Data Science for Engineers

Module III : Machine Learning

By

Dr Sushila Palwe

sushila.palwe@mitwpu.edu.in

Contents:

Machine Learning:

Introduction to machine learning

Supervised and Unsupervised Learning

Splitting datasets: Training and Testing

Regression: Simple Linear Regression

Classification: Naïve Bayes classifier

Clustering: K-means

Evaluating model performance

Python libraries for machine learning.

What you guess(infer) from the following data

RollNo	Practical	DBMS	TOC	Machine Learning	Attend the Orientation at 3.30??
	8.30	10.45	11.45	12.45	
1	A	P	P	P	Yes
2	A	A	A	A	NO
3	A	P	P	P	Yes
4	A	P	P	P	Yes
5	A	A	A	A	NO
6	A	A	A	A	NO
7	P	P	P	P	YES
8	P	P	P	P	Yes
9	A	P	P	P	??

Introduction : What is Machine Learning

"Machine Learning allows the machines to learn and make predictions based on its experience(data)"

Definition by Tom Mitchell (1998):

Machine Learning is the study of algorithms that

- improve their performance P
- at some task T
- with experience E.

A well-defined learning task is given by <P, T, E>.

Defining the Learning Task :Improve on task T, with respect to performance metric P, based on experience E

Q. Define the learning task for Automated handwritten word recognition

T: Recognizing hand-written words

P: Percentage of words correctly classified

E: Database of human-labeled images of handwritten words

Q. Define a learning task for Automated Spam filter

T: Categorize email messages as spam or legitimate.

P: Percentage of email messages correctly classified.

E: Database of emails, some with human-given labels

When Do We Use Machine Learning?

ML is used when:

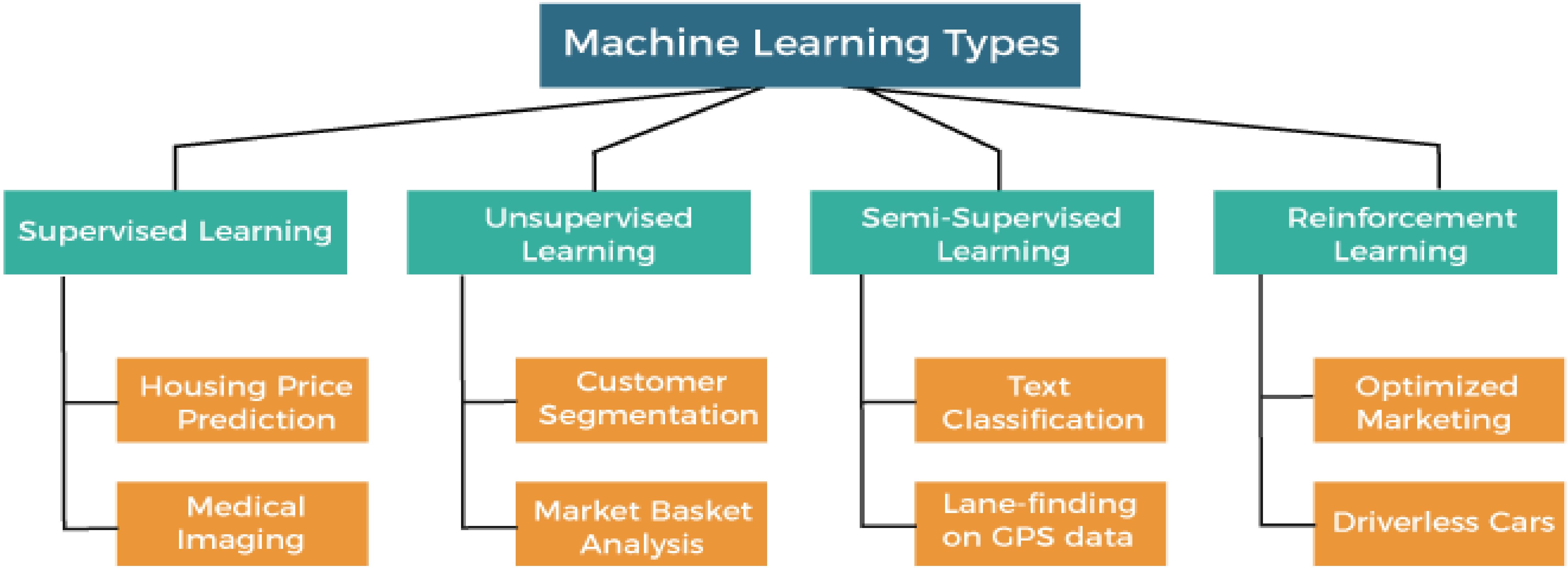
- Human expertise does not exist (navigating on Mars)
- Models must be customized (personalized medicine)
- Models are based on huge amounts of data (genomics)

Learning isn't always useful:

- There is no need to "learn" to calculate payroll

Some more examples of tasks that are best solved by using a learning algorithm

- Recognizing patterns:
  - Handwritten or spoken words
  - Medical images
- Generating patterns:
  - Generating images or motion sequences
- Recognizing anomalies:
  - Unusual credit card transactions
  - Unusual patterns of sensor readings in a nuclear power plant
- Prediction:
  - Future stock prices or currency exchange rates



