Bias-Variance Trade-Off

Introduction

Let us talk about the weather. It rains only if it's a little humid and does not rain if it's windy, hot or freezing. In this case, how would you train a predictive model and ensure that there are no errors in forecasting the weather? You may say that there are many learning algorithms to choose from. They are distinct in many ways but there is a major difference in what we expect and what the model predicts. That's the concept of Bias and Variance Tradeoff.

Bias-Variance Trade-Off

A machine learning model's performance is evaluated based on how accurate is its prediction and how well it generalizes on another independent dataset it has not seen

The errors in a machine learning model can be broken down into 2 parts:
1. Reducible Error
2. Irreducible Error

Irreducible errors are errors that cannot be reduced even if you use any other machine learning model.

What exactly is Bias?

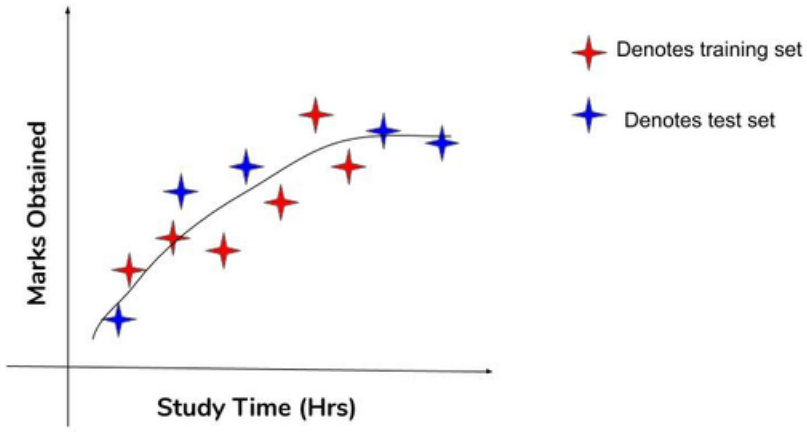
Bias is the inability of a machine learning model to capture the true relationship between the data variables. It is caused by the erroneous assumptions that are inherent to the learning algorithm. For example, in linear regression, the relationship between the X and the Y variable is assumed to be linear, when in reality the relationship may not be perfectly linear.

Bias is defined as the difference of the average value of prediction from the true relationship function *f(x)*

$$\text{bias}[\hat{f}(x)] = \mathbb{E}[\hat{f}(x)] - f(x)$$

What exactly is Bias?

This graph shows the original relationship between the variables. Notice, there is a limit to the marks you can get on the test. That is even if you study an extraordinary amount of time, there is always a certain 'maximum mark' you can score. You can see the line flattening beyond a certain value of the X-axis. So the relationship is only piecewise linear. This sort of error will not be captured by the vanilla linear regression model.

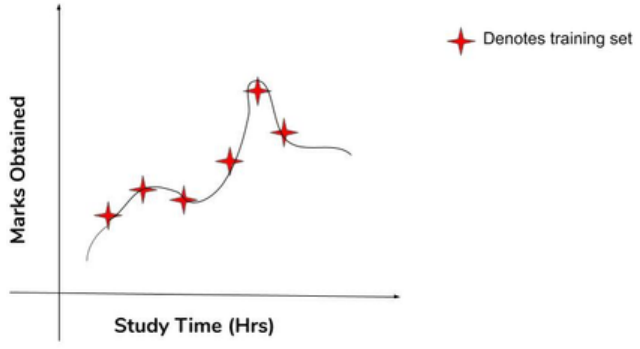


You can expect an algorithm like linear regression to have high bias error, whereas an algorithm like decision tree has lower bias. Why? because decision trees don't make such hard assumptions. So is the case with algorithms like k-Nearest Neighbours, Support Vector Machines, etc

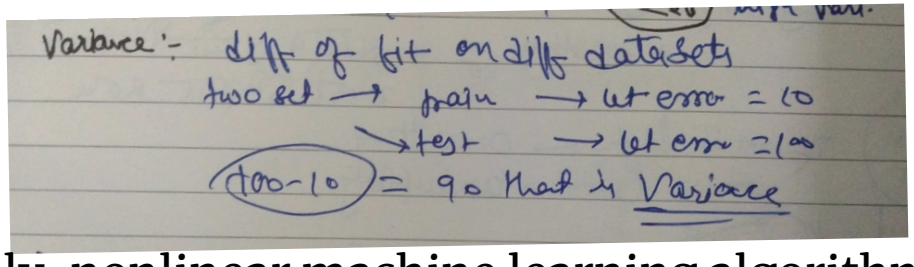
- High Bias indicates more assumptions in the learning algorithm about the relationships between the variables.
- Less Bias indicates fewer assumptions in the learning algorithm.

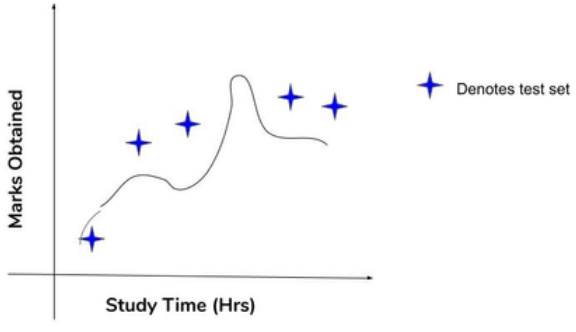
What is the Variance Error?

This is nothing but the concept of the model overfitting on a particular dataset. If the model learns to fit very closely to the points on a particular dataset, when it used to predict on another dataset it may not predict as accurately as it did in the first.



Variance is the difference in the fits between different datasets





- Generally, nonlinear machine learning algorithms like decision trees have a high variance. It is even higher if the branches are not pruned during training.
- Low-variance ML algorithms: Linear Regression, Logistic Regression, Linear Discriminant Analysis.
- High-variance ML algorithms: Decision Trees, k-NN, and Support Vector Machines.

Mathematically

We have independent variables *x* that affect the value of a dependent variable *y*. *F*unction *f* denotes the true relationship between *x* and *y*. In real life problems it is very hard to know this relationship. *y* is given by this formula along with some noise which is represented by the random variable ϵ with zero mean and variance $\sigma\epsilon^2$:

$$y = f(x) + \epsilon$$

$$\text{bias}[\hat{f}(x)] = \mathbb{E}[\hat{f}(x)] - f(x)$$

Bias is defined as the difference of the average value of prediction from the true relationship function *f(x)*.

$$\text{var}(\hat{f}(x)) = \mathbb{E}[(\hat{f}(x) - \mathbb{E}[\hat{f}(x)])^2]$$

Variance is defined as the expectation of the squared deviation of $\hat{f}(x)$ from its expected value $\mathbb{E}[\hat{f}(x)]$.

$$\begin{aligned}
\mathbb{E}[(y - \hat{f}(x))^2] &= \mathbb{E}[(f(x) + \epsilon - \hat{f}(x))^2] \\
&= \mathbb{E}[(f(x) - \hat{f}(x))^2] + \mathbb{E}[\epsilon^2] + 2\mathbb{E}[(f(x) - \hat{f}(x))\epsilon] \\
&= \mathbb{E}[(f(x) - \hat{f}(x))^2] + \underbrace{\mathbb{E}[\epsilon^2]}_{=\sigma_\epsilon^2} + 2\mathbb{E}[(f(x) - \hat{f}(x))] \underbrace{\mathbb{E}[\epsilon]}_{=0} \\
&= \mathbb{E}[(f(x) - \hat{f}(x))^2] + \sigma_\epsilon^2
\end{aligned}$$

