Notes

1. Linear Regression in Matrix Form.
2. Normal Equation.
3. Gradient Descent
4. Gradient Descent for Linear Regression

1. Linear Regression in Matrix Form.

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots \beta_p x_p$$

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \cdots \hat{\beta}_p x_p$$

For n training Data:

(sample)

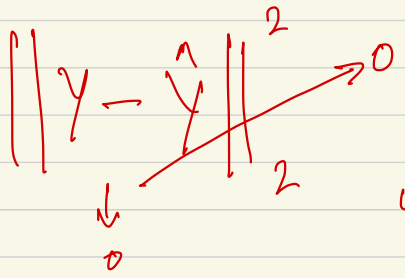
$$\hat{y}^{(1)} = \hat{\beta}_{0(1)} + \hat{\beta}_1 x_1^1 + \hat{\beta}_2 x_2^1 + \cdots \beta_p x_p^1$$

$$\hat{y}^{2} = \hat{\beta}_{0(1)} + \hat{\beta}_1 x_1^2 + \hat{\beta}_2 x_2^2 + \cdots \beta_p x_p^2$$

$$\hat{y}^{n} = \hat{\beta}_{0(1)} + \hat{\beta}_1 x_1^n + \hat{\beta}_2 x_2^n + \cdots \beta_p x_p^n$$

$$\Rightarrow \underset{n \times 1}{\begin{bmatrix} \hat{y} \end{bmatrix}} = \underset{n \times (p+1)}{\begin{bmatrix} X \end{bmatrix}} \underset{(p+1) \times 1}{\begin{bmatrix} \hat{\beta} \end{bmatrix}}$$

We want to minimize.



we estimated $\hat{Y}$ with $\hat{\beta}$ and X

i.e: $Y - X\hat{\beta} = 0.$

$\Rightarrow \hat{\beta} = (X^T X)^{-1} X^T Y$

Pros:

Accurate Analytical Solution

Cons.

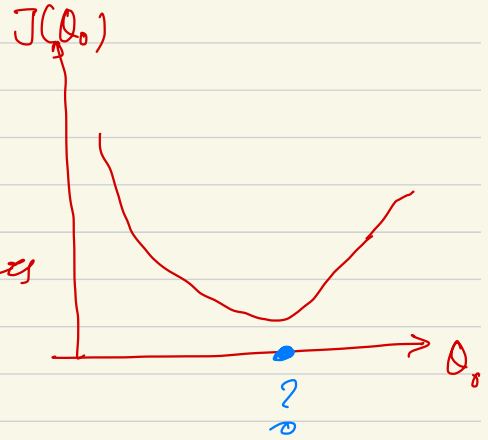
- Matrix X has to be invertible.
- As n and p increases, computationally expensive

Gradient Descent.

- Iterative Optimization Algorithm for finding a local minimum of a differential function.
- Generally involves two steps:
  1. Finding the direction of gradient descent & gradient
  2. Updating arguments based on 1

To understand intuition behind GD we will use a 1-D fn. ie function which depends on 1 i/p variable.

Let $J(\theta_0)$ is some function we want to minimize.



$$\min_{\theta_0} J(\theta_0)$$

→ Give me θ_0 that minimizes my function $J(\theta_0)$

→ We start with some random guess initial value for the parameter we want to find.

→ Usually 0

1. Then we find Gradient of function at that value with respect to each parameters.

$$\frac{dJ(\theta_0)}{d\theta_0}$$

If there are more than 1 variable we take partial derivatives w.r.t each.

Example: $J(\theta_0, \theta_1)$

$\Rightarrow$ Compute: $\frac{\partial J(\theta_0, \theta_1)}{\partial \theta_0}$ and $\frac{\partial J(\theta_0, \theta_1)}{\partial \theta_1}$

(2) In $k+1^{th}$ iteration

Update the parameter θ.

