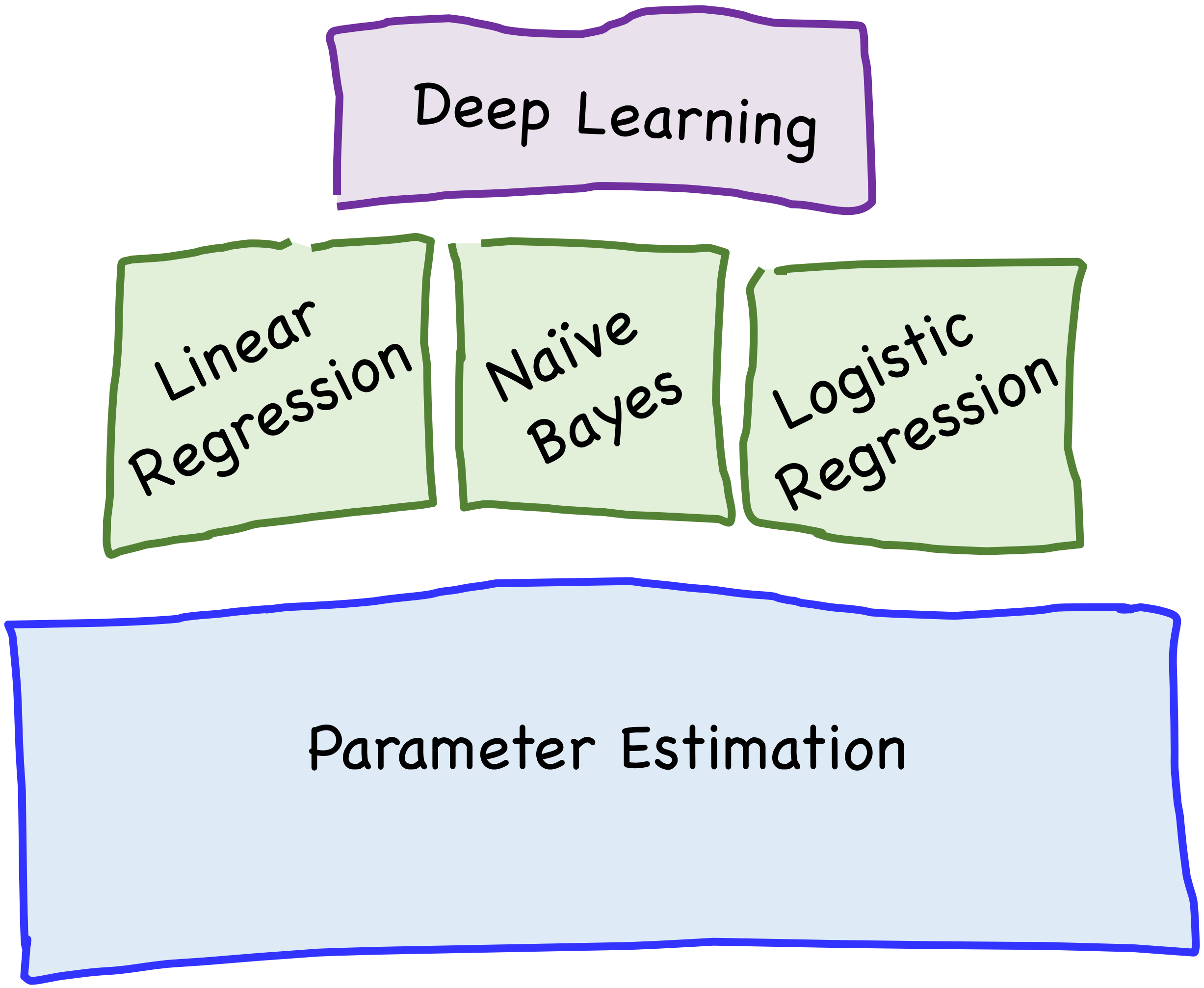
Gradient Ascent

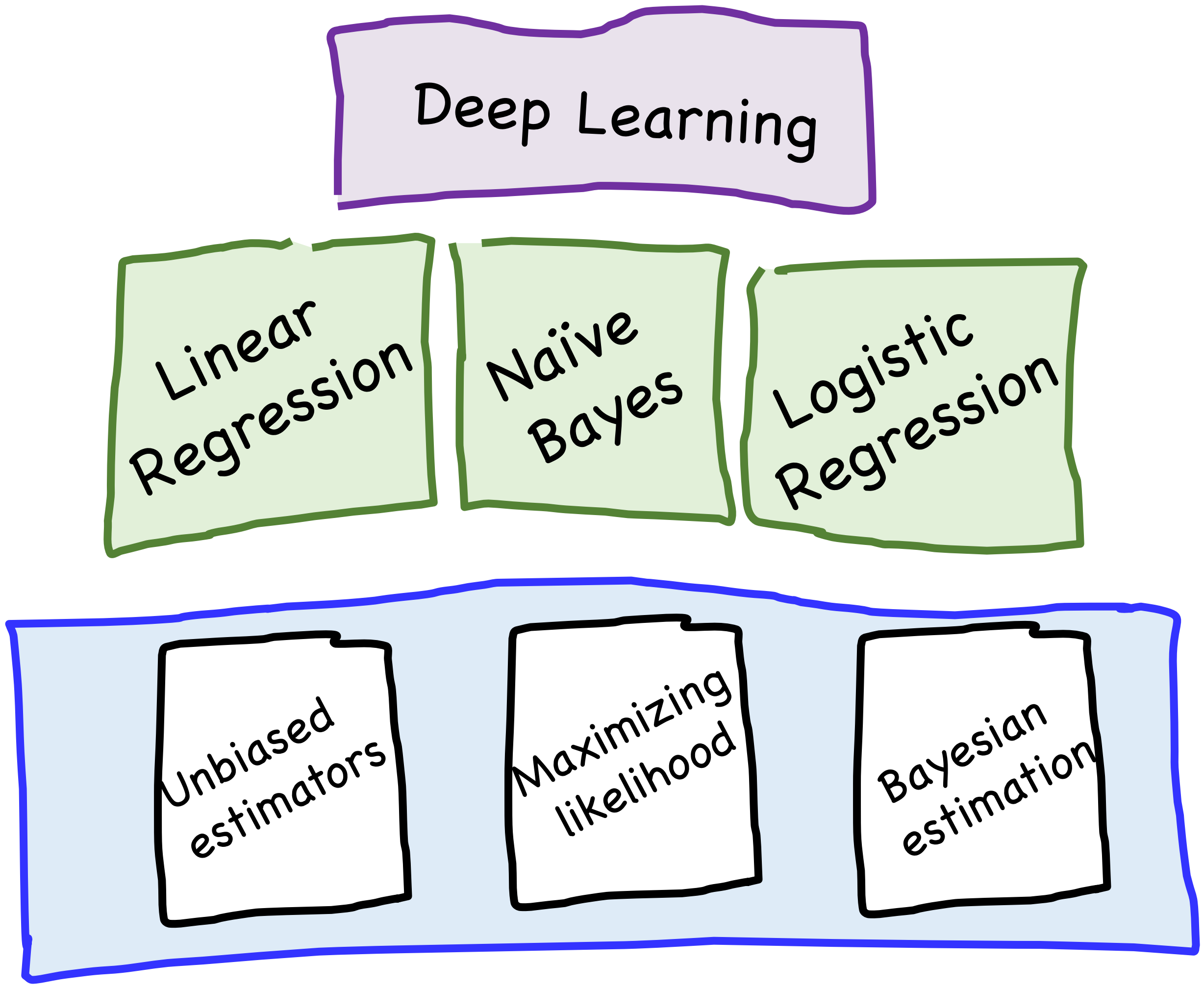
Chris Piech
CS109, Stanford University

Our Path



Deep Learning
Linear Regression
Naïve Bayes
Logistic Regression
Parameter Estimation

Our Path


Deep Learning
Linear Regression
Naïve Bayes
Logistic Regression
Unbiased estimators
Maximizing likelihood
Bayesian estimation

Parameter Learning

- Consider *n* I.I.D. random variables $X_1, X_2, ..., X_n$
  - X_i is a sample from density function $f(X_i \mid \theta)$
  - What are the best choice of parameters θ ?

Likelihood (of data given parameters):

$$L(\theta) = \prod_{i=1}^{n} f(X_i \mid \theta)$$

Maximum Likelihood

$$L(\theta) = \prod_{i=1}^{n} f(X_i \mid \theta)$$

$$LL(\theta) = \sum_{i=1}^{n} \log f(X_i|\theta)$$

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \; LL(\theta)$$

Argmax?

Option #1: Straight optimization

Computing the MLE

- General approach for finding MLE of θ
  - Determine formula for $LL(\theta)$
  - Differentiate $LL(\theta)$ w.r.t. (each) θ : $\frac{\partial LL(\theta)}{\partial \theta}$
  - To maximize, set $\frac{\partial LL(\theta)}{\partial \theta} = 0$
  - Solve resulting (simultaneous) equations to get θ_{MLE}
    - Make sure that derived $\hat{\theta}_{MLE}$ is actually a maximum (and not a minimum or saddle point). E.g., check $LL(\theta_{MLE} \pm \varepsilon) < LL(\theta_{MLE})$
      - This step often ignored in expository derivations
      - So, we'll ignore it here too (and won't require it in this class)

