Indian Institute of Information Technology, Allahabad

Introduction to Machine Learning

Linear Regression

By

Dr. Shiv Ram Dubey
Associate Professor
Computer Vision and Biometrics Lab
Department of Information Technology
Indian Institute of Information Technology, Allahabad
Web: https://profile.iiita.ac.in/srdubey/

DISCLAIMER

The content (text, image, and graphics) used in this slide are adopted from many sources for academic purposes. Broadly, the sources have been given due credit appropriately. However, there is a chance of missing out some original primary sources. The authors of this material do not claim any copyright of such material.

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

MLE: Learning 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:

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

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

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

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

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

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

The Regression Problem, 1-dimensional

Given past sales data on zillow.com, predict: # Goal: learn p(y|x)

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

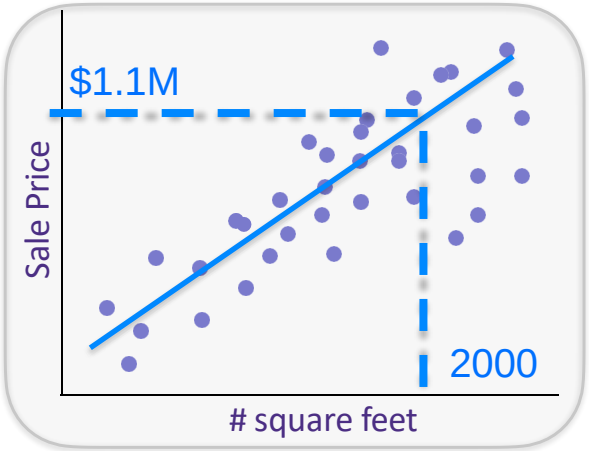
x = **{# sq. ft.}**

now we have label y

previously we just had x

Training Data:

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



The Regression Problem, 1-dimensional

Given past sales data on zillow.com, predict:

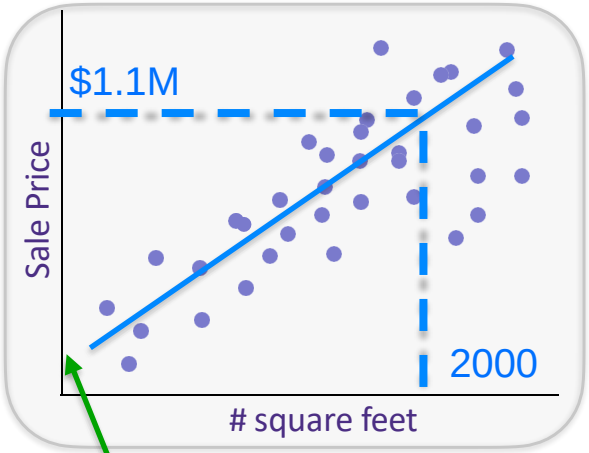
Goal: learn p(y|x)

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

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

now we have label y

previously we just had x



(no intercept for now to make the math easier)

Training Data:

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

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*

Fit a Function to Our Data, 1-d

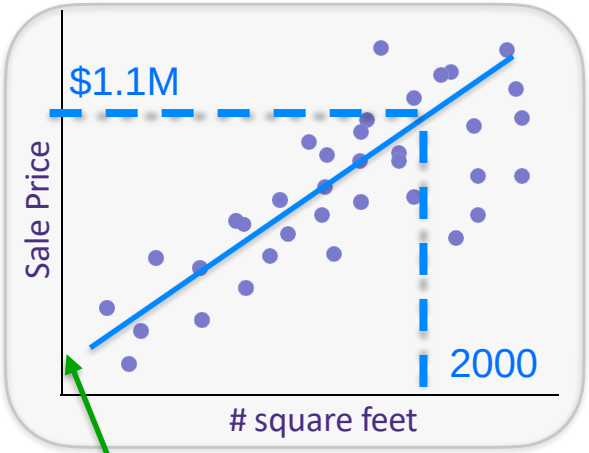
Given past sales data on zillow.com, predict: # Goal: learn p(y|x)

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

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

now we have label y

previously we just had x



(no intercept for now to make the math easier)

Training Data:

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

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?

Fit a Function to Our Data, 1-d

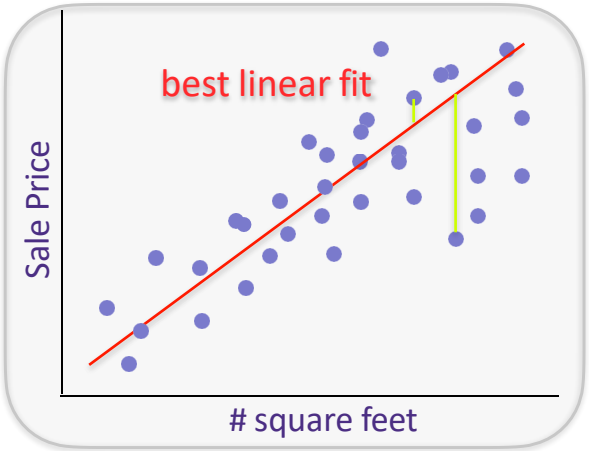
Given past sales data on zillow.com, predict: # Goal: learn p(y|x)

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

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

now we have label y

previously we just had x



Training Data:

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

Hypothesis/Model: linear

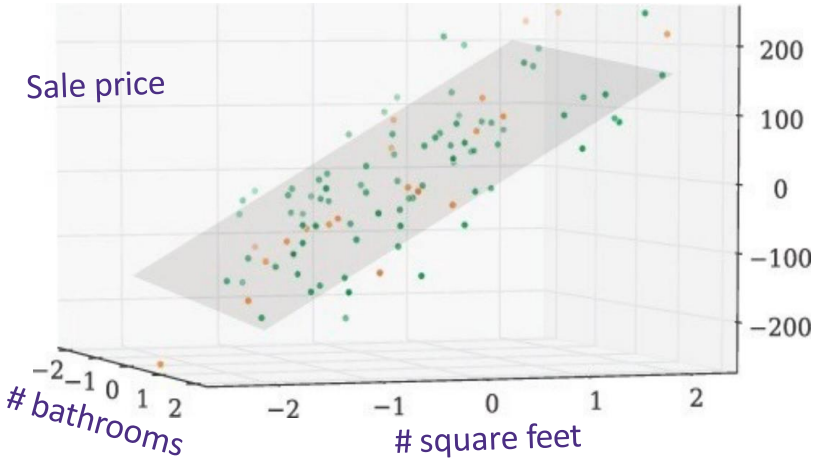
$$y_i = x_i w + \epsilon_i \qquad \epsilon_i \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma^2)$$