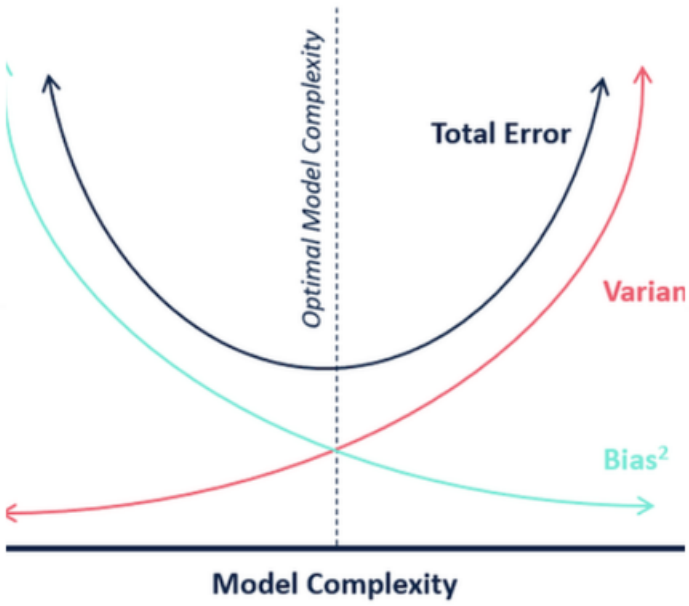
Mathematically

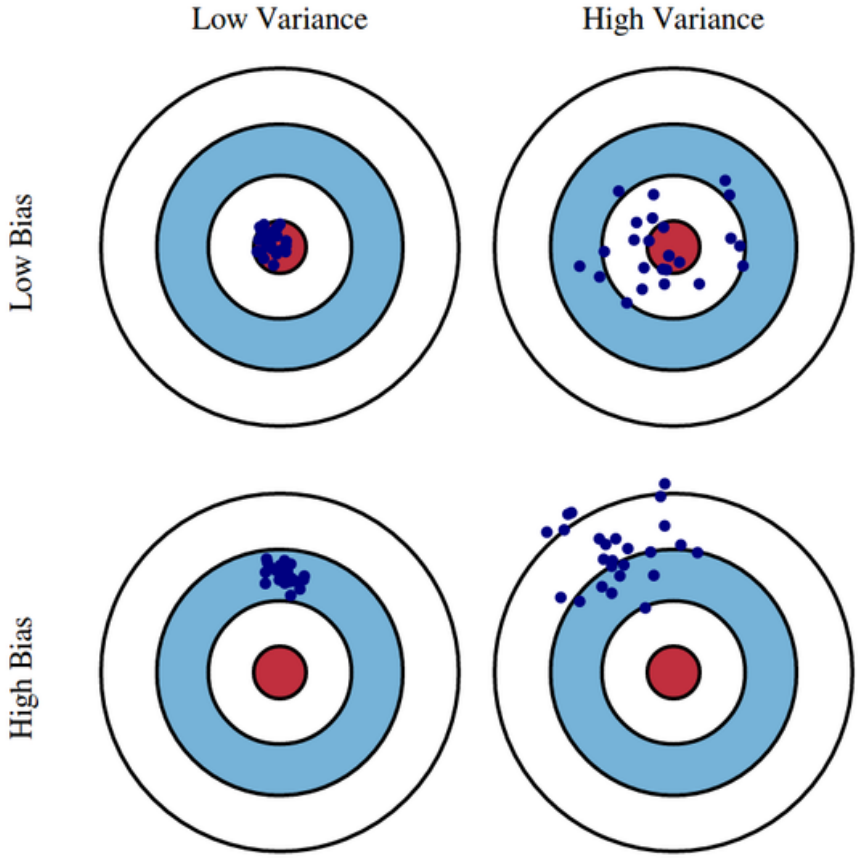
we proceed by expanding further using the linear property of expectation and independence of the random variables ϵ and f̂. Then using the properties of ϵ and the fact that hat when two random variables are independent, the expectation of their product is equal to the product of their expectations.

$$
\begin{aligned}
\mathbb{E}[(f(x) - \hat{f}(x))^2] &= \mathbb{E}\left[\left((f(x) - \mathbb{E}[\hat{f}(x)]) - (\hat{f}(x) - \mathbb{E}[\hat{f}(x)])\right)^2\right] \\
&= \mathbb{E}\left[\left(\mathbb{E}[\hat{f}(x)] - f(x)\right)^2\right] + \mathbb{E}\left[\left(\hat{f}(x) - \mathbb{E}[\hat{f}(x)]\right)^2\right] \\
&\quad - 2\mathbb{E}\left[\left(f(x) - \mathbb{E}[\hat{f}(x)]\right)\left(\hat{f}(x) - \mathbb{E}[\hat{f}(x)]\right)\right] \\
&= \underbrace{(\mathbb{E}[\hat{f}(x)] - f(x))^2}_{=\text{bias}[\hat{f}(x)]} + \underbrace{\mathbb{E}\left[\left(\hat{f}(x) - \mathbb{E}[\hat{f}(x)]\right)^2\right]}_{=\text{var}(\hat{f}(x))} \\
&\quad - 2\left(f(x) - \mathbb{E}[\hat{f}(x)]\right)\mathbb{E}\left[\left(\hat{f}(x) - \mathbb{E}[\hat{f}(x)]\right)\right] \\
&= \text{bias}[\hat{f}(x)]^2 + \text{var}(\hat{f}(x)) \\
&\quad - 2\left(f(x) - \mathbb{E}[\hat{f}(x)]\right)\left(\mathbb{E}[\hat{f}(x)] - \mathbb{E}[\hat{f}(x)]\right) \\
&= \text{bias}[\hat{f}(x)]^2 + \text{var}(\hat{f}(x))
\end{aligned}
$$

$$
\mathbb{E}[(y - \hat{f}(x))^2] = \text{bias}[\hat{f}(x)]^2 + \text{var}(\hat{f}(x)) + \sigma_\epsilon^2
$$

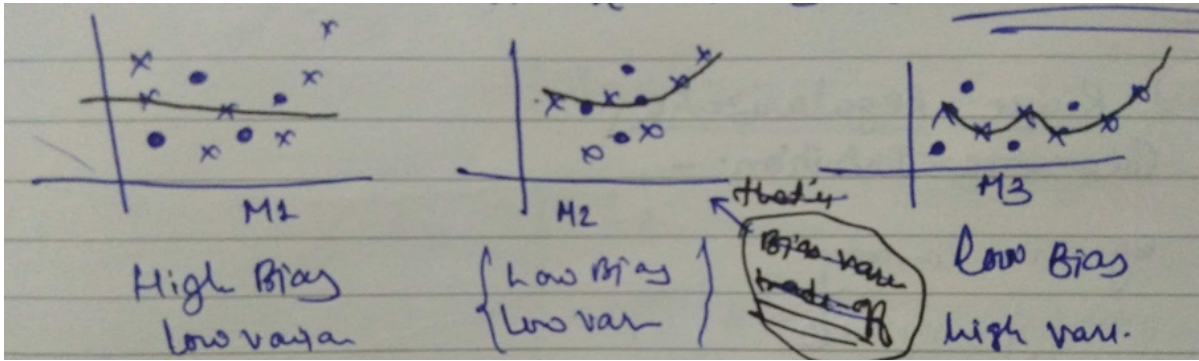
Bias – Variance Tradeoff



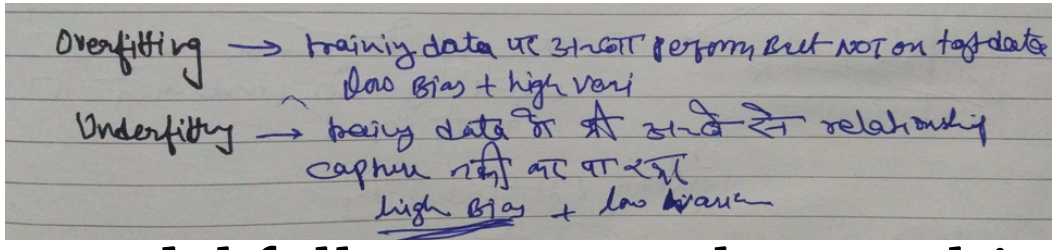


Example a Coin, Student,Company...

How to Overcome:
Regularization, Bagging, Boosting



Let's summarize

1. 

2. If a model follows a complex machine learning model, then it will have high variance and low bias(overfitting the data).
3. You need to find a good balance between the bias and variance of the model we have used. This tradeoff in complexity is what is referred to as bias and variance tradeoff. An optimal balance of bias and variance should never overfit or underfit the model.
4. This tradeoff applies to all forms of supervised learning: classification, regression, and structured output learning.

Regularization

- This technique prevents the model from overfitting by adding extra information to it.
- It is a form of regression that shrinks the coefficient estimates towards zero. In other words,this technique forces us not to learn a more complex or flexible model, to avoid the problem of overfitting.
- In the Regularization technique, we reduce the magnitude of the independent variables by keeping the same number of variables". It maintains accuracy as well as a generalization of the model

How does Regularization Work?

Regularization works by adding a penalty or complexity term or shrinkage term with Residual Sum of Squares (RSS) to the complex model.

Regularization

Techniques of Regularization

Mainly, there are two types of regularization techniques, which are given below:

- Ridge Regression
- Lasso Regression
- other is Elastic Net

Regularization

A simple relation for linear regression looks like this. Here Y represents the learned relation and β represents the coefficient estimates for different variables or predictors(X).

$Y \approx \beta 0 + \beta 1X1 + \beta 2X2 + ...+ \beta pXp$

The fitting procedure involves a loss function, known as residual sum of squares or RSS. The coefficients are chosen, such that they minimize this loss function.

$$\mathrm{RSS} = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 .$$

Now, this will adjust the coefficients based on your training data. If there is noise in the training data, then the estimated coefficients won't generalize well to the future data. This is where regularization comes in and shrinks or regularizes these learned estimates towards zero.

