Machine learning

(18/02)

Machine learning is a subfield of AI where algorithms learn patterns from data and make predictions.

The three main types of ML are :

① supervised learning (Eg: Linear Regression, decision trees)

→ This involves training the model on labelled data, where input-output pairs are explicitly provided

- The algo learns to map inputs to outputs and hence can predict outcomes for new unseen data

Eg: Image recognition, identifying whether ite a dog or cat based on labeled data

Span detection, classifying emails as spam or not spam

② unsupervised learning (Eg: K-means clustering, PCA)

→ In this, the algorithm is trained on unlabeled data and identifies patterns, clusters within the data without explicit guidance.

Eg: clustering – like grouping customers based on purchasing patterns.

Recommendation system – suggesting products based on user behaviour and similarities.

* Semi-supervised learning (Eg: self trained models)

→ this combines a small amount of labeled data with a large amount of unlabeled data.

- Bridges the gap between unsupervised & supervised learning by leveraging both types of data.

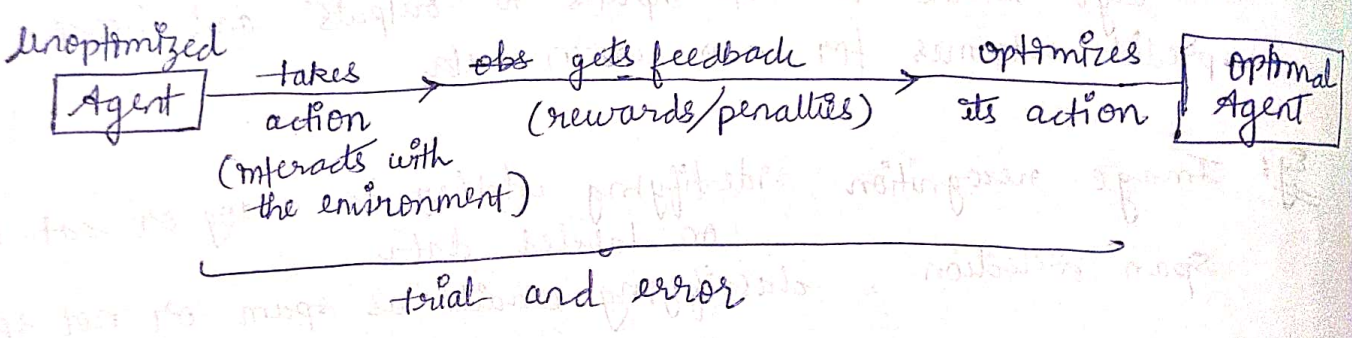
Eg: Speech recognition – training models for speech-to-text conversion using limited labeled ~~dat~~ audio clips alongside vast amount of unlabeled audio.

③ Reinforcement learning (RL)

→ In this, a agent learns to make decisions by interacting with ~~the~~ an environment to achieve a specific goal.

- Agent takes actions, observes the outcomes, and gets feedback in the form of rewards or penalties.

→ Over time, it learns to optimize its actions to maximize the net rewards (through trial and error). → GOAL



Supervised learning

Data Preprocessing → Model Training → Prediction → Evaluation (using metrics like accuracy, precision, MSE)

Types

① Regression

→ Regression is used when the target variable (output) is continuous, meaning it can take any value within a range.

- input data & output data, both need not to be continuous.
  - input data: continuous or categorical
  - output data: continuous

Eg: Predicting house prices
Forecasting stock prices.

★ If you need a number, → regression
★ If you need a category, → classification

Common regression algos:
i) Linear, multiple
ii) Polynomial
iii) ~~Decision~~ Lasso
iv) Ridge

② Classification

→ classification is used when the target variable is categorical, meaning it belongs to a predefined class or category

Eg: Spam/Not Spam
Digit Recognition

Common classification algos:
i) Logistic Regression
ii) SVM
iii) Decision Tree
iv) Random Forest
v) Neural Networks

Linear Regression

- using this algorithm, we find out the linear relation between independent variables and a dependent variable

Independent variables (Regressors or predictors)

Dependent Variable (response)

- **Simple Linear Regression**

→ using this algo, we find the linear relation betn the linearly dependent variable and the independent variable

$$y = mx + c$$

How to Solve

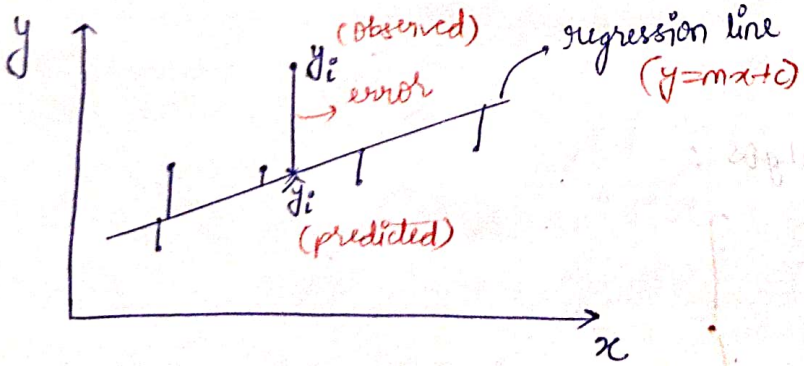
1. Least square Method

→ used to find the eqn of best fitting curve to the set of data points by minimizing the sum of squared differences betn the observed and predicted values

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

$y_i \rightarrow$ observed
$\hat{y}_i \rightarrow$ predicted

$x \rightarrow$ independent variable on x-coordinate
$y \rightarrow$ dependent on y-coordinate



$\because \bar{y} = m\bar{x} + c$

$\Rightarrow \boxed{c = \bar{y} - m\bar{x}}$

$$m = \frac{\sum (x - \bar{x})(y - \bar{y})}{\sum (x - \bar{x})^2} \Rightarrow \boxed{m = \frac{n(\sum xy) - \sum x \sum y}{n\sum x^2 - (\sum x)^2}}$$

Q. Find the line of best fit for data points using LSM:

$(x, y) = (1,3), (2,4), (4,8), (6,10), (8,15)$

x	y	xy	x^2	y'	$\|e\|^2$
1	3	3	1	2.6336	0.134
2	4	8	4	4.3106	0.0964
4	8	32	16	7.6646	0.1125
6	10	60	36	11.0186	1.0375
8	15	120	64	14.3726	0.3936
21	40	223	121		

$$m = \frac{5(223) - 21(40)}{5(121) - (21)^2} = \frac{275}{164} = 1.677$$

$c = 8 - 1.677(4.2)$

$c = 8 - 7.0434 = 0.9566$

eqn, $y = 1.677x + 0.9566$

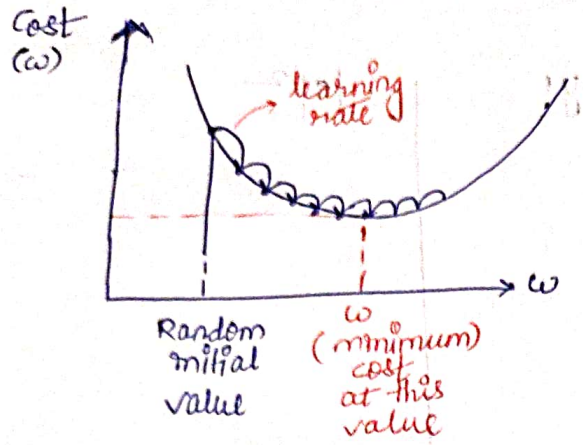
Assumption of the LSM

× i. to apply LSM to predict

1. linear relationship b/w the variables
2. the observations are independent of each other.
3. variance of residual is constant with a mean of
4. Error are distributed normally.

2. Gradient Descent Method

* this is used to find the global minima of a differentiable function



Here,

Aim is to find the values of 'm' and 'c' that defines the relationship betn x and y, correctly with the help of loss function, and minimize the cost function.

→ steps involved in linear regression with GD.

1. Initialize the weight and bias randomly or start with '0'
2. Make prediction with the initial weight and bias.
3. Calculate loss function
4. With the help of differentiation, calculate how loss function changes with respect to weight and bias.
5. Update weight and bias term so as to minimize the loss function (cost fun^n).

