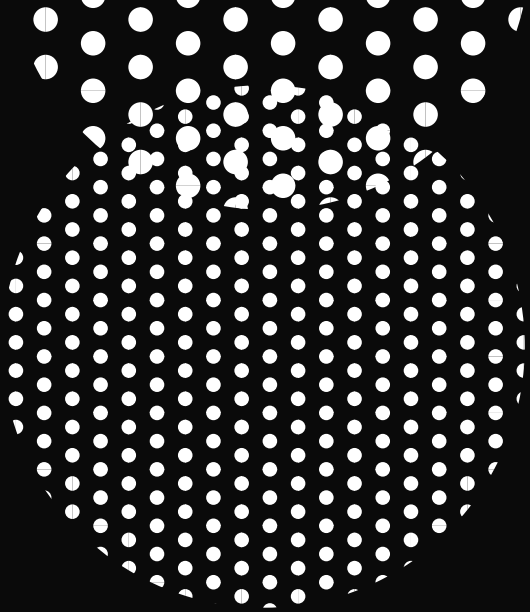


* Machine Learning Notes *

Aniruddha Rao

What is machine learning ?

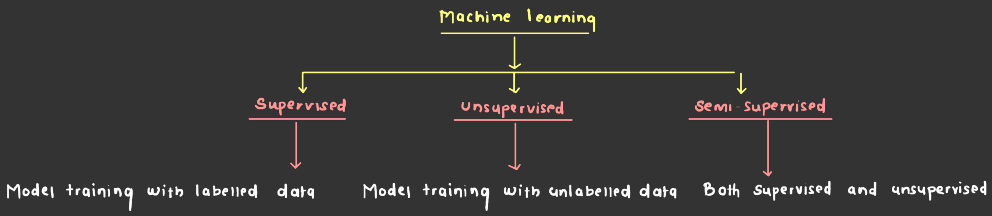
Machine learning is subset of AI.

Statistical model that enabled computer to learn like human.

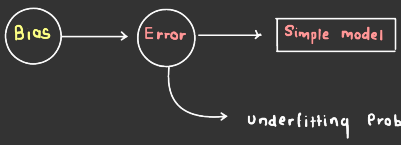
ML rely on data to learn

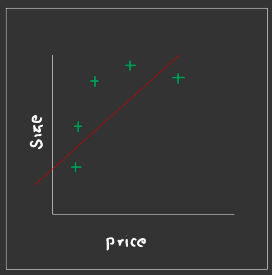
Improve thier performance on specific task without being explicitly programmed.

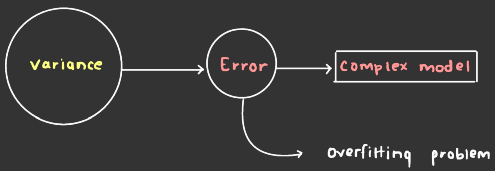
Types of Machine learning:

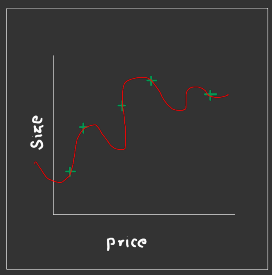


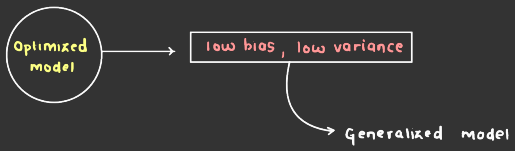
Bias and variance in machine learning











Simple Linear Regression

Simple linear regression statistical analysis that establish linear relationship between x and y.

x ⟶ Independent Variable ⟶ Input

y ⟶ dependent Variable ⟶ Output

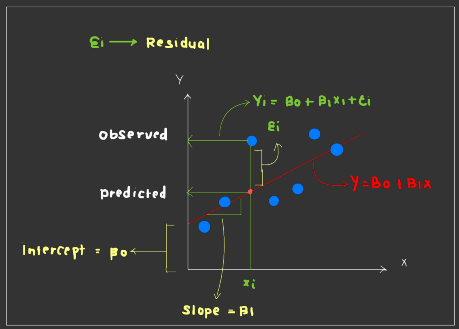
Aim: To find a best fit line using equation ⟶ $y_i = B_0 + B_1 x_i$

B_0 ⟶ Constant, intercept

B_1 ⟶ Slope

x_i ⟶ independent variable

y_i ⟶ dependent variable



The goal is to find best value for B_0, B_1

Random Error (Residual) ⟶ $E_i = y_{predicted} - y_i$

↳ Error between actual and predicted

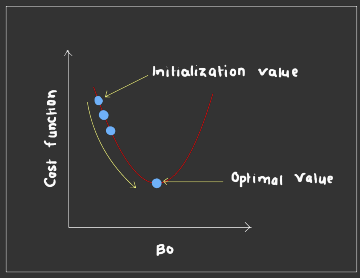
where $y_{predicted} = B_0 + B_1 x_i$

Cost function

$$MSE = \frac{1}{N} \sum_{i=1}^{n} (y_i - (B_1 x_i + B_0))^2$$ ⟶ Mean Squared Error