Ridge Regression

$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}\beta_j^2=\mathrm{RSS}+\lambda\sum_{j=1}^{p}\beta_j^2$$

Above image shows ridge regression, where the RSS is modified by adding the shrinkage quantity. Now, the coefficients are estimated by minimizing this function. Here, **λ is the tuning parameter that decides how much we want to penalize the flexibility of our model**. The increase in flexibility of a model is represented by increase in its coefficients, and if we want to minimize the above function, then these coefficients need to be small. This is how the Ridge regression technique prevents coefficients from rising too high. Also, notice that we shrink the estimated association of each variable with the response, except the intercept β0, This intercept is a measure of the mean value of the response when xi1 = xi2 = ...= xip = 0.

Ridge Regression

When λ = 0, the penalty term has no effect, and the estimates produced by ridge regression will be equal to least squares. However, as λ→∞, the impact of the shrinkage penalty grows, and the ridge regression coefficient estimates will approach zero. As can be seen, selecting a good value of λ is critical. Cross validation comes in handy for this purpose. The coefficient estimates produced by this method are also known as the L2 norm.

The coefficients that are produced by the standard least squares method are scale equivariant, i.e. if we multiply each input by c then the corresponding coefficients are scaled by a factor of 1/c. Therefore, regardless of how the predictor is scaled, the multiplication of predictor and coefficient(Xjβj) remains the same. However, this is not the case with ridge regression, and therefore, we need to standardize the predictors or bring the predictors to the same scale before performing ridge regression. The formula used to do this is given below.

$$\tilde{x}_{ij} = \frac{x_{ij}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \overline{x}_j)^2}}$$

Lasso Regression

$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}|\beta_j| \;=\mathrm{RSS}+\lambda\sum_{j=1}^{p}|\beta_j|.$$

Lasso is another variation, in which the above function is minimized. Its clear that this variation differs from ridge regression only in penalizing the high coefficients. It uses |βj|(modulus)instead of squares of β, as its penalty. In statistics, this is known as the L1 norm. Lets take a look at above methods with a different perspective. The ridge regression can be thought of as solving an equation, where summation of squares of coefficients is less than or equal to s. And the Lasso can be thought of as an equation where summation of modulus of coefficients is less than or equal to s. Here, s is a constant that exists for each value of shrinkage factor λ. These equations are also referred to as constraint functions.

Lasso Regression

Consider their are 2 parameters in a given problem. Then according to above formulation, the ridge regression is expressed by $\beta1^2 + \beta2^2 \leq s$. This implies that ridge regression coefficients have the smallest RSS(loss function) for all points that lie within the circle given by $\beta1^2 + \beta2^2 \leq s$.
Similarly, for lasso, the equation becomes, $|\beta1|+|\beta2| \leq s$. This implies that lasso coefficients have the smallest RSS(loss function) for all points that lie within the diamond given by $|\beta1|+|\beta2| \leq s$.

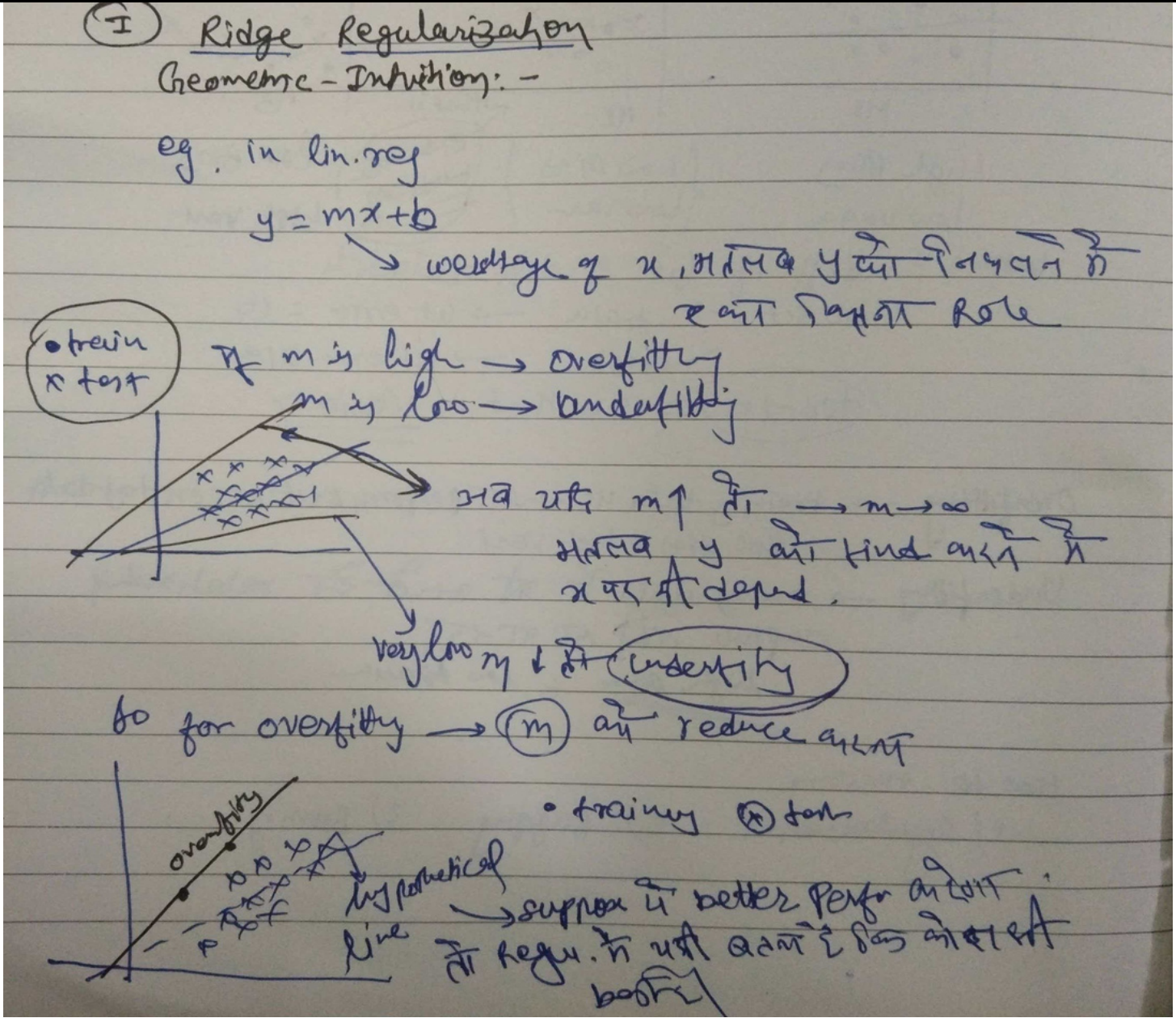
the lasso method also performs variable selection and is said to yield sparse models.

Lasso stands for least absolute shrinkage and selection operator

What does Regularization achieve?

A standard least squares model tends to have some variance in it, i.e. this model won't generalize well for a data set different than its training data. Regularization, significantly reduces the variance of the model, without substantial increase in its bias. So the tuning parameter λ, used in the regularization techniques described above, controls the impact on bias and variance. As the value of λ rises, it reduces the value of coefficients and thus reducing the variance. Till a point, this increase in λ is beneficial as it is only reducing the variance(hence avoiding overfitting), without loosing any important properties in the data. But after certain value, the model starts loosing important properties, giving rise to bias in the model and thus underfitting. Therefore, the value of λ should be carefully selected.

