

446 Section L-2 Norm

TA: Varun Ananth

Plans for today!

1. This
2. Reminders
3. Problems
 - a. P1 a, b
 - b. Review Linear Regression
 - c. P2 (writing)

Reminders

- **HW0** due **Oct 8** (next Wednesday)

Some tips:

- Use office hours to your advantage
 - Student TA OH for homework questions
 - Professor OH for conceptual questions
- Skim the homeworks the day they are assigned and try one problem
 - Motivates you to get things done on time, starting an untouched assignment can be daunting
- Keep a tab open with the lecture slides while you do the homework for reference

Problem 1

Context

(1a)

We consider the linear measurement model (parameterized by w), $y_i = x_i^\top w + v_i$ for $i = 1, 2, \dots, m$. The noise v_i for different measurements (x_i, y_i) are all independent and identically distributed. Under our assumption of a linear model, $v_i = y_i - x_i^\top w$. Note Per the principle of maximum likelihood estimation, we seek to maximize

$$\log p_w((x_1, y_1), \dots, (x_m, y_m)) = \log \prod_{i=1}^m p(y_i - x_i^\top w).$$

- (a) Show that when the noise measurements follow a Gaussian distribution ($v_i \sim \mathcal{N}(0, \sigma^2)$), the maximum likelihood estimate of w is the solution to $\min_w \|Xw - Y\|_2^2$. Here each row in X corresponds to a x_i , and each row in Y to y_i .

$$\underline{y_i = x_i^T w + v_i} \quad \text{Linear model}$$

y_i ; Prediction for datapoint i

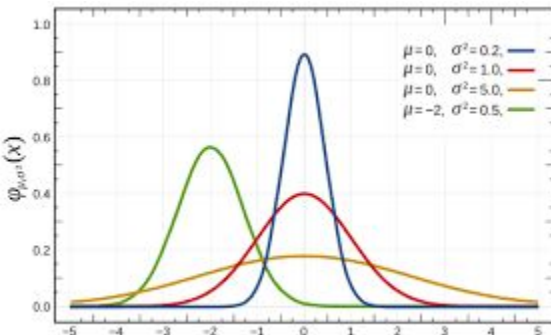
x_i ; datapoint i

w ; model weights

v_i ; Noise term

Noise (v_i) is i.i.d from the
Gaussian Distribution

$$p(v) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-v^2/2\sigma^2}$$



We want to show: MLE of the weights for linear model with a Gaussian noise term reduces to minimizing sum of squared errors

$$\log p_w((x_1, y_1), \dots, (x_m, y_m)) = \log \prod_{i=1}^m p(y_i - x_i^T w).$$

$$\begin{aligned} \hat{w}_{MLE} &= \underset{w}{\operatorname{argmax}} \log \left[\prod_{i=1}^m p(y_i - x_i^T w) \right] \\ &= \underset{w}{\operatorname{argmax}} \sum_{i=1}^m \log [p(y_i - x_i^T w)] \end{aligned}$$

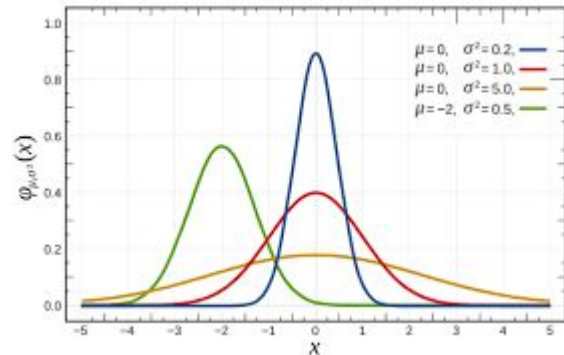
Linear model: $y_i = x_i^T w + v_i$

$$= \underset{w}{\operatorname{argmax}} \sum_{i=1}^m \log [P(y_i - x_i^T w)]$$

Important:

$$\left. \begin{aligned} v_i &\stackrel{\text{iid}}{\sim} N(0, \sigma^2) \\ \therefore P(v_i) &= \frac{1}{\sqrt{2\pi}\sigma^2} e^{\frac{-v^2}{2\sigma^2}} \end{aligned} \right\} \text{Definition of } N(0, \sigma^2)$$

$$\begin{aligned} v_i &= y_i - x_i^T w \\ \therefore P(y_i - x_i^T w) &= \frac{1}{\sqrt{2\pi}\sigma^2} e^{\frac{-v^2}{2\sigma^2}} \end{aligned}$$