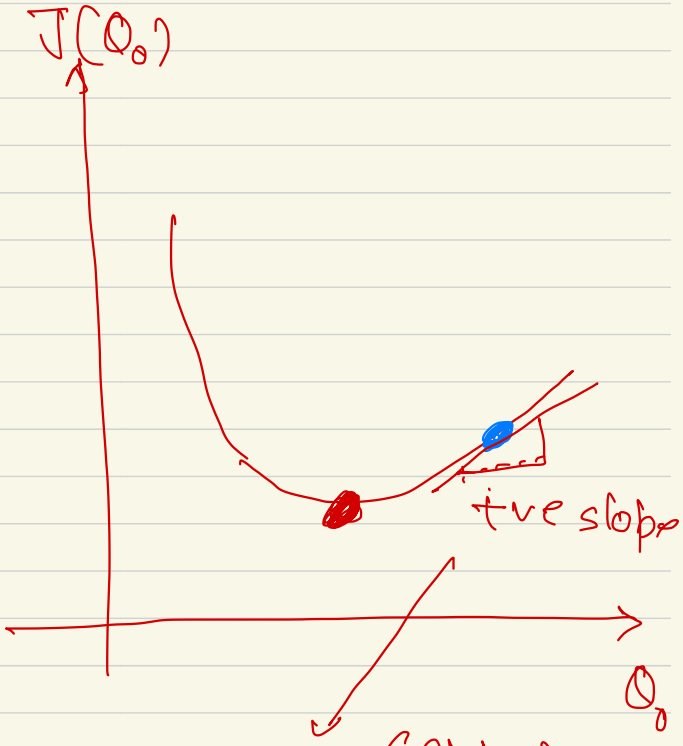
$$\theta_0^{k+1} := \theta_0^k - \alpha \frac{d J(\theta_0)}{d \theta_0}$$

(α → learning rate)

If there are more than 1, we update all acc to:

$$\theta_j := \theta_j - \alpha \frac{\partial J(\theta_0, \theta_1, \ldots \theta_j)}{\partial \theta_j} \qquad j = 1, 2 \ldots p$$

What's happening in Step ② Update:



$\frac{d J(\theta_0)}{d(\theta_0)}$ (slope) $\Rightarrow$ +ve quantity.

$$\theta_0^{k+1} := \theta_0^k - \alpha \frac{d J(\theta_0)}{d \theta_0}$$

(small +ve)

Smaller value of θ_0 than in k^{th}

If our initial guess were on other side of minimum.

Slope → ve.

→ we will move in +ve direction

When do we stop?

→ One general criteria is:

① Stop until our slope is almost 0
At minimum slope = 0.

or

② Fixed # of iteration.

or

③ ~~Until~~ Until the $J^{(k+1)}(\theta) - J^{(k)}(\theta)$ is a some very small value.

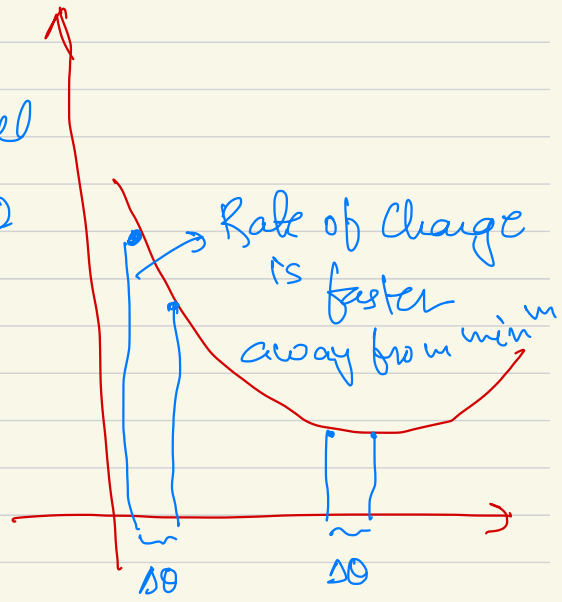
Some Interesting points about Gradient descent and Learning parameter

① Gradient Descent can stuck at local minimum

② Often times on first few iteration the cost decreases rapidly then starts decreasing slowly. This is because the further the away θ is from optimal, gradient is term is higher

③ For small learning rate we are traversing through small rate of change (Because we change θ by a small value).

④ For longer learning rate the rate of change is faster but might also bounce in opposite sides of optimal value.



Simple Gradient Descent in Linear Regression.