Ridge(Geometric Intuition)



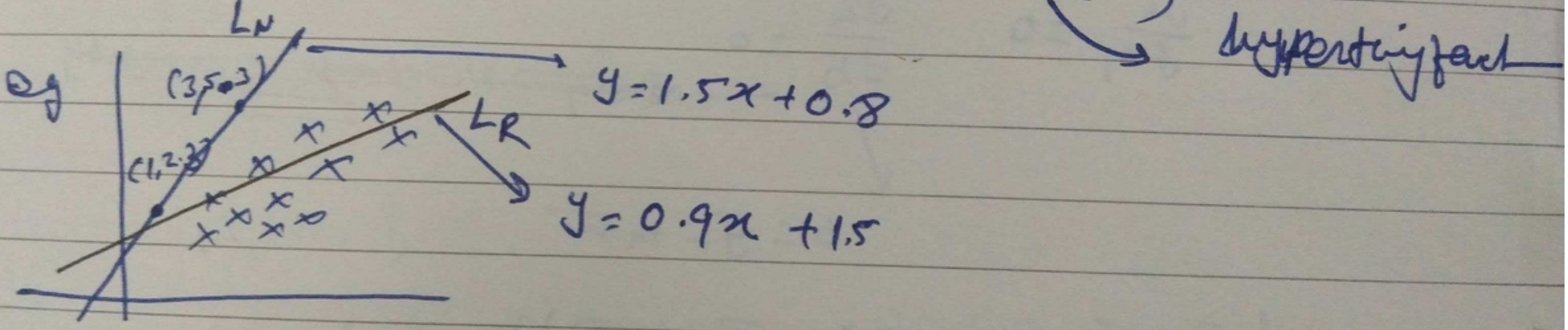
Ridge(Geometric Intuition)

$$L = \sum_{i=1}^{n} (\theta_i - \hat{y}_i)^2 \longleftarrow \text{minimize}$$

In Regularization

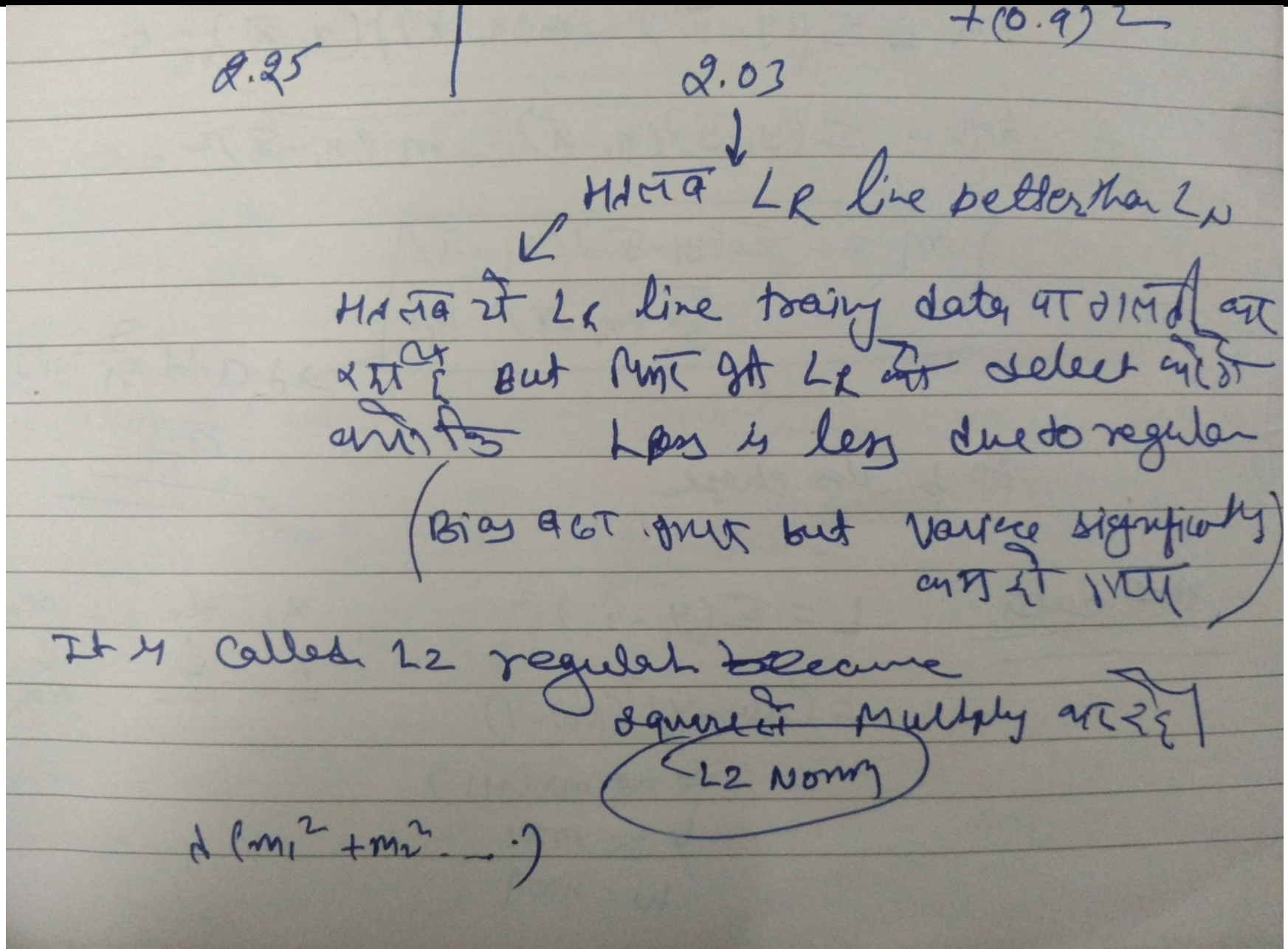
$$L = \sum (y_i - \hat{y}_i)^2 + \alpha(m^2)$$

hyperparameter

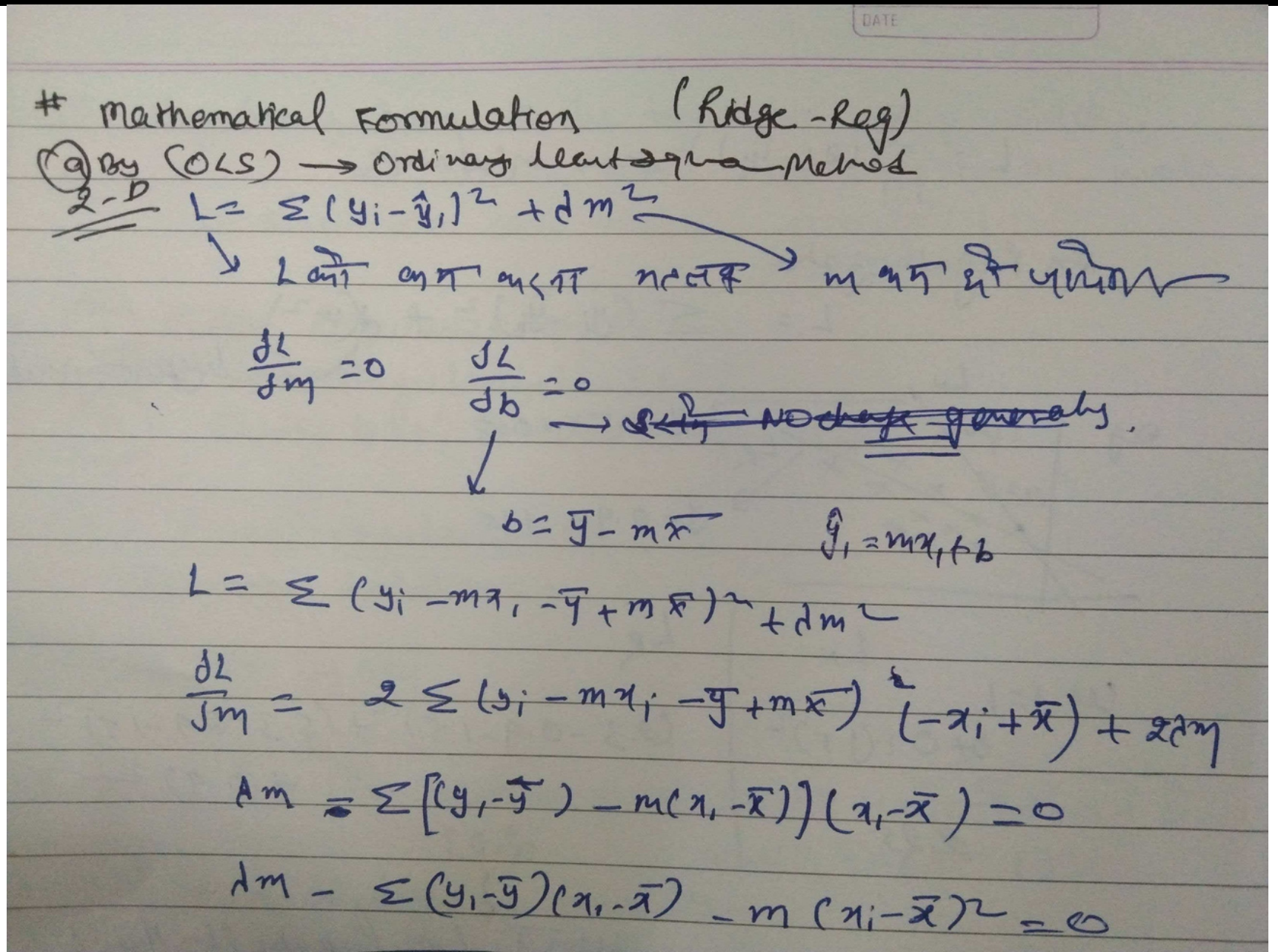


L_N	L_R
Let $\alpha = 1$	$(2.3 - 0.9 - 1.5)^2 + (5.3 - 2.7 - 1.5)^2$
$0 + 0 + (1.5)^2$	$+ (0.9)^2$
2.25	2.03