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

↳ MSE

$y = mx + c$

$$\frac{\partial(\text{cost fn})}{\partial m} = \frac{\partial}{\partial m}\left(\frac{1}{n}\sum(y_i - \hat{y}_i)^2\right)$$

$$= \frac{\partial}{\partial m}\left(\frac{1}{n}\sum(y_i - (mx+c))^2\right)$$

$$= \frac{1}{n}\frac{\partial}{\partial m}\sum\left(y_i^2 + (mx+c)^2 - 2y_i(mx+c)\right)$$

$$= \frac{1}{n}\frac{\partial}{\partial m}\sum\left(2x(mx+c) - 2y_i x\right)$$

$$= \frac{2}{n}\sum[y_i - (mx+c)](-x)$$

$$\frac{\partial(\text{cost fn})}{\partial c} = \frac{\partial}{\partial c}\left[\frac{1}{n}\sum(y_i - (mx+c))^2\right]$$

$$= \frac{1}{n}\,2\sum(y_i - (mx+c))(-1)$$

$$= -\frac{2}{n}\sum(y_i - (mx+c))$$

Now, update m & c using the following formula

$$m = m - \alpha\frac{\partial(\text{loss fn})}{\partial m}$$

$$c = c - \alpha\frac{\partial(\text{loss fn})}{\partial c}$$

{ $\alpha \rightarrow$ learning rate, $0 < \alpha < 1$

$$\therefore \quad m = m - \alpha\left[\frac{-2}{n}\sum(y_i - (mx+c))x\right]$$

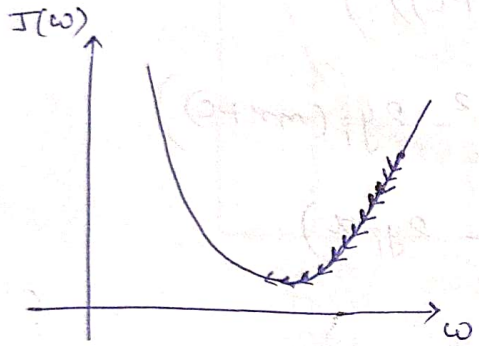
$$\Rightarrow \boxed{m = m - \alpha\left[\frac{-2}{n}\sum(y_i - \hat{y}_i)x_i\right]}$$

& $$\boxed{c = c - \alpha\left[\frac{-2}{n}\sum(y_i - \hat{y}_i)\right]}$$

Learning Rate

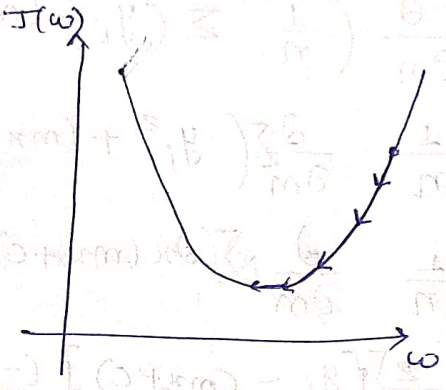
★ Rate of convergence of gradient descent depends upon the learning rate.

Choosing learning rate is a matter of trial and error.



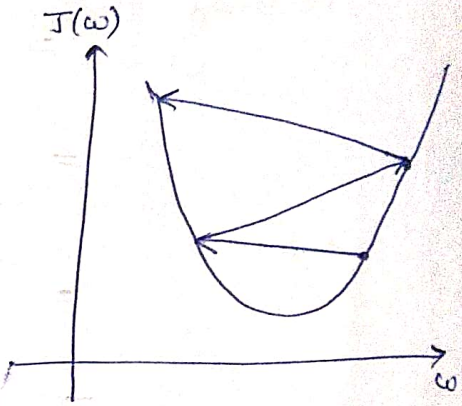
★ Too Low

[requires many iterations]



★ Just Right

[swiftly reaches the min point]



★ Too high

[causes drastic updates which leads to divergent behaviours]

★ α - fixed → the algorithm might get trapped in local minima

Q. Apply G.D on following obs and observe MSE value of 3 iterations $\alpha = 0.0001$ with $m = 10$, $b = 300$

age(x)	salary(y)
30	800
32	950
25	600
43	1050
50	1200
29	740
46	1100

initially ~~m=c=0~~ $m = 10$ $b = 300$

so, $\hat{y} = 10x + 300$.

$$\frac{\partial(cost)}{\partial m} = \frac{-2x}{n}(y_i - \hat{y}_i)$$

$$cost = MSE = \frac{1}{n}\sum(y_i - \hat{y}_i)^2$$

$$m = m - \alpha\left[\frac{-2}{n}\sum(y_i - \hat{y}_i)x\right] \qquad c = c - \alpha\left[\frac{-2}{n}\sum(y_i - \hat{y}_i)\right]$$

$\sum (y_i - \hat{y}_i) = [800-600] + [950-620] + [600-55$
$[1050-730] + [1200-800] + [740-590] +$
$[1100-760]$
$= 1790$

$MSE = \frac{1}{7}\sum (y_i - \hat{y}_i)^2 = 1790 \cdot$ ~~457128.57~~ 78842.8

$m_{new} = 10 - 0.0001\left[\frac{-2}{7}(71560)\right]$

$= 10 + 2.04457$
$= 7.9554$ ~~12.04457~~

$b_{new} = 300 - 0.0001\left[\frac{-2}{7}(1790)\right]$

$= 299.998$ ~~300.05114~~

$\hat{y}_i = 12.04457x + 300.05114$ (2^{nd} epoch)

$\sum (y_i - \hat{y}_i)$
$= [800-661.388] + [950-685.477] + [600-601.165]$
$+ [1050-817.967] + [1200-902.279] + [740-649.34$
$+ [1100-854.101]$

$= 1268.283$

$MSE = \frac{1}{7}(300349.72) = 42907.040$

$m_{new} = 12.04457 - 0.0001\left[\frac{-2}{7}(51397.93)\right]$

$= 13.513$

$b_{new} = 300.05114 - 0.0001\left[\frac{-2}{7}(1268.283)\right] = 300.087$

Multiple Linear Regression

In multiple linear regression, we model relationship bet^n one dependent variable and different multiple independent variable.

Ways to solve :

① Gradient Descent Method (Open form Equation)

For two independent variable ie x_1, x_2

$$y = \omega_1 x_1 + \omega_2 x_2 + b$$

mean squared error

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

$$\omega_1 = \omega_1 - \alpha \left[\frac{-2}{n} \sum (y_i - \hat{y}_i) x_{1i} \right]$$

$$\omega_2 = \omega_2 - \alpha \left[\frac{-2}{n} \sum (y_i - \hat{y}_i) x_{2i} \right]$$

$$b = b - \alpha \left[\frac{-2}{n} \sum (y_i - \hat{y}_i) \right]$$

$$y = (\sum x_i \omega_i) + b$$

Q): Consider the following dataset. Find out linear relation between the response var 'y' and two predictor variables x_1 & x_2 using GD.

y	x_1	x_2	$\hat{y}$
140	60	22	147.5
155	62	25	149.9
159	67	24	166.1
179	70	20	179.9
192	71	15	188.9
200	72	14	193.1
212	75	14	202.1
215	78	11	214.7

$\omega_1 = 3$

$\omega_2 = -1.2$

$b = -6.1$

$\alpha = 0.001$

epoch = 2

$\Rightarrow \quad y = w_1 x_1 + w_2 x_2 + b$

$y = 3x_1 - 1.2x_2 - 6.1$

We need to update w_1, w_2 & b.

$$w_{1\,(new)} = w_1 - \alpha \left[\frac{-2}{n} \sum (y_i - \hat{y_i}) x_{1i} \right]$$

$$\sum (y_i - \hat{y_i}) = (140 - 147.5) + (155 - 149.5) + (159 - 166.1) + (179 - 179.9) + (192 - 188.9) + (200 - 193.1) + (212 - 202.1) + (215 - 214.7)$$

$$= (-7.5) + (5.5) + (-7.1) + (-0.9) + (3.1) + (6.9) + (9.9) + (0.3)$$

$$= 10.2$$

$$\sum (y_i - \hat{y_i}) x_{1i} = (-7.5 \times 147.5) + (5.5 \times 149.9) + (-7.1 \times 166.1) + (-0.9 \times 70) + (3.1 \times 71) + (6.9 \times 72) + (9.9 \times 75) + (0.3 \times 78)$$

$$= 835.1$$

$$\sum (y_i - \hat{y_i}) x_{2i} = 69.1$$

$$w_1 = 3 - 0.001 \left[\frac{-2}{8} \times 835.1 \right] = 3.208$$

$$w_2 = -1.2 - (0.001) \times \left(\frac{-2}{8} \times 69.1 \right) = -1.1892$$

$$b = -6.1 + 0.001 \left(\frac{-2}{8} \times 10.2 \right) = -6.09$$

$\therefore$ $\boxed{y = 3.208x_1 - 1.1892x_2 - 6.09}$

Multiple Linear Regression using Least Square Method

MLR can be expressed as $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k$

$\boxed{y = X\beta}$ $(y = X\beta + \epsilon)$ error will always be there

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1k} \\ x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & & \vdots \\ x_{N1} & x_{N2} & \cdots & x_{Nk} \end{bmatrix} * \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_k \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_k \end{bmatrix} \quad \text{— (1)}$$