MSE is loss function quantify error than we will update B_0, B_1 parameters using convergence algorithm

Gradient descent ⟶ Optimization Algorithm

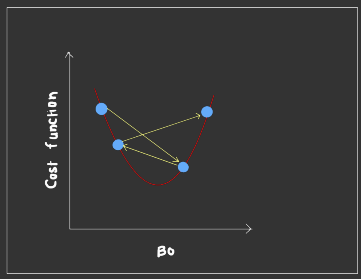


Optimizing ⟶ Cost function ⟶ Reducing cost function for all datapoints

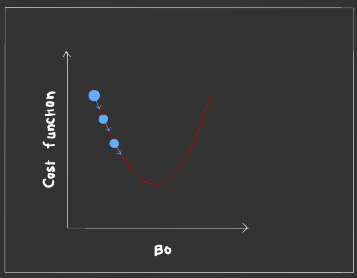
Updating Bo and B1 iteratively untill we get optimal solution

the number of steps you are taking to reach optimal value is known as learning rate

When you choose high α learning rate ⟶



When you choose low α learning rate ⟶



Best value for learning rate ⟶ 0.01

To update Bo and B1, we take gradients from cost function. To find gradients, we take partial derivatives for Bo, B1

$$J = \frac{1}{n} \sum_{i=1}^{n} (Bo + B1 \cdot xi - yi)^2$$

$$\frac{\partial J}{\partial Bo} = \frac{1}{n} \sum_{i=1}^{n} (Bo + B1 \cdot xi - yi)^2$$

$$\frac{\partial J}{\partial B1} = \frac{1}{n} \sum_{i=1}^{n} (Bo + B1 \cdot xi - yi)^2$$

$$\frac{\partial J}{\partial Bo} = \frac{2}{n} \sum_{i=1}^{n} (Bo + B1 \cdot xi - yi)$$

$$\frac{\partial J}{\partial B1} = \frac{2}{n} \sum_{i=1}^{n} (Bo + B1 \cdot xi - yi) \cdot xi$$

$$B_0 = B_0 - \alpha \cdot \frac{2}{n} \sum_{i=1}^{n} (Y_{pred} - y_i) \qquad B_1 = B_1 - \alpha \cdot \frac{2}{n} \sum_{i=1}^{n} (Y_{pred} - y_i) \cdot x_i$$

α ⟶ learning rate ⟶ Control Convergence Speed

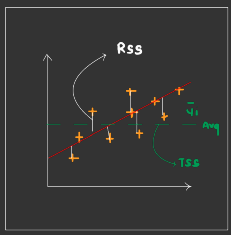
Evaluation metrics for linear regression

After training model ⟶ Evaluation of model ⟶ How accurate model is predicting result.

RMSE ⟶ Root mean squared error $= \sqrt{MSE}$

- lowering RMSE ⟶ Better result
- RMSE ⟶ 0 ⟶ A perfect model
- Sensitive to outliers
- it expressed same unit making it more interpretable

Coefficient of determination OR R-squared (R2)



- R2 always range between ⟶ 0 to 1
- The higher the value of R2 ⟶ better the model fit data

$$R^2 = 1 - \frac{RSS}{TSS}$$

RSS ⟶ Residual sum of squared

TSS ⟶ Total sum of Squared

$$RSS = \sum_{i=1}^{n} (Y_{actual} - Y_{predicted})^2 \rightarrow (y_i - \hat{y}_i)^2$$

$$TSS = \sum_{i=1}^{n} (Y_{actual} - \bar{Y}_{predicted})^2 \rightarrow (y_i - \hat{y}_i)^2$$

Problem with R2 metric

R^2 tends to increase ↑ as you add more independent variable. ⟶ lead overfitting

↳ To solve this we can adjusted R2

Adjusted R^2 metric

↳ Penalizing Overfitting

$$\text{Adjusted } R^2 = 1 - \left[\frac{(1-R^2)(n-1)}{(n-k-1)}\right]$$

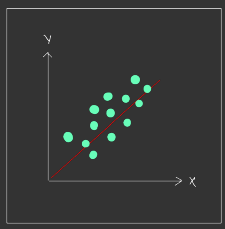
$n \longrightarrow$ no of datapoints

$K \longrightarrow$ no of independent variable

$R \longrightarrow R^2$ value

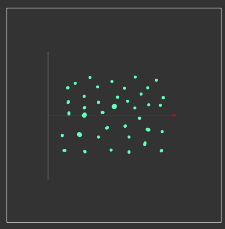
Assumption of linear regression

linear relationship →

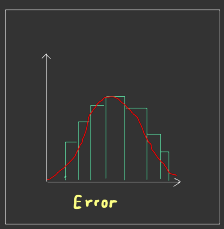


independent of residual →

↳ The error should not dependent on one another.