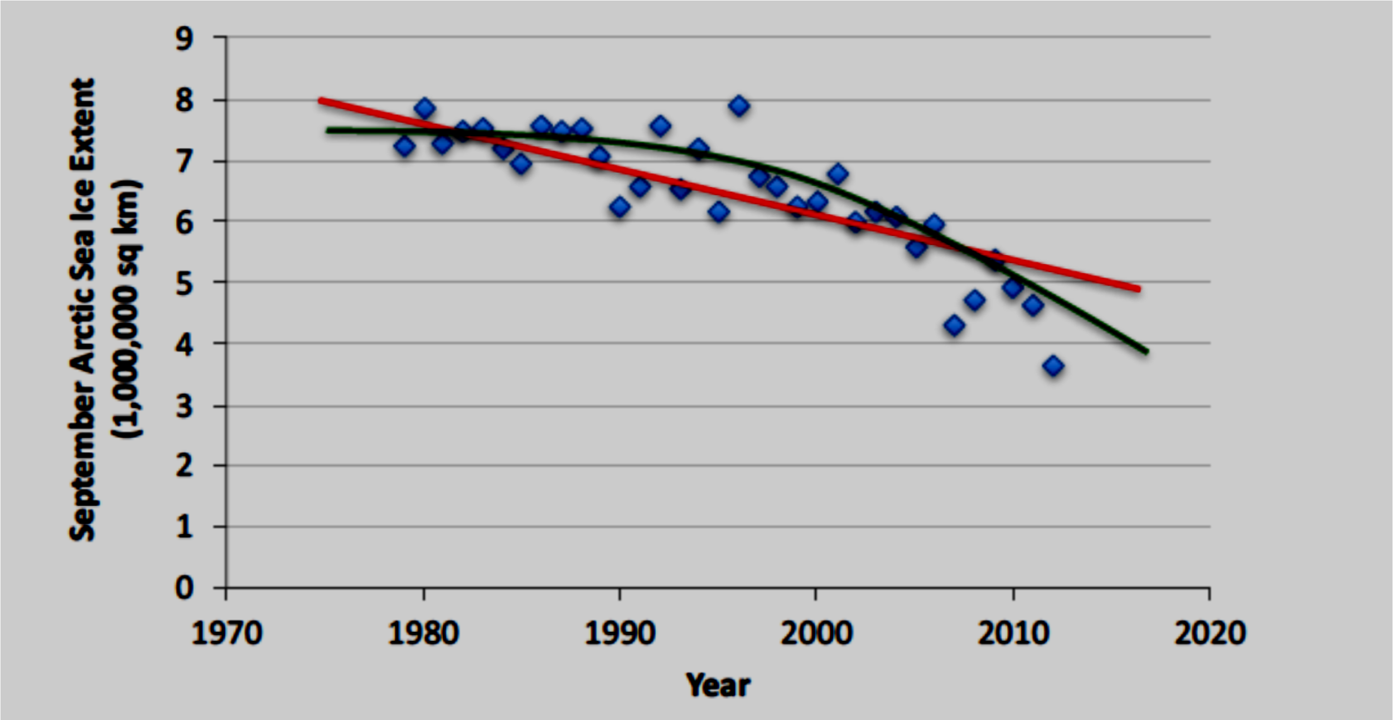
- Regression:
  - Simple Linear Regression Algorithm
  - Multivariate Regression Algorithm
  - Decision Tree Algorithm
  - Lasso Regression
- Classification:
  - Naïve Bayes Classifier Algorithm
  - Random Forest Algorithm
  - Decision Tree Algorithm
  - Logistic Regression Algorithm
  - Support Vector Machine Algorithm

- Clustering
  - K-Means Clustering algorithm
  - Mean-shift algorithm
  - DBSCAN Algorithm
  - Principal Component Analysis
  - Independent Component Analysis
- Association

Supervised Learning: Regression

- Given $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$
- Learn a function $f(x)$ to predict y given x

