CSE 446
Linear Regression

Natasha Jaques

- *A ML workhorse*
- *Easy/fast to fit*
- *Interpretable*

UNIVERSITY of WASHINGTON

Recap: Maximum Likelihood Estimation

Observe $X_1, X_2, \ldots, X_n$ drawn IID from $f(x;\theta)$ for some "true" $\theta = \theta_*$

Likelihood function $L_n(\theta) = \prod_{i=1}^{n} f(X_i;\theta)$

Log-Likelihood function $l_n(\theta) = \log(L_n(\theta)) = \sum_{i=1}^{n} \log(f(X_i;\theta))$

Maximum Likelihood Estimator (MLE) $\widehat{\theta}_{MLE} = \arg\max_{\theta} L_n(\theta)$

Recap: MLE to learn Gaussian parameters

- MLE:

$$\widehat{\mu}_{MLE} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\widehat{\sigma^2}_{MLE} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \widehat{\mu}_{MLE})^2$$

- MLE for the variance of a Gaussian is **biased**

$$\mathbb{E}[\widehat{\sigma^2}_{MLE}] \neq \sigma^2$$

- Unbiased variance estimator:

$$\widehat{\sigma^2}_{unbiased} = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \widehat{\mu}_{MLE})^2$$

Recap: Maximum Likelihood Estimation

ML “algorithms” we’ve learned so far:

- Maximum Likelihood Estimation (MLE)
  - Fit a Bernoulli distribution (coin flips)
  - Fit a Gaussian distribution (μ, σ)

- Now: fit a linear predictor $x \rightarrow y$, where y is a continuous variable

Maximum Likelihood Estimation

Observe $X_1, X_2, \ldots, X_n$ drawn IID from $f(x;\theta)$ for some "true" $\theta = \theta_*$

Likelihood function $L_n(\theta) = \prod_{i=1}^{n} f(X_i;\theta)$

prob of dataset given **model** parameters

Log-Likelihood function $l_n(\theta) = \log(L_n(\theta)) = \sum_{i=1}^{n} \log(f(X_i;\theta))$

Maximum Likelihood Estimator (MLE) $\widehat{\theta}_{MLE} = \arg\max_{\theta} L_n(\theta)$

- *We will use the same recipe as last time, to fit Linear Regression with MLE*
- *What part do we need to change?*

The MLE is a "recipe" that begins with a *model* for data $f(x;\theta)$

Regression

The regression problem, 1-dimensional

Given past sales data on zillow.com, predict:

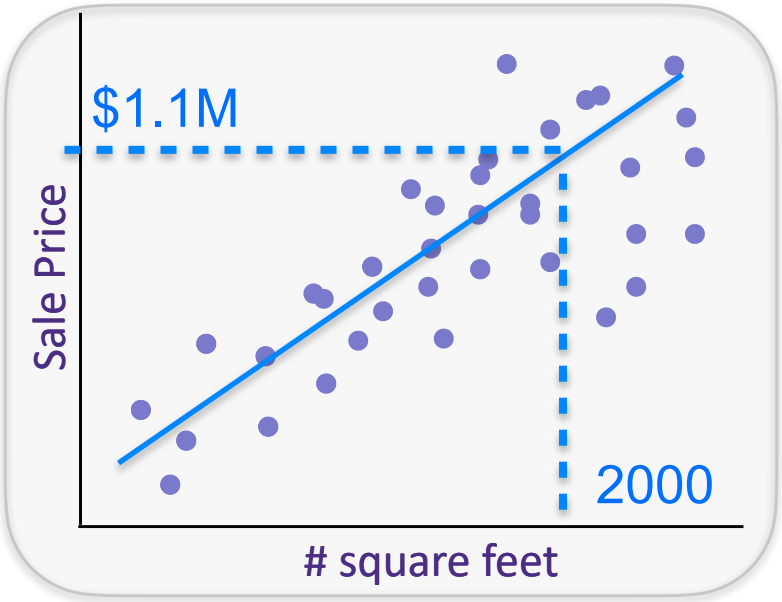
y = **House sale price** *from*

x = **{# sq. ft.}**

Goal: learn p(y|x)

now we have label y

previously we just had x



Training Data:

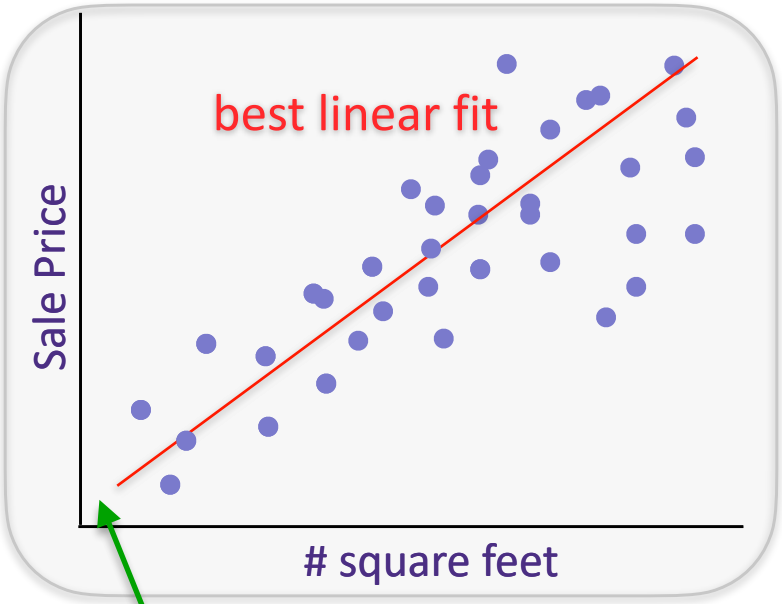
$$\{(x_i, y_i)\}_{i=1}^{n} \qquad x_i \in \mathbb{R} \quad y_i \in \mathbb{R}$$

Fit a function to our data, 1-d

Given past sales data on zillow.com, predict:

y = **House sale price** *from*

x = **{# sq. ft.}**



Training Data: $\{(x_i, y_i)\}_{i=1}^{n}$ $x_i \in \mathbb{R}$ $y_i \in \mathbb{R}$

Hypothesis/Model: linear

$$y_i = x_i w + \epsilon_i \qquad \epsilon_i \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma^2)$$

$w \in \mathbb{R}$ *slope of line*

- why the noise?