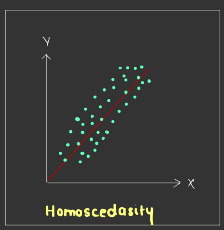


Normal distribution of residual →



The equal variance of residual →

↳ Homoscedasticity



Manual Calculation of Simple linear regression

dataset

X → Height	65	67	70	→ Inches
Y → Weight	127	160	175	→ Pounds

Calculate mean of x and y

$$\bar{x} = \frac{63 + 67 + 70}{3} = 66.67 \qquad \bar{y} = \frac{127 + 160 + 175}{3} = 154.00$$

Calculate the slope

$$\beta_1 = \frac{\sum((x_i - \bar{x})(y_i - \bar{y}))}{\sum (x_i - \bar{x})^2} \qquad \beta_1 = \frac{144.00}{22.33} = 6.45$$

Calculate the intercept

$$\beta_0 = \hat{y} - \beta_1 * \bar{x}$$

$$\beta_0 = 154 - 6.45 * 66.67 = 154 - 430.12 = -276.12$$

Now regression equation

$$\text{Weight} = -276.12 + 6.45 * \text{height}$$

→ We can make prediction using this equation

Calculate the Cost ⟶ Error

$$\text{Mse} = \sum \frac{1}{n}(y_i - \hat{y})^2$$

Error for 1st data points ⟶ 27759.78

Error for 2nd data points ⟶ 178.61

Error for 3rd data points ⟶ 4541.48

Update B_0 and B_1 using gradient descent

Start initial value ⟶ $B_0 = 0$, $B_1 = 1$

Calculate ⟶ Gradient ⟶ for B_0, B_1 ⟶ using cost function and datapoints.

datapoint ⟶ Height = 63 and Weight = 127

$$\frac{\partial L}{\partial B_1} = -\frac{2}{n} \times \sum (x_i * (y_i - (B_0 + B_1 * x_i)))$$

$$= -\frac{2}{3} \times (63 * (127 - (-216.12 + 6.45 \times 63)))$$

$$dB_1 = 83.54$$

$$\frac{\partial L}{\partial B_0} = -\frac{2}{n} \times \sum (y_i - (B_0 + B_1 * x_i))$$

$$= -\frac{2}{3} \times (127 - (-216.12 + 6.45 * 63))$$

$$dB_0 = 7.99$$

Update B_1 and B_0

$B_1 = B_1 - \alpha * dB_1$

$B_1 = 0 - 0.01 * 83.54$

$B_1 = -0.84$

$B_0 = B_0 - \alpha * dB_0$

$B_0 = 0 - 0.01 * 7.99$

$B_0 = -0.84$

You can repeat the above steps until you obtain best values for B_0 and B_1

Hypothesis in linear regression

Whether the coefficient has a statistically meaningful impact on dependent variable

- Slope → β_1
  - Significantly different from zero

Whether the independent variable has impact on the dependent variable

Example:

x ⟶ no of hours studied

y ⟶ Exam's scores

We want to test whether the no of hours studied is significant predictor of exam scores.

(significant → means something important or meaningful)

H_0 ⟶ no significant relationship between x and y

$\beta_1 = 0$

H_1 ⟶ significant relationship between x and y

$\beta_1 \neq 0$

let's say we have collected data from 20 students

Calculate coefficient of hours studied

$$\beta_1 = \sum \frac{(x_i - \bar{x})(y_i - \bar{y})}{(x_i - \bar{x})^2}$$

$$\text{Standard error} = \frac{\sqrt{\sum (y_i - \hat{y})^2 / (n-2)}}{\sqrt{\sum (x_i - \bar{x})^2}}$$

let assume $\beta_1 = 0.5$, standard error = 0.1

Calculate t-stats

$$t\text{-stats} = \frac{(\beta_1 - 0)}{SE} = \frac{0.5}{0.1} = 0.5$$

degree of freedom

$$df = n - 1 = 20 - 2 = 18$$

Now p-value

At $t = 5$, $df = 18$ ⟶ p-value = 0.001

Conf interval = $\alpha = 0.05$

As we can our p-value is less than conf interval

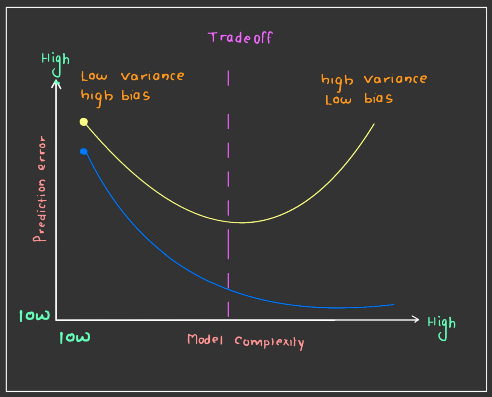
Conclusion

Reject ⟶ H_0 : null hypothesis

The coefficient of hours studied β_1 is significantly different from zero.

means x and y have strong relationship.

