CX4240 Spring 2026

Supervised Learning: Linear Regression

Bo Dai
School of CSE, Georgia Tech
bodai@cc.gatech.edu

Organization

- ***Background knowledge***
  - Probability and Statistics, Linear Algebra, Optimization, Coding skills (PyTorch and JAX)
- ***Supervised learning***
- ***Unsupervised learning***
- ***Advanced Topics***

Syllabus

- ***Background knowledge***
  - Probability and Statistics, Linear Algebra, Optimization, Coding skills
- ***Supervised learning***
- ***Unsupervised learning***
- ***Advanced Topics: LLM & RL***

}

Modeling: what to learn

Learning: how to learn

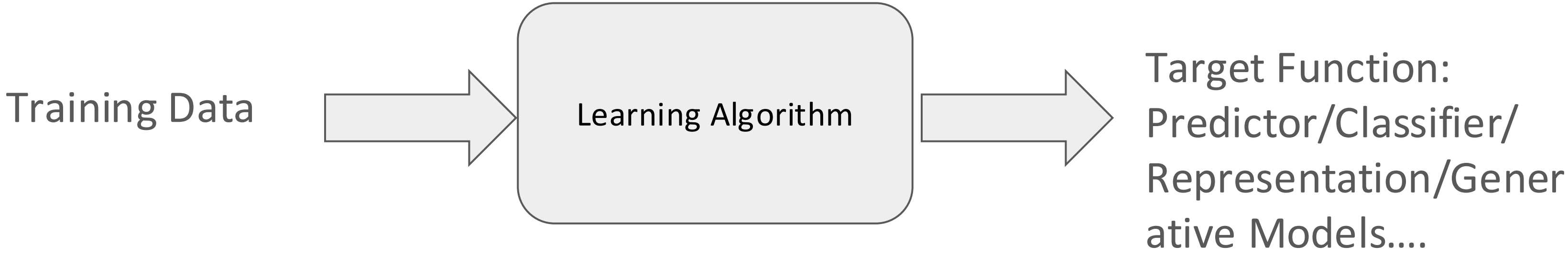
Implementation

Syllabus

- ***Background knowledge***
  - Probability and Statistics, Linear Algebra, Optimization, Coding skills
- ***Supervised learning***
- ***Unsupervised learning***
- ***Advanced Topics: LLM & RL***

Modeling: Probability and Statistics, Linear Algebra
Learning: Optimization, Linear Algebra
Implementation: Coding (PyTorch/JAX)

ML Algorithm Pipeline

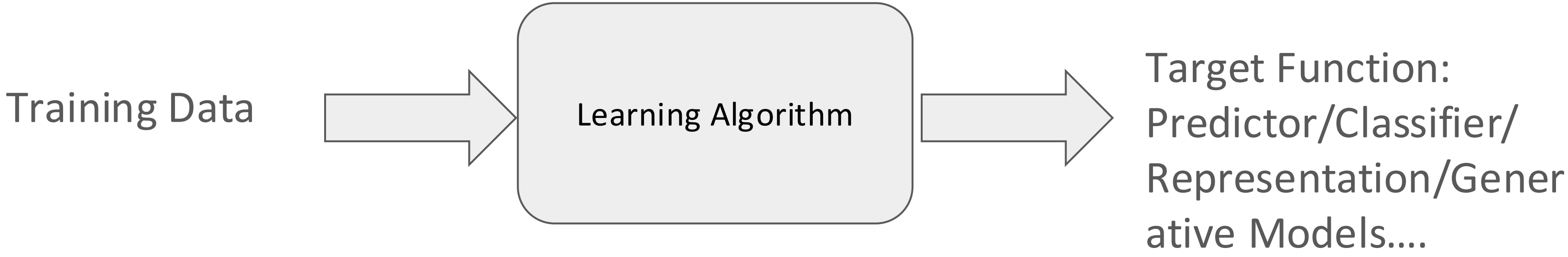


Training Data

Learning Algorithm

Target Function: Predictor/Classifier/ Representation/Generative Models….

ML Algorithm Pipeline



Training Data

Learning Algorithm

Target Function: Predictor/Classifier/Representation/Generative Models....

General ML Algorithm Pipeline

1. Build probabilistic models
2. Derive loss function (by MLE/MAP, maximum margin, contrastive…)
3. Select optimizer

Syllabus

- ***Background knowledge***
  - Probability and Statistics, Linear Algebra, Optimization, Coding skills
- ***Supervised learning***
- ***Unsupervised learning***
- ***Advanced Topics***

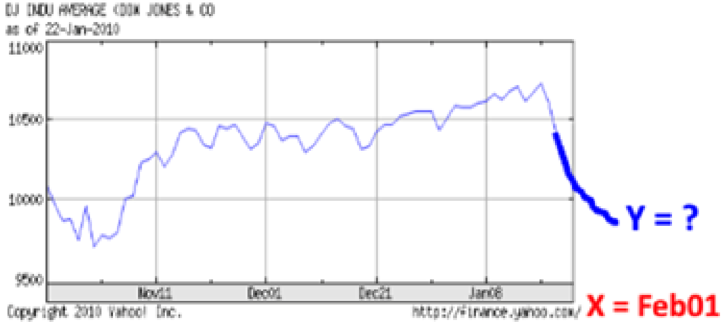
Modeling: Probability and Statistics, Linear Algebra
Learning: Optimization, Linear Algebra
Implementation: Coding

Supervised Learning

Goal: Construct a **predictor** f: X → Y

→ Sports
Science
News

**Classification:
discrete categories**

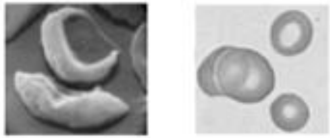


**Regression:
Real-valued numbers**

Classification Tasks

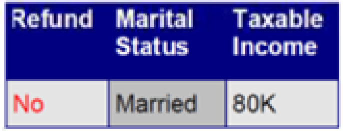
	Feature, X	Label, Y

Diagnosing sickle cell anemia

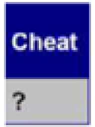


Anemic cell
Healthy cell

Tax Fraud Detection



Refund	Marital Status	Taxable Income
No	Married	80K



Cheat
?