The Regression Problem, d-dim

Given past sales data on zillow.com, predict: # Goal: learn p(y|x)

y* = House sale price *from

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



Training Data:

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

Hypothesis/Model: linear

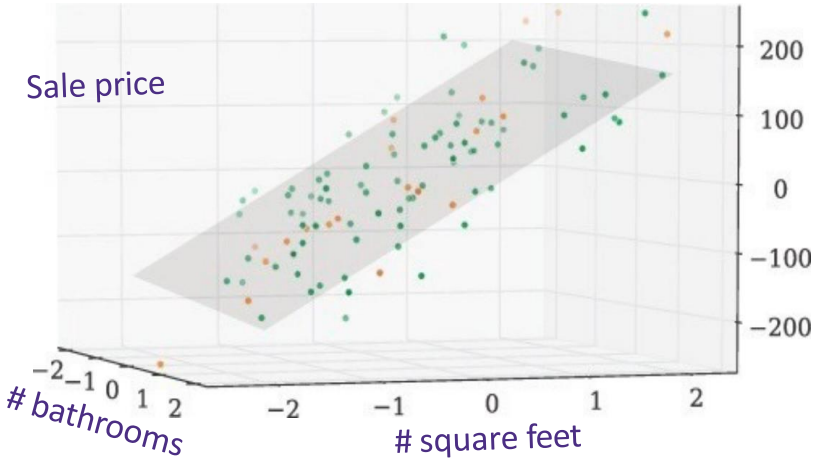
$y_i = x_i w + \epsilon_i$ $\epsilon_i \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma^2)$

The Regression Problem, d-dim

Given past sales data on zillow.com, predict: # Goal: learn p(y|x)

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

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

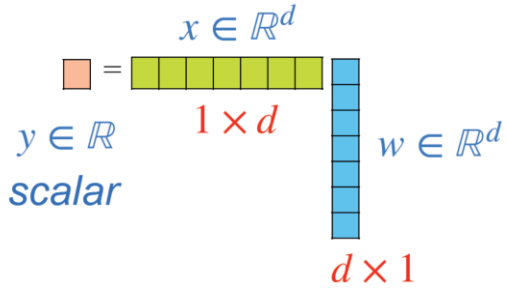


Training Data:

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

Hypothesis/Model: linear

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

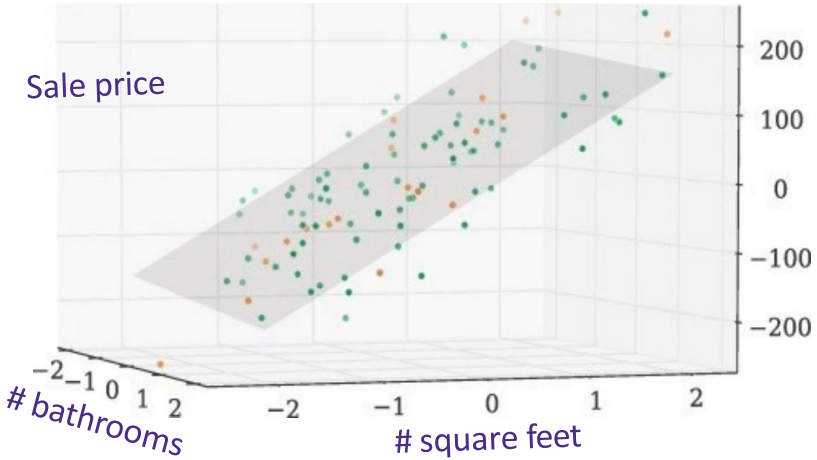


The Regression Problem, d-dim

Given past sales data on zillow.com, predict: # Goal: learn p(y|x)

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

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



Training Data:

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

Hypothesis/Model: linear

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

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

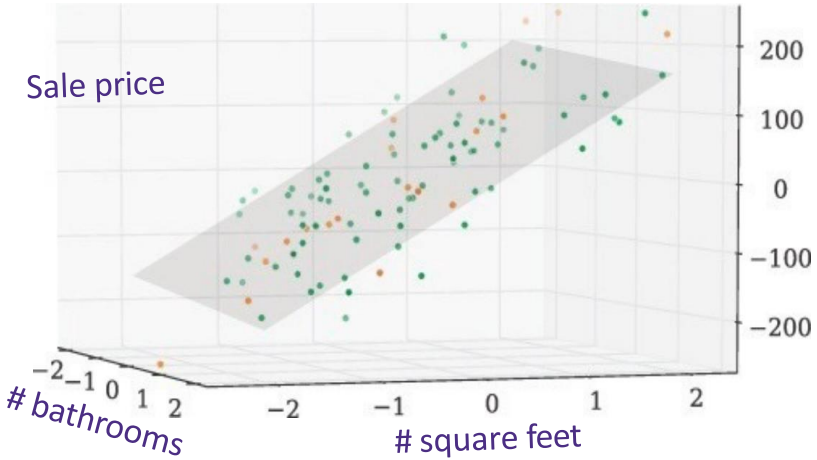
go back to formula sheet, retrieve PDF of Gaussian…

The Regression Problem, d-dim

Given past sales data on zillow.com, predict: # Goal: learn p(y|x)

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

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



Training Data:

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

Hypothesis/Model: linear

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

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

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

Maximizing Log-likelihood

Training Data:

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

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

Likelihood: $$P(\mathcal{D}|w, \sigma) = \prod_{i=1}^{n} p(y_i|x_i, w, \sigma)$$

Maximizing Log-likelihood

Training Data:

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

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

Likelihood:

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

Maximizing Log-likelihood

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

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

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

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

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

Maximizing Log-likelihood

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

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

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

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

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

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

pull out terms that don't depend on *i*

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

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

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

pull out terms that don't depend on *i*

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

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

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!

Maximizing a negative?

Maximizing Log-likelihood

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

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

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

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

Minimizing Mean Squared Error (MSE)

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.

