Matrix Derivatives

- Let A be a $m \times n$ matrix and $f : \mathbb{R}^{m \times n} \to \mathbb{R}$ be a real function from m-by-n matrices, to a real number.

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & & \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

- Now, gradient of f with respect to A is also a $m \times n$ matrix and is given as below.

$$\nabla_A f_A = \begin{bmatrix} \frac{\partial f}{\partial a_{11}} & \frac{\partial f}{\partial a_{12}} & \cdots & \frac{\partial f}{\partial a_{1n}} \\ \frac{\partial f}{\partial a_{21}} & \frac{\partial f}{\partial a_{22}} & \cdots & \frac{\partial f}{\partial a_{2n}} \\ \vdots & & & \vdots \\ \frac{\partial f}{\partial a_{m1}} & \frac{\partial f}{\partial a_{m2}} & \cdots & \frac{\partial f}{\partial a_{mn}} \end{bmatrix}$$

gradient of A w.r.t. f.

Example:

Let $A = \begin{bmatrix} w & x \\ y & z \end{bmatrix}$ and $f(w, x, y, z) = 2w + 3x + yz$.

Find $\nabla_A f_A$.

Soln:

$$\begin{bmatrix} \frac{\partial f}{\partial w} & \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial} & \frac{\partial f}{\partial z} \end{bmatrix} = \begin{bmatrix} 2 & 3 \\ z & y \end{bmatrix}$$

#

Trace of Matrix

- For a $n \times n$ matrix A_1, trace of A is defined as sum of diagonal elements.

  i.e. $\text{tr } A = \sum_{i=1}^{n} a_{ii}$

- For a real number a $\text{tr}.a = a$.
- If multiplication of A and B is square matrix then

  $\text{tr } AB = \text{tr } BA$

Verify that $\text{tr}.AB = \text{tr}.BA$

Let A and B are matrices of order 2×3 and 3×2. respectively.

i.e.

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix} \qquad B = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \end{bmatrix}$$

Now,

$$AB = \begin{bmatrix} a_{11}\cdot b_{11} + a_{12}\cdot b_{21} + a_{13}\cdot b_{31} & a_{11}\cdot b_{12} + a_{12}\cdot b_{22} + a_{13}\cdot b_{32} \\ a_{21}\cdot b_{11} + a_{22}\cdot b_{21} + a_{23}\cdot b_{31} & a_{21}\cdot b_{12} + a_{22}\cdot b_{22} + a_{23}\cdot b_{32} \end{bmatrix}$$

$$\text{tr}.AB = a_{11}\cdot b_{11} + a_{12}\cdot b_{21} + a_{13}\cdot b_{31} + a_{21}\cdot b_{12} + a_{22}\cdot b_{22} + a_{23}\cdot b_{32}$$

Similarly

$$BA = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix}$$

$$= \begin{bmatrix} b_{11}\cdot a_{11} + b_{12}\cdot a_{21} & b_{11}\cdot a_{12} + b_{12}\cdot a_{22} & b_{11}\cdot a_{13} + b_{12}\cdot a_{23} \\ b_{21}\cdot a_{11} + b_{22}\cdot a_{21} & b_{21}\cdot a_{12} + b_{21}\cdot a_{22} & b_{21}\cdot a_{13} + b_{22}\cdot a_{23} \\ b_{31}\cdot a_{11} + b_{32}\cdot a_{21} & b_{31}\cdot a_{12} + b_{32}\cdot a_{22} & b_{31}\cdot a_{13} + b_{32}\cdot a_{23} \end{bmatrix}$$

$$\text{tr}.BA = \left[b_{11}\cdot a_{11} + b_{12}\cdot a_{21} + b_{21}\cdot a_{12} + b_{22}\cdot a_{22} + b_{31}\cdot a_{13} + b_{32}\cdot a_{23} \right]$$

$$= \left[a_{11}\cdot b_{11} + a_{12}\cdot b_{21} + a_{13}\cdot b_{31} + a_{21}\cdot b_{12} + a_{22}\cdot b_{22} + a_{23}\cdot b_{32} \right]$$

Thus,

Corollaries of tr. AB = tr. BA

tr. ABC = tr. CAB = tr. BCA

tr. ABCD = tr. DABC = tr. BCDA

tr. ABC =

Prove that : tr. CAB = tr. BCA

tr. ABC = tr. (AB)C

= tr. CAB

Similarly,

tr. ABC = tr. A(BC)

= tr BCA

Thus,

tr. ABC = tr. CAB = tr. BCA

proved :

Prove that : tr. ABCD = tr. DABC = tr. BCDA

⇒

If A and B are square matrices and a is a real number then trace operator exhibits following properties;

tr. A = tr. A^T

tr. ABCD = tr. (ABC) D

= tr. DABC

Similarly,

tr. ABCD = tr. A (BCD)

= tr. BCDA

Thus,

tr. ABCD = tr. DABC = tr. BCDA

proved ∴

If A and B are square matrices and a is a real number then trace operator exhibits following properties:

- $tr.A = tr.A^T$
- $tr.(A+B) = tr.A + tr.B$
- $tr.\ aA = a\ tr.A$

- $tr.A = tr.A^T$

Proof:

Let $A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & & \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$

$$tr.A = \sum_{i=1}^{n} a_{ii}$$

Again,

$A^T = \begin{bmatrix} a_{11} & a_{21} & \cdots & a_{n1} \\ a_{12} & a_{22} & \cdots & a_{n2} \\ \vdots & & & \\ a_{1n} & a_{2n} & \cdots & a_{nn} \end{bmatrix}$

$$tr.A^T = \sum_{i=1}^{n} a_{ii}$$

Thus, $tr.A = tr.A^T$ $\left(\text{i.e. } \sum_{i=1}^{n} a_{ii}\right)$

- $tr(A+B) = tr.A + tr.B$.

