* Regularization Part 1 :-

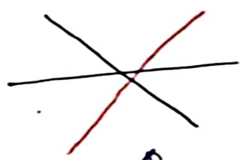
* Bias - Var Trade off *

Reducible error $(f(x) - f'(x))$

$f(x) \to y$, $f'(x) \to \hat{y}$

- Bias :- the inability of a ML model to fit the training data. (Deviation from actual relation)
- Variance :- how much the ML model changes upon changing the training data.

low variance



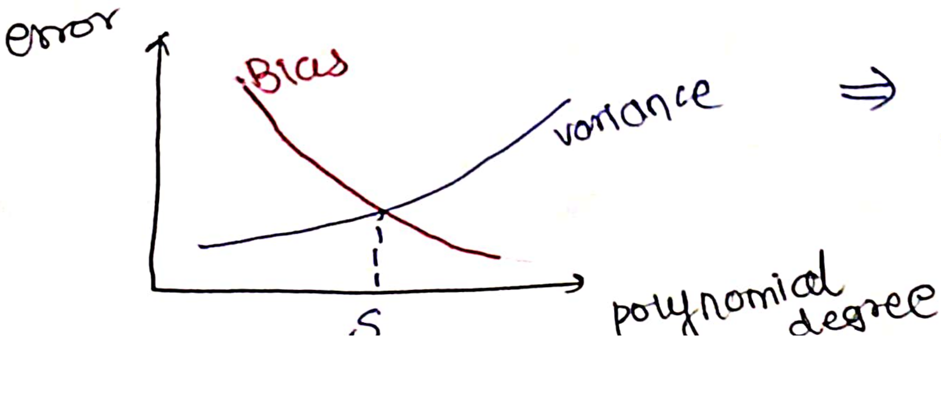
high variance.

↓

overfitting.

→ high bias ⇒ underfitting

$$\boxed{\text{Bias} \propto \frac{1}{\text{Variance}}}$$



⇒ target is to find this S.

- Expected value :- (pop. mean)

↳ represents the average outcome of a random variable over a large number of trials.

$$E[X] = x_1 P(x_1) + x_2 P(x_2) + \cdots + x_n P(x_n)$$

descrete rV

cont. rV $\longrightarrow$ $E[x] = \int x_i P(x_i)\, dx$

→ pop. of variance.

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

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

- what are bias & variance mathematically :

→ Bias :- It refers to systematic error that a model introduces bcoz it cannot capture the true relationship in the data.

It represents the diff. b/w expected prediction of our model and the correct value which we are trying to predict.

Actual relation $= f(x)$

model relation $= f'(x)$ → for a sample

Suppose, we take multiple samples and for each we have a specific $f'(x)$, so we take the expected value $E[f'(x)]$

Now,

$$\boxed{\text{Bias} = E[f'(x)] - f(x)}$$

→ This can never be calculated as we never know $f(x)$, it's just an interpretation.

→ variance :-

↳ refers to amount by which the prediction of our model will change if we used a diff. training data.

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

→ using Ridge:

```
from mlxtend.evaluate import
                    bias_variance_decomp
alphas = np.linspace(0, 30, 100)
loss = [ ]
bias = [ ]
variance = [ ]
for i in alphas:
    reg = Ridge (alpha=i)
    avg_expected_loss, avg_bias, avg_var =
    bias_variance_decomp (reg, X_train, y_train,
        X_test, y_test, loss = 'mse',
    random_seed = 123)
```

* Regularization Part 2 :-

- Bias Variance Decomposition :-

LOSS = bias + variance + irreducible error

→ Derivation :-

$$mse = \frac{\Sigma (y-\hat{y})^2}{n} \rightarrow \text{Expected value.}$$

loss

Assumption = $E(\epsilon) = 0$
$V[\epsilon] = \sigma^2$

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

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

$$mse = E[(y-\hat{y})^2]$$

considering $\epsilon = a$
$\theta - \hat{\theta} = b$

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

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

$$= E[(\theta - \hat{\theta})^2] + E[\epsilon^2] + 2\,\cancel{E[\epsilon]}^{\,0}\,E(\theta - \hat{\theta})$$

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

we know, $V[\epsilon] = \sigma^2 = E[(\epsilon - \cancel{E[\epsilon]}^{\,0})^2]$

$$= E[\epsilon^2]$$

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

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

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

$$= \underbrace{E[(\theta - E[\hat{\theta}])^2]}_{(\theta - E[\hat{\theta}])^2 \rightarrow Bias^2} + \underbrace{E[(E[\hat{\theta}] - \hat{\theta})^2]}_{\rightarrow Var(\hat{\theta})}$$

$$+ \underbrace{E[2]}_{2} \; \underbrace{E[\theta - E[\hat{\theta}]]}_{\theta - E[\hat{\theta}]} \; \underbrace{E[E[\hat{\theta}] - \hat{\theta}]}_{0}$$

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

$$\therefore \quad Loss = \underbrace{Bias^2 + variance}_{reducible} + \underbrace{var(\epsilon)}_{irreducible}$$

- Regularization → Reduce variance.
  ↓
  reduce overfitting.