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

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.

Maximizing Log-likelihood

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

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

Maximizing Log-likelihood

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

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

Set gradient=0, solve for w

Maximizing Log-likelihood

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

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

Set gradient=0, solve for w

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

Recall: $w \in \mathbb{R}^d$ $\nabla_w f(x) = \begin{bmatrix} \frac{\partial}{\partial w_0} f(x) \\ \frac{\partial}{\partial w_1} f(x) \\ \vdots \end{bmatrix}$

Gradient wrt *w* is a vector of partial derivatives

Maximizing Log-likelihood

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

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

Set gradient=0, solve for w

Recall: $w \in \mathbb{R}^d$ $\quad \nabla_w f(x) = \begin{bmatrix} \frac{\partial}{\partial w_0} f(x) \\ \frac{\partial}{\partial w_1} f(x) \\ \vdots \end{bmatrix}$

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

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

Maximizing Log-likelihood

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

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

Set gradient=0, solve for w

Recall: $w \in \mathbb{R}^d$ $\quad \nabla_w f(x) = \begin{bmatrix} \frac{\partial}{\partial w_0} f(x) \\ \frac{\partial}{\partial w_1} f(x) \\ \vdots \end{bmatrix}$

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

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

$\nabla_w x^T w = x$

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!

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

Maximizing Log-likelihood

Set gradient=0, solve for w

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

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

$d \times 1$

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

check dimensions!

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

Maximizing Log-likelihood

Set gradient=0, solve for w

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

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

$d \times 1$

check dimensions!

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

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

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

Now for some tricks

$x^T w = a$ # scalar

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

Maximizing Log-likelihood

Set gradient=0, solve for w

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

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

$d \times 1$

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

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

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

Reminder

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

xx^T = outer product dxd

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

How can we isolate w?

Maximizing Log-likelihood

Set gradient=0, solve for w

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

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

$d \times 1$

check dimensions!

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

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

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

Now for some tricks

$x^T w = a$ # scalar

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

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!

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

The equation for fitting our model parameters *w* to our data

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!

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

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

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

Regression Problem in Matrix Notation

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

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

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

Regression Problem in Matrix Notation

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

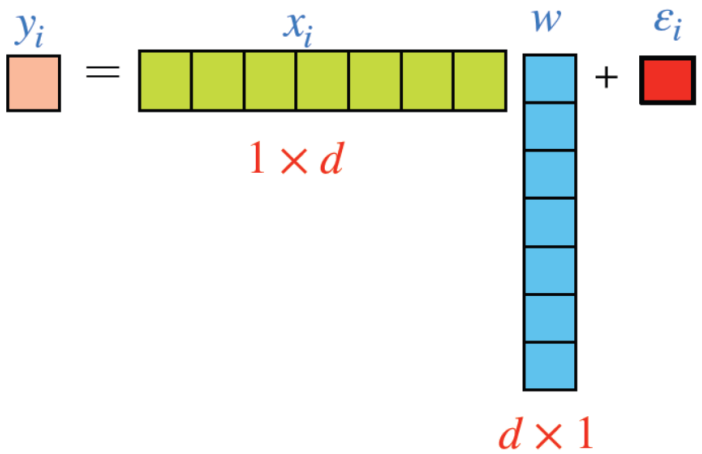
$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \qquad \mathbf{X} = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix} \begin{matrix} d \times 1 \\ d \times 1 \\ \\ d \times 1 \end{matrix}$$

d : # of features

n : # of examples/datapoints

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

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



Regression Problem in Matrix Notation

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

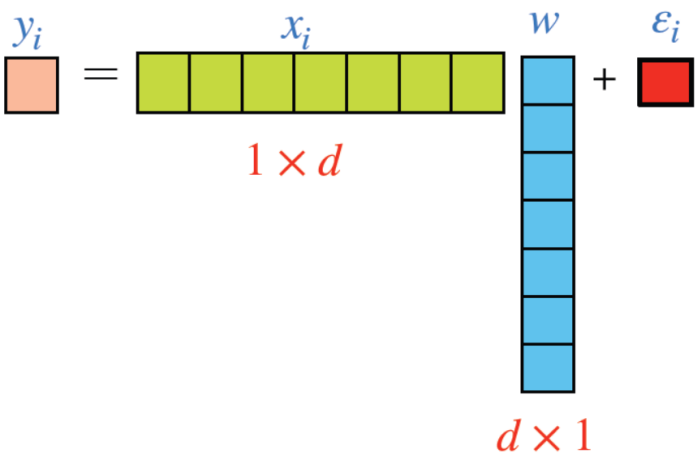
$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \qquad \mathbf{X} = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix} \begin{matrix} d \times 1 \\ d \times 1 \\ \\ d \times 1 \end{matrix}$$

d : # of features

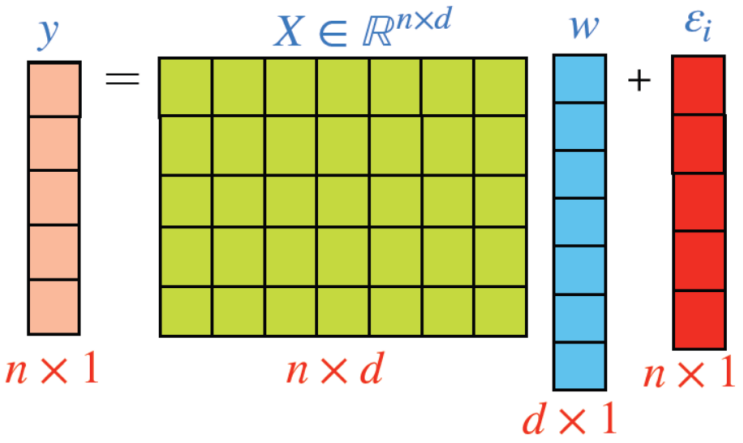
n : # of examples/datapoints

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

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



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



Regression Problem in Matrix Notation

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

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

d : # of features

n : # of examples/datapoints

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

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

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

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

Regression Problem in Matrix Notation

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

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

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

Regression Problem in Matrix Notation

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

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

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

$$= \arg\min_{w} \; y^T y - y^T X w - (Xw)^T y + (Xw)^T (Xw)$$

More identities:

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

Regression Problem in Matrix Notation

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

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

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

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

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

1×1 $\qquad$ 1×1

More identities:

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

Minimizing wrt *w*

Check dimensions

Regression Problem in Matrix Notation

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

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

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

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

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

1×1 $\qquad$ 1×1

More identities:

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

Minimizing wrt *w*

Check dimensions

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

Regression Problem in Matrix Notation

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

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

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

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

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

1×1 $\quad$ 1×1

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

More identities:

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

Minimizing wrt *w*

Check dimensions

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

Regression Problem in Matrix Notation

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

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

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*

Check dimensions

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

Pretty simple!