Let $A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & & \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$, $B = \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \\ \vdots & & & \\ b_{n1} & b_{n2} & \cdots & b_{nn} \end{bmatrix}$

Now,

$$A+B = \begin{bmatrix} a_{11}+b_{11} & a_{12}+b_{12} & \cdots & a_{1n}+b_{1n} \\ a_{21}+b_{21} & a_{22}+b_{22} & \cdots & a_{2n}+b_{2n} \\ \vdots & & & \\ a_{n1}+b_{n1} & a_{n2}+b_{n2} & \cdots & a_{nn}+b_{nn} \end{bmatrix}$$

$$tr.(A+B) = a_{11}+b_{11} + a_{22}+b_{22} + \cdots + a_{nn}+b_{nn}$$

$$tr.(A+B) = \sum_{i=1}^{n} a_{ii} + b_{ii}$$

Then,

$tr.A = a_{11} + a_{22} + \cdots + a_{nn}$ $\left(\sum_{i=1}^{n} a_{ii}\right)$

$tr.B = b_{11} + b_{22} + \cdots + b_{nn}$ $\left(\sum_{i=1}^{n} b_{ii}\right)$

$$tr.A + tr.B = (a_{11} + a_{22} + \cdots + a_{nn}) + (b_{11} + b_{22} + \cdots + b_{nn})$$
$$= a_{11} + b_{11} + a_{22} + b_{22} + \cdots + a_{nn} + b_{nn}$$

$$tr.A + tr.B = \sum_{i=1}^{n} a_{ii} + \sum_{i=1}^{n} b_{ii}$$

$$= \sum_{i=1}^{n} a_{ii} + b_{ii}$$

Thus,

$$tr.(A+B) = tr.A + tr.B \quad \#$$ $\left(\text{i.e. } \sum_{i=1}^{n} a_{ii} + b_{ii}\right)$

- $tr.\ aA = a\ tr.\ A$
- Let,

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

$$aA = \begin{bmatrix} a.a_{11} & a.a_{12} \\ a.a_{21} & a.a_{22} \end{bmatrix}$$

$$tr.\ aA = a.a_{11} + a.a_{22}$$

Now,

$$tr.\ A = a_{11} + a_{22}$$

$$a.\ tr.A = a.a_{11} + a.a_{22}$$

Thus,

$$tr.\ aA = a\ tr.A \quad \#$$

[i.e. $a.a_{11} + a.a_{22}$]

Some facts about matrix derives

- $\nabla_A\ tr.AB = B^T$
- $\nabla_{A^T}\ f(A) = (\nabla_A \cdot f(A))^T$
- $\nabla_A\ tr.ABA^TC = CAB + C^TAB^T$
- $\nabla_A\ |A| = |A|(A^{-1})^T$

Prove that $\nabla_A \text{tr.}\, AB = B^T$

Sol:

Let order of A is 3×2 and order of B is 2×3.

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix}, \quad B = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \end{bmatrix}$$

$$AB = \begin{bmatrix} a_{11}\cdot b_{11} + a_{12}\cdot b_{21} & a_{11}\cdot b_{12} + a_{12}\cdot b_{22} & a_{11}\cdot b_{13} + a_{12}\cdot b_{23} \\ a_{21}\cdot b_{11} + a_{22}\cdot b_{21} & a_{21}\cdot b_{12} + a_{22}\cdot b_{22} & a_{21}\cdot b_{13} + a_{22}\cdot b_{23} \\ a_{31}\cdot b_{11} + a_{32}\cdot b_{21} & a_{31}\cdot b_{12} + a_{32}\cdot b_{22} & a_{31}\cdot b_{13} + a_{32}\cdot b_{23} \end{bmatrix}$$

$$\text{tr.}\, AB = a_{11}\cdot b_{11} + a_{12}\cdot b_{21} + a_{21}\cdot b_{12} + a_{22}\cdot b_{22} + a_{31}\cdot b_{13} + a_{32}\cdot b_{23}$$

And,

$$\nabla_A \text{tr.}\, AB = \begin{bmatrix} b_{11} & b_{21} \\ b_{12} & b_{22} \\ b_{13} & b_{23} \end{bmatrix}$$

Now,

$$B^T = \begin{bmatrix} b_{11} & b_{21} \\ b_{12} & b_{22} \\ b_{13} & b_{23} \end{bmatrix}$$