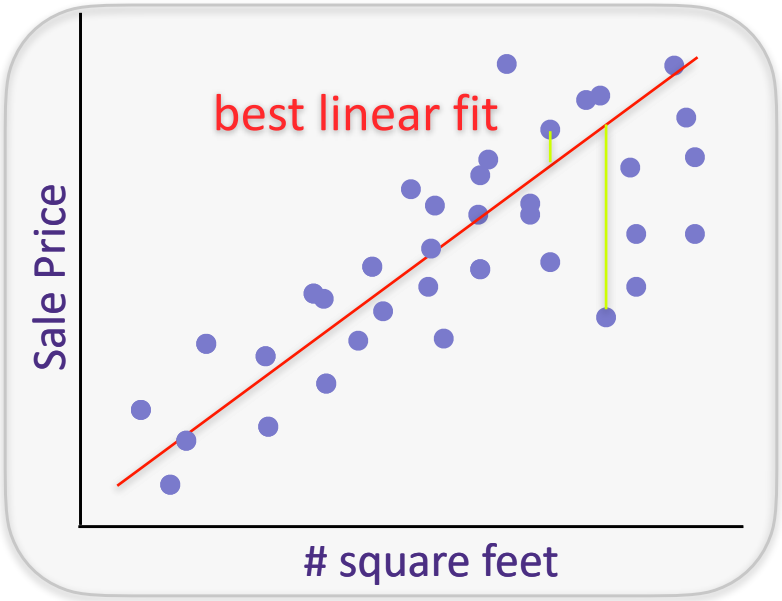
(no intercept for now to make the math easier)

Fit a function to our data, 1-d

Given past sales data on zillow.com, predict:

y = **House sale price** *from*

x = **{# sq. ft.}**



Training Data: $\{(x_i, y_i)\}_{i=1}^{n}$ $x_i \in \mathbb{R}$ $y_i \in \mathbb{R}$

Hypothesis/Model: linear

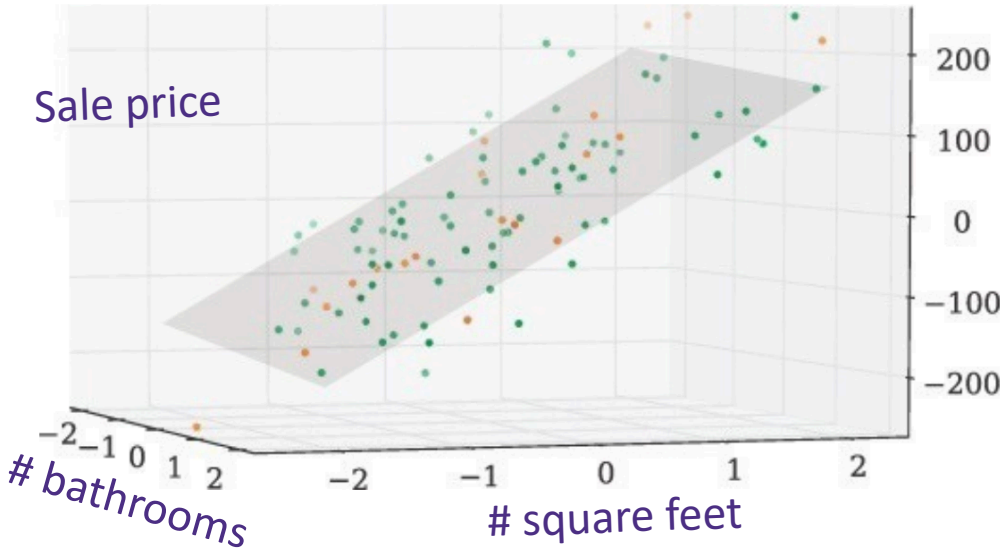
$$y_i = x_i w + \epsilon_i \qquad \epsilon_i \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma^2)$$

The regression problem, d-dim

Given past sales data on zillow.com, predict:

y = **House sale price** *from*

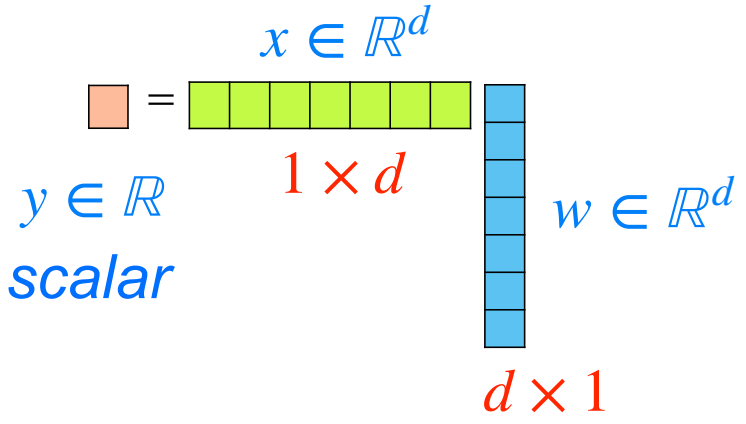
x = **{# sq. ft., # baths, date of sale, etc.}** $x \in \mathbb{R}^d$