Step 1.

$$J(\beta_0, \beta_1) = \frac{1}{2n} \sum_{i=1}^{n} \left(h_\beta(x^{(i)}) - y^{(i)} \right)^2 \quad -(1)$$

where: $h_\beta(x^{(i)})$ = Approx $\hat{y}$ of y using i/p X and parameters β

$$\hat{y} = \hat{\beta}_0 + \beta_1 x_1$$

Taking Partial Derv. of (1) w.r.t β_j

$j = 0, 1$

$$\Rightarrow \frac{\partial J(\beta_0, \beta_1)}{\partial \beta_j} = \frac{\partial}{\partial \beta_j} \frac{1}{2n} \sum_{i=1}^{n} \left(h_\beta(x^{(i)}) - y^{(i)} \right)^2$$

$$= \frac{1}{2n} \frac{\partial}{\partial \beta_j} \sum_{i=1}^{n} \left(\beta_0 + \beta_1 x^{(i)} - y^{(i)} \right)^2$$

For $j = 0$.

$$\Rightarrow \frac{\partial J(\beta_0, \beta_1)}{\partial \beta_0} = \frac{1}{n} \sum_{i=1}^{n} \underbrace{\left(h_\beta(x^i) - y^i \right)}_{\text{Error}}$$

& $$\frac{\partial J(\beta_0, \beta_1)}{\partial \beta_1} = \frac{1}{n} \sum_{i=1}^{n} \underbrace{\left(h_\beta(x^{(i)}) - y^{(i)} \right)}_{\text{Error}} \underbrace{x^{(i)}}_{\text{variable.}}$$

Step 2

Update each β using Update Rule

$$\beta_j^{k+1} = \beta_j^k - \alpha \frac{\partial J(\beta)}{\partial \theta_j}$$

Step 3 Repeat (1) and (2) until convergence.

Gradient Descent For Logistic Regression

Gradient Descent for logistic Regression:

Recall: for 1 i/p feature in logistic Regression

$$P(X) = \frac{e^{\beta_0 + \beta_1 X}}{1 + e^{\beta_0 + \beta_1 X}}$$

$\Rightarrow P(X) > 0.5 \Rightarrow X \in Y=1$ else, $X \in Y=0$

For p inputs

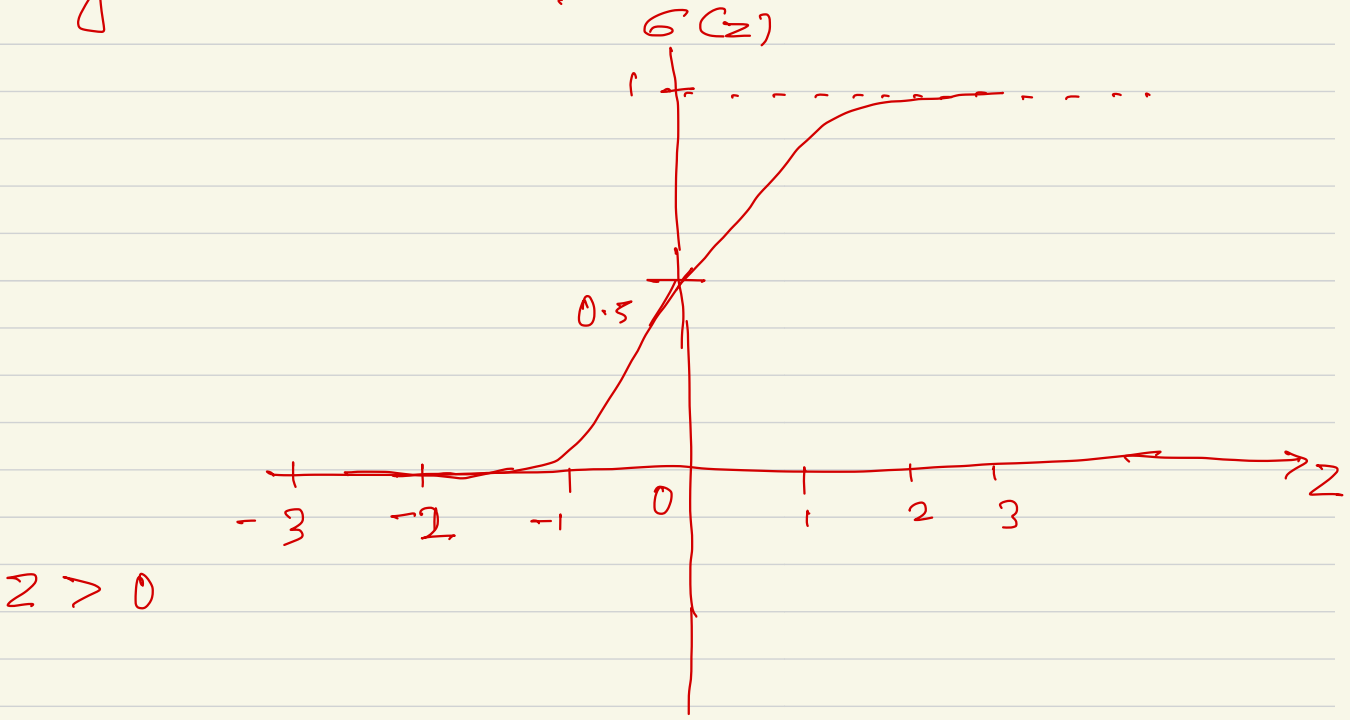
$$P(X) = \frac{e^{\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots \beta_p X_p}}{1 + e^{\underbrace{\beta_0 + \beta_1 X_1 + \cdots \beta_p X_p}_{\beta^T X}}}$$