Thus,

$$\nabla_A \text{tr.}\, AB = B^T$$

i.e. $\begin{bmatrix} b_{11} & b_{21} \\ b_{12} & b_{22} \\ b_{13} & b_{23} \end{bmatrix}$

Date 03/08

Deriving Normal Equations for Linear Regression

Given a training set, Let X be a $m \times n$ matrix that contains input values of training examples in its rows and let $\vec{y}$ be a m dimensional vector containing target values.

$$\begin{bmatrix} — (x^1)^T — \\ — (x^2)^T — \\ \vdots \\ — (x^m)^T — \end{bmatrix} \qquad \vec{y} = \begin{bmatrix} y^1 \\ y^2 \\ \vdots \\ y^m \end{bmatrix}$$

We know that,

$$y = f(x) = (x^i)^T w;$$

where, w is coefficient vector.

Now,

$$\text{error} = f(x) - \vec{y}$$
$$= xw - \vec{y}$$
$$= \begin{bmatrix} (x^1)^T w \\ (x^2)^T w \\ \vdots \\ (x^m)^T w \end{bmatrix} - \begin{bmatrix} y^1 \\ y^2 \\ \vdots \\ y^m \end{bmatrix}$$

Now,

Loss function can be written as:

$$L = \frac{1}{2} \sum_{i=1}^{m} \left(f(x^i) - y^i \right)^2$$
$$= \frac{1}{2} (xw - \vec{y})^T (xw - \vec{y})$$

To minimize loss, we need to calculate derivative / gradient of L w.r.t w.

$$\nabla_w L = \frac{1}{2} \nabla_w (Xw - \vec{y})^T (Xw - \vec{y})$$

$$= \frac{1}{2} \nabla_w (w^T X^T - \vec{y}^T)(Xw - \vec{y})$$

$$= \frac{1}{2} \nabla_w (w^T X^T Xw - w^T X^T \vec{y} - \vec{y}^T Xw + \vec{y}^T \vec{y})$$

$$= \frac{1}{2} \nabla_w \, tr\,(w^T X^T Xw - w^T X^T \vec{y} - \vec{y}^T Xw + \vec{y}^T \vec{y})$$

$\nabla_w \vec{y}^T \vec{y} = 0$

$tr\, A = tr\, A^T$

$\nabla_A tr\, AB = B^T$

$\nabla_A tr\, ABA^T C = B^T A^T C^T + BA^T C$

$tr(w^T X^T \vec{y}) = tr\,(w^T X^T \vec{y})^T$

$= tr\, \vec{y}^T Xw$

$$= \frac{1}{2} \nabla_w (tr\, w^T X^T Xw - 2\, tr\, \vec{y}^T Xw)$$

$$= \frac{1}{2} (X^T Xw + X^T Xw - 2X^T \vec{y})$$

$$= X^T Xw - X^T \vec{y}$$

At minima, gradient is always zero.

$\therefore X^T Xw - X^T \vec{y} = 0$

$\Rightarrow w = \frac{X^T \vec{y}}{X^T X} = (X^T X)^{-1} X^T \vec{y}$

This is called normal equation for linear regression.

Example:

Consider the following training data.

x_1	1	2	1	3
x_2	1	1	2	3
y	3	5	2	5

Use normal equation to find coefficient of linear regression line fitted through above data.

Soln:

$$X = \begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 1 & 2 \\ 3 & 3 \end{bmatrix}$$

Augment X for w_0.

$$X = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \\ 1 & 3 & 3 \end{bmatrix}$$

$X = 4 \times 3$

$X^T = 3 \times 4$

$(X^TX)^{-1} = 3 \times 3$

$(X^TX)^{-1} X^T = 3 \times 4$ (3×3 · 3×4)

$(X^TX)^{-1} X^T \vec{y} = 3 \times 1$ (3×4 · 4×1)

$$\vec{y} = \begin{bmatrix} 3 \\ 5 \\ 2 \\ 5 \end{bmatrix}$$

$$W = (X^TX)^{-1} X^T \vec{y}$$

$$= \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 1 & 3 \\ 1 & 1 & 2 & 3 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \\ 1 & 3 & 3 \end{bmatrix} \right)^{-1} \cdot \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 1 & 3 \\ 1 & 1 & 2 & 3 \end{bmatrix} \cdot \begin{bmatrix} 3 \\ 5 \\ 2 \\ 5 \end{bmatrix}$$

$$= \begin{bmatrix} 1+1+1+1 & 1+2+1+3 & 1+1+2+3 \\ 1+2+1+3 & 1+4+1+9 & 1+2+2+9 \\ 1+1+2+3 & 1+2+2+9 & 1+1+4+9 \end{bmatrix}^{-1} \cdot \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 1 & 3 \\ 1 & 1 & 2 & 3 \end{bmatrix} \cdot \begin{bmatrix} 3 \\ 5 \\ 2 \\ 5 \end{bmatrix}$$

$$= \begin{bmatrix} 4 & 7 & 7 \\ 7 & 15 & 14 \\ 7 & 14 & 15 \end{bmatrix}^{-1} \cdot \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 1 & 3 \\ 1 & 1 & 2 & 3 \end{bmatrix} \cdot \begin{bmatrix} 3 \\ 5 \\ 2 \\ 5 \end{bmatrix}$$

$$= \begin{bmatrix} 4 & 7 & 7 \\ 7 & 15 & 14 \\ 7 & 14 & 15 \end{bmatrix}^{-1} \cdot \begin{bmatrix} 3+5+2+5 \\ 3+10+2+15 \\ 3+5+4+15 \end{bmatrix}$$

$$= \begin{bmatrix} 4 & 7 & 7 \\ 7 & 15 & 14 \\ 7 & 14 & 15 \end{bmatrix}^{-1} \cdot \begin{bmatrix} 15 \\ 30 \\ 27 \end{bmatrix}$$