So now what?

Regression Problem in Matrix Notation

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

$$= \arg\min_{w} (\mathbf{y} - \mathbf{X}w)^T (\mathbf{y} - \mathbf{X}w) \quad = \arg\min_{w} -2y^T Xw + w^T X^T Xw$$

Take derivative, set to 0!

Regression Problem in Matrix Notation

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

$$= \arg\min_{w} (\mathbf{y} - \mathbf{X}w)^T(\mathbf{y} - \mathbf{X}w) = \arg\min_{w} -2y^T Xw + w^T X^T Xw$$

$$\nabla_w[-2y^T Xw + w^T X^T Xw] = 0$$

Take derivative, set to 0!

Regression Problem in Matrix Notation

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

$$= \arg\min_w (\mathbf{y} - \mathbf{X}w)^T(\mathbf{y} - \mathbf{X}w) = \arg\min_w -2y^TXw + w^TX^TXw$$

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

Take derivative, set to 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 $\nabla_w w^T Aw = 2Aw$

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

Regression Problem in Matrix Notation

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

$$= \arg\min_w (\mathbf{y} - \mathbf{X}w)^T(\mathbf{y} - \mathbf{X}w) = \arg\min_w -2y^TXw + w^TX^TXw$$

$$\nabla_w[-2y^TXw + w^TX^TXw] = 0$$

(B) (C)

$$= -2X^Ty + 2X^TXw = 0$$

Take derivative, set to 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 $\nabla_w w^T Aw = 2Aw$

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

Regression Problem in Matrix Notation

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

$$= \arg\min_w (\mathbf{y} - \mathbf{X}w)^T(\mathbf{y} - \mathbf{X}w) = \arg\min_w -2y^TXw + w^TX^TXw$$

$$\nabla_w[-2y^TXw + w^TX^TXw] = 0$$

(B) (C)

Take derivative, set to 0!

$$= -2X^Ty + 2X^TXw = 0$$

$$X^TXw = X^Ty$$

Useful matrix gradient identities

(A) $\nabla_w x^Tw = x$

(B) $\nabla_w x^TAw = A^Tx$

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

Quadratic form

If A is symmetric ($A = A^T$),

Then $\nabla_w w^TAw = 2Aw$

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

Regression Problem in Matrix Notation

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

$$= \arg\min_w (\mathbf{y} - \mathbf{X}w)^T(\mathbf{y} - \mathbf{X}w) = \arg\min_w -2y^TXw + w^TX^TXw$$

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

Take derivative, set to 0!

$$= -\cancel{2}X^Ty + \cancel{2}X^TXw = 0$$

$$X^TXw = X^Ty$$

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

Useful matrix gradient identities

(A) $\nabla_w x^Tw = x$

(B) $\nabla_w x^TAw = A^Tx$

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

Quadratic form

If A is symmetric ($A = A^T$),

Then $\nabla_w w^TAw = 2Aw$

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

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

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

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

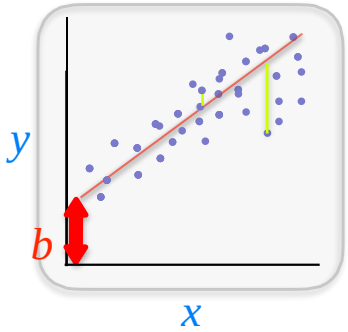
$$\begin{aligned} \widehat{w}_{LS} &= \arg\min_{w} ||\mathbf{y} - \mathbf{X}w||_2^2 \\ &= \arg\min_{w} (\mathbf{y} - \mathbf{X}w)^T (\mathbf{y} - \mathbf{X}w) \end{aligned}$$

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

"Ordinary Least Squares"

Regression Problem in Matrix Notation

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



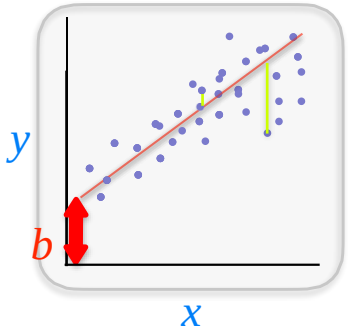
What about an offset?

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

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

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…

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

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…

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

Computes the exact solution to a system of linear equations in the matrix form

$$\begin{bmatrix} w \\ b \end{bmatrix}$$

= np.linalg.solve() # There's a better way…

Dealing with an Offset

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

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

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

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

Dealing with an Offset

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

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

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

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

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

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?

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

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?

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

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, calculate 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?

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, calculate 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

