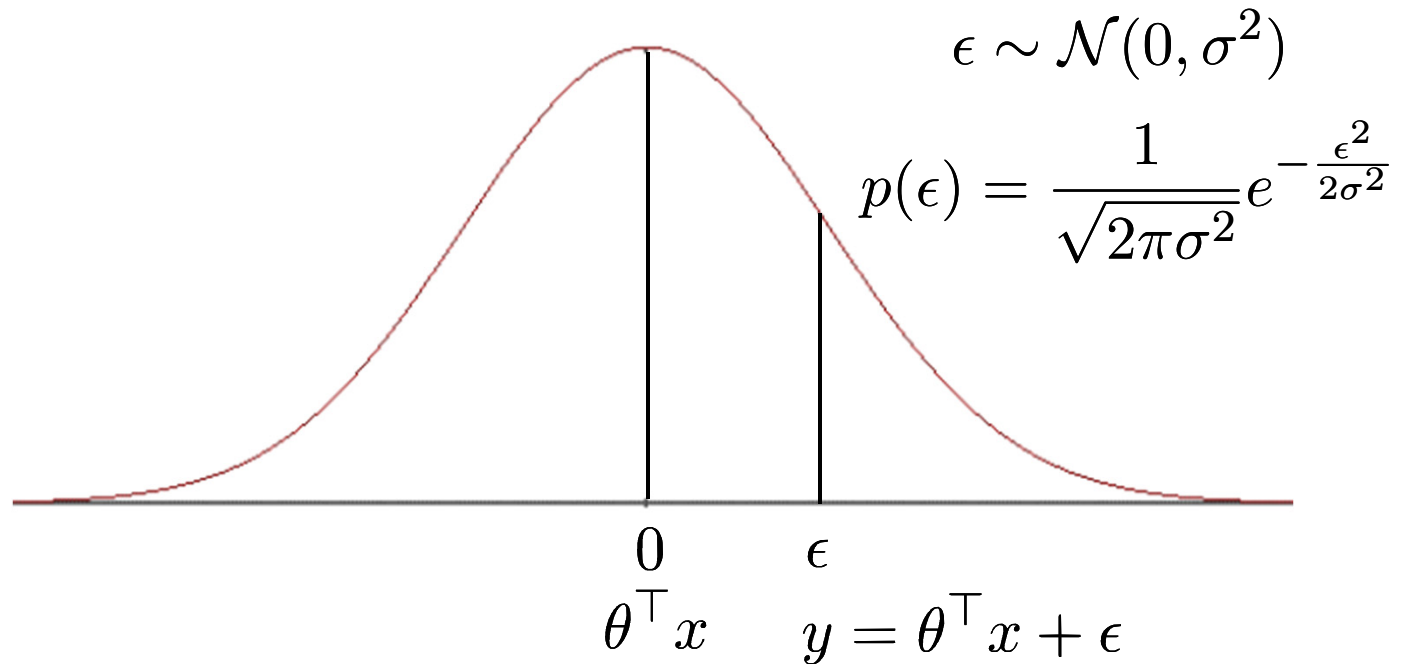
- Discriminative model
 - modeling the dependence of unobserved variables on observed ones
 - also called conditional models.
 - Deterministic: $y = f_{\theta}(x)$
 - **Probabilistic:** $p_{\theta}(y|x)$
- Linear regression with Gaussian noise model

$$y = f_{\theta}(x) + \epsilon = \theta_0 + \sum_{j=1}^d \theta_j x_j + \epsilon = \theta^{\top} x + \epsilon$$

$$\epsilon \sim \mathcal{N}(0, \sigma^2)$$

$$x = (1, x_1, x_2, \dots, x_d)$$

Objective: Likelihood



- Data likelihood

$$p(y|x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y - \theta^\top x)^2}{2\sigma^2}}$$

Learning

- Maximize the data likelihood

$$\max_{\theta} \prod_{i=1}^N \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_i - \theta^\top x_i)^2}{2\sigma^2}}$$

- Maximize the data log-likelihood

$$\begin{aligned} \log \prod_{i=1}^N \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_i - \theta^\top x_i)^2}{2\sigma^2}} &= \sum_{i=1}^N \log \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_i - \theta^\top x_i)^2}{2\sigma^2}} \\ &= -\sum_{i=1}^N \frac{(y_i - \theta^\top x_i)^2}{2\sigma^2} + \text{const} \end{aligned}$$