Bias Variance tradeoff



● ⟶ Test data

● ⟶ Train data

Tradeoff ⟶ Optimized or generalized model

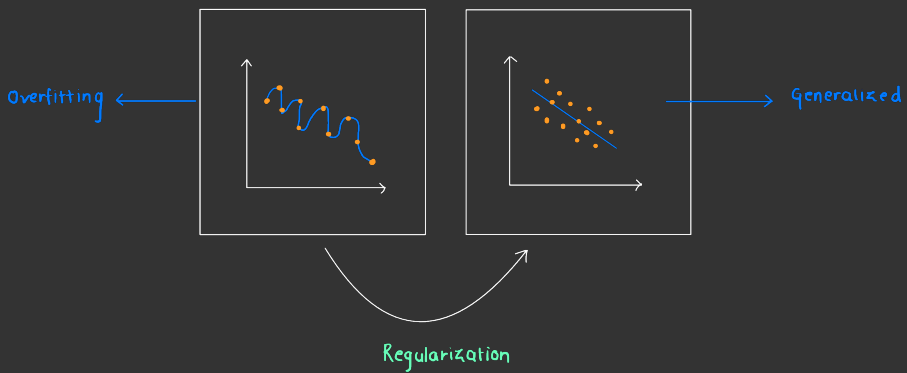
left side area ⟶ underfitting

Right side area ⟶ Overfitting

As we increase model complexity ⟶ prediction error of test data ↑

REGULARIZATION

Regularization ⟶ Technique ⟶ Prevent ⟶ Overfitting ⟶ in model.



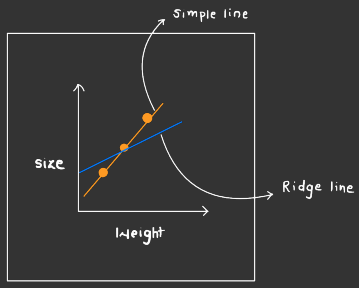
Ridge Regularization

Add penalty to loss function that is proportional to absolute value of model's coefficient

Encouraging ⟶ some coefficient ⟶ to become zero

↳ Not exactly zero.

Ridge regularization ⟶ also known as ⟶ L2 regularization



Cost function = loss + $\lambda * \sum_{i=1}^{n} (\beta_i)^2$

loss ⟶ cost function ⟶ MSE

λ ⟶ strength of regularization or hyperparameter

β_i ⟶ slope ⟶ Model's coefficient

As λ increases ⟶ coefficient leads to shrinkage ⟶ towards the zero

Ridge ⟶ introducing ⟶ bias in model

lasso Regularization

Add penalty term to linear regression equation

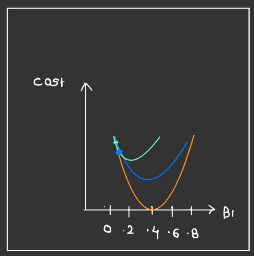
Encouraging ⟶ model to have ⟶ smaller coefficient

or

even push some of them to became ⟶ Exactly zero.

↳ Some of features will get discard

↳ We can say lasso can be use as feature selection



Cost function = loss + $\lambda \sum_{i=1}^{n} |\beta_i|$

λ ⟶ Responsible to set coefficient value exact zero as well ⟶ Not all the time but sometime

loss ⟶ cost function

λ ⟶ strength of regularization

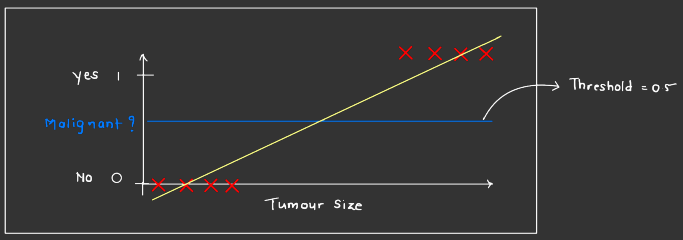
β_i ⟶ slope

Logistic Regression

logistic Regression is a statistical model used for binary classification

→ (0,1), (yes, no)