Parametric Vs. Non-parametric Algorithms

Parametric Algorithms | Models

- General form of learning model is assumed.
- Fixed number of parameters, we just need to adjust values of parameters.
- Fast compared to non-parametric.
- can show better performance with few training data.
- Example: Linear regression, locally weighted logistic regression.

General form of linear regression is:

$$y = w_0 + w_1 x^{(1)} + w_2 x^{(2)}$$

Non-parametric

- General form of mapping function is not assumed. Mapping function can take any form.
- Size of parameters may not be fixed.
- Slow compared to parametric algorithms.
- Need large volume of training data to show better performance.

Example: K-nearest neighbor (KNN) algorithm

Probabilistic Implementation of Least square Regression

In case of least square regression, input and output variables are related via following equation.

$$y = w^T x^{(i)} + \epsilon^{(i)}$$

where,

$\epsilon^{(i)}$ is error term that represents unmodeled effect and random noise.

Assume that $\epsilon^{(i)}$ follows IID (Independently and Identically Distributed). Thus, for zero mean σ^2 standard deviation, probability density function for gaussian distribution can be written as below:

$$P(\epsilon^{(i)}) = \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(\frac{-(\epsilon^{(i)})^2}{2\sigma^2}\right)$$

$$\Rightarrow P(y^i / x^i, w) = \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(\frac{-(\epsilon^{(i)})^2}{2\sigma^2}\right)$$

If we view above function as function of w, it is called likelihood function and is given as below:

$$L(w) = P(\vec{y} / | x, w)$$

$$= \prod_{i=1}^{m} P(y^i | x^i, w)$$

$$= \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(\frac{-(\epsilon^{(i)})^2}{2\sigma^2}\right)$$

$$= \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(\frac{-(y^{(i)} - w^T \cdot x^{(i)})^2}{2\sigma^2}\right)$$

Taking log on both sides;

$$\log(L(w)) = \log \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - w^T \cdot x^{(i)})^2}{2\sigma^2}\right)$$

$$l(w) = \sum_{i=1}^{m} \log\left\{\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - w^T \cdot x^{(i)})^2}{2\sigma^2}\right)\right\}$$

where,

$$l(w) = \log(L(w))$$

$$= \sum_{i=1}^{m} \log \frac{1}{\sqrt{2\pi}\sigma} + \sum_{i=1}^{m} \log\left(\exp\left(-\frac{(y^{(i)} - w^T \cdot x^{(i)})^2}{2\sigma^2}\right)\right)$$

$$= m \cdot \log \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{2\sigma^2} \sum_{i=1}^{m} (y^{(i)} - w^T \cdot x^{(i)})^2$$

Thus,

$$l(w) = m \cdot \log \frac{1}{\sqrt{2\pi} \cdot \sigma} - \frac{1}{\sigma^2} \cdot \frac{1}{2} \sum_{i=1}^{m} (y^{(i)} - w^T \cdot x^{(i)})^2 \quad \text{--- (1)}$$

From eqⁿ (1), we can say that likelihood is maximized when the term $\frac{1}{2} \sum_{i=1}^{m} (y^{(i)} - w^T \cdot x^{(i)})^2$ is minimized

Hence, we can say that maximizing likelihood is equivalent to minimizing squarred sum of error or maximizing likelihood is equivalent to least square regression.

Perceptron Learning Rule

- Perceptron is single neuron used for binary classification.
- It commonly classify linearly separable patterns.
- In case of logistic regression, output is predicted as below:

$$\hat{y} = g(y) = \frac{1}{1+e^{-y}} \quad \text{(logistic activation function)}$$

where,

$$y = w_0 + w_1 x^{(1)} + w_2 x^{(2)} + \dots$$

- If we use following function 'g', above equation works for perceptron.

