@mla unit-2-3

unit-2

# Unit 2: Dimensionality Reduction

*By Prof. Ranjitha U.N*

Introduction

In machine learning as the dimensionality of the data rises, the amount of data required to provide a reliable analysis grows exponentially. Bellman referred to this phenomenon as the "curse of dimensionality"

Cases:

1. A motorbike rider in racing competitions. His position and movement gets measured by GPS sensor on bike, gyro meters, multiple video feeds and his smart watch. Because of respective errors in recording, the data would not be exactly same. However, there is very little incremental information on position gained from putting these additional sources.

Now assume that an analyst sits with all this data to analyze the racing strategy of the biker – he/ she would have a lot of variables / dimensions which are similar and of little (or no) incremental value. This is the problem of high unwanted dimensions and needs a treatment of dimension reduction

2. **Image processing**. Facebook application – "Which Celebrity Do You Look Like?".

To identify the matched celebrity image, we use pixel data and each pixel is equivalent to one dimension. In every image, there are high number of pixels i.e. high number of dimensions. And every dimension is important here. You can't omit dimensions randomly to make better sense of your overall data set. In such cases, dimension reduction techniques help you to find the significant dimension(s) using various method(s).

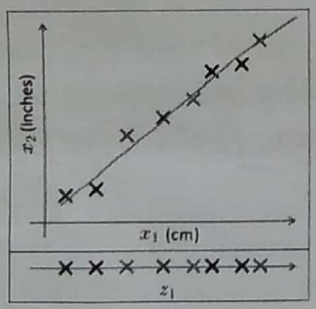
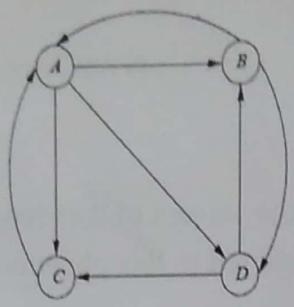



*Figure 1: 2D converted to one dimension*

3.There are many sources of data that can be viewed as a large matrix.Web can be represented as a transition matrix.




$$M = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 1/2 & 1 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix} \end{array}$$

Fig: Hypothetical example of web

*While Google was not the first search engine, it was the first, able to defeat the spammers, PageRank was established as an essential technique for a search engine to defeat link spam*

Social network can also be represented as a transition matrix.

In many of these matrix applications, the matrix can be summarized by finding "narrower" matrices (linear algebra,dot product and eigen values) that in some sense are close to the original. These narrow matrices have only a small number of rows or a small number of columns, and therefore can be used much more efficiently than can the original large matrix. The process of finding these narrow matrices is called dimensionality reduction.

**Definition** : Dimension Reduction refers to the process of converting a set of data having vast dimensions into data with lesser dimensions ensuring that it conveys similar information concisely inturn to minimize the complexity of any classifier or regressor

**Reason /objective**:

1. In most learning algorithms, the complexity depends on the number of input dimensions, d, as well as on the size of the data sample, N, and for reduced memory and computation, we are interested in reducing the dimensionality of the problem. Decreasing d also decreases the complexity of the inference algorithm during testing.

2. When an input is decided to be unnecessary, we save the cost of extracting it.

3. Simpler models are more robust on small datasets (exhibit less variance)

4. When data can be explained with fewer features, we get a better idea about the process that underlies the data and this allows knowledge extraction.

5. When data can be represented in a few dimensions without loss of information, it can be plotted and analyzed visually for structure and outliers

**Methods**: There are two main methods for reducing dimensionality: feature selection and feature extraction.(can be supervised or unsupervised)

In feature selection, finding k of the d dimensions that give us the most information and we discard the other (d − k) dimensions. Ex. Random forest , decision tree

In feature extraction, we are interested in finding a new set of k dimensions that are combinations of the original d dimensions ex: Principal Components Analysis (PCA) and Linear Discriminant Analysis (LDA),

## Subset selection (feature subset selection):

In subset selection, we are interested in finding the best subset of the set of features. The best subset contains the least number of dimensions that most contribute to accuracy .

For d dimesions ,there are $2^d$ possible subsets of d variables, but we cannot test for all of them.

There are two 2 approaches in selecting the subset:

Forward selection
Backward selection

*Forward selection*: - we start with no variable and add them one by one, at each step adding the one that decreases the error the most, until any further addition does not decrease the error

Sequential forward selection (algorithm)

E(F) ⇒ error on {F}
misclassification error

*Start with no features: $F = \emptyset$. F, is a feature set of dimension $x_i, i = 1 \dots d$

*At each step, for all possible $x_i$ , train our model on the training set and calculate $E(F \cup x_i)$ on the validation set.

* choose that input $x_j$ that causes the least error

*$j = \arg\min_i E(F \cup x_i)$ and add $x_j$ to F if $E(F \cup x_j) < E(F)$

*Drawback*: because this algorithm is greedy and adds attributes one by one, it may not be able to detect the effectiveness of $x_i$ and $x_j$ , as both together can reduce error

*Backward selection*: - we start with all variables and remove them one by one, at each step removing the one that decreases the error the most, until any further removal increases the error significantly.



Forward selection iris data set 4 attribs 3 classes.
(11) classifier Nearest mean → 1d 4 features accuracy 0.76, 0.52, 0.92 & 0.94
0.87 F1, F4
0.92 F3, F4
in 3d also 0.94
0.96 F2, F4
$d + (d-1) + (d-2) \dots + (d-k)$
$\Theta(d^2)$

Let F ---- feature set of input dimensions, $x_i$, i = 1. . . d
E (F) ----- denotes the error incurred on the validation sample (distinct from training sample)
the error is either the mean square error or misclassification error

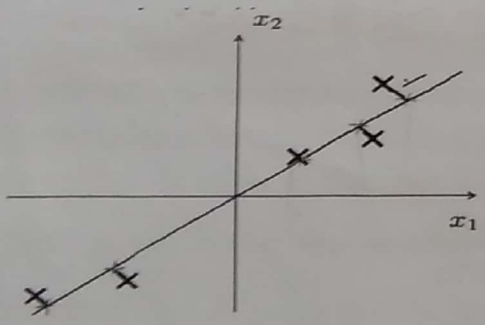
Sequential backward selection
*Start with F containing all features
*remove one attribute from F as the one that causes the least error i.e.
$j_i = \arg\min_i E(F - x_i)$
and we remove $x_j$ from F if $E(F - x_j) < E(F)$

Drawback: training a system with more features is more costly than training a system with fewer features .