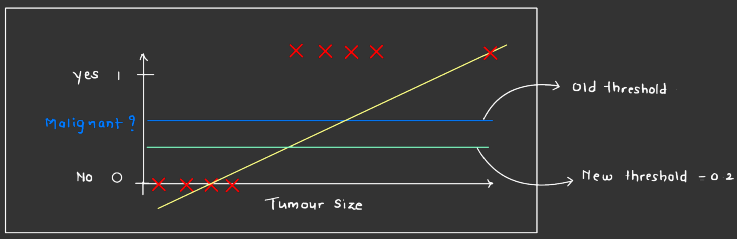
Why logistic regression?



if y ⟶ greater than ⟶ 0.5 then outcome ⟶ 1 ⟶ Malignant tumour.

if y ⟶ less than ⟶ 0.5 then outcome ⟶ 0 ⟶ Benign tumour

let suppose in our dataset we have outlier



Now you can see our linear regression is affected by outliers due to which probability may exceeds 1 or go below 0.

To solve this we need logistic regression for binary classification problem.

logistic regression ⟶ always give output ⟶ (0, 1)

logistic function

$$p = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$$

Equation of best fit line ⟶ $Y = \beta_0 + \beta_1 X$

let say y ⟶ Propability ⟶ But this will exceed 1 or go below 0

but we know the the range of probability ⟶ 0-1

To overcome this we take "Odd" of p

$$P = \beta_0 + \beta_1 x$$

$$\frac{P}{1-P} = \beta_0 + \beta_1 x$$

Odd will always post ⟶ will range between $(0, +\infty)$

Odd is nothing but ratio of ⟶ $\frac{\text{Prob of success}}{\text{Prob of failure}}$

We know that range is restricted

Problem if we restrict the range than our correlation will decrease

To control this we can take log of odd

$$\log\left(\frac{P}{1-P}\right) = \beta_0 + \beta_1 x$$

Now we can take exponent on both side ⟶ To predict outcomes

$$\exp\left[\log\left(\frac{P}{1-P}\right)\right] = \exp(\beta_0 + \beta_1 x)$$

$$e^{\ln\left[\frac{P}{1-P}\right]} = e^{(\beta_0 + \beta_1 x)}$$

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

$$P = e^{(\beta_0 + \beta_1 x)} - Pe^{(\beta_0 + \beta_1 x)}$$

Take p common

$$P = P\left[\frac{e^{(\beta_0 + \beta_1 x)}}{P} - e^{(\beta_0 + \beta_1 x)}\right]$$

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

$$P\left[1 + e^{(B_0 + B_1 x)}\right] = e^{(B_0 + B_1 x)}$$

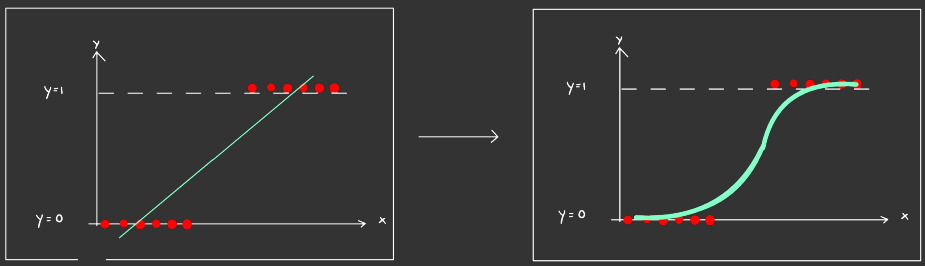
$$P = \frac{e^{(B_0 + B_1 x)}}{1 + e^{(B_0 + B_1 x)}}$$

now divide by $\longrightarrow e^{(B_0 + B_1 x)}$

$$P = \frac{1}{1 + e^{-(B_0 + B_1 x)}} \longrightarrow \text{Sigmoid function}$$

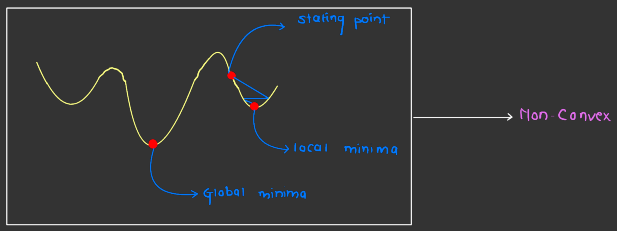
Also known as logistic

Now our linear reg become ⟶ logistic regression



Cost function

In logistic regression we are not dealing with linearity as you can see above graph.



Problems :

Complicated to reach at ⟶ Global minima ⟶ Because we also getting local minima

To solve this we use different cost function called ⟶ log loss

log loss ⟶ derived from the maximum likehood estimation method

$$\text{log loss} = \frac{1}{N} \sum_{i=1}^{N} -(y_i * \log(\hat{y}_i)) + (1-y_i) * (1-\hat{y}_i))$$

What is use of maximum likehood estimator ?