Training Data: $\{(x_i, y_i)\}_{i=1}^n$ $x_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$

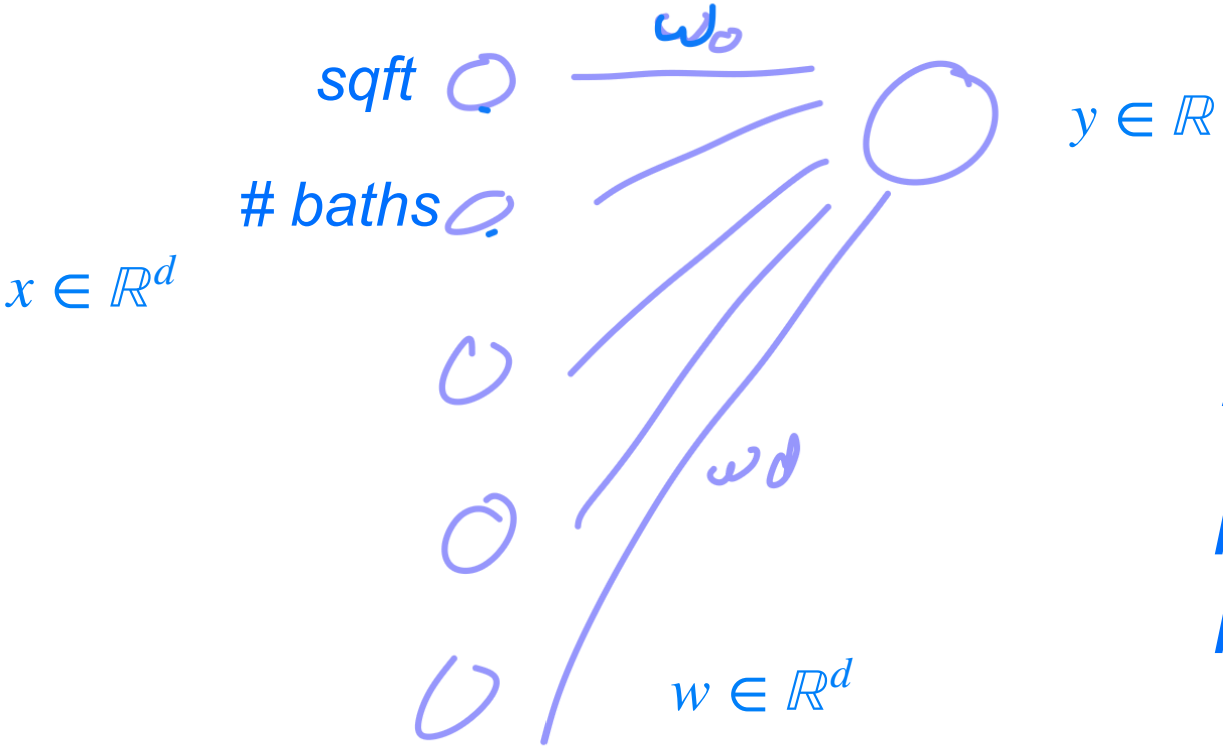
Hypothesis/Model: linear

$$y_i = x_i^T w + \epsilon_i \qquad \epsilon_i \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma^2)$$



Linear regression with d features

What simple model is this equivalent to?



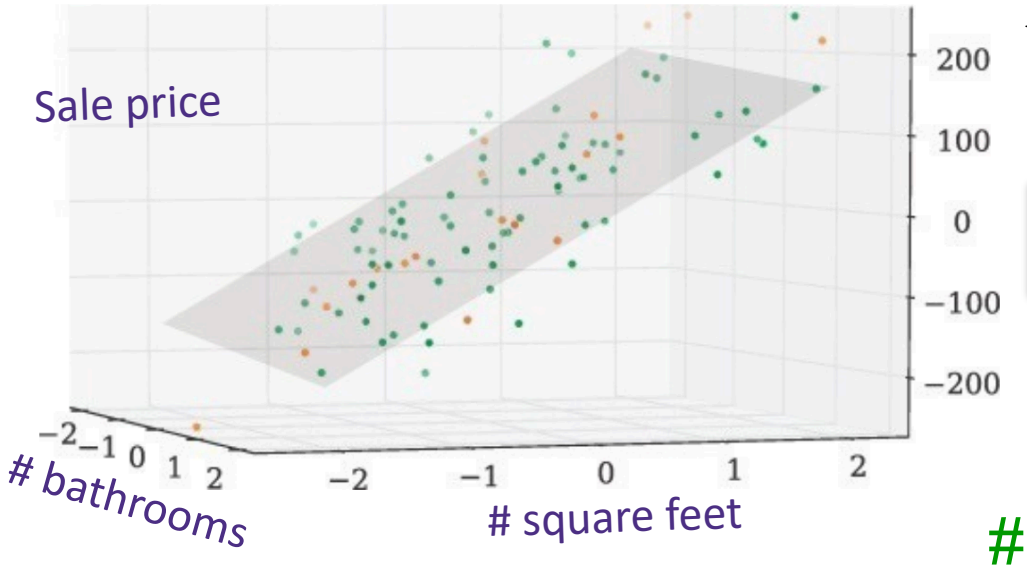
A 1 layer neural network!

