Regularization

The hidden truth (Trailer)

1000 std

x_1 Cgpa	x_2 iq	lpa
8	80	8
9	90	9

→ model → Predict → Cgpa | iq → lpa

X Y → $Y = f(x)$ → Some mathematical condition

x_1 → y_1

Cgpa | iq → lpa → for all students

Sample → pop predict

Regularization

$Y = f(x) + \text{error}$

$f(x)$ → ml model

error → Irreducible error

$Y = f(x)$ → pop

$\hat{Y} = f'(x)$

$f(x) - f(x)$ → Reducible error

Bias Variance Trade off

reducible error

Reducible error = Bias + Var

Bias Variance Tradeoff

Generally, Sample data → ML Model → Predict for population

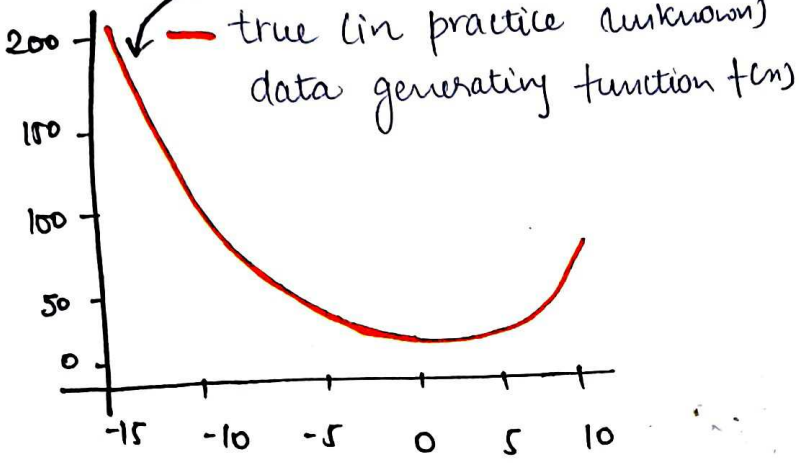
but we do reverse

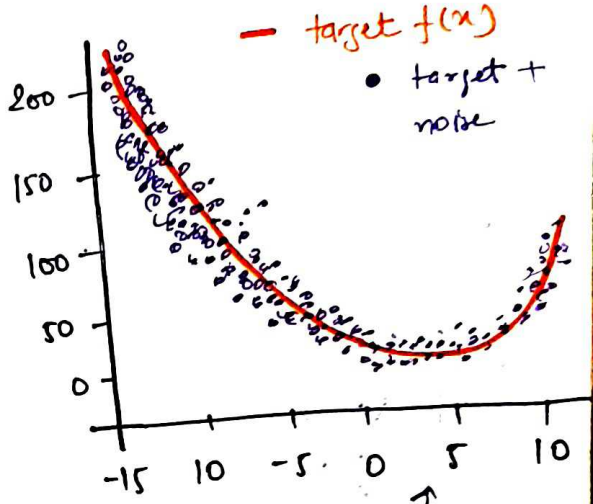
pop → Sample

$Y = f(x) = x^2$ $[-15, 10]$ ← range

$Y = x^2$

* In real world, we can't find this because we'll have sample data.



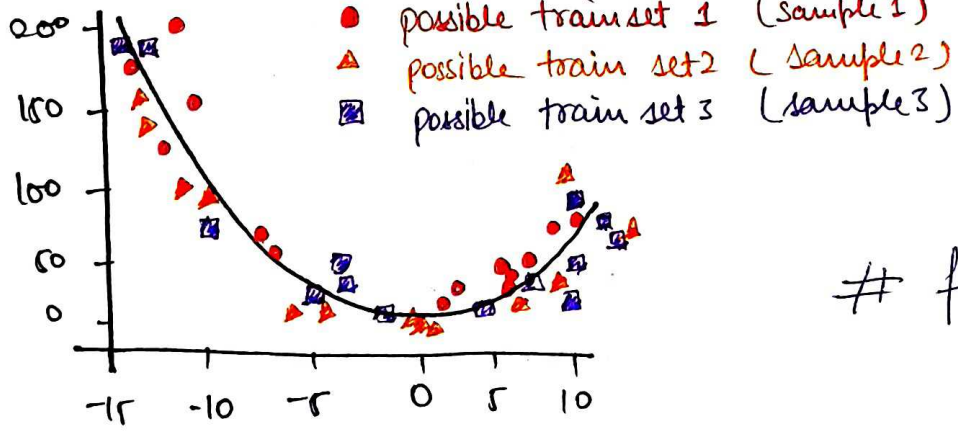


$Y = x^2 + \text{error}$

$[-15, 10]$

↳ add some error into pop (1000) points

Draw 3 random samples data

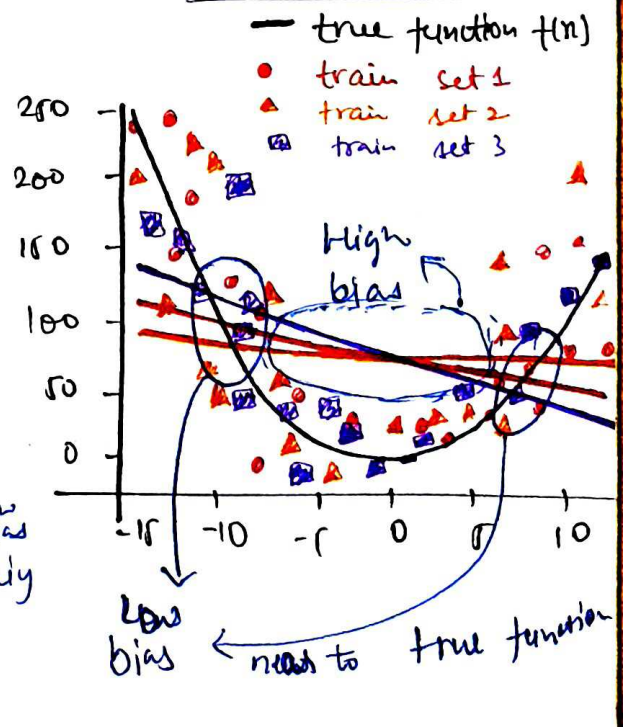


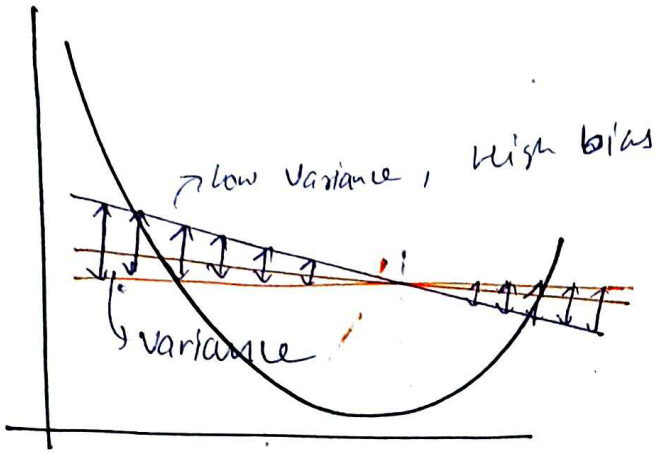
* all three samples try to predict like true function.