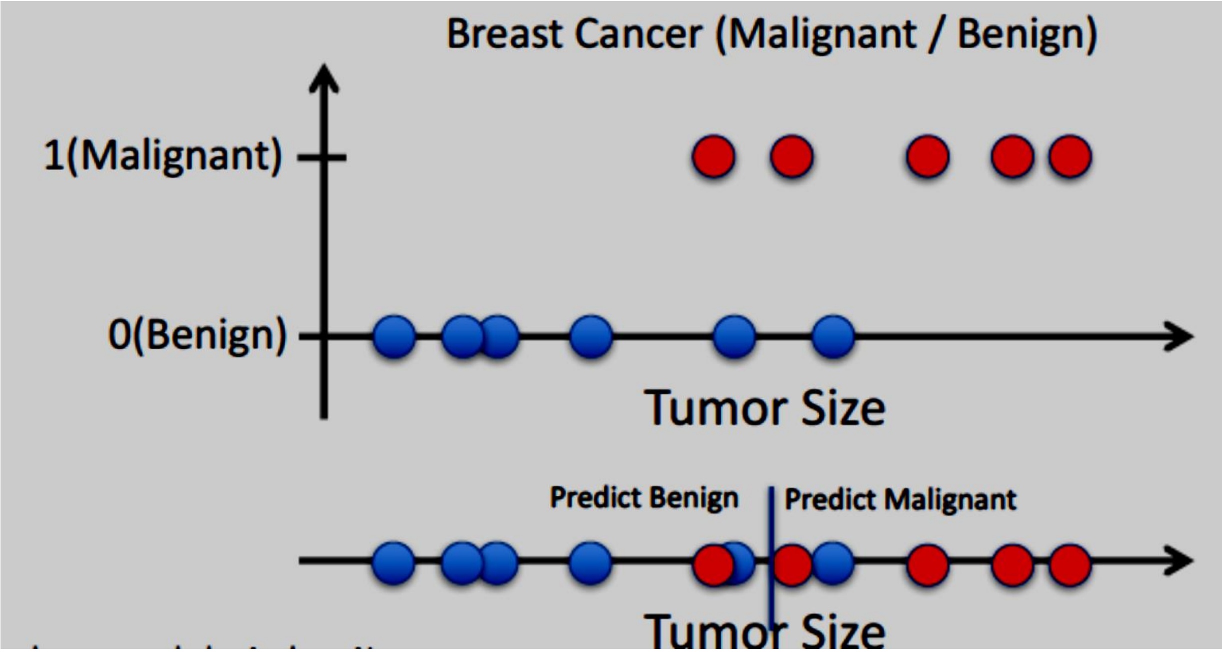
– y is real-valued == regression



Supervised Learning: Classification

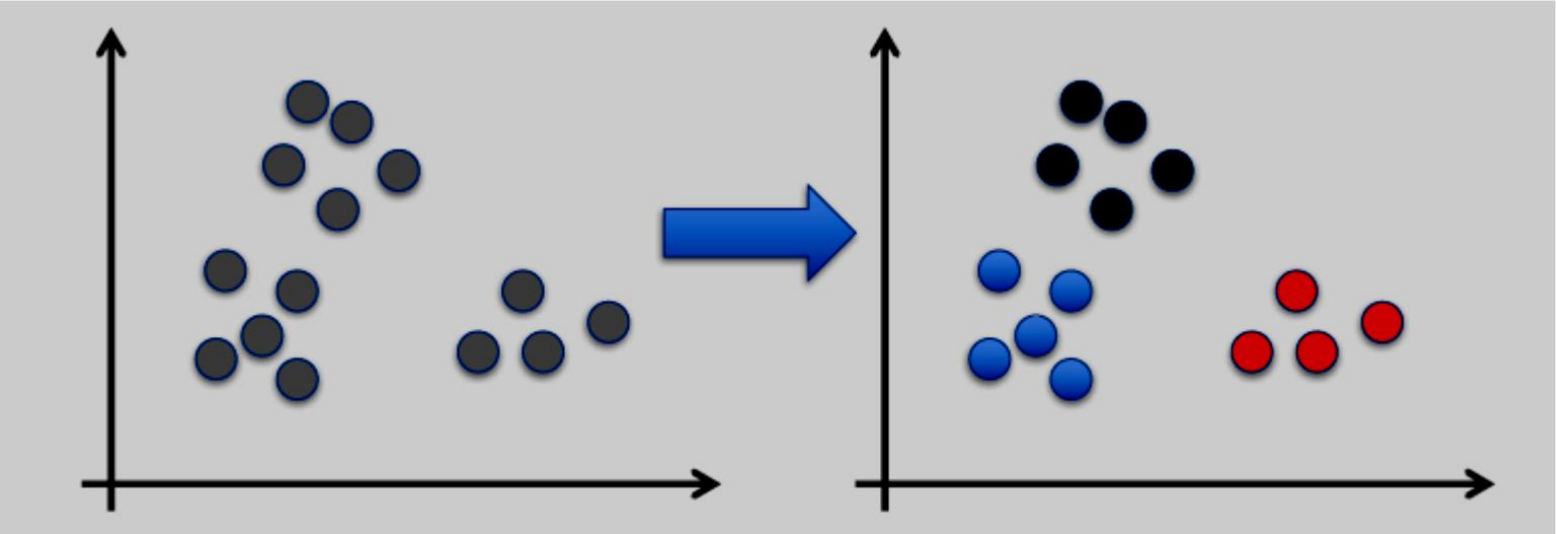
- Given (x1, y1), (x2, y2), ..., (xn, yn)
- Learn a function f(x) to predict y given x

– y is categorical == classification



Unsupervised Learning

- Given x1, x2, ..., xn (without labels)
- Output hidden structure behind the x's
  – E.g., clustering



Python Libraries for Data Science

SciKit-Learn:

- provides machine learning algorithms: classification, regression, clustering, model validation etc.

- built on NumPy, SciPy and matplotlib

Link: http://scikit-learn.org/

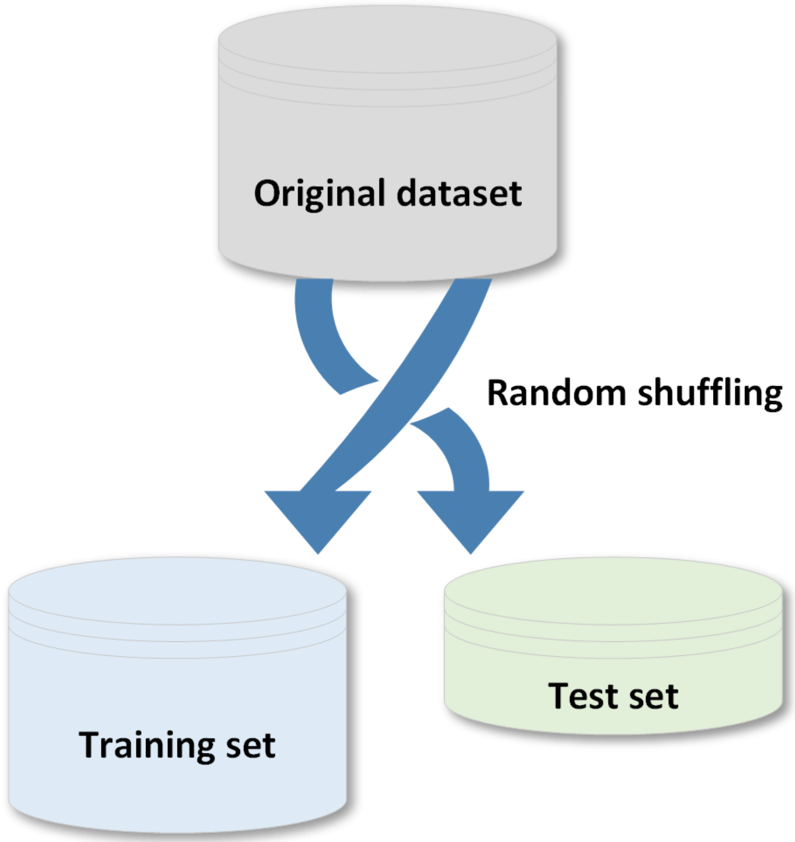
Basic Steps of Machine Learning

- Understand domain, prior knowledge, and goals
- Data integration, selection, cleaning, pre-processing, etc.
- Learn models
- Interpret results
- Consolidate and deploy discovered knowledge

Creating training and test sets

- When a dataset is large enough, it's a good practice to split it into training and test sets; the former to be used for training the model and the latter to test its performances.