The regression problem, d-dim

Given past sales data on zillow.com, predict:

y* = House sale price *from

***x* = {# sq. ft., # baths, date of sale, etc.}**



Training Data: $\{(x_i, y_i)\}_{i=1}^n$ $x_i \in \mathbb{R}^d$ $y_i \in \mathbb{R}$

Hypothesis/Model: linear

$$y_i = x_i^T w + \epsilon_i \qquad \epsilon_i \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma^2)$$

$$p(y|x, w, \sigma) = \mathcal{N}(x^T w, \sigma^2)$$

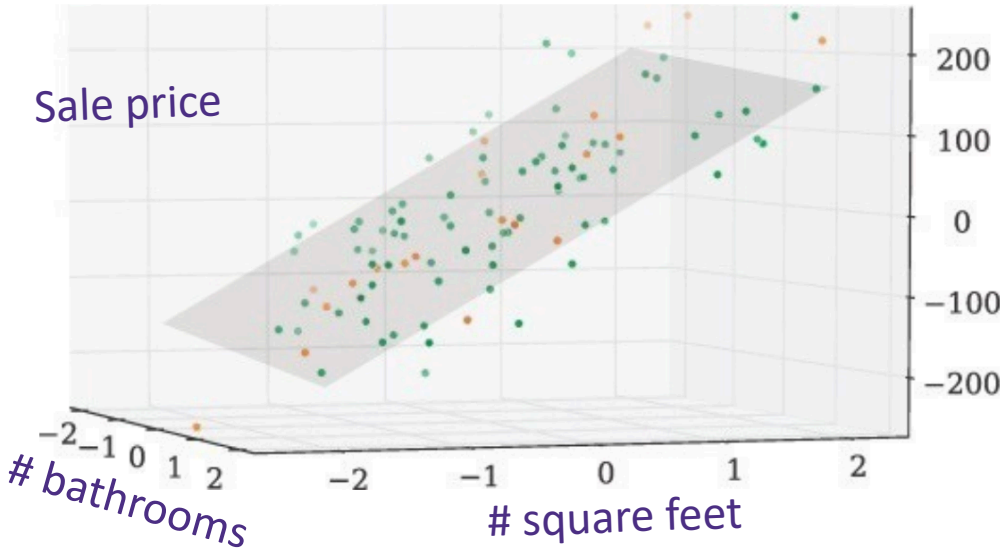
go back to formula sheet, retrieve PDF of Gaussian…

The regression problem, d-dim

Given past sales data on zillow.com, predict:

y* = House sale price *from

***x* = {# sq. ft., # baths, date of sale, etc.}**



Training Data: $\{(x_i, y_i)\}_{i=1}^n$ $x_i \in \mathbb{R}^d$ $y_i \in \mathbb{R}$

Hypothesis/Model: linear

$$y_i = x_i^T w + \epsilon_i \qquad \epsilon_i \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma^2)$$

$$p(y|x, w, \sigma) = \mathcal{N}(x^T w, \sigma^2)$$

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

Maximizing log-likelihood

Training Data: $\{(x_i, y_i)\}_{i=1}^n$ $x_i \in \mathbb{R}^d$ $y_i \in \mathbb{R}$

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

Likelihood: $$P(\mathcal{D}|w, \sigma) = \prod_{i=1}^n p(y_i|x_i, w, \sigma) = \prod_{i=1}^n \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(y_i - x_i^\top w)^2/2\sigma^2}$$

Yay, we have a new likelihood! Now, how do we find MLE?

Maximizing log-likelihood

Training Data: $\{(x_i, y_i)\}_{i=1}^n$ $x_i \in \mathbb{R}^d$ $y_i \in \mathbb{R}$

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

Likelihood: $$P(\mathcal{D}|w, \sigma) = \prod_{i=1}^n p(y_i|x_i, w, \sigma) = \prod_{i=1}^n \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(y_i - x_i^\top w)^2/2\sigma^2}$$

Maximize (wrt w): $$\log P(\mathcal{D}|w, \sigma) = \log \left(\prod_{i=1}^n \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(y_i - x_i^\top w)^2/2\sigma^2} \right)$$ # take log

first manipulate log likelihood to make it easier to work with

$$= \sum_{i=1}^n \log \left[\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(- \frac{(y_i - x_i^T w)^2}{2\sigma^2} \right) \right]$$

what is log of product?

$$\log AB = \log A + \log B$$

Maximizing log-likelihood

Maximize (wrt w): $\log P(\mathcal{D}|w,\sigma) = \log\left(\prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(y_i - x_i^\top w)^2/2\sigma^2}\right)$ # take log

first manipulate log likelihood to make it easier to work with

$$= \sum_{i=1}^{n} \log\left[\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_i - x_i^T w)^2}{2\sigma^2}\right)\right]$$

$$= n\log\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) - \sum_{i=1}^{n} \frac{(y_i - x_i^T w)^2}{2\sigma^2}$$

pull out terms that don't depend on i

we're going to max wrt w, so
get rid of terms that don't depend on w

$$\arg\max_w \; -\sum_{i=1}^{n} \frac{(y_i - x_i^T w)^2}{2\sigma^2}$$

We found our simplified LL!

Now wat do?

Maximizing a negative?