Bias → The inability of a ML model to fit the training data (i) high bias (ii) low bias

Variance → ML model predict when training data is change.

fitting Linear Regression

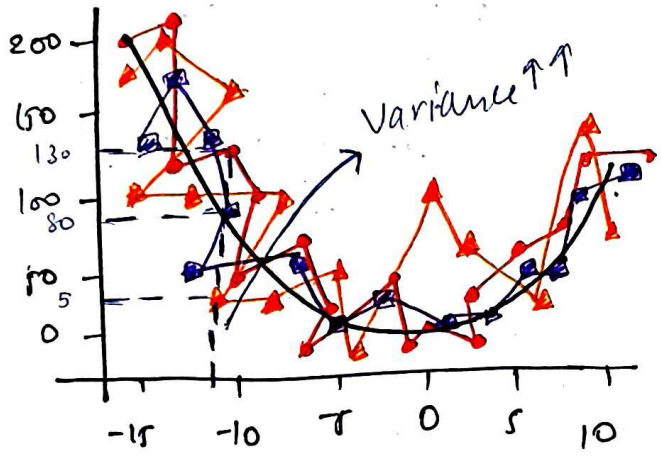




(1, 2, or 3 +)

Polynomial High degree

* High bias ✗
* low bias ✓
* High Varriance ✓



High Varriance is closely related overfitting

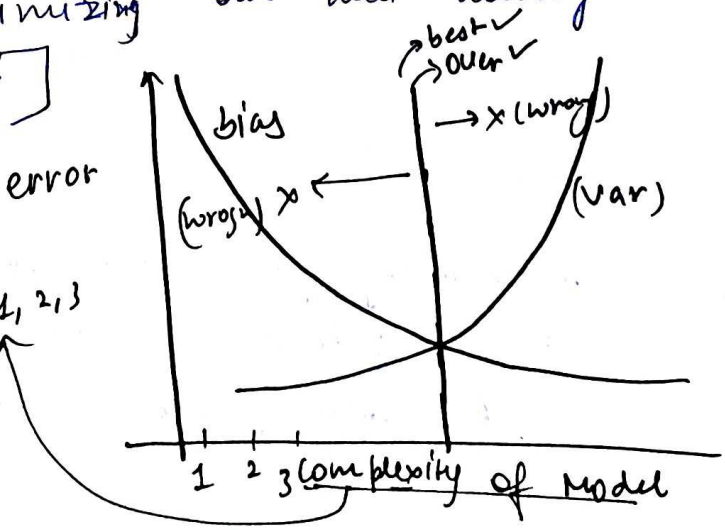
High bias is closely related to underfitting

* Idly output what we want is * low bias → good result in trainly data
* low varian → change data but result is same

But not possible

[The "trade-off" in bias-variance trade-off refers to the fact that minimizing bias will usually increase and vice versa.]

{ bias
variance }



Polynomial degree → 1, 2, 3

Some Question

1. How would you define bias and variance mathematically?
2. How is bias and variance related to overfitting and underfitting mathematically?
3. Why is there a tradeoff between bias and variance mathematically?

Expected Value and Variance

Expected value represents the average outcome of a random variable over a large number of trials or experiments.

In a simple sense, the expected value of a random variable is the long term average value of repetitions of the experiments it represents. For example, the expected value of rolling a six-side die is 3.5 because over many rolls, we would expect to average about 3.5.

A die roll 100k times

X = rolling die

$$E[X] = X_1 P(X_1) + X_2 P(X_2) + X_3 P(X_3) + X_4 P(X_4) + X_5 P(X_5) + X_6 P(X_6)$$

$X \rightarrow X_1 \rightarrow 1$
$\rightarrow X_2 \rightarrow 2$
$\rightarrow X_3 \rightarrow 3$
$X_4 \rightarrow 4$
X_5
X_6
$\rightarrow 5$

$E[X] = X_1 P(X_1) + X_2 P(X_2) + \ldots + X_n P(X_n)$

$= 1 \times \frac{1}{6} + 2 \times \frac{1}{6} + 3 \times \frac{1}{6} + 4 \times \frac{1}{6} + 5 \times \frac{1}{6} + \frac{6 \times 1}{6}$

$= \frac{1+2+3+4+5+6}{6} = \frac{21}{6} = 3.5$

↑ overall avg. mean

example

5	3	4	5	3	5

→ mean

↳ $\frac{5+3+4+5+3+5}{6} = \frac{25}{6} = 4.1$

* another method

$\frac{3(5) + 2(3) + 1(4)}{6}$

* we can also write like this

$\frac{3}{6}(5) + \frac{2}{6}(3) + \frac{1}{6}(4)$

~~mean~~ → expected value

Obsolete Random Variable → (X)

$E[X] = \sum_{i=1}^{n} X_i P(X_i)$

Continous Random Variable

$E[X] = \int X_i f(X_i) \, dx$

expected value $\xrightarrow{\text{estimate}}$ pop mean

Var(X) → var of pop

$$\text{Var}(X) = E[X^2] - (E[X])^2$$

Derivation

$$\text{Var} = \frac{\sum (x_i - \bar{x})^2}{n}$$ ← sample mean

$$E[(x - E[X])^2] = \text{Var}(x)$$

↳ pop mean

$(x-(E[x])^2)$ → $\sum (x_i - \bar{x})^2$ → n → $E[x]$

$$= E[x^2 + (E[X])^2 - 2X E[X]]$$

Random variable

$$= E[x^2] + E[(E[X])^2] - E[2X E[X]]$$

constant ↑ ; constant mean

$$= E[x^2] + E[(E[X])^2] - \underline{E[2]}\, E[X]\, \underline{E[E[X]]}$$

constant ↓ ; ↳ mean constant

$$= E[x^2] + E[(E[X])^2] - 2E[X]E[X]$$

$$= E[x^2] + E[(E[X])^2] - 2(E[X])^2$$

↳ constant

$$= E[x^2] + (E[X])^2 - 2(E[X])^2$$

$$= E[x^2] - E[X]^2$$

$$\text{Var}(X) = E[x^2] - E[x]^2$$ hence proved

↳ True for both discrete Random Variable and continous random variable.