Web Classification

Sports
Science
News

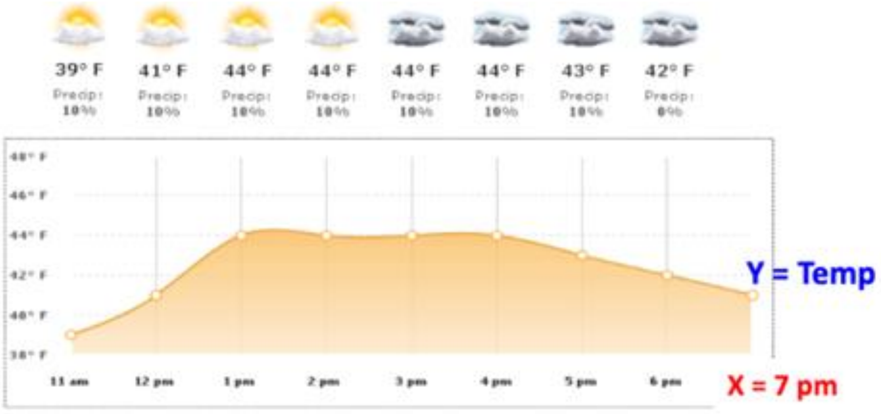
Predict Fulton resident

Drive to GT, Hawk's fan, Shop at Publix

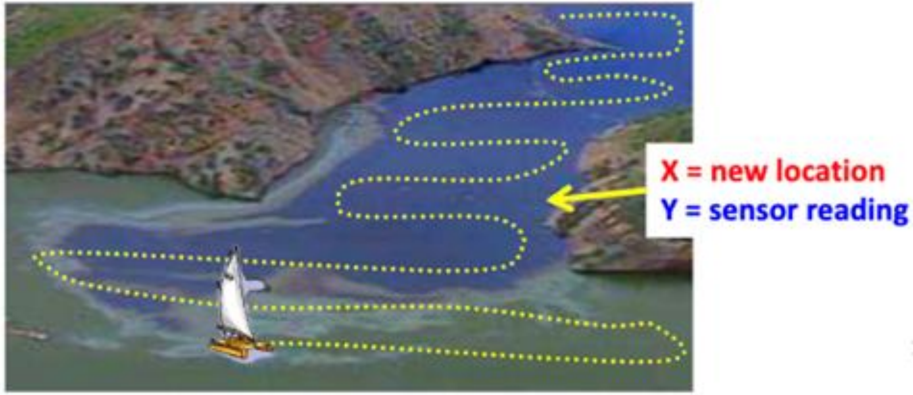
Resident
Not resident

Regression Tasks

Weather Prediction



Estimating Contamination

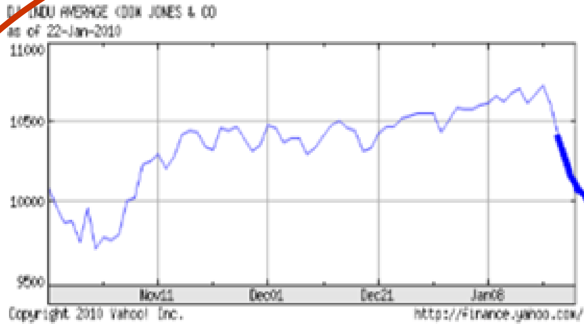


Supervised Learning

Goal: Construct a **predictor** f: X → Y to
minimize a risk (performance measure) R(f).

Sports
Science
News

Classification:
discrete categories

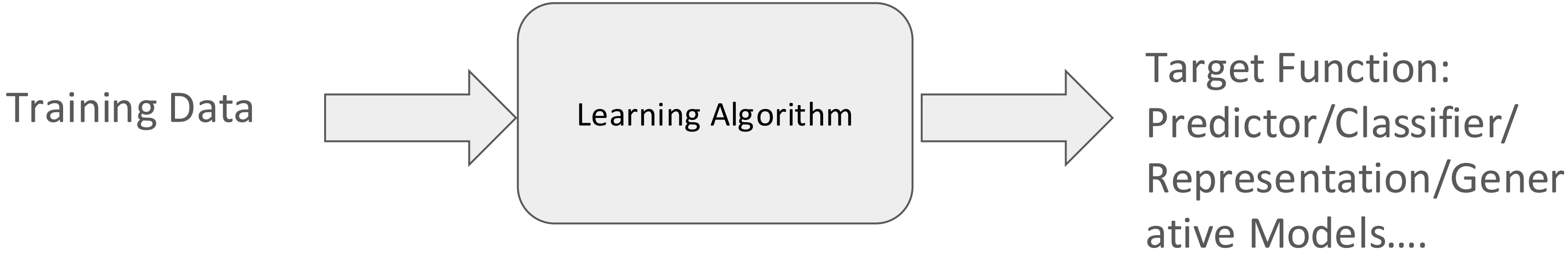


Y = ?

X = Feb01

Regression:
Real-valued number

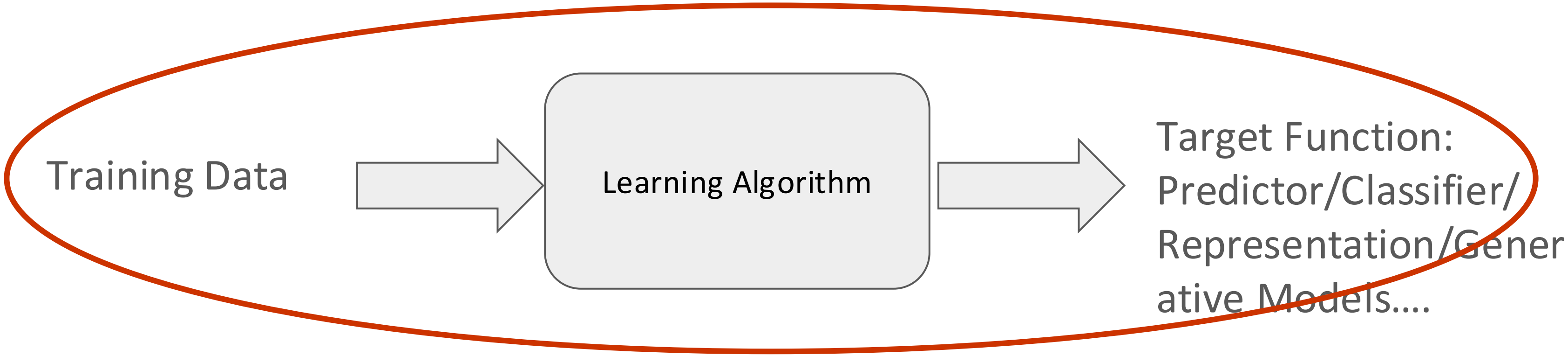
ML Algorithm Pipeline


Training Data
Learning Algorithm
Target Function: Predictor/Classifier/ Representation/Generative Models….

General ML Algorithm Pipeline

1. Build probabilistic models
2. Derive loss function (by MLE or MAP)
3. Select optimizer

ML Algorithm Pipeline



General ML Algorithm Pipeline

1. Build probabilistic models
2. Derive loss function (by MLE or MAP)
3. Select optimizer

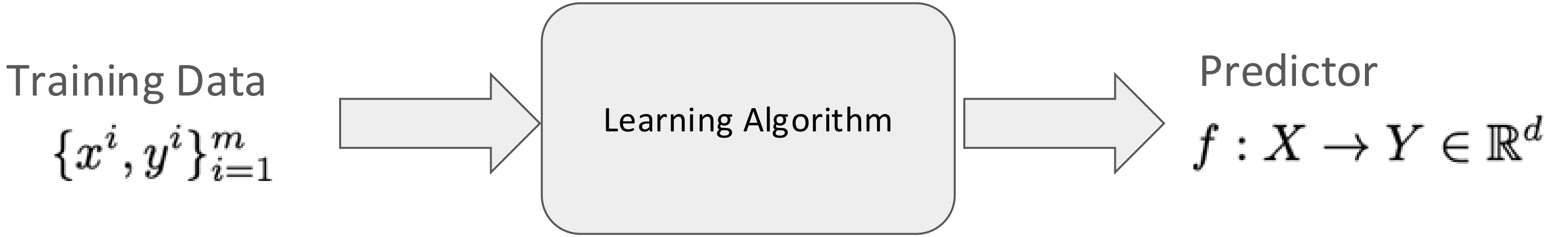
Machine Learning for Apartment Hunting

- Suppose you are to move to Atlanta
- And you want to find the most reasonably priced apartment satisfying your needs:

X y

Living area (ft^2)	# bedroom	Monthly rent ($)
230	1	900
506	2	1800
433	2	1500
190	1	800
...		
150	1	?
270	1.5	?