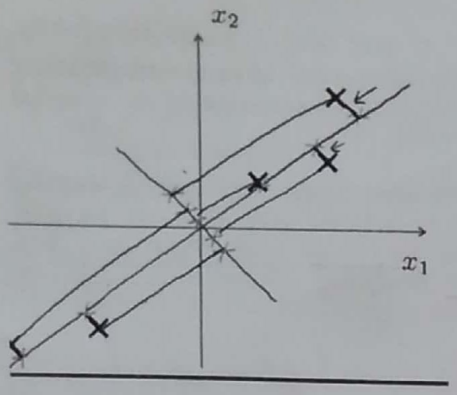
Note: forward search may be preferable especially if we expect many useless features.
In supervised learning, output is considered to calculate the error during subset selection.

## Feature Extraction:

In applications like face recognition, removal of feature (pixel), may lead to loss of information as group of pixels together carry information. Such cases deal with feature extraction than feature removal.
PCA is an unsupervised feature extraction algorithm




Reduce 2 dimension to one dimension (similary n dimension to k dimension).
To do so find a direction 9vector on to which the data is projected so as to minimize the projection error (blue lines in fig)
Note: PCA is not linear regression,




The criterion in PCA is to maximize variance .(the line in red is better when compared to magenta line which has more projection error well as less variance.
It is always better to maintain good distance between the points i.e. difference between the sample points becomes most apparent

In general the projection of x on the direction of w is
$z = w^T x$

## Factor analysis

Preread : There has been significant controversy in the field over differences between the two techniques ,exploratory factor analysis and principal components analysis ). PCA can be considered as a more basic version of exploratory factor analysis (EFA) that was developed in the early days prior to the advent of high-speed computers.

Both PCA and factor analysis aim to reduce the dimensionality of a set of data.

**Factor analysis** :- is clearly designed with the objective to identify certain unobservable factors from the observed variables, whereas PCA does not directly address this objective

There are two uses of factor analysis:
① It can be used for knowledge extraction , → grouping of highly correlated variables under one factor and others in other factor through subset creation
② It can also be used for dimensionality reduction when $k < d$
FOR DIMIENSIONALITY REDUCTION

In factor analysis (FA), we assume that there is a set of unobservable, latent factors $z_j$ , $j = 1, \ldots, k$, which when acting in combination generate x.

When we are interested in dimensionality reduction, we need to be able to find the factor scores, $z_j$, from $x_i$ . We want to find the loadings $w_{ji}$

*Explanation of factor loading*



Note: Large no. of variables into fewer no. of factors.

unobservable
*Latent factors $z_j$, $j = 1 \ldots K$ → Latent variables acting in combination generate $\underline{x}$ (i/p)

| Variables | Factor 1 |
|---|---|
| Income | 0.65 |
| Education | 0.59 |
| Occupation | 0.48 |
| House value | 0.38 |
| Number of public parks in neighborhood | 0.13 |
| Number of violent crimes per year in neighborhood | 0.23 |

*Note: The relationship of each variable to the underlying factor is expressed by the so-called factor loading.*

*The correlation between the variable and the factor is given by the score. Hence the FA deals with the correlation where PCA deals with variance .Here, latent variable is socio-economic status*

We want to find the loadings $w_{ji}$, such that

$$z_j = \sum_{i=1}^{d} w_{ji} x_i + \epsilon_i, j = 1, \ldots, k$$ (Eq. 1), where xi are centered to have mean 0

→ without loss of generality

In vector form, for observation T , it can be written as

$$\mathbf{z}^t = \mathbf{W}^T \mathbf{x}^t + \epsilon, \forall t = 1, \ldots, N$$

This is a linear model with $d$ inputs and $k$ outputs. Its transpose can be written as

$$(\mathbf{z}^t)^T = (\mathbf{x}^t)^T \mathbf{W} + \epsilon^T, \forall t = 1, \ldots, N$$

Note: X is N×d data matrix
N → no. of instances
d → i/p dimensionality

For a sample of N observations ,we have

$$\mathbf{Z} = \mathbf{X}\mathbf{W} + \Xi$$ — (2) $\Xi$ is $N \times k$ of zero-mean noise.

W can be found as

$$\begin{aligned} \mathbf{W} &= (N-1)(\mathbf{X}^T\mathbf{X})^{-1}\frac{\mathbf{X}^T\mathbf{Z}}{N-1} \\ &= \left(\frac{\mathbf{X}^T\mathbf{X}}{N-1}\right)^{-1}\frac{\mathbf{X}^T\mathbf{Z}}{N-1} \\ &= \mathbf{S}^{-1}\mathbf{V} \end{aligned}$$ Substituting the values for W

— (3)

$$\mathbf{Z} = \mathbf{X}\mathbf{W} = \mathbf{X}\mathbf{S}^{-1}\mathbf{V}$$

*Comparison between PCA and FA*

FA offers no advantage over PCA except the interpretability of factors allowing the identification of common causes, a simple explanation, and knowledge extraction. For example, in the context of speech recognition, x corresponds to the acoustic signal,

articulators, namely, jaw, tongue, velum, lips, and mouth, which are positioned appropriately to shape the air as it comes out of the lungs and generate the speech sound.

If speech signal could be transformed to these attributes, then recognising would be easy.