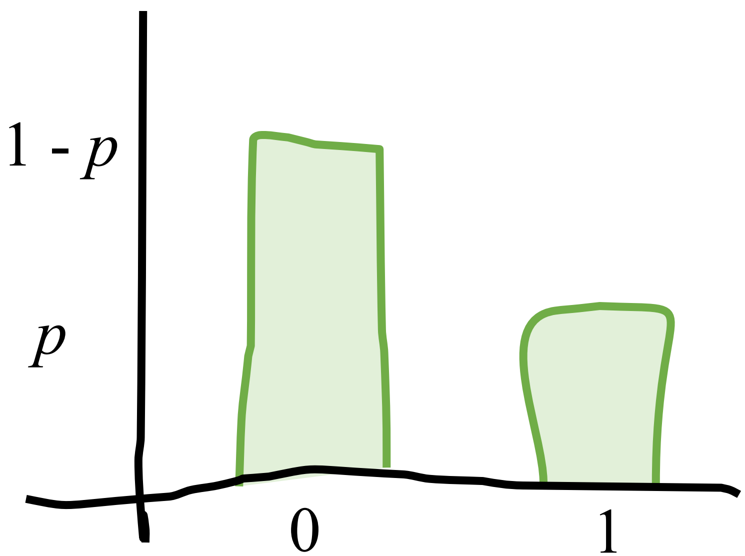
Maximizing Likelihood with Bernoulli

- Consider I.I.D. random variables $X_1, X_2, ..., X_n$
  - $X_i \sim \text{Ber}(p)$
  - Probability mass function, $f(X_i \mid p)$:

Maximizing Likelihood with Bernoulli

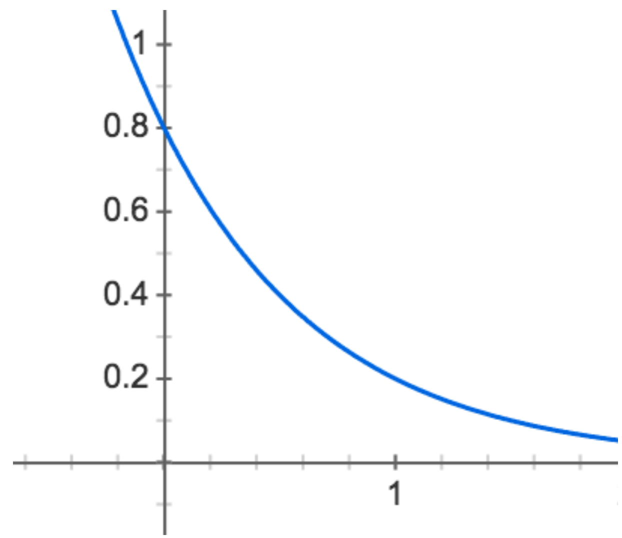
- Consider I.I.D. random variables $X_1, X_2, \ldots, X_n$
  - $X_i \sim \text{Ber}(p)$
  - Probability mass function, $f(X_i \mid p)$:

PMF of Bernoulli



$$f(X_i \mid p) = p^{x_i}(1-p)^{1-x_i}$$

PMF of Bernoulli (p = 0.2)



$$f(x) = 0.2^x(1-0.2)^{1-x}$$

Bernoulli PMF

$$X \sim \mathrm{Ber}(p)$$

$$f(X = x|p) = p^x(1-p)^{1-x}$$

Maximizing Likelihood with Bernoulli

- Consider I.I.D. random variables $X_1, X_2, \ldots, X_n$
  - $X_i \sim \text{Ber}(p)$
  - Probability mass function, $f(X_i \mid p)$, can be written as:

$$f(X_i \mid p) = p^{x_i}(1-p)^{1-x_i} \quad \text{where } x_i = 0 \text{ or } 1$$

  - Likelihood: $L(\theta) = \prod_{i=1}^{n} p^{X_i}(1-p)^{1-X_i}$
  - Log-likelihood:

$$LL(\theta) = \sum_{i=1}^{n} \log(p^{X_i}(1-p)^{1-X_i}) = \sum_{i=1}^{n} \left[X_i(\log p) + (1-X_i)\log(1-p)\right]$$

$$= Y(\log p) + (n-Y)\log(1-p) \quad \text{where } Y = \sum_{i=1}^{n} X_i$$

  - Differentiate w.r.t. p, and set to 0:

$$\frac{\partial LL(p)}{\partial p} = Y\frac{1}{p} + (n-Y)\frac{-1}{1-p} = 0 \quad \Rightarrow \quad p_{MLE} = \frac{Y}{n} = \frac{1}{n}\sum_{i=1}^{n} X_i$$

Isn't that the same as unbiased estimator?

Yes. For Bernoulli.