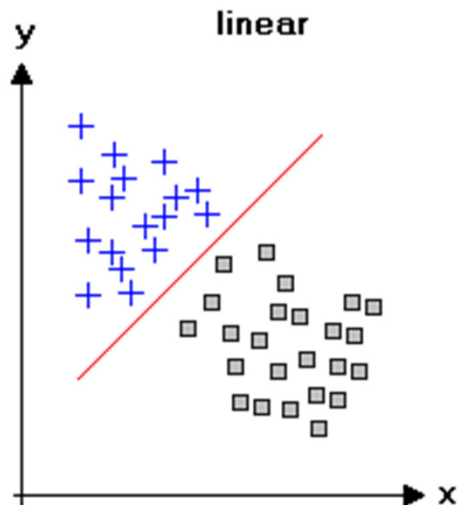
$$\min_{\theta} \sum_{i=1}^N (y_i - \theta^\top x_i)^2 \quad \text{Equivalent to least square error learning}$$

Linear Classification

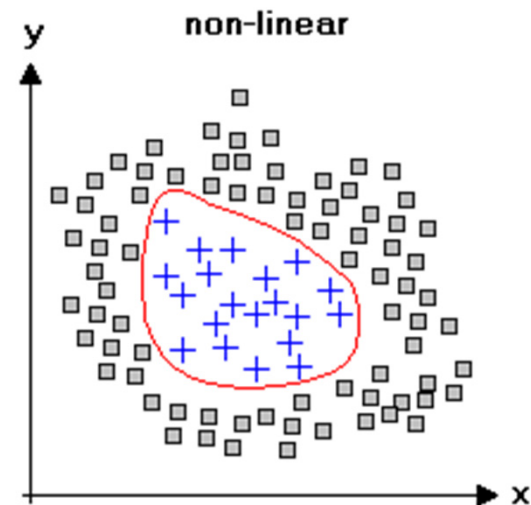
Classification Problem

- Given:
 - A description of an instance, $x \in \mathbb{X}$, where \mathbb{X} is the instance space.
 - A fixed set of categories: $C = \{c_1, c_2, \dots, c_m\}$
- Determine:
 - The category of $x: f(x) \in C$, where $f(x)$ is a categorization function whose domain is \mathbb{X} and whose range is C
 - If the category set binary, i.e. $C = \{0, 1\}$ ({false, true}, {negative, positive}) then it is called binary classification.

Binary Classification



Linearly inseparable



Non-linearly inseparable

Linear Discriminative Models

- Discriminative model
 - modeling the dependence of unobserved variables on observed ones
 - also called conditional models.
 - Deterministic: $y = f_{\theta}(x)$
 - Non-differentiable
 - **Probabilistic:** $p_{\theta}(y|x)$
 - Differentiable
- For binary classification

$$p_{\theta}(y = 1|x)$$

$$p_{\theta}(y = 0|x) = 1 - p_{\theta}(y = 1|x)$$

Loss Function

- Cross entropy loss

Discrete case: $H(p, q) = - \sum_x p(x) \log q(x)$

Continuous case: $H(p, q) = - \int_x p(x) \log q(x) dx$

- For classification problem

Ground Truth	0	1	0	0	0
Prediction	0.1	0.6	0.05	0.05	0.2

$$\mathcal{L}(y, x, p_\theta) = - \sum_k \delta(y = c_k) \log p_\theta(y = c_k | x)$$
$$\delta(z) = \begin{cases} 1, & z \text{ is true} \\ 0, & \text{otherwise} \end{cases}$$

Cross Entropy for Binary Classification

	Class 1	Class 2
Ground Truth	0	1
Prediction	0.3	0.7

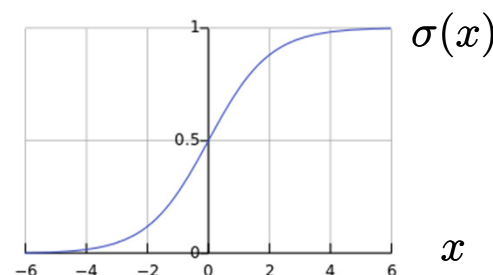
- Loss function

$$\begin{aligned}\mathcal{L}(y, x, p_{\theta}) &= -\delta(y = 1) \log p_{\theta}(y = 1|x) - \delta(y = 0) \log p_{\theta}(y = 0|x) \\ &= -y \log p_{\theta}(y = 1|x) - (1 - y) \log(1 - p_{\theta}(y = 1|x))\end{aligned}$$

Logistic Regression

- Logistic regression is a binary classification model

$$p_{\theta}(y = 1|x) = \sigma(\theta^{\top} x) = \frac{1}{1 + e^{-\theta^{\top} x}}$$
$$p_{\theta}(y = 0|x) = \frac{e^{-\theta^{\top} x}}{1 + e^{-\theta^{\top} x}}$$



- Cross entropy loss function

$$\mathcal{L}(y, x, p_{\theta}) = -y \log \sigma(\theta^{\top} x) - (1 - y) \log(1 - \sigma(\theta^{\top} x))$$