**Linear Discriminant Analysis**

Linear Discriminant Analysis is a supervised method for dimensionality reduction for classification problems. Now considering two classes, then generalizing to K >2 classes.

* considering samples from two classes C1 and C2, a new direction defined by a vector $w$ is found such that when data are projected onto $w$, the examples from the two classes are well separated.

$z = w^T x \Rightarrow$ projection of $x$ onto $w$. [$\Rightarrow$ D Reduction from d to 1]

* $m_1 \rightarrow$ means of sample from C1 before projection

$m_1' \rightarrow$ " " " C1 after projection

$m_1 \in R^d$ & $m_1' \in R$

* For a given sample $X = \{x^t, r^t\}$ (one example)

$r^t = 1$ if $x^t \in C_1$ ; $r^t = 0$ if $x^t \in C_2$

$$m_1' = \frac{\sum_t w^T x^t r^t}{\sum_t r^t} = w^T m_1$$

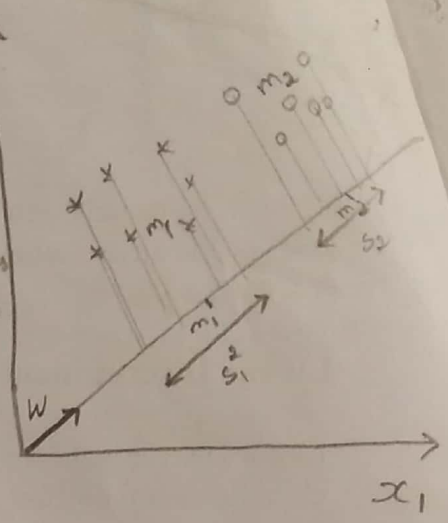
$$m_2' = \frac{\sum_t w^T x^t (1 - r^t)}{\sum_t (1 - r^t)} = w^T m_2$$

The scatter of samples from $C_1$ and $C_2$ after projection are

$$s_1^2 = \sum_t \left(w^T x^t - m_1'\right)^2 r^t$$

$$s_2^2 = \sum_t \left(w^T x^t - m_2'\right)^2 (1 - r^t)$$

scatter = variance = sum of square differences b/w projected samples and their class mean.




For the two classes to be well seperated, the means need to be as far as possible and the examples of classes be scattered in as small region as possible.

ie (from fig) we need $|m_1' - m_2'|$ to be large and $s_1^2 + s_2^2$ to be small $\Rightarrow$ (Fisher's linear discriminant is $w$ that maximizes below eq)

$$J(w) = \frac{(m_1' - m_2')^2}{s_1^2 + s_2^2} \quad \text{——①}$$

Rewriting the numerator, we get

$$\begin{aligned}(m_1' - m_2')^2 &= (w^T m_1 - w^T m_2)^2 \\ &= w^T \underbrace{(m_1 - m_2)(m_1 - m_2)^T}_{S_B} w \\ &= w^T S_B w\end{aligned}$$

Where, $S_B = (m_1 - m_2)(m_1 - m_2)^T$ is the between-class scatter matrix.

The denominator of eq ① (which the sum of scatter) can be written as

Note:-
scatter matrix is a statistic used to make a estimate of covariance matrix

$(w^T m_1 - w^T m_2)($

$m_1$

$$s_1^2 = \sum_t (w^T x^t - m_1')^2 r_t^t$$

$$= \sum_t \{ w^T (x^t - m_1)(x^t - m_1)^T w r^t$$

$$= w^T S_1 w$$

where $S_1 = \sum_t (x^t - m_1) r^t (x^t - m_1)^T$ is the within class scatter matrix

Applying the same (above) analysis to $S_2$

we have $s_1^2 + s_2^2 = w^T S_w w$. $S_w = s_1^2 + s_2^2$

↓

Total within class scatter matrix.

Note: $\frac{s_1^2 + s_2^2}{\text{Total no. of samples}}$ = Variance of pooled data

When $K > 2$ classes

matrix $w$ needs to be found such that

$z = w^T x$. $z \rightarrow K$ dimensional

$w \rightarrow d \times K$

within-class matrix for $c_i$ is

$$S_i = \sum_t r_i^t (x^t - m_i)(x^t - m_i)^T$$

→

Total within-class scatter is

$$S_w = \sum_{i=1}^{K} S_i$$

The between-class scatter matrix is

→ $S_B = \sum_{i=1}^{K} N_i (m_i - m)(m_i - m)^T$ $N_i = \sum_t r_i^t$.

For the analysis, we need

1. the between-class scatter matrix $w^T S_B w$ to be large
2. the within-class scatter matrix $w^T S_w w$ to be small

**Note:** For a scatter matrix,

Measure of the spread = determinant of the matrix

Determinant of a matrix = product of eigenvalues

and

Eigenvalues gives variance along its eigenvector

∴ The interest is in the matrix $w$ that maximizes

$$J(w) = \left| \frac{w^T S_B w}{w^T S_w w} \right|$$

The largest eigenvectors of $S_w^{-1} S_B$ are the solution

## Multidimensional scaling

- It is also a dimentionality reduction algorithm like PCA and FA,but the approach taken for dim reduction varies.
- It focusses on the distance between the related item (data point)and not on their spatial co-ordinates(positions).
- Implementation of MDS, can be
  i) Metric ie Classical,Multidimensional Scaling (CMDS) because it tries to reproduce the original metric or distances when the data is in metric form
  ii) Non-Metric Multidimensional Scaling (NMMDS), this method produces a map which tries to reproduce ranks when the data is in the form of ranking or ordinal data

Consider the following problem: looking at a map showing a number of cities, one is interested in the distances between them. These distances are easily obtained by measuring them using a ruler. Apart from that, a mathematical solution is available:knowing the coordinates x and y , the Euclidean distance between two cities a and b is defined by

$$d_{ab} = \sqrt{(x_a - x_b)^2 + (y_a - y_b)^2}.$$

Now consider the inverse problem: having only the distances, is it possible to obtain the map? Classical MDS, addresses this problem.