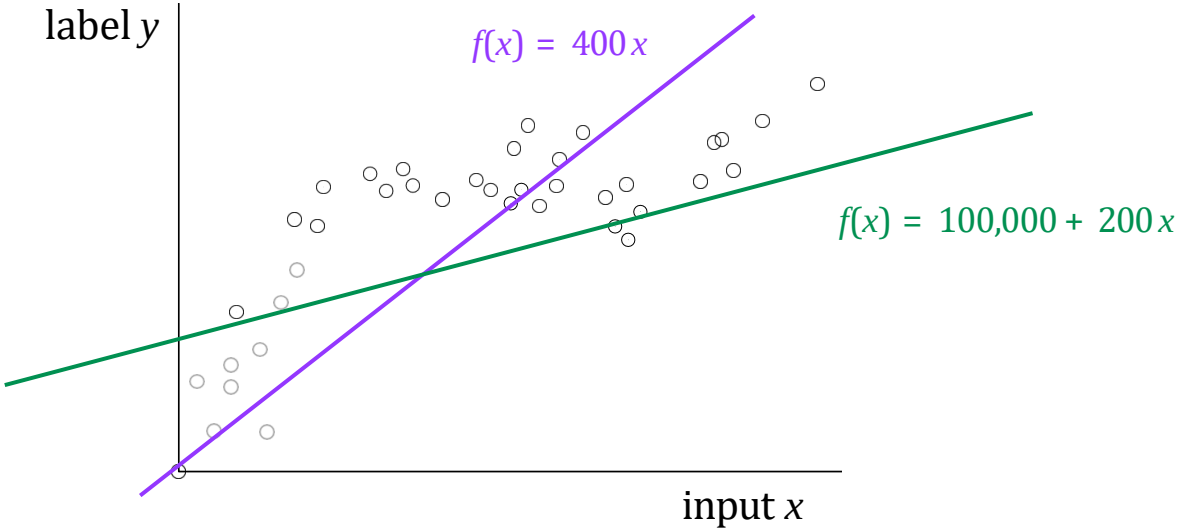
Process

- Decide on a **model** for the likelihood function $f(x; \mathrm{w})$

- 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

Recap: Linear Regression



In general high-dimensions, we fit a linear model with intercept

$$y_i \simeq w^T x_i + b\,, \quad \text{or equivalently} \quad y_i = w^T x_i + b + \epsilon_i$$

with model parameters $(w \in \mathbb{R}^d, b \in \mathbb{R})$ that minimizes ℓ_2-loss

$$\mathscr{L}(w,b) = \sum_{i=1}^{n} \underbrace{(y_i - (w^T x_i + b))^2}_{\text{error } \epsilon_i}$$

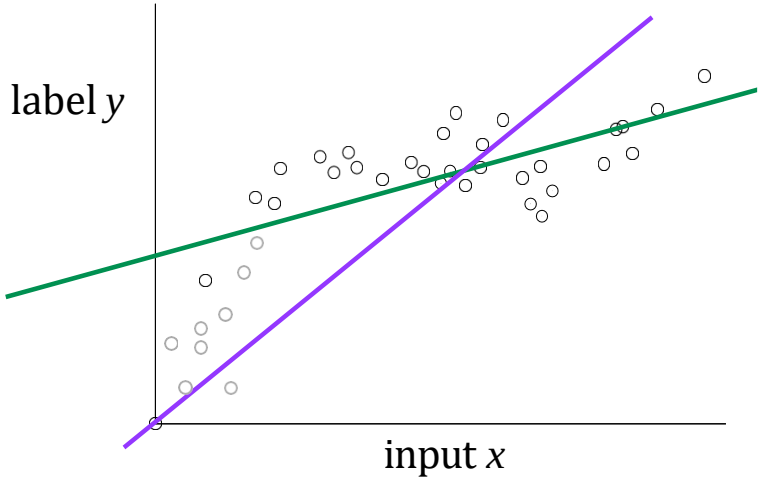
MSE

Regression in 1-dimension

Data: $\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$, $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$

Linear model with parameter (b, w_1)**:**

- $\hat{y}_i = b + w_1 x_i$



Quadratic Regression in 1-dimension

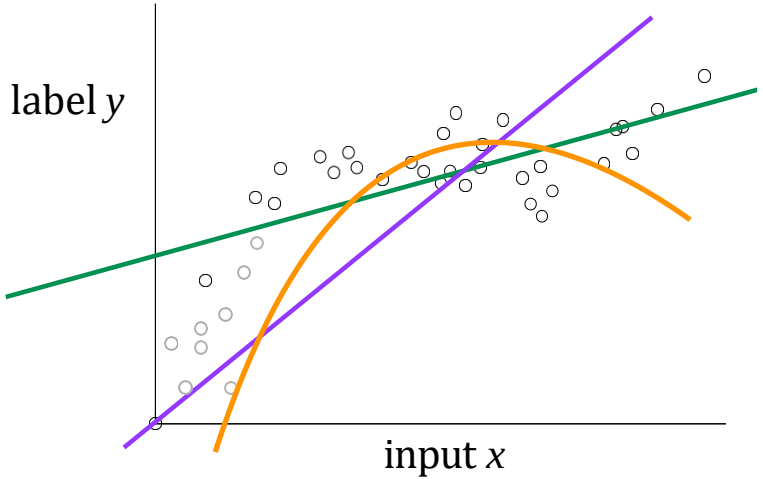
Data: $\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$, $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$

Linear model with parameter (b, w_1)**:**

- $\hat{y}_i = b + w_1 x_i$

Quadratic model with parameter $(b, w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix})$**:**

- $\hat{y}_i = b + w_1 x_i + w_2 x_i^2$



$$h(x_i) = \begin{bmatrix} 1 \\ x_i \\ x_i^2 \end{bmatrix} \quad \hat{y} = h(x_i)^T \begin{bmatrix} b \\ w_1 \\ w_2 \\ \vdots \end{bmatrix}$$

Still linear regression, but on quadratic features

Linear layer fit to non-linear features

Polynomial Regression in 1-dimension

Data: $\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$, $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$

Linear model with parameter (b, w_1):

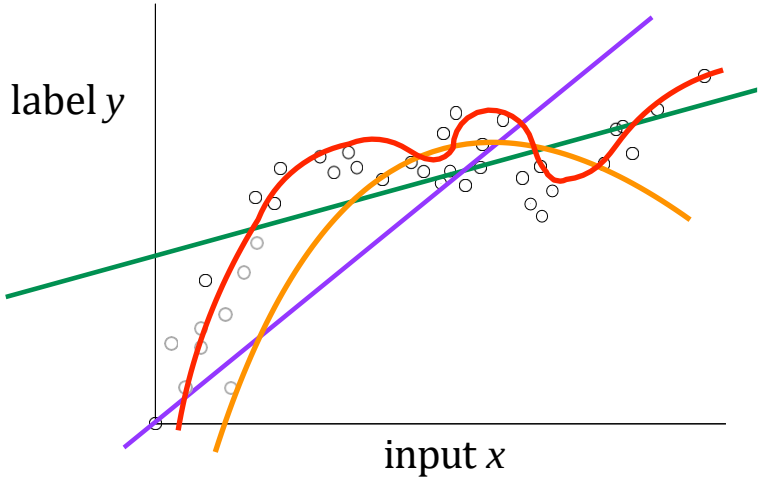
- $\hat{y}_i = b + w_1 x_i$

Quadratic model with parameter $(b, w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix})$:

- $\hat{y}_i = b + w_1 x_i + w_2 x_i^2$

Degree-p polynomial model with *p* parameters

- $\hat{y}_i = b + w_1 x_i + w_2 x_i^2 + \ldots + w_p x_i^p$



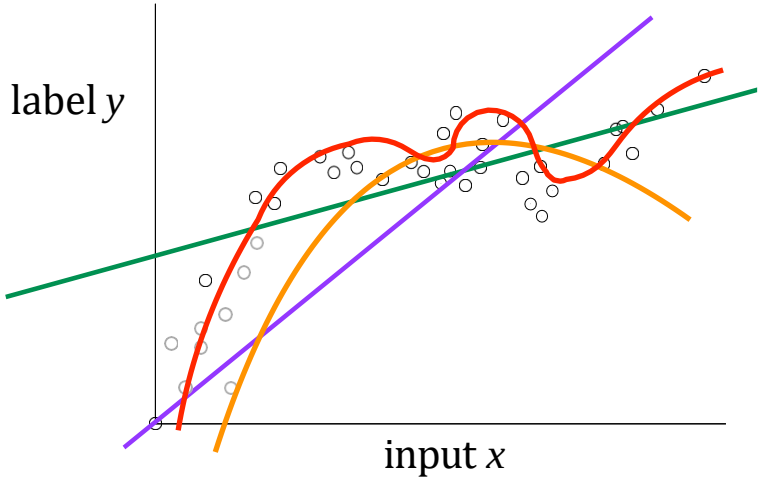
$$h(x_i) = \begin{bmatrix} 1 \\ x_i \\ x_i^2 \end{bmatrix} \quad \hat{y} = h(x_i)^T \begin{bmatrix} b \\ w_1 \\ w_2 \\ \vdots \end{bmatrix}$$

Still linear regression, but on quadratic features

Linear layer fit to non-linear features

Polynomial Regression in 1-dimension

Data: $\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$, $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$



Linear model with parameter (b, w_1):

- $\hat{y}_i = b + w_1 x_i$

Quadratic model with parameter $(b, w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix})$:

- $\hat{y}_i = b + w_1 x_i + w_2 x_i^2$

Degree-p polynomial model with *p* parameters

- $\hat{y}_i = b + w_1 x_i + w_2 x_i^2 + \ldots + w_p x_i^p$

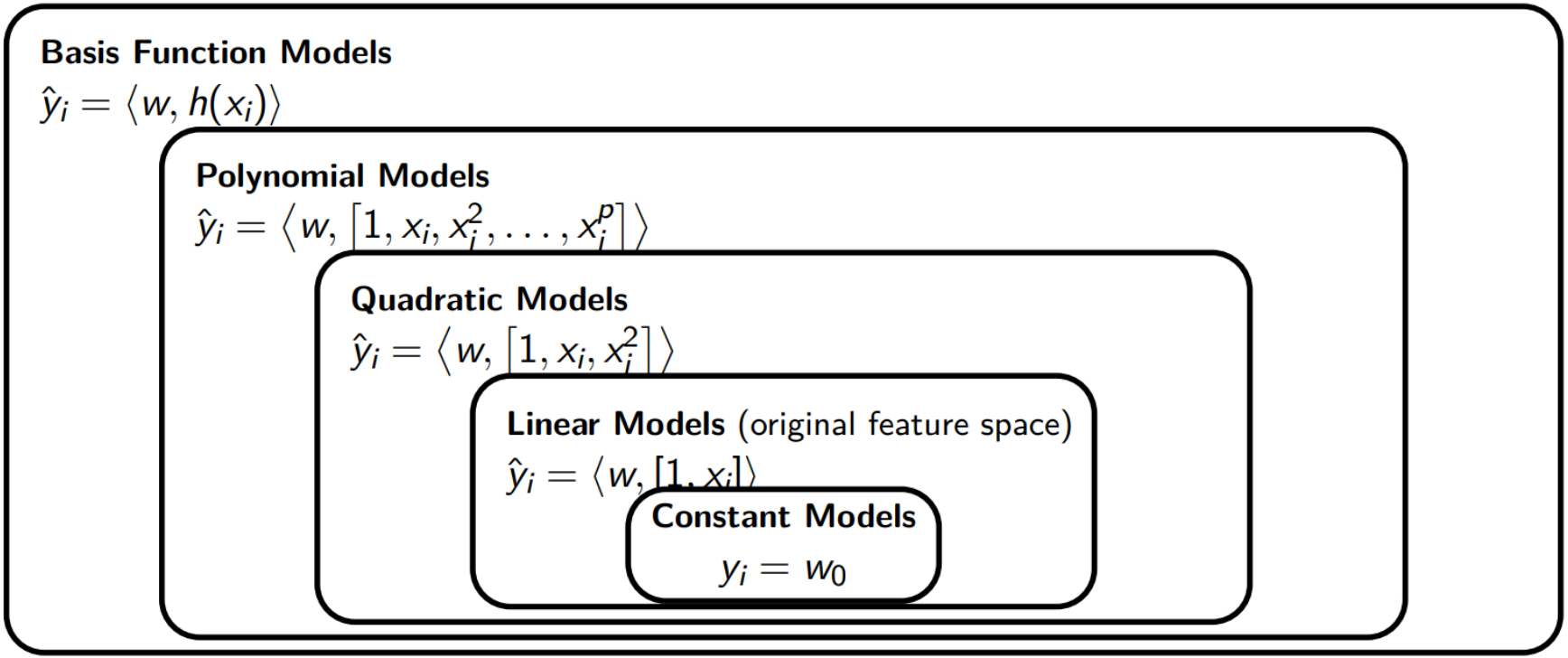
$$h(x_i) = \begin{bmatrix} 1 \\ x_i \\ x_i^2 \end{bmatrix} \quad \hat{y} = h(x_i)^T \begin{bmatrix} b \\ w_1 \\ w_2 \\ \vdots \end{bmatrix}$$

Inner Product

General p-features with parameter $w = \begin{bmatrix} w_1 \\ \vdots \\ w_p \end{bmatrix}$: $\hat{y}_i = \langle w, h(x_i) \rangle$ where $h : \mathbb{R} \rightarrow \mathbb{R}^p$