Maximum Likelihood Algorithm

1. Decide on a model for the distribution of your samples. Define the PMF / PDF for your sample.

2. Write out the log likelihood function.

3. State that the optimal parameters are the argmax of the log likelihood function.

4. Use an optimization algorithm to calculate argmax

Maximizing Likelihood with Poisson

- Consider I.I.D. random variables X_1, X_2, ..., X_n
  - $X_i \sim \text{Poi}(\lambda)$
  - PMF: $f(X_i \mid \lambda) = \dfrac{e^{-\lambda}\lambda^{x_i}}{x_i!}$ Likelihood: $L(\theta) = \prod_{i=1}^{n} \dfrac{e^{-\lambda}\lambda^{X_i}}{X_i!}$
  - Log-likelihood:

$$LL(\theta) = \sum_{i=1}^{n} \log\left(\frac{e^{-\lambda}\lambda^{X_i}}{X_i!}\right) = \sum_{i=1}^{n} \left[-\lambda \log(e) + X_i \log(\lambda) - \log(X_i!)\right]$$

$$= -n\lambda + \log(\lambda)\sum_{i=1}^{n} X_i - \sum_{i=1}^{n} \log(X_i!)$$

  - Differentiate w.r.t. λ, and set to 0:

$$\frac{\partial LL(\lambda)}{\partial \lambda} = -n + \frac{1}{\lambda}\sum_{i=1}^{n} X_i = 0 \quad \Rightarrow \quad \lambda_{MLE} = \frac{1}{n}\sum_{i=1}^{n} X_i$$

Its so general!

Maximizing Likelihood with Normal

- Consider I.I.D. random variables $X_1, X_2, ..., X_n$
  - $X_i \sim N(\mu, \sigma^2)$
  - PDF: $f(X_i \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(X_i-\mu)^2/(2\sigma^2)}$
  - Log-likelihood:

$$LL(\theta) = \sum_{i=1}^{n} \log\left(\frac{1}{\sqrt{2\pi}\sigma} e^{-(X_i-\mu)^2/(2\sigma^2)}\right) = \sum_{i=1}^{n} \left[-\log(\sqrt{2\pi}\sigma) - (X_i-\mu)^2/(2\sigma^2)\right]$$

  - First, differentiate w.r.t. μ, and set to 0:

$$\frac{\partial LL(\mu,\sigma^2)}{\partial \mu} = \sum_{i=1}^{n} 2(X_i-\mu)/(2\sigma^2) = \frac{1}{\sigma^2}\sum_{i=1}^{n}(X_i-\mu) = 0$$

  - Then, differentiate w.r.t. σ, and set to 0:

$$\frac{\partial LL(\mu,\sigma^2)}{\partial \sigma} = \sum_{i=1}^{n} -\frac{1}{\sigma} + 2(X_i-\mu)^2/(2\sigma^3) = -\frac{n}{\sigma} + \sum_{i=1}^{n}(X_i-\mu)^2/(\sigma^3) = 0$$

Being Normal, Simultaneously

- Now have two equations, two unknowns:

$$\frac{1}{\sigma^2}\sum_{i=1}^{n}(X_i-\mu)=0 \qquad -\frac{n}{\sigma}+\sum_{i=1}^{n}(X_i-\mu)^2/(\sigma^3)=0$$

- First, solve for μ_{MLE}:

$$\frac{1}{\sigma^2}\sum_{i=1}^{n}(X_i-\mu)=0 \;\Rightarrow\; \sum_{i=1}^{n}X_i=n\mu \;\Rightarrow\; \mu_{MLE}=\frac{1}{n}\sum_{i=1}^{n}X_i$$

- Then, solve for σ^2_{MLE}:

$$-\frac{n}{\sigma}+\sum_{i=1}^{n}(X_i-\mu)^2/(\sigma^3)=0 \;\Rightarrow\; n\sigma^2=\sum_{i=1}^{n}(X_i-\mu)^2$$

$$\sigma^2_{MLE}=\frac{1}{n}\sum_{i=1}^{n}(X_i-\mu_{MLE})^2$$

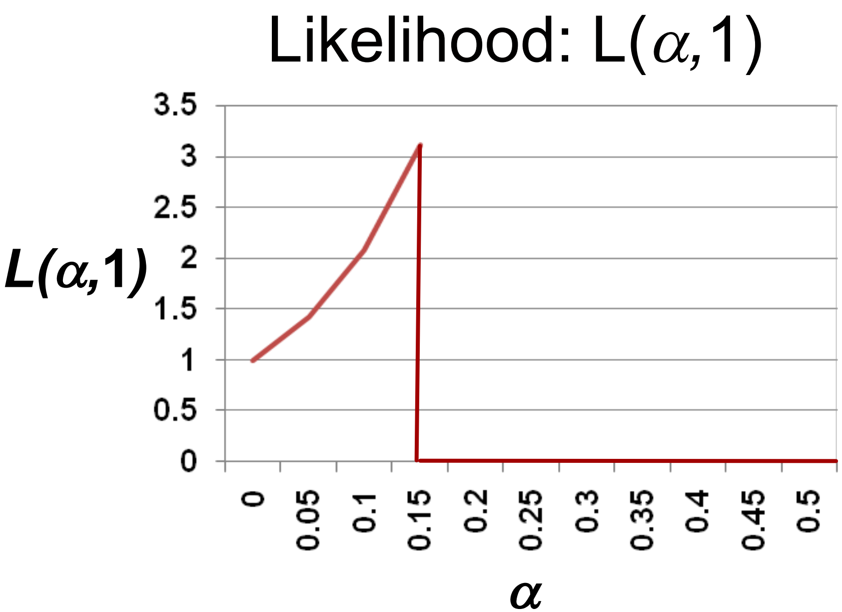
- Note: μ_{MLE} unbiased, but σ^2_{MLE} biased

