

Artificial Intelligence

Chapter 4

**INTERIM SEMESTER 2021-22
BPL
CSE3007-LT-AB306
FACULTY: SIMI V.R.**



Machine learning algorithms

Regression

- ❑ Technique used for the modeling and analysis of numerical data
- ❑ Exploits the relationship between two or more variables so that we can gain information about one of them through knowing values of the other
- ❑ Regression can be used for prediction, estimation, hypothesis testing, and modeling causal relationships.
- ❑ Linear functions provide the basis for many learning algorithms.
- ❑ Can solve the problem of predicting a real-valued function from training examples.
- ❑ Regression is a set of techniques for estimating relationships

Simple Linear Regression

We're going to fit a line $y = b_0 + b_1x$ to our data. Here, x is called the independent variable or predictor variable, and y is called the dependent variable or response variable.

Before we talk about how to do the fit, let's take a closer look at the important quantities from the fit:

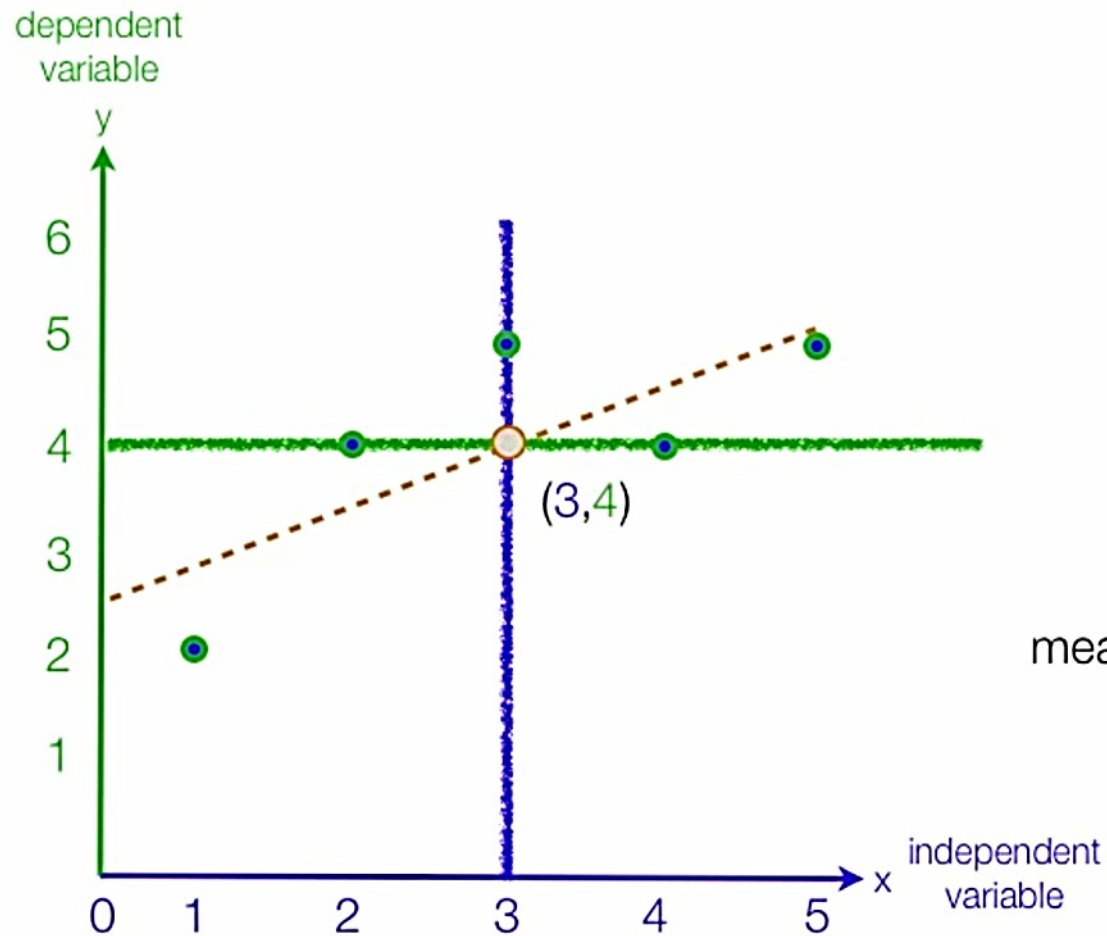
- b_1 is the slope of the line: this is one of the most important quantities in any linear regression analysis.

A value very close to 0 indicates little to no relationship; large positive or negative values indicate large positive or negative relationships, respectively.

- b_0 is the intercept of the line.

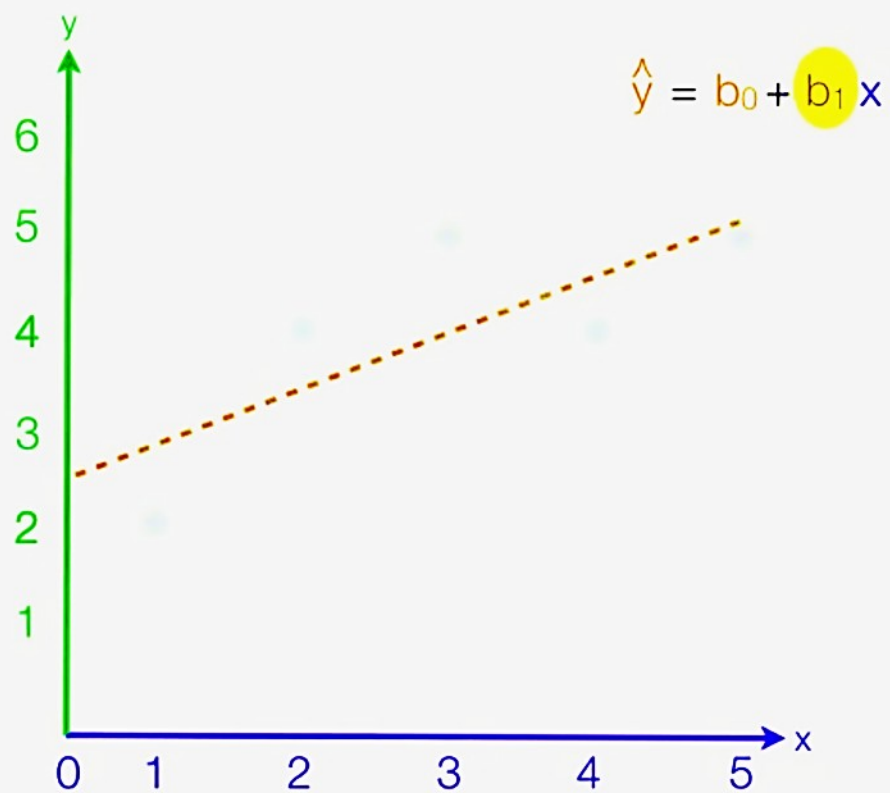
In order to actually fit a line, we'll start with a way to quantify how good a line is. We'll then use this to fit the “best” line we can.

Linear Regression



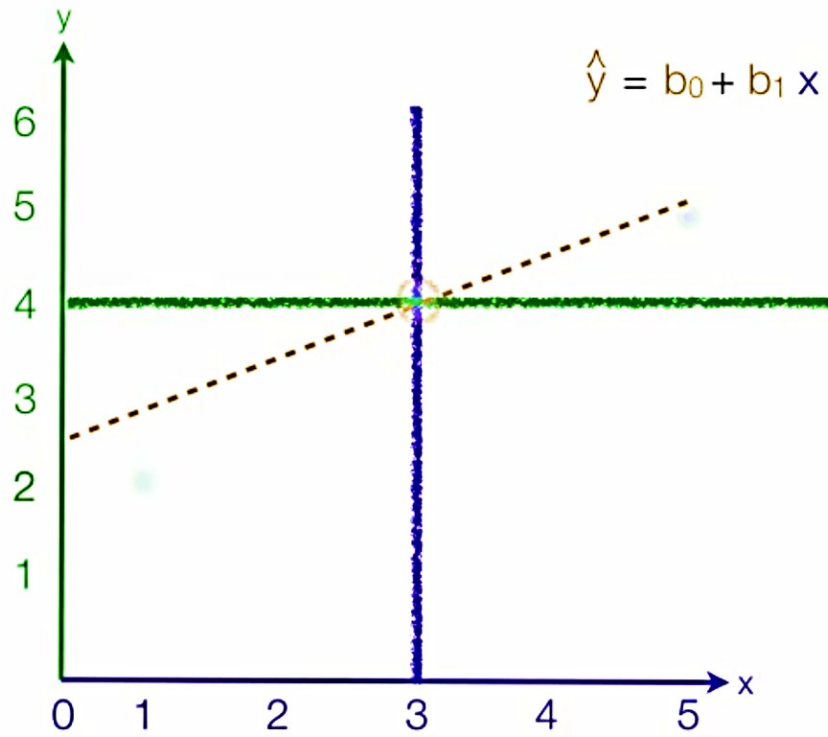
independent variable	dependent variable
x	y
1	2
2	4
3	5
4	4
5	5

mean



x	y	$x - \bar{x}$	$y - \bar{y}$	$(x - \bar{x})^2$	$(x - \bar{x})(y - \bar{y})$
1	2	-2	-2	4	4
2	4	-1	0	1	0
3	5	0	1	0	0
4	4	1	0	1	0
5	5	2	1	4	2

mean 3 4



x	y	$x - \bar{x}$	$y - \bar{y}$	$(x - \bar{x})^2$	$(x - \bar{x})(y - \bar{y})$
1	2	-2	-2	4	4
2	4	-1	0	1	0
3	5	0	1	0	0
4	4	1	0	1	0
5	5	2	1	4	2

mean

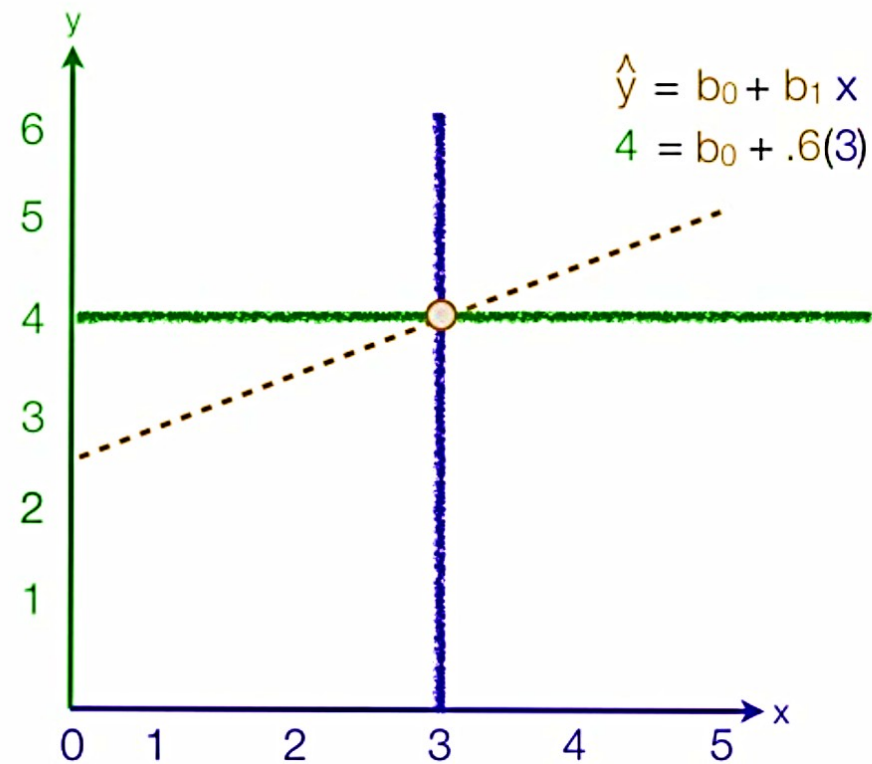
3

4

10

6

$$b_1 = \frac{6}{10} = .6 = \frac{\sum (x - \bar{x})(y - \bar{y})}{\sum (x - \bar{x})^2}$$