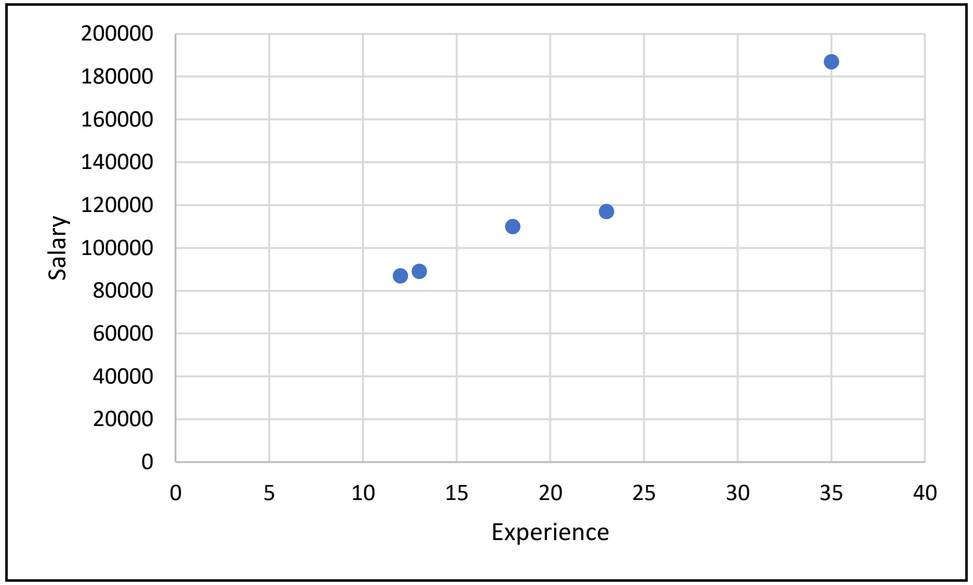
```
from sklearn.model_selection import train_test_split
>>> X_train, X_test, Y_train, Y_test = train_test_split(X, Y,
test_size=0.25, random_state=1)
```

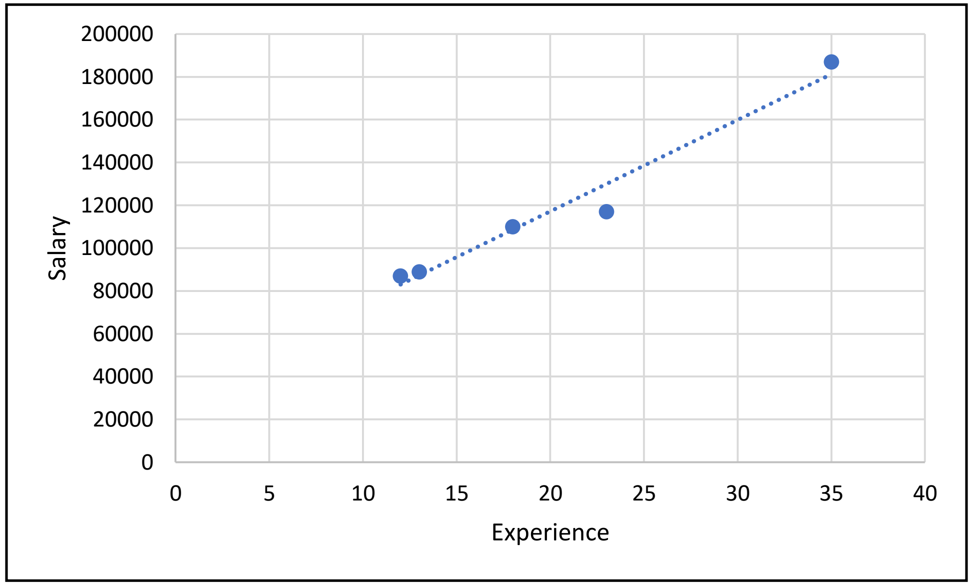


Linear Regression

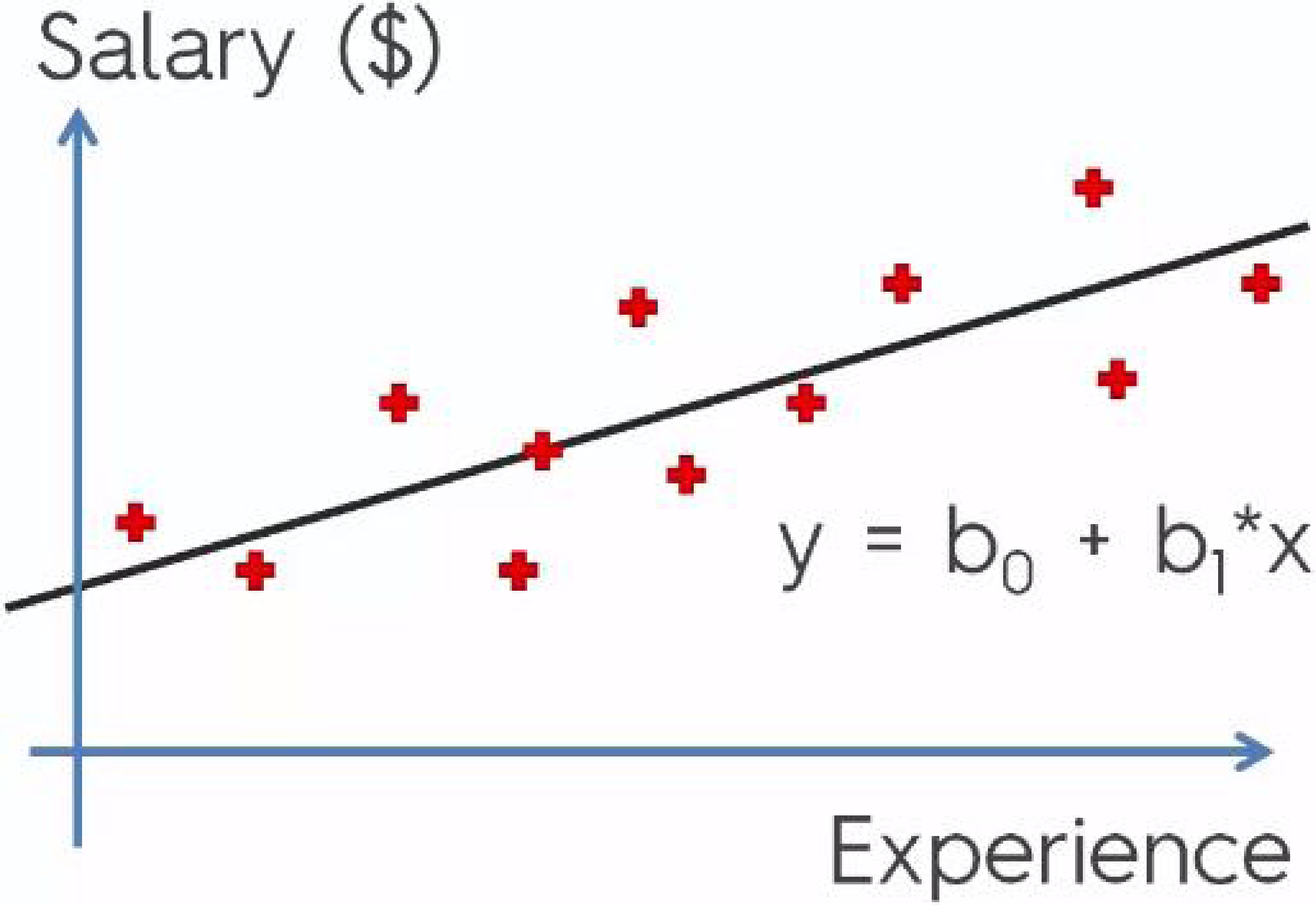
Example

Experience	Salary
12	87000
35	187000
23	117000
13	89000
18	110000





We can solve this using Linear Regression



Contd..