$$\beta \to R^p$$

Then: $$P(X) = \frac{e^{\beta_0 + \beta^T X}}{1 + e^{\underbrace{\beta_0 + \beta^T X}_{z}}}$$

$$P(X) = \frac{e^z}{1 + e^z} = \frac{1}{1 + e^{-z}}$$

Sigmoid Function:



$z > 0$

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

When we are calculating Prob. in log Reg:

$$P(x) = \sigma(z) \quad \text{where} \quad z = \beta_0 + \beta^T X$$

$$P(y=1) = \sigma(z) = \sigma(\beta_0 + \beta^T X)$$

$$P(y=0) = 1 - \sigma(z)$$

For Sigmoid

$$1 - \sigma(z) = \sigma(-z)$$

$$\therefore \quad P(Y=1) = \sigma(z) = \frac{1}{1+e^{-z}}$$

$$P(Y=0) = 1 - \sigma(z) = \sigma(-z) = \frac{1}{1+e^{+z}}$$

$$= \frac{e^{-z}}{1+e^{-z}}$$

Decision Boundary:

$$\text{Decision}(x) = \begin{cases} 1 : \text{if } P(Y=1|x) > 0.5 \\ 0 \quad \text{Otherwise} \end{cases}$$

We have established:

$$\hat{y} = \sigma(z) = \sigma(\beta_0 + \beta^T x)$$

↳ Note: If $z > 0$ (i.e $\sigma(z) > 0.5$) we decide $\hat{y} = 1$

Recall: Probabilities are calculated "using" sigmoid in logistic Regression.

Loss Function

$L(y, \hat{y})$

↙ True ↘ Predicted

Recall Bernoulli

$$f(k, p) = p^k (1-p)^{1-k}$$

We need to find β_0 and β such that our loss is minimized

In other words,

We need to find β_0 and β such that

$\hat{y} = \sigma(\beta_0 + \beta^T x) = \sigma(z)$ to be closer to y which is either 1 or 0

Bernoulli's Distribution

lets Define a rule that maps $\hat{y}$ to 1 or 0

$$P(y|x) = \hat{y}^y (1-\hat{y})^{1-y} \Rightarrow \text{Likelihood}$$

let's take a look at nature of this expression.

When $y = 1$

$$P(y|x) = \hat{y}^1 (1-\hat{y})^{1-1}$$

$$= \hat{y}$$

Recall: sigmoid will map $\hat{y}$ to 1 when $\sigma(z) = P(y) > 0.5$

When $y = 0$

$$P(y|x) = \hat{y}^0 (1-\hat{y})^{1-0}$$

$$= 1 - \hat{y}$$

So writing $P(y|x) = \hat{y}^y (1-\hat{y})^{1-y}$

Note: Our sigmoid function maps $\hat{y}$ to 1 or to 0 based on the Decision Rule.