Maximizing log-likelihood

Training Data: $\{(x_i, y_i)\}_{i=1}^n$ $\quad x_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$

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

Likelihood: $$P(\mathcal{D}|w, \sigma) = \prod_{i=1}^n p(y_i|x_i, w, \sigma) = \prod_{i=1}^n \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(y_i - x_i^\top w)^2/2\sigma^2}$$

Maximize (wrt w): $$\log P(\mathcal{D}|w, \sigma) = \log \left(\prod_{i=1}^n \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(y_i - x_i^\top w)^2/2\sigma^2} \right)$$

$$\widehat{w}_{MLE} = \arg\min_w \sum_{i=1}^n (y_i - x_i^\top w)^2$$

Maximizing log-likelihood → minimizing mean squared error (MSE)

$$\widehat{w}_{MLE} = \arg\min_{w} \sum_{i=1}^{n} (y_i - x_i^\top w)^2$$

- Wow, minimizing squared error seems like a reasonable thing to do to fit a predictor!
  - When noise is Gaussian, maximizing likelihood of our linear model is equivalent to minimizing the sum of squared errors. We get this from cranking through the math.
  - However, if we had chosen a different noise distribution, we'd get a different estimator. E.g. Laplacian noise -> minimize absolute error
  - Which one is better depends on assumptions about our data. Gaussian has no long tails, mean squared error incurs high penalty for outliers
  - **TL;DR: Model of data <—math—> what to optimize**

Maximizing log-likelihood

So how do we actually optimize this / fit our model?

$$\widehat{w}_{MLE} = \arg\min_{w} \sum_{i=1}^{n} (y_i - x_i^\top w)^2$$

Set gradient=0, solve for w

Recall: $w \in \mathbb{R}^d$

$$\nabla_w f(x) = \begin{bmatrix} \frac{\partial}{\partial w_0} f(x) \\ \frac{\partial}{\partial w_1} f(x) \\ \vdots \end{bmatrix}$$

$$\nabla_w \sum_{i=1}^{n} (y_i - x_i^T w)^2$$

Gradient wrt w is a vector of partial derivatives

$$= \sum_{i=1}^{n} 2(y_i - x_i^T w)^1 \nabla_w (y_i - x_i^T w)$$

Chain rule $\nabla_w f(x)^2 = 2f(x) \cdot \nabla_w f(x)$

$$= \sum_{i=1}^{n} 2(y_i - x_i^T w)(-x_i)$$

"Matrix Cookbook" identities (cheatsheet fodder)

$$\nabla_w x^T w = x$$

Maximizing log-likelihood

Set gradient=0, solve for w

$$\nabla_w \sum_{i=1}^{n} (y_i - x_i^T w)^2 = \sum_{i=1}^{n} \underbrace{2(y_i - x_i^T w)(-x_i)}_{d \times 1} = 0$$

$1 \times 1 \quad 1 \times 1 \quad d \times 1 \qquad d \times 1$

check dimensions!

$\vec{0} \in \mathbb{R}^d$

$$= 2\sum_{i=1}^{n} [-y_i x_i + x_i^T w x_i] = 0$$

Now for some tricks

$x^T w = a$ # scalar

$a\vec{v} = \vec{v}a$

$$= 2\sum_{i=1}^{n} [-x_i y_i + x_i x_i^T w] = 0$$

Reminder

$x^T x$ = inner product or dot product 1x1

xx^T = outer product dxd

$$\sum_{i=1}^{n} x_i y_i = \left(\sum_{i=1}^{n} x_i x_i^T\right)^{-1} w$$

$d \times d$

A matrix

How can we isolate w?

Take the inverse!

Maximizing log-likelihood

Set gradient=0, solve for w

$$\sum_{i=1}^{n} x_i y_i = \left(\sum_{i=1}^{n} x_i x_i^T\right)^{-1} w$$

$d \times d$

A matrix

How can we isolate w?

Take the inverse!

$AA^{-1} = I$

$$\left(\sum_{i=1}^{n} x_i x_i^T\right)^{-1} \sum_{i=1}^{n} x_i y_i = \hat{w}_{\text{MLE}}$$

The equation for fitting our model parameters w to our data

- What constraints does taking the inverse place on our data?
  - Matrix must be invertible
  - n >= d
- For this model we can analytically derive how to find the optimal weights. Will not always be true for more complex models

Maximizing log-likelihood

$$\widehat{w}_{MLE} = \arg\min_{w} \sum_{i=1}^{n} (y_i - x_i^\top w)^2$$

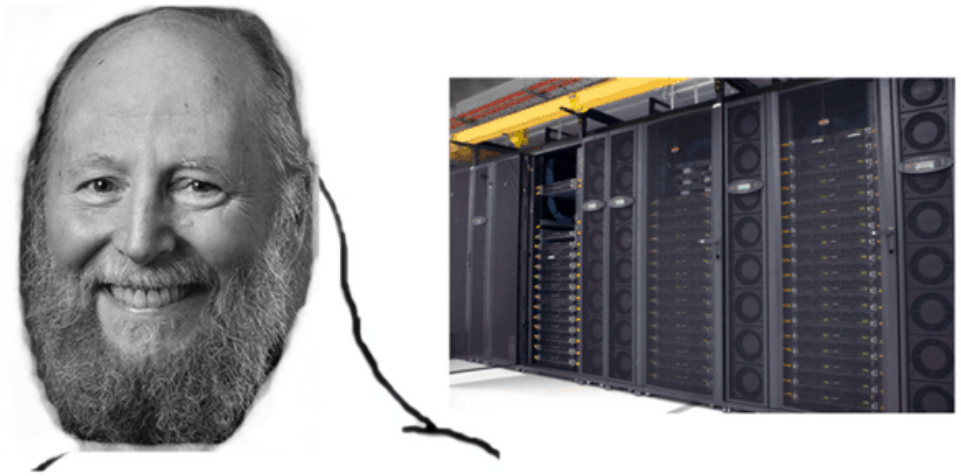
Set gradient=0, solve for w

$$\widehat{w}_{MLE} = \left(\sum_{i=1}^{n} x_i x_i^\top \right)^{-1} \sum_{i=1}^{n} x_i y_i$$

The regression problem in matrix notation

We ❤️ matrix notation!

$$\widehat{w}_{MLE} = \arg\min_{w} \sum_{i=1}^{n} (y_i - x_i^\top w)^2$$



haha gpus go brrrrr

The regression problem in matrix notation

$$\widehat{w}_{MLE} = \arg\min_{w} \sum_{i=1}^{n} (y_i - x_i^\top w)^2$$

We ❤️ matrix notation!