Regression algorithms



Training Data

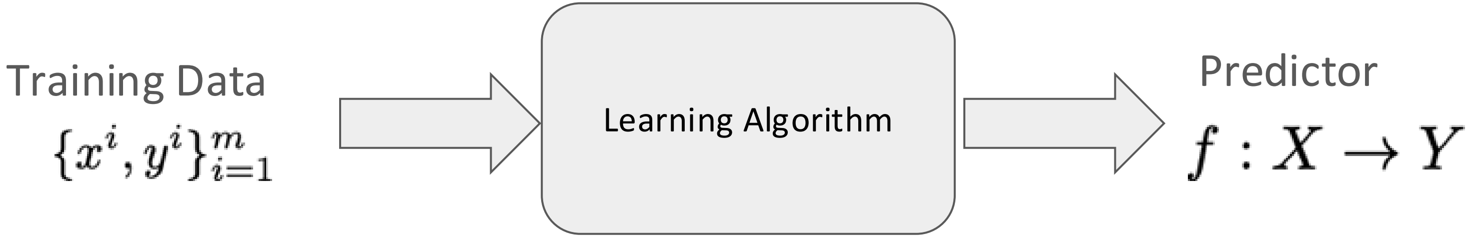
$\{x^i, y^i\}_{i=1}^m$

Learning Algorithm

Predictor

$f : X \to Y \in \mathbb{R}^d$

Regression algorithms



Training Data

$\{x^i, y^i\}_{i=1}^m$

Learning Algorithm

Predictor

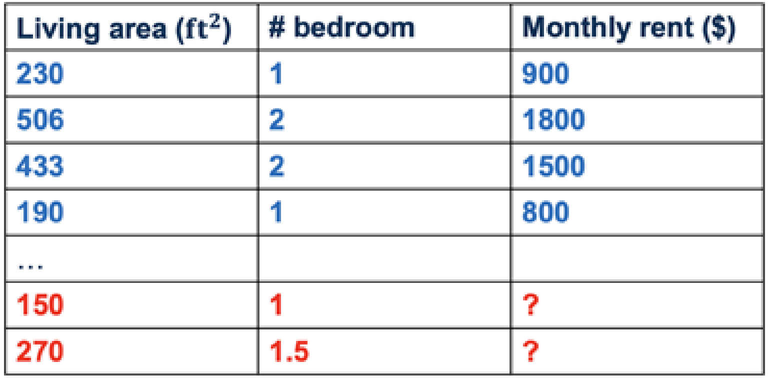
$f : X \to Y$

- **Features:**
  - Living area, distance to campus, # of bedroom …
  - Denotes as $x = (x_1, x_2, \ldots, x_n)^\top$
- **Target**
  - Rent
  - Denote as y
- **Training set**

  $X = (x^1, x^2, \ldots, x^m)$

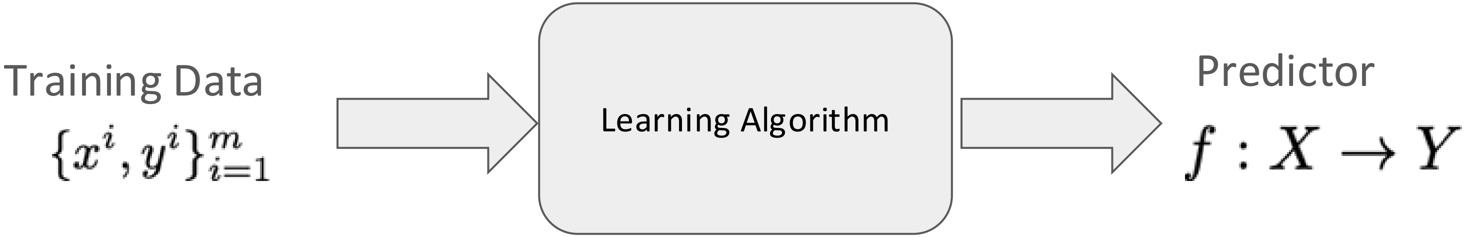
  $y = (y^1, y^2, \ldots, y^m)^\top$

$$y = \theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n$$



Living area (ft^2)	# bedroom	Monthly rent ($)
230	1	900
506	2	1800
433	2	1500
190	1	800
…		
150	1	?
270	1.5	?

Regression algorithms



General ML Algorithm Pipeline

1. Build probabilistic models
2. Derive loss function (by MLE or MAP)
3. Select optimizer

Probabilistic Model:
Linear Regression Model with Gaussian Noise

- Assume y is a linear function of x (features) plus noise ϵ

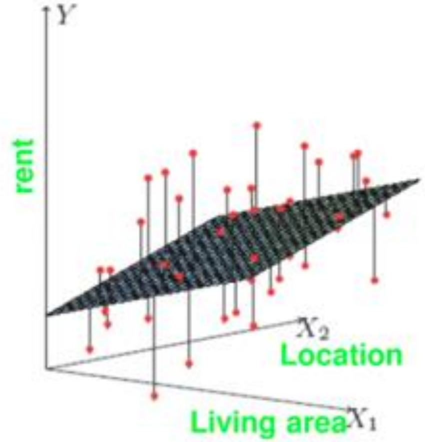
$$y = \theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n + \epsilon$$

where ϵ is an error model as Gaussian $N(0, \sigma^2)$

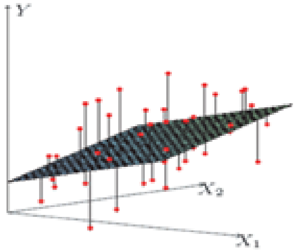
- Let $\theta = (\theta_0, \theta_1, \ldots, \theta_n)^\top$, and augment data by one dimension