Here we use the fact that v_i is i.i.d from a 0-mean Gaussian to introduce the Gaussian distribution equation into our proof

$$= \underset{w}{\operatorname{argmax}} \sum_{i=1}^m \log \left[\frac{1}{\sqrt{2\pi}\sigma^2} e^{\frac{-(y_i - x_i^T w)^2}{2\sigma^2}} \right]$$

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

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

Important: 

Constant offset does not affect optimization

$$\text{Ex. } \operatorname{argmax}_x [-x^2] = \operatorname{argmax}_x [-x^2 + 10] = 0$$

We are looking for the argument that maximizes the function, not the maximum

$$= \underset{w}{\operatorname{argmax}} \sum_{i=1}^m \log \left[\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_i - x_i^T w)^2}{2\sigma^2}} \right]$$

Apply logarithm rules

$$= \underset{w}{\operatorname{argmax}} \sum_{i=1}^m \left[\log \left[\frac{1}{\sqrt{2\pi\sigma^2}} \right] + \log \left[e^{-\frac{(y_i - x_i^T w)^2}{2\sigma^2}} \right] \right]$$

Constant offset does not influence answer

$$= \underset{w}{\operatorname{argmax}} \sum_{i=1}^m \log \left[e^{-\frac{(y_i - x_i^T w)^2}{2\sigma^2}} \right]$$

Apply logarithm rules

$$= \underset{w}{\operatorname{argmax}} \sum_{i=1}^m \left[\frac{1}{2\sigma^2} - (y_i - x_i^T w)^2 \right]$$

Constant offset does not influence answer

$$= \underset{w}{\operatorname{argmax}} \sum_{i=1}^m - (y_i - x_i^T w)^2$$

$$= \underset{w}{\operatorname{argmin}} \sum_{i=1}^m (y_i - x_i^T w)^2 \quad \left[\begin{array}{l} \text{minimize} \\ \text{sum of squared errors (SSE)} \end{array} \right]$$

$$= \underset{w}{\operatorname{argmin}} \sum_{i=1}^m (y_i - x_i^T w)^2 \quad \left. \vphantom{\sum_{i=1}^m} \right] \begin{array}{l} \text{minimize} \\ \text{sum of squared errors (SSE)} \end{array}$$

$$\hat{w}_{MLE} = \underset{w}{\operatorname{argmin}} \|Xw - Y\|_2^2$$

Begin closed form solution proof...

We have shown that using MLE on a linear model with a Gaussian noise term results in the same optimization objective as minimizing the sum of squared errors between the ground truth and the prediction!

MLE \rightarrow Linear Regression

$$\hat{\mathbf{w}} = \underset{w}{\operatorname{argmin}} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

Find \mathbf{w} that minimizes the SSE. This is $\hat{\mathbf{w}}$.

$$= \underset{w}{\operatorname{argmin}} \sum_{i=1}^n (y_i - (\mathbf{x}_i^\top \mathbf{w} + b))^2$$

Plug in $\hat{y}_i = \mathbf{x}_i^\top \mathbf{w} + b$.

$$= \underset{w}{\operatorname{argmin}} \sum_{i=1}^n (y_i - (\mathbf{x}_i^\top \mathbf{w}))^2$$

Disregard b , since an intercept can be represented by appending a 1 to \mathbf{x} .