$\therefore E[X+Y] = E[X] + E[Y]$

$\therefore E[XY] = E[X]E[Y]$ given X and Y independent

$\therefore E[\text{constant}] = \text{constant}$

* Another formula is

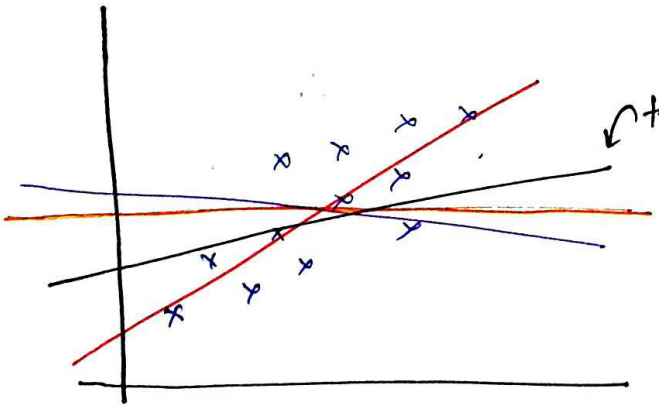
$$Var(X) = E[(X - E[X])^2]$$

we can use both formula.

What exactly are Bias and Variance Mathematically

Bias

In the context of Machine Learning and statistics, bias refers to the systematic error that a model introduce because it cannot capture the true relationship in data. It represents the difference between the expected prediction of our model and the correct value which we are trying to predict. More bias leads to underfitting, where the model does not fit the training data well.



true function $f(x)$

$f'(x)$ ← model apply

$E[f'(x)]$

↳ Expected Value

$$Bias = E[f'(x)] - f(x)$$

* If I am using Linear Regression Model, then we can calculate coefficient and intercept mean and compare true coefficient and intercept.

$E[f'(x)]$ —Linear Regression→ $E[m]$

↳ generally formula ↳ coefficient

$f'(x) - f(x) = 0 \rightarrow$ bias

* If bias is zero then this is called unbiased predictor

bias = 0

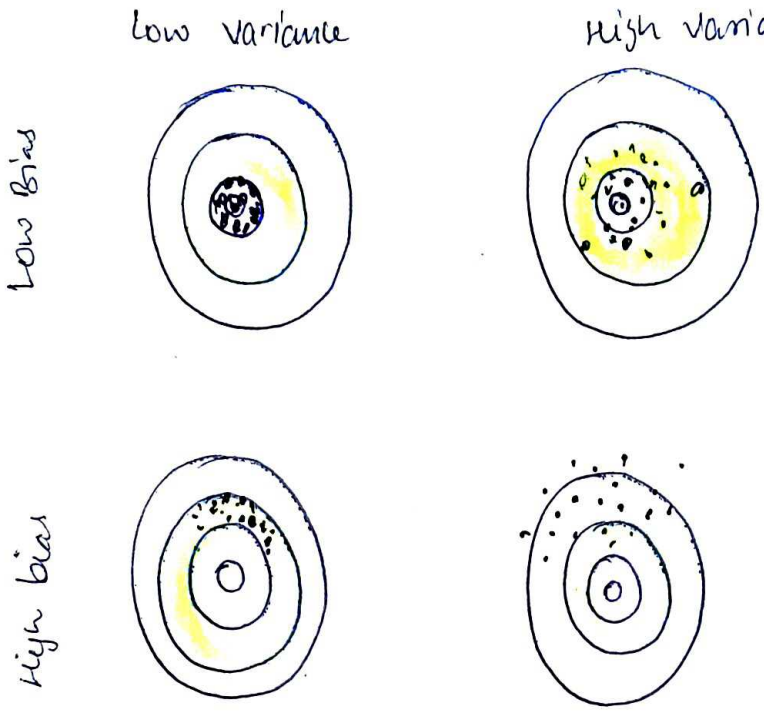
↳ unbiased predictor

↳ which means close to true function (pop function)

Variance

In the content of machine learning and statistics, variance refers to the amount by which the prediction of our model will change if we used a different training data set. In other words, it measure how much the predictions for a given point vary between different realizations of the models.

$$Var(f'(x)) = E[(f'(x) - E[f'(x)])^2]$$



Bias Variance Decomposition

Bias-variance decomposition is a way of analysis a learning algorithm's expected generalization error with respect to a particular problem by expressing it as the sum of three very different quantities: bias, variance and irreducible error.

1. Bias: This is the error from erroneous assumptions in the learning algorithms. High bias can cause an algorithm to miss the relevant relations between features and target outputs (underfitting)

2. **Variance**: This is the error from sensitivity to small fluctuation in the training set. High variance can cause an algorithm to model the random noise in the training data, rather than the intended outputs (overfitting).

3. **Irreducible Error**: This is the noise term. This part of the error is due to the inherent noise in the problem itself, and can't be reduced by any model.

$$\text{Loss} = \underbrace{(\text{bias})^2 + \text{Variance}}_{\text{reducible error}} + \underbrace{\text{var}(\epsilon)}_{\text{irreducible error}}$$

ϵ → epsilon

* irreducible error
  - mean = 0
  - var = σ (constant, assume)

Derivation

$$mse = \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{n} = E[(y - \hat{y})^2]$$

$$y = f(x) + \epsilon$$ (just change sign) $$y = \theta + \epsilon$$

we can also write

$$\hat{y} = f'(x) = \hat{\theta}$$

$$mse = E[(\theta + \epsilon - \hat{\theta})^2]$$

$$= E[(\underbrace{\theta - \hat{\theta}}_{a} + \underbrace{\varepsilon}_{b})^2] \quad \because$$

$$= E[(\theta - \hat{\theta})^2 + \varepsilon^2 + 2\varepsilon(\theta - \hat{\theta})] \quad \because (a+b)^2 = a^2 + b^2 + 2ab$$

$$= E[(\theta - \hat{\theta})]^2 + E[\varepsilon^2] + E[2\varepsilon(\theta - \hat{\theta})] \quad \because E[x+y] = E[x] + E[y]$$

$$= E[(\theta - \hat{\theta})^2] + E[\varepsilon^2] + E[\varepsilon]\,E[\theta - \hat{\theta}] \quad \because E[xy] = E[x]\,E[y]$$

$$= E[(\theta - \hat{\theta})^2] + E[\varepsilon^2] + 0 \quad \because E[\varepsilon] = 0 \text{ because we assume irreducible error mean is 0.}$$

$$\because \text{Var}(\varepsilon) = \sigma^2 = E[(\varepsilon - \underbrace{E[\varepsilon]}_{0})^2]$$