$[N \times k]$ $(N >> k)$

If we consider β_0 — then eq (1) can be expressed as:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1k} \\ 1 & x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{N1} & x_{N2} & \cdots & N_{Nk} \end{bmatrix} * \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_k \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_N \end{bmatrix}$$

$[N * (k+1)]$ $(N >> k+1)$

URBAN EDGE — COLORFUL YOU

To estimate the parameters of β, we need to solve n equations and it will be difficult to solve this way because

$N >> k+1$

Consider the sum of sq of residuals, in order to estimate β we have to minimize

$$\sum_{i=1}^{N} e_i^2$$

$$e_i^2 = e'e = \begin{bmatrix} e_1 & e_2 & \dots & e_N \end{bmatrix} \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_N \end{bmatrix}$$

This can be written as a optimization problem or equation i.e.

$$\min_{\beta} e'e$$

$$\frac{\partial e'e}{\partial \beta} = 0. \qquad [\because e = y - X\beta]$$

$$\Rightarrow \frac{\partial}{\partial \beta}[(y-X\beta)'(y-X\beta)]$$

$$\Rightarrow \frac{\partial}{\partial \beta}(y'y - y'X\beta - (X\beta)'y + (X\beta)'X\beta) = 0$$

$$\Rightarrow \frac{\partial}{\partial \beta}(y'y - (y'X\beta)' - X'\beta'y + (X\beta)'X\beta) = 0$$

$$\Rightarrow \frac{\partial}{\partial \beta}(y'y - yX'\beta' - X'\beta'y + (X\beta)'X\beta) = 0$$

$$\Rightarrow \frac{\partial}{\partial \beta}(y'y - 2\beta'X'y + \beta'X'X\beta) = 0$$

Q]: Using LSM find out best regression eqn of the following dataset.

x_1	x_2	y
2	1	1.7
5	3	2.6
6	4	2.8
8	9	2.3
10	11	2.7
11	14	2.4

here, $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2$

$$\begin{bmatrix} 1.7 \\ 2.6 \\ 2.8 \\ 2.3 \\ 2.7 \\ 2.4 \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ 5 & 3 \\ 6 & 4 \\ 8 & 9 \\ 10 & 11 \\ 11 & 14 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \end{bmatrix}$$

We know that, $\beta = (X^T X)^{-1} X^T y$

$$X^T X = \underset{2\times 6}{\begin{bmatrix} 2 & 5 & 6 & 8 & 10 & 11 \\ 1 & 3 & 4 & 9 & 11 & 14 \end{bmatrix}} \underset{6\times 2}{\begin{bmatrix} 2 & 1 \\ 5 & 3 \\ 6 & 4 \\ 8 & 9 \\ 10 & 11 \\ 11 & 14 \end{bmatrix}} = \underset{2\times 2}{\begin{bmatrix} 350 & 377 \\ 377 & 424 \end{bmatrix}}$$

$$[X^T X]^{-1} = \frac{\text{adj}^\circ (X^T X)}{|X^T X|} = \begin{bmatrix} 424/6211 & -377/6211 \\ -377/6211 & 350/6211 \end{bmatrix}$$

$$[X^T X]^{-1} = \begin{bmatrix} & \end{bmatrix}$$

$$(X^T Y) = \begin{bmatrix} 2 & 5 & 6 & 8 & 10 & 11 \\ 1 & 3 & 4 & 9 & 11 & 14 \end{bmatrix}_{2\times 6} \begin{bmatrix} 1.7 \\ 2.6 \\ 2.8 \\ 2.3 \\ 2.7 \\ 2.4 \end{bmatrix}_{6\times 1}$$

$$= \begin{bmatrix} 105 \\ 104.7 \end{bmatrix}$$

Now $(X^T X)^{-1} (X^T Y) = \begin{bmatrix} 424 \\ \\ \end{bmatrix}$

$$\beta = \begin{bmatrix} -0.242 \\ 0.462 \end{bmatrix}$$

$$\boxed{y = -0.242 x_1 + 0.462 x_2}$$

→ Types of gradient descent

① Batch GD

→ The entire dataset is used to calculate the gradient and update the parameters.

② Stochastic GD

→ A single sample is used to update the parameters at a time.

- converges quickly than batch GD
- can escape from local maxima

③ Mini Batch GD

→ It is a fine balance between stochastic and batch GD.

→ In this the whole dataset is divided into batches and then batch run one after another and update the parameter.

→ We do feature scaling with the help of these two techniques to have all the features on the same scale for better results (less variance)

* Normalisation

$$x_{new} = \left(\frac{x - x_{min}}{x_{max} - x_{min}}\right)$$

To bring in (a,b) ranges

$$x_{new} = a + \left(\frac{x - x_{min}}{x_{max} - x_{min}}\right)(b-a)$$

→ * when features are of diff. scales

* Standardisation (Z-score normalisation) * affected by outliers

$$x_{new} = \frac{x - \bar{x}}{\sigma}$$

* when we want to ensure zero mean and unit sd
* much less affected by outliers.

$y = f(x) = \hat{f}(x, \theta) + \epsilon$

ϵ (error) → Irreducible error

Reducible error

Bias & Variance

★ Training Error

$\text{Bias} = E(\hat{Y}) - Y$

$\text{Bias} = E[\hat{f}(x, \theta)] - Y$

Bias: It is defined as the diff. between expectation of estimated output ($\hat{Y}$) to true output (Y)

- Bias is introduced when a model is approximating a real-world problem (which is actually complex) with a simplified model
  - → Results in underfitting — fails to capture the underlying patterns in the data

Variance: Happens when a model is too complex.
- → Overfitting: the model fits the training data very closely but may fail with the new data

$\text{var}\ \hat{f}(x, \theta) = E\left[\hat{f}(x) - E[\hat{f}(x)]\right]^2$

The tradeoff

Simple model — High bias, low variance
→ underfit (misses patterns)

Complex model — High variance, low bias
→ overfit (too sensitive)

* We need to find the balance.

- Low variance means $\hat{f}(x)$ based on different training set doesn't change much.
- High variance means $\hat{f}(x)$ has high variance ~~in the~~ (small change in the training data set can result large change in $\hat{f}(x)$.)

$\because$ Bias $= [E[\hat{y}] - y]$

Variance $= E\left[\hat{f}(x,\theta) - E[\hat{f}(x,\theta)]\right]^2$

$$\Rightarrow \quad MSE = E[(y-\hat{y})^2] = \left[(E[\hat{y}]-y)^2\right] + E\left[\hat{f}(x,\theta) - E[\hat{f}(x,\theta)]\right]^2$$