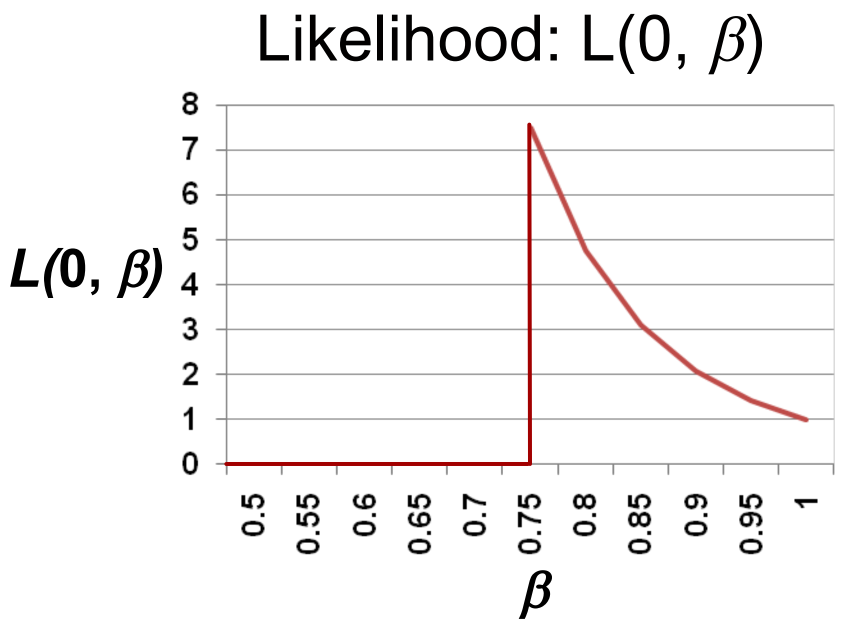
Maximizing Likelihood with Uniform

- Consider I.I.D. random variables $X_1, X_2, ..., X_n$
  - $X_i \sim \text{Uni}(\alpha, \beta)$
  - PDF: $f(X_i \mid \alpha, \beta) = \begin{cases} \frac{1}{\beta - \alpha} & \alpha \le x_i \le \beta \\ 0 & \text{otherwise} \end{cases}$
  - Likelihood: $L(\theta) = \begin{cases} \left(\frac{1}{\beta - \alpha}\right)^n & \alpha \le x_1, x_2, ..., x_n \le \beta \\ 0 & \text{otherwise} \end{cases}$
    - Constraint $\alpha \le x_1, x_2, \ldots, x_n \le \beta$ makes differentiation tricky
    - Intuition: want interval size $(\beta - \alpha)$ to be as small as possible to maximize likelihood function for each data point
    - But need to make sure all observed data contained in interval
      - If all observed data not in interval, then $L(\theta) = 0$
  - Solution: $\alpha_{MLE} = \min(x_1, \ldots, x_n)$ $\beta_{MLE} = \max(x_1, \ldots, x_n)$

Understanding MLE with Uniform

- Consider I.I.D. random variables $X_1, X_2, \ldots, X_n$
  - $X_i \sim \text{Uni}(0, 1)$
  - Observe data:
    - 0.15, 0.20, 0.30, 0.40, 0.65, 0.70, 0.75





Small Samples = Problems

- How do small samples affect MLE?
  - In many cases, $\mu_{MLE} = \frac{1}{n}\sum_{i=1}^{n} X_i$ = sample mean
    - Unbiased. Not too shabby…
  - As seen with Normal, $\sigma^2_{MLE} = \frac{1}{n}\sum_{i=1}^{n}(X_i - \mu_{MLE})^2$
    - Biased. Underestimates for small *n* (e.g., 0 for n = 1)
  - As seen with Uniform, $\alpha_{MLE} \geq \alpha$ and $\beta_{MLE} \leq \beta$
    - Biased. Problematic for small *n* (e.g., $\alpha = \beta$ when n = 1)
  - Small sample phenomena intuitively make sense:
    - Maximum likelihood $\Rightarrow$ best explain data we've seen
    - Does not attempt to generalize to unseen data