$$\text{Var}(\varepsilon) = E[(\varepsilon - 0)^2] = E[\varepsilon^2]$$

$$\text{mse} = \underbrace{E[(\theta - \hat{\theta})^2]}_{} + \text{Var}(\varepsilon)$$

$$E[(\theta - \hat{\theta})^2] = E[(\underbrace{\theta - E[\hat{\theta}]}_{a} + \underbrace{E[\hat{\theta}] - \hat{\theta}}_{b})^2] \quad \because \text{add and subtract with } E[\hat{\theta}]$$

$$E[(\theta - E[\hat{\theta}])^2 + (E[\hat{\theta}] - \hat{\theta})^2 + 2(\theta - E[\hat{\theta}])(E[\hat{\theta}] - \hat{\theta})]$$

$$\because (a+b)^2 = a^2 + b^2 + 2ab$$

$$= E[(\theta - E[\hat{\theta}])^2] + E[(E[\hat{\theta}] - \hat{\theta})^2] + E[2(\theta - E[\hat{\theta}])(E[\hat{\theta}] - \hat{\theta})]$$

$\because E[X+Y] = E[X] + E[Y]$

$$\therefore E[2(\theta - E[\hat{\theta}])(E[\hat{\theta}] - \hat{\theta})]$$

$$E[2] \quad E[(\theta - E[\hat{\theta}])] \quad E[(E[\hat{\theta}] - \hat{\theta})]$$

$$2 \quad (\theta - E[\hat{\theta}]) \{E[E[\hat{\theta}]] - E[\hat{\theta}]\}$$

$$2(\theta - E[\hat{\theta}]) \quad E[\hat{\theta}] - E[\hat{\theta}] = 0$$

$$0$$

$$= E[(\theta - E[\hat{\theta}])^2] + E[(E[\hat{\theta}] - \hat{\theta})^2] + \text{Var}(\varepsilon)$$

$E[(E[\hat{\theta}] - \hat{\theta})^2]$ ↓ Variance

$(\theta - E[\hat{\theta}])^2$

$(\theta - E[\hat{\theta}])^2$ ↓ $(\text{bias})^2$

$$\text{MSE} = (\text{bias})^2 + \text{Variance} + \text{irreducible error}$$

Hence proved

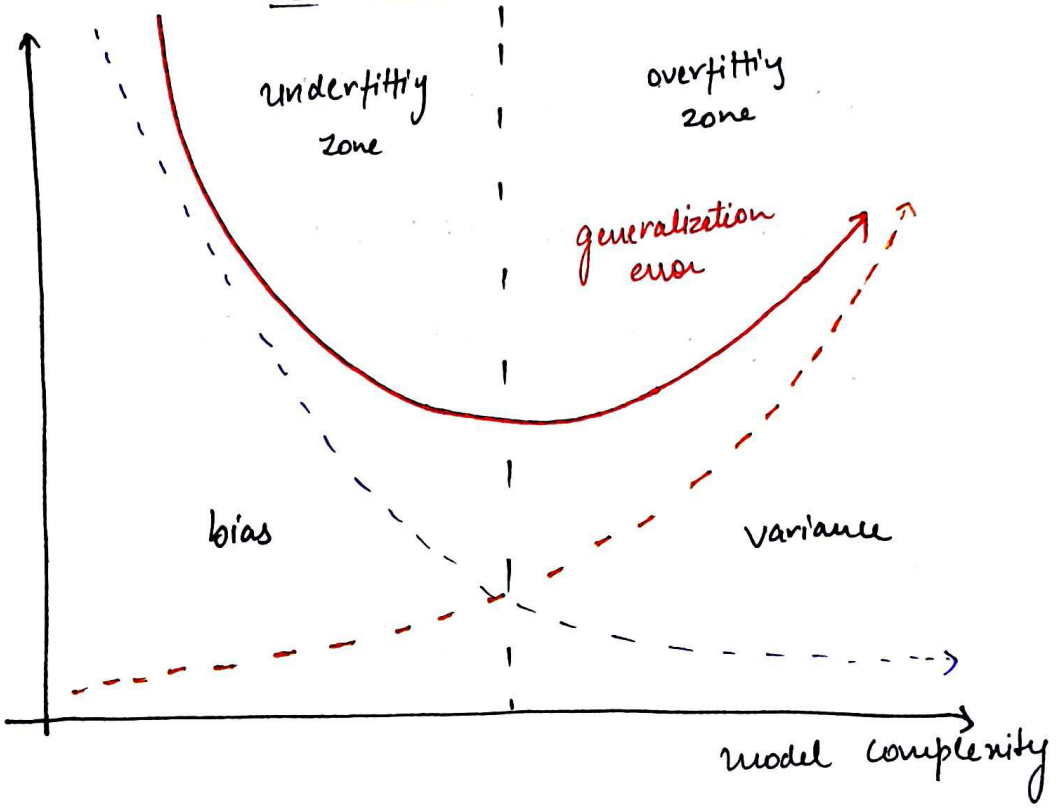
reducible error

$$\text{MSE} = \text{reducible error} + \text{irreducible error}$$

* for improving, bias we can use different ML algo.
or ~~reduce~~

* for reduce variance use Regularization.

the bias vs. variance trade-off



When to Use Regularization?

1. **Preventing Overfitting**: Regularization is most commonly used as a tool to prevent overfitting. If your model performs well on the training data but poorly on the validation or test data, it might be overfitting and regularization could help.

2. **High Dimensionality**: Regularization is particular useful when you have a high number of features compared to the number of data points. In such scenarios, models tend to overfit easily, and regularization can help by effectively reducing the complexity of the model.

3. **Multicollinearity**: When features are highly correlated (multicollinearity), it can destabilize your model and make the model's estimates sensitive to minor changes in the model. L2 regularization (Ridge regression) can help is such cases by distributing the coefficient estimate among correlated features.

4. **Feature Selection**: If you have a lot of feature and you have belive many of them might be irrelevant, L1 regularization (Lasso) can help. It tends to produce sparse solutions, driving the coefficients of irrelevant features to zero, thus performing feature selection.