$$g(y) = \begin{cases} 1 & \text{if } y \geq 0 \\ 0 & y < 0 \end{cases}$$

Thus, weight update rule for logistic regression is same as weight update rule for perceptron, which is given as below:

$$w_0 = w_0 + \alpha \frac{\partial L}{\partial w_0}$$

$$w_1 = w_1 + \alpha \frac{\partial L}{\partial w_1} x^{(1)}$$

$$w_2 = w_2 + \alpha \frac{\partial L}{\partial w_2} x^{(2)}$$

where,

$$\frac{\partial L}{\partial w_0} = \hat{y} - y$$

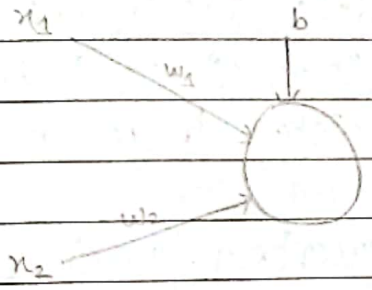
$$\frac{\partial L}{\partial w_1} = (\hat{y} - y) x^{(1)}$$

$$\frac{\partial L}{\partial w_2} = (\hat{y} - y) x^{(2)}$$

Now,

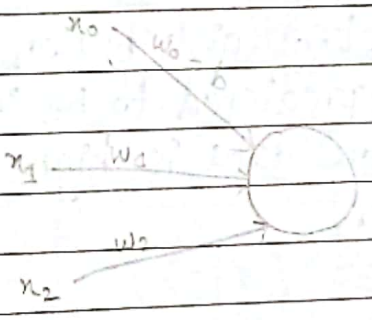
Generalized perceptron learning rule can be written as below:

$$w_i = w_i + \alpha(\hat{y} - y)\, x^{(i)}$$



$$v = w_1 x_1 + w_2 x_2 + b$$

$$y = f(v)$$



$$v = \underbrace{w_0 x_0}_{b} + w_1 x_1 + w_2 x_2$$

Gaussian Discriminant Analysis (GDA)

There are two broad categories of supervised classification algorithms:

- Discriminative Learning Algorithms
- Generative Learning Algorithms

⇒ Discriminative learning algorithms finds decision between classes. For a new data, if the data lies in one side of the decision boundary then it is predicted to be in one class otherwise it is predicted to be in another class.

Example: Logistic regression, perceptron learning, etc.

⇒ Generative learning algorithms capture distribution of each class. For a new data, it is predicted to be in the class for which resemblance of the data is higher.

Example: Naive Baye's Classifier.

Naive Baye's Classifier

- also called Bayesian classifier
- It is based on Baye's theorem, which is given as below:

$$P(H|X) = \frac{P(X|H) \cdot P(H)}{P(X)}$$

$P(X|H)$: likelihood

$P(H|X)$: posterior probability

$P(H)$, $P(X)$: prior probabilities

- Let D be the dataset and $C_1, C_2, \ldots, C_m$ are m classes. For this, Baye's rule can be written as below:

$$P(C_i|X) = \frac{P(X|C_i) \cdot P(C_i)}{P(X)}$$

where,

X is a tuple containing n attributes

i.e. $X = \{x_1, x_2, \ldots, x_n\}$

- Let us assume the input attributes are independent of each other. Now, the probability $P(X|C_i)$ can be written below:

$$P(X|C_i) = P(x_1|C_i) \times P(x_2|C_i) \times \ldots \times P(x_n|C_i)$$

$$= \prod_{k=1}^{n} P(x_k|C_i)$$

$$\Rightarrow \boxed{P(C_i|X) = \frac{P(C_i) \times \prod_{k=1}^{n} P(x_k|C_i)}{P(X)}}$$

In the above equation, denominator is same for all classes. Therefore, it can be ignored.

Finally, while making prediction, X is predicted to be in class C_i for which $P(C_i|X)$ is maximum.

MA = Middle Aged

Example:

Age	Income	Student	Credit Rating	Buys Computer()
Youth	High	No	Fair	No
Youth	High	No	Excellent	No
MA	High	No	Fair	Yes
Senior	Medium	No	Fair	Yes
Senior	Low	Yes	Fair	Yes
Senior	Low	Yes	Excellent	No
MA	Low	Yes	Excellent	Yes
Youth	Medium	No	Fair	No
Youth	Low	Yes	Fair	Yes
Senior	Medium	Yes	Fair	Yes
Youth	Medium	Yes	Excellent	Yes
MA	Medium	No	Excellent	Yes
MA	High	Yes	Fair	Yes
Senior	Medium	No	Excellent	No

Predict class label for X = {Age = Youth, Income = medium, Student = yes, Credit Rating = Fair}.

Sol:

$P(Buys = Yes) = 9/14$

$P(Buys = No) = 5/14$

For Age:

$P(Age = Youth \mid Buys = Yes) = 2/9$

$P(Age = Youth \mid Buys = No) = 3/5$

For Income:

$P(Income = Medium \mid Buys = Yes) = 4/9$

$P(Income = Medium \mid Buys = No) = 2/5$

For Student:

$P(\text{Student} = \text{Yes} \mid \text{Buys} = \text{Yes}) = 6/9$

$P(\text{Student} = \text{Yes} \mid \text{Buys} = \text{No}) = 1/5$

For Credit Rating:

$P(\text{Credit Rating} = \text{Fair} \mid \text{Buys} = \text{Yes}) = 6/9$

$P(\text{Credit Rating} = \text{Fair} \mid \text{Buys} = \text{No}) = 2/5$

Now,

$$P(\text{Buys} = \text{Yes} \mid X) = P(\text{Buys} = \text{Yes}) \times \prod_{k=1}^{n} P(x_k \mid \text{Buys} = \text{Yes})$$

$$= P(\text{Buys} = \text{Yes}) \times P(\text{Age} = \text{Youth} \mid \text{Buys} = \text{Yes}) \times P(\text{Income} = \text{Medium} \mid \text{Buys} = \text{Yes}) \times P(\text{Student} = \text{Yes} \mid \text{Buys} = \text{Yes}) \times P(\text{Credit Rating} = \text{Fair} \mid \text{Buys} = \text{Yes})$$

$$= \frac{9}{14} \times \frac{2}{9} \times \frac{4}{9} \times \frac{6}{9} \times \frac{6}{9}$$

$$= 0.028$$

Similarly,

$$P(\text{Buys} = \text{No} \mid X) = P(\text{Buys} = \text{No}) \times \prod_{k=1}^{n} P(x_k \mid \text{Buys} = \text{No})$$

$$= P(\text{Buys} = \text{No}) \times P(\text{Age} = \text{Youth} \mid \text{Buys} = \text{No}) \times P(\text{Income} = \text{Medium} \mid \text{Buys} = \text{No}) \times P(\text{Student} = \text{Yes} \mid \text{Buys} = \text{No}) \times P(\text{Credit Rating} = \text{Fair} \mid \text{Buys} = \text{No})$$

$$= \frac{5}{14} \times \frac{3}{5} \times \frac{2}{5} \times \frac{1}{5} \times \frac{2}{5}$$

$$= 0.007$$

Since, $P(\text{Buys} = \text{Yes} \mid X) > P(\text{Buys} = \text{No} \mid X)$;

Predicted class ~~for~~ label for the given data is Buys Computer = Yes.

Variants of Naive Baye's Classifier

1. Gaussian Naive Bayes

- Gaussian Naive Bayes is used when features of data set have gaussian distribution.
- If features of dataset have continuous values, they follow gaussian distribution.
- In this case, probability of feature x_i belonging to class C is estimated using gaussian probability distribution function as given below:

$$P(x_i|C) = \frac{1}{\sqrt{2\pi}\,\sigma_c} \exp\left(-\frac{(x_i-\mu_c)^2}{2\sigma_c^2}\right)$$

where,

μ_c and σ_c are mean and standard deviation of the feature belonging to class C.

2. Multinomial Naive Bayes

- It is used when input features follows multinomial distribution. It is primarily used in text classifications.
- It calculates probability of feature x_i belonging to class C as below:

$$P(x_i|C) = \frac{N_{ic} + \alpha}{N_c + \alpha \times n}$$

where,

N_{ic} is no. of occurrences of feature x_i in class C.
N_c is the no. of samples belonging to class C.
α is smoothing factor (Normally $\alpha = 1$ is used).
n is no. of dimensions.

3. Bernoulli Naive Bayes

- It is used when input features follows multivariate Bernoulli distribution. When input features of dataset are binary-value, they follow Bernoulli distribution.

- It calculates probability of feature x_i belonging to class C as below:

$$P(x_i|C) = P(x_i|C)x_i + (1 - P(x_i|C))(1 - x_i)$$

Laplace Smoothing

- It is the technique used to handle zero probability in Naive Bayes classifier.
- It estimates probability of feature x_i belonging to class C as below: (Case of multinomial Naive Bayes)

$$P(x_i|C) = \frac{N_{ic} + \alpha}{N_c + \alpha \times n}$$

where,

N_{ic} is no. of occurrences of feature x_i in class C.
N_c is the no. of samples belonging to class C.
α is smoothing factor (Normally $\alpha = 1$ is used).
n is no. of dimensions.

Example:

Consider the following dataset.

Email	Class
Send us your password	Spam
Send your account	Spam
Review your account	Ham
Review your password	Ham
Review our profile	Spam
Send us money	Spam
Review your profile	?

Sol:

Assume $\alpha = 1$.

Vocabulary = {send, us, your, password, account, review, our, profile, money}

$P(\text{Spam}) = 4/6$

$P(\text{Ham}) = 2/6$

Send: $P(\text{send} \mid \text{Spam}) = \frac{3+1}{4+1\times 9} = \frac{4}{13}$

$P(\text{send} \mid \text{Ham}) = \frac{0+1}{2+1\times 9} = \frac{1}{11}$

us: $P(\text{us} \mid \text{Spam}) = \frac{2+1}{4+1\times 9} = \frac{3}{13}$

$P(\text{us} \mid \text{Ham}) = \frac{0+1}{2+1\times 9} = \frac{1}{11}$

your: $P(\text{your} \mid \text{Spam}) = \frac{2+1}{4+1\times 9} = \frac{3}{13}$

$P(\text{your} \mid \text{Ham}) = \frac{2+1}{2+1\times 9} = \frac{3}{11}$

Password: $P(\text{Password} \mid \text{Spam}) = \frac{1+1}{4+1\times9} = \frac{2}{13}$

$P(\text{Password} \mid \text{Ham}) = \frac{1+1}{2+1\times9} = \frac{2}{11}$

account: $P(\text{account} \mid \text{Spam}) = \frac{1+1}{4+1\times9} = \frac{2}{13}$