- Gradient

$$\begin{aligned} \frac{\partial \mathcal{L}(y, x, p_{\theta})}{\partial \theta} &= -y \frac{1}{\sigma(\theta^{\top} x)} \sigma(z)(1 - \sigma(z))x - (1 - y) \frac{-1}{1 - \sigma(\theta^{\top} x)} \sigma(z)(1 - \sigma(z))x \\ &= (\sigma(\theta^{\top} x) - y)x \\ \theta &\leftarrow \theta + \eta(y - \sigma(\theta^{\top} x))x \end{aligned}$$

$$\frac{\partial \sigma(z)}{\partial z} = \sigma(z)(1 - \sigma(z))$$

Label Decision

- Logistic regression provides the probability

$$p_{\theta}(y = 1|x) = \sigma(\theta^{\top} x) = \frac{1}{1 + e^{-\theta^{\top} x}}$$

$$p_{\theta}(y = 0|x) = \frac{e^{-\theta^{\top} x}}{1 + e^{-\theta^{\top} x}}$$

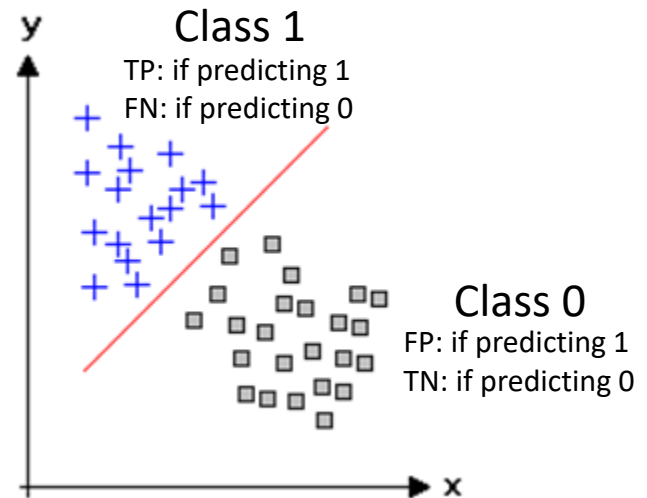
- The final label of an instance is decided by setting a threshold h

$$\hat{y} = \begin{cases} 1, & p_{\theta}(y = 1|x) > h \\ 0, & \text{otherwise} \end{cases}$$

Evaluation Measures

	Prediction	
	1	0
Label	1	True Positive False Negative
	0	False Positive True Negative

- True / False
 - True: prediction = label
 - False: prediction \neq label
- Positive / Negative
 - Positive: predict $y = 1$
 - Negative: predict $y = 0$



Evaluation Measures

		Prediction	
		1	0
Label	1	True Positive	False Negative
	0	False Positive	True Negative

- Accuracy: the ratio of cases when prediction = label

$$Acc = \frac{TP + TN}{TP + TN + FP + FN}$$

Evaluation Measures

		Prediction	
		1	0
Label	1	True Positive	False Negative
	0	False Positive	True Negative

- **Precision:** the ratio of true class 1 cases in those with prediction 1

$$\text{Prec} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

		Prediction	
		1	0
Label	1	True Positive	False Negative
	0	False Positive	True Negative

- **Recall:** the ratio of cases with prediction 1 in all true class 1 cases

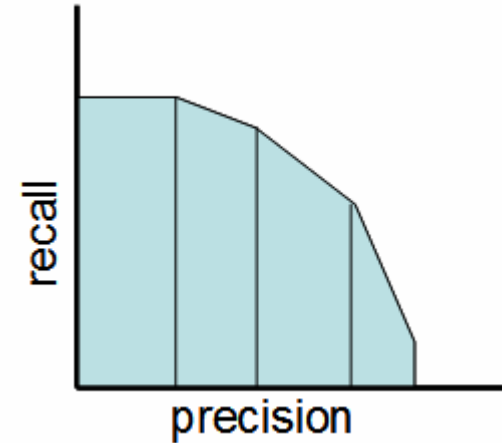
$$\text{Rec} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Evaluation Measures

- Precision-recall tradeoff

$$\hat{y} = \begin{cases} 1, & p_{\theta}(y = 1|x) > h \\ 0, & \text{otherwise} \end{cases}$$

- Higher threshold, higher precision, lower recall
 - Extreme case: threshold = 0.99
- Lower threshold, lower precision, higher recall
 - Extreme case: threshold = 0