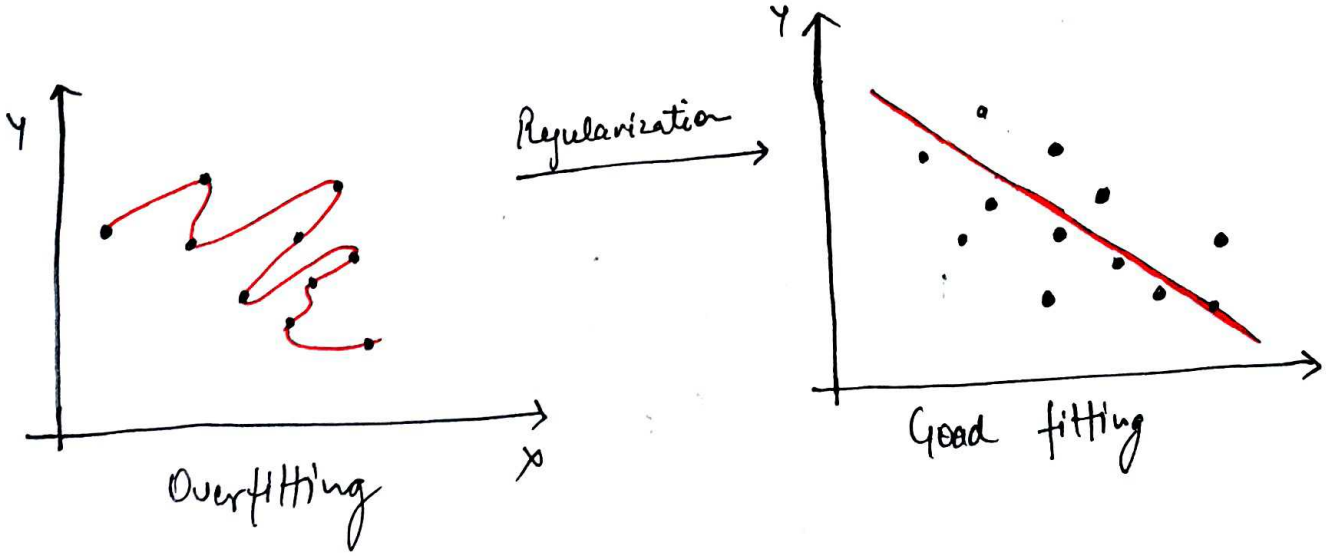
5. **Interpretability**: If model interpretability is important and you want a simpler model, regularization can help achieve this by constraining the model's complexity.

6. **Model Performance**: Even if you're not particularly worried about overfitting, regularization might still improve your model's out-of-sample prediction performance.

Regularization

- Regularization is one of the ways to improve our model to work on unseen data by ignoring the less important features.
- Regularization minimizes the validation loss and tries to improve the accuracy of the model
- It avoids overfitting by adding a penalty to the model with high variance, thereby shrinking the beta coefficients to zero.

Regularization refers to technique that are used to calibrate machine learning models in order to minimize the adjusted loss function and prevent overfitting or underfitting.



Ridge Regularition (L_2)

Also known as Rigde Regression, it modifies the over-fitted and under-fitted models by adding the penalty equivalent to the sum of the squares of the magnitude of coefficient.

Ridge Traits This means that the mathematical function representing our machine learning model is minimized and coefficients are calculated. The magnitude of coefficient is squared and added. Ridge Regression performs regularization by shrinking the coefficients present. The function depicted below shows the cost function of ridge regression

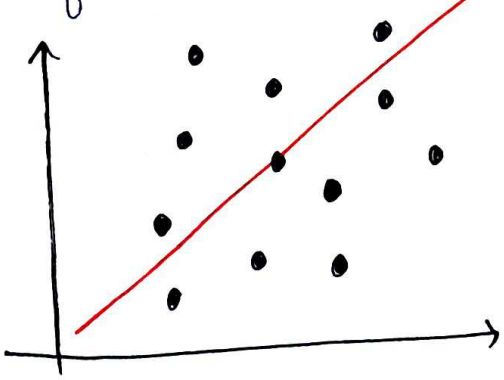
$$\text{cost function} = \text{Loss} + \lambda \times \sum \|w\|^2$$

Here

Loss = Sum of the squared residual

λ = Penalty for the errors

w = slope of the curve / line



In the cost function, the penalty term is represented by Lambda λ. By changing the values of the Penalty function, we are controlling the penalty term. The higher the penalty, it reduce the magnitude of coefficients. It shrink the parameter Therefore, it is used to prevent multicollinearity and it reduce the model complexity by coefficient shrinkage.

* Overfitting → Linear Regression

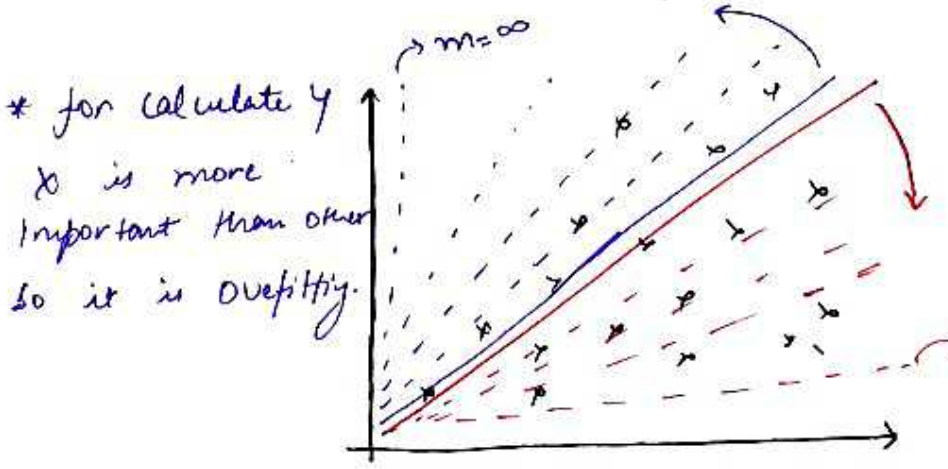
$y = mx + b$

↳ value of m is high in overfitting

* Underfitting → Linear Regression

$y = mx + b$

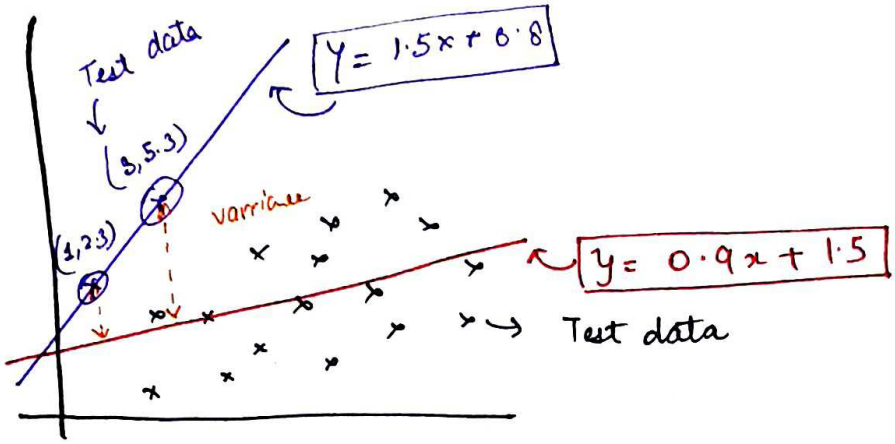
↳ value of m is low in underfitting



* for calculate Y
x is more important than other so it is overfitting.