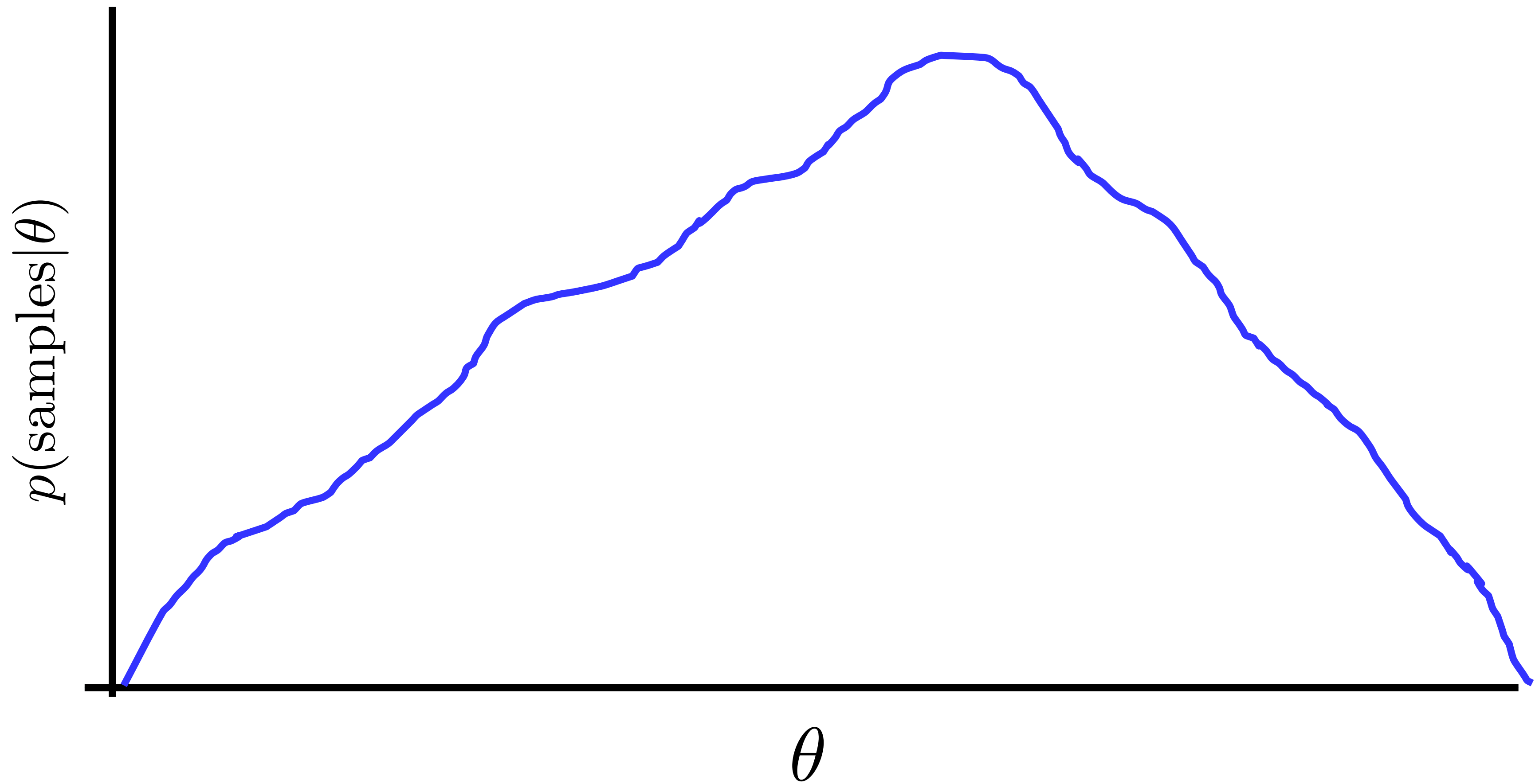
Properties of MLE

- Maximum Likelihood Estimators are generally:
  - Consistent: $\lim_{n\to\infty} P(|\hat{\theta}-\theta|<\varepsilon)=1$ for $\varepsilon > 0$
  - Potentially biased (though asymptotically less so)
  - Asymptotically optimal
    - Has smallest variance of "good" estimators for large samples
  - Often used in practice where sample size is large relative to parameter space
    - But be careful, there are some very large parameter spaces

Argmax?