$$b_0 = 2.2$$

$$b_1 = .6$$

$$\hat{y} = 2.2 + .6x$$

x	y	$x - \bar{x}$	$y - \bar{y}$	$(x - \bar{x})^2$	$(x - \bar{x})(y - \bar{y})$
1	2	-2	-2	4	4
2	4	-1	0	1	0
3	5	0	1	0	0
4	4	1	0	1	0
5	5	2	1	4	2
mean		3	4	10	6

$$\begin{array}{r}
 4 = b_0 + .6(3) \\
 4 = b_0 + 1.8 \\
 \underline{-1.8} \quad \underline{-1.8} \\
 2.2 = b_0
 \end{array}$$

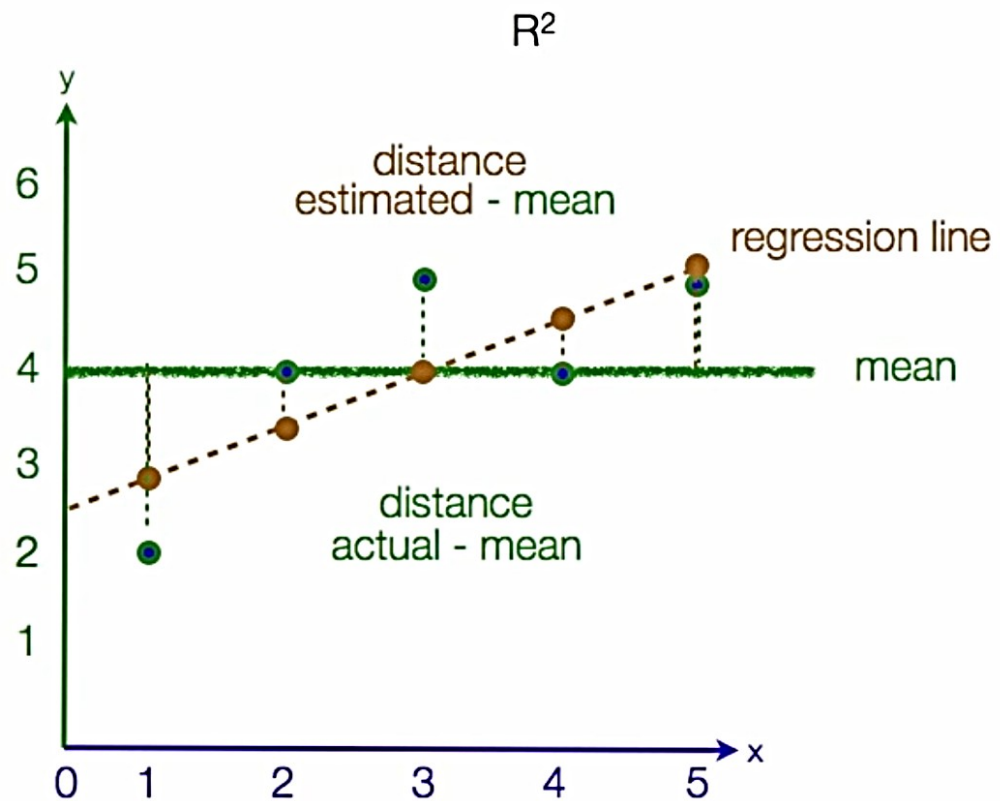
$$b_1 = \frac{6}{10} = .6 = \frac{\sum (x - \bar{x})(y - \bar{y})}{\sum (x - \bar{x})^2}$$

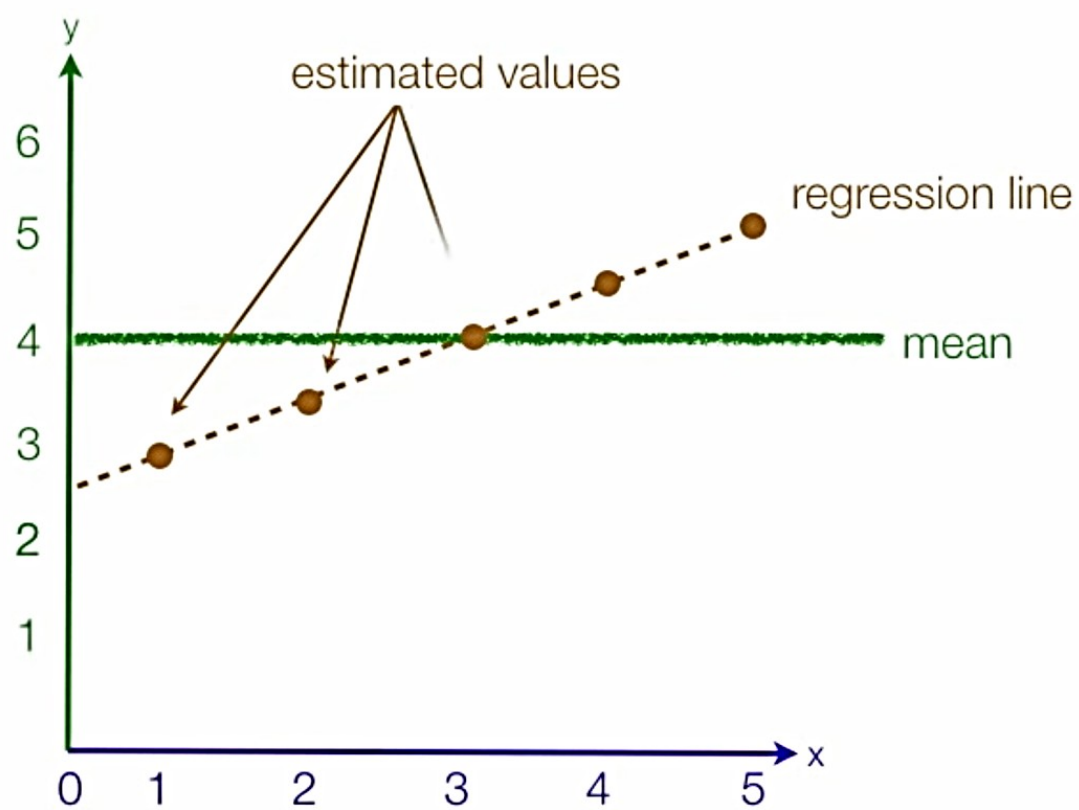
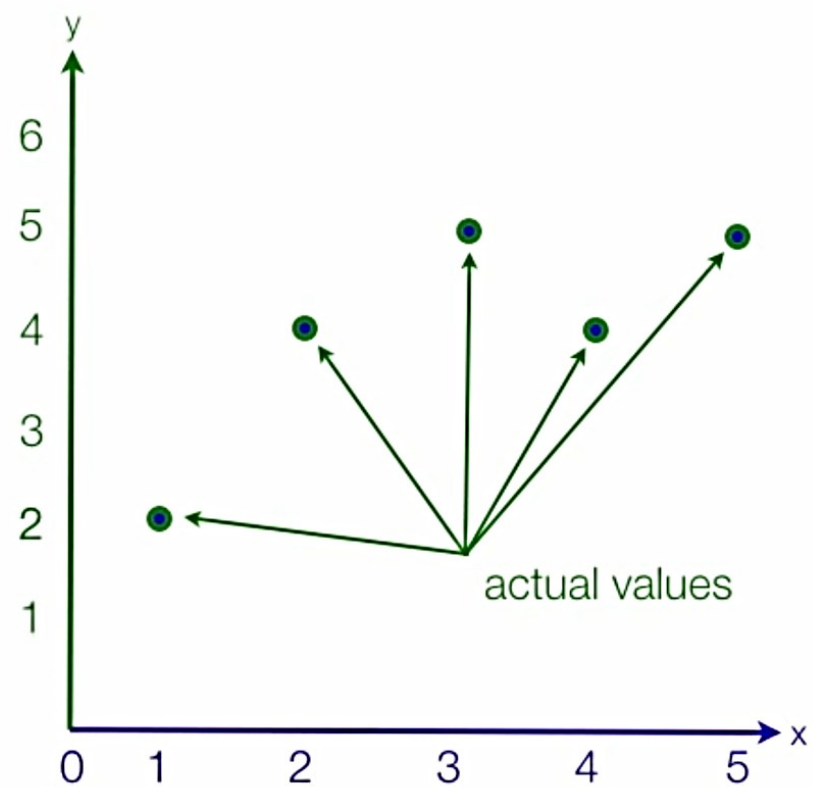
Goodness of fit - R^2

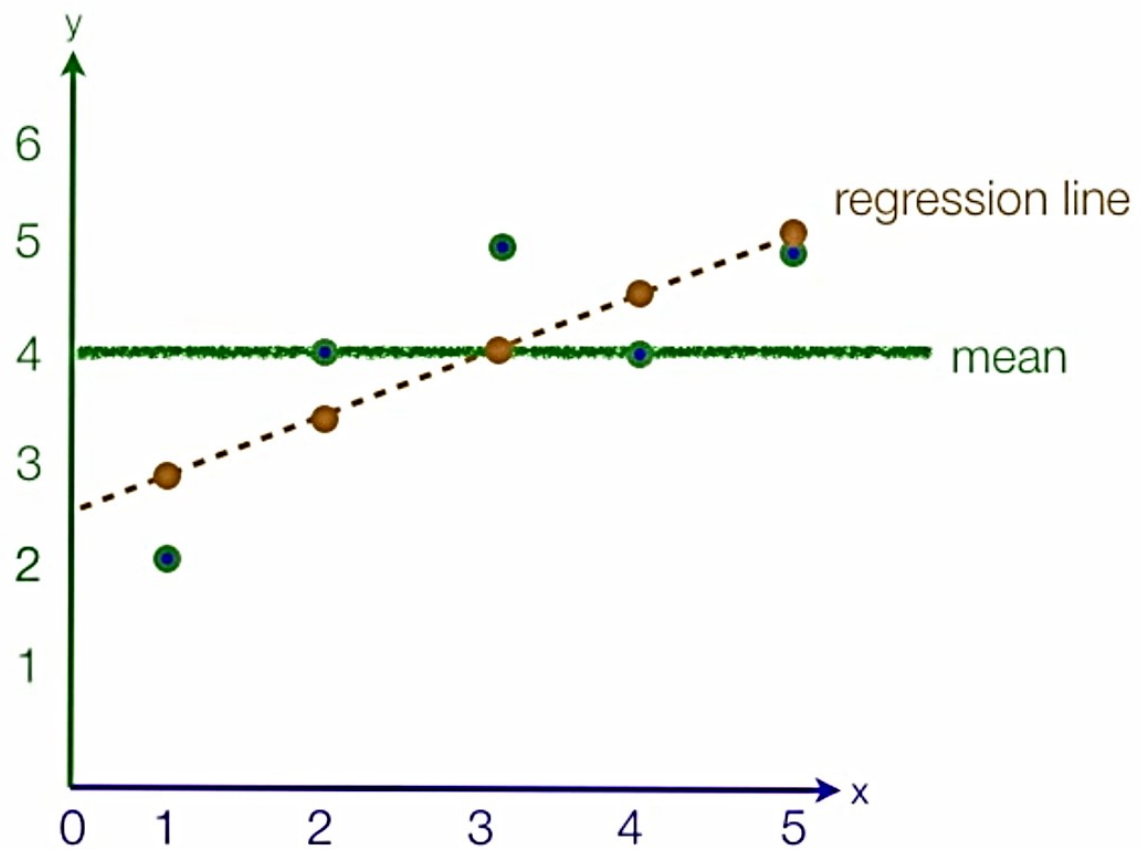
WHAT IS R-SQUARED?