$P(\text{account} \mid \text{Ham}) = \frac{1+1}{2+1\times9} = \frac{2}{11}$

review: $P(\text{review} \mid \text{Spam}) = \frac{1+1}{4+1\times9} = \frac{2}{13}$

$P(\text{review} \mid \text{Ham}) = \frac{2+1}{2+1\times9} = \frac{3}{11}$

our: $P(\text{our} \mid \text{Spam}) = \frac{1+1}{4+1\times9} = \frac{2}{13}$

$P(\text{our} \mid \text{Ham}) = \frac{0+1}{2+1\times9} = \frac{1}{11}$

Profile: $P(\text{Profile} \mid \text{Spam}) = \frac{1+1}{4+1\times9} = \frac{2}{13}$

$P(\text{Profile} \mid \text{Ham}) = \frac{0+1}{2+1\times9} = \frac{1}{11}$

money: $P(\text{money} \mid \text{Spam}) = \frac{1+1}{4+1\times9} = \frac{2}{13}$

$P(\text{money} \mid \text{Ham}) = \frac{0+1}{2+1\times9} = \frac{1}{11}$

Now,

$P(\text{Spam} \mid \text{"Review your profile"}) = P(\text{Spam} \mid [0,0,1,0,0,1,0,1,0])$

$= P([0,0,1,0,0,1,0,1,0] \mid \text{Spam}) \times P(\text{Spam})$

$= \left(1-\frac{4}{13}\right) \times \left(1-\frac{3}{13}\right) \times \frac{3}{13} \times \left(1-\frac{2}{13}\right) \times \left(1-\frac{2}{13}\right) \times \frac{2}{13} \times \left(1-\frac{2}{13}\right) \times \frac{2}{13} \times \left(1-\frac{2}{13}\right) \times \frac{4}{6}$

$= 0.000994$

$P(\text{Ham} \mid \text{"Review your profile"}) = P(\text{Ham} \mid [0,0,1,0,0,1,0,1,0])$

$= P([0,0,1,0,0,1,0,1,0] \mid \text{Ham}) \times P(\text{Ham})$

$= \left(1-\frac{1}{11}\right) \times \left(1-\frac{1}{11}\right) \times \frac{3}{11} \times \left(1-\frac{2}{11}\right) \times \left(1-\frac{2}{11}\right) \times \frac{3}{11} \times \left(1-\frac{1}{11}\right) \times \frac{1}{11} \times \left(1-\frac{1}{11}\right) \times \frac{2}{6}$

$= 0.00103$

Since, $P(\text{Ham} \mid \text{"Review your profile"}) > P(\text{Spam} \mid \text{"Review your profile"})$.

Predicted class label for the given email message is "Ham".

Decision Tree Classifier

- Decision tree is an algorithm that constructs model in the form of decision tree from training data.
- Decision tree is the tree structured model where internal nodes represents test on attribute, edges represents test outcomes, and leaf nodes represent class label.
- At the time of prediction, we just need to trace path from root node to leaf node on the basis of attribute values given in the tuple. Class label of the leaf node will be predicted class label for the given tuple.

Outline of Decision tree Algorithm

1. Initially, all training tuples are considered in root node.
2. Partition the tuples recursively on the basis of selected attribute.
3. If all samples of the given node belongs to same class,
   - Label the class.
4. Else if there are no remaining attributes for further partitioning
   - Use majority voting to label the class
5. Else
   - Go to step 2.

- There are many variations of decision tree. Some of them are : ID3, C 4.5, CART, etc.
- There are different attribute selection measure used by various decision tree algorithm. Some of the attribute selection measures are : Information Gain, Gain Ratio, Gini Index, etc.

ID3 Algorithm

- ID3 stands for iterative dichotomiser - 3. It uses information gain as attribute selection measure. Information gain is calculated from entropy.

- Entropy is the measure of homogeniety of data sample and is calculated as below:

$$E(\mathcal{D}) = \sum_{i=1}^{m} -P_i \log_2 P_i$$ (before partitioning)

where,

m is no. of classes

P_i is probability of tuple in $\mathcal{D}$ belonging to class C_i.

- P_i can be calculated as below:

$$\frac{|C_i, \mathcal{D}|}{|\mathcal{D}|}$$

where,

$|C_i, \mathcal{D}|$ is no. of tuple belonging to class C_i and $|\mathcal{D}|$ is total no. of tuples in data set.

- Suppose dataset $\mathcal{D}$ is partitioned on the basis of attribute A having v distinct values. Thus, V partition are created $\{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_v\}$. Now total entropy of the dataset after partitioning on A can be calculated as below:

$$E_A(\mathcal{D}) = \sum_{i=1}^{v} \frac{|\mathcal{D}_i|}{|\mathcal{D}|} E(\mathcal{D}_i)$$

- Information gain is reduction in entropy after partitioning. Thus, information after partitioning on A can be calculated as below.

$$\text{Gain}(A) = E(\theta) - E_A(\theta)$$

- ID3 selects attribute with highest information for partitioning.