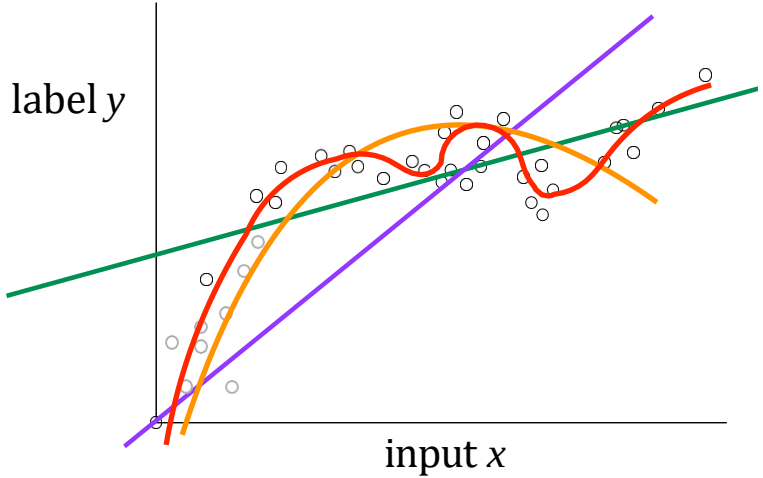
Basis Function

Regression in 1-dimension: Landscape of Models



Basis Functions $\supset$ Polynomials $\supset$ Quadratics $\supset$ Linear $\supset$ Constants

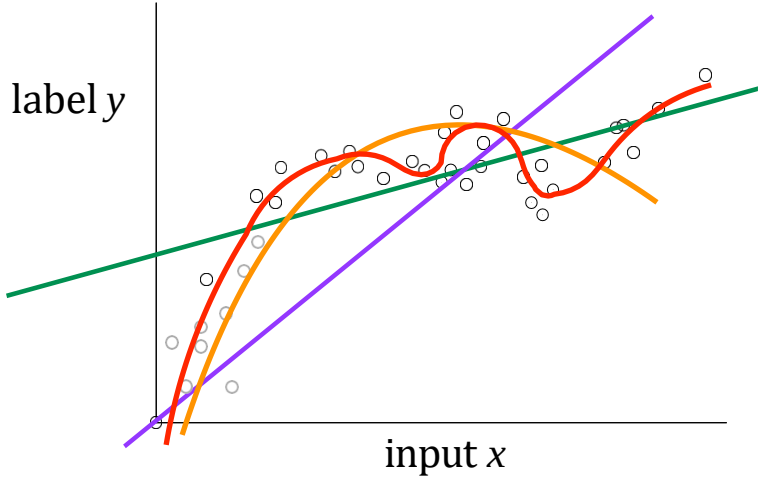
Regression in 1-dimension



Which line gives us the best fit?

- **Green** & **purple**?
- **Orange**?
- **Red**?

Regression in 1-dimension

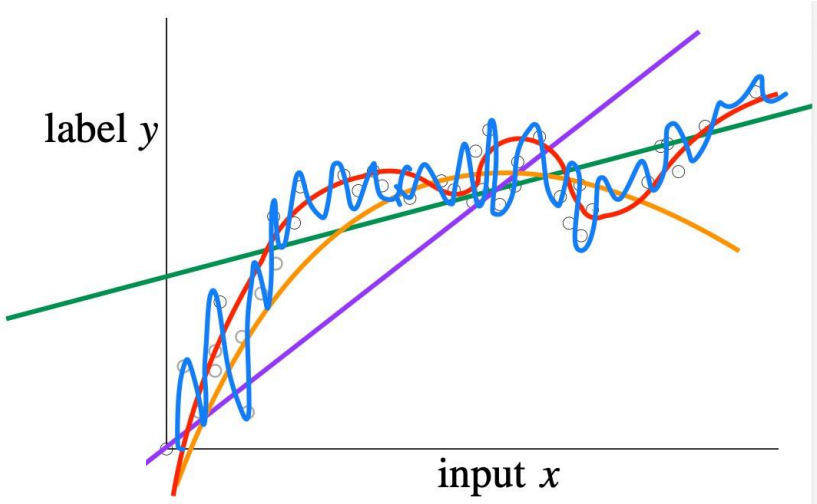


Which line gives us the best fit?

- **Green** & **purple**? Can't capture variance. **Underfitting**
- **Orange**? Can't capture variance. **Underfitting**
- **Red**? Just right?

  Could we take it too far?

Regression in 1-dimension

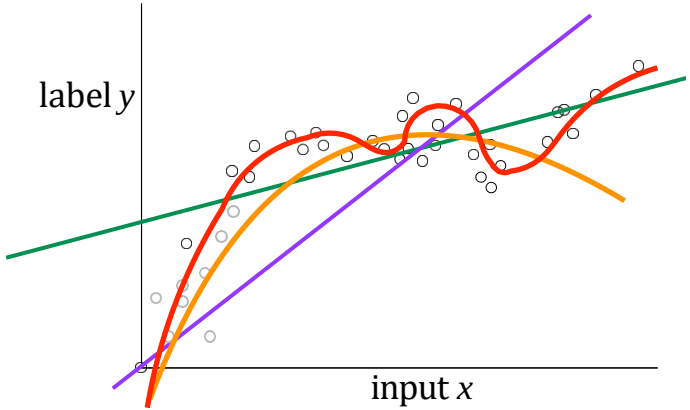


Which line gives us the best fit?

- **Green** & **purple**? Can't capture variance. **Underfitting**
- **Orange**? Can't capture variance. **Underfitting**
- **Red**? Just right?
- **Blue**? Fits random fluctuations / measurement errors in training dataset. **Overfitting.**

Regression in 1-dimension

Data: $\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$, $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$



General p-features with parameter $w = \begin{bmatrix} w_1 \\ \vdots \\ w_p \end{bmatrix}$:

- $\hat{y}_i = \langle w, h(x_i) \rangle$ where $h : \mathbb{R} \rightarrow \mathbb{R}^p$

Note: *h* can be arbitrary non-linear functions!

$$h(x) = \left[\log(x), x^2, \sin(x), \sqrt{x}\right]^\top$$

The model is linear in the new feature space,
but it's nonlinear in the original feature space.