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \qquad \mathbf{X} = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix} \begin{matrix} d \times 1 \\ d \times 1 \\ \\ d \times 1 \end{matrix}$$

d : # of features

n : # of examples/datapoints

$x \in \mathbb{R}^d$

The regression problem in matrix notation

$$\widehat{w}_{MLE} = \arg\min_{w} \sum_{i=1}^{n} (y_i - x_i^\top w)^2$$

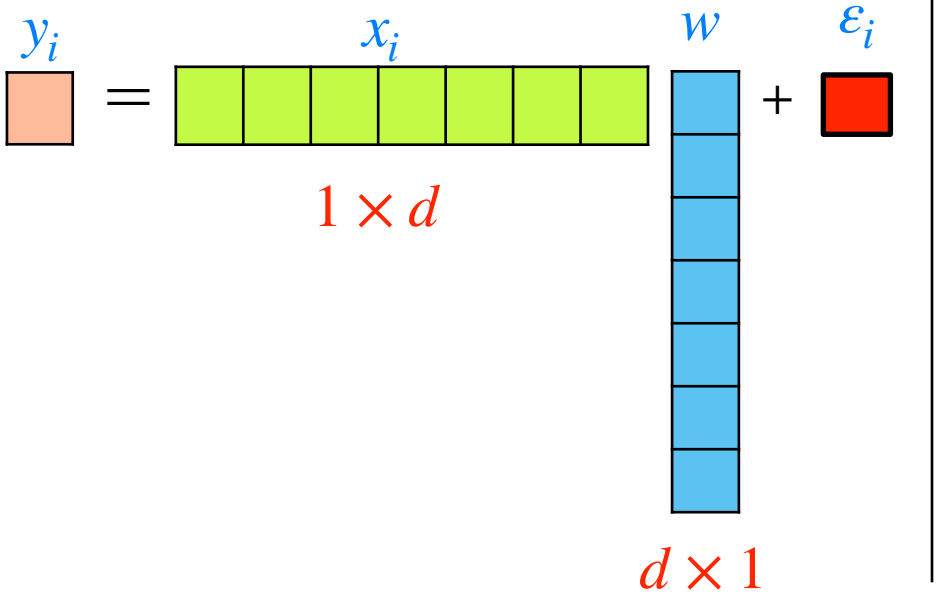
$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \qquad \mathbf{X} = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix} \begin{matrix} d \times 1 \\ d \times 1 \\ \\ d \times 1 \end{matrix}$$

d : # of features

n : # of examples/datapoints

$x \in \mathbb{R}^d$

Previously… $y_i = x_i^T w + \epsilon_i$



The regression problem in matrix notation

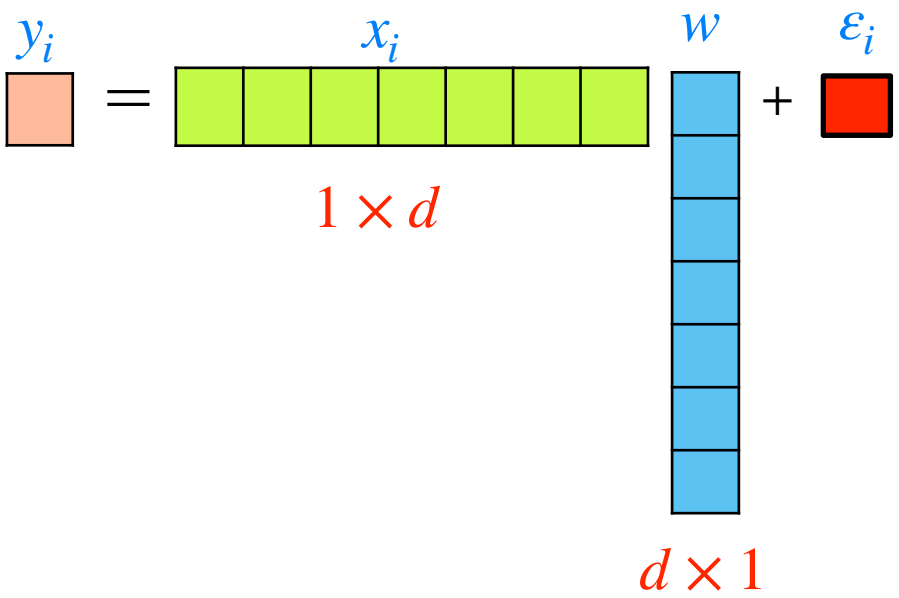
$$\widehat{w}_{MLE} = \arg\min_{w} \sum_{i=1}^{n} (y_i - x_i^\top w)^2$$

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \qquad \mathbf{X} = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix}$$