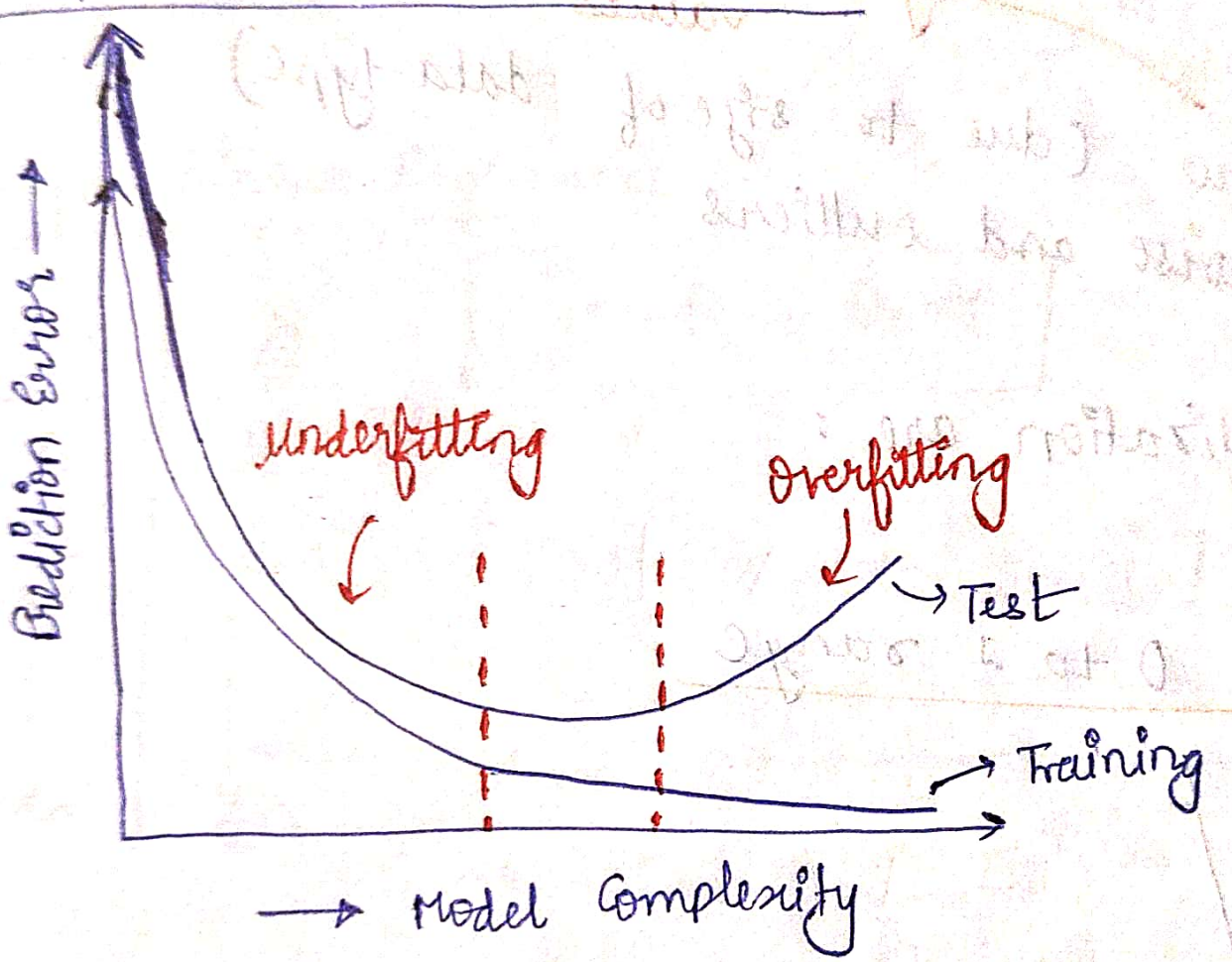
$$\text{i.e.} \quad \boxed{MSE = Bias^2 + Variance}$$

$$\boxed{MSE = Bias^2 + Variance}$$

Bias ↑↑ ⇒ Underfitting

Variance ↑↑ ⇒ Overfitting

→ Supervised Model Evaluation



Regularisation

→ Regularisation is a technique used to reduce overfitting in a model and improve predictive performance.

① ~~Ridge~~ Regularisation (Least Absolute selection and Shrinkage Operator) LASSO
(L1 Regularisation)

- In this, we add the ~~sum of~~ absolute sum of the coefficients to the MSE (or RSS)

$$\text{Loss} = \text{MSE} + \lambda \sum_{i=1}^{n} |w_i|$$

② Ridge Regularisation
(L2 Regularisation)

- In this, we add sum of square of the coefficients to the MSE (or RSS)

$$\text{loss} = \text{MSE} + \lambda \sum_{i=1}^{n} w_i^2$$

* In ridge regression, the coefficient shrinks towards 0 whereas in lasso regression, it shrinks to 0.

- So when feature selection is done, lasso wale case mein, the feature is eliminated and in case of ridge, the impact of that feature is reduced as the value of coefficient is shrinked.

Logistic Regression

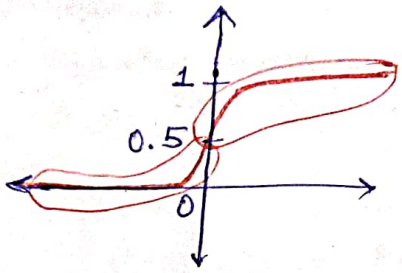
→ supervised L. Algo used for classification.

. The goal of logistic regression is to predict the probability that an instance belongs to a class or not.

- used for binary classification.
- uses sigmoid fn that produces a probability value

$$\text{Class} = \begin{cases} 0 & ; \quad p(z) < 0.5 \\ 1 & ; \quad p(z) \geq 0.5 \end{cases}$$

* Sigmoid Function



$$f(z) = \frac{1}{1+e^{-z}}$$

$$z = \beta_1 x_1 + \beta_0$$
$$z = \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k + \beta_0$$

Q]: Consider the dataset of 5 observations of no. of hours studied by a student.

(i) Predict the class level of student who studied for 30 hrs, 34 hrs.

[ii] Calculate how many hours a student should study so that the student will pass with 95% probability.

$\beta_1 = 2$
$\beta_0 = -64$

Hrs of studies	Pass/Fail (Class)
28	0
14	0
32	1
27	1
38	1

(i) Here, $z = \beta_0 + \beta_1 x$

for 30: $z = -64 + 2(30) = -4$

$p = \frac{1}{1+e^{+4}} = 0.017 < 0.5$

Fail

for 34: $z = -64 + 2(34) = 4$

$p = \frac{1}{1+e^{-4}} = 0.98 > 0.5$

Pass

[ii] $p = 0.95$

$\frac{1}{1+e^{-z}} = 0.95$

$\Rightarrow z = 2.94$

$\Rightarrow \boxed{x = 33.47}$

NAIVE-BAYES CLASSIFIER

→ It is supervised learning algorithm used for classification.

• It is named as 'NAIVE' 'BAYES'

'NAIVE' — because it assumes that one feature does not affect another feature (-feature independence)

'BAYES' — based on Bayes Theorem

Assumptions :

1) Feature independence
2) Features are equally important
3) No missing data

Bayes Formula

$$P(Y|X) = \frac{P(X|Y) \cdot P(Y)}{P(X)} \quad ; \; P(X) \neq 0$$

here, $P(Y|X)$ is called posterior probability

$P(X|Y)$ → Likelihood or conditional probability

$P(Y)$ → Prior Probability

Y → class variable & X → Feature vector of size n

$$P(Y|X = x_1, x_2, x_3, \ldots x_n) = \frac{P(x_1|Y) \cdot P(x_2|Y) \cdot \ldots P(x_n|Y) \cdot P(Y)}{P(x_1) \cdot P(x_2) \cdot P(x_3) \ldots P(x_n)}$$

$$P(Y|X = x_1, x_2, \ldots, x_n) = \frac{P(Y) \cdot \prod_{i=1}^{n} P(x_i|Y)}{\prod P(x_i)}$$

- Example: consider given dataset & find out the test sample today (X) = {Sunny, Hot, Normal, False} The class level using naive bayes classification

To reach out the result we need to calculate

$$P(y|(X = x_1, x_2, \ldots x_n) = \frac{P(x_1|y) \cdot P(x_2|y) \cdot P(x_n|y) \cdot P(y)}{P(x_1) \cdot P(x_2) \cdot \ldots P(x_n)}$$

OR $$P(y|x_1 x_2 x_3 \ldots x_n) = P(y) \prod_{i=1}^{n} P(x_i|y)$$

$$y = \arg\max P(y) \prod_{i=1}^{n} P(x_i|y)$$

	Outlook	Temp	Humidity	Windy	Play Golf
1	Rainy	Hot	High	false	No
2	Rainy	Hot	High	True	No
3	Overcast	Hot	High	F	Yes
4	Sunny	Mild	High	F	Yes
5	Sunny	Cool	Normal	F	Yes
6	Sunny	Cool	Normal	T	No
7	Overcast	Cool	Normal	T	Yes
8	Rainy	Mild	High	F	No
9	Rainy	Cool	Normal	F	Yes
10	Sunny	Mild	Normal	F	Yes
11	Rainy	Mild	Normal	T	yes
12	Overcast	Mild	High	T	Yes
13	Overcast	Hot	Normal	F	yes
14	Sunny	Mild	High	T	No

Class	Outlook	Temp	Humidity	Windy	Prior Prob P(C)
Yes	P(Sunny|Y) = 3/9	P(H|Y) = 2/9	P(N|Y) = 6/9	P(F|Y) = 6/9	P(Y) = 9/14
NO	P(Sunny|N) = 2/5	P(H|N) = 2/5	P(N|N) = 1/5	P(F|N) = 2/5	P(N) = 5/14

$P(C_k|X) = P(X|C_k) \cdot P(C_k)$

$P(Yes|today) = P(x_1|C_k) \cdot P(x_2|C_k) \ldots P(x_n|C_k) P(C_k)$

$= 3/9 \times 2/9 \times 6/9 \times 6/9 \times 9/14$

$= 0.021$

$P(No|today) = 2/5 \times 2/5 \times 1/5 \times 2/5 \times 5/14$

$= 0.004$

$y = \text{argmax}_y \, P(y) * \prod_{i=1}^{n} P(x_i|y)$

class label of the test sample today is 'yes'.