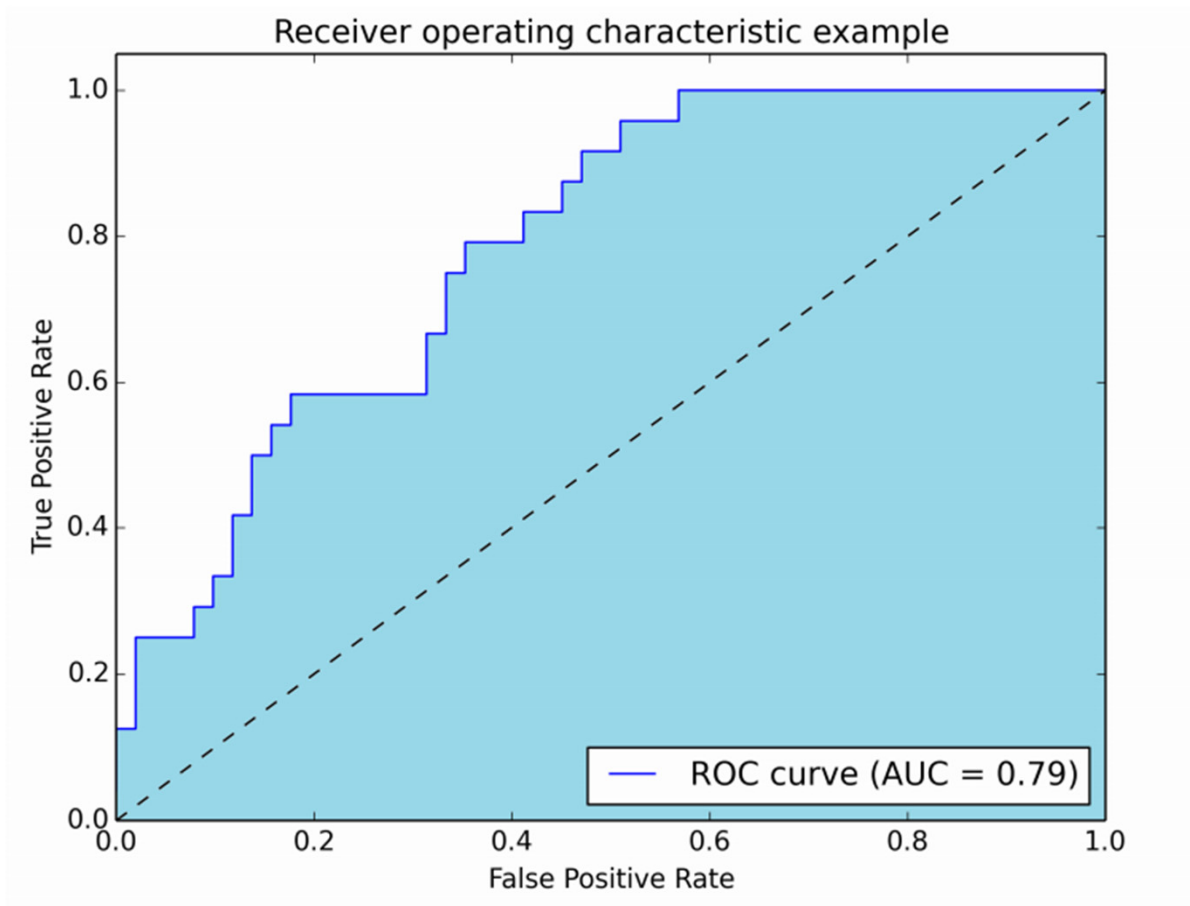


- F1 Measure

$$F1 = \frac{2 \times \text{Prec} \times \text{Recall}}{\text{Prec} + \text{Rec}}$$