We got here
from MLE!

$$0 = \frac{\partial}{\partial w} \sum_{i=1}^n (y_i - (\mathbf{x}_i^\top \hat{\mathbf{w}}))^2$$

Find $\hat{\mathbf{w}}$ by taking the partial derivative of the argmin term and setting it equal to 0.

$$= \sum_{i=1}^n \frac{\partial}{\partial w} (y_i - (\mathbf{x}_i^\top \hat{\mathbf{w}}))^2$$

Property of derivatives.

$$= \sum_{i=1}^n 2(y_i - \mathbf{x}_i^\top \hat{\mathbf{w}})(-\mathbf{x}_i)$$

Derive.

Now let's
complete the
proof

MLE \rightarrow Linear Regression

$$= \sum_{i=1}^n (y_i - \mathbf{x}_i^\top \hat{\mathbf{w}})(\mathbf{x}_i)$$

Divide by -2

$$= \sum_{i=1}^n (\mathbf{x}_i)(y_i - \mathbf{x}_i^\top \hat{\mathbf{w}})$$

The term $y_i - \mathbf{x}_i^\top \hat{\mathbf{w}}$ is a scalar. $c\mathbf{x} = \mathbf{x}c$

$$= \sum_{i=1}^n (\mathbf{x}_i y_i - \mathbf{x}_i \mathbf{x}_i^\top \hat{\mathbf{w}})$$

Distribute

$$= \sum_{i=1}^n \mathbf{x}_i y_i - \sum_{i=1}^n \mathbf{x}_i \mathbf{x}_i^\top \hat{\mathbf{w}}$$

Distribute

$$\left(\sum_{i=1}^n \mathbf{x}_i \mathbf{x}_i^\top \right) \hat{\mathbf{w}} = \sum_{i=1}^n \mathbf{x}_i y_i$$

Move second term to LHS.

MLE \rightarrow Linear Regression

$$(X^T X) \hat{\mathbf{w}} = X^T \mathbf{y}$$

Convert from vector summation form to matrix form

$$\begin{aligned}(X^T X)^{-1} (X^T X) \hat{\mathbf{w}} &= (X^T X)^{-1} X^T \mathbf{y} \\ \hat{\mathbf{w}} &= (X^T X)^{-1} X^T \mathbf{y}\end{aligned}$$

Left multiply by $(X^T X)^{-1}$

Cancel

I would try and understand this well! It's really cool.

This is mostly in summation form until the very end. The lecture notes (should) have it in matrix form until the very end. Look and study the one that makes most sense to you.

Any questions?

Context (1b)