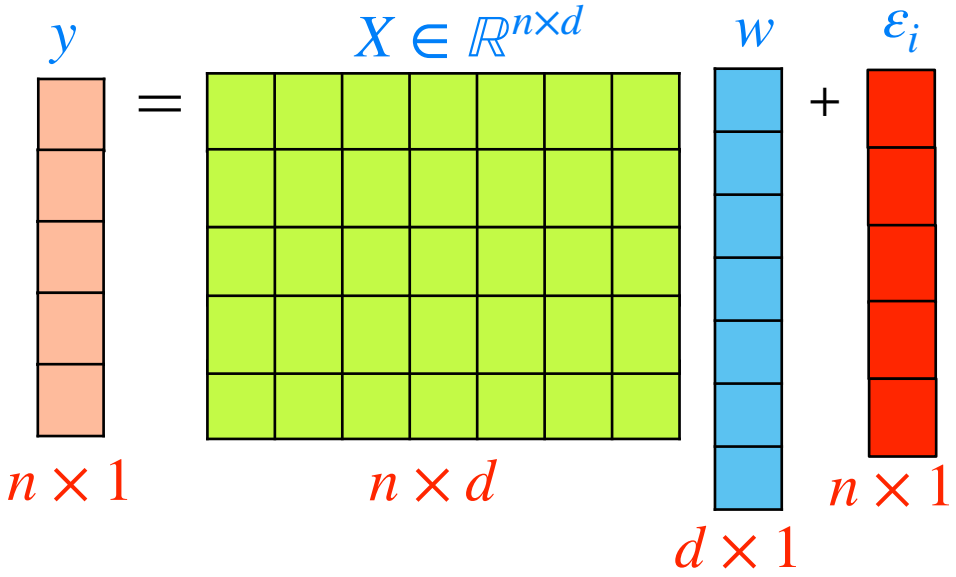
d : # of features

n : # of examples/datapoints

Previously… $y_i = x_i^T w + \epsilon_i$



Now: $\mathbf{y} = \mathbf{X}w + \epsilon$



The regression problem in matrix notation

$$\widehat{w}_{MLE} = \arg\min_w \sum_{i=1}^{n} (y_i - x_i^\top w)^2$$

How to make matrix-y?

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \quad \mathbf{X} = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix}$$

d : # of features

n : # of examples/datapoints

Now: $\mathbf{y} = \mathbf{X}w + \epsilon$

More identities: ℓ_2 norm: $\|z\|_2 = \sqrt{\sum_{i=1}^n z_i^2} = \sqrt{z^\top z}$

Let $z_i = (y_i - x_i^T w)$ Then

$$\widehat{w}_{LS} = \arg\min_w ||\mathbf{y} - \mathbf{X}w||_2^2$$

$$= \arg\min_w (\mathbf{y} - \mathbf{X}w)^T (\mathbf{y} - \mathbf{X}w)$$

The regression problem in matrix notation

$$= \arg\min_w (\mathbf{y} - \mathbf{X}w)^T (\mathbf{y} - \mathbf{X}w)$$

As before, first simplify the function to make it easier to work with

$$= \arg\min_w \; y^T y - y^T Xw - (Xw)^T y + (Xw)^T (Xw)$$

$$= \arg\min_w \; y^T y - \underbrace{y^T Xw}_{1\times n \; n\times d \; d\times 1} - \underbrace{(w^T X^T y)^T}_{1\times d \; d\times n \; n\times 1} + w^T X^T Xw$$

1×1 $\qquad$ 1×1

$$= \arg\min_w \; - y^T Xw - y^T Xw + w^T X^T Xw$$

$$= \arg\min_w \; - 2y^T Xw + w^T X^T Xw$$

More identities:

$$(ABC)^T = C^T B^T A^T$$

Minimizing wrt w, so?

Check dimensions

$$s \in \mathbb{R} \rightarrow s = s^T$$

Pretty simple!

So now what?

The regression problem in matrix notation

$$= \arg\min_w (\mathbf{y} - \mathbf{X}w)^T (\mathbf{y} - \mathbf{X}w)$$

Take derivative, set to 0!

$$\nabla_w [\underbrace{-2y^T Xw}_{(B)} + \underbrace{w^T X^T Xw}_{(C)}] = 0$$

Useful matrix gradient identities

(A) $\nabla_w x^T w = x$

(B) $\nabla_w x^T Aw = A^T x$

(C) $\nabla_w w^T Aw = (A + A^T)w$

Quadratic form

If A is symmetric ($A = A^T$),
Then $\nabla_w w^T Aw = 2Aw$

$$= -2X^T y + 2X^T Xw = 0$$

Symmetric: $(X^T X)^T = X^T X$

$$X^T Xw = X^T y$$

$$\hat{w}_{\text{MLE}} = (X^T X)^{-1} X^T y$$

The regression problem in matrix notation

$$\widehat{w}_{MLE} = \arg\min_{w} \sum_{i=1}^{n} (y_i - x_i^\top w)^2$$

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \quad \mathbf{X} = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix}$$

d : # of features
n : # of examples/datapoints

More identities: ℓ_2 norm: $\|z\|_2 = \sqrt{\sum_{i=1}^{n} z_i^2} = \sqrt{z^\top z}$

$$\begin{aligned} \widehat{w}_{LS} &= \arg\min_{w} ||\mathbf{y} - \mathbf{X}w||_2^2 \\ &= \arg\min_{w} (\mathbf{y} - \mathbf{X}w)^T (\mathbf{y} - \mathbf{X}w) \end{aligned}$$

$$\hat{w}_{LS} = \hat{w}_{MLE} = \left(\boldsymbol{X}^T \boldsymbol{X}\right)^{-1} \boldsymbol{X}^T \boldsymbol{y}$$

Why LS? “Ordinary Least Squares”

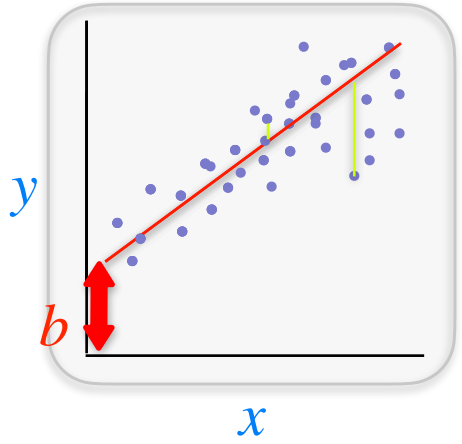
The regression problem in matrix notation

$$\widehat{w}_{MLE} = \left(\sum_{i=1}^{n} x_i x_i^\top \right)^{-1} \sum_{i=1}^{n} x_i y_i$$

$$\hat{w}_{MLE} = \left(\boldsymbol{X}^T \boldsymbol{X} \right)^{-1} \boldsymbol{X}^T \boldsymbol{y}$$

The regression problem in matrix notation

$$\widehat{w}_{LS} = \arg\min_{w} ||\mathbf{y} - \mathbf{X}w||_2^2$$
$$= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$



What about an offset?