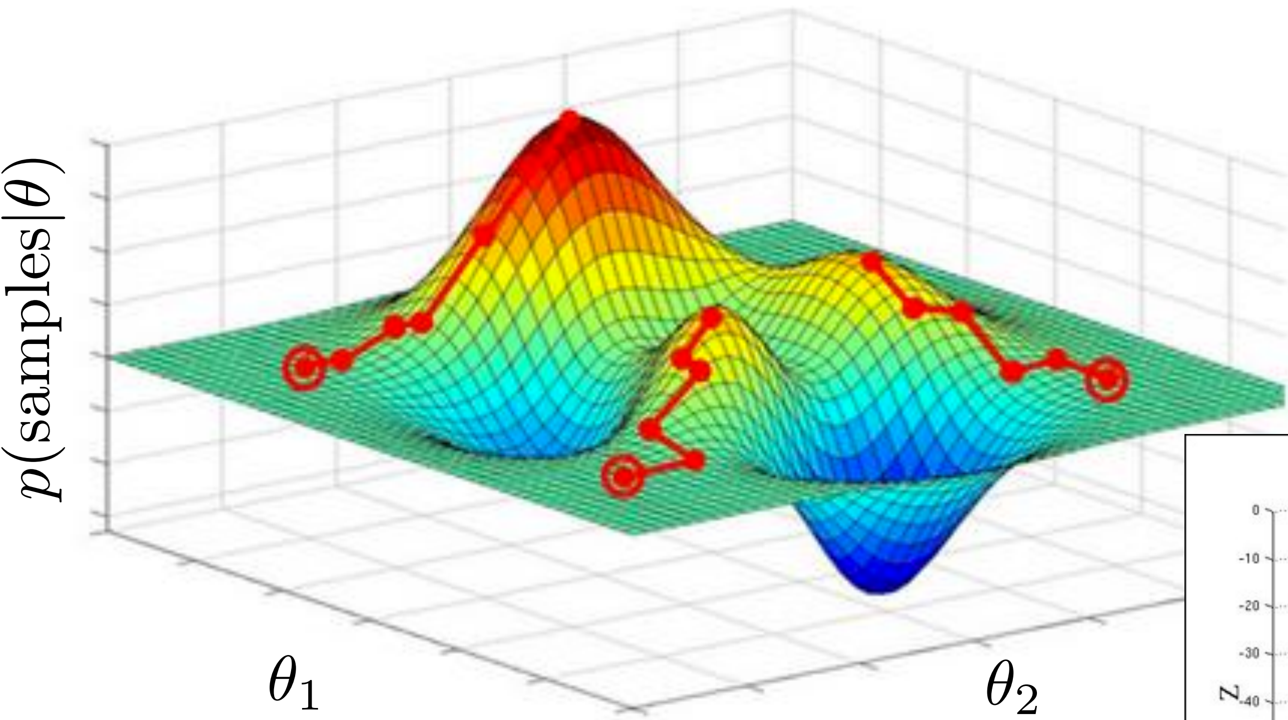
Option #2: Gradient Ascent

Gradient Ascent

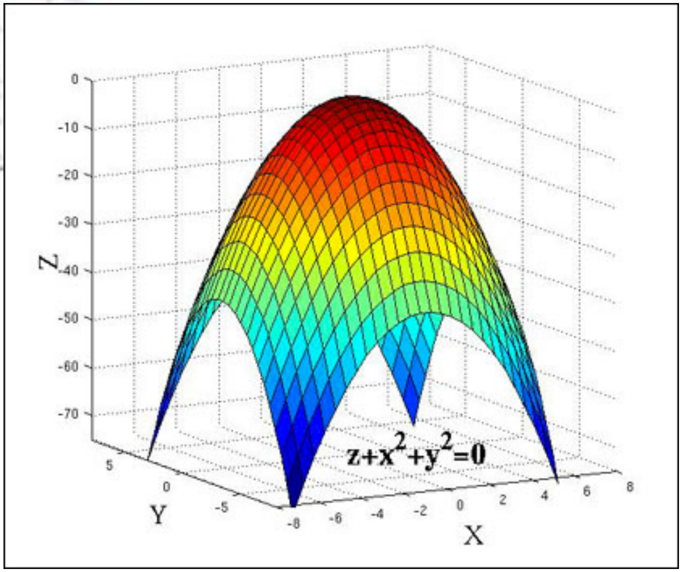


Walk uphill and you will find a local maxima
(if your step size is small enough)

Gradient Ascent



Especially good if function is convex



Walk uphill and you will find a local maxima
(if your step size is small enough)

Gradient ascent is your bread and butter algorithm for optimization (eg argmax)

Gradient Ascent

Initialize: $\theta_j = 0$ **for all** $0 \leq j \leq m$

Calculate all θ_j

Gradient Ascent

`Initialize:` θ_j `= 0 for all 0` $\leq j \leq m$

`Repeat many times:`

`gradient[j] = 0 for all 0` $\leq j \leq m$

Calculate all `gradient[j]`*'s based on data*

θ_j `+=` η `* gradient[j] for all 0` $\leq j \leq m$

Gradient Ascent

```
Initialize: θ_j = 0 for all 0 ≤ j ≤ m

Repeat many times:
    gradient[j] = 0 for all 0 ≤ j ≤ m

    For each training example (x, y):
        For each parameter j:
            Update gradient[j] for current training example

    θ_j += η * gradient[j] for all 0 ≤ j ≤ m
```

Linear Regression

Predicting Warriors

X_1 = Opposing team ELO

X_2 = Points in last game

X_3 = Curry playing?

X_4 = Playing at home?