- ❑ R-squared is a statistical measure of how close the data are to the fitted regression line.
- ❑ It is also known as the coefficient of determination, or the coefficient of multiple determination for multiple regression.
- ❑ The definition of R-squared is fairly straight-forward; it is the percentage of the response variable variation that is explained by a linear model.
- ❑ $R\text{-squared} = \text{Explained variation} / \text{Total variation}$

Goodness of fit - R^2

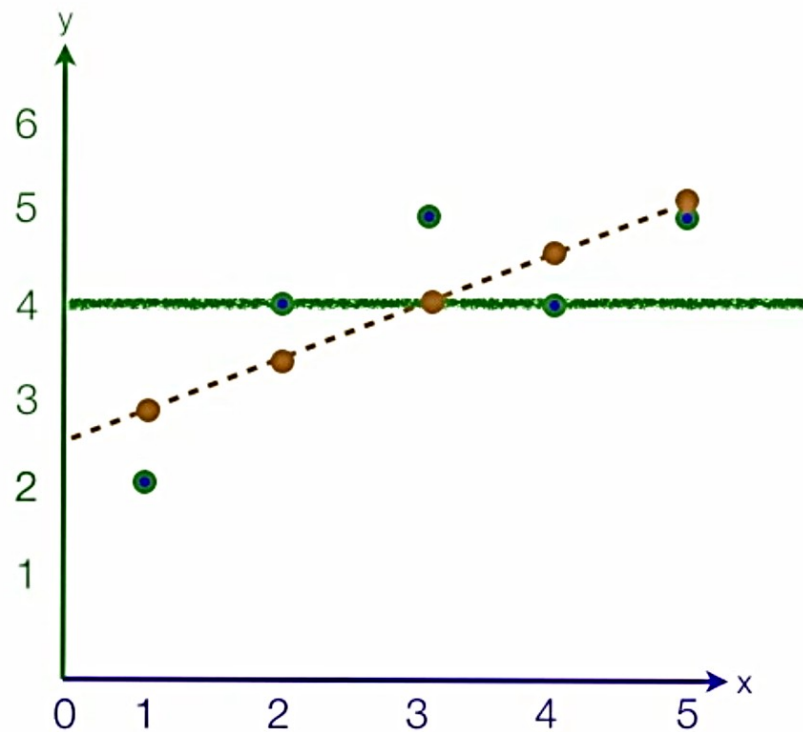






distance
actual - mean

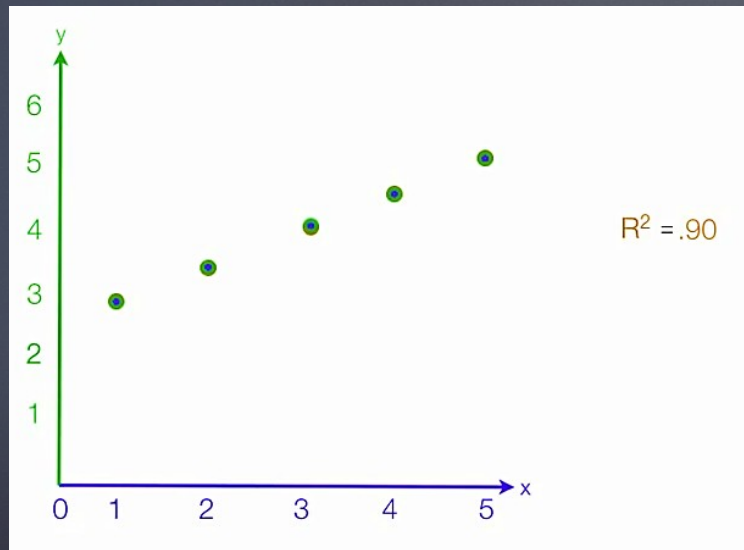
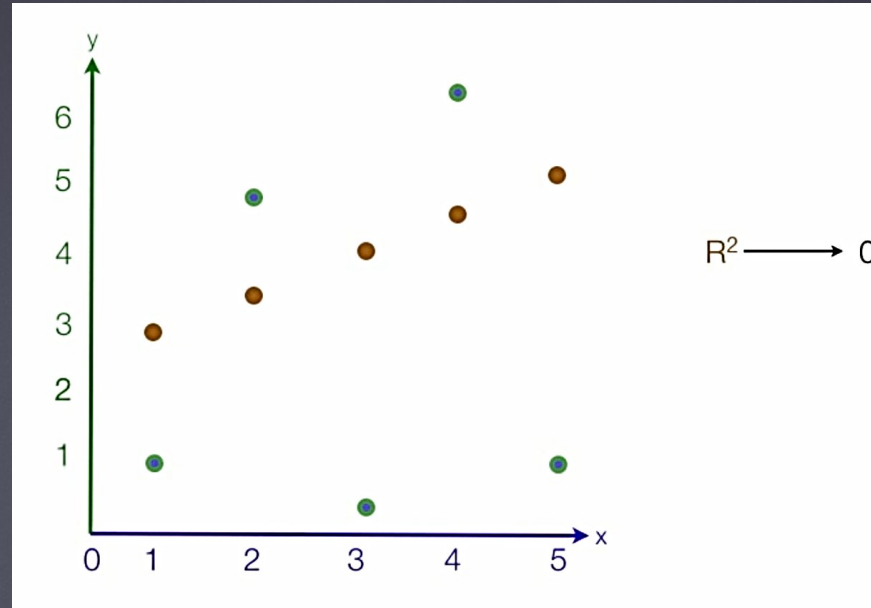
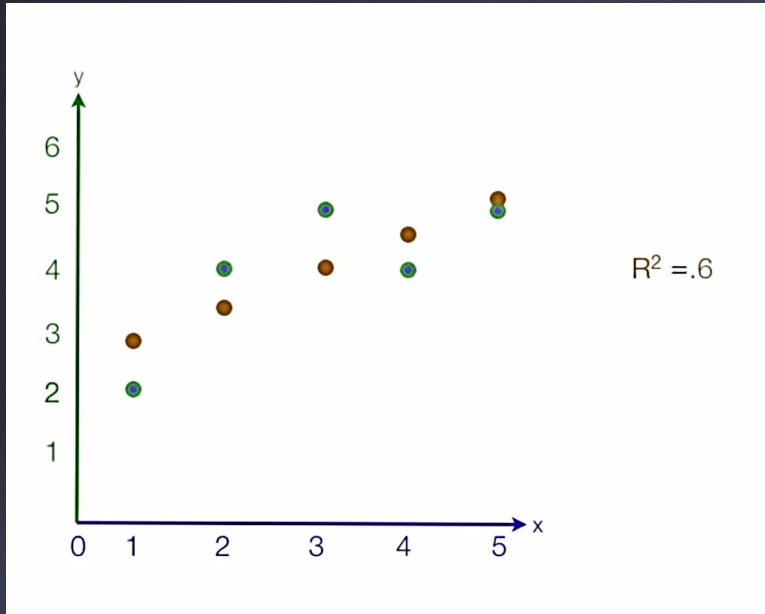
distance
estimated - mean



x	y	$y - \bar{y}$	$(y - \bar{y})^2$	\hat{y}	$\hat{y} - \bar{y}$	$(\hat{y} - \bar{y})^2$
1	2	-2	4	2.8	-1.2	1.44
2	4	0	0	3.4	-.6	.36
3	5	1	1	4	0	0
4	4	0	0	4.6	.6	.36
5	5	1	1	5.2	1.2	1.44
mean		4	6			3.6

$$R^2 = \frac{3.6}{6} = .6 = \frac{\sum (\hat{y} - \bar{y})^2}{(y - \bar{y})^2}$$

Interpretation of values of R^2



$$R^2=1$$

Regression line is a
Perfect fit on actual
values

$$R^2=0$$

There is larger
distance between
Actual and predicted
values.

Mean Squared Error

$$MSE = \frac{1}{n} \sum \left(\underbrace{y - \hat{y}}_{\substack{\text{The square of the difference} \\ \text{between actual and} \\ \text{predicted}}} \right)^2$$

Mean Squared Error

Definition

- ❑ The mean squared error (MSE) tells you how close a regression line is to a set of points.
- ❑ It does this by taking the distances from the points to the regression line (these distances are the “errors”) and squaring them.
- ❑ The squaring is necessary to remove any negative signs.
- ❑ It also gives more weight to larger differences.
- ❑ It's called the mean squared error as you're finding the average of a set of errors.
- ❑ The lower the MSE, the better the forecast.

Mean Squared Error

General steps to calculate the MSE from a set of X and Y values:

- 1. Find the regression line.**
- 2. Insert your X values into the linear regression equation to find the new Y values (Y').**
- 3. Subtract the new Y value from the original to get the error.**
- 4. Square the errors.**
- 5. Add up the errors (the Σ in the formula is summation notation).**
- 6. Find the mean.**

Mean Squared Error

Example Problem:

Find the MSE for the following set of values: (43,41), (44,45), (45,49), (46,47), (47,44).