$Y = B_0 + B_1X$

Where,

- Y = Dependent Variable
- X = Independent Variable
- B_0 = Constant term
- B_1 = Coefficient of relationship between 'X' & 'Y' *(B_1 explains the change in Y with a change in X by one unit. In other words, if we increase the value of 'X' by one unit then what will be the change in value of Y)*

Linear Regression

- Regression Analysis can be defined as the process of developing a mathematical model that can be used to predict one variable by another variable.
- Linear regression is supervised machine learning statistical technique wherein we use linear modeling approach to build relationship between dependent variable and independent variable.
- Regression is a statistical way to establish a relationship between a dependent variable and a set of independent variable(s).
- Technique is called Simple linear regression if only one independent variable (x)is analyzed
- In Multiple linear regression, multiple independent variables(x1,x2,..) are analyzed.
- If Multiple dependent variable(y1,y2..) are predicted , it is called as multivariate linear regression.

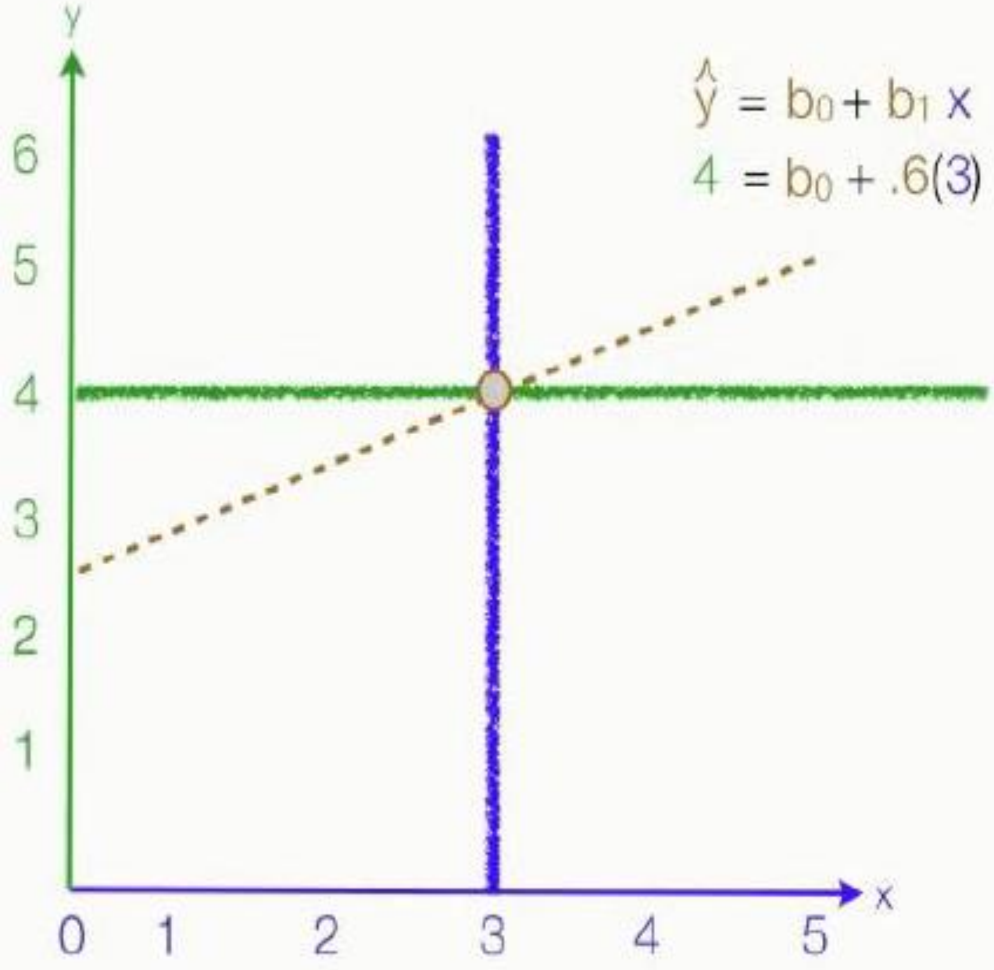
Properties of linear regression line

- Regression line always passes through mean of independent variable (x) as well as mean of dependent variable (y)
- Regression line minimizes the sum of “Square of Residuals”.

- The differences between the actual and estimated function values on the training examp $\epsilon_i = f(x_i) - \hat{f}(x_i).$

$$b_0 = 2.2$$
$$b_1 = .6$$
$$\hat{y} = 2.2 + .6x$$

$$\hat{y} = b_0 + b_1 x$$
$$4 = b_0 + .6(3)$$



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
				10	6
mean 3	4				

$$4 = b_0 + .6(3)$$

$$\begin{array}{rcl} 4 & = & b_0 + \cancel{1.8} \\ -1.8 & & \cancel{-1.8} \\ \hline 2.2 & = & b_0 \end{array}$$

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

Example

- What linear regression equation best predicts statistics performance, based on math aptitude scores?
- If a student made an 80 on the Math test, what grade would we expect her to make in statistics?
- How well does the regression equation fit the data?