Find out the class label of the test sample $X = (Rainy, Hot, High, False)$ using naive Bayes

Class	Outlook	Temp	Humidity	Windy	Prior Prob
Yes	2/9	2/9	3/9	6/9	P(Y) = 9/14
No	3/5	2/5	4/5	2/5	P(N) = 5/14

$P(Yes|today) = 2/9 \times 2/9 \times 3/9 \times 6/9 \times 9/14 = 0.007$

$P(No|today) = 3/5 \times 2/5 \times 4/5 \times 2/5 \times 5/14$

$= 0.027$

class is 'no'.

Advantage:

- easy to implement and computationaly efficient
- it is effective with a large no. of feature
- it performs well with the limited training data
- it also performs well in the presence of categorical features
- for numerical features, it performs well if it shows normal distribution

Disadvantage:

- it may be influenced by irrelevant attributes so feature selection is required before applying naive Bayes classifier.
- It may assign zero probability to unseen events leading to poor generalisation
- As it assumes all features are independent which may not always hold true in the real world data.

Q. Consider the given dataset. Predict the class level of given test sample $X = \{f11, f21, f31\}$ using naive Bayes classifier based on the following obs

Using NB classifier based on following obs

class label	f11	f21	f31	Total
obj1	350	450	0	650
obj2	400	300	350	400
obj3	50	100	50	150

Summarized table or frequency table

$X = \{f11, f21, f31\}$ class table.

$$P(obj1 \mid X) = P(X \mid obj1) \cdot P(obj1)$$
$$= P(f11 \mid obj1) \cdot P(f21 \mid obj1) \cdot P(f31 \mid obj1) \cdot P(obj1)$$
$$= \frac{350}{650} \times \frac{450}{650} \times \frac{0}{650} \times \frac{650}{1200}$$

$$P(obj2 \mid X) = P(X \mid obj2) \cdot P(obj2)$$
$$= P(f11 \mid obj2) \cdot P(f22 \mid obj2) \cdot P(f31 \mid obj2) \cdot P(obj2)$$
$$= \frac{400}{400} \times \frac{800}{400} \times \frac{350}{400} \times \frac{400}{1200} = 0.218$$

$$P(obj3 \mid X) = P(f11 \mid obj3) \cdot P(f21 \mid obj3) \cdot P(f31 \mid obj3) \cdot P(obj3)$$
$$= \frac{50}{150} \times \frac{100}{150} \times \frac{50}{150} \times \frac{150}{1200} = 0.009$$

Gaussian Naive Bayes Classification

$$P(Y=c \mid X) = P(X \mid Y=c) \cdot P(Y=c)$$

$$P(X \mid Y=c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} e^{-\frac{(x-\mu_c)^2}{2\sigma_c^2}}$$

μ_c = mean of class c

σ_c = std. deviation of class c

2. Based on given dataset determine class level of the test sample $X = \{x_1 = 6, x_2 = 130, x_3 = 8\}$ using Naive Bayes classifier.

f_1	f_2	f_3	class
6	180	12	C1
5.92	190	11	C1
5.58	170	12	C1
5.78	165	10	C1
5	100	6	C2
5.5	150	8	C2
5.4	130	7	C2
5.72	150	9	C2

$\mu_{f_1C_1} = 6+5.92+5.58+5.78/4 = 5.82$

$\mu_{f_1C_2} = 5+5.5+5.4+5.72/4 = 5.405$

$$\sigma_{f_1C_1} = \sqrt{\frac{\sum (x-\bar{x})^2}{n-1}} = \sqrt{\frac{(6-5.8)^2+(5.92-5.8)^2+(5.58-5.8)^2+(5.78-}{3}}$$

$= 0.185$

$$\sigma_{f_1C_2} = \sqrt{\frac{(5-5.405)^2+(5.5-5.405)^2+(5.4-5.405)^2+(5.72-5.405)^2}{3}}$$

$= 0.301$

$P(C_1|X) = P(f_1|C_1) \cdot P(f_2|C_1) \cdot P(C_1) \cdot P(f_3|C_1)$

$\mu_{f_2C_1} = 180+190+170+165/4 = 176.25$

$\mu_{f_2C_2} = 100+150+130+150/4 = 132.5$

$$\sigma_{f_2C_1} = \sqrt{\frac{(180-176.25)^2+(190-176.25)^2+(170-176.25)^2+(165-176.25)^2}{3}}$$

$= 11.08$

$\sigma_{f_2C_2} = 23.63$

$\mu_{f3C1} = 12+11+12+10/4 = 11.25$

$\mu_{f3C2} = 6+8+7+9/4 = 7.5$

$\sigma_{f3C1} = 0.96$

$\sigma_{f3C2} = 1.3$

$$P(f1|C1) = \frac{1}{\sqrt{2\pi(0.185)^2}} e^{-\frac{(6-5.82)^2}{2(0.185)^2}} = 0.14$$

$$P(f2|C1) = \frac{1}{\sqrt{2\pi(11.08)^2}} e^{-\frac{(130-176.25)^2}{2(11.08)^2}} = 0.000037$$

$$P(f3|C1) = \frac{1}{\sqrt{2\pi(0.96)^2}} e^{-\frac{(8-11.25)^2}{2(0.96)^2}} = 0.00073$$

$P(C1|X) = P(C1) \cdot P(f1|C1) \cdot P(f2|C1) \cdot P(f3|C1)$

$= 0.5 \times 0.147 \times 0.00003 \times 0.00073 = 0.000063$

$= 1.8 \times 10^{-9}$

$$P(f1|C2) = \frac{1}{\sqrt{2\pi(0.3)^2}} e^{-\frac{(6-5.4)^2}{2(0.3)^2}} = 0.03386$$

$$P(f2|C2) = \frac{1}{\sqrt{2\pi(23.6)^2}} e^{-\frac{(130-132.5)^2}{2(23.6)^2}} = 0.22$$

$$P(f3|C2) = \frac{1}{\sqrt{2\pi(1.3)^2}} e^{-\frac{(8-7.5)^2}{2(1.3)^2}} = 0.21$$

$P(C2|X) = 0.5 \times 0.143 \times 0.124 \times 0.0004 = 0.0000083$

$= 7.6 \times 10^{-4}$

C2

KNN Algorithm

→ It is a supervised learning algorithm which is used to predict the class of a sample based on the class of K-nearest neighbors.

- used for classification as well as regression.
- It is called Lazy learner Algorithm because it does not train a model explicitly ; it just memorizes the data and makes predictions in real time.

* Choosing the right K-value and distance metric is crucial.

Distance Metric

for two points $x(x_1, x_2)$ & $y(y_1, y_2)$

- Manhattan distance

$$d = |x_1 - y_1| + |x_2 - y_2|$$

- Euclidean distance

$$d = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$$

- Minkowski distance

→ $p = 1$ Manhattan

→ $p = 2$ Euclidean

$$d = \sqrt[p]{|x_1 - y_1|^p + |x_2 - y_2|^p}$$

Steps to perform K-NN

step ① Initialize K

→ many ways to choose K
- odd number
- sq. root of n (idk what n is!?)
- elbow method
- cross validation

step ② Calculate distance from each training sample to the test sample using the dist. formula

step ③ Arrange the distances calculated in ascending order

step ④ Pick the first K training samples and get their class.

step ⑤ Declare the class of test sample, based on the majority of class of k-neighbors.