$$x \leftarrow (1, x)^\top$$

Then $y = \theta^\top x + \epsilon$



Probabilistic Model: Gaussian Likelihood

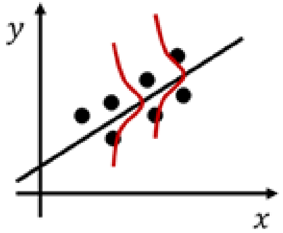


- Assume y is a linear in x plus noise ϵ

$$y = \theta^\top x + \epsilon$$

- Assume ϵ follows a Gaussian $N(0, \sigma)$

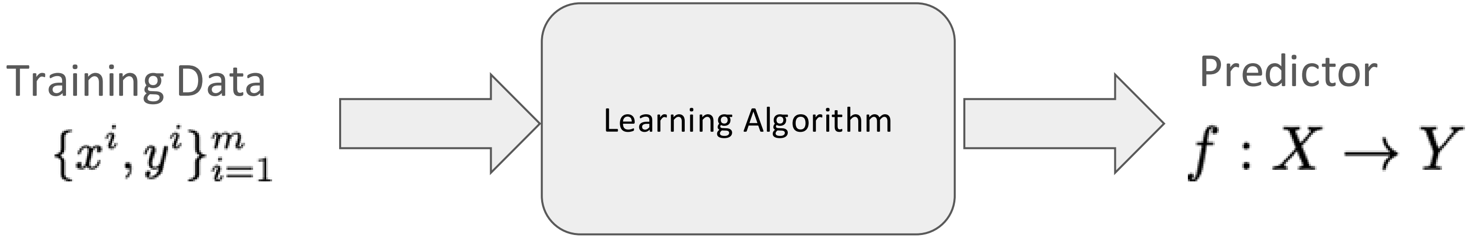
$$p(y^i \mid x^i; \theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\left(y^i - \theta^\top x^i\right)^2}{2\sigma^2}\right)$$



- By independence assumption, likelihood is

$$L(\theta) = \prod_i^m p(y^i \mid x^i; \theta) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^m \exp\left(-\frac{\sum_i^m \left(y^i - \theta^\top x^i\right)^2}{2\sigma^2}\right)$$

Regression algorithms



Training Data

$\{x^i, y^i\}_{i=1}^m$

Learning Algorithm

Predictor

$f : X \rightarrow Y$

General ML Algorithm Pipeline

1. Build probabilistic models
2. Derive loss function (by MLE or MAP)
3. Select optimizer

Maximum log-Likelihood Estimation (MLE)

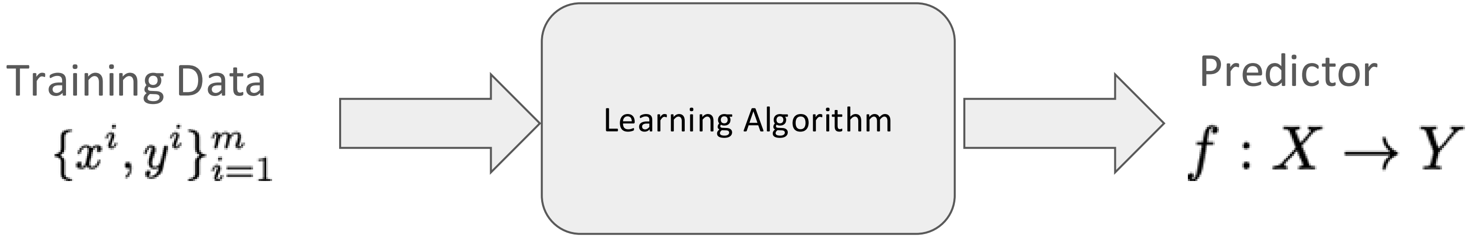
$$L(\theta) \;= \prod_{i}^{m} p(y^i|x^i;\theta) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^m \exp\left(-\frac{\sum_i^m \left(y^i - \theta^\top x^i\right)^2}{2\sigma^2}\right)$$

Maximum log-Likelihood Estimation (MLE)

$$L(\theta) \; = \prod_{i}^{m} p(y^i | x^i; \theta) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^m \exp\left(-\frac{\sum_i^m (y^i - \theta^\top x^i)^2}{2\sigma^2}\right)$$

$$\max_{\theta} \; \log L(\theta) = -\frac{1}{2\sigma^2} \sum_{i=1}^{m} (y^i - \theta^\top x^i)^2 - m \log(\sqrt{2\pi}\sigma)$$

Regression algorithms



Training Data

$\{x^i, y^i\}_{i=1}^m$

Learning Algorithm

Predictor

$f : X \to Y$

General ML Algorithm Pipeline

1. Build probabilistic models
2. Derive loss function (by MLE or MAP)
3. Select optimizer

Select Optimizer

$$\min_{\theta} \; -\log L(\theta) \propto \frac{1}{m}\sum_{i=1}^{m}(y^i - \theta^\top x^i)^2$$

Select Optimizer

$$\min_{\theta} \; -\log L(\theta) \propto \frac{1}{m}\sum_{i=1}^{m}(y^i - \theta^\top x^i)^2$$

- Necessary Condition
- (Stochastic) Gradient Descent

Necessary Condition

$$\min_{\theta} \; -\log L(\theta) \propto \frac{1}{m}\sum_{i=1}^{m}(y^i - \theta^\top x^i)^2$$

$$\frac{\partial \log L(\theta)}{\partial \theta} \qquad\qquad\qquad\qquad = 0$$

Necessary Condition

$$\min_{\theta} \; -\log L(\theta) \propto \frac{1}{m} \sum_{i=1}^{m} (y^i - \theta^\top x^i)^2$$

$$\frac{\partial \log L(\theta)}{\partial \theta} = -\frac{2}{m} \sum_{i=1}^{m} (y^i - \theta^\top x^i) x^{i\top} = 0$$

Necessary Condition

$$\min_{\theta} \; -\log L(\theta) \propto \frac{1}{m}\sum_{i=1}^{m}(y^i - \theta^\top x^i)^2$$

$$\frac{\partial \log L(\theta)}{\partial \theta} = -\frac{2}{m}\sum_{i=1}^{m}(y^i - \theta^\top x^i)x^{i\top} = 0$$

$$\Leftrightarrow -\frac{2}{m}\sum_{i=1}^{m} y^i x^i + \frac{2}{m}\sum_{i=1}^{m} x^i x^{i\top}\theta = 0$$

Necessary Condition

$$\frac{\partial \log L(\theta)}{\partial \theta} = -\frac{2}{m}\sum_{i=1}^{m} y^i x^i + \frac{2}{m}\sum_{i=1}^{m} x^i {x^i}^\top \theta = 0$$

Necessary Condition

$$\frac{\partial \log L(\theta)}{\partial \theta} = -\frac{2}{m}\sum_{i=1}^{m} y^i x^i + \frac{2}{m}\sum_{i=1}^{m} x^i {x^i}^\top \theta = 0$$

$$\frac{\partial \log L(\theta)}{\partial \theta} = -\frac{2}{m}(x^1 \dots, x^m)(y^1 \dots, y^m)^\top + \frac{2}{m}(x^1, \dots x^m)(x^1, \dots x^m)^\top \theta = 0$$