Step 1: Find the regression line.

$$\text{Regression line } y = 9.2 + 0.8x.$$

Step 2: Find the new Y' values:

$$9.2 + 0.8(43) = 43.6$$

$$9.2 + 0.8(44) = 44.4$$

$$9.2 + 0.8(45) = 45.2$$

$$9.2 + 0.8(46) = 46$$

$$9.2 + 0.8(47) = 46.8$$

x	y		y-	(y-) ²
43	41	43.6	41 - 43.6 = -2.6	6.76
44	45	44.4	45 - 44.4 = 0.6	0.36
45	49	45.2	49 - 45.2 = 3.8	14.44
46	47	46	47 - 46 = 1	1
47	44	46.8	44 - 46.8 = -2.8	7.84

Step 3: Find the error (Y - Y'):

$$41 - 43.6 = -2.6$$

$$45 - 44.4 = 0.6$$

$$49 - 45.2 = 3.8$$

$$47 - 46 = 1$$

$$44 - 46.8 = -2.8$$

Step 4: Square the Errors:

$$-2.6^2 = 6.76$$

$$0.6^2 = 0.36$$

$$3.8^2 = 14.44$$

$$1^2 = 1$$

$$-2.8^2 = 7.84$$

Step 5: Add all of the squared errors up: $6.76 + 0.36 + 14.44 + 1 + 7.84 = 30.4$

Step 6: Find the mean squared error: $30.4 / 5 = 6.08$.

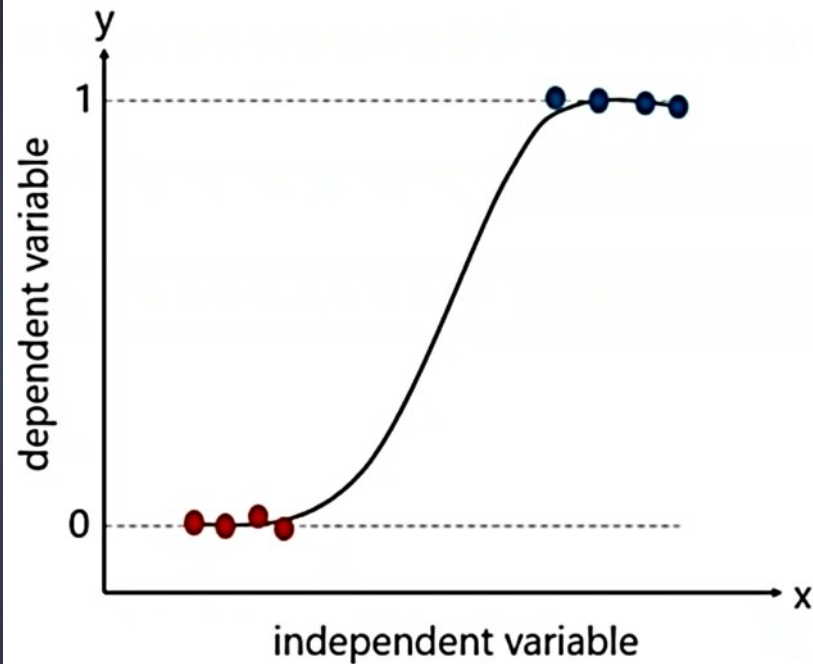
What does the Mean Squared Error Tell You?

- ▶ The smaller the mean squared error, the closer you are to finding the line of best fit.
- ▶ Depending on your data, it may be impossible to get a very small value for the mean squared error.
- ▶ For example, the above data is scattered wildly around the regression line, so 6.08 is as good as it gets (and is in fact, the line of best fit).

The smallest MSE would be the line of best fit.

Logistic regression

Logistic Regression is a method used to predict a dependent variable, given a set of independent variables, such that the dependent variable is categorical.



- *Dependent variable (Y):*
The response binary variable holding values like 0 or 1, Yes or No, A, B or C
- *Independent variable (X):*
The predictor variable used to predict the response variable.

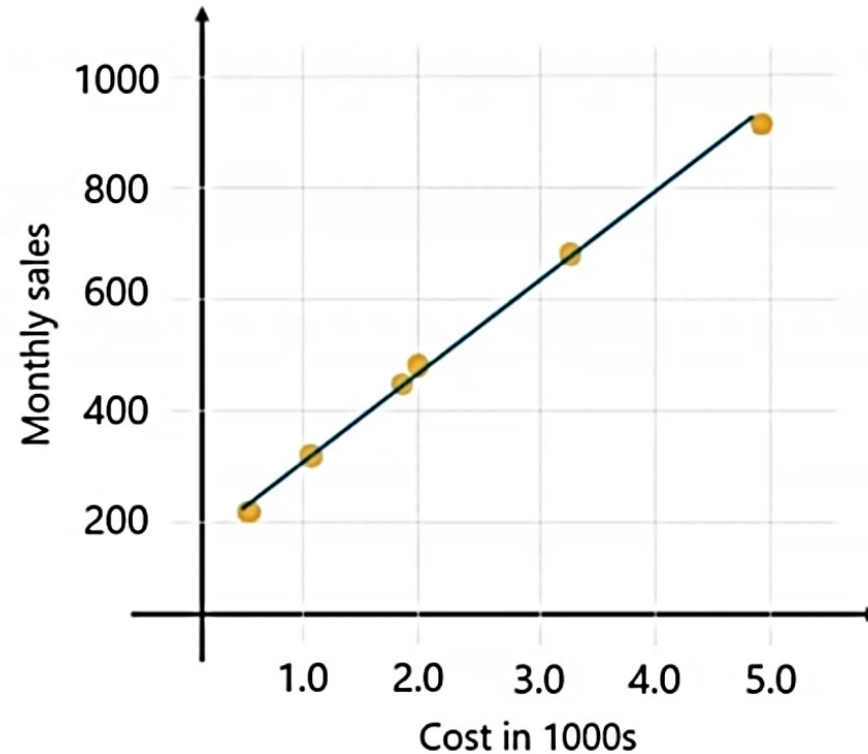
The following equation is used to represent a linear regression model:

$$\text{Log}\left(\frac{P}{1-P}\right) = a_0 + a_1x_1 + a_2x_2 + \dots + a_nx_n$$

Linear Regression Use Case

To forecast monthly sales by studying the relationship between the monthly e-commerce sales and the online advertising costs.

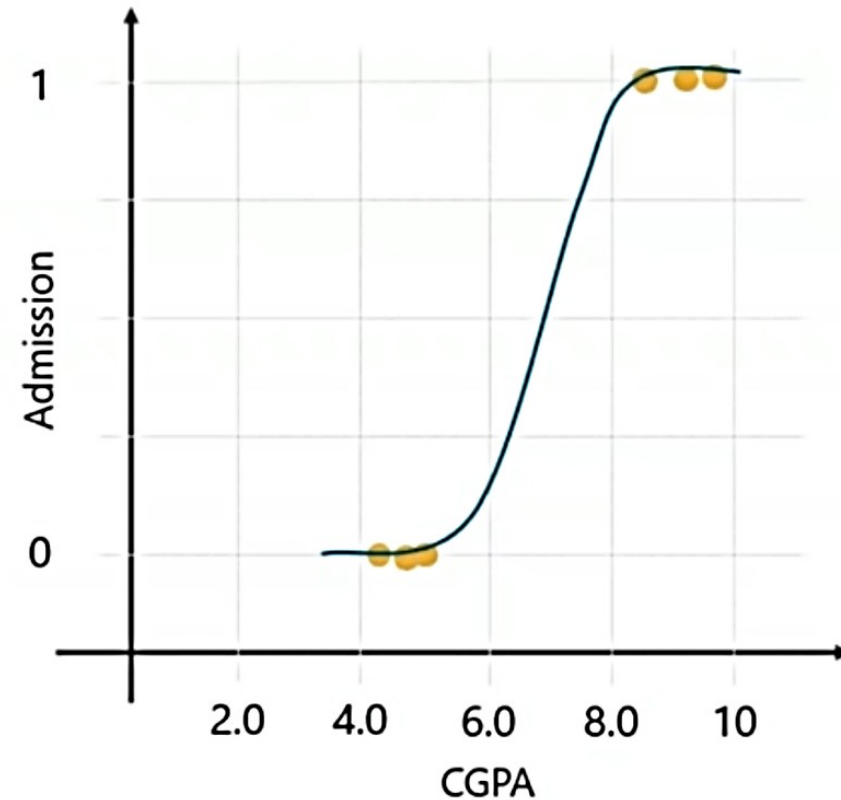
Monthly sales	Advertising cost In 1000s
200	0.5
900	5
450	1.9
680	3.2
490	2.0
300	1.0



Logistic Regression Use Case

To predict if a student will get admitted to a school based on his CGPA.