Ridge(Geometric Intuition)



Ridge(Mathematical Formulation)



Ridge(Mathematical Formulation)

$$\frac{\partial L}{\partial m} = \sum [(y_i - \bar{y}) - m(x_i - \bar{x})](x_i - \bar{x}) = 0$$

$$\frac{\partial L}{\partial m} = \sum (y_i - \bar{y})(x_i - \bar{x}) - m \sum (x_i - \bar{x})^2 = 0$$

$$m = \frac{\sum (y_i - \bar{y})(x_i - \bar{x})}{\sum (x_i - \bar{x})^2 + \lambda}$$

→ λ · add करने पर

So b also change

Q(1)

n-D data

$$L = \sum (y_i - \hat{y}_i)^2$$

$$= (xw - y)^T (xw - y)$$

$x \rightarrow m \times (n+1)$

$y = m \times 1$

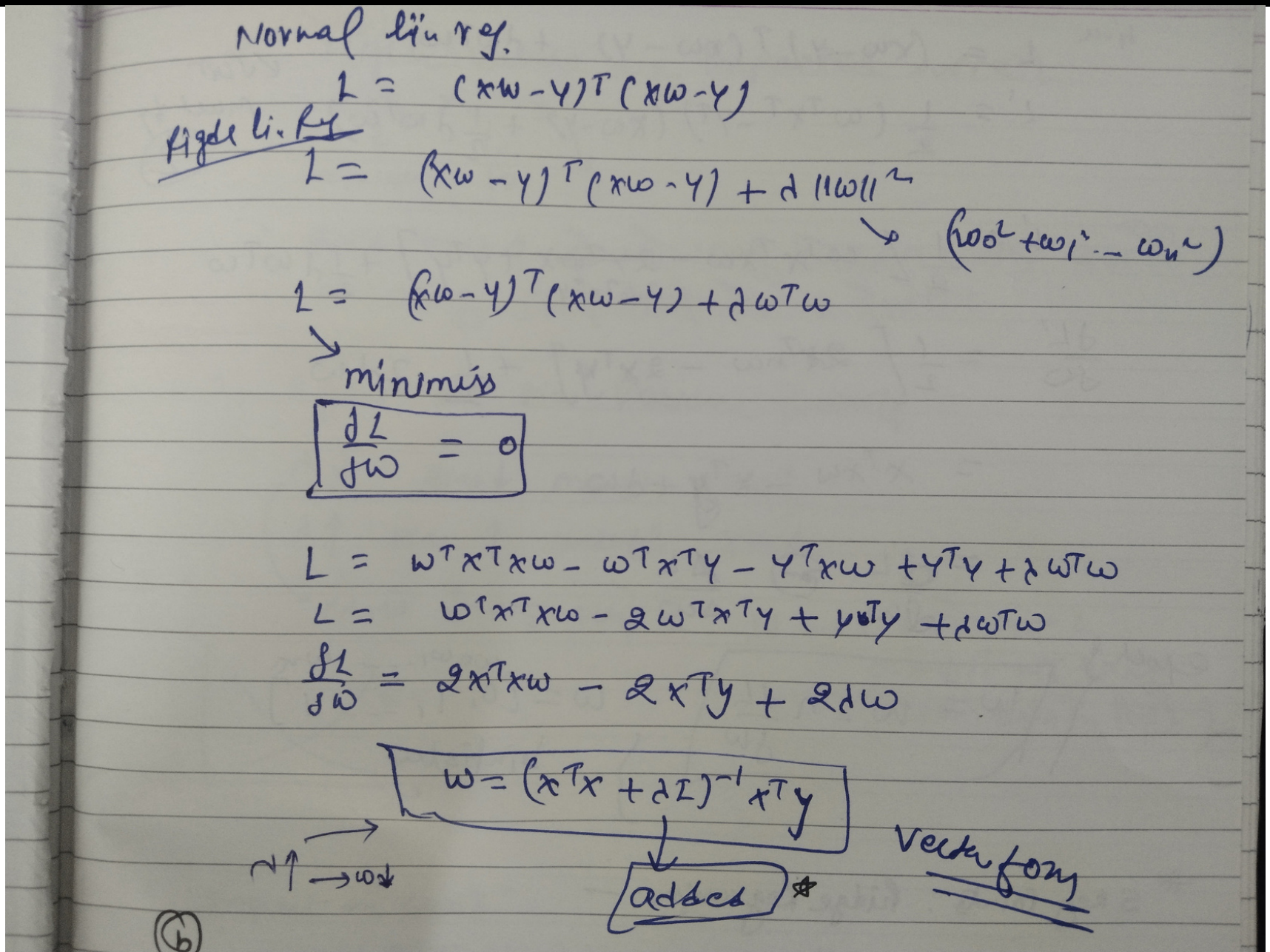
$w = n \times 1$

$x_1 \quad x_2 \quad \dots \quad x_n$

$\downarrow \quad \downarrow \quad \quad \downarrow$

$w_1 \quad w_2 \quad \quad w_n$

Ridge(Mathematical Formulation)



Ridge(Using GD)

(b)

Ridge Reg using Gradient Descent

$$w_{new} = w_{old} - \eta \frac{\Delta L}{\Delta w} \rightarrow \text{gradient } \frac{dL}{dw_1} \ldots$$

$$w_0 = w_0 - \eta \frac{\partial L}{\partial w_0}$$

$$w_1 = w_1 - \eta \frac{\partial L}{\partial w_1}$$

$$\vdots$$

$$\}$$

Ridge(Using GD)

Given $L = (Xw - Y)^T (Xw - Y) + \lambda w^T w$

$L' = \frac{1}{2}(w^T X^T - Y^T)(Xw - Y) + \frac{1}{2}\lambda w^T w$ (Just multiply by 1/2)

$L' = \frac{1}{2}[w^T X^T X w - 2Y^T X w + Y^T Y] + \frac{1}{2}\lambda w^T w$ ($2w^T X^T Y$)

$\frac{dL'}{dw} = \frac{1}{2}[2X^T X w - 2X^T Y] + \frac{1}{2} \cdot 2\lambda w$

$= X^T X w - X^T Y + \lambda w$

$= \frac{dL}{dw}$ (or) $\frac{\Delta L}{\Delta w}$

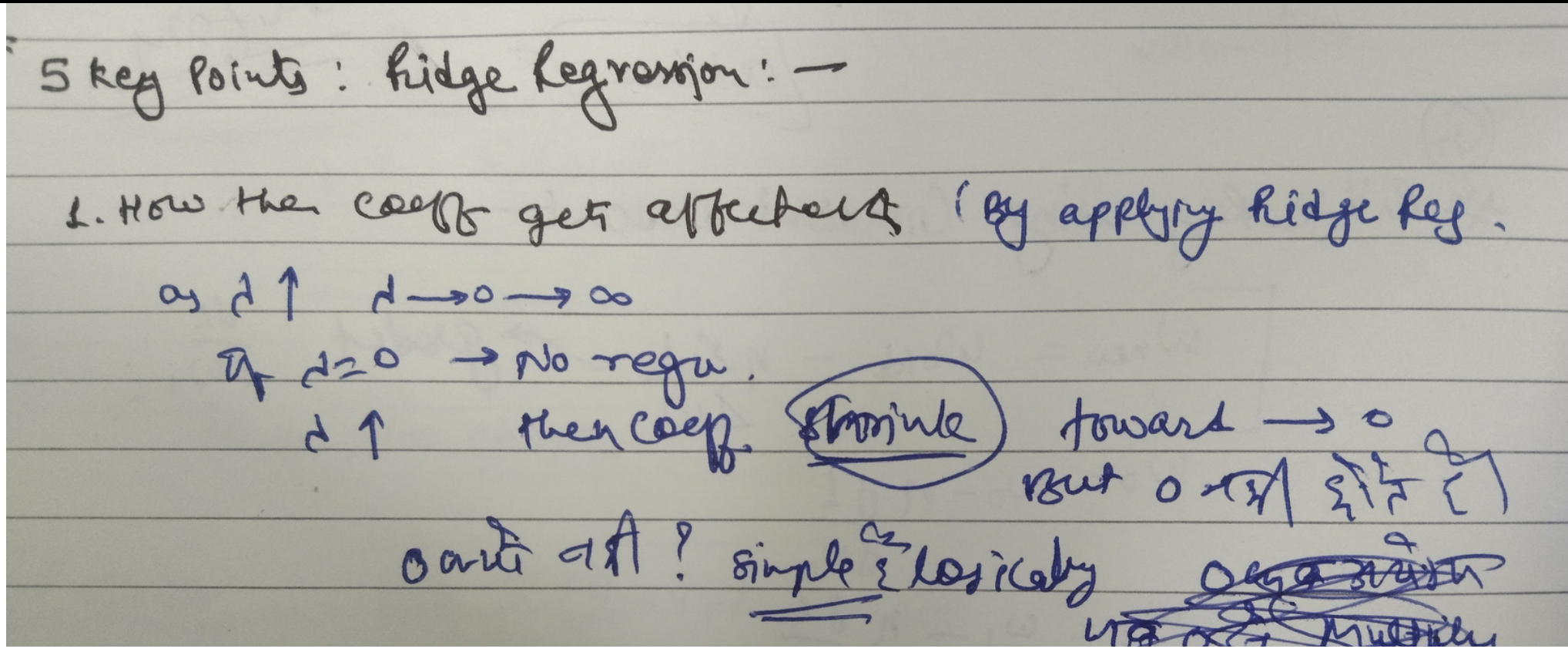
epoch

$w = w - \eta \frac{dL}{dw}$

$w = [w_0, w_1, \ldots, w_n]$

$w = [0, 1, \ldots, 1]$ initially

5 key Points (Ridge Regression)