## Steps of a classical MDS algorithm

***NOTE**: Classical MDS algorithms typically involve some linear algebra.*
*The classical MDS algorithm rests on the fact that the coordinate matrix X can be derived by eigenvalue decomposition from the scalar product matrix $B = XX'$.*
*The problem of constructing B from the proximity matrix P is solved by the method called double centering Where the squared proximity matrix is multiplied with the matrix J*
$\mathbf{J} = \mathbf{I} - n^{-1}\mathbf{1}\mathbf{1}'$. *, where n is the number of objects.*

The following steps summarize the algorithm of classical MDS

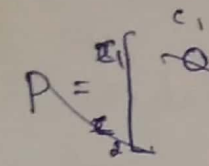
- Set up the matrix of squared $\mathbf{P^{(2)}} = [p^2]$ proximities
- Apply double $\mathbf{B} = -\frac{1}{2}\mathbf{J}\mathbf{P^{(2)}}\mathbf{J}$ centering
- Extract the m largest positive eigenvalues $\lambda_1 \ldots \lambda_m$ of B and the corresponding m Eigen vectors $e_1 \ldots e_m$
- A m -dimensional spatial configuration of the n objects is derived from the coordinate matrix $\mathbf{X} = \mathbf{E_m}\mathbf{\Lambda_m^{1/2}}$.

Where $E_m$ is the matrix of m eigenvectors and $\Lambda_m$ is the diagonal matrix of m eigenvalues of B, respectively

Psy. Rajitha U.N

## Illustration of MDS.

Assume that we have a proximity matrix that has been obtained from the distance measured between ~~city $c_1$ & $c_2$ on a map~~ cities $c_1$, $c_2$, $c_3$ and $c_4$ on a map




[in(mm)]

the proximity matrix of the cities is given by

$$P = \begin{array}{c} \\ c_1 \\ c_2 \\ c_3 \\ c_4 \end{array} \begin{array}{c} \begin{array}{cccc} c_1 & c_2 & c_3 & c_4 \end{array} \\ \begin{bmatrix} 0 & 93 & 82 & 133 \\ 93 & 0 & 52 & 60 \\ 82 & 52 & 0 & 111 \\ 133 & 60 & 111 & 0 \end{bmatrix} \end{array}$$

**Step 1:**

$$P^2 = \begin{bmatrix} 0 & 8649 & 6724 & 17689 \\ 8649 & 0 & 2704 & 3600 \\ 6724 & 2704 & 0 & 12321 \\ 17689 & 3600 & 12321 & 0 \end{bmatrix}$$

Since $n = 4$ objects, the matrix J is calculated by

$$J = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} - 0.25 \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 0.75 & -0.25 & -0.25 & -0.25 \\ -0.25 & 0.75 & -0.25 & -0.25 \\ -0.25 & -0.25 & 0.75 & -0.25 \\ -0.25 & -0.25 & -0.25 & 0.75 \end{bmatrix}$$

(diagonal matrix; unit matrix)

Obtaining double centered matrix B

$$B = -\tfrac{1}{2} J P^2 J = \begin{bmatrix} 3035.06 & -1553.06 & 258.93 & -3740.9 \\ -1553.06 & 507.8 & 5.312 & 1039 \\ 258.9 & 5.31 & 2206 & -247.06 \\ -3740.9 & 1039.9 & -2471.06 & 5173.06 \end{bmatrix}$$

For a two-dimensional representation of the four cities, the first two largest eigen values and the corresponding eigenvectors of B have to be extracted.

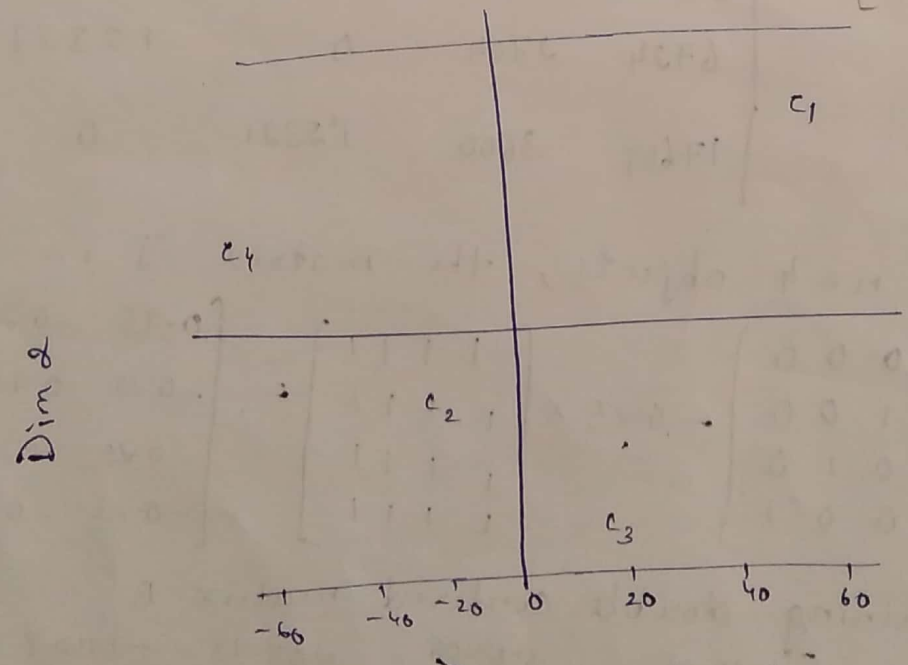
$\lambda_1 = 9724.16 \quad \lambda_2 = 3160.98$