Example: (eg. of Naive Baye's)

Soln:

Entropy of the dataset before partitioning is given as below:

$$E(\theta) = -\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14}$$

$$= 0.94$$

Entropy of the dataset after partitioning on "Age".

$$E_{Age}(\theta) = \frac{5}{14}\left\{-\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5}\right\} +$$

$$\frac{4}{14}\left\{-\frac{4}{4}\log_2\frac{4}{4} - \frac{0}{4}\log_2\frac{0}{4}\right\} +$$

$$\frac{5}{14}\left\{-\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5}\right\}$$

$$= 0.694$$

Thus, Information Gain after partitioning on Age is

$$\text{Gain}(Age) = E(\theta) - E_{Age}(\theta)$$

$$= 0.246$$

Similarly,

Gain (Income) = 0.029

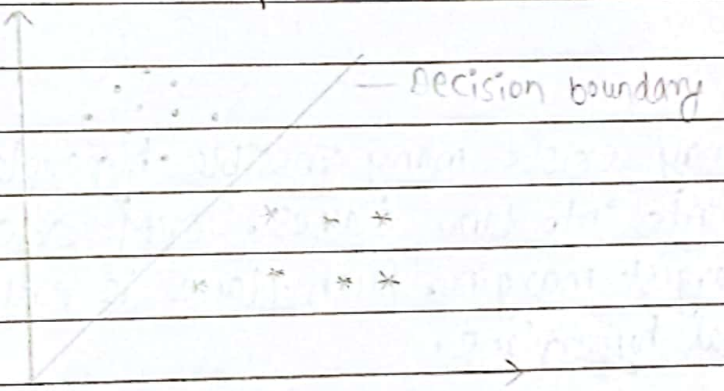
Gain (Student) = 0.151

Gain (Credit Rating) = 0.048

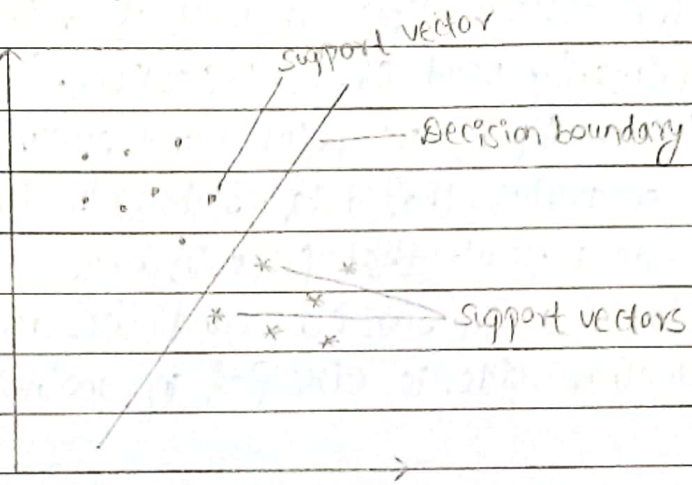
Since, information gain of age is highest, it is selected as partitioning attribute and the decision tree looks like below.

Support Vector Machine (SVM)

- SVM is the supervised learning algorithm that can be use for classification as well as regression. However, it is primarily used for classification.
- It takes input data points and outputs a hyperplane that best separates the data points into two classes.
- Any data point that falls in one side of the hyperplane is classified in one class and the data points that falls in another side is classified in another class.



- Support vectors are the data points that are closest to the decision boundary.



- There may exists many possible hyperplane that divides data points into two classes. SVM selects the plane with largest margin. Such plane is called maximal marginal hyperplane.

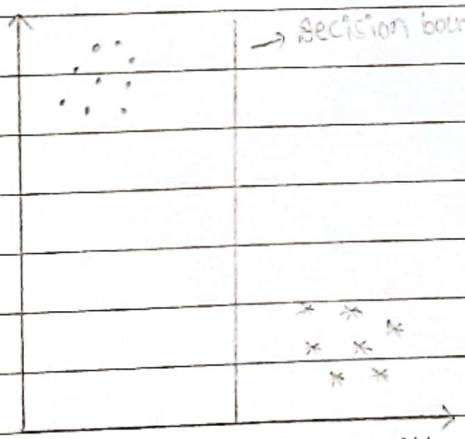


Fig: Decision boundary with small margin

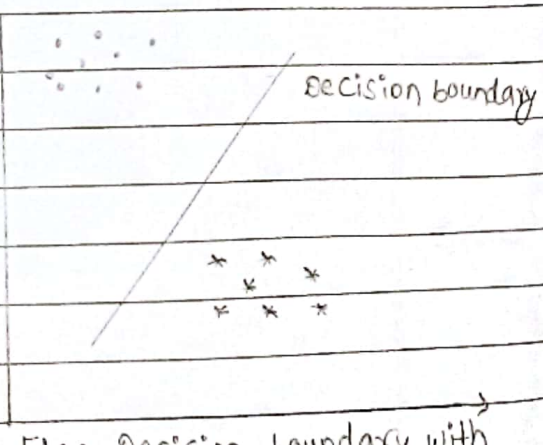


Fig: Decision boundary with large margin

- Let H_1 and H_2 are two hyperplanes that passes through support vectors and parallel to the hyperplane of decision boundary. Margin is the distance H_1 and H_2 such that distance between H_1 and decision boundary must be equal to the distance between H_2 and the decision boundary.