5 key Points (Ridge Regression)

2. (Coeffs.) Higher values are impacted more!

x_1 x_2 x_3

$w_1 = 1000$ 10 1

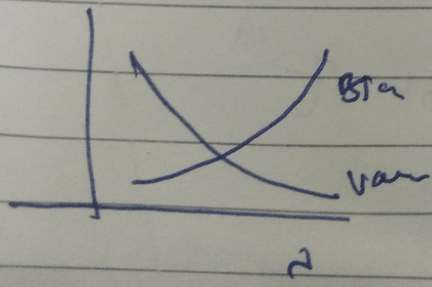
- ~~as~~ as $\lambda \uparrow$ तो जो coeffs जितना ज्यादा बड़ा है वो coeffs उतनी तेजी से ↓ होगा.

3. Effect on Bias - Variance Tradeoff

$\lambda \downarrow$ Bias ↓ overfit var ↑

$\lambda \uparrow$ Bias ↑ underfit var ↓

मतलब ये बहुत सरल है $\lambda \downarrow \rightarrow \lambda \downarrow$ तो बस λ overfit बहुत ज्यादा (train data के लिए)



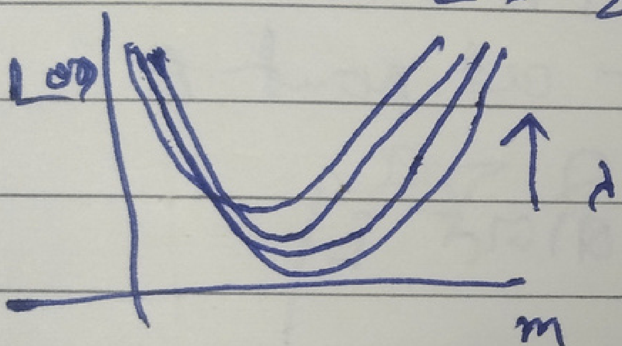
5 key Points (Ridge Regression)

4. Impact on Loss f^n

$$L = \sum (y_i - \hat{y}_i)^2 + \lambda \|w\|^2$$

let $b = \text{const}$ (say $= 0$)

$$L = \sum (y_i - m x_i)^2 + \lambda m^2$$



λ↑ ⇒ coeff ↓ → tends to 0

λ↑ loss बढ़ता जा और graph close होता जाता | (shrink)

towards centre जा रहा

~~(towards 0)~~

5 key Points (Ridge Regression)

5. Why called Ridge reg. → Ridge

Soft Constraint Ridge Reg.

One more **Hard Constraint** Ridge Reg.

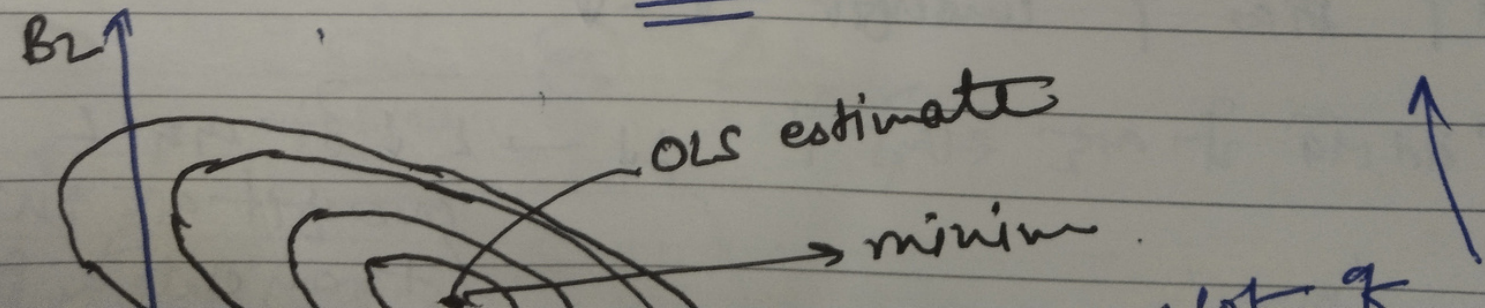
$L_R = MSE + \lambda \|w\|^2$ Let two coeff B_1, B_2,

$MSE = \sum (y_i - (B_0 + B_1 x_{i1} + B_2 x_{i2}))^2$

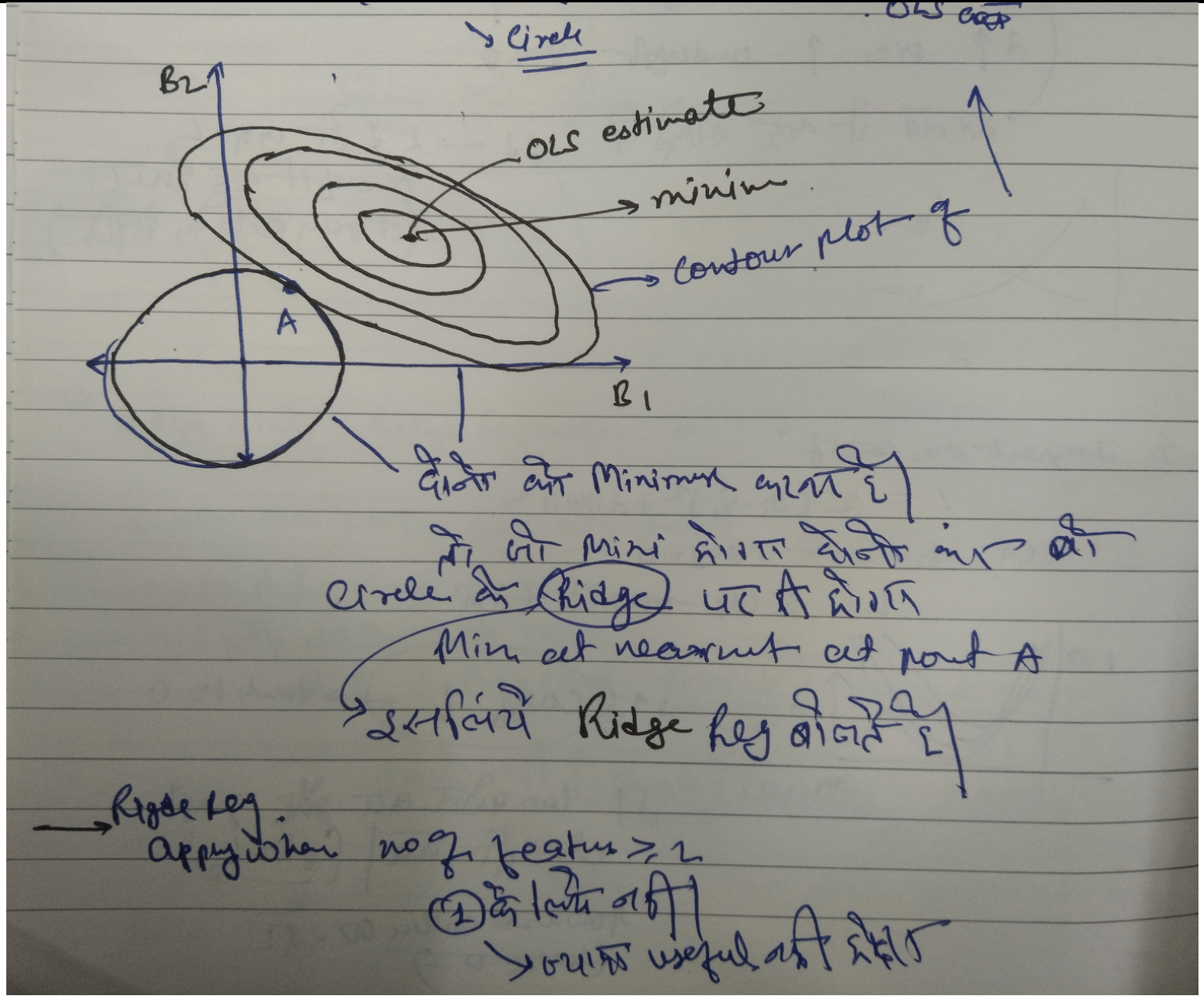
$\lambda (B_1^2 + B_2^2)$

↳ Circle

→ OLS cost



5 key Points (Ridge Regression)



Lasso Regression (L1 Regularization)

Lasso Regression (L1 Regularization)

$L = MSE + \lambda \|w\|_1$ $\quad \|w\|_1 \rightarrow$ L1 Norm

$\lambda [|w_1| + |w_2| \dots + |w_n|]$

$\lambda \uparrow$ so जो coeff कम imp है वो 0 हो जायेंगे जो कि Ridge में नहीं हो रहे थे।

So feature selection करता, dim. ↓ automatically

So Lasso is prefer (mostly higher dim.)