Recall: Sigmoid will map $\hat{y}$ to 0, when $\sigma(z) = P(\hat{y}) \leq 0.5$.

Taking log on both:

$$\log P(y|x) = y \log \hat{y} + (1-y) \log (1-\hat{y})$$

⇒ This is called log-likelihood that we want to maximize such that our loss $L(y, \hat{y})$ is minimized.

Now; In terms of loss i.e something we should minimize, if we flip sign of log-likelihood, that's our loss.

$$\text{loss}(y, \hat{y}) = -\log P(y|x)$$

→ We could do this because log is a monotonic function.

$$L_{CE}(y, \hat{y}) = -\log P(y|x)$$

Cross-Entropy loss

$$L_{CE}(y, \hat{y}) = -[y \log \hat{y} + (1-y) \log(1-\hat{y})]$$

Remember: $\hat{y} = \sigma(z)$

$= \sigma(\beta_0 + \beta^T x)$

Example:

Let's see cross-entropy makes sense or not.

Case: → when we made right estimate:

Actual $(y) = 1$ Predicted $(\hat{y}) = 0.7$

$$L_{CE}(y, \hat{y}) = -[1 \cdot \log(0.7) + (1-1) \log(1-0.7)]$$

($\log(1-0.7)$ → 0)

$= -\log(0.7)$

$= 0.36$

Case: When we made wrong estimate:

Actual $(y) = 0$ Predicted $\hat{y} = 0.7$

$$L_{CE}(y, \hat{y}) = -\left[0 \cdot \log(0.7) + (1-0)\log(1-0.7)\right]$$

$$= -\left[\log(0.3)\right]$$

$$= 1.2$$

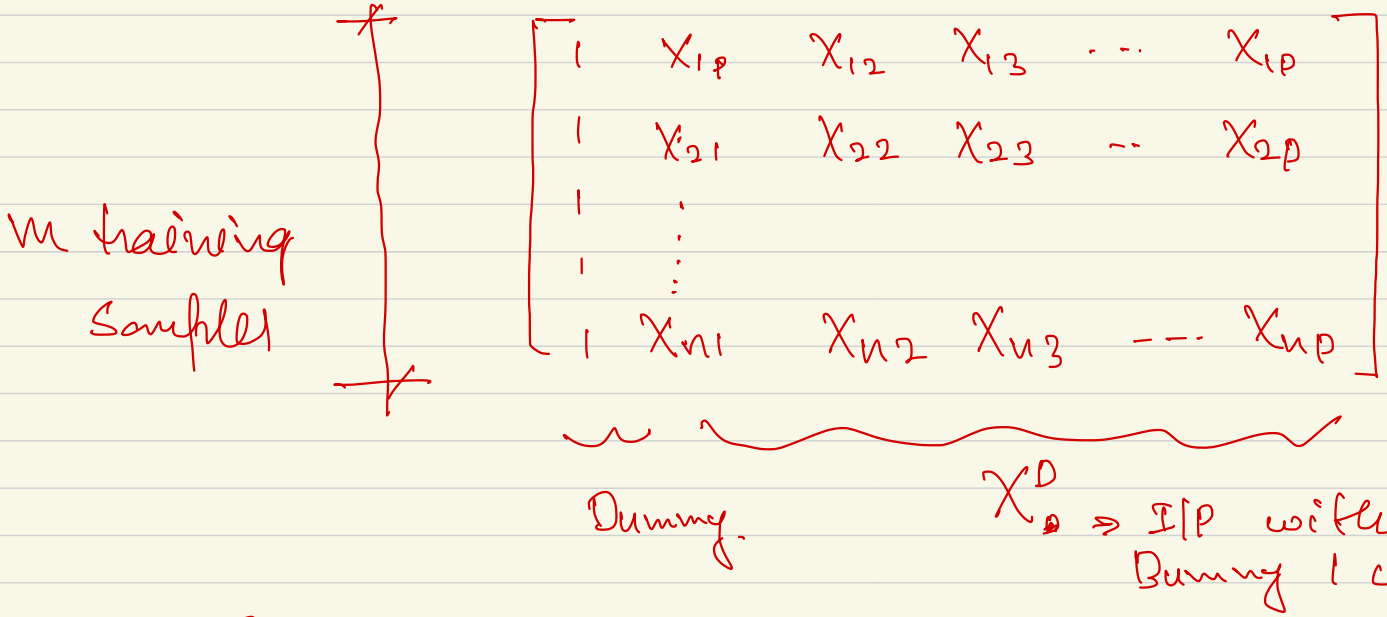
Note: loss is smaller when we make right prediction and vice-versa.