$$e_1 = \begin{bmatrix} -0.63 \\ 0.18 \\ -0.25 \\ 0.704 \end{bmatrix}, \quad e_2 = \begin{bmatrix} -0.586 \\ 0.214 \\ 0.706 \\ -0.334 \end{bmatrix}$$

m dimensional spatial configuration gives the co-ordinates of the cities are obtained by multiplying eigen values and vectors. $\therefore X = E_m \Lambda_m^{1/2}$

$$X = \begin{bmatrix} -0.63 & -0.58 \\ 0.18 & 0.214 \\ -0.25 & 0.706 \\ 0.704 & -0.334 \end{bmatrix} \begin{bmatrix} \sqrt{9724.16} & 0 \\ 0 & \sqrt{3160.98} \end{bmatrix} = \begin{matrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{matrix} \begin{bmatrix} -62.831 & -32.9 \\ 18.403 & 12.02 \\ -24.96 & 39.71 \\ 69.38 & -18.76 \end{bmatrix}$$

$\begin{bmatrix} 98.61 & \text{or} \\ 0 & 56.22 \end{bmatrix} = \begin{bmatrix} 62.12 + 0 & 32 \end{bmatrix}$



Rajitha V-N

## Decision tree Learning :-

* It is a practical method for inductive inference.
* This method search a completely expressive hypothesis space and avoid the difficulties of restricted hypothesis spaces.
* widely used D.T algorithms are ID3, ASSISTANT etc
* It represents a flow-chart kind of model which can also be represented as sets of if-then rules.
* Applications of Decision tree algorithm varies from diagnose of medical case to learning to assess credit risk of loan applicants.

## Decision tree Representation :-

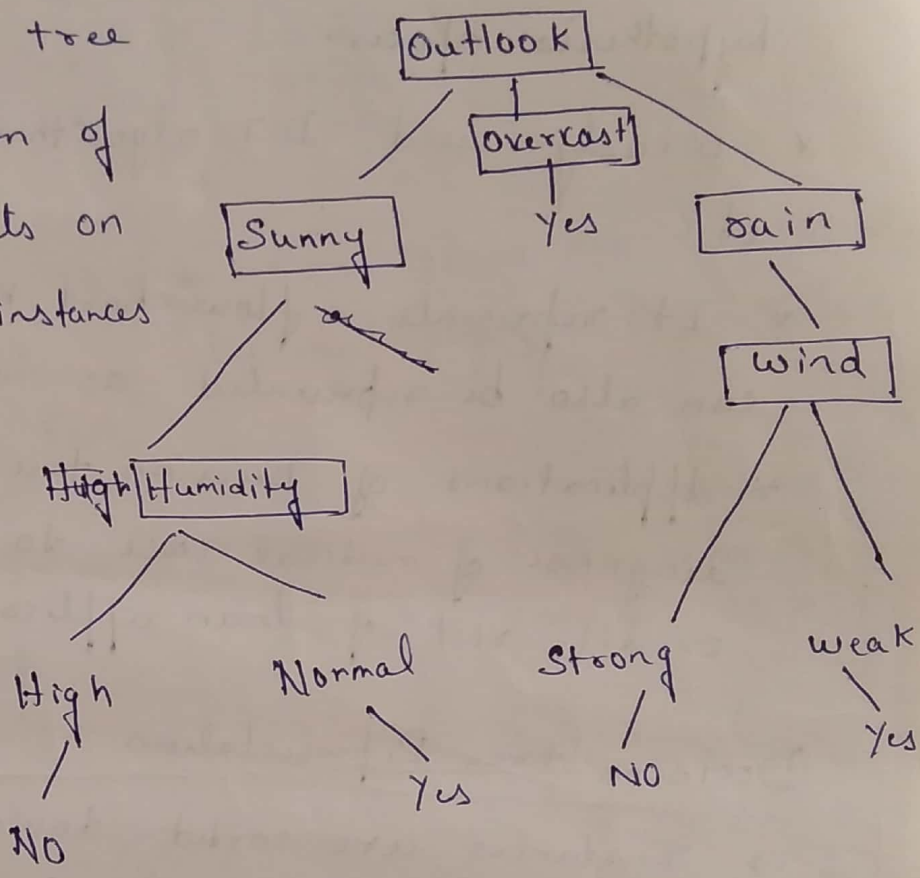
→ Instances are sorted down the tree from the root to some leaf node to achieve classification.

→ Each node in the tree specifies a test of some attribute of the Instance and each branch from that node corresponds to one of the possible values for this attribute.

## Illustration of decision tree :-

Classification of saturday mornings according to whether they are suitable for playing tennis.

In general, decision tree represents a disjunction of conjuctions of constraints on the attribute values of instances




The decision tree shown corresponds to the expression

(outlook = sunny ∧ Humidity = Normal)

∨ (outlook = Overcast)

∨ (outlook = Rain ∧ wind = weak)

# Appropriate problems for Decision Tree learning

Decision tree learning is generally best suited to problems with the following characteristics:

1. Instances are represented by attribute-value pairs.

   Easiest situation for decision tree learning is when each attribute takes on a possible values.

   Ex: If temperature is a variable (attribute) possible values would be hot, mild, cold.

2. The target function has discrete output values.

   Ex: O/p with Boolean values
   More than two values
   Some real valued values.

3. Requirement of disjunctive (or) descriptions

4. Decision tree learning methods are robust to errors. Errors being in training examples and also the errors in attribute values.

5. The training data may contain missing attribute values

(5) ~~$= -P_{\oplus} \log_2 P_{\oplus} - P_{\ominus} \log_2 P_{\ominus}$~~

## Which attribute is the best classifier?

For classification using decision tree, we need to select an attribute ~~that is most~~ which gives the best result.

In order to achieve best result a statistical property called Information gain is considered.

ID3 uses this information gain measure to select among the candidate attributes at each step while growing the tree.