Y = Warriors points

Predicting CO_2 (simple)

X = CO_2 level

Y = Average Global Temperature

N training datapoints

$$(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \ldots (\mathbf{x}^{(n)}, y^{(n)})$$

Linear Regression Model

$$Y = \theta \cdot X + Z \qquad Z \sim N(0, \sigma^2) \qquad Y|X \sim N(\theta X, \sigma^2)$$

Linear Regression (simple)

`Initialize:` θ `= 0`

`Repeat many times:`

`gradient = 0`

`For each training example (x, y):`

Update `gradient` *for current training example*

θ `+=` η `* gradient`

Linear Regression (simple)

```
Initialize: θ = 0

Repeat many times:
    gradient = 0
    For each training example (x, y):
        gradient += (y - θ x)(-x)
    θ += η * gradient
```

Linear Regression

Predicting CO_2

X_1 = Temperature

X_2 = Elevation

X_3 = CO_2 level yesterday

X_4 = GDP of region

X_5 = Acres of forest growth

Y = CO_2 levels

Linear Regression

Problem: Predict real value Y based on observing variable X

Model: Linear weight every feature

$$
\begin{aligned}
\hat{Y} &= \theta_1 X_1 + \cdots + \theta_m X_m + \theta_{m+1} \\
&= \theta^T \mathbf{X}
\end{aligned}
$$

Training: Gradient ascent to chose the best thetas to describe your data

$$
\hat{\theta}_{MLE} = \underset{\theta}{\operatorname{argmax}} \; -\sum_{i=1}^{n} (Y^{(i)} - \theta^T \mathbf{x}^{(i)})^2
$$

Linear Regression

Initialize: $\theta_j = 0$ **for all** $0 \le j \le m$

Repeat many times:

gradient[j] = 0 for all $0 \le j \le m$

For each training example (x, y):

For each parameter j:

gradient[j] += $(y - \theta^T x)(-x[j])$

θ_j **+=** η *** gradient[j] for all** $0 \le j \le m$

Predicting Warriors

Y = Warriors points

$$\hat{Y} = \theta_1 X_1 + \cdots + \theta_m X_m + \theta_{m+1}$$
$$= \theta^T \mathbf{X}$$

X_1 = Opposing team ELO — θ_1 = -2.3

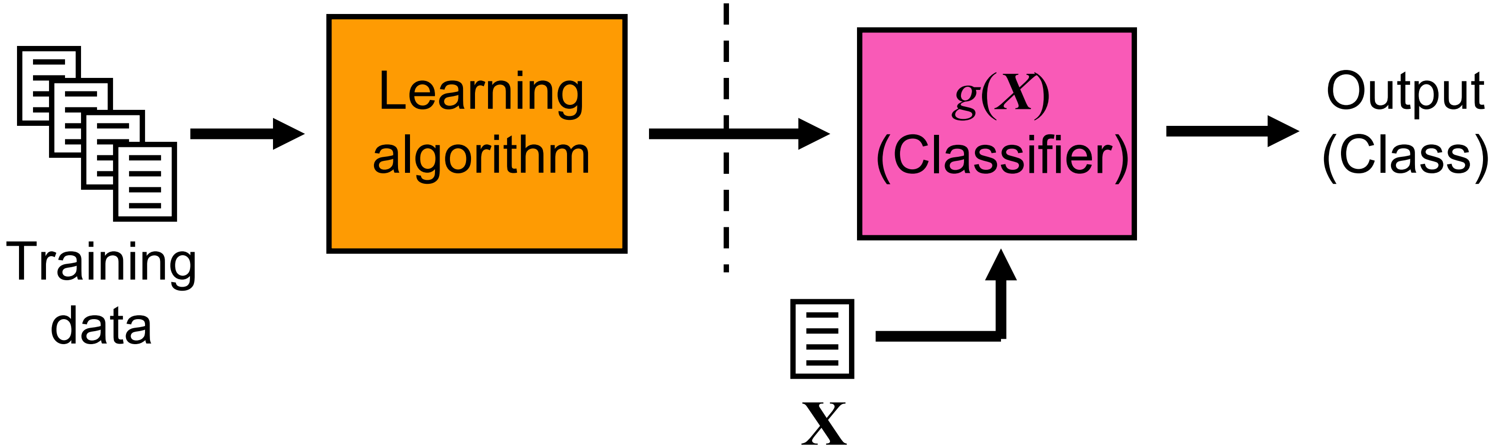
X_2 = Points in last game — θ_2 = +1.2

X_3 = Curry playing? — θ_3 = +10.2

X_4 = Playing at home? — θ_4 = +3.3

θ_5 = +95.4

The Machine Learning Process



- Training data: set of N pre-classified data instances
  - N training pairs: $(\boldsymbol{x}^{(1)}, y^{(1)}), (\boldsymbol{x}^{(2)}, y^{(2)}), \ldots, (\boldsymbol{x}^{(n)}, y^{(n)})$
    - Use superscripts to denote *i*-th training instance
- Learning algorithm: method for determining $g(\boldsymbol{X})$
  - Given a new input observation of $\boldsymbol{x} = x_1, x_2, \ldots, x_m$
  - Use $g(\boldsymbol{x})$ to compute a corresponding output (prediction)