Linear model: $y_i = x_i^\top w + v_i$

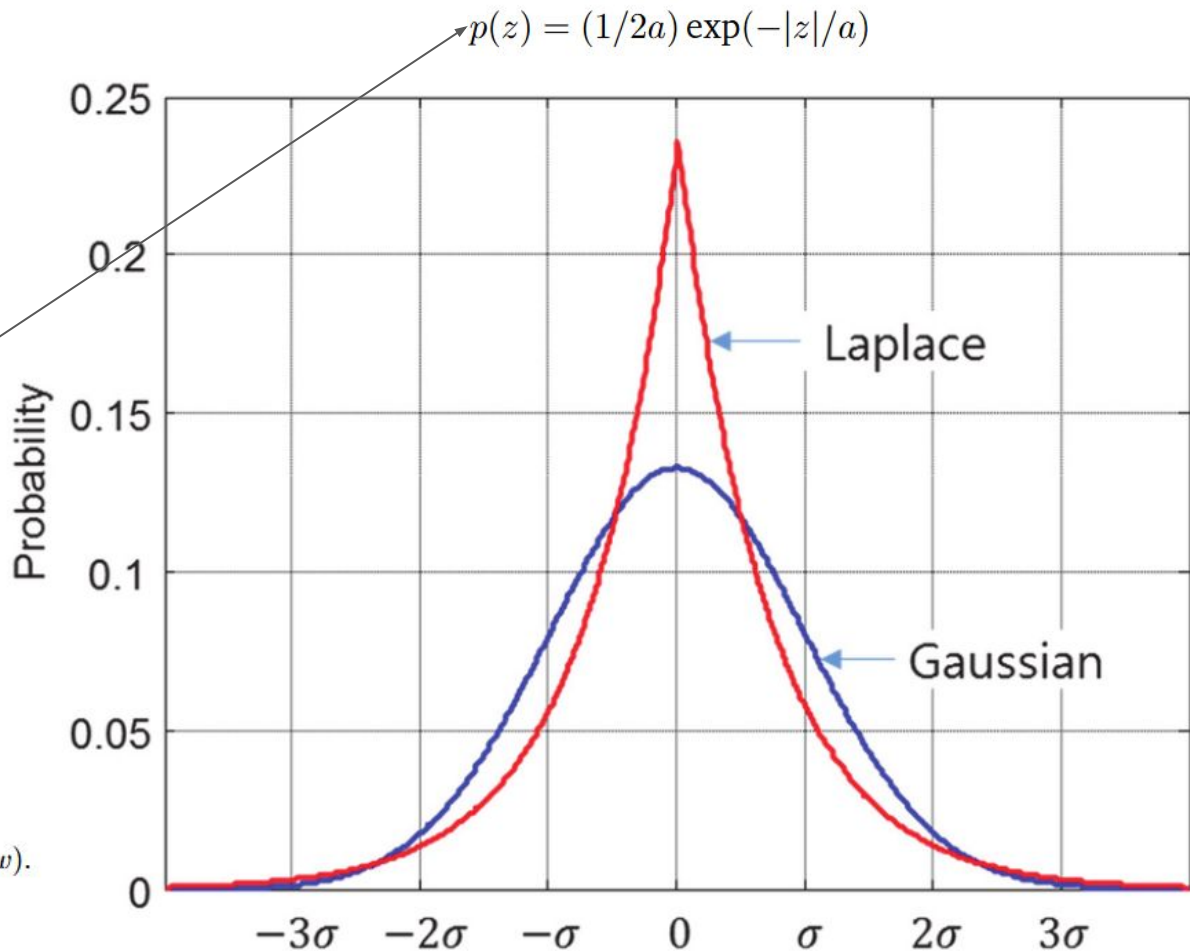
Noise (v_i) is i.i.d from the *Laplacian Distribution*

So: $v_i = y_i - x_i^\top w$

This RHS comes from the Laplacian Distribution

We wish to maximize:

$$\log p_w((x_1, y_1), \dots, (x_m, y_m)) = \log \prod_{i=1}^m p(y_i - x_i^\top w).$$



$$\hat{w}_{MLE} = \arg \max_w \log p_w((x_1, y_1), \dots, (x_m, y_m))$$

$$= \arg \max_w \log \prod_{i=1}^m p(y_i - x_i^\top w)$$

$$= \arg \max_w \sum_{i=1}^m \log \underbrace{p(y_i - x_i^\top w)}_{\text{What do we plug in here?}} \quad [\log(ab) = \log a + \log b]$$

Let's look at the steps *before* considering the distribution:

$$= \arg \max_w \sum_{i=1}^m \log \left(\frac{1}{2a} \cdot e^{\frac{-|y_i - x_i^T w|}{a}} \right)$$

$$= \arg \max_w \sum_{i=1}^m \log \left(\frac{1}{2a} \right) - \frac{|y_i - x_i^T w|}{a}$$