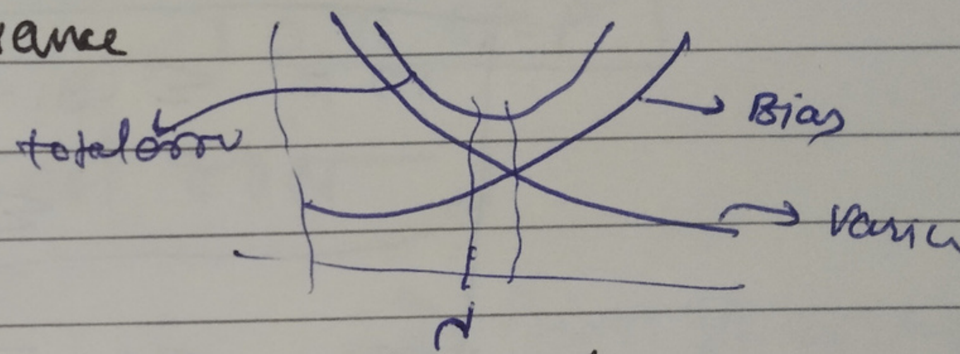
Key Points

1. How are coff affected? feature selection by λ But value नहीं ; Right underfitting

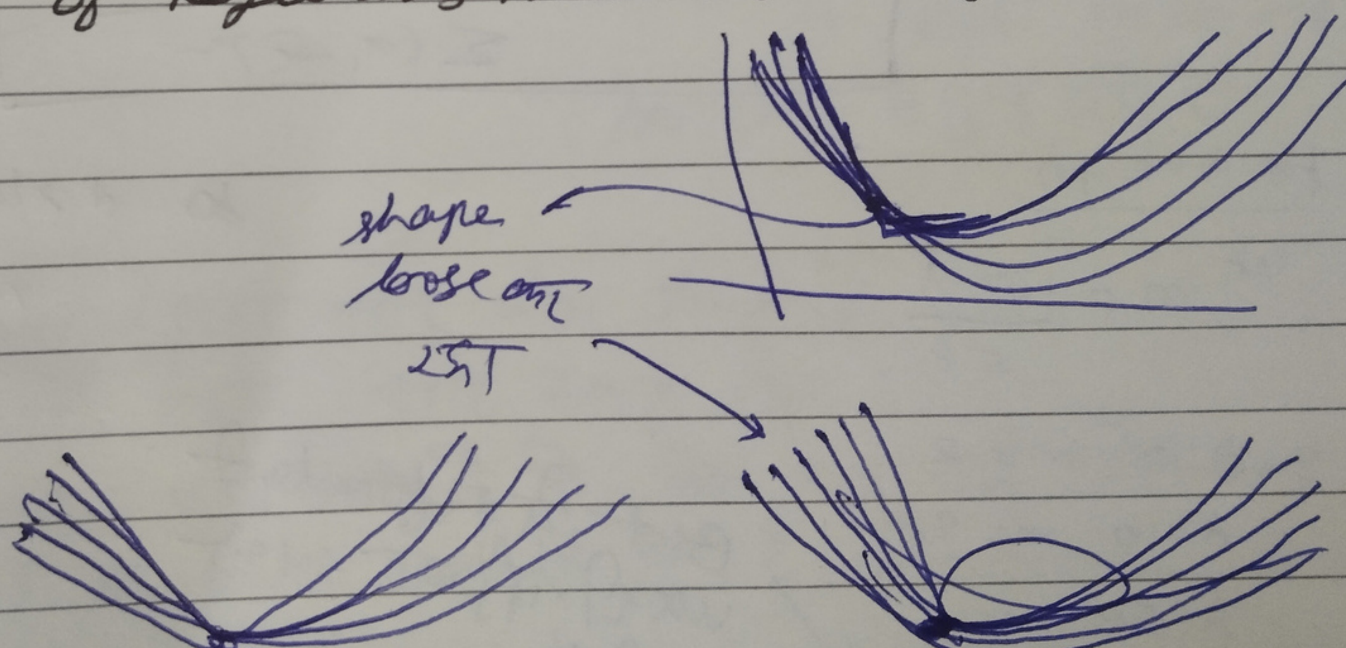
2. Higher Coeffs are affected more बड़े coeff पर ज्यादा effect

Lasso Regression (L1 Regularization)

3. Impact on Bias & Variance



4. Effect of Regularization on Loss f^n



Why Lasso Regression creates Sparsity ?

Q* Why Lasso Regression creates sparsity ? जितना $\lambda \uparrow$ $w \to 0$ & Why Ridge NOT able to do ? जाएगा $w = 0$

for $\lambda \uparrow$ (high) all $w = 0$

Let consider case of 1-feature

$y = mx + b$

for Simple LR for Ridge

$$m = \frac{\sum (y_i - \bar{y})(x_i - \bar{x})}{\sum (x_i - \bar{x})^2} \qquad m = \frac{\sum (y_i - \bar{y})(x_i - \bar{x})}{\sum (x_i - \bar{x})^2 + \lambda}$$

for Lasso

$$L = \sum (y_i - \hat{y}_i)^2 + \lambda |m|$$

$m > 0$

$$L' = \sum (y_i - mx_i - \bar{y} + m\bar{x})^2 + \lambda |m|$$

$$\frac{dL}{dm} = 2 \sum (y_i - mx_i - \bar{y} + m\bar{x}) \cdot (-x_i + \bar{x}) + \lambda = 0$$

Why Lasso Regression creates Sparsity ?

$$m = \frac{\sum (y_i - \bar{y})(x_i - \bar{x}) - \lambda}{\sum (x_i - \bar{x})^2}$$ when $m > 0$

$$m = \frac{\sum (y_i - \bar{y})(x_i - \bar{x}) + \lambda}{\sum (x_i - \bar{x})^2}$$ when $m < 0$