Q]:

X_1	X_2	class	distance of (25,80)
40	20	C_1	61.84
50	50	C_2	39.05
60	90	C_2	36.40
10	25	C_1	57.00
25	80	? (Test sample)	

Let k=3

$y_1 = 25 \quad y_2 = 80$

$$d_1 = \sqrt{(25-40)^2 + (80-20)^2} = 61.84$$

sorting these distance values,

$d_3 < d_2 < d_4 < d_1$

$(C_2)\ (C_2)\ (C_1)\ (C_1)$

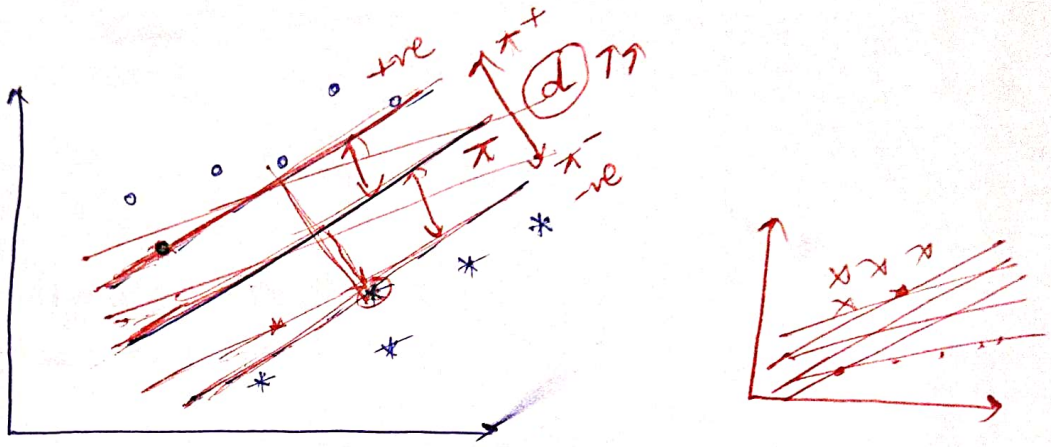
from the 3 nearest neighbors, we get class of (25,80) as C_2.

SVM (Support Vector Machine)

→ This is a popular supervised learning algorithm generally used for classification.

- The goal of SVM is to find a hyperplane that can segregate 'n' dimensional space into classes.

hyperplane → best line or decision boundary.



Terminologies in SVM

Support vectors : The data points (or vectors) that are closest to the hyperplane & which affect the position of hyperplane are termed as support vectors.

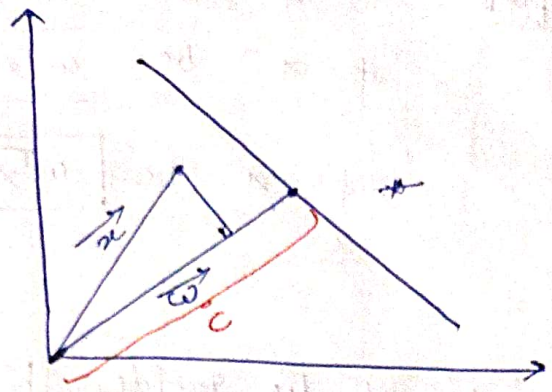
Hyperplane : Multiple decision boundaries can be drawn for segregating the classes; the best boundary is known as hyperplane of SVM.

- Dimension of hyperplane = (n-1) where n → no. of features in the dataset

Margin : The distance between the support vectors and the hyperplane is called as margin.

→ Goal of SVM is to maximize the margin
↓
optimal hyperplane

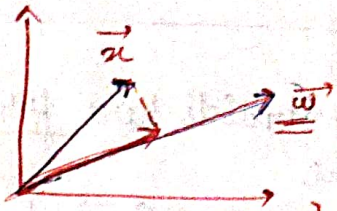
Use of dot product in SVM



- $\vec{x}$ → vector to a random point
- $\vec{\omega}$ → vector to the hyperplane
- c → distance from origin to the hyperplane

Linear SVM

Mathematically,



- $\vec{x}.\vec{\omega} = c$ (point lies on decision boundary)
- $\vec{x}.\vec{\omega} > c$ (positive samples) — right side of hyperplane
- $\vec{x}.\vec{\omega} < c$ (negative samples) — left side of hyperplane

$\vec{x}.\frac{\vec{\omega}}{\|\omega\|}$

Find value of w and b (hyperplane) for which the margin is maximum.

$\vec{\omega}.\vec{x} > c$ (+ve samples)

$\Rightarrow \vec{\omega}.\vec{x} - c > 0$

$\Rightarrow \boxed{\vec{\omega}.\vec{x} + b > 0}$ Decision Rule

$-c \rightarrow +b$

$\vec{\omega}.\vec{x} - c < 0$

$\vec{\omega}.\vec{x} + b < 0$

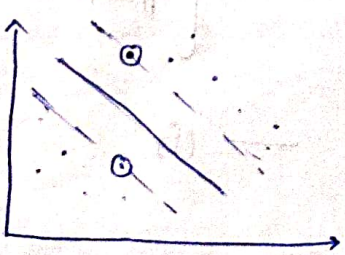
+ve value for positive samples (+1)

-ve value for negative samples (-1)

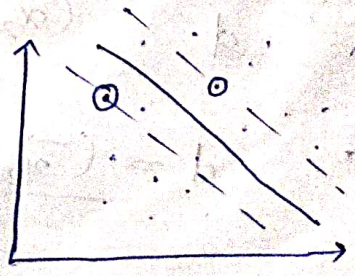
Decision Rule

$$\hat{y}_i = \begin{cases} +1 & ; \text{ if } \vec{\omega}.\vec{x_i} + b \geq 0 \\ -1 & ; \text{ if } \vec{\omega}.\vec{x_i} + b < 0 \end{cases}$$

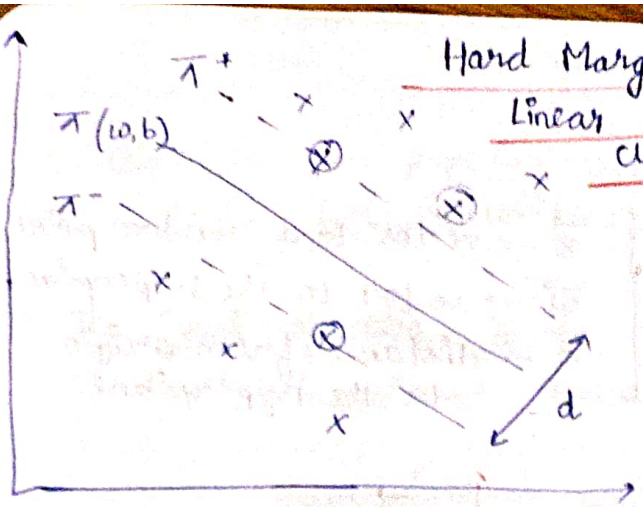
Hard Margin



Soft Margin



Hard Margin Linear SVM classifier



Let us assume the eq^n of

π^+ be $\omega^T x + b = 1$ ①

& π^- be $\omega^T x + b = -1$ ②

& π be $\boxed{\omega^T x + b = 0}$

The constraints here for the points to not lie inside this hyperplane are

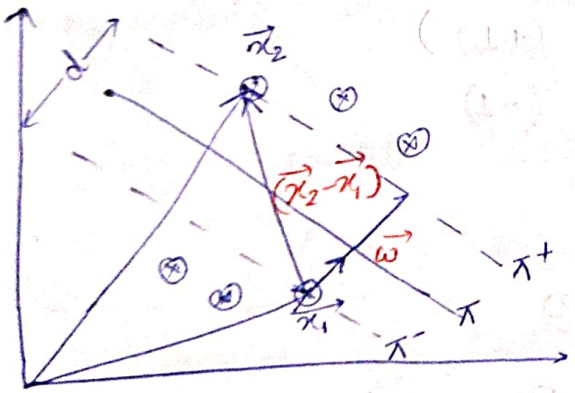
$$\left.\begin{array}{l} \vec{\omega}\cdot\vec{x} + b \geq 1 \\ \& \quad \vec{\omega}\cdot\vec{x} + b \leq -1 \end{array}\right\}$$

If we multiply the above constraints with the labels of these points ie $+1$ & -1 : we get

$$\left.\begin{array}{l} \hat{y}\,(\vec{\omega}\cdot\vec{x} + b) \geq 1 \quad *(+1) \\ \hat{y} * (\vec{\omega}\cdot\vec{x} + b) \geq +1 \quad *(-1) \end{array}\right\} \quad y_i(\vec{\omega}\cdot\vec{x_i} + b) \geq 1$$

for all vectors

For support vectors, $y_i(\vec{\omega}\cdot\vec{x_i} + b) = 1$



To calculate d,

we need to find the projection of $(\vec{x_2} - \vec{x_1})$ on $\vec{\omega}$.