Admission	CGPA
0	4.2
0	5.1
0	5.5
1	8.2
1	9.0
1	9.1

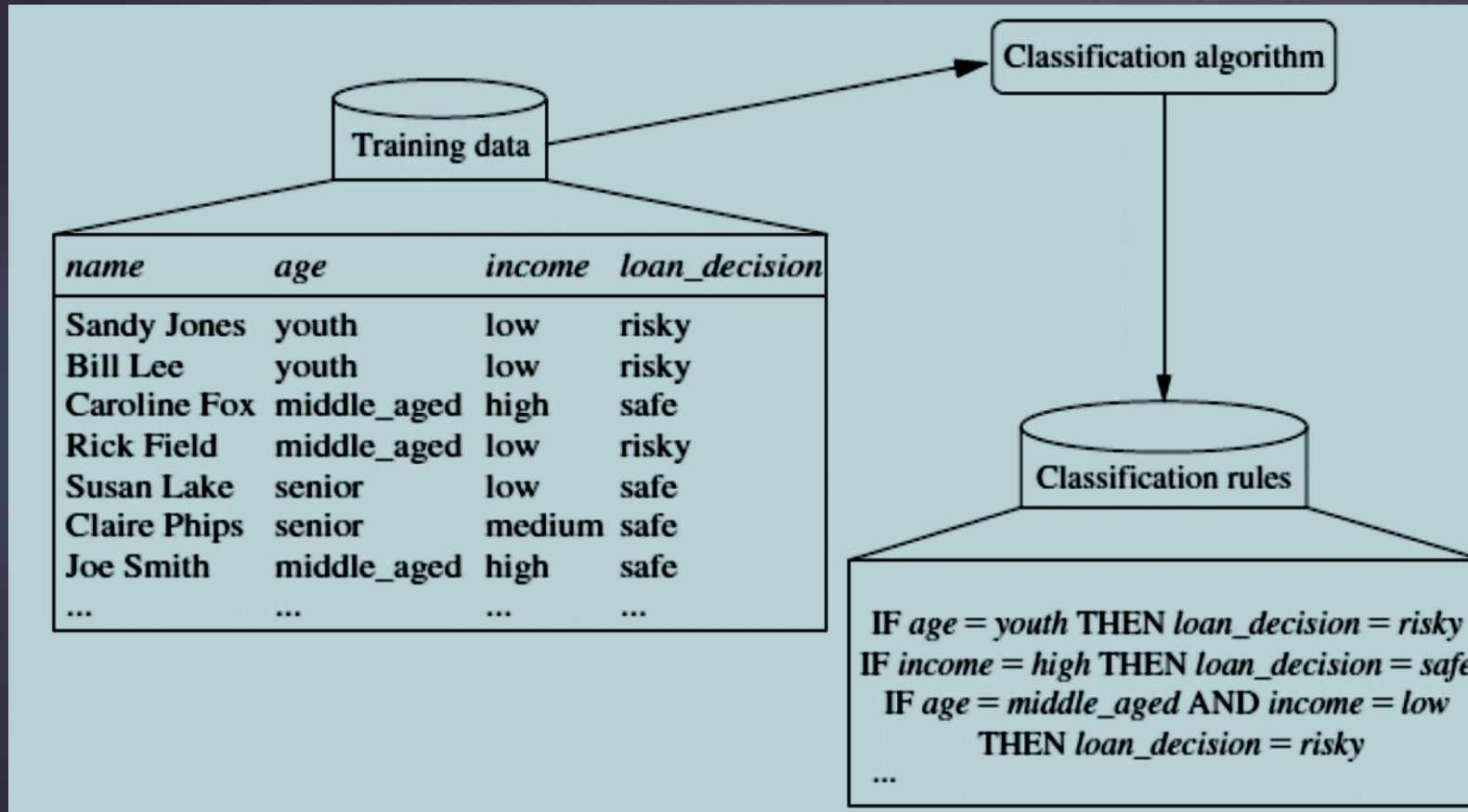


	Linear Regression	Logistic Regression
1. Definition	To predict a continuous dependent variable based on values of independent variable	To predict a categorical dependent variable based on values of independent variables
2. Variable Type	Continuous dependent variable	Categorical dependent variable
3. Estimation Method	Least square estimation	Maximum likelihood estimation
4. Equation	$Y = a_0 + a_1x$	$\text{Log}() = a_0 + a_1x_1 + a_2x_2 + \dots + a_nx_n$
5. Best fit line	Straight line	Curve
6. Relationship between dependent and independent variable	Linear	Non linear
7. Output	Predicted Integer value	Predicted binary value (0/1)

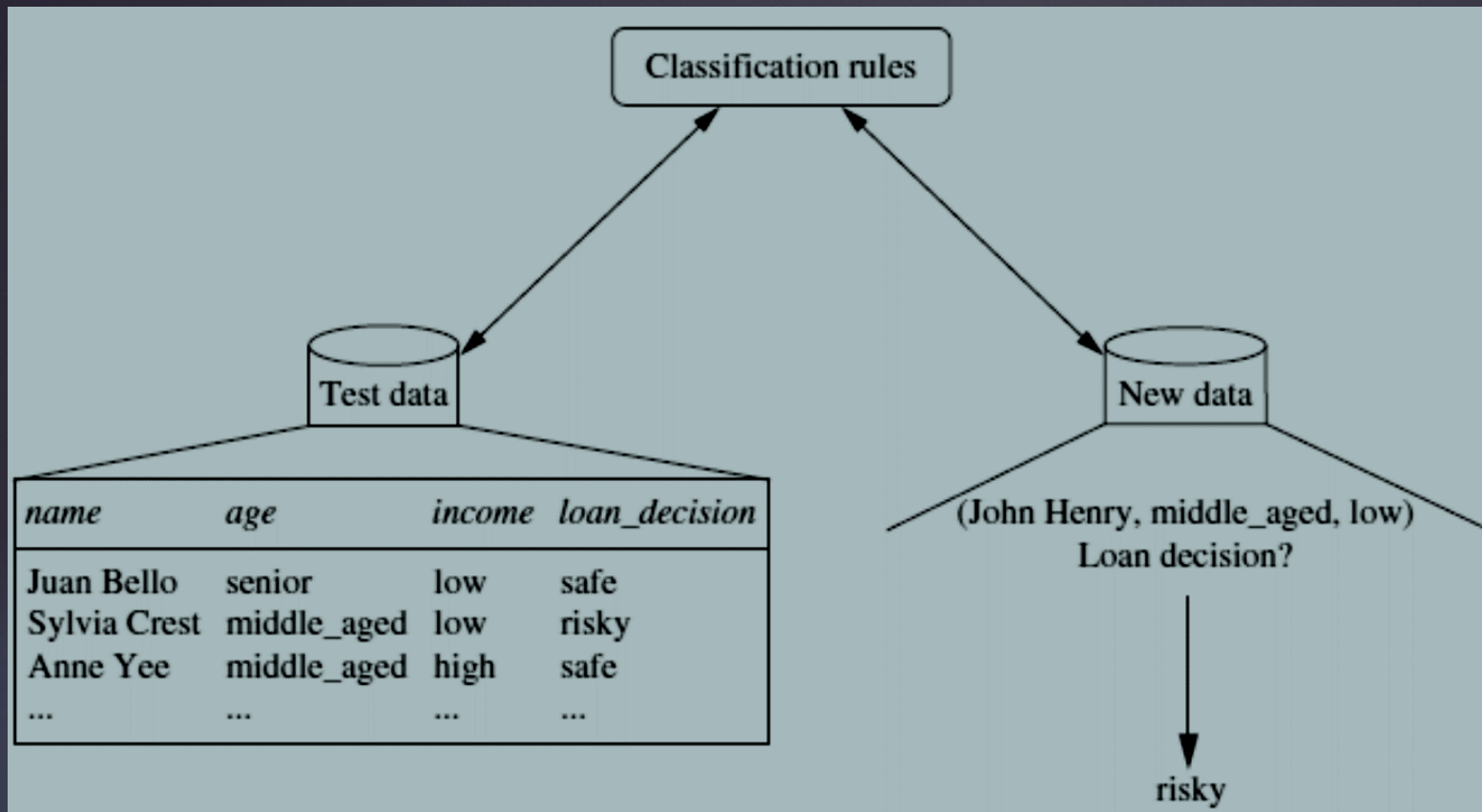
Classification

Data classification is a two-step process

1. Consisting of a *learning step* (where a classification model is constructed)
2. A *classification step* (where the model is used to predict class labels for given data).



Learning: Training data are analyzed by a classification algorithm. Here, the class label attribute is *loan decision*, and the learned model or classifier is represented in the form of classification rules.



Classification:

Test data are used to estimate the accuracy of the classification rules. If the accuracy is considered acceptable, the rules can be applied to the classification of new data tuples.

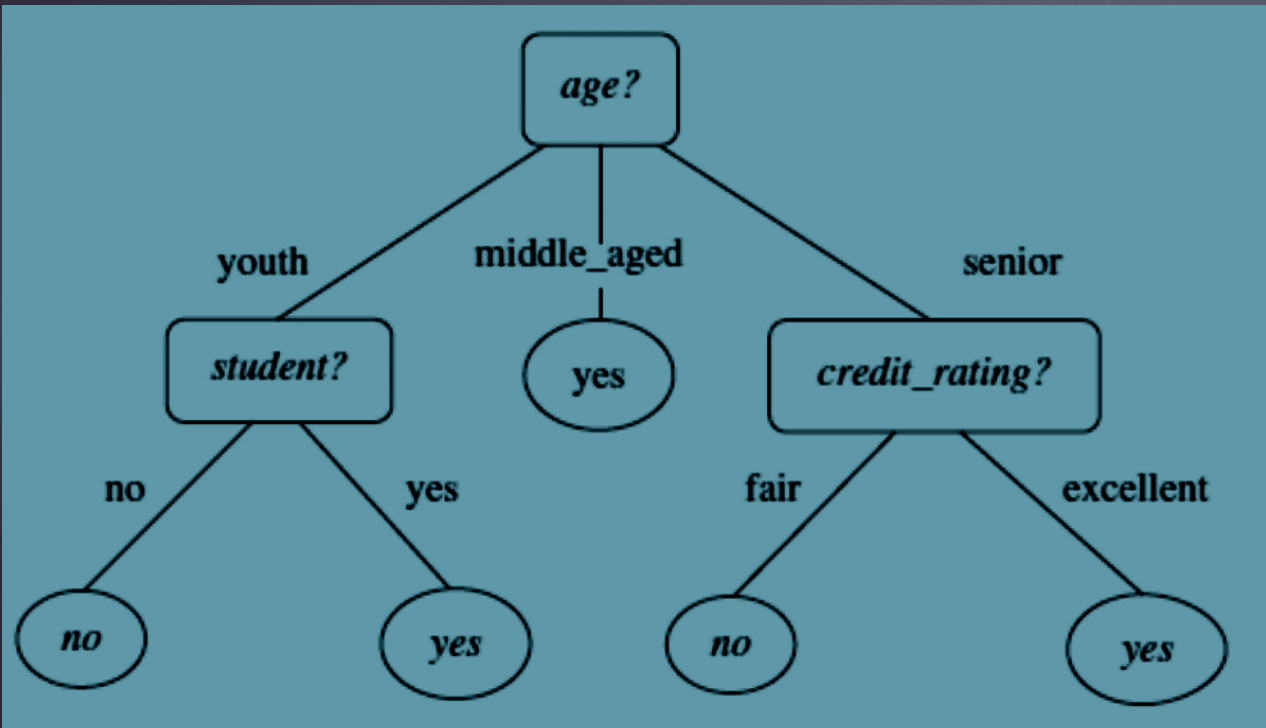
Decision tree induction : learning of decision trees from class-labeled training tuples.

- The **accuracy** of a classifier on a given test set is the percentage of test set tuples that are correctly classified by the classifier.
- The associated class label of each test tuple is compared with the learned classifier's class prediction for that tuple.