RollNo	Maths Marks	Stat Marks
1	95	85
2	85	95
3	80	70
4	70	65
5	60	70

Solution

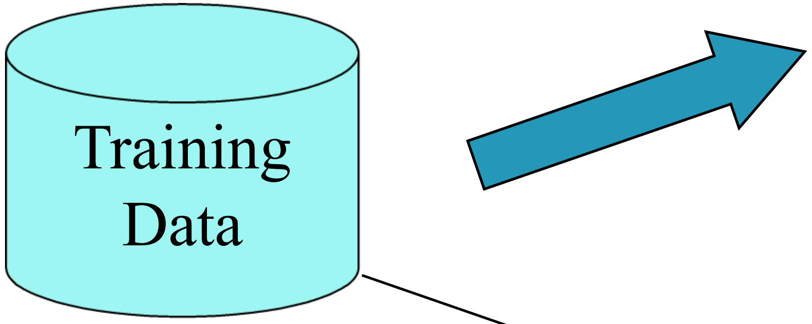
- $b_1 = 0.644$
- $b_0 = 26.768$

Classification

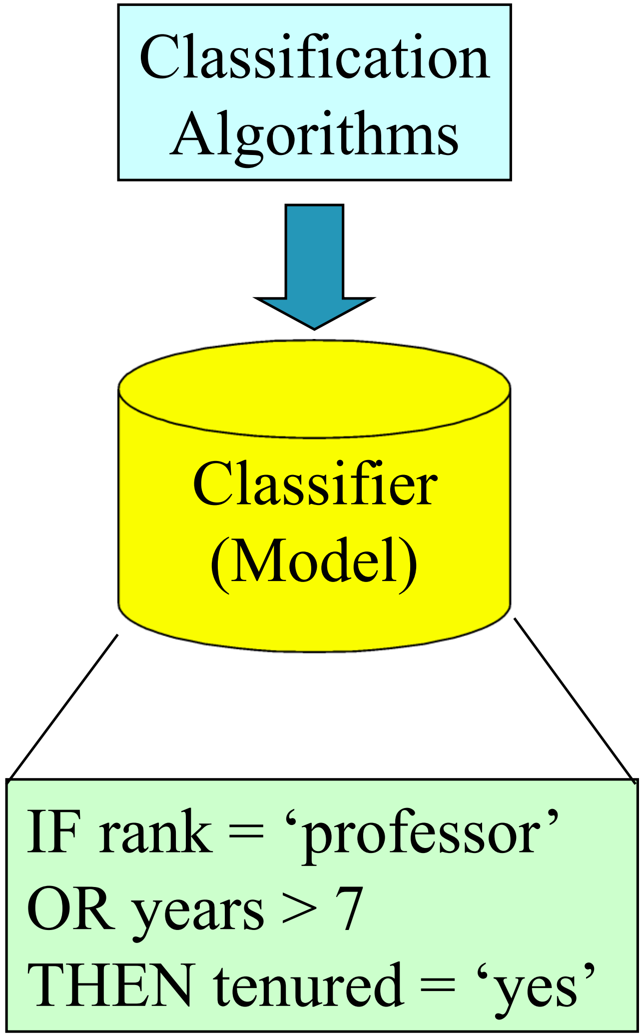
Classification Process

- Model construction: describing a set of predetermined classes
  - Each tuple/sample is assumed to belong to a predefined class, as determined by the class label attribute
  - The set of tuples used for model construction is training set
  - The model is represented as classification rules, decision trees, or mathematical formulae
- Model usage: for classifying future or unknown objects
  - Estimate accuracy of the model
    - The known label of test sample is compared with the classified result from the model
    - Accuracy rate is the percentage of test set samples that are correctly classified by the model
    - Test set is independent of training set, otherwise over-fitting will occur
  - If the accuracy is acceptable, use the model to classify data tuples whose class labels are not known

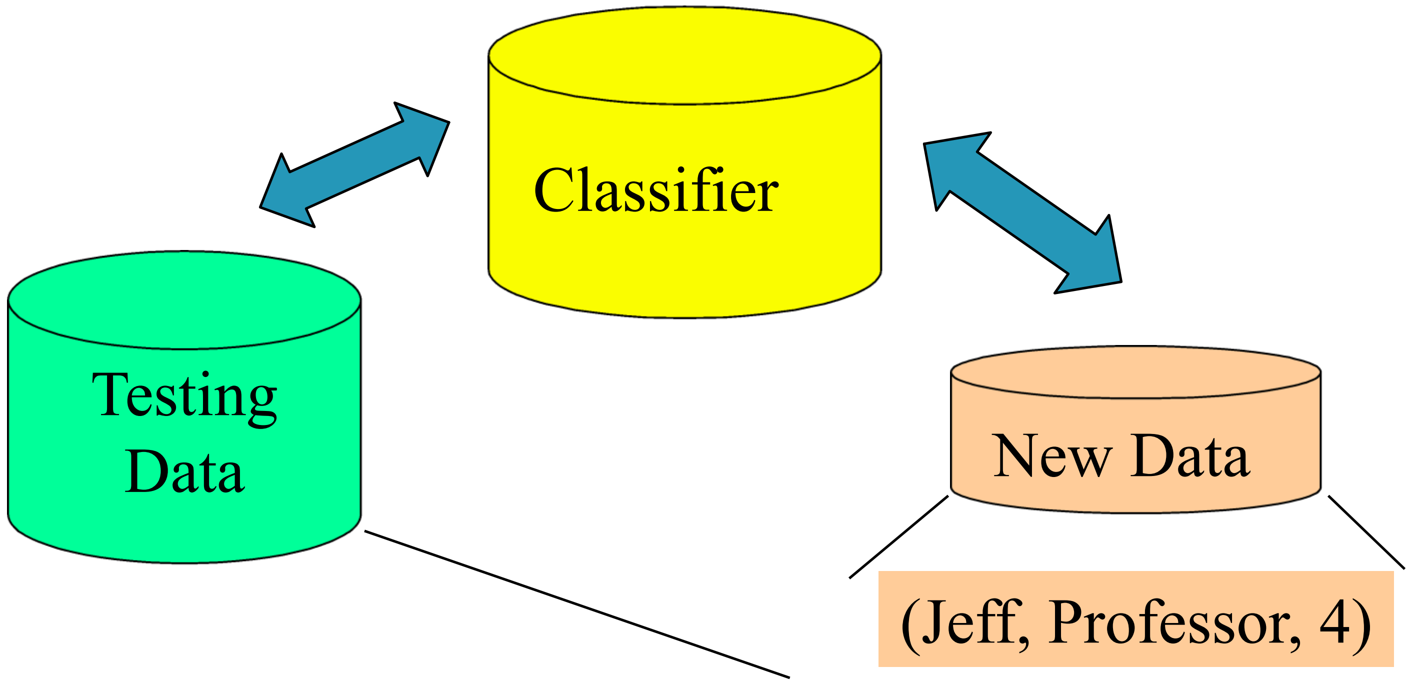
Step 1: Model Construction



NAME	RANK	YEARS	TENURED
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no



Step 2: Model Usage



NAME	RANK	YEARS	TENURED
Tom	Assistant Prof	2	no
Merlisa	Associate Prof	7	no
George	Professor	5	yes
Joseph	Assistant Prof	7	yes

Tenured?