GRADIENT DESCENT.

We want to minimize the loss over all training samples.

Lets say we have n samples

We can rewrite $\beta_0 + \beta^T X$ as:



$$\beta_0 + \beta^T X$$

$$\beta = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{bmatrix}$$

$$\rightarrow \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{bmatrix} \Rightarrow \beta^D$$

Coeff vector with coeff of Dummy i/p included.

$$\Rightarrow \beta_0 + \beta^T X \Rightarrow (\beta^D)^T X$$

$$\hat{\beta}^{D} = \underset{\beta^{D}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} L_{CE}\left(\underbrace{f(x^{(i)}; \beta^{D})}_{\hat{y}}, y^{(i)}\right)$$

$i \Rightarrow i^{th}$ Training sample.

$\Rightarrow$ Minimize average loss over all samples.

Cost Function.

All we need is this $\underset{\sim}{\beta^{D}} \in \mathbb{R}^{p+1}$

$\downarrow$ coefficients.

Gradient Descent:

① Find Gradient

② Update parameters

} Recall our approach in Linear Reg^{n}.

$$L\left(y^{(i)}, \hat{y}^{(i)}\right) = L\left(f_{B^0}(x^{(i)}), y^{(i)}\right)$$

$\hat{y}^{(i)}$

General Form

① $$\frac{\partial L(\hat{y}^{(i)}, y)}{\partial B^0} =$$

$$\begin{bmatrix} \frac{\partial f_{B^0}(x)}{\partial B_0} \\ \frac{\partial f_{B^0}(x)}{\partial B_1} \\ \frac{\partial f_{B^0}(x)}{\partial B_2} \\ \vdots \\ \frac{\partial f_{B^0}(x)}{\partial B_p} \end{bmatrix}$$

→ Find Gradient

$$\nabla L(f_{B^0}(x), y) = \begin{bmatrix} \\ \sim \\ \\ \end{bmatrix}$$

Update:

$$B^0_{t+1} = B^0_t - \alpha \nabla L(f_{B^0}(x), y)$$

$t+1$ ↓ iteration

For logistic Regression:

We know

$$L(y, \hat{y}) \rightarrow L_{CE}(\hat{y}, y)$$

$$\therefore L_{CE}(\hat{y}, y) = -\left[y \log \underbrace{\sigma(B^{0T} X^0)}_{\hat{y}} + (1-y) \log\left(1 - \sigma(B^{0T} X^0)\right) \right]$$

$$\Rightarrow \frac{\partial L_{CE}(\hat{y}, y)}{\partial \beta_j} = \left[\underbrace{\sigma(B^{0T} X^0)}_{\hat{y}} - y \right] X_j$$

$$= [\hat{y} - y] X_j$$

$$j = 0, 1, 2, \ldots p$$

This means:

Derivative of cross entropy loss w.r.to j^{th} Coeff β_j is equal to product of j^{th} input X_j and difference Between predicted ($\hat{y}$) and actual y

Once, we know this Gradient

⇒ We can calculate this for all training samples and calculate average

$$\frac{\partial \frac{1}{n} \sum_{i=1}^{n} L_{CE}\left(f_{BD}(x^{(i)}), y^{(i)}\right)}{\partial \beta_j}$$

$$= \frac{1}{n} \sum \left(\frac{\partial L_{CE}\left(f_{BD}(x^{(i)}), y^{(i)}\right)}{\partial \beta_j} \right)$$

→ We know how to calc. this.

(2) We can update parameters using Gradient.

$$\frac{\partial L_{CE}(y, \hat{y})}{\partial \beta_j} = \frac{1}{n} \sum (\hat{y}^{(i)} - y^{(i)}) x_j^{(i)}$$

$i = 1, 2, \ldots n.$ → #of Training examples.