$$= \arg \max_w \sum_{i=1}^m -\frac{|y_i - x_i^T w|}{a}$$

constant offset doesn't affect optimization

$$= \frac{1}{a} \cdot \arg \max_w \sum_{i=1}^m -|y_i - x_i^T w|$$

$$= \arg \max_w \sum_{i=1}^m -|y_i - x_i^T w|$$

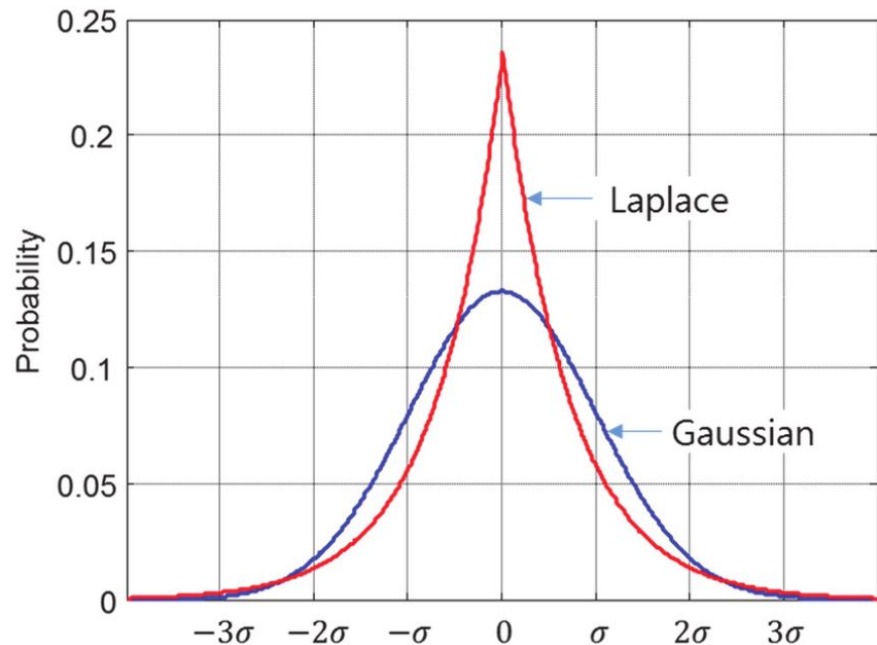
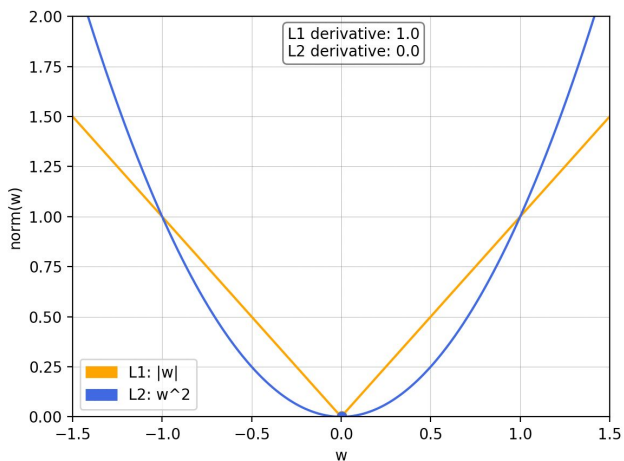
constant offset doesn't affect optimization

$$= \arg \min_w \sum_{i=1}^m |y_i - x_i^T w| = \arg \min_w \|Xw - Y\|_1$$

Therefore the maximum likelihood estimate of w is $\hat{w} = \arg \min_w \|Xw - Y\|_1$.

Cool Connection

Visual similarities exist between the L1 norm vs. the Laplacian, and the L2 norm vs. the Gaussian



Using Laplacian (sharp) as noise results in the L1 norm (sharp) in the optimization equation

Using Gaussian (smooth) as noise results in the L2 norm (smooth) in the optimization equation

Problem 2

Context (2)

Standardization

$$\tilde{x}_i^{(j)} = \frac{x_i^{(j)} - \mu_i}{\sigma_i}$$

Normalization

$$\tilde{x}_i^{(j)} = \frac{x_i^{(j)} - x_i^{min}}{x_i^{max} - x_i^{min}}$$

Not the same!

2.1. Data Standardization

Data standardization is the task of transforming each feature in our dataset to have mean 0 and variance 1. The typical way to do this is using the Z-Score, which is defined as below:

$$\tilde{x}_i^{(j)} = \frac{x_i^{(j)} - \mu_i}{\sigma_i}$$

Where μ_i is the mean of each feature and σ_i is the standard deviation of each feature, which are empirically calculated from the data.

Question: what should you do when $\sigma_i = 0$ for some i ?

Solution:

Having $\sigma_i = 0$ for some feature means that the value for that feature is constant in our dataset. If we leave it as 0, we will encounter a divide by 0 error. Since the feature is constant, once we subtract the mean, the new value for the feature will be 0, so we can divide by anything except 0 to avoid this error.