[$\because$ $\vec{\omega}$ is perpendicular to all three lines]

$$d = (\vec{x_2} - \vec{x_1}) \cdot \frac{\vec{\omega}}{\|\omega\|}$$

$\left\{\frac{\vec{\omega}}{\|\omega\|} \rightarrow$ unit vector in direct^n of $\vec{\omega}\right\}$

$$\Rightarrow \quad d = \frac{(\vec{x_2}\cdot\vec{\omega} - \vec{x_1}\cdot\vec{\omega})}{\|\omega\|} \quad \text{— (1)}$$

Now, for x_2 ;

$$y_i(\vec{\omega}.\vec{x_i}+b) = 1$$

$$1(\vec{\omega}.\vec{x_2}+b) = 1$$

$$\Rightarrow \vec{\omega}.\vec{x_2} = (1-b) \quad \text{---(ii)}$$

for x_1 ;

$$(-1)(\vec{\omega}.\vec{x_1}+b) = 1$$

$$\Rightarrow -\vec{\omega}.\vec{x_1} - b = 1$$

$$\Rightarrow \vec{\omega}.\vec{x_1} = (-1-b) \quad \text{---(iii)}$$

Putting values from eqn (ii) & (iii) in eqn (i) ;

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

$$\Rightarrow \boxed{d = \frac{2}{\|\omega\|}}$$

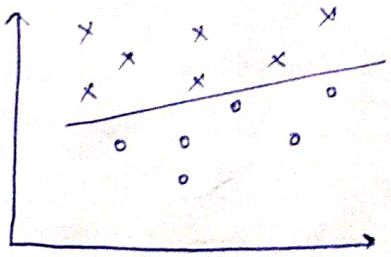
HARD MARGIN SVM

* $\underset{(\omega^*, b^*)}{\text{argmax}} \frac{2}{\|\omega\|}$

find such values of ω & b for which $\frac{2}{\|\omega\|}$ is max^M

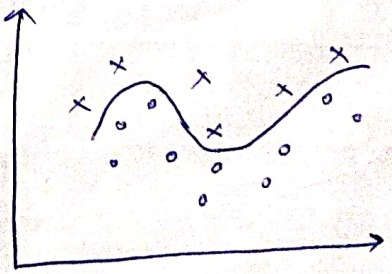
given that $y_i(\vec{\omega}.\vec{x_i}+b) \geq 1$

Linear SVM



The two classes can be separated using a straight line

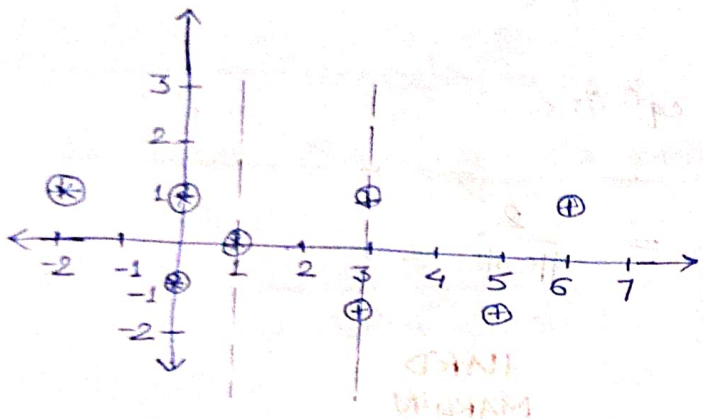
Non-linear SVM



The two classes cannot be separated using a straight line

Q]: Determine the eqn of hyperplane that divides the datapoints into two classes.

+vely labeled [(3,1), (3,-1), (6,1), (5,-1)]
-vely labeled [(1,0), (-2,1), (0,-1), (0,1)]



Support vector of -ve class
$S_1(1,0)$

support vector of +ve class
$S_2(3,1)$
$S_3(3,-1)$
} both of them

⇒ We will get the support vectors from the graph

$$S_1\begin{pmatrix}1\\0\end{pmatrix}, S_2\begin{pmatrix}3\\1\end{pmatrix}, S_3\begin{pmatrix}3\\-1\end{pmatrix}$$

assigning 1 in all 3 support vectors as bias

$$\bar{S}_1 = \begin{pmatrix}1\\0\\1\end{pmatrix}, \bar{S}_2 = \begin{bmatrix}3\\1\\1\end{bmatrix}, \bar{S}_3 = \begin{pmatrix}3\\-1\\1\end{pmatrix}$$

Lets take three parameters α_1, α_2 & α_3 for S_1, S_2 & S_3 calculate the weight vector, we need to go for the following eqn.

$$\alpha_1 \bar{S}_1 \bar{S}_1 + \alpha_2 \bar{S}_2 \bar{S}_1 + \alpha_3 \bar{S}_3 \bar{S}_1 = -1 \quad \text{-vely labelled}$$
$$\alpha_1 \bar{S}_1 \bar{S}_2 + \alpha_2 \bar{S}_2 \bar{S}_2 + \alpha_3 \bar{S}_3 \bar{S}_2 = 1$$
$$\alpha_1 \bar{S}_1 \bar{S}_3 + \alpha_2 \bar{S}_2 \bar{S}_3 + \alpha_3 \bar{S}_3 \bar{S}_3 = 1 \quad \text{+vely labelled}$$

$$\alpha_1 \begin{pmatrix}1\\0\\1\end{pmatrix}\begin{pmatrix}1\\0\\1\end{pmatrix} + \alpha_2 \begin{pmatrix}3\\1\\1\end{pmatrix}\begin{pmatrix}1\\0\\1\end{pmatrix} + \alpha_3 \begin{pmatrix}3\\-1\\1\end{pmatrix}\begin{pmatrix}1\\0\\1\end{pmatrix} = -1$$

$$\Rightarrow 2\alpha_1 + 4\alpha_2 + 4\alpha_3 = -1 \quad \text{---(i)}$$

$$\alpha_1 \begin{pmatrix}1\\0\\1\end{pmatrix}\begin{pmatrix}3\\1\\1\end{pmatrix} + \alpha_2 \begin{pmatrix}3\\1\\1\end{pmatrix}\begin{pmatrix}3\\1\\1\end{pmatrix} + \alpha_3 \begin{pmatrix}3\\-1\\1\end{pmatrix}\begin{pmatrix}3\\1\\1\end{pmatrix} = 1$$

$$\Rightarrow \quad 4\alpha_1 + 11\alpha_2 + \alpha_3 = 1 \quad \text{---(ii)}$$

$$\alpha_1 \begin{pmatrix}1\\0\\1\end{pmatrix}\begin{pmatrix}3\\-1\\1\end{pmatrix} + \alpha_2 \begin{pmatrix}3\\1\\1\end{pmatrix}\begin{pmatrix}3\\1\\1\end{pmatrix} + \alpha_3 \begin{pmatrix}3\\-1\\1\end{pmatrix}\begin{pmatrix}3\\-1\\1\end{pmatrix} = 1$$

$$\Rightarrow \quad 4\alpha_1 + 9\alpha_2 + 11\alpha_3 = 1 \quad \text{---[iii]}$$

$$\alpha_1 = -3.5, \quad \alpha_2 = 0.75 \text{ \& } \alpha_3 = 0.75$$

$$\overline{\omega} = \sum \alpha_i \overline{s_i} = \alpha_1 \overline{s_1} + \alpha_2 \overline{s_2} + \alpha_3 \overline{s_3}$$

$$\Rightarrow \overline{\omega} = -3.5 \begin{pmatrix}1\\0\\1\end{pmatrix} + 0.75 \begin{pmatrix}3\\1\\1\end{pmatrix} + 0.75 \begin{pmatrix}3\\-1\\1\end{pmatrix} = \begin{pmatrix}1\\0\\2\end{pmatrix}$$

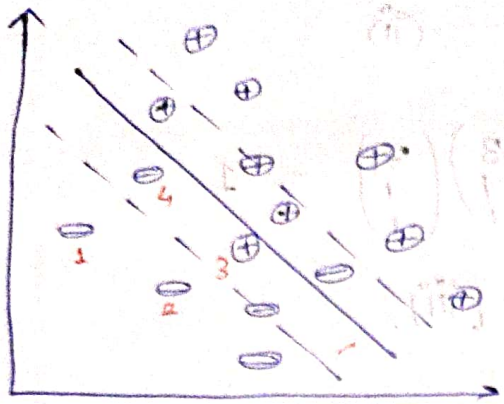
Equating it with hyperplane offset;

$$y = \omega x + b \quad \text{where} \quad \omega = \begin{pmatrix}1\\0\end{pmatrix} \quad b = -2$$

$$\frac{x}{d/\omega_1} + \frac{y}{d/\omega_2} = 1 \Rightarrow \frac{x}{2/1} + \frac{y}{2/0} = 1 \Rightarrow \boxed{x = 2}$$

$$\left\{ d = \frac{2}{\|\omega\|} = \frac{2}{\sqrt{1^2+0^2}} = 2 \right\}$$

Soft Margin Linear SVM Classifier



If the datapoints are almost linearly separable, thats why hard margin is not possible.