for +ve m

let $m = \frac{100 - \lambda}{50}$

$\lambda = 0 \quad m = 2$

$\lambda = 10 \quad m = 1.5$

$\lambda = 50 \quad m = 1$

$\lambda = 100 \quad m = 0$

$\lambda > 100 \quad m < 0$

So $\lambda > 100$

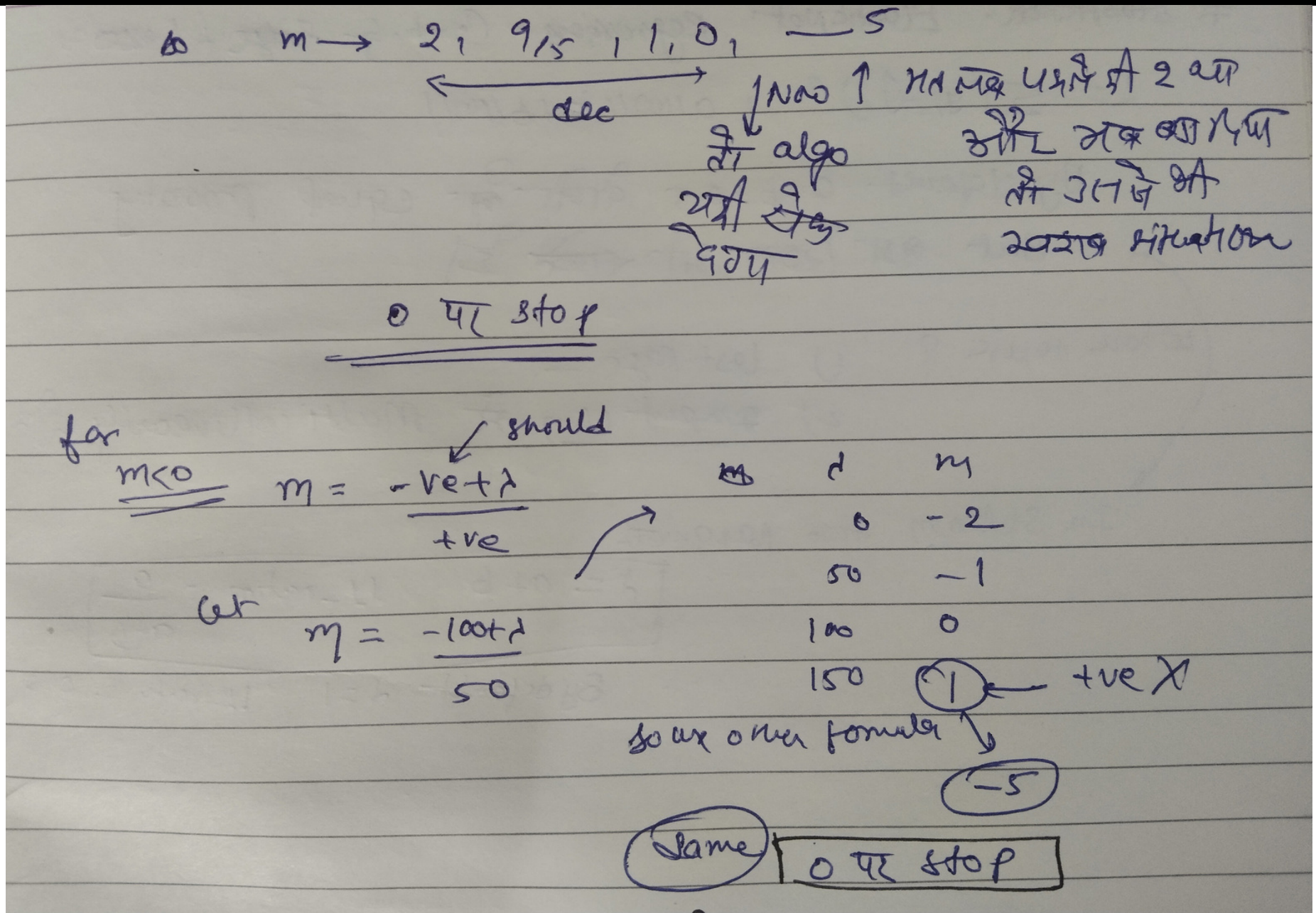
let $\lambda = 150$

$m = \frac{100 + \lambda}{50}$

$m = 5$

But ये तो formula की use की गई value है $m > 0$

Why Lasso Regression creates Sparsity ?



Why not Ridge Regression?

Ridge Reg में sparsity क्यों नहीं आता |

$$m = \frac{\sum (y_i - \bar{y})(x_i - \bar{x})}{\sum (x_i - \bar{x})^2 + \lambda}$$

λ को कितना भी बड़ा बढ़ा दो ~~best~~ m=0 नहीं होता

so close to पहुंच पाते हैं

Ridge ← use जब सारे col Imp हों

Lasso ← use जब सारे Imp नहीं हैं

But लेकिन बहुत बड़ा data है तो पता नहीं कौन Imp है कौन नहीं | ? so which should use?

→ that is Elastic Net Reg

ElasticNet Regression

~~Elasticnet~~ - ElasticNet Regression Comb. of Ridge + Lasso

$$L = \sum (y_i - \hat{y}_i)^2 + a\|w\|^2 + b\|w\|$$

By default $a = b = 0.5$ दोनों को equal priority.
But अलग रख भी सकते है।

where to use ? 1) lost Rage —
2) Input clm में Multicollinearity हो तो

In Sklearn two parameter

$$\lambda = a + b \qquad l1\_ratio = \frac{a}{a+b}$$

By default $\lambda = 1$, l1_ratio $= 0.5$

Why we need Elastic Reg?

Lasso regression it tries to shrink the coefficients to the absolute zero and if not possible to shrink to the absolute zero, then it eliminates the coefficient from the models. The ridge regression does not eliminate the coefficients from the model, which means it does not differentiate between important and less important predictive variables in the model and includes all of them by providing l2 penalty. It tries to shrink the unbiased coefficient by putting them with their squared magnitude into the model.

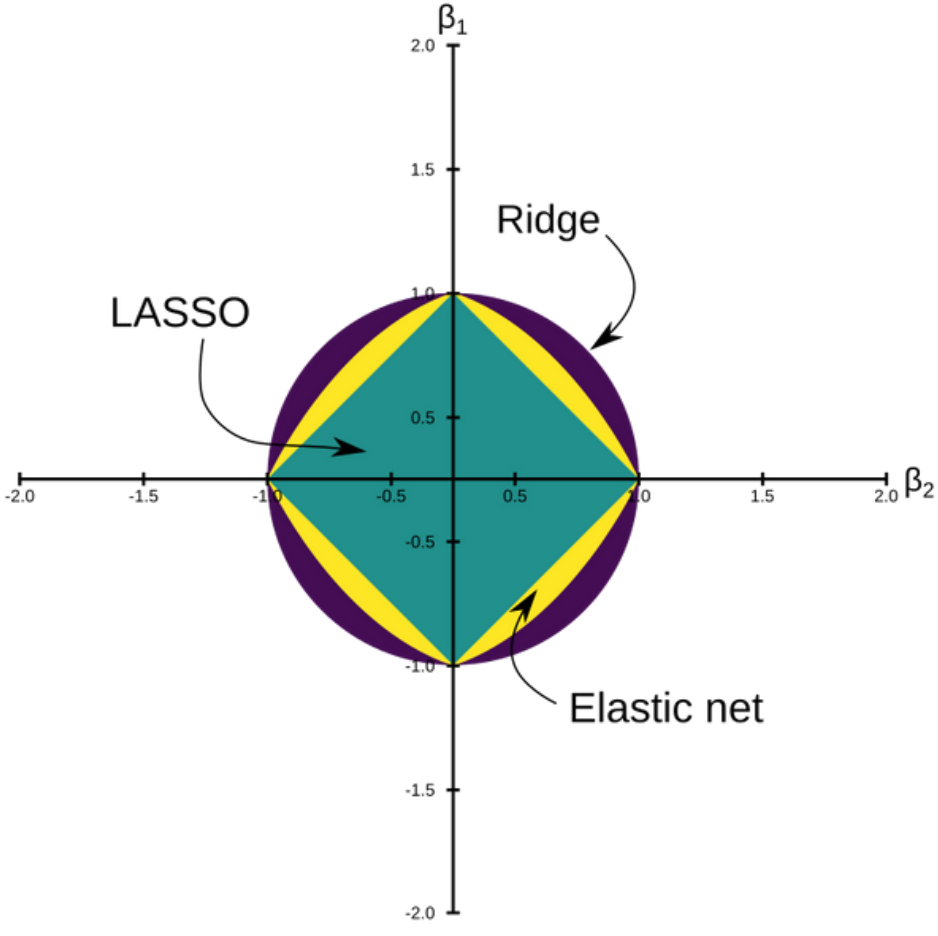
Why we need Elastic Reg?

But there are certain limitations of these models- ridge regression decreases the complexity of the model in performance but does not eliminate the unbiased variables hence we can increase the model's accuracy in a large dataset to a point. The new unbiased variable generated model can stop performing well. The lasso regression model picks the points according to the number of observations, not the predictor presented in the data. This kind of limitation can be handled and removed by the elastic net regression model where it includes both kinds of (l1 and l2) penalties in the model. (MULTICOLINIRITY) incapability of lasso is choosing the number of predictors.

How Elastic Reg Solve this Problem?

Elastic Net is a regularized regression model that combines l1 and l2 penalties, i.e., lasso and ridge regression. We have discussed the limitations of lasso regression, where we found the incapability of lasso is choosing the number of predictors. The elastic net includes the penalty of lasso regression, and when used in isolation, it becomes the ridge regression. In the procedure of regularization with an elastic net, first, we find the coefficient of ridge regression. After this, we perform a lasso algorithm on the ridge regression coefficient to shrink the coefficient.

How Elastic Reg Solve this Problem?



Here we can see that after performing the ridge regression, the lasso regression takes part in the procedure that considers all the variables from the dataset.

Mathematically we can represent the elastic net as follows.

$$\hat{\beta} \equiv \underset{\beta}{\operatorname{argmin}}(\|y - X\beta\|^2 + \lambda_2 \|\beta\|^2 + \lambda_1 \|\beta\|_1).$$