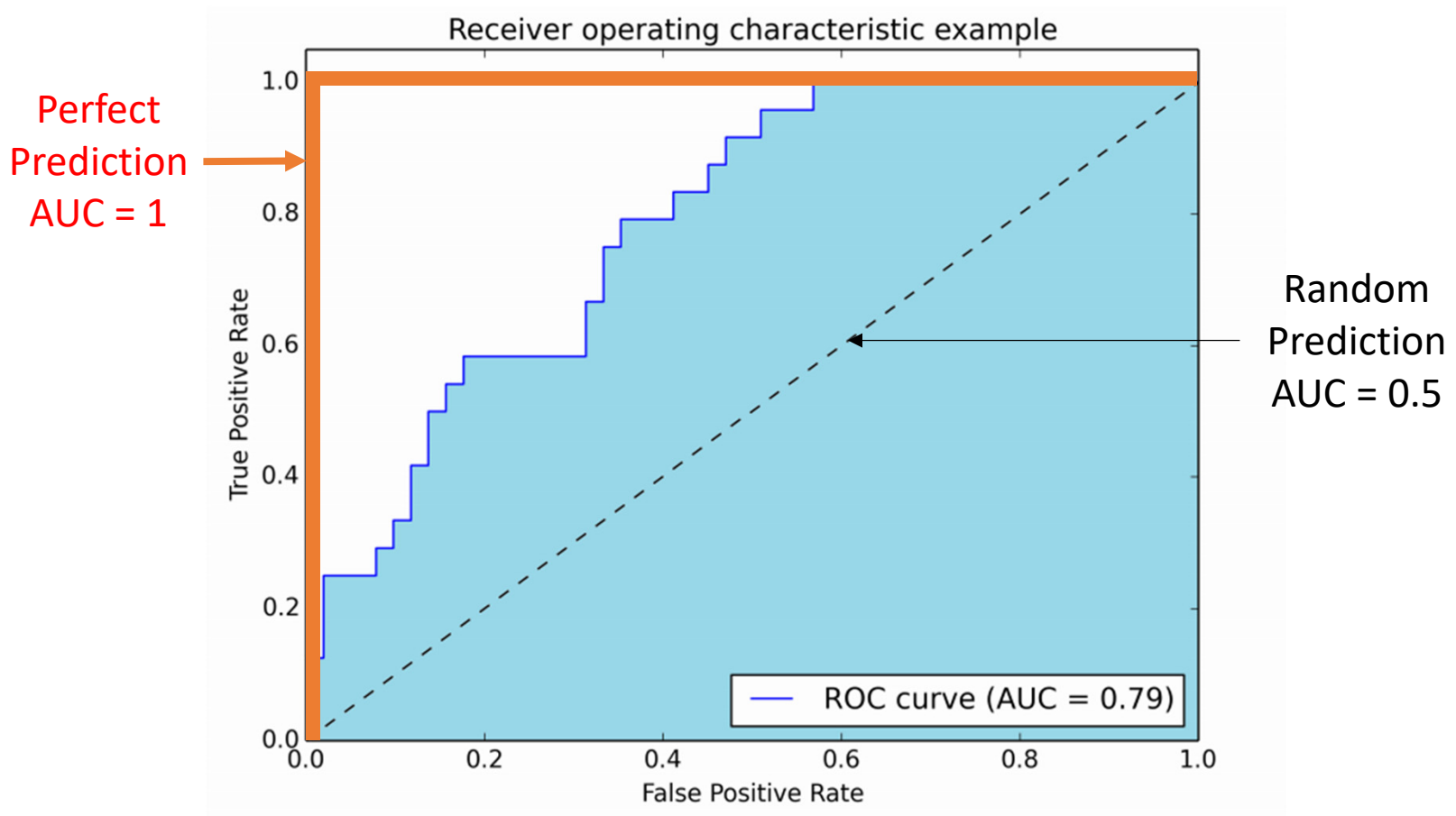
Evaluation Measures

- Ranking-based measure: Area Under ROC Curve (AUC)



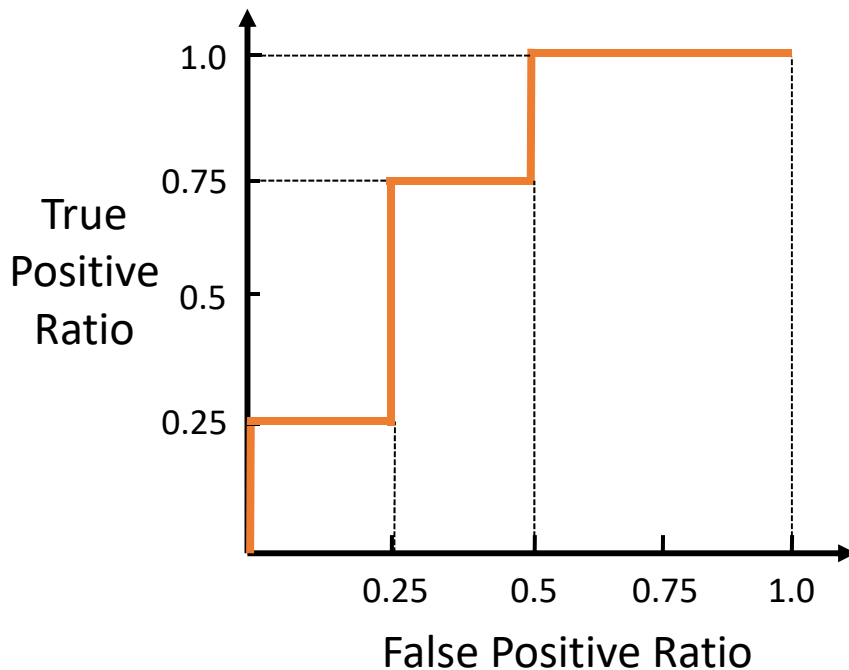
Evaluation Measures

- Ranking-based measure: Area Under ROC Curve (AUC)