Yes

Bayesian Classification

- A statistical classifier: performs *probabilistic prediction, i.e.,* predicts class membership probabilities
- Foundation: Based on Bayes' Theorem.
- Performance: A simple Bayesian classifier, *naïve Bayesian classifier*, has comparable performance with decision tree and selected neural network classifiers

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

X = (age =31..40, Income = low, Student = yes , Credit_rating= Fair)

Probability

- *Probability:* How likely something is to happen

- Probability of an event happening =

$$\frac{\text{Number of ways it can happen}}{\text{Total number of outcomes}}$$

Bayesian Theorem Basics

- Let **X** be a data sample ("*evidence*"): class label is unknown
- Let H be a *hypothesis* that X belongs to class C

$$P(H \mid \mathbf{X}) = \frac{P(\mathbf{X} \mid H) P(H)}{P(\mathbf{X})}$$

- Classification is to determine P(H|**X**), the probability that the hypothesis holds given the observed data sample **X**
- P(H) (*prior probability*), the initial probability
  - E.g., **X** will buy computer, regardless of age, income, …
- P(**X**): probability that sample data is observed
- P(**X**|H) (*posteriori probability*), the probability of observing the sample **X**, given that the hypothesis holds
  - E.g., Given that **X** will buy computer, the prob. that X is 31..40, medium income

Bayesian Theorm

- Given training data **X**, *posteriori probability of a hypothesis* H, P(H|**X**), follows the Bayes theorem

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

- Informally, this can be written as

  posteriori = likelihood x prior/evidence

- ***Predicts X belongs to C_2 iff the probability $P(C_i|X)$ is the highest among all the $P(C_k|X)$ for all the k classes***
- Practical difficulty: require initial knowledge of many probabilities, significant computational cost

Naiive Bayesian

- Let D be a training set of tuples and their associated class labels, and each tuple is represented by an n-D attribute vector **X** = $(x_1, x_2, \ldots, x_n)$
- Suppose there are *m* classes $C_1, C_2, \ldots, C_m$.
- Classification is to derive the maximum posteriori, i.e., the maximal $P(C_i|\mathbf{X})$
- This can be derived from Bayes' theorem

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

- Since P(X) is constant for all classes, only

$$P(C_i|\mathbf{X}) = P(\mathbf{X}|C_i)P(C_i)$$

needs to be maximized

Training Data

Class:
C1:buys_computer = 'yes'
C2:buys_computer = 'no'

Data sample
X = (age =31..40,
Income = low,
Student = yes
Credit_rating = fair)

$$P(C_i|\mathbf{X}) = P(\mathbf{X}|C_i)P(C_i)$$

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Test for X = (age <= 30 , income = medium, student = yes, credit_rating = fair)

- $P(C_i)$: **P(buys_computer = "yes") = 9/14 = 0.643** **P(buys_computer = "no") = 5/14= 0.357**
- Compute $P(X|C_i)$ for each class

P(age = "<=30" | buys_computer = "yes") = 2/9 = 0.222

P(age = "<=30" | buys_computer = "no") = 3/5 = 0.6

P(income = "medium" | buys_computer = "yes") = 4/9 = 0.444

P(income = "medium" | buys_computer = "no") = 2/5 = 0.4

P(student = "yes" | buys_computer = "yes) = 6/9 = 0.667

P(student = "yes" | buys_computer = "no") = 1/5 = 0.2

P(credit_rating = "fair" | buys_computer = "yes") = 6/9 = 0.667

P(credit_rating = "fair" | buys_computer = "no") = 2/5 = 0.4

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

- **$P(X|C_i)$:** P(X|buys_computer = "yes") = 0.222 x 0.444 x 0.667 x 0.667 = 0.044

  P(X|buys_computer = "no") = 0.6 x 0.4 x 0.2 x 0.4 = 0.019

$P(X|C_i)*P(C_i)$: P(X|buys_computer = "yes") * P(buys_computer = "yes") = 0.028

P(X|buys_computer = "no") * P(buys_computer = "no") = 0.007

0.028>0.007 ..
Therefore, X belongs to class ("buys_computer = yes")

Clustering

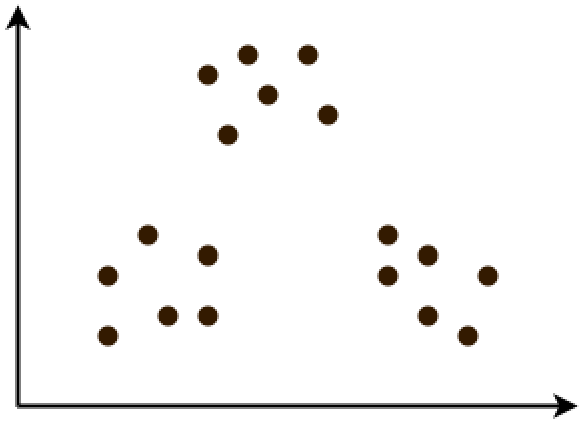
Unsupervised learning: no predefined classes (i.e., learning by observations vs. learning by examples: supervised)