* Information gain is precisely calculated using entropy

$$\text{Entropy}(s) = -P_{\oplus}\log_2 P_{\oplus} - P_{\ominus}\log_2 P_{\ominus}$$

s relates to boolean classification

$P_{\oplus}$ → Portion of positive examples in s

$P_{\ominus}$ → Portion of negative examples in s

* If $P_{\oplus} = 9/14$ & $P_{\ominus} = 5/14$ then

$$\text{Entropy}([9+, 5-]) = -(9/14)\log_2(9/14) - 5/14\log_2(5/14)$$

$$= 0.940$$

(+ 0.483 + 0.159 + 0.530)

Note: If all members of s belong to the same class ie $P_{\oplus} = 1$, then $P_{\ominus} = 0$ then $s = 0$

ie. $s = -14/14\log_2 14/14 - 0.\log_2 0$

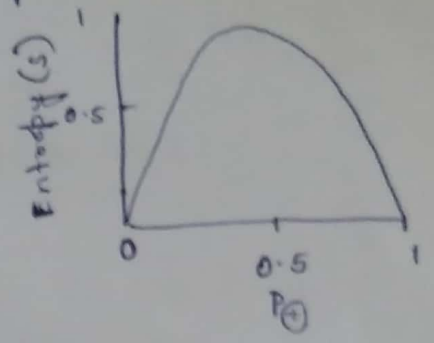
$= 0$

~~S = +~~

Note: If the sample contains equal no. of $P_{\oplus}$ & $P_{\ominus}$ then entropy $S = 1$

If collection contains unequal no. of positive & negative examples, the entropy is between 0 & 1

Represented as




Entropy (from Information theory) prospective) is the min no. of bits of info" needed to encode the classification ie.

If $P_+$ is 1; Receiver knows drawn sample is true ⇒ no msg need to be sent ie $S = 0$

Note: When the target attribute can take on c different values, then the entropy of S is given by

$$\text{Entropy}(S) = \sum_{i=1}^{c} -P_i \log P_i$$

The measure, information gain is the expected reduction in entropy caused by partitioning the examples according to the attribute ie.

$$\text{Gain}(S, A) = \text{Entropy}(S) - \sum_{v \in \text{values}(A)} \frac{|S_v|}{|S|} \text{Entropy}(S_v)$$

expected entropy

A → attribute

S → collection of samples

values(A) → all possible values of attribute A

Calculating gain for each attribute (outlook, temp, Hu, wind)

Information gain for outlook (sunny, overcast, rain)

For Sunny :- P = 2
N = 3

$$gain(S, Sunny) \; E_{sunny} = -2/5 \log_2 2/5 - 3/5 \log_2 3/5$$
$$= 0.529 + 0.442$$
$$= 0.971.$$

for outlook = overcast
P = 4 & N = 0

$$Entropy_{overcast} = 4/4 \log_2 4/4 - 0$$
$$= 0$$

$$\log_2(x) = \frac{\log_{10}(x)}{\log_{10}(2)}$$

for outlook = rain
P = 3 & N = 2

III[y] $Entropy_{rain} = 0.971$

Entropy of outlook =

$$E(outlook) = 5/14 \; I(2,5) + 4/14 \; I(4,0) + 5/14 \; I(3,2)$$
$$= 5/14 \times 0.971 + 5/14 * 0.971 = 0.694$$

$$Gain(outlook) = I(P,n) - E(outlook)$$
$$= 0.940 - 0.694$$
$$= 0.246$$

Information gain for Humidity (High, Normal)

For high : P = 3
N = 4

$$E_{high} = -\frac{3}{7}\log_2\left(\frac{3}{7}\right) - \frac{4}{7}\log_2\left(\frac{4}{7}\right)$$

$$= -\frac{3}{7}\log_2(0.43) - \frac{4}{7}\log(0.57)$$

$$= -\frac{3}{7}(-1.21) - \frac{4}{7}(-0.81)$$

$$= 0.52 + 0.46$$

$$= 0.97$$

$E_{normal}$ = For Normal:
P = 6 , N = 1

$$E_{normal} = -\frac{6}{7}\log_2\left(\frac{6}{7}\right) - \frac{1}{7}\log_2\left(\frac{1}{7}\right)$$

$$= 0.19 + 0.40$$

$$= 0.59$$

Entropy of Humidity =

$$E(\text{Humidity}) = \frac{7}{14}(0.97) + \frac{7}{14}(0.59)$$

$$= 0.48 + 0.29$$

$$= 0.77$$

$$\text{Gain}(S, \text{humidity}) = I(P, n) - E(\text{humidity})$$

$$= 0.940 - 0.77$$

$$= 0.17$$

Information gain for wind (weak, strong)

$$= E(P,T) - \sum_{v \in (w,str)} \frac{|S_v|}{|S|} E(S_v)$$

$$= 0.94 - \left[ \frac{8}{14} E(6,2) + \frac{6}{14} E(3,3) \right]$$

$$= 0.94 - \left[ \frac{8}{14} \left( -\frac{6}{8} \log_2 \frac{6}{8} - \frac{2}{8} \log_2 \frac{2}{8} \right) \right]$$