$y = 0(x) + b$
$y = b$

underfitting
* x value less value than b to calculate Y



$$L = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda(m^2)$$

Train

* Train data → 2 points and set purple best fit line
* Test data → more than train data and predict red best fit line
* Varriance (↑↑) between Train best fit line and Test best fit line.

Loss L_n (Train) $\lambda = 1$

residual error = 0 (only two points)

$0 + 1(1.5)^2$

Loss = 2.25

Loss L_p (Test)

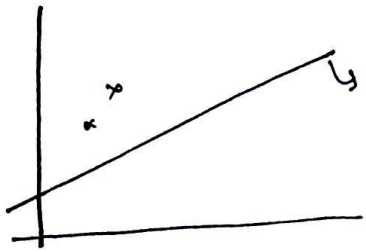
$\lambda = 1$

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

$(2.3 - (0.9(1) + 1.5))^2 + (5.3 - (0.9(3) + 1.5)^2 + (0.9)^2$

Loss = 2.03

* Choose low loss value $(y = 0.9x + 1.5)$ for Train and Test.



↳ $\boxed{y = 0.9x + 1.5}$

* Lil bit bias increasly

L2 → because $\lambda(m_1 + m_2 + m_3)$

↳ for multi feature

Derivation

$$L = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda m^2$$

→ $\frac{\partial L}{\partial b} = 0$ ↘ b

→ $\frac{\partial L}{\partial m} = 0$ ↗ m

$$L = \sum_{i=1}^{n} (y_i - mx_i - b)^2 + \lambda m^2$$

$b = \bar{y} + m\bar{x}$ where $\bar{y}$ → y-mean

$\bar{x}$ → x-mean

m → slope

$$L = \sum_{i=1}^{n} (y_i - mx_i - \bar{y} + m\bar{x})^2 + \lambda m^2$$

differentiate with respect to m

$$\frac{\partial L}{\partial m} = 2\sum_{i=1}^{n} (y_i - mx_i - \bar{y} - m\bar{x})(0 - x_i - 0 + \bar{x}) + 2\lambda m = 0$$

$$= -2\sum_{i=1}^{n}(y_i - \bar{y} - m x_i + m\bar{x})(x_i - \bar{x}) + 2\lambda m = 0$$

$$= \lambda m - \sum_{i=1}^{n}[(y_i - \bar{y}) - m(x_i - \bar{x})](x_i - \bar{x}) = 0$$

$$= \lambda m - \sum_{i=1}^{n}(y_i - \bar{y})(x_i - \bar{x}) - m(x_i - \bar{x})^2 = 0$$

$$= \lambda m - \sum_{i=1}^{n}(y_i - \bar{y})(x_i - \bar{x}) + m\sum_{i=1}^{n}(x_i - \bar{x})^2 = 0$$

$$= \lambda m + m\sum_{i=1}^{n}(x_i - \bar{x})^2 = \sum_{i=1}^{n}(y_i - \bar{y})(x_i - \bar{x})$$

$$= m\left(\sum_{i=1}^{n}(x_i - \bar{x})^2 + \lambda\right) = \sum_{i=1}^{n}(y_i - \bar{y})(x_i - \bar{x})$$

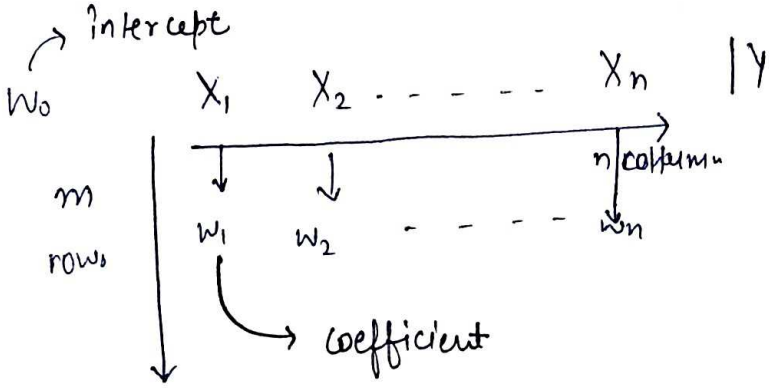
$$= m = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(x_i - \bar{x})}{\sum_{i=1}^{n}(x_i - \bar{x})^2 + \lambda}$$

$$m = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(x_i - \bar{x})}{\sum_{i=1}^{n}(x_i - \bar{x})^2 + \lambda}$$

hyperparameter (alpha)

code

Ridge Regression for nD data



total col = n+1

$$L = \sum_{i=1}^{m} (y_i - \hat{y}_i)^2$$

Metrics form

$$= (Xw - Y)^T (Xw - Y)$$

$$Y = \begin{bmatrix} \cdot \\ \cdot \\ \cdot \\ \cdot \end{bmatrix}_{m \text{ value}} \qquad w = \underset{(n\times 1)}{\begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_n \end{bmatrix}} \qquad X = \begin{bmatrix} 1 & x_{11} & x_{12} \cdots & x_{1n} \\ 1 & x_{21} & x_{22} \cdots & x_{2n} \\ 1 & \vdots & & \\ 1 & x_{m1} & x_{m2} \cdots & x_{mn} \end{bmatrix}$$

$$L = (Xw - Y)^T (Xw - Y) + \lambda \|w\|^2$$

$$\lambda \|w\|^2 \left\{ \lambda w_0^2 + \lambda w_1^2 + \lambda w_2^2 + \cdots + \lambda w_n^2 \right.$$

$$\lambda (w_0^2 + w_1^2 + \cdots w_n^2)$$

$$w^T w$$

$$\begin{bmatrix} w_0 & w_1 & w_2 \cdots & w_n \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_n \end{bmatrix}$$

$$\boxed{L = (Xw - Y)^T (Xw - Y) + \lambda w^T w}$$

$$L = \left[(Xw)^T - (Y)^T\right](Xw - Y) + \lambda w^T w \quad \because (a-b)^T = a^T - b^T$$

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

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

(same proved in Multiple Linear Reg.)

$$= w^T \overbrace{X^T X}^{\text{constant}} w - 2 w^T X^T Y + Y^T Y + \lambda w^T w$$

differentiate with $\frac{dL}{dw}$

$$\frac{dL}{dw} = 2X^T w - 2X^T Y + 0 + 2\lambda w = 0$$

Vector derivatives

- $X^T B \rightarrow B$
- $X^T b \rightarrow b$
- $X^T X \rightarrow 2X$
- $X^T B X \rightarrow 2BX$