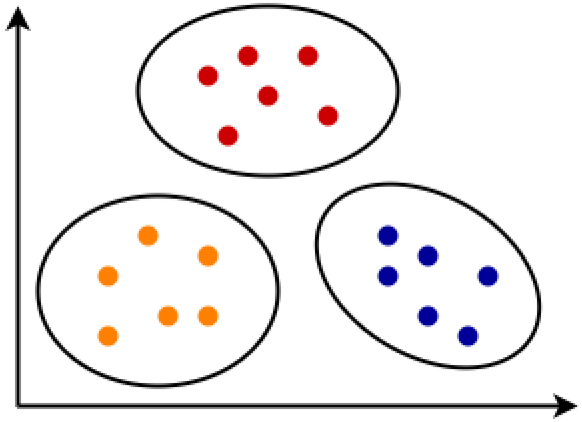
Cluster: A collection of data objects

- similar (or related) to one another within the same group dissimilar (or unrelated) to the objects in other groups

Cluster analysis (or clustering, data segmentation, …)

- Finding similarities between data according to the characteristics found in the data and grouping similar data objects into clusters





Similarity Measure

Dissimilarity/Similarity metric

Similarity is expressed in terms of a distance function, typically metric: *d*(*i, j*)

Distance Calculation

- Distance of Data from each centroid can be calculated using following distance functions

Euclidean $$\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$$

Manhattan $$\sum_{i=1}^{k}|x_i - y_i|$$

Minkowski $$\left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{1/q}$$

K Means Algorithm

The K-means algorithm starts by randomly choosing a centroid value for each cluster.

After that the algorithm iteratively performs three steps:

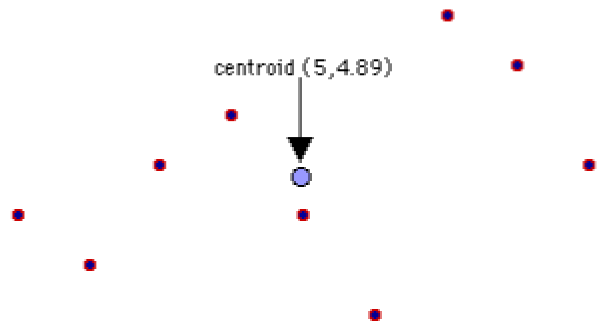
(i) Find the Euclidean distance between each data instance and centroids of all the clusters;

(ii) Assign the data instances to the cluster of the centroid with nearest distance;

(iii) Calculate new centroid values based on the mean values of the coordinates of all the data instances from the corresponding cluster.

Centroid of the Cluster

Centroid Calculation Function



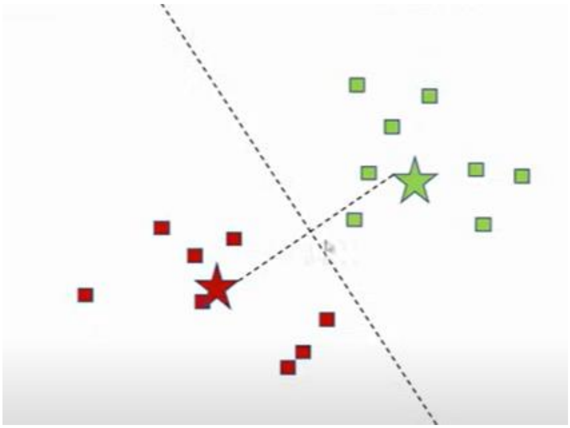
The centroid of the points $(x_1, y_1), (x_2, y_2), (x_3, y_3), \ldots, (x_n, y_n)$ is

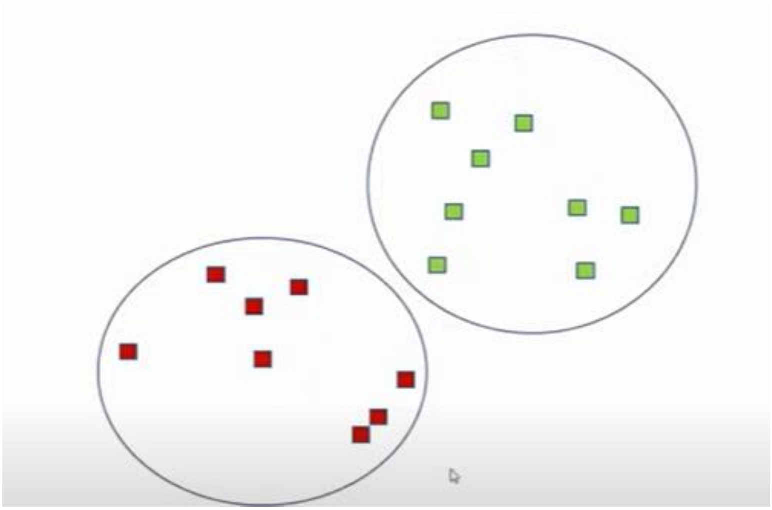
$$\left(\frac{x_1 + x_2 + x_3 + \cdots + x_n}{n}, \frac{y_1 + y_2 + y_3 + \cdots + y_n}{n}\right)$$

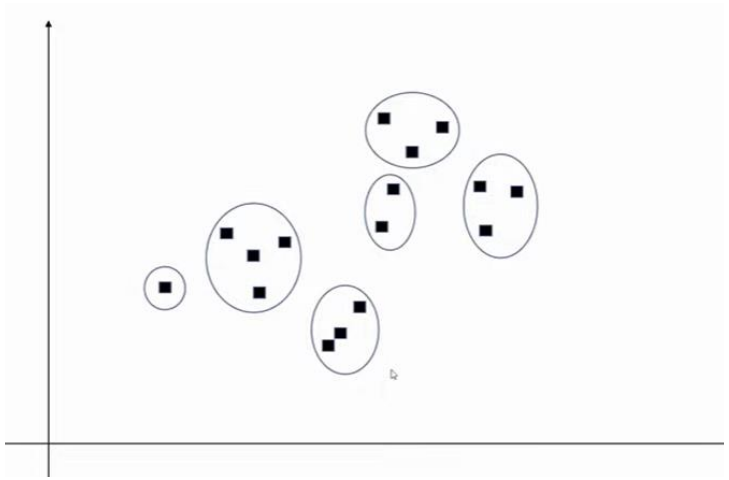
Where n is the number of stored points in your system.

points having 2-dimensional coordinates (x and y)

how