Gain(s, wind) = 0.048

Information gain for temperature (hot, mild, cool)

$$= 0.94 - \left[ \frac{4}{14} E(2,2) + \frac{6}{14} E(4,2) + \frac{4}{14} E(3,1) \right]$$

$$= 0.94 - \left[ \frac{4}{14} \underbrace{\left( -\frac{2}{4} \log_2 \left(\frac{2}{4}\right) - \frac{2}{4} \log_2 \left(\frac{2}{4}\right) \right)}_{\text{hot}} + \frac{6}{14} \underbrace{\left( -\frac{4}{6} \log_2 \left(\frac{4}{6}\right) - \frac{2}{6} \log_2 \left(\frac{2}{6}\right) \right)}_{\text{mild}} + \frac{4}{14} \underbrace{\left( -\frac{3}{4} \log_2 \left(\frac{3}{4}\right) - \frac{1}{4} \log_2 \left(\frac{1}{4}\right) \right)}_{\text{cool}} \right]$$

$$= 0.94 - 0.906$$

$$= 0.029$$

Gain (S, outlook) = 0.246

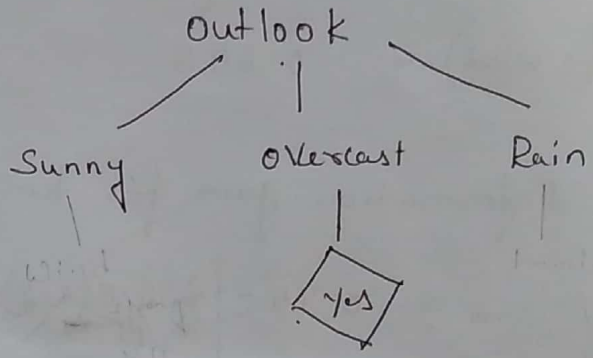
Gain (S, humidity) = 0.151

Gain (S, wind) = 0.048

Gain (S, temp) = 0.029

Comparing the info gain for all the attributes, outlook attribute provides the best prediction of the target attribute.

Here outlook being overcast is always ~~ideal~~ yes for the target function.




Note:- Leaves of the decision tree represent class label

Branches represent conjuctions

Top most decision node in a tree corresponds to the best predictor called root node.

Apply recursion again with ~~unadd~~ attributes humidity, temperature and rain left.

* Overcast is (value of outlook) is a yes, so now need to address sunny & rain.

* Computing entropy for sunny.

Table for values of outlook s. = Sunny:

| Temperature | Humidity | Wind | play tennis |
|---|---|---|---|
| Hot | high | weak | no |
| Hot | high | strong | NO |
| Mild | high | weak | NO |
| cool | normal | weak | Yes |
| mild | normal | strong | Yes |

$$E_{sunny} = -\frac{2}{5}\log_2\left(\frac{2}{5}\right) - \frac{3}{5}\log_2\left(\frac{3}{5}\right)$$

$$= 0.96$$

IIIrd to previous steps compute Information gain for humidity, temp and wind.

Gain(Sunny, humidity) = 0.96

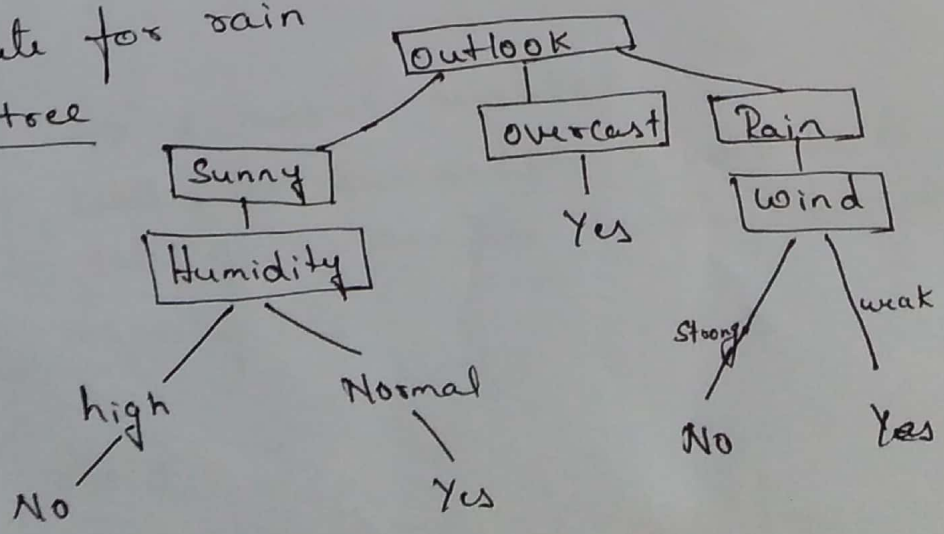
Gain(Sunny, Temp) = 0.57

Gain(Sunny, wind) = 0.019

Gain of sunny with humidity is high

IIIrdly calculate for rain

Final decision tree



# K-Nearest-Neighbors classifiers

* Nearest-neighbor classifiers are based on comparing a given test tuple with training tuples that are similar to it.

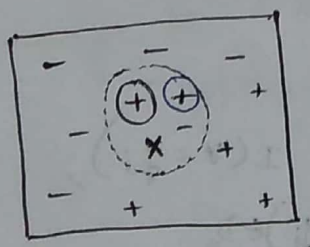
* All of the training tuples are stored in an n-dimensional pattern space. Each tuple represents a point in n-dimensional space.

* When an unknown tuple is given, a K-nearest-neighbor classifier searches the pattern space for the K training tuples that are closest to the unknown tuple. Thus the K nearest neighbors of the unknown tuple is found.

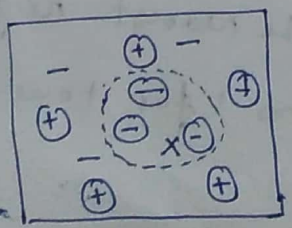
* closeness is defined in terms of a distance metric such as Euclidean distance.

For 2 tuples $X_1 = (x_{11}, x_{12} \cdots x_{1n})$ & $X_2 = (x_{21}, x_{22} \cdots x_{2n})$

$$dist(X_1, X_2) = \sqrt{\sum_{i=1}^{n} (x_{1i} - x_{2i})^2}$$

* 

→ 2 nearest neighbor.
Label of the new sample 'x' will be the label of '+' samples as they are the majority vote.



K = 3

## Algorithm:

1. Let K be the number of nearest neighbors and D be the set of training examples.
2. for each test example $z = (x', y')$ do
3. Compute $d(x', x)$, the distance between $z$ and every example $(x, y) \in D$.
4. Select $D_z \subseteq D$, the set of K closest training examples to $z$.
5. $y' = \underset{v}{\operatorname{argmax}} \sum_{(x_i, y_i) \in D_z} I(v = y_i)$
6. end for.