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

$$(X^T X + \lambda I) w = X^T Y$$

↳ Identity matrix because we cannot extract w directly so we use identity matrix

$$w = X^T Y (X^T X + \lambda I)^{-1}$$

if 3 col. then (4x4) (feature)

nx1, nx1

code

Ridge Regression Using Gradient Descent

Vector form Loss

$$L = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

$$\hookrightarrow L = (XW - Y)^T (XW - Y) + \lambda \|W\|^2$$

$$\boxed{L = (XW - Y)^T (XW - Y) + \lambda W^T W}$$

$x_1 \quad x_2 \quad \cdots \quad x_n \mid Y$

m rows ↓

$$X = \begin{bmatrix} 1 & x_{11} & x_{12} & x_{13} & \cdots & x_{1n} \\ 1 & x_{21} & x_{22} & x_{23} & \cdots & x_{2n} \\ \vdots & \vdots & \vdots & & & \\ 1 & x_{m1} & x_{m2} & x_{m3} & \cdots & x_{mn} \end{bmatrix}$$

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \qquad W = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} \rightarrow (n+1)$$

$$w_0 = w_0 - \eta \frac{\partial L}{\partial w_0}, \quad w_1 = w_1 - \eta \frac{dL}{dw_1} \ldots\ldots$$

η → learning rate

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

$$w_{new} = w_{old} - \eta \left[\frac{\Delta L}{\Delta w}\right] \rightarrow \text{gradient} \begin{bmatrix} \frac{\partial L}{\partial w_0} \\ \frac{\partial L}{\partial w_1} \\ \vdots \\ \frac{\partial L}{\partial w_n} \end{bmatrix}$$

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

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

Multiply $\frac{1}{2}$ both equation

$$L = \frac{1}{2}(w^T X^T - Y^T)(Xw - Y) + \frac{1}{2}\lambda w^T w$$

$$L = \frac{1}{2}(w^T X^T Xw - w^T X^T Y - Y^T Xw + Y^T Y] + \frac{1}{2}\lambda w^T w$$

Same proceed in liner regression Gradient descent

$$= \frac{1}{2}\left[w^T X^T Xw - 2X^T w^T Y + Y^T Y\right] + \frac{1}{2}\lambda w^T w$$

$$\frac{dL}{dw} = \frac{1}{2}\left[2X^T Xw - 2X^T Y\right] + \frac{1}{2} 2\lambda w$$

$$= X^T Xw - X^T Y + \lambda w = \frac{dL}{dw} \quad \left(\frac{\Delta L}{\Delta w}\right)$$

$w_0\ w_1 \quad \ldots \quad w_n$

$w = [0, 1 \ldots\ldots 1)$ Startly

Epochs

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

$$\frac{dL}{dw} = X^T Xw - X^T y + \lambda w$$

5 key Understanding

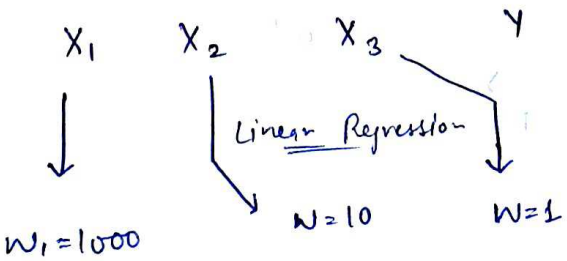
1. How the coefficient get affected ?

$\lambda \uparrow\uparrow$ $\quad \lambda \to 0 \to \infty$

$w_0\ w_1 \ldots w_n$ ↷↓ Shrinks

code

2. Higher Variance are impacted more



* w_1 play max role to find or predict Y
* If λ value increase then w_1 value highly decrease because w_1 have large value.

code

3. Bias - Variance trade off

Bias and Variance both depend on λ

$\lambda \downarrow$ = Bias ↓ overfit Variance ↑

$\lambda \uparrow$ = Bias ↑ underfit Variance ↓

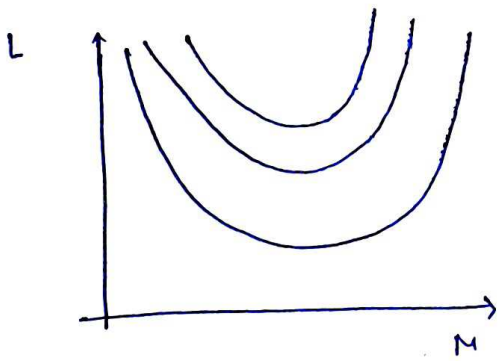
4. Impact on the Loss function

$\lambda \to$ $L = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \|w\|^2$

$x, y \to$ [m, b] → b is constant (assume $b = 6$)

$$L = \sum_{i=1}^{n} (y_i - m x_i)^2 + \lambda m^2$$

(y_i, x_i: constant; λ: constant; both in data)