From hard margin, $\underset{(w^*, b^*)}{\text{argmax}} \frac{2}{\|w\|}$

$\rightarrow \underset{(w^*, b^*)}{\text{argmin}} \frac{\|w\|}{2}$

Now, we add a ~~hyperparameter~~ to account for classification

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

(ξ_i: slack variable (zeta))

$\frac{\|w\|}{2}$: margin error; $C\sum_{i=1}^{n} \xi_i$: classification error

we need to minimize both

Here, C is regularisation constant.

A higher value of C implies a stricter penalty for margin violation.

This leads to a smaller margin with fewer misclassifications.

★ Hinge loss

Hinge loss is the loss function used in SVM that ensures that the misclassified points contribute to the loss and the correctly classified points do not.

* It penalizes points that are within the margin or misclassified. (uses slack variable to do so)

- $\xi = 0$: point is correctly classified and outside the margin.
- $0 < \xi \leq 1$: point is inside the margin but still correctly classified (on the correct side of the decision boundary)
- $\xi > 1$: the point is misclassified (on the wrong side of the dec. boundary)

Max 2 Iterations for MLR

Advantages

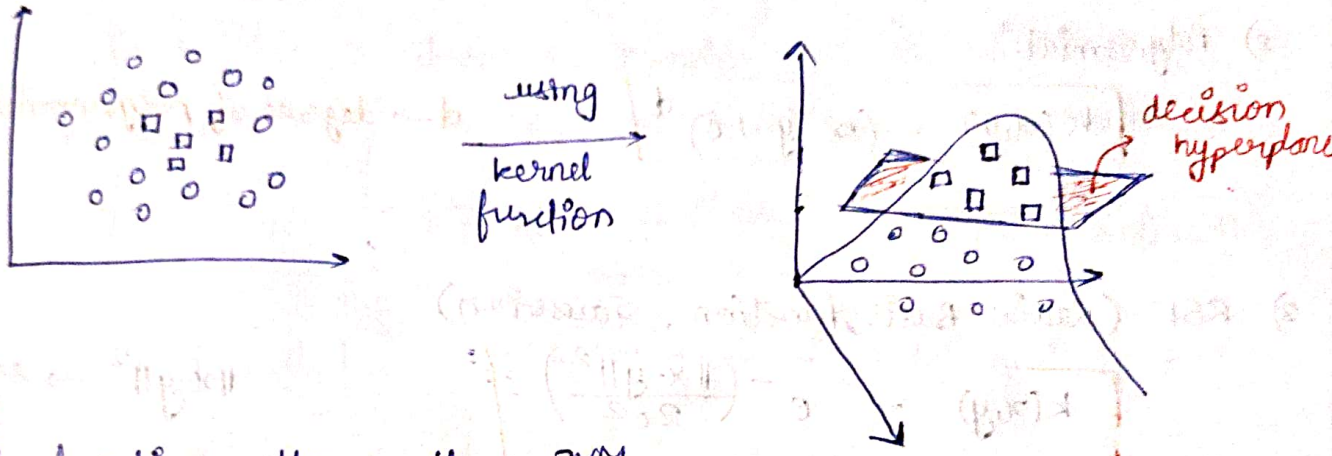
- Simple algo, less complex
- very few parameters to be considered
- does not require model training (Lazy Learner Algo)
- works well for small dataset

Disadvantages

- slow for large datasets
- choosing the right k is tricky.

Non-Linear SVM

When the datapoints are linearly not separable, we use non-linear SVM with kernel function to make it separable with a decision boundary/hyperplane.



- The kernel function allows the SVM to solve both linear and non-linear problems by mapping the data points into a higher dimensional space.

And then uses a decision hyperplane to separate them into classes.

→ different types of kernel functions:

- Linear Kernel F.
- Polynomial KF
- Radial Basis Function (RBF) – Gaussian Kernel
- Bessel Function
- Sigmoid KF

} Non-linear

1) Linear kernel

$$k(x,y) = x^T y$$

2) Polynomial

$$k(x,y) = (x^T y + c)^d$$

d → degree of polynomial

3) RBF (Radial Basis Function, Gaussian)

$$k(x,y) = e^{-\left(\frac{\|x-y\|^2}{2\sigma^2}\right)}$$

$\|x-y\|^2$ → sq. euclidean dist betn x & y vectors

σ → parameter controlling the spread

Eg: $x(1,2)$, $y(3,4)$ $\sigma = 1$

$$k(x,y) = e^{\frac{-8}{2(1)^2}} = e^{-4}$$

$$\|x-y\| = \sqrt{2^2+2^2} = \sqrt{8}$$

$$\Rightarrow k(x,y) = 0.018$$

4) Sigmoid

$$k(x,y) = \tanh(\alpha x^T y + c)$$

α → slope parameter
c → constant

Eg: $x(1,2)$ $y(3,4)$ $c=1$ $\alpha=0.5$

$$k(x,y) = \tanh(0.5(11)+1) = 0.99$$

Primal Problem

The primal problem is the direct formulation of the SVM optimization problem.

- It aims to find the optimal hyperplane $w^T x + b = 0$
  - maximizes margin
  - minimizes misclassifications

Dual Primal Problem

∵ primal problem involves inequality constraints, solving it directly is difficult.

So we convert it into a dual problem using Lagrange's multipliers.

$$L(w, w_0, \alpha) = \frac{1}{2}\|w\|^2 + \sum_{i=1}^{n} \alpha_i \left(1 - y_i(w^T x_i + w_0)\right) \quad ; \alpha_i \geq 0$$

↳ eq(1)

To solve, we find out derivatives wrt w & w_0

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{n} \alpha_i y_i x_i = 0$$

$$\Rightarrow \boxed{w = \sum_{i=1}^{n} \alpha_i y_i x_i}$$

$$\frac{\partial L}{\partial w_0} = -\sum_{i=1}^{n} \alpha_i y_i = 0$$

$$\Rightarrow \boxed{\sum_{i=1}^{n} \alpha_i y_i = 0}$$

$$\|w\|^2 = w^T w$$

$$\|w\|^2 = \left(\sum_{i=1}^{n} \alpha_i y_i x_i\right)^T \left(\sum \alpha_j y_j x_j\right)$$

$$= \sum_{j=1}^{n}\sum_{i=1}^{n} \alpha_i \alpha_j y_i y_j x_i x_j^T$$

Putting these values in eq①

~~$w^T w$~~

$$w^T x_i = \left(\sum \alpha_j y_j x_j\right)^T x_i$$

$$= \sum_{j=1}^{n} \alpha_j y_j x_j^T x_i$$

$$L(w, w_0, \alpha) = \frac{1}{2}\left(\sum\sum \alpha_i \alpha_j y_i y_j x_i x_j^T\right) + \sum \alpha_i - \sum\sum \alpha_i \alpha_j y_i y_j x_i x_j^T - \cancel{\sum \alpha_i y_i w_0}^{\,0}$$

$$L(w, w_0, \alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n} \alpha_i \alpha_j y_i y_j x_i x_j^T$$

maximize such that $\alpha_i \geq 0 \ \forall i$

& $\sum \alpha_i y_i = 0$

KKT conditions

For a solution to be optimal, the following KKT conditions:

① Stationarity ~~$\frac{\partial}{\partial w}(L(x, \lambda, \mu)) = 0$~~

$$\Rightarrow \frac{\partial L}{\partial w} = 0 \Rightarrow w = \sum_{i=1}^{n} \alpha_i y_i x_i$$

$$\Rightarrow \frac{\partial L}{\partial w_0} = 0 \Rightarrow \sum_{i=1}^{n} \alpha_i y_i = 0$$

② Primal Feasibility $y_i(w^T x_i + b) \geq 1 \ \forall i$

③ Dual Feasibility $\alpha_i \geq 0 \ \forall i$

④ ~~Universal~~ Complementary Slackness $\alpha_i [y_i (w^T x_i + b) - 1] = 0 \ \forall i$