Decision Tree

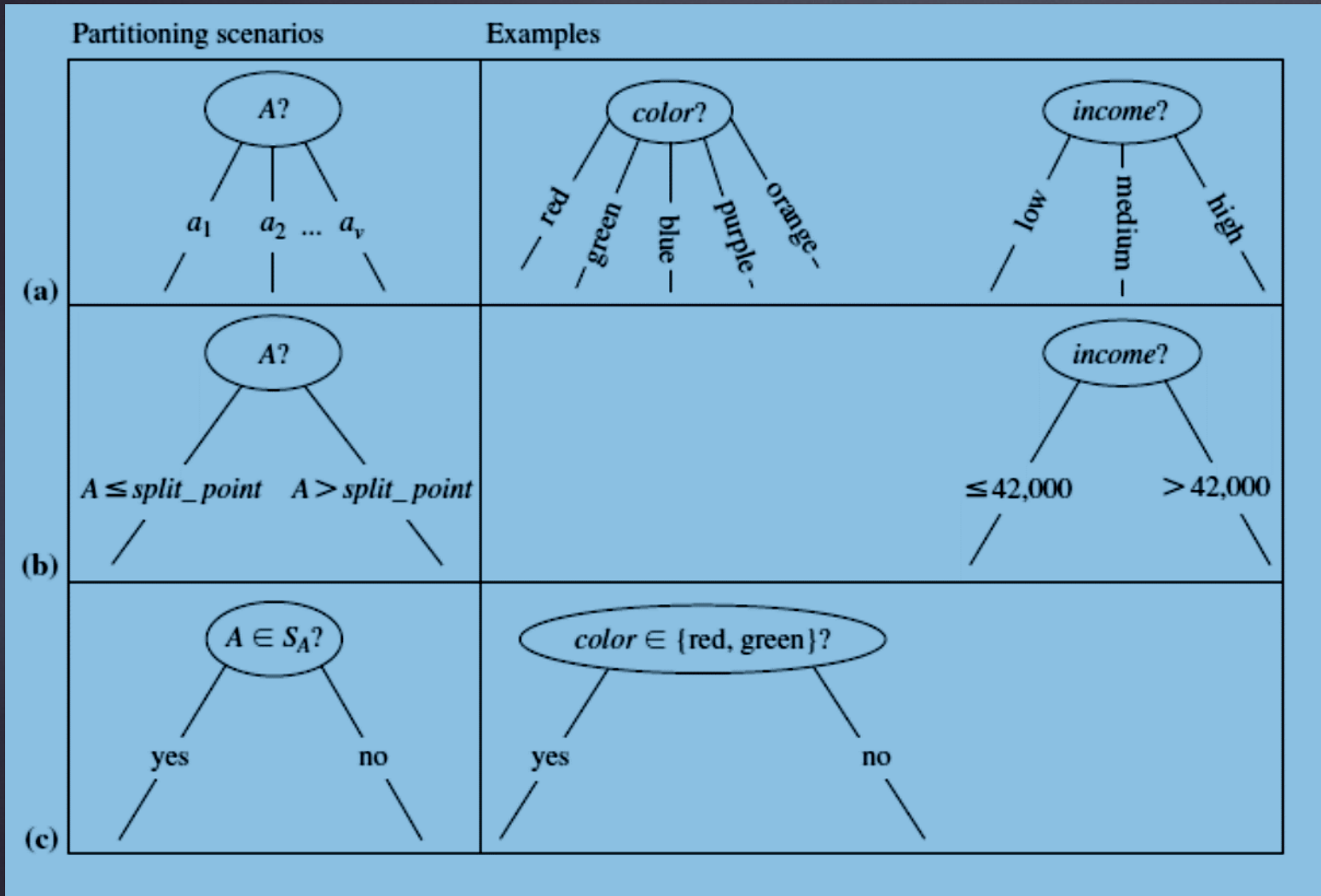
Decision tree : flowchart-like tree structure, where each **internal node** (nonleaf node) denotes a test on an attribute, each **branch** represents an outcome of the test, and each **leaf node** (or *terminal node*) holds a class label.

The topmost node in a tree is the **root node**.



- A decision tree for the concept *buys computer* that is, it predicts whether a customer at *AllElectronics* is likely to purchase a computer.
- Internal nodes are denoted by rectangles, and leaf nodes are denoted by ovals.
- Some decision tree algorithms produce only *binary* trees (where each internal node branches to exactly two other nodes), whereas others can produce nonbinary trees.
- Each internal (nonleaf) node

Partitioning based on the splitting criterion



This figure shows three possibilities for partitioning tuples based on the splitting criterion, each with examples. Let A be the splitting attribute.

(a) If A is discrete-valued, then one branch is grown for each known value of A .

(b) If A is continuous-valued, then two branches are grown, corresponding to

$A \leq \text{split_point}$ and $A > \text{split_point}$.

(c) If A is discrete-valued and a binary tree must be produced, then the test is of the form $A \in S_A$, where S_A is the splitting subset for A .

Attribute Selection Measures

- ❖ An attribute selection measure is a heuristic for selecting the splitting criterion that “best” separates a given data partition, D , of class-labeled training tuples into individual classes.
- ❖ If we were to split D into smaller partitions according to the outcomes of the splitting criterion, ideally each partition would be pure (i.e., all the tuples that fall into a given partition would belong to the same class).
- ❖ Conceptually, the “best” splitting criterion is the one that most closely results in such a scenario.
- ❖ Attribute selection measures are also known as splitting rules because they determine how the tuples at a given node are to be split.
- ❖ **Three popular attribute selection measures**

Information gain, gain ratio, and Gini index.

Information Gain/ Entropy

- * This measure is based on pioneering work by Claude Shannon on information theory, which studied the value or “information content” of messages.
- * Let node N represent or hold the tuples of partition D .
- * The attribute with the highest information gain is chosen as the splitting attribute for node N .
- * This attribute minimizes the information needed to classify the tuples in the resulting partitions and reflects the least randomness or “impurity” in these partitions.
- * Such an approach minimizes the expected number of tests needed to classify a given tuple and guarantees that a simple (but not necessarily the simplest) tree is found.

The expected information needed to classify a tuple in D is given by

$$Info(D) = - \sum_{i=1}^m p_i \log_2(p_i),$$

Where p_i is the nonzero probability that an arbitrary tuple in D belongs to class C_i .

Note that, at this point, the information we have is based solely on the proportions of tuples of each class.

- $Info(D)$ is just the average amount of information needed to identify the class label of a tuple in D
- $Info(D)$ is also known as the **entropy** of D .

Now, suppose we were to partition the tuples in D on some attribute A having v distinct values, $\{a_1, a_2, \dots, a_v\}$, as observed from the training data. If A is discrete-valued, these values correspond directly to the v outcomes of a test on A . Attribute A can be used to split D into v partitions or subsets, $\{D_1, D_2, \dots, D_v\}$, where D_j contains those tuples in D that have outcome a_j of A . These partitions would correspond to the branches grown from node N . Ideally, we would like this partitioning to produce an exact classification of the tuples. That is, we would like for each partition to be pure. However, it is quite likely that the partitions will be impure (e.g., where a partition may contain a collection of tuples from different classes rather than from a single class).

How much more information would we still need (after the partitioning) to arrive at an exact classification? This amount is measured by

$$Info_A(D) = \sum_{j=1}^v \frac{|D_j|}{|D|} \times Info(D_j).$$

The term $\frac{|D_j|}{|D|}$ acts as the weight of the j th partition. $Info_A(D)$ is the expected information required to classify a tuple from D based on the partitioning by A . The smaller the expected information (still) required, the greater the purity of the partitions.

Information gain is defined as the difference between the original information requirement (i.e., based on just the proportion of classes) and the new requirement (i.e., obtained after partitioning on A). That is,

$$Gain(A) = Info(D) - Info_A(D).$$

In other words, $Gain(A)$ tells us how much would be gained by branching on A . It is the expected reduction in the information requirement caused by knowing the value of A . The attribute A with the highest information gain, $Gain(A)$, is chosen as the splitting attribute at node N . This is equivalent to saying that we want to partition on the attribute A that would do the “best classification,” so that the amount of information still required to finish classifying the tuples is minimal (i.e., minimum $Info_A(D)$).

Induction of a decision tree using information gain

Example

Class-Labeled Training Tuples from the *AllElectronics* Customer Database

RID	age	income	student	credit_rating	Class: buys_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

- The presents a training set, D , of class-labeled tuples randomly selected from the *AllElectronics* customer database.
- In this example, each attribute is discrete valued.
- The class label attribute, *buys computer*, has two distinct values (namely, {yes, no}); therefore, there are two distinct classes (i.e., $m = 2$).
- Let class C_1 correspond to yes and class C_2 correspond to no.

There are nine tuples of class *yes* and five tuples of class *no*. A (root) node *N* is created for the tuples in *D*. To find the splitting criterion for these tuples, we must compute the information gain of each attribute. We first use Equation given below.

$$Info(D) = - \sum_{i=1}^m p_i \log_2(p_i),$$

To compute the expected information needed to classify a tuple in *D*:

$$Info(D) = -\frac{9}{14} \log_2 \left(\frac{9}{14} \right) - \frac{5}{14} \log_2 \left(\frac{5}{14} \right) = 0.940 \text{ bits.}$$

Next, we need to compute the expected information requirement for each attribute. Let's start with the attribute *age*.

We need to look at the distribution of *yes* and *no* tuples for each category of *age*.

For the *age* category “youth,” there are two *yes* tuples and three *no* tuples.

For the category “middle aged,” there are four *yes* tuples and zero *no* tuples.

For the category “senior,” there are three *yes* tuples and two *no* tuples.

The expected information needed to classify a tuple in *D* if the tuples are partitioned according to *age* is

(Hint: use equation)

$$Info_A(D) = \sum_{j=1}^v \frac{|D_j|}{|D|} \times Info(D_j).$$

$$\begin{aligned} Info_{age}(D) &= \frac{5}{14} \times \left(-\frac{2}{5} \log_2 \frac{2}{5} - \frac{3}{5} \log_2 \frac{3}{5} \right) + \frac{4}{14} \times \left(-\frac{4}{4} \log_2 \frac{4}{4} \right) \\ &\quad + \frac{5}{14} \times \left(-\frac{3}{5} \log_2 \frac{3}{5} - \frac{2}{5} \log_2 \frac{2}{5} \right) \\ &= 0.694 \text{ bits.} \end{aligned}$$

Hence, the gain in information from such a partitioning would be

$$\text{Gain}(\text{age}) = \text{Info}(D) - \text{Info}_{\text{age}}(D) = 0.940 - 0.694 = 0.246 \text{ bits.}$$

Similarly, we can compute

$\text{Gain}(\text{income}) = 0.029$ bits,

$\text{Gain}(\text{student}) = 0.151$ bits, and

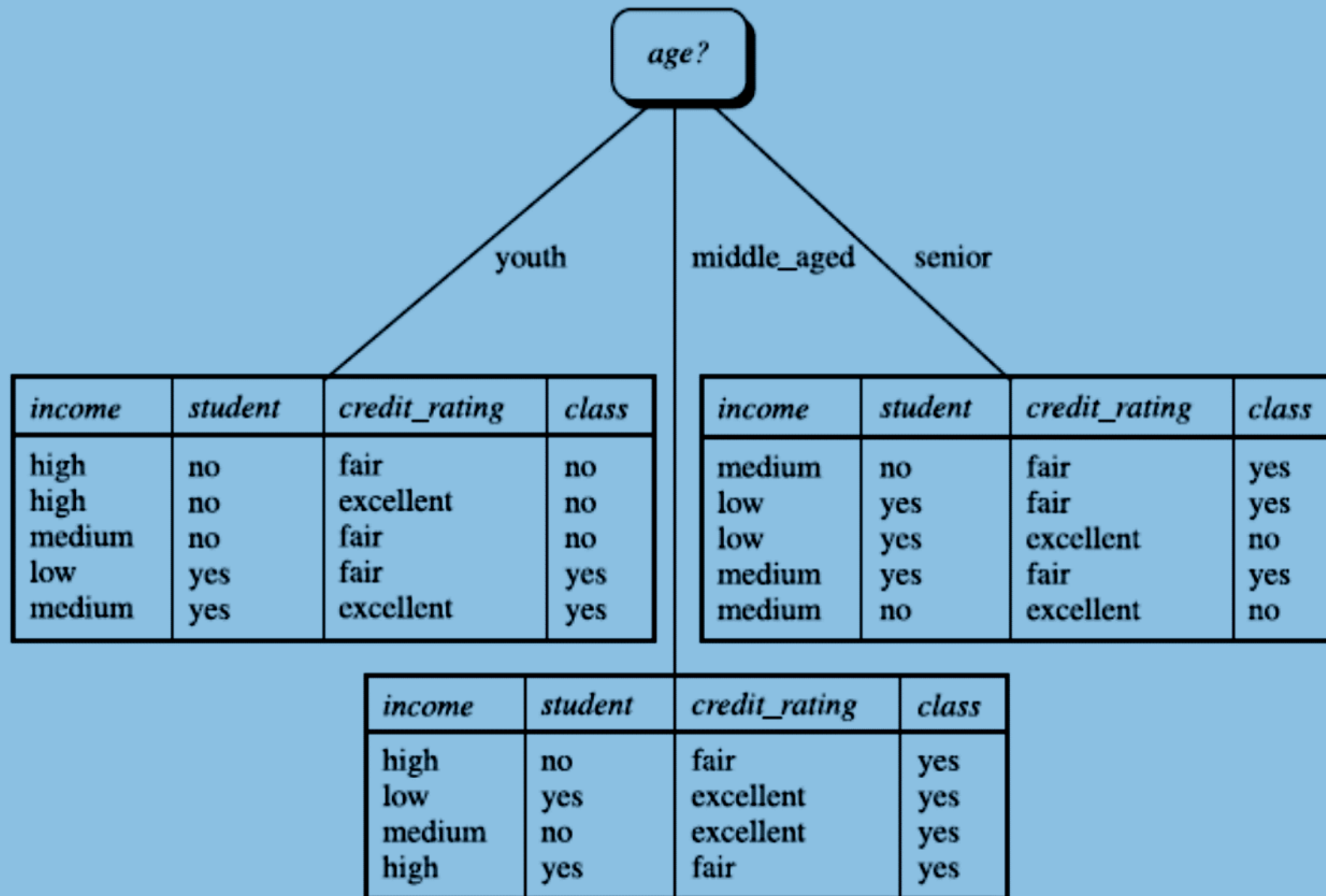
$\text{Gain}(\text{credit rating}) = 0.048$ bits.