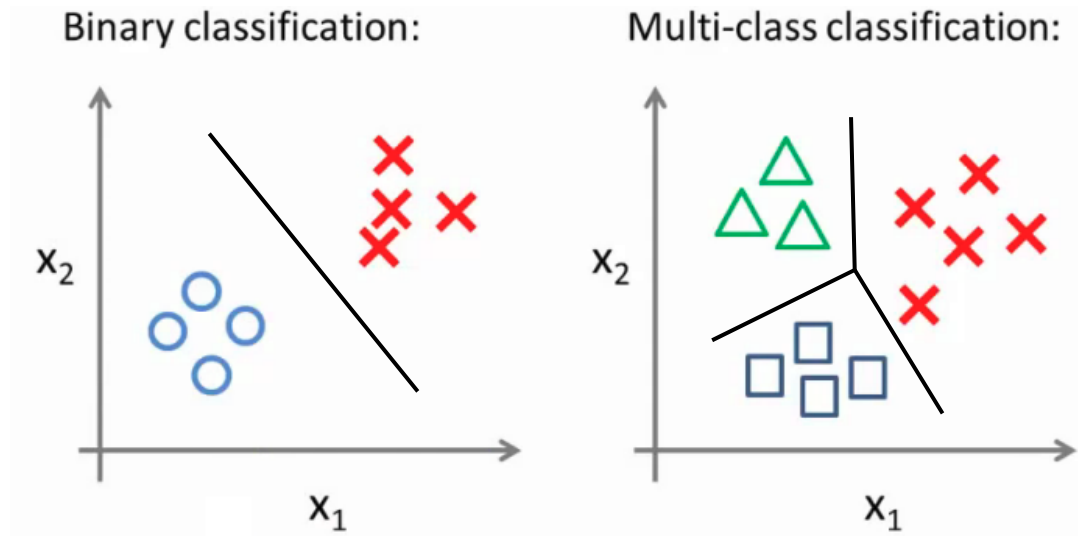
Evaluation Measures

- A simple example of Area Under ROC Curve (AUC)



Prediction	Label
0.91	1
0.85	0
0.77	1
0.72	1
0.61	0
0.48	1
0.42	0
0.33	0

Multi-Class Classification



- Still cross entropy loss

Ground Truth	<div>0</div>	<div>1</div>	<div>0</div>
Prediction	<div>0.1</div>	<div>0.7</div>	<div>0.2</div>

$$\mathcal{L}(y, x, p_{\theta}) = - \sum_k \delta(y = c_k) \log p_{\theta}(y = c_k | x) \quad \delta(z) = \begin{cases} 1, & z \text{ is true} \\ 0, & \text{otherwise} \end{cases}$$

Multi-Class Logistic Regression

- Class set $C = \{c_1, c_2, \dots, c_m\}$
- Predicting the probability of $p_\theta(y = c_j|x)$

$$p_\theta(y = c_j|x) = \frac{e^{\theta_j^\top x}}{\sum_{k=1}^m e^{\theta_k^\top x}} \quad \text{for } j = 1, \dots, m$$

- Softmax
 - Parameters $\theta = \{\theta_1, \theta_2, \dots, \theta_m\}$
 - Can be normalized with m-1 groups of parameters

Multi-Class Logistic Regression

- Learning on one instance $(x, y = c_j)$
 - Maximize log-likelihood

$$\max_{\theta} \log p_{\theta}(y = c_j | x)$$

- Gradient

$$\begin{aligned} \frac{\partial \log p_{\theta}(y = c_j | x)}{\partial \theta_j} &= \frac{\partial}{\partial \theta_j} \log \frac{e^{\theta_j^{\top} x}}{\sum_{k=1}^m e^{\theta_k^{\top} x}} \\ &= x - \frac{\partial}{\partial \theta_j} \log \sum_{k=1}^m e^{\theta_k^{\top} x} \\ &= x - \frac{e^{\theta_j^{\top} x} x}{\sum_{k=1}^m e^{\theta_k^{\top} x}} \end{aligned}$$

Application Case Study

Click-Through Rate (CTR) Estimation in Online Advertising

Ad Click-Through Rate Estimation

大陆



河南省公安厅彻查“封丘36人入警 35人身份不合规”

中封丘县公安局的36名受训人员，35人是公安局内部的文职或临时人员，与“民警必须具备公务员身份”的国家规定不符，引发该局内部

- 上海至成都沿江高铁提上日程 串联长江沿线22城市
- 2016号歼-20原型机曝光 已滑翔测试(图)
- 日媒：中国或派万吨海警船巡钓鱼岛 打消耗战
- 外媒：中国开始研制隐身武装直升机 预计2020年交付
- 习近平关于中美关系的十个判断
- 住建部黑臭水沟整治指南：9成百姓满意才能达标
- 陕西：职校“校长”让女学生陪酒 学校被撤除
- 揭秘“团团伙伙”的武钢漩涡和落马高官

国际



巴塞罗那200万人游行 呼吁加泰罗尼亚独立(图)

- 李炜光：收税是不公平的恶？
- 许章润：超级大国没有纯粹内政
- 刘昀献：国外政党联系群众的路径研究

时局观



民革中央副主席：中共从未否定国民党抗战作用

- 施芝鸿：文革基础上搞改革致一个时期市场官场乱象
- 朱维群回应争议：尊重民族差异而不强化
- 伊协副会长：穆斯林不应因宗教功修忽视社会责任

领袖圈



奥巴马54岁啦，当7年总统人苍老了头发也

Click or not?