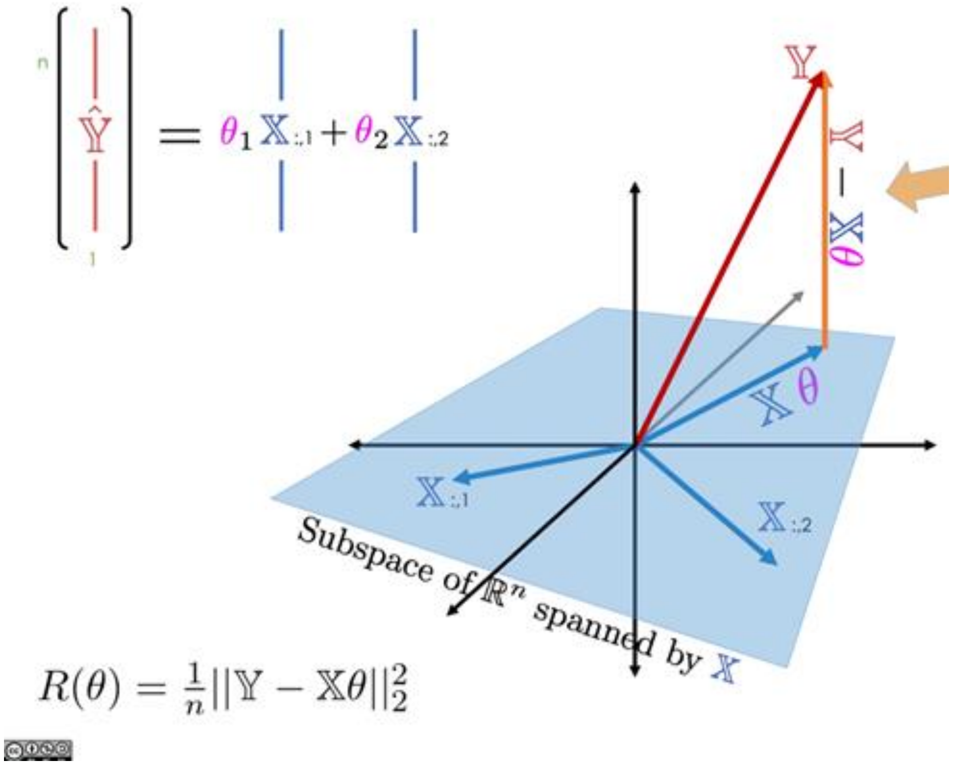
Define $X = (x^1, x^2, \dots x^m), y = (y^1, y^2, \dots, y^m)^\top$, gradient becomes

$$\frac{\partial \log L(\theta)}{\partial \theta} = -\frac{2}{m} Xy + \frac{2}{m} XX^\top \theta = 0$$

$$\Rightarrow \hat{\theta} = (XX^\top)^{-1} Xy$$

Geometric Interpretation

$$\hat{\theta} = (XX^{\top})^{-1}Xy$$



$$\hat{\mathbb{Y}} = \theta_1 \mathbb{X}_{:,1} + \theta_2 \mathbb{X}_{:,2}$$

$$R(\theta) = \frac{1}{n}||\mathbb{Y} - \mathbb{X}\theta||_2^2$$

n
1
$\mathbb{Y}$
$\mathbb{Y} - \mathbb{X}\theta$
$\mathbb{X}\theta$
$\mathbb{X}_{:,1}$
$\mathbb{X}_{:,2}$
Subspace of $\mathbb{R}^n$ spanned by $\mathbb{X}$

Select Optimizer

$$\min_{\theta} \; -\log L(\theta) \propto \frac{1}{m} \sum_{i=1}^{m} (y^i - \theta^\top x^i)^2$$

- Necessary Condition
- (Stochastic) Gradient Descent

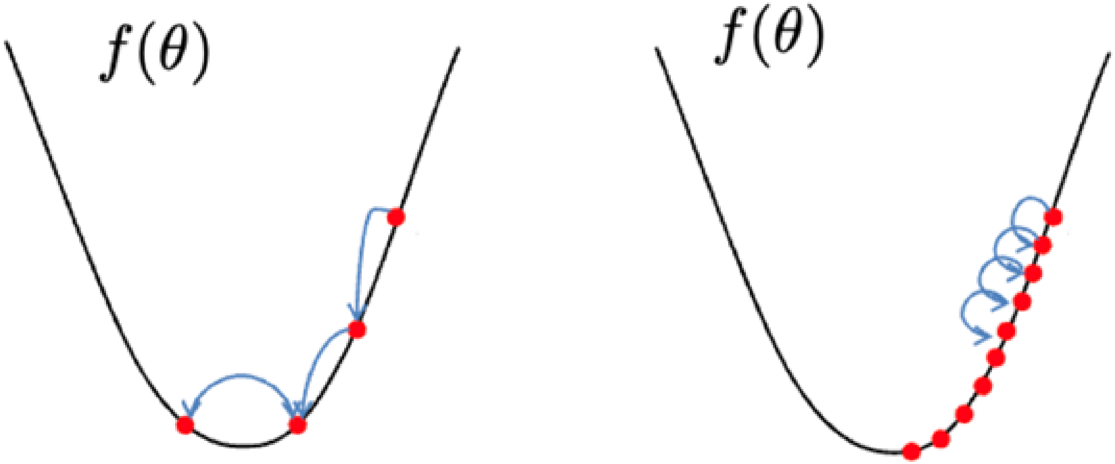
Gradient Method Revisit

$$\min_{\theta} \; \underbrace{-\log L(\theta) \propto \frac{1}{m}\sum_{i=1}^{m}(y^i - \theta^\top x^i)^2}_{f(\theta)}$$

choose an initial $\theta^1 \in \mathbf{R}^d$ and $h^1 > 0$ (e.g., $\theta^1 = 0, h^1 = 1$)
for $k = 1,2,\ldots,k^{\max}$

1. compute $\nabla f(\theta^k)$; quit if $\|\nabla f(\theta^k)\|_2$ is small enough
2. form tentative update $\theta^{\text{tent}} = \theta^k - h^k \nabla f(\theta^k)$
3. if $f(\theta^{\text{tent}}) < f(\theta^k)$, set $\theta^{k+1} = \theta^{\text{tent}}, h^{k+1} = 1.2h^k$
4. else set $h^k := 0.5h^k$ and go to step 2

Effect of Learning Rate in GD



Large α ⇒ Fast convergence but larger residual error. Also possible oscillation.
Small α ⇒ Slow convergence but small residual error.