Because *age* has the highest information gain among the attributes, it is selected as the splitting attribute.

Node *N* is labeled with *age*, and branches are grown for each of the attribute's values. The tuples are then partitioned accordingly, as shown in Figure below.

Notice that the tuples falling into the partition for *age* *D middle aged* all belong to the same class.

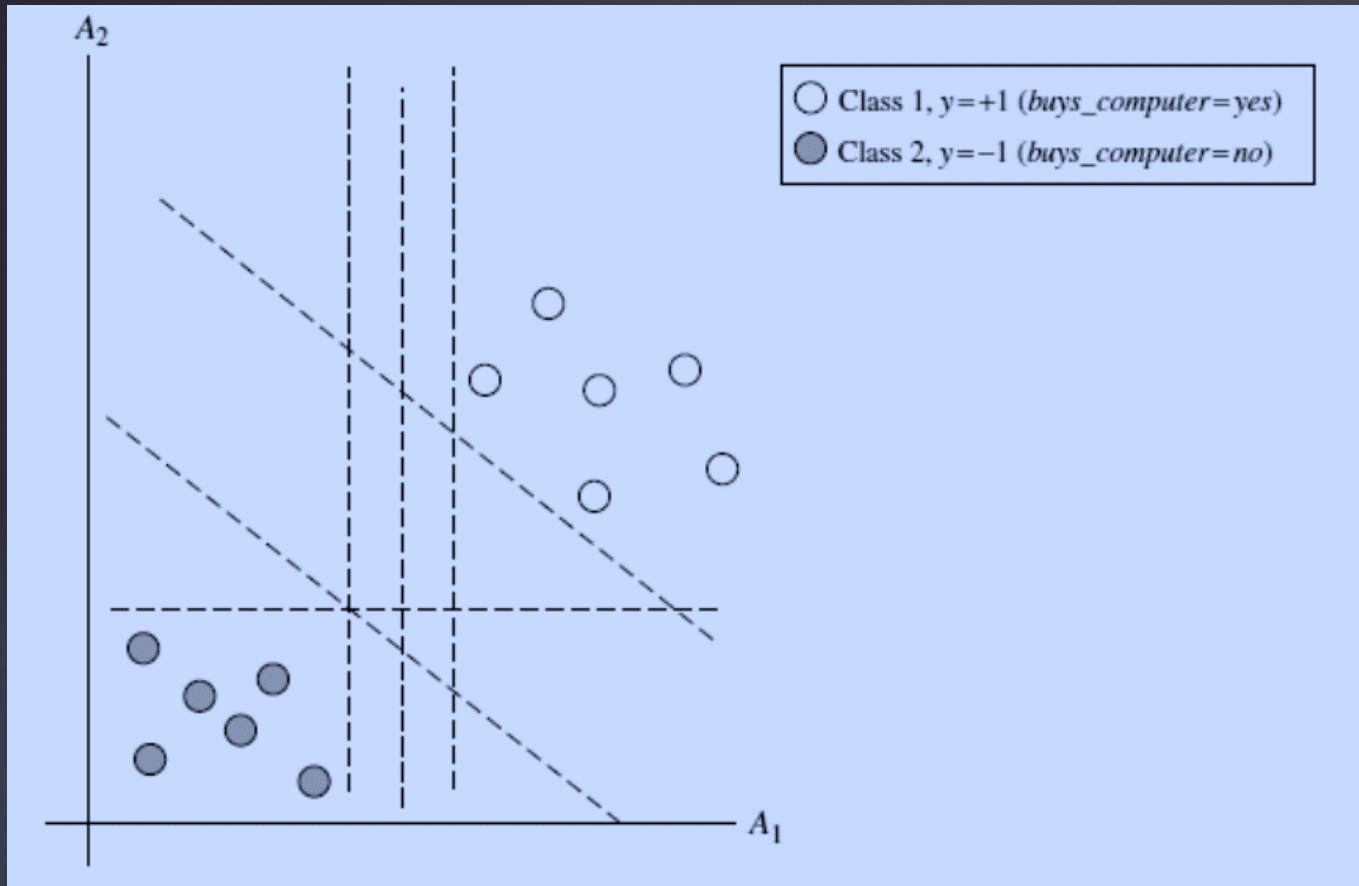
Because they all belong to class “yes” a leaf should therefore be created at the end of this branch and labeled “yes”



- The attribute *age* has the highest information gain and therefore becomes the splitting attribute at the root node of the decision tree.
- Branches are grown for each outcome of *age*.
- The tuples are shown partitioned accordingly.

Support Vector Machines (SVM)

- ❑ SVM is a non-probabilistic linear classification approach that chooses an optimal separating hyperplane such that it maximizes the distance between data points of different classes.
- ❑ To construct the best solution for separating hyperplane, training data points are used which are considered as vectors or support vectors.
- ❑ These support vectors help in determining the width of the hyperplane. SVM can be extended for the cases where data is not separated by a hard margin.
- ❑ Hence, a trade-off parameter is used which allows margin to be flexible and separates non linearly separable data. In addition to that, SVM can be used to classify nonlinear data using kernel trick.
- ❑ This is the reason behind extensive use to SVM as it achieves to an optimal solution for both linear and nonlinear data.



The 2-D training data are linearly separable.

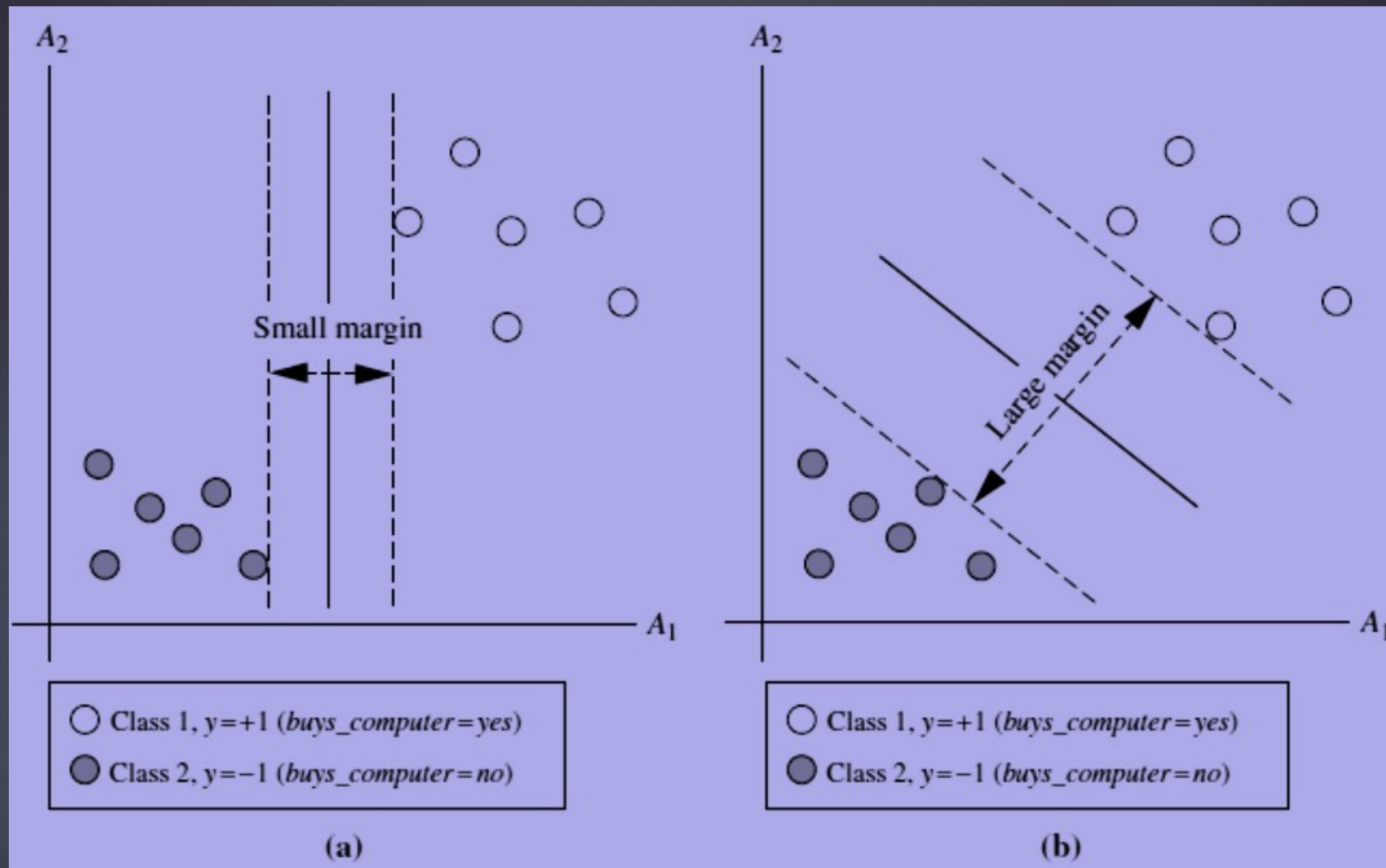
There are an infinite number of possible separating hyperplanes or “decision boundaries,” some of which are shown here as dashed lines.

Which one is best?

Generalizing to n dimensions, we want to find the best *hyperplane*.