$\hat{y}_i = x_i^T w + b$ # Prediction for a single point

$\left(\hat{y}_i - y_i\right)^2$ # Still want to minimize squared error

$$\widehat{w}_{LS}, \widehat{b}_{LS} = \arg\min_{w,b} \sum_{i=1}^{n} \left(y_i - (x_i^T w + b)\right)^2$$
$$= \arg\min_{w,b} ||\mathbf{y} - (\mathbf{X}w + \mathbf{1}b)||_2^2 \qquad b \in \mathbb{R}$$

$n \times 1$

Dealing with an offset

$$\widehat{w}_{LS}, \widehat{b}_{LS} = \arg\min_{w,b} ||\mathbf{y} - (\mathbf{X}w + \mathbf{1}b)||_2^2$$

$$\mathbf{X}^T\mathbf{X}\widehat{w}_{LS} + \widehat{b}_{LS}\mathbf{X}^T\mathbf{1} = \mathbf{X}^T\mathbf{y}$$
$$\mathbf{1}^T\mathbf{X}\widehat{w}_{LS} + \widehat{b}_{LS}\mathbf{1}^T\mathbf{1} = \mathbf{1}^T\mathbf{y}$$

Could think of this as system of equations…

$$\begin{bmatrix} X^TX & X^T1 \\ 1^TX & 1^T1 \end{bmatrix} \begin{bmatrix} w \\ b \end{bmatrix} = \begin{bmatrix} X^Ty \\ 1^Ty \end{bmatrix}$$

$$\begin{bmatrix} w \\ b \end{bmatrix}$$ = np.linalg.solve() # There's a better way…

Dealing with an offset

$$\widehat{w}_{LS}, \widehat{b}_{LS} = \arg\min_{w,b} ||\mathbf{y} - (\mathbf{X}w + \mathbf{1}b)||_2^2$$

$$\mathbf{X}^T\mathbf{X}\widehat{w}_{LS} + \widehat{b}_{LS}\mathbf{X}^T\mathbf{1} = \mathbf{X}^T\mathbf{y}$$

$$\mathbf{1}^T\mathbf{X}\widehat{w}_{LS} + \widehat{b}_{LS}\mathbf{1}^T\mathbf{1} = \mathbf{1}^T\mathbf{y}$$

$$\tilde{x}_i = \begin{bmatrix} x_i \\ 1 \end{bmatrix} \rightarrow \text{Scalar} \in \mathbb{R} \qquad \tilde{\omega} = \begin{bmatrix} w \\ b \end{bmatrix} \qquad \hat{y}_i = \tilde{x}_i^T\tilde{\omega}$$

$$\hat{y} = \tilde{X}^T\tilde{\omega}$$

$$\tilde{X} = \begin{bmatrix} \tilde{x}_i^T \\ \vdots \\ \tilde{x}_n^T \end{bmatrix} = \begin{bmatrix} X & 1 \\ & \end{bmatrix}$$

$$\hat{\tilde{\omega}}_{\text{MLE}} = (\tilde{X}^T\tilde{X})^{-1}\tilde{X}^T y$$

Same equation on modified input matrix containing all 1s feature

Dealing with an offset

$$\widehat{w}_{LS}, \widehat{b}_{LS} = \arg\min_{w,b} ||\mathbf{y} - (\mathbf{X}w + \mathbf{1}b)||_2^2$$

$$\mathbf{X}^T\mathbf{X}\widehat{w}_{LS} + \widehat{b}_{LS}\mathbf{X}^T\mathbf{1} = \mathbf{X}^T\mathbf{y}$$
$$\mathbf{1}^T\mathbf{X}\widehat{w}_{LS} + \widehat{b}_{LS}\mathbf{1}^T\mathbf{1} = \mathbf{1}^T\mathbf{y}$$

It's good practice to pre-process your features to have a mean of zero

If $\mathbf{X}^T\mathbf{1} = 0$ (i.e., if each feature is mean-zero) then

$$\widehat{w}_{LS} = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

$$\widehat{b}_{LS} = \frac{1}{n}\sum_{i=1}^{n} y_i$$

Make Predictions

$$\widehat{w}_{LS} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}$$

$$\widehat{b}_{LS} = \frac{1}{n}\sum_{i=1}^{n} y_i$$

A new house is about to be listed. What should it sell for?

$$\hat{y}_{\text{new}} = x_{\text{new}}^T \hat{w}_{LS} + \hat{b}_{LS}$$

How to normalize features to have mean 0?

For each feature, calc the mean ***of the training data*** and subtract it from each data point

So what do I need to do to x_{new} to be able to make a prediction?

Subtract the mean ***of the training data*** for each feature

Process

Decide on a **model** for the likelihood function $f(x; \theta)$

Find the function which fits the data best

Choose a loss function- least squares

Pick the function which minimizes loss on data

Use function to make prediction on new examples