Gradient Calculation

$$\frac{\partial \log L(\theta)}{\partial \theta} = -\frac{2}{m}\sum_{i=1}^{m}(y^i - \theta^\top x^i)x^i$$

form tentative update $\theta^{\text{tent}} = \theta^k - h^k \nabla f(\theta^k)$

$$\theta^k + \frac{h^k}{m}\sum_{i=1}^{m}(y^i - (\theta^k)^\top x^i)(x^i)^\top$$

Gradient Method Revisit

```
choose an initial $\theta^1 \in \mathbf{R}^d$ and $h^1 > 0$ (e.g., $\theta^1 = 0, h^1 = 1$ )
for $k = 1,2,\dots,k^{\text{max}}$
                    Stochastic Approximation
    1.  compute $\nabla \cancel{f(\theta^k)}$; quit if $\|\nabla f(\theta^k)\|_2$ is small enough
    2.  form tentative update $\theta^{\text{tent}} = \theta^k - h^k \nabla f(\theta^k)$
    3.  if $f(\theta^{\text{tent}}) < f(\theta^k)$, set $\theta^{k+1} = \theta^{\text{tent}}, h^{k+1} = 1.2h^k$
    4.  else set $h^k := 0.5h^k$ and go to step 2
```

Stochastic Gradient Descent

$$\frac{\partial \log L(\theta)}{\partial \theta} = -\frac{2}{m}\sum_{i=1}^{m}(y^i - \theta^\top x^i)x^i \quad \approx \quad \left(y^i - \hat{\theta}^{t\top} x^i\right) x^i$$

Initialize $\theta^0 \in \mathbb{R}^d$ randomly Iterate until convergence:

1 Randomly sample a point (x_i, y_i) from the n data points

2 Compute noisy gradient $\tilde{g}^t = (y^i - (\theta^t)^\top x^i)(x^i)^\top$

3 Update (GD): $\theta^{t+1} = \theta^t - \eta \tilde{g}^t$

Recap

- Stochastic gradient update rule

$$\hat{\theta}^{t+1} \leftarrow \hat{\theta}^{t} + \beta \left(y^{i} - (\hat{\theta}^{t})^{\top} x^{i} \right) x^{i}$$

  - Pros: online, low per-step cost
  - Cons: coordinate, (sometimes) slow-converging
- Gradient descent

$$\hat{\theta}^{t+1} \leftarrow \hat{\theta}^{t} + \frac{\alpha}{m} \sum_{i=1}^{m} \left(y^{i} - (\hat{\theta}^{t})^{\top} x^{i} \right) x^{i}$$

  - Pros: fast-converging, easy to implement
  - Cons: need to read all data
- Solve normal equations

$$(X^{\top} X)\hat{\theta} = X^{\top} y$$

  - Pros: a single-shot algorithm! Easiest to implement
  - Cons: need to compute inverse, expensive, numerical issue (e.g., matrix is singular …)

Summary

General ML Algorithm Pipeline

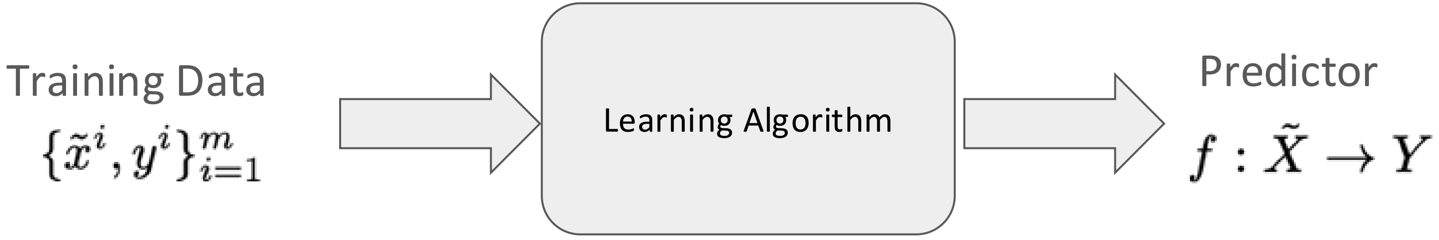
1. Build probabilistic models
2. Derive loss function (by MLE or MAP)
3. Select optimizer

Summary

General ML Algorithm Pipeline

1. Build probabilistic models
2. Derive loss function (by MLE or MAP)
3. Select optimizer

Polynomial Regression



Training Data

$\{\tilde{x}^i, y^i\}_{i=1}^{m}$

Learning Algorithm

Predictor

$f : \tilde{X} \to Y$

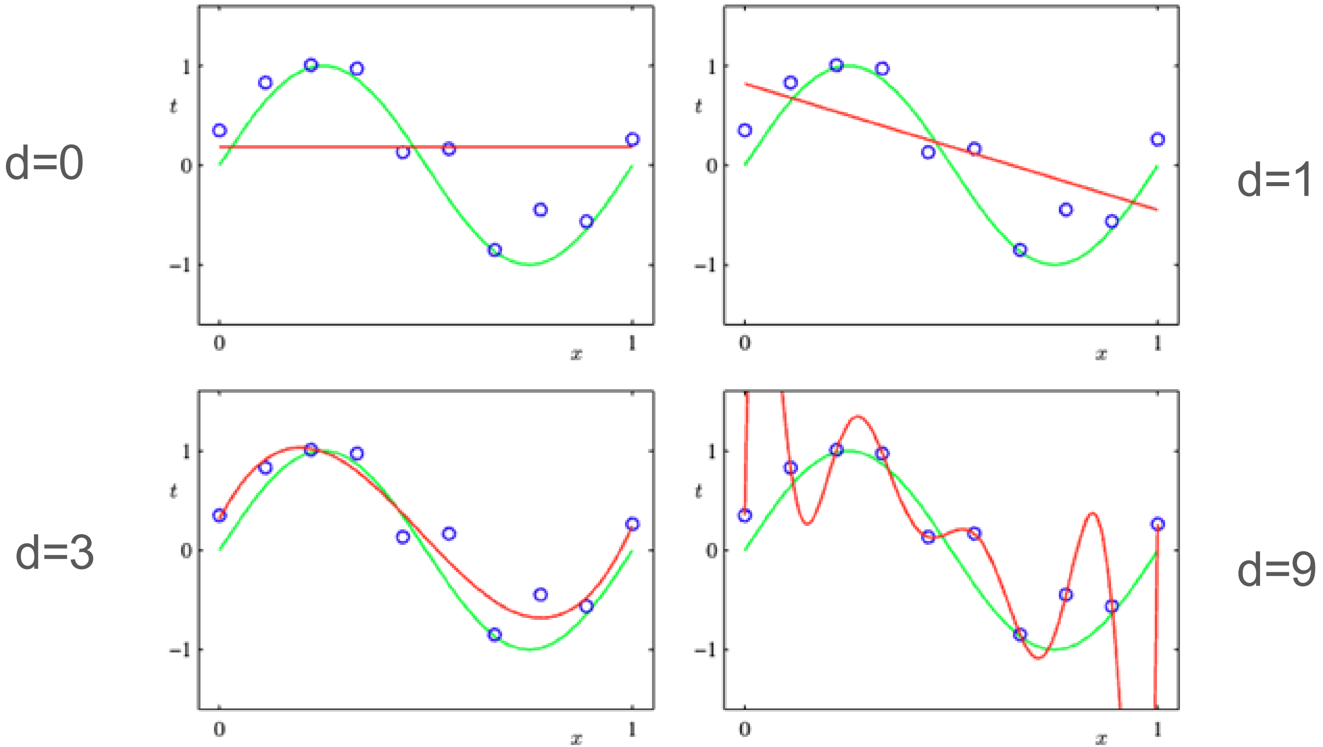
$$
\begin{aligned}
y = \theta_0 &+ \theta_1^1 x_1 + \theta_1^2 (x_1)^2 + \theta_1^3 (x_1)^3 + \ldots + \theta_1^d (x_1)^d \\
&+ \theta_2^1 x_2 + \theta_2^2 (x_2)^2 + \theta_2^3 (x_2)^3 + \ldots + \theta_2^d (x_2)^d \\
&+ \ldots \\
&+ \theta_n^1 x_n + \theta_n^2 (x_n)^2 + \theta_n^3 (x_n)^3 + \ldots + \theta_n^d (x_n)^d
\end{aligned}
$$