We will use “hyperplane” to refer to the decision boundary that we are seeking, regardless of the number of input attributes.

- How can we find the best hyperplane?
- An SVM approaches this problem by searching for the maximum marginal hyperplane.



A separating hyperplane can be written as $W \cdot X + b = 0$,

Where W is a weight vector, $W = \{w_1, w_2, \dots, w_n\}$

n is the number of attributes; and b is a scalar, often referred to as a bias.

Training tuples are 2-D, e.g., $X = (x_1, x_2)$, where x_1 and x_2 are the values of attributes A_1 and A_2 , respectively, for X . If we think of b as an additional weight, w_0 , we can rewrite hyperplane as

$$w_0 + w_1x_1 + w_2x_2 = 0.$$

Thus, any point that lies above the separating hyperplane satisfies

$$w_0 + w_1x_1 + w_2x_2 > 0.$$

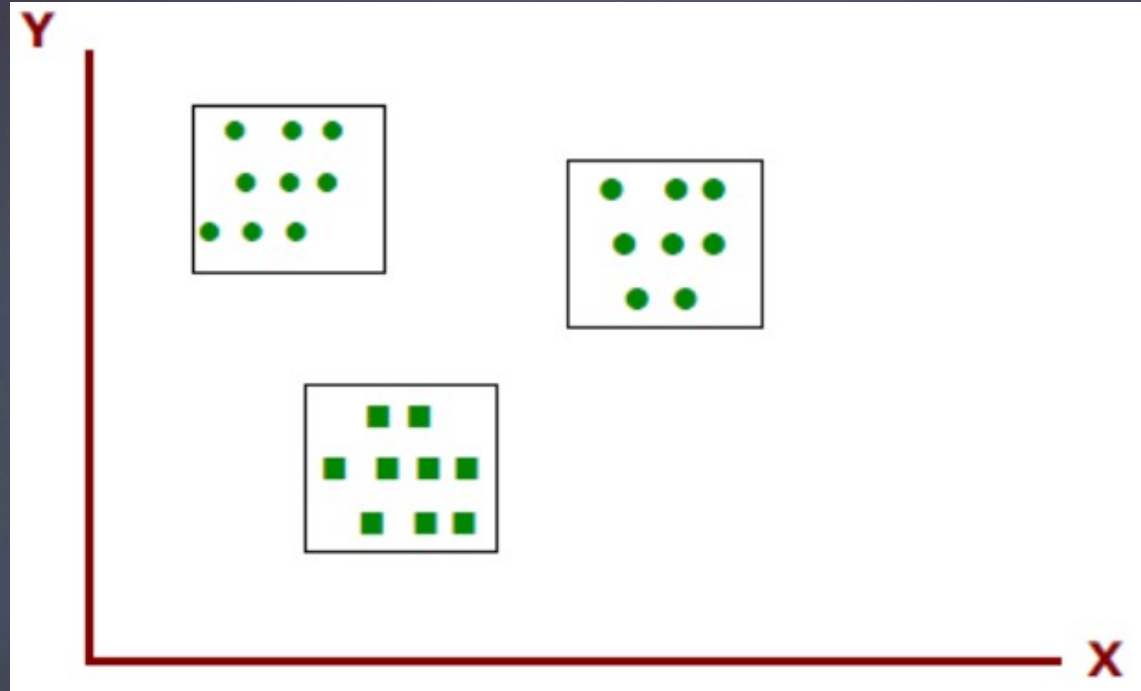
Similarly, any point that lies below the separating hyperplane satisfies

$$w_0 + w_1x_1 + w_2x_2 < 0.$$

Clustering

Clustering is a way to form **natural groupings** or clusters of patterns.
Clustering is often called an **unsupervised learning**.

Example : Three natural groups of data points, i.e., 3 natural clusters.



Example : Three natural groups of data points, i.e., 3 natural clusters.

k-Means Clustering

k-Means Clustering

- ❑ k-means clustering is an algorithm to classify or to group objects based on attributes/features into K number of group.
- ❑ k is positive integer number.
- ❑ The grouping is done by minimizing the sum of squares of distances between data and the corresponding cluster centroid.

Steps involved in k-means clustering

Step 1: Select the number of clusters you want to identify in the data (k)

step 2: Randomly select k distinct data points as initial cluster centre (Centroid).

step 3: Group remaining data points to the cluster centres (Centroids) based on shortest distance criteria.

step 5: Calculate mean of each clusters and rearrange the cluster centres according to new mean.

step 6: Re-cluster the remaining datapoints according to new centroids

step 7: Repeat 5 and 6 since clustering did not change at all during the last iteration.

K-means clustering

Algorithm: k -means. The k -means algorithm for partitioning, where each cluster's center is represented by the mean value of the objects in the cluster.

Input:

- k : the number of clusters,
- D : a data set containing n objects.

Output: A set of k clusters.

Method:

- (1) arbitrarily choose k objects from D as the initial cluster centers;
- (2) **repeat**
- (3) (re)assign each object to the cluster to which the object is the most similar,
 based on the mean value of the objects in the cluster;
- (4) update the cluster means, that is, calculate the mean value of the objects for
 each cluster;
- (5) **until** no change;

Hierarchical Methods

In some situations we may want to partition our data into groups at different levels such as in a hierarchy.

Hierarchical clustering method - works by grouping data objects into a hierarchy or “tree” of clusters.

Representing data objects in the form of a hierarchy is useful for data summarization and visualization.

Example 1:

Organize employees in a Company into major groups such as executives, managers, and staff. You can further partition these groups into smaller subgroups. For instance, the general group of staff can be further divided into subgroups of senior officers, officers, and trainees. All these groups form a hierarchy.

Advantage: We can easily summarize or characterize the data that are organized into a hierarchy, which can be used to find, say, the average salary of managers and of officers.

Hierarchical Agglomerative Clustering

Example 2: Consider handwritten character recognition as another example. A set of handwriting samples may be first partitioned into general groups where each group corresponds to a unique character. Some groups can be further partitioned into subgroups since a character may be written in multiple substantially different ways. If necessary, the hierarchical partitioning can be continued recursively until a desired granularity is reached.

Click to add text

A hierarchical clustering method can be either *agglomerative or divisive*, depending on whether the hierarchical decomposition is formed in a **bottom-up (merging)** or top down (splitting) fashion.

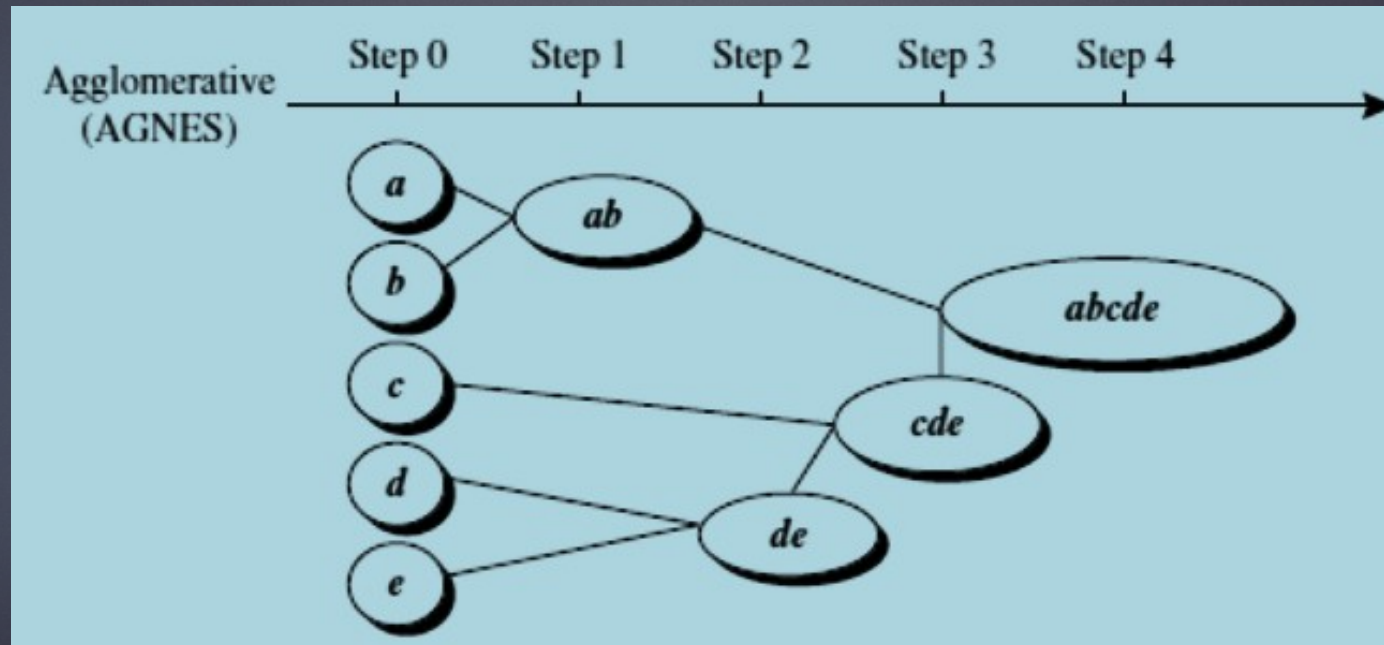
Agglomerative hierarchical clustering:


An agglomerative hierarchical clustering method uses a bottom-up strategy.

It typically starts by letting each object form its own cluster and iteratively merges clusters into larger and larger clusters, until all the objects are in a single cluster or certain termination conditions are satisfied.

Agglomerative hierarchical clustering

- ❑ The single cluster becomes the hierarchy's root.
- ❑ For the merging step, it finds the two clusters that are closest to each other (according to some similarity measure), and combines the two to form one cluster.
- ❑ Because two clusters are merged per iteration, where each cluster contains at least one object, an agglomerative method requires at most n iterations.



- 
- ❑ Figure shows the application of **AGNES** (**AG**glomerative **NESting**), an agglomerative hierarchical clustering method, on a data set of five objects, $\{a, b, c, d, e\}$
 - ❑ Initially, AGNES, places each object into a cluster of its own.
 - ❑ The clusters are then merged step-by-step according to some criterion.

For example, clusters C_1 and C_2 may be merged if an object in C_1 and an object in C_2 form the minimum Euclidean distance between any two objects from different clusters. This is a single-linkage approach in that each cluster is represented by all the objects in the cluster, and the similarity between two clusters is measured by the similarity of the closest pair of data points belonging to different clusters.

- ❑ The cluster-merging process repeats until all the objects are eventually merged to form one cluster.

References:

J. Han, J. Pei, & M. Kamber, (2011). Data mining: concepts and techniques. Elsevier.

http://www.myreaders.info/html/soft_computing.html

J-S R Jang and C-T Sun, Neuro-Fuzzy and Soft Computing, Prentice Hall, 1997

S. Rajasekaran and G.A. Vijayalaksmi Pai ,Neural Network, Fuzzy Logic, and Genetic Algorithms - Synthesis and Applications, (2005), Prentice Hall.