$j = 0, 1, 2, \ldots P$ → I/P variable

Note:

when $j = 0$, $x_j = 1$

Once we have Gradient:

$$\beta_j^{t+1} = \beta_j^t - \alpha \frac{\partial L_{CE}(y, \hat{y})}{\partial \beta_j}$$

$$= \beta_j^t - \alpha \frac{1}{n} \sum (\hat{y}^{(i)} - y) x_j^{(i)}$$

Random Init. — # of Training. — Note we do $\hat{y} - y$ not $y - \hat{y}$

We are more interested on the sign of $(\hat{y}^{(i)} - y^{(i)})$ than the actual value. Sign give us the direction of the gradient. That is what we are interested in so that we can update our parameters β to get to an optimal.

MSE estimation is a special case of MLE.

→ Recall in Linear Regression, we used Residual Sum of Squares (RSS) mean. i.e Mean Squared Error to estimate parameter β of Linear Regression.

→ But we used Max. Likelihood Estimator (MLE) to estimate parameter β in Logistic, Poisson's and other Regressions.

→ Our goal in this section is to show MSE is actually a special case of MLE.

To show that we will first briefly discuss about MLE.

② Take an Example of Gaussian Dist^n to understand MLE.

Maximum likelihood Estimator

→ Given some Data samples (X_i), $i = 1, \dots N$, MLE allows us to Estimate the parameter of (unknown/known) distribution that maximizes the chances of getting these Data samples.

Given: $x_1, x_2, x_3 \dots x_N$, The likelihood function is defined as:

$$\mathcal{L}(\theta | x_1, x_2, x_3 \dots x_N)$$

Joint probability of observed Data as a function of the parameters θ. of the chosen statistical model.

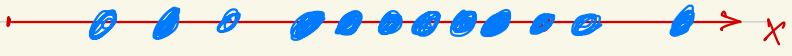
In MLE, we want to estimate parameters θ such that $\mathcal{L}$ the likelihood function is maximized.

$$\text{i.e } \hat{\theta} = \underset{\theta}{\operatorname{argmax}}\, \mathcal{L}(\theta | X)$$

To understand Concept of LE and MLE, lets taken an Example of Statistical Model i.e Gaussian.

Example:

Consider we are measuring size of Tumor (X) in patients.



Now, if we want to model this data is coming from a Gaussian, we need to estimate two parameters

- Mean
- Sigma^2 (Variance)

$\theta : \{(\mu, \sigma^2)\}$

→ Well this is easy right?

Our intuition says, just calculate mean and variance of the sample (Training) Data.

i.e. $\hat{\mu} = \frac{x_1 + x_2 + x_3 + \dots x_n}{N}$

$\sigma \approx x - \hat{\mu}$

But ① is our intuition correct?

② Do max. likelihood estimator give us same estimation of μ and σ?

Gaussian Dist^n:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

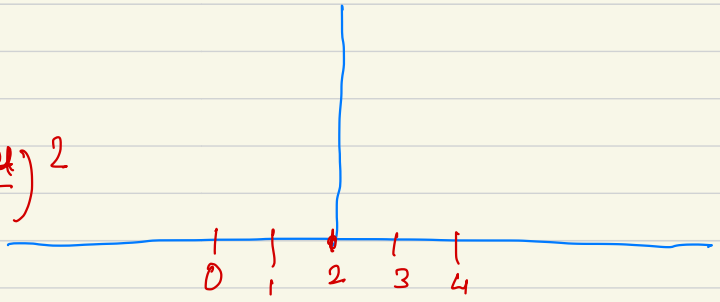
Prob Density function.

Let's see how MLE estimates μ.