## Summarizing the algorithm

Algorithm computes the distance between each test example $z = (x', y')$ and all the training example $(x, y) \in D$ to determine its nearest neighbor list $D_z$.

Once the nearest neighbor list is obtained, the test example is classified based on majority class of its nearest neighbors.

Majority voting: $y' = \underset{v}{\operatorname{argmax}} \sum_{(x_i, y_i) \in D_z} I(v = y_i)$,

$v \rightarrow$ class label

$y_i \rightarrow$ class label for one of the nearest neighbors.

$I(\cdot) \rightarrow$ Indicator function returns 1 for true & 0 otherwise.

Apply KNN on the dataset of the company which produces tissues for laboratory. Predict the acceptability of the new type of tissue with acid durability being 3 & strength being 7: for K = 2.

Sol:

| Name | Acid Durability | Strength | Class |
|---|---|---|---|
| Type-1 | 7 | 7 | Bad |
| Type-2 | 7 | 4 | Bad |
| Type-3 | 3 | 4 | Good |
| Type-4 | 1 | 4 | Good. |

Distance

$D(\text{Type 5}, \text{type 1}) = \sqrt{(7-3)^2 + (7-7)^2} = 4 \Rightarrow$ Bad

$D(\text{"}, \text{type 2}) = \sqrt{(7-3)^2 + (4-7)^2} = 4 \Rightarrow$ Bad

$D(\text{type 5}, \text{type 3}) = 1 \Rightarrow$ Good

$D(\text{type 5}, \text{type 4}) = 2 \Rightarrow$ Good

for K = 2 it considering 2 immediate neighbour belongs to class "Good".

Type-5 belongs to class Good.

* Apply KNN algorithm and predict the type of food to which Tomato belongs to considering the attribute (sweet = 6 & crunch = 4).

Sol

| Ingredient | sweet | crunch | food type |
|---|---|---|---|
| Grape | 8 | 5 | Fruit |
| Greenbean | 3 | 7 | vegetable |
| Nuts | 3 | 6 | protein |
| Orange | 7 | 3 | Fruit |

$D(\text{tomato}, \text{grape}) = \sqrt{(8-6)^2+(4-5)^2} = 2.2$

$D(\text{,,}, \text{Greenbean}) = \sqrt{(6-3)^2+(4-7)^2} = 4.2$

$D(\text{,,}, \text{Nuts}) = \sqrt{(6-3)^2+(4-6)^2} = \sqrt{13} = 3.6$

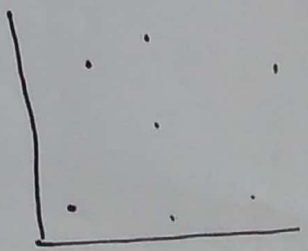
$D(\text{,,}, \text{orange}) = \sqrt{(6-7)^2+(4-3)^2} = 1.4$

The min distance is 1.4. Hence Tomato will belong to food type of orange is Fruit.

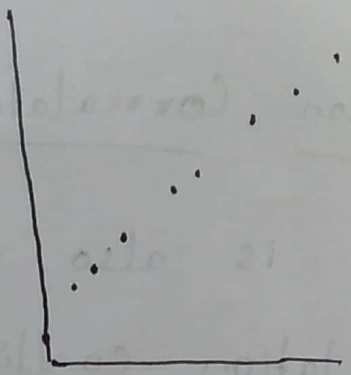
# Pearson Correlation Coefficient

* It is also known as the "product moment correlation coefficient" PMCC or simply correlation.
* Pearson's correlations are suitable only for metric variables.
* Correlation is a technique for investigating the relationship between two quantitative, continuous variables (ex: age v/s blood pressure).
* Pearson's coefficient ($r$) is a measure of the strength of the association between the two variables
* The value of the coeff $r$ varies from -1 to 1
* If $r = -1$, data lie on a perfect straight line with a negative slope

* If $r = 0$, no linear relation b/w the variables



If $r = +1$, data lie on a perfect straight line with a positive slope



* positive correlation ($r = +1$) indicates that both variable increases or decreases together.
Negative correlation ($r = -1$) indicates that as one variable increases, the other decreases & vice versa.

Pearson's coefficient

$$r = \frac{\frac{1}{n-1}\left(\sum (x-\bar{x})(y-\bar{y})\right)}{\sqrt{\frac{1}{n-1}\sum (x-\bar{x})^2}\cdot\sqrt{\frac{1}{n-1}\sum (y-\bar{y})^2}}$$

Simplified as

$$r = \frac{\sum xy - \frac{\sum x \sum y}{n}}{\sqrt{\sum x^2 - \frac{(\sum x)^2}{n}}\sqrt{\sum y^2 - \frac{(\sum y)^2}{n}}}$$

compute Pearson's coefficient of correlation between advertisement cost and sales as per the data given below.

| Advertisement cost in 1000's | 39 | 65 | 62 | 90 | 82 | 75 | 25 | 98 | 36 | 78 |
|---|---|---|---|---|---|---|---|---|---|---|
| Sales in lakhs | 47 | 53 | 58 | 86 | 62 | 68 | 60 | 91 | 51 | 84 |

**Solu** From the data, $n = 10$

$$\Sigma x = 650 \quad \Sigma y = 660 \quad \Sigma xy = 45604$$

$$\Sigma x^2 = 47648 \quad \Sigma y^2 = 45784$$

$$r = \frac{45604 - \frac{(650)(660)}{10}}{\sqrt{47648 - \frac{(650)^2}{10}}\sqrt{45784 - \frac{(660)^2}{10}}}$$

$$r = \frac{45604 - 42900}{(73.47)(47.1)} = 0.7804$$

Correlation coeff is positively correlated.