→ complex models are good way to reduce bias.

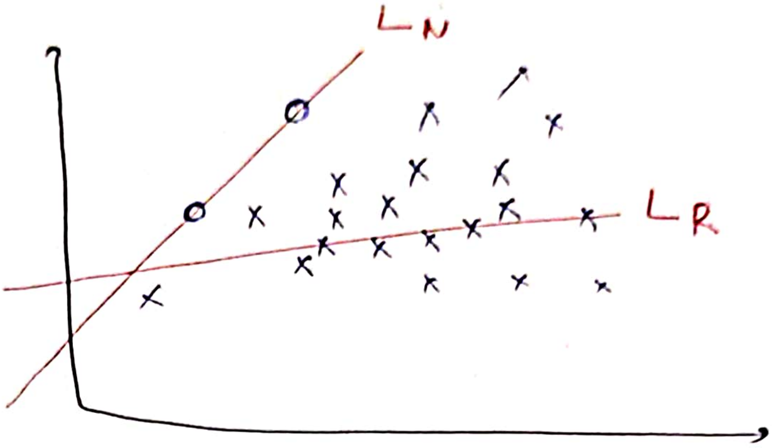
* Ridge Regression :-

↳ L2 Regularization.

→ Let's consider a case where training data has only 2 points.

o → training points

x → test points.



we can see, the best fit line according to the training data is not best for test.

∴ we need to convince the model to take LR as best fit.

Regularization term

$$\Rightarrow \boxed{L = \Sigma (y_i - \hat{y}_i)^2 + \lambda (m^2)}$$

Bias slightly increases.

L2

for multiple LR

$= \lambda (\Sigma m_i^2)$

```
from sklearn.linear_model import Ridge
R = Ridge (alpha = 0.0001)
R.fit (X_train, y_train)
y = R.predict (X_test)
```

λ

λ → less value ⇒ less reduction in slope

→ high value ⇒ high redn in slope

↳ may result in underfitting.

Similar to the formula of 'm' in linear regression,

m for Ridge regression,

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

Extra term

→ for multiple linear regression :

$$L = (XB - Y)^T (XB - Y) + \lambda B^T B$$

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

- for Ridge regression with gradient descent :

```
from sklearn.linear_model import SGDRegressor
reg = SGDRegressor(penalty = 'l2', max_iter = 50
      eta0 = 0.1 , learning_rate = 'constant',
            alpha = 0.001)
```

Now fit & predict

OR

use solver = 'sparse_cg' in Ridge.

Q.1 How the coefficients get affected?

λ — 0 ——— ∞

0 → Simple Linear Regression

∞ → coefficient near to 0.

Higher values are affected more, at $\alpha > 1000$, all values come near to 0.

→ for low λ → Bias is low, variance is high (over fit)

→ for high λ → Bias is high, variance is low (underfit)

Bias-variance trade off.

→ Bias-var ke graph ke intersection se thode pehle wale point ki alpha value leni hai.

* Lasso Regression :-

L1 Regularization

$$\boxed{L = MSE + \lambda \| m \|}$$

L1

In case of ridge regression, upon increasing the value of λ, corresponding coefficient goes near to zero, but in case of Lasso, the coefficient corresponding to variables which are are not imp. for prediction becomes zero. → feature selection.

```
from sklearn.linear-model import Lasso
reg = Lasso(alpha= )
reg.fit(X_train, y_train)
```

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

for $m > 0 \rightarrow -\lambda$

for $m < 0 \rightarrow +\lambda$

for $m = 0 \rightarrow$ no λ term

* ElasticNet Regression :-

↳ combination of L1 & L2 :

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

α

$\lambda = a + b$

$l_1\_ratio = \dfrac{a}{a+b}$ } hyperparameters.

→ If there is multicollinearity in input cols then we should use ElasticNet