code

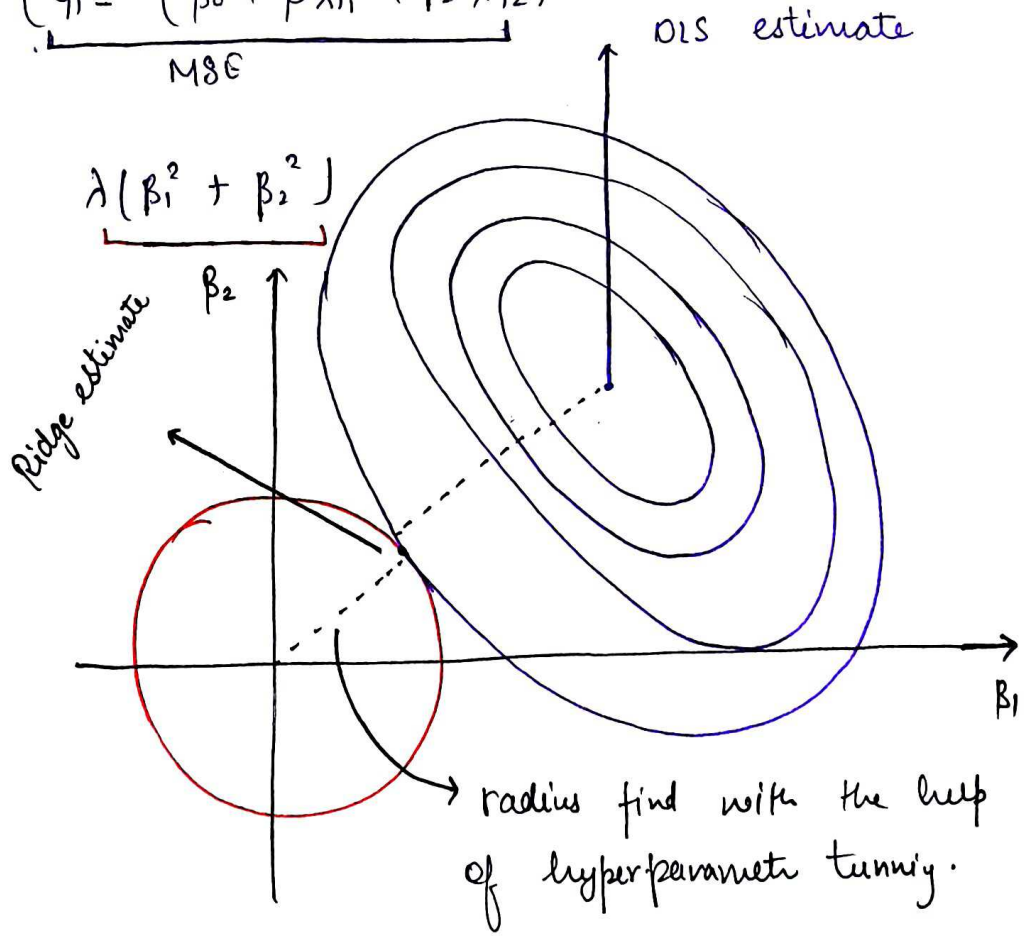
5. Why called Ridge

Hard Constraint Ridge Constrain → topic

2 coef β_1 β_2 β_0

$$l = \underbrace{MSE}_{} + \underbrace{\lambda \|w\|^2}_{}$$

$$\sum_{i=1}^{n} \underbrace{(y_i - (\beta_0 + \beta x_{1i} + \beta_2 x_{12})^2}_{MSE}$$



Practicle Tip

Use Ridge when there are 2 or more than 2 Inputs.

x_1 x_2 ..... ≥ 2

Lasso Regression (L1)

$L = MSE + \lambda \|W\|$ → $\lambda [|W_1| + |W_2| + |W_3| + \dots + |W_n|]$

$\lambda \downarrow\downarrow\downarrow$ → behaves like linear Regress → Overfitting

$\lambda \uparrow\uparrow\uparrow$ → underfitting

$X_1 \quad X_2 \quad X_3 \quad \dots \quad X_n \quad Y$

↳ Lasso → $\lambda \uparrow$
- ↳ ~~not~~ less important column → correspondig coef. = 0
- ↳ feature selection

* In Ridge Regression → know less important column but still remain some value.
* In Lasso → less important column = 0

Use Lasso for large no. of columns.

Lasso Sparsity
↳ $\lambda \uparrow$ $w \to 0$
↳ feature ↔ corresponding coefficient

In Ridge Regression $\lambda \uparrow$ → coefficient → not equal 0.
↳ $\lambda \uparrow$ → close to 0. but not equal to

In Lasso Regression $\lambda \uparrow$ → coefficient becoming 0.
↳ feature Selection

Mathematical

Simple Linear Regression

$X \mid Y \longrightarrow y = mx + b$

$$m = \frac{\sum_{i=1}^{n} (y_i - \bar{y})(x_i - \bar{x})}{\sum_{i=1}^{n} (x_i - x)^2}$$

$b = \bar{y} - m\bar{x}$

$\bar{y} \rightarrow$ mean (y)

$\bar{x} \rightarrow$ mean (x)

Ridge Regression

$$m = \frac{\sum_{i=1}^{n} (y_i - \bar{y})(x_i - \bar{x})}{\sum_{i=1}^{n} (x_i - x)^2 + \lambda}$$

$b = \bar{y} - m\bar{x}$

Lasso Regression

$b = \bar{y} - m\bar{x}$ $\quad m = ?$

$$L = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda |m|$$

Let considered $m > 0$

$$L = \sum_{i=1}^{n} (y_i - mx_i - \bar{y} + m\bar{x})^2 + 2\lambda |m|$$

→ just for easy mathematics

$$\frac{d}{dm} \sum_{i=1}^{n} (y_i - mx_i - \bar{y} + m\bar{x})^2 + 2\lambda m$$

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

$$-2\sum [(x_i - \bar{y}) - m(x_i - \bar{x})(x_i - \bar{x}) + 2\lambda = 0$$

$$-\sum [(y_i - \bar{y})(x_i - \bar{x}) - m(x_i - \bar{x})^2] + 2\lambda = 0 \quad \therefore \text{eq}^{n} \text{ divide by } 2.$$

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

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

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

Conditions

for $m > 0$

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

for $m = 0$

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

for $m < 0$

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

Discuss Sparsity

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

let

$$m = \frac{yx - \lambda}{x^2} \qquad \begin{cases} yx = 100 \\ x^2 = 50 \end{cases}$$

$\lambda = 0$

$$m = \frac{100}{50} = 2$$

$\lambda = 0$	$\lambda = 10$	$\lambda = 50$	$\lambda = 100$	$\lambda = 150$
$m = 2$	$m = 9/5$	$m = 1$	$m = 0$	$m = -1$

$$m = \frac{yx + \lambda}{x^2}$$ ← 3rd condition $m < 0$

$$m = \frac{100 + 150}{50} = 5$$

So, algorithm stop at 0 because after 0 value of m is increasing.

2, 9/5, 1, 0, 5

$\boxed{m < 0}$ $\quad \lambda > 0$

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

$$m = \frac{-100 + \lambda}{50} \qquad \lambda = 0$$

$$m = -2$$

$$m = \frac{-100 + \lambda}{50} \qquad \lambda = 150$$

$$m = -1$$

$m = \frac{-100 + \lambda}{50}$ $\lambda = 100$ $m = 1$

↳ m is positive

so, formula use $m = \frac{-100 - \lambda}{50} = -5$

Why sparcity not in Ridge Regression?

$m = \frac{\sum (y_i - \bar{y})(x_i - \bar{x})}{\sum (x_i - \bar{x})^2 + \lambda}$ → In numerator, if value is 0 still not be 0 after calculation. Because lambda add in denominator for 0. numerator should be 0. Thats why Ridge Regression value near or close to 0. not exact 0!

Ridge

Elastic Net Regression

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

$\lambda = a + b$

l1-ratio = $\frac{a}{a+b}$ or $\frac{a}{\lambda}$

λ, l1-ratio

hyperparameter

let $\lambda = 1$

l1_ratio = 0.9

↳ 90% ridge and 10% lasso

$a = 0.5$ $b = 0.5$

When use?

↳ Input cols → multicollinearity → used ElasticNet

Code