Regression in 1-dimension

Data: $\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$, $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$

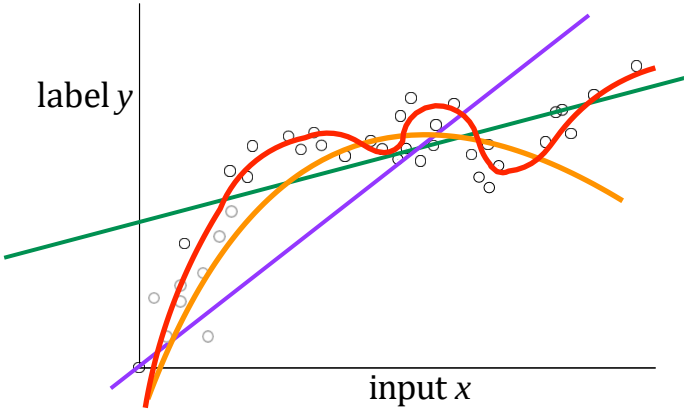
General p-features with parameter $w = \begin{bmatrix} w_1 \\ \vdots \\ w_p \end{bmatrix}$:

- $\hat{y}_i = \langle w, h(x_i) \rangle$ where $h : \mathbb{R} \rightarrow \mathbb{R}^p$

Note: *h* can be arbitrary non-linear functions!

$$h(x) = \left[\log(x), x^2, \sin(x), \sqrt{x}\right]^\top$$

The model is linear in the new feature space, but it's nonlinear in the original feature space.



Discussion:

- When would you use sin(x)?
- How should I transform the house feature: zip code?

Regression in 1-dimension

Data: $\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$, $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$

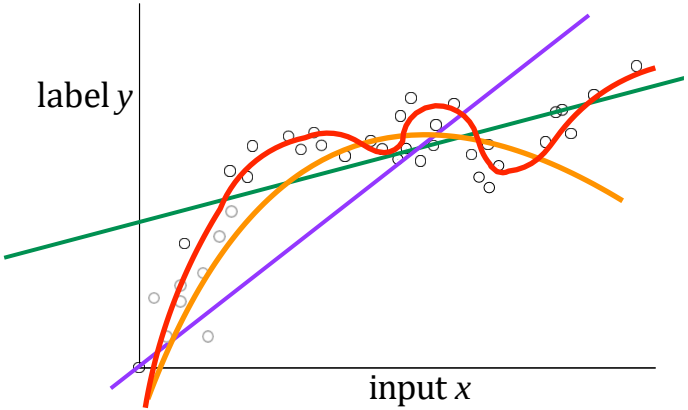
General p-features with parameter $w = \begin{bmatrix} w_1 \\ \vdots \\ w_p \end{bmatrix}$:

- $\hat{y}_i = \langle w, h(x_i) \rangle$ where $h : \mathbb{R} \rightarrow \mathbb{R}^p$

Note: *h* can be arbitrary non-linear functions!

$$h(x) = \left[\log(x), x^2, \sin(x), \sqrt{x}\right]^\top$$

The model is linear in the new feature space, but it's nonlinear in the original feature space.



Discussion:

- When would you use sin(x)?

  # Cyclic data e.g. weather

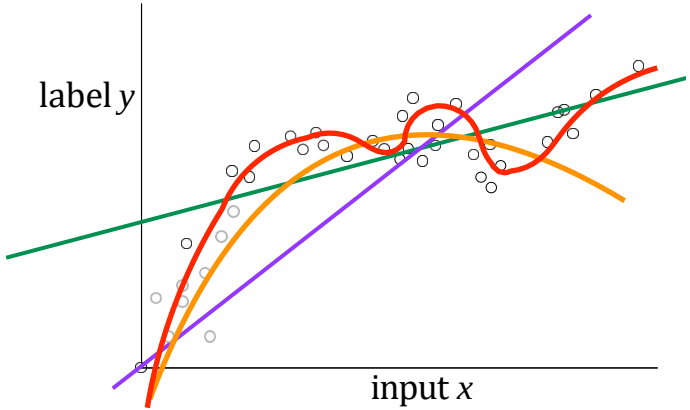
- How should I transform the house feature: zip code?

  # Lat / long

This is the art of feature engineering

Regression in 1-dimension

Data: $\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$, $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$



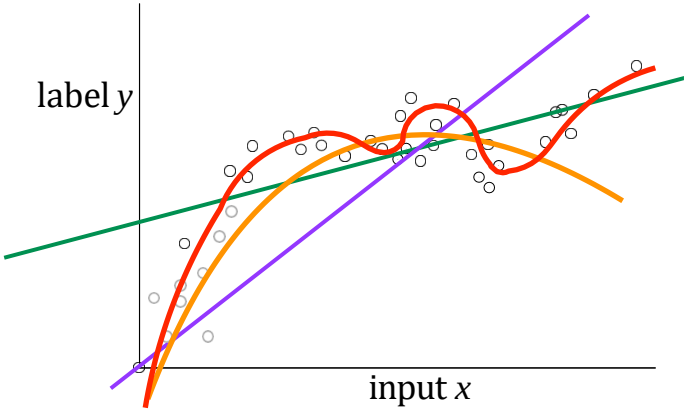
General p-features with parameter $w = \begin{bmatrix} w_1 \\ \vdots \\ w_p \end{bmatrix}$:

- $\hat{y}_i = \langle w, h(x_i) \rangle$ where $h : \mathbb{R} \rightarrow \mathbb{R}^p$

How do we learn w?

Regression in 1-dimension

Data: $\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$, $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$



General p-features with parameter $w = \begin{bmatrix} w_1 \\ \vdots \\ w_p \end{bmatrix}$:

- $\hat{y}_i = \langle w, h(x_i) \rangle$ where $h : \mathbb{R} \to \mathbb{R}^p$

How do we learn w?

$$\mathbf{H} = \begin{bmatrix} - - h(x_1)^\top - - \\ \vdots \\ - - h(x_n)^\top - - \end{bmatrix} \in \mathbb{R}^{n \times p}$$

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

For a new test point x, predict

$\hat{y} = \langle \widehat{w}, h(x) \rangle$

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

Acknowledgement

Most of the slides are based on the ML course of:

- Natasha Jaques, University of Washington

THANK YOU