Having $\sigma_i = 0$ is rare, and may be a sign something is wrong with your data or code. One specific case to watch out for is appending your bias column of ones before standardizing.

2.2. Data Normalization

Data normalization refers to the task of rescaling each feature in our dataset to have range [0, 1].

One such method to achieve this is min-max scaling:

$$\tilde{x}_i^{(j)} = \frac{x_i^{(j)} - x_i^{\min}}{x_i^{\max} - x_i^{\min}}$$

Where x_i^{\min}, x_i^{\max} are the minimum and maximum values of feature i in our dataset, respectively.

When training and evaluating your model, you should calculate the parameters for your normalization or standardization function on the training set **ONLY!**

Question 1: Should we always choose x_i^{\min} and x_i^{\max} based on train data? Can we sometimes do better? Think about cases when we have some underlying information about data.

Solution:

Consider RGB images. These are typically encoded as arrays of shape (3, height, width), with each value being an integer in range [0, 255]. In this case we should just use $x_i^{\max} = 255$ and $x_i^{\min} = 0$ to normalize the data.

In general there can be many cases in which we will know max and/or min values of distribution. **Always examine and visualize data** before transforming it.

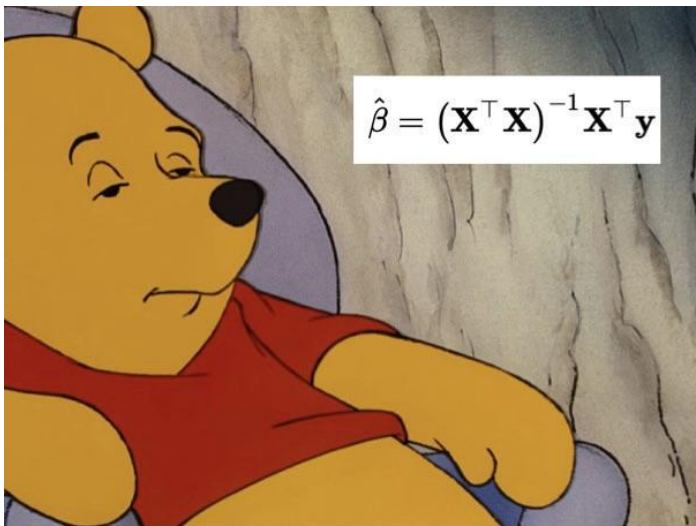
Question 2: When can values outside of $[0, 1]$ range in test set cause issues?

Solution:

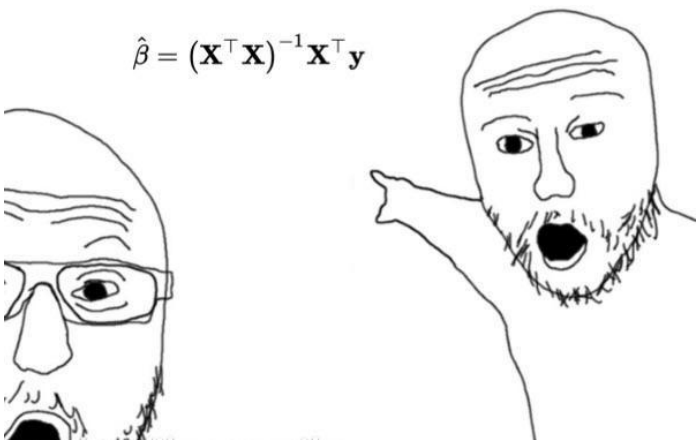
This might lead to an issue if our model performs any transformations on data that have a limited domain. Consider a model $f(x) = \log(x)^T w$. In this case if test datapoint have a value below 0, this code will fail, as log has domain $[0, \infty)$.

In general, after you visualize the data, think about what transforms are needed for it to be well behaved. Always pay attention to domains and ranges of each transform since these may lead to NaNs.

2.1 and 2.2 will be important in HW1!



Employers
when you tell
them your app
uses linear
regression



Employers
when you tell
them your app
uses “machine
learning and
A.I.”

Questions/Chat Time!