- **Features:**
  - Living area, distance to campus, # of bedroom …
  - Denotes as $\tilde{x} = [1, x_1, (x_1)^2, \ldots, (x_1)^d, \ldots, x_n, \ldots, (x_n)^d]$
- **Target**
  - Rent
  - Denote as y
- **Training set**

  $\tilde{X} = [\tilde{x}^1, \tilde{x}^2, \ldots, \tilde{x}^m]$

  $y = (y^1, y^2, \ldots, y^m)^\top$

Overfitting with Increased Degree



Summary

General ML Algorithm Pipeline

1. Build probabilistic models
2. Derive loss function (by MLE or MAP)
3. Select optimizer

Maximum a Posteriori (MAP)

$$L(\theta) \;= \prod_{i}^{m} p(y^i | x^i; \theta) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^m \exp\left(-\frac{\sum_i^m \left(y^i - \theta^\top x^i\right)^2}{2\sigma^2}\right)$$ Likelihood

$$p(\theta) \propto \exp(-\lambda \|\theta\|^2)$$ Gaussian Prior

Maximum a Posteriori (MAP)

$$L(\theta) \ = \prod_{i}^{m} p(y^i|x^i;\theta) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^m \exp\left(-\frac{\sum_i^m \left(y^i - \theta^\top x^i\right)^2}{2\sigma^2}\right)$$

Likelihood

$$p(\theta) \propto \exp(-\lambda\|\theta\|_2^2)$$

Gaussian Prior

$$p(\theta|\{x^i, y^i\}_{i=1}^m) = \frac{\prod_{i=1}^m p(y^i|x^i,\theta)p(\theta)}{\int \prod_{i=1}^m p(y^i|x^i,\theta)p(\theta)d\theta}$$

Posterior: Bayes' Rule

Maximum a Posteriori (MAP)

$$L(\theta) \ = \prod_{i}^{m} p(y^i|x^i;\theta) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^m \exp\left(-\frac{\Sigma_i^m (y^i - \theta^\top x^i)^2}{2\sigma^2}\right)$$

Likelihood

$$p(\theta) \propto \exp(-\lambda \|\theta\|_2^2)$$

Gaussian Prior

$$p(\theta|\{x^i, y^i\}_{i=1}^m) = \frac{\prod_{i=1}^m p(y^i|x^i,\theta)p(\theta)}{\int \prod_{i=1}^m p(y^i|x^i,\theta)p(\theta)d\theta}$$

Posterior: Bayes' Rule

$$\max_{\theta} \ \log p(\theta|\{x^i, y^i\}_{i=1}^m)$$

MAP

Maximum a Posteriori (MAP)

$$L(\theta) = \prod_{i}^{m} p(y^i|x^i;\theta) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^m \exp\left(-\frac{\Sigma_i^m\left(y^i - \theta^\top x^i\right)^2}{2\sigma^2}\right)$$

Likelihood

$$p(\theta) \propto \exp(-\lambda\|\theta\|_2^2)$$

Gaussian Prior

$$p(\theta|\{x^i, y^i\}_{i=1}^m) = \frac{\prod_{i=1}^m p(y^i|x^i,\theta)p(\theta)}{\int \prod_{i=1}^m p(y^i|x^i,\theta)p(\theta)d\theta}$$

Posterior: Bayes' Rule

$$\max_\theta \; \log p(\theta|\{x^i, y^i\}_{i=1}^m) = \log L(\theta) + \log p(\theta)$$

Ridge Regression

$$\propto -\frac{1}{m}\sum_{i=1}^m (y^i - \theta^\top x^i)^2 - \lambda\|\theta\|_2^2$$

Maximum a Posteriori (MAP)

$$L(\theta) = \prod_{i}^{m} p(y^i | x^i; \theta) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^m \exp\left(-\frac{\Sigma_i^m (y^i - \theta^\top x^i)^2}{2\sigma^2}\right)$$

Likelihood

$$p(\theta) \propto \exp(-\lambda \|\theta\|_1)$$

Laplacian Prior

$$p(\theta | \{x^i, y^i\}_{i=1}^m) = \frac{\prod_{i=1}^m p(y^i | x^i, \theta) p(\theta)}{\int \prod_{i=1}^m p(y^i | x^i, \theta) p(\theta) d\theta}$$

Posterior: Bayes' Rule

$$\max_{\theta} \log p(\theta | \{x^i, y^i\}_{i=1}^m) = \log L(\theta) + \log p(\theta)$$

$$\propto -\frac{1}{m} \sum_{i=1}^{m} (y^i - \theta^\top x^i)^2 - \lambda \|\theta\|_1$$

Lasso

Select Optimizer

$$\min_{\theta} \; -\log p(\theta | \{x^i, y^i\}_{i=1}^{m}) \propto \frac{1}{m} \sum_{i=1}^{m} (y^i - \theta^\top x^i)^2 + \lambda \|\theta\|_2^2$$

- Necessary Condition
- (Stochastic) Gradient Descent

Necessary Condition

$$\min_{\theta} \; -\log p(\theta|\{x^i, y^i\}_{i=1}^m) \propto \frac{1}{m}\sum_{i=1}^{m}(y^i - \theta^\top x^i)^2 + \lambda\|\theta\|_2^2$$

$$\frac{\partial \log L(\theta)}{\partial \theta} = -\frac{2}{m}\sum_{i=1}^{m}(y^i - \theta^\top x^i)x^i \qquad \frac{\partial \lambda \theta^\top \theta}{\partial \theta} = 2\lambda\theta$$

$$\frac{2}{m}\sum_{i=1}^{m} y^i x^i - \frac{2}{m}\sum_{i=1}^{m} x^i (x^i)^\top \theta + 2\lambda\theta = 0$$