Let's say only one sample is available. $x_1 = 2$

Assume we have estimated $\sigma = 2$.

$$f(x) = \frac{1}{\sqrt{2\pi 2^2}} e^{-\frac{1}{2}\left(\frac{2-\mu}{2}\right)^2}$$



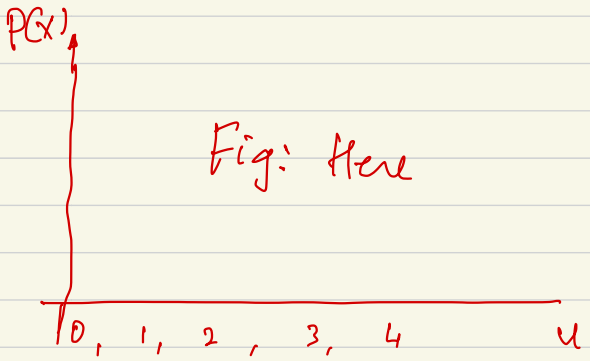


Fig: Here

Lets Say 2 data points are available:

$x = 2, 6$

Then: $L(\mu \mid x=2, x=6) = P(x=2) \cdot P(x=6)$

Because we say samples are independently drawn.

$$= \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{2-\mu}{\sigma}\right)^2} \cdot \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{6-\mu}{\sigma}\right)^2}$$

Fig 2 Here.

Similarly: If n samples are given:

$$\mathcal{L}(\mu,\sigma \mid x = x_1, x_2, \ldots x_n)$$

$$= \prod_{i=1}^{n} p(x_i) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{x_i-\mu}{\sigma}\right)^2}$$

$$= \prod_{i=1}^{n} \text{Prob. density function of the model.}$$

If you want to prove that for Gaussian, optimal μ estimator from sample Data is mean (average) of the sample data.

We could take partial derivative of $\mathcal{L}(\mu,\sigma)$ w.r.t μ and σ and setting them to zero.

$$\Rightarrow \quad \mu = \frac{x_1 + x_2 + \cdots x_n}{n}$$

$$\sigma^2 = \frac{(x_1-\mu)^2 + (x_2-\mu)^2 + \cdots (x_n-\mu)^2}{n}$$

MSE as a special case of MLE

Now, we know about MLE, lets show that MSE is a special case of MLE

when we assume the statistical model to be Gaussian.

In Linear Regression:

$$y = \beta_0 + \beta_1 x_1 + \cdots \beta_p x_p + \epsilon$$

→ p inputs not samples

Now; In L-R we assume our model is Gaussian,

So: $y \sim N(u, \sigma^2)$

→ Note this σ^2 is not the σ^2 of noise ϵ.

So: likelihood of y:

$$\mathcal{L}(u, \sigma \mid X = x_1, x_2 \cdots x_n) = \sqrt{\frac{1}{2\pi\sigma^2}} \prod_{x=x_1,x_2 \cdots x_n} e^{-\frac{(y-u)^2}{2\sigma^2}}$$

we are estimating u ~~[illegible]~~ by our parameter β

$$\therefore \mathcal{L}(\beta, \sigma^2) = \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n \prod_{i=1}^{n} e^{-\frac{(y_i - \beta^T x_i)}{2\sigma^2}}$$

∴ log-likelihood.

$$\log \mathcal{L}(\beta, \sigma^2) = \log\left[\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n \prod_{i=1}^{n} e^{-\frac{(y_i - \beta^T x_i)^2}{2\sigma^2}}\right]$$

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

Now: If we say negative of likelihood as our loss:

Then: loss $= -\log \mathcal{L}(\beta, \sigma^2)$

$$\text{Loss} = -n \log\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) + \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - \beta^T x_i)^2 \quad (-B)$$

Now Recall that: We are interested in optimal solution not optimal value.

So, We are are interested on those values of β (parameter of L.R) which minimize loss $(-B)$ or equiv. mon. likelihood in (A).

Which we can find by setting Partial derivative of loss to 0.

That is parameters β that minimizes (B)

also minimize any scalar multiplication or addition to function (B).

i.e

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}}\ \mathcal{L}(\beta, \sigma)$$

$$= \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{n} \left(y_i - \beta^T x_i\right)^2$$

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

↳ Sum of square Error.

Hence: In linear Regression, ~~MLE~~ MSE is a special case of MLE when we assume the model is Gaussian.

In Conclusion:

In linear Regression, Logistic Regression, Poisson's Regression,

we are always estimating the parameter (β) of the model which maximizes the likelihood of the training data.

So, all these methods are based on Maximum likelihood Estimation.