↳ Main aim is to find value of parameters and maximize the likehood function

↳ estimates the coefficient values

↳ B_0, B_1

likehood function ⟶ in simple word its like scorecard it tells you how good or bad your model is explaining real world data

Two outcomes ⟶ Success or failure

↳ Bernouli trials

$y \sim Ber(p)$

Where the p ⟶ Sigmoid function

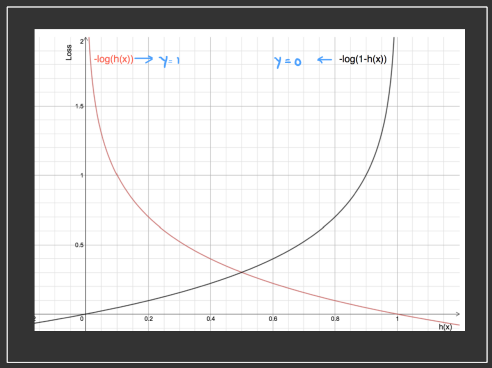
$$L(B_0, B_1, B_2 \dots, B_k) = \prod_{i=1}^{n} [\, p_i^{y_i} * (1-p_i)^{(1-y_i)} \,]$$

P_i ⟶ predicted Probability

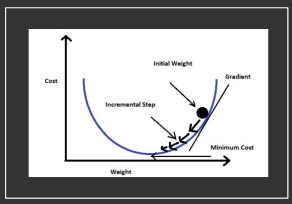
y_i ⟶ Actual outcomes

$$\log(L) = \sum [\, y_i * \log(p_i) + (1-y_i) * \log(1-p_i) \,]$$

$P_i = \hat{y}_i$ ⟶ Both are same



Gradient descent optimization



Update Rule ⟶ $\theta_{new} = \theta_{old} - \eta \frac{\partial J(\theta)}{\partial \theta_j}$

Derivation of cost function

$$\frac{d}{dx}\sigma(x) = \frac{d}{dx}\left[\frac{1}{1+e^{-x}}\right] = \frac{d}{dx}(1+e^{-x})^{-1}$$

$$= -1 * (1+e^{-x})^{-2}(-e^{-x})$$

$$= \frac{-e^{-x}}{-(1+e^{-x})^2}$$

$$= \frac{e^{-x}}{(1+e^{-x})^2}$$

$$= \frac{1}{(1+e^{-x})}\,\frac{e^{-x}}{(1+e^{-x})}$$

$$= \frac{1}{(1+e^{-x})}\,\frac{e^{-x}(1-1)}{(1+e^{-x})}$$

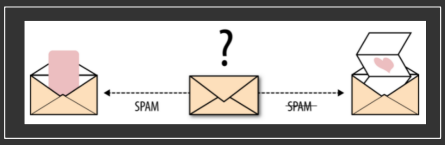
$$= \frac{1}{(1+e^{-x})}\,\frac{(1+e^{-x})-1}{(1+e^{-x})}$$

$$= \frac{1}{(1+e^{-x})}\left[\frac{(1+e^{-x})}{(1+e^{-x})} - \frac{1}{(1+e^{-x})}\right]$$

$$= \frac{1}{(1+e^{-x})}\left[1 - \frac{1}{(1+e^{-x})}\right]$$

$$\frac{d}{d(x)}\sigma(x) = \sigma(x)(1-\sigma(x))$$

Matrics to evaluate classification



Accuracy ⟶ Measure the percentage of correctly classified instances.

Example : 200 ⟶ emails

140 emails ⟶ Correctly predicted as spam ⟶ TP

50 emails ⟶ Correctly predicted as ham ⟶ TN

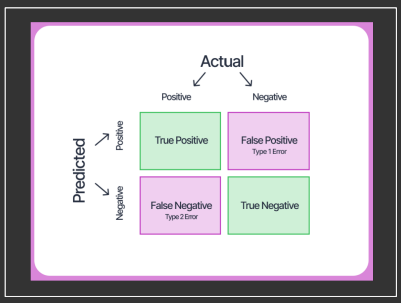
5 emails ⟶ Incorrectly predicted as spam but they are not ⟶ FP

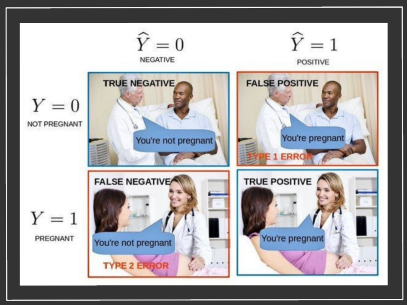
5 emails ⟶ Incorrectly predicted as ham but they are actually spam ⟶ FN

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} = \frac{140+50}{140+50+5+5} = \frac{180}{200} = 0.95 = 95\%$$

↳ may be the not best metric in case of imbalanced dataset

Confusion metric ⟶ Table with combination of predicted and actual value





Precision metrics ⟶ how many instances are correctly predicted as positive are actually correct.

use precision ⟶ When false positive ⟶ higher

$$\text{Precision} = \frac{TP}{TP+FP} \longrightarrow \frac{\text{No of true post}}{\text{No of predicted positive}}$$

Class imbalance ⟶ pos^{t} class lesser than neg^{-} class

↳ use precision.