海绵城市 未来之城
水危机：青岛告急
探访中国绿化博览会
帝都吸引华人首富
凤凰房产 诚邀加盟

谈华山论剑与中国精神
黑龙江创新驱动三步棋
《印记》之江城夜未眠
办公环境搜查令
圈层生活尽在凤凰会

精彩视频

凤凰联播台



菲媒曝菲律宾军演针对中国 直指南海生命线
播放数：2602282

User Response Estimation Problem

- Problem definition

One instance data

- Date: 20160320
- Hour: 14
- Weekday: 7
- IP: 119.163.222.*
- Region: England
- City: London
- Country: UK
- Ad Exchange: Google
- Domain: yahoo.co.uk
- URL: <http://www.yahoo.co.uk/abc/xyz.html>
- OS: Windows
- Browser: Chrome
- Ad size: 300*250
- Ad ID: a1890
- User occupation: Student
- User tags: Sports, Electronics



Corresponding label

Click (1) or not (0)?

Predicted CTR (0.15)

One-Hot Binary Encoding

- A standard feature engineering paradigm

$x = [\text{Weekday=Friday}, \text{Gender=Male}, \text{City=Shanghai}]$



The diagram shows three arrows pointing from the feature names in the text above to the corresponding binary values in the vector below. The first arrow points from 'Weekday=Friday' to the red '1' in the first group of zeros. The second arrow points from 'Gender=Male' to the blue '1' in the second group of zeros. The third arrow points from 'City=Shanghai' to the green '1' in the third group of zeros.

$x = [0, 0, 0, 0, 1, 0, 0 \quad 0, 1 \quad 0, 0, 1, 0 \dots 0]$

Sparse representation: $x = [5:1 \quad 9:1 \quad 12:1]$

- High dimensional sparse binary feature vector
 - Usually higher than 1M dimensions, even 1B dimensions
 - Extremely sparse

Training/Validation/Test Data

- Examples (in LibSVM format)

```
1 5:1 9:1 12:1 45:1 154:1 509:1 4089:1 45314:1 988576:1
0 2:1 7:1 18:1 34:1 176:1 510:1 3879:1 71310:1 818034:1
...
```

- Training/Validation/Test data split
 - Sort data by time
 - Train:validation:test = 8:1:1
 - Shuffle training data

Training Logistic Regression

- Logistic regression is a binary classification model

$$p_{\theta}(y = 1|x) = \sigma(\theta^{\top} x) = \frac{1}{1 + e^{-\theta^{\top} x}}$$

- Cross entropy loss function with L2 regularization

$$\mathcal{L}(y, x, p_{\theta}) = -y \log \sigma(\theta^{\top} x) - (1 - y) \log(1 - \sigma(\theta^{\top} x)) + \frac{\lambda}{2} \|\theta\|_2^2$$

- Parameter learning

$$\theta \leftarrow (1 - \lambda\eta)\theta + \eta(y - \sigma(\theta^{\top} x))x$$

- Only update non-zero entries

Experimental Results

- Datasets
 - Criteo Terabyte Dataset
 - 13 numerical fields, 26 categorical fields
 - 7 consecutive days out of 24 days in total (about 300 GB) during 2014
 - 79.4M impressions, 1.6M clicks after negative down sampling
 - iPinYou Dataset
 - 65 categorical fields
 - 10 consecutive days during 2013
 - 19.5M impressions, 937.7K clicks without negative down sampling

Performance

Model	Linearity	AUC		Log Loss	
		Criteo	iPinYou	Criteo	iPinYou
Logistic Regression	Linear	71.48%	73.43%	0.1334	5.581e-3
Factorization Machine	Bi-linear	72.20%	75.52%	0.1324	5.504e-3
Deep Neural Networks	Non-linear	75.66%	76.19%	0.1283	5.443e-3

- Compared with non-linear models, linear models
 - Pros: standardized, easily understood and implemented, efficient and scalable
 - Cons: modeling limit (feature independent assumption), cannot explore feature interactions

Generalized Linear Models

Review: Linear Regression

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}^{(1)} \\ \mathbf{x}^{(2)} \\ \vdots \\ \mathbf{x}^{(n)} \end{bmatrix} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & x_3^{(1)} & \dots & x_d^{(1)} \\ x_1^{(2)} & x_2^{(2)} & x_3^{(2)} & \dots & x_d^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_1^{(n)} & x_2^{(n)} & x_3^{(n)} & \dots & x_d^{(n)} \end{bmatrix} \quad \boldsymbol{\theta} = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_d \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

- Prediction $\hat{\mathbf{y}} = \mathbf{X}\boldsymbol{\theta} = \begin{bmatrix} \mathbf{x}^{(1)}\boldsymbol{\theta} \\ \mathbf{x}^{(2)}\boldsymbol{\theta} \\ \vdots \\ \mathbf{x}^{(n)}\boldsymbol{\theta} \end{bmatrix}$

- Objective $J(\boldsymbol{\theta}) = \frac{1}{2}(\mathbf{y} - \hat{\mathbf{y}})^\top (\mathbf{y} - \hat{\mathbf{y}}) = \frac{1}{2}(\mathbf{y} - \mathbf{X}\boldsymbol{\theta})^\top (\mathbf{y} - \mathbf{X}\boldsymbol{\theta})$

Review: Matrix Form of Linear Reg.