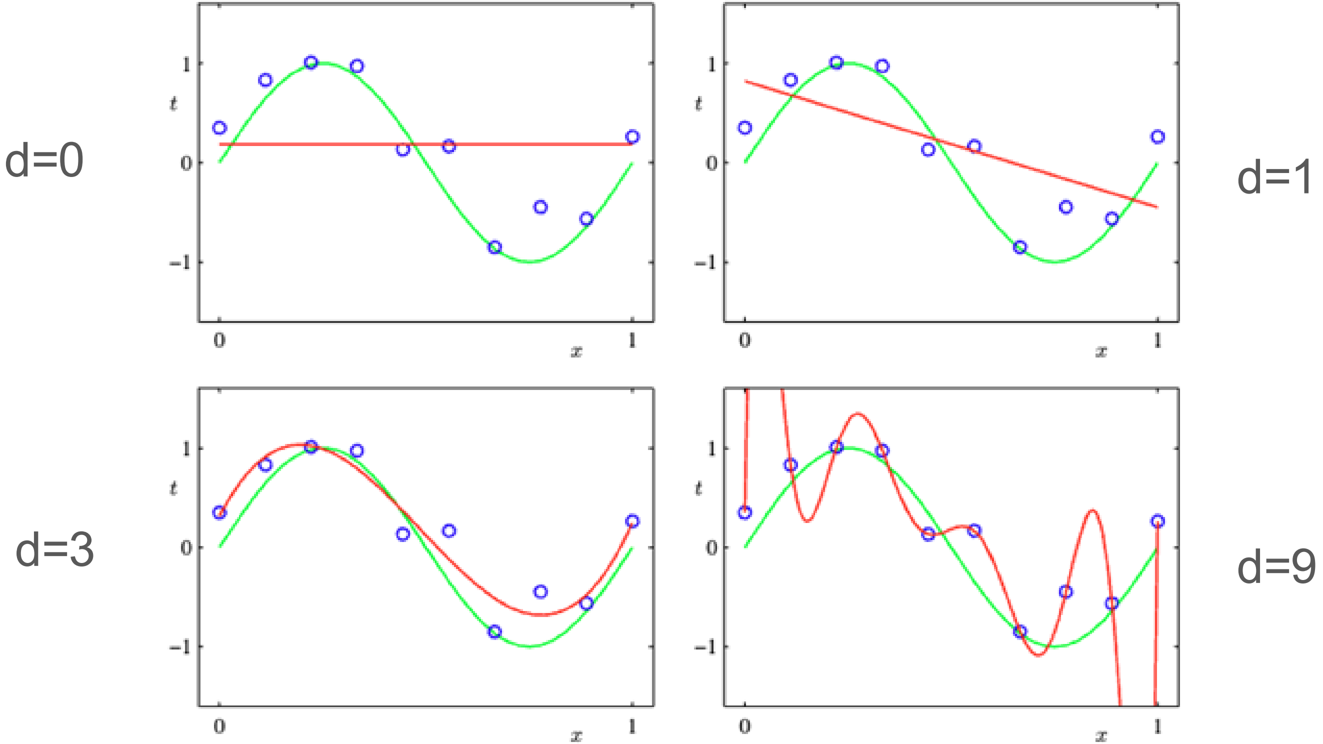
Necessary Condition

$$\frac{2}{m}\sum_{i=1}^{m} y^i x^i - \frac{2}{m}\sum_{i=1}^{m} x^i (x^i)^\top \theta + 2\lambda\theta = 0$$

$$\frac{2}{m} X y - \frac{2}{m} X X^\top \theta + 2\lambda\theta = 0$$

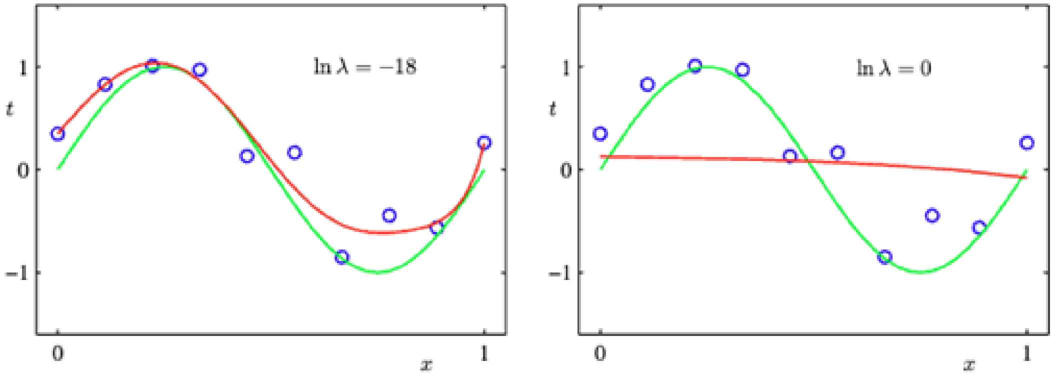
$$\Rightarrow \hat{\theta} = (X X^\top + \lambda m I)^{-1} X y$$

Overfitting with Increased Degree



Best Degree?

d=9



$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
0.35	0.35	0.13
232.37	4.74	-0.05
-5321.83	-0.77	-0.06
48568.31	-31.97	-0.05
-231639.30	-3.89	-0.03
640042.26	55.28	-0.02
-1061800.52	41.32	-0.01
1042400.18	-45.95	-0.00
-557682.99	-91.53	0.00
125201.43	72.68	0.01

- MLE with appropriate d
- MAP with large d, regularization will select the appropriate model

MLE vs. MAP

MLE

- We chose the “best” θ that maximized the likelihood given data
- No prior

$$\hat{\theta} = (XX^\top)^{-1}Xy$$

- Numerical issue
- Overfitting

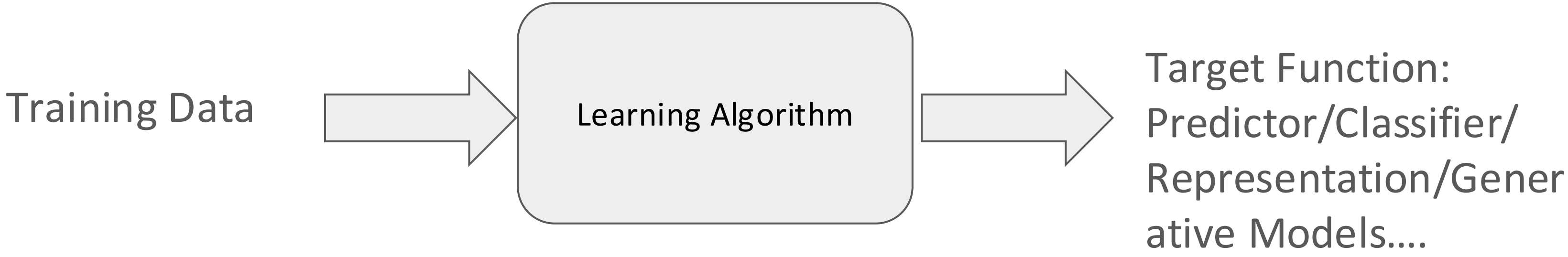
MAP

- We chose the “best” θ that maximized the posterior given data
- Prior matters

$$\hat{\theta} = (XX^\top + \lambda mI)^{-1}Xy$$

- No numerical issue
- Mitigate overfitting

ML Algorithm Pipeline



General ML Algorithm Pipeline

1. Build probabilistic models
2. Derive loss function (by MLE/MAP, maximum margin, contrastive…)
3. Select optimizer

Q&A