Example: 100 emails ⟶ (80) spam emails

↳ out of this 75 were actually positive.

5 emails ⟶ false positive

$$\text{precision} = \frac{75}{75+5} = \frac{75}{80} = 0.93$$

93% of email classifier as spam

Application:

Medical, fraud detection, Anomaly detection

Recall metrics ⟶ sensitivity ⟶ True positive rate ⟶ how many positive cases we are able to predict correctly with our model.

use when ⟶ false negative ⟶ higher

$$\text{Recall} = \frac{TP}{TP + FN}$$

Application:

Spam mail detection, customer churn prediction

f1-score ⟶ harmonic mean of recall and precision

use f1-Score when we not sure whethe recall to use or precision

$$\text{f1-Score} = 2 * \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

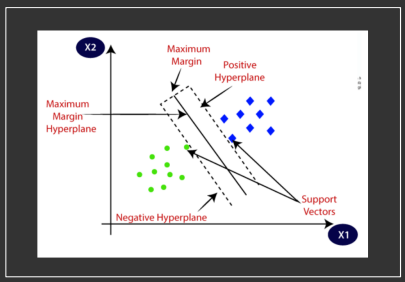
Support Vector Machine

Support vector machine ⟶ Supervised machine learning model ⟶ used for classification and regression.

Aim ⟶ find best hyperplane ⟶ separates ⟶ data into different class.

Types

- linear
  - Data ⟶ linearly separable
- Non-linear
  - Data ⟶ Not linearly separable



Importance Term :

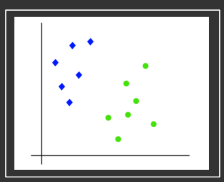
Support vectors ⟶ points that are closest to hyperplane

Margin ⟶ distance between planes.

Hard margin ⟶ aim: perfectly separates the classes
- No missclassification in training data.
- sensitive to outliers.
- ↳ Not always.

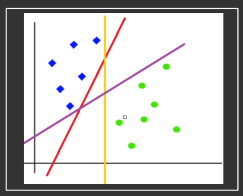
Soft margin ⟶ Missclassification will be there.

How does SVM work ?

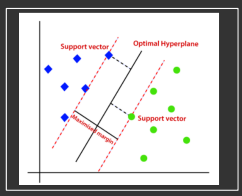


2-dimensional plot

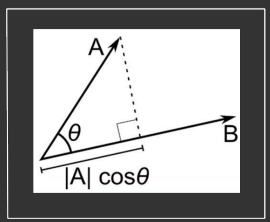
⟶ Separate class by drawing decision boundry



⟶



Mathematical intitution behind SVM



⟶ $A \cdot B = |A| \cdot \cos\theta \, |B|$ (|B| : Magnitude of B)

($|A| \cos\theta$: Projection of A on B)

But in SVM ⟶ |B| ⟶ Not required

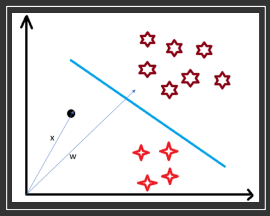
Now |B| ⟶ treated as unit vector

$A \cdot B = |A| \cdot \cos\theta \cdot$ unit vector of B

Use of dot product in SVM

Consider ⟶ point ⟶ (X)

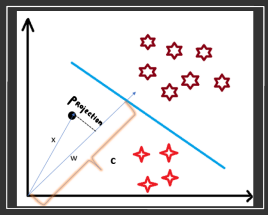
Now we want to know point x lies on left side of plane or right side of plane.



Assume x ⟶ vector than we make ⟶ vector w.