- Objective

$$J(\boldsymbol{\theta}) = \frac{1}{2}(\mathbf{y} - \mathbf{X}\boldsymbol{\theta})^\top (\mathbf{y} - \mathbf{X}\boldsymbol{\theta}) \quad \min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$

- Gradient

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = -\mathbf{X}^\top (\mathbf{y} - \mathbf{X}\boldsymbol{\theta})$$

- Solution

$$\begin{aligned} \frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \mathbf{0} &\rightarrow \mathbf{X}^\top (\mathbf{y} - \mathbf{X}\boldsymbol{\theta}) = \mathbf{0} \\ &\rightarrow \mathbf{X}^\top \mathbf{y} = \mathbf{X}^\top \mathbf{X} \boldsymbol{\theta} \\ &\rightarrow \hat{\boldsymbol{\theta}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y} \end{aligned}$$

Generalized Linear Models

- Dependence

$$y = f(\theta^\top \phi(x))$$

- Feature mapping function $\phi(x) : \mathbb{R}^d \mapsto \mathbb{R}^h$
- Mapped feature matrix $\Phi_{n \times h}$

$$\Phi = \begin{bmatrix} \phi(x^{(1)}) \\ \phi(x^{(2)}) \\ \vdots \\ \phi(x^{(i)}) \\ \vdots \\ \phi(x^{(n)}) \end{bmatrix} = \begin{bmatrix} \phi_1(x^{(1)}) & \phi_2(x^{(1)}) & \cdots & \phi_h(x^{(1)}) \\ \phi_1(x^{(2)}) & \phi_2(x^{(2)}) & \cdots & \phi_h(x^{(2)}) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(x^{(i)}) & \phi_2(x^{(i)}) & \cdots & \phi_h(x^{(i)}) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(x^{(n)}) & \phi_2(x^{(n)}) & \cdots & \phi_h(x^{(n)}) \end{bmatrix}$$

Matrix Form of Kernel Linear Regression

- Objective

$$J(\boldsymbol{\theta}) = \frac{1}{2}(\mathbf{y} - \boldsymbol{\Phi}\boldsymbol{\theta})^\top (\mathbf{y} - \boldsymbol{\Phi}\boldsymbol{\theta}) \quad \min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$

- Gradient

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = -\boldsymbol{\Phi}^\top (\mathbf{y} - \boldsymbol{\Phi}\boldsymbol{\theta})$$

- Solution

$$\begin{aligned} \frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \mathbf{0} &\rightarrow \boldsymbol{\Phi}^\top (\mathbf{y} - \boldsymbol{\Phi}\boldsymbol{\theta}) = \mathbf{0} \\ &\rightarrow \boldsymbol{\Phi}^\top \mathbf{y} = \boldsymbol{\Phi}^\top \boldsymbol{\Phi} \boldsymbol{\theta} \\ &\rightarrow \hat{\boldsymbol{\theta}} = (\boldsymbol{\Phi}^\top \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^\top \mathbf{y} \end{aligned}$$

Matrix Form of Kernel Linear Regression

- With the Algebra trick

$$(\mathbf{P}^{-1} + \mathbf{B}^{\top} \mathbf{R}^{-1} \mathbf{B})^{-1} \mathbf{B}^{\top} \mathbf{R}^{-1} = \mathbf{P} \mathbf{B}^{\top} (\mathbf{B} \mathbf{P} \mathbf{B}^{\top} + \mathbf{R})^{-1}$$

- The optimal parameters with L2 regularization

$$\begin{aligned} \hat{\boldsymbol{\theta}} &= (\boldsymbol{\Phi}^{\top} \boldsymbol{\Phi} + \lambda \mathbf{I}_h)^{-1} \boldsymbol{\Phi}^{\top} \mathbf{y} \\ &= \boldsymbol{\Phi}^{\top} (\boldsymbol{\Phi} \boldsymbol{\Phi}^{\top} + \lambda \mathbf{I}_n)^{-1} \mathbf{y} \end{aligned}$$

for prediction, we never actually need to access $\boldsymbol{\Phi}$

$$\begin{aligned} \hat{\mathbf{y}} &= \boldsymbol{\Phi} \hat{\boldsymbol{\theta}} = \boldsymbol{\Phi} \boldsymbol{\Phi}^{\top} (\boldsymbol{\Phi} \boldsymbol{\Phi}^{\top} + \lambda \mathbf{I}_n)^{-1} \mathbf{y} \\ &= \mathbf{K} (\mathbf{K} + \lambda \mathbf{I}_n)^{-1} \mathbf{y} \end{aligned}$$

where the kernel matrix $\mathbf{K} = \{K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})\}$