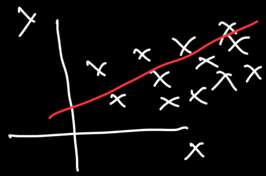
Linear Regression

3D – Plane

4D & more – hyperplane

- **Simple LR** — One input

x	y



Best fit line

$y = mx + b$

(m, b)

- **Multiple LR** — Multiple input

x_1	x_2	x_3	y

$y = mx_1 + nx_2 + ox_3 + b$

$$\boxed{y = \beta_0 + \beta_1 x_1 + \beta_2 x_2}$$

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots \beta_n x_n$$

$$\boxed{y = \beta_0 + \sum_{i=1}^{n} \beta_i x_i}$$

$$dPa = \beta_0 + \beta_1 \times cgPa + \beta_2 \times iq$$

Coefficient : $\beta_i \longrightarrow$ weights (how important feature is)

Intercept : $\beta_0 \rightarrow$ offset

- **Polynomial**

Mathematical Formulation

cgPa	iq	Gender	dPa
x_1	x_2	x_3	y

$\rightarrow$ Actual

Predicted $\longleftarrow \hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$

$$\begin{bmatrix} \hat{Y}_1 \\ \hat{Y}_2 \\ \vdots \\ \hat{Y}_{100} \end{bmatrix} = \begin{bmatrix} \beta_0 & \beta_1 x_{11} & \beta_2 x_{12} & \beta_3 x_{13} \\ \beta_0 & \beta_1 x_{21} & \beta_2 x_{22} & \beta_3 x_{23} \\ \vdots & \vdots & \vdots & \vdots \\ \beta_0 & \beta_1 x_{1001} & \beta_2 x_{1002} & \beta_3 x_{1003} \end{bmatrix}$$

Now for m columns and n rows
X

$$\hat{Y} = \begin{bmatrix} \hat{Y}_1 \\ \hat{Y}_2 \\ \vdots \\ \hat{Y}_n \end{bmatrix} = \begin{bmatrix} \beta_0 & \beta_1 X_{11} & \beta_2 X_{12} & \beta_3 X_{13} & \cdots & \beta_m X_{1m} \\ \beta_0 & \beta_1 X_{21} & \beta_2 X_{22} & \beta_3 X_{23} & \cdots & \beta_m X_{2m} \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ \beta_0 & \beta_1 X_{n1} & \beta_2 X_{n2} & \beta_3 X_{n3} & \cdots & \beta_m X_{nm} \end{bmatrix}$$

$$\hat{Y} = \begin{bmatrix} 1 & X_{11} & X_{12} & X_{13} & \cdots & X_{1m} \\ 1 & X_{21} & X_{22} & X_{23} & \cdots & X_{2m} \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 1 & X_{n1} & X_{n2} & X_{n3} & \cdots & X_{nm} \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_m \end{bmatrix}$$

$$\boxed{\underbrace{\hat{Y}}_{\text{Prediction matrix}} = \underbrace{X}_{\text{Input Matrix}} \; \underbrace{\beta}_{\text{Coefficient matrix}}} \quad \text{——— (i)}$$

$$\underbrace{Y}_{\text{Actual output matrix}} = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix}$$

$$e = Y - \hat{Y} = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix} - \begin{bmatrix} \hat{Y}_1 \\ \hat{Y}_2 \\ \vdots \\ \hat{Y}_n \end{bmatrix}$$

$$\underbrace{e}_{\text{Error matrix}} = \begin{bmatrix} Y_1 - \hat{Y}_1 \\ Y_2 - \hat{Y}_2 \\ \vdots \\ Y_n - \hat{Y}_n \end{bmatrix}$$

Single LR

$$E = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \longrightarrow E = e^T e$$

$$= \left[(y_1-\hat{y}_1)\ (y_2-\hat{y}_2)\ \dots\ (y_n-\hat{y}_n)\right] \begin{bmatrix} (y_1-\hat{y}_1) \\ (y_2-\hat{y}_2) \\ \vdots \\ (y_n-\hat{y}_n) \end{bmatrix}$$

$$= (y_1-\hat{y}_1)^2 + (y_2-\hat{y}_2)^2 + \dots (y_n-\hat{y}_n)^2$$

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

$$(A+B)^T = A^T + B^T$$

$$(A-B)^T = A^T - B^T$$

$$(AB)^T = B^T A^T$$

$$E = e^T e = (y - \hat{y})^T (y - \hat{y})$$

$$E = (y^T - \hat{y}^T)(y - \hat{y})$$

$$E = \left[y^T - (X\beta)^T\right](y - X\beta)$$

$$E = y^T y - \underline{y^T X \beta} - \underline{(X\beta)^T y} + (X\beta)^T X\beta$$

$$\boxed{E = y^T y - 2y^T X\beta + \beta^T X^T X \beta}$$

$y^T X \beta \qquad (X\beta)^T y$

$$A^T B = B^T A \quad \text{— (i)}$$

$$(A^T B)^T = B^T A \quad \text{— (ii)}$$

$$A^T B = (A^T B)^T \longrightarrow \text{let } A^T B = C$$

$$\Rightarrow C = y^T X \beta$$

then $\boxed{C = C^T}$

$$y^T X\beta = (y^T X \beta)^T = (X\beta)^T y \qquad \therefore y^T X\beta = (X\beta)^T y$$

Cost / Loss function

$$\boxed{E = Y^T Y - 2Y^T X\beta + \beta^T X^T X \beta}$$

$$\frac{dE}{d\beta} = \frac{d}{d\beta}\left[Y^T Y - 2Y^T X\beta + \beta^T X^T X\beta\right] = 0$$

$$\Rightarrow 0 - 2Y^T X + \frac{d}{d\beta}\left[\beta^T X^T X\beta\right] = 0$$

↳ Matrix differentiation

$$\Rightarrow -2Y^T X + 2X^T X\beta^T = 0$$

$$\Rightarrow \cancel{2}X^T X\beta^T = \cancel{2}Y^T X$$

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

$$\Rightarrow (\beta^T)^T = \left[Y^T X (X^T X)^{-1}\right]^T$$

$$\Rightarrow \beta = \left[(X^T X)^{-1}\right]^T (Y^T X)^T$$

$$\Rightarrow \beta = \left[(X^T X)^{-1}\right]^T X^T Y$$

[Transpose of inverse Square matrix: Shape is Same.]

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

$X \longrightarrow$ X_train

$Y \longrightarrow$ Y_train

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

$$\begin{bmatrix}\beta_0\\ \beta_1\\ \vdots\\ \beta_m\end{bmatrix}_{(m+1)\times 1}$$

$$[(m+1)\times(m+1)]\,[(m+1)\times n]\,[n\times 1]$$

$$[(m+1)\times n]\,[n\times 1]$$

$$(m+1)\times 1$$

⟹ Why Gradient descent

Reason is this $(X^TX)^{-1}$ inverse operation

$$\boxed{\beta = (X^TX)^{-1}X^TY}$$

Wikipedia : Computational Complexity of mathematical operations

Matrix algebra : Matrix inversion $\xrightarrow[\text{Gauss-Jordan elimination}]{\text{Algorithm}}$ Time Complexity $O(n^3)$

⟹ This takes too much time and makes our algorithm slow

⟹ Solution : Gradient Descent › Here we don't use formula, its a approximation technique where we slowly slowly go to right value.

SK Learn

→ OLS — LinearRegression class

→ Gradient Descent — SGDRegressor class