$w \perp$ plane

Suppose ⟶ distance of w from origin to plane. ⟶ c

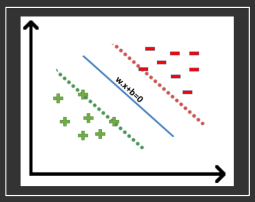


Now ⟶ $\vec{x}.\vec{w} = c$ ⟶ point on decision boundry

$\vec{x}.\vec{w} > c$ ⟶ positive sample

$\vec{x}.\vec{w} < c$ ⟶ Negative sample

Margin in SVM



equation of hyperplane ⟶ $w_1 \cdot x + b = 0$

↳ Normal to plane

To classify a point an neg: or post we need decision rule.

$$\vec{x}.\vec{w} - c \geq 0$$

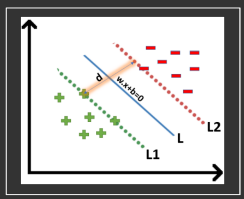
putting -c as b, we get

$$\vec{x}.\vec{w} + b \geq 0$$

hence,

$$y = \begin{cases} +1 \text{ if } \vec{x}.\vec{w} + b \geq 0 & \longrightarrow \text{positive} \\ -1 \text{ if } \vec{x}.\vec{w} + b < 0 & \longrightarrow \text{Negative} \end{cases}$$

Now we need w and b value such that the margin has maximum distance



let say distance ⟶ d

To calculate distance we need L1 and L2 equation.

Assumptions ⟶ $w_1 \cdot x + b = 1 \longrightarrow L1$ and $w_1 \cdot x + b = -1 \longrightarrow L2$

↳ With this assumption we will have equal distance from both classes.

Optimization

for all red points $\longrightarrow \vec{w} \cdot \vec{x} + b \leq -1$

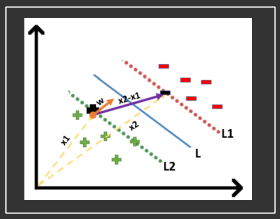
for all green points $\longrightarrow \vec{w} \cdot \vec{x} + b \geq 1$

$\longrightarrow$ Constraints

negative class $\longrightarrow$ $y = -1$ positive class $\longrightarrow$ $y = 1$

Aim: Maximize the distance between l1 & l2 plane.

Take support vector from both class and distance between these two vectors x1 and x2 will be (x2 - x1)



finding projection vector on another vector

$$\frac{(x_2 - x_1) \cdot \vec{w}}{\|w\|}$$

$$\frac{x_2 \vec{w} - x_1 \vec{w}}{\|w\|} \longrightarrow (1)$$

for positive point $y = 1$

$$\vec{w} \cdot x_1 + b = 1$$

$$\vec{w} \cdot x_1 = 1 - b \longrightarrow (2)$$

for negative point $y = -1$

$$\vec{w} \cdot x_1 + b = -1$$

$$\vec{w} \cdot x_1 = -b - 1 \longrightarrow (3)$$

putting (2) and (3) in (1)

$$= \frac{(1-b) - (-b-1)}{\|w\|} = \frac{1 - b + b + 1}{\|w\|} = \frac{2}{\|w\|} = d$$

Hence, the equation which we have to maximize is

$$\text{argmax}(w^*, b^*) \frac{2}{\|w\|} \text{ such that } y_i (\vec{w} \cdot \vec{x_i} + b) \geq 1$$

$\longrightarrow$ Basically this is for Hard margin which means all data are correctly classified

But the the above thing is not always possible so we will look into concept of soft margin.

Soft margin SVM

we know that $\longrightarrow$ max(f(x)) $\longrightarrow$ can written as $\longrightarrow$ min(1/f(x))

$$\text{argmin}(w^*, b^*) \frac{\|w\|}{2} \text{ such that } y_i (\vec{w} \cdot \vec{x} + b) \geq 1$$

To make soft margin equation we add 2 more terms $\longrightarrow$ Xeta, C

$$\text{argmin}(w^*, b^*) \frac{\|w\|}{2} + C \cdot \sum_{i=1}^{n} \zeta_i$$

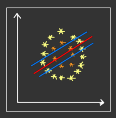
for correctly classified $\longrightarrow$ xeta = 0

for incorrectly classified $\longrightarrow$ distance of that $\perp$ point from its correct hyperplane

Svm kernels

Importance of Kernel ⟶ Transform input data into higher dimensional spaces and making them more easily seperable or capturing more complex relationship among data points

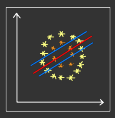
Handles non-linear Separable data



⟶ Non linear data

Accuracy is less ⟶ 50%

Hence we got clear idea that linear Separation is not possible

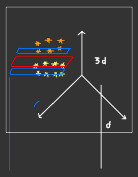


⟶ perform transformation using kernels

↓

Applying mathematical formula

this will convert our data into higher dimension ⟶



Example ⟶ 2d to 3d data

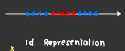
Now linear svc is possible and will get better accuracy

dataset

X	Y
1	yes
2	no
3	yes

let's plot dataset

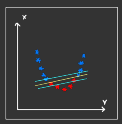
↓



1d Representation

Apply kernel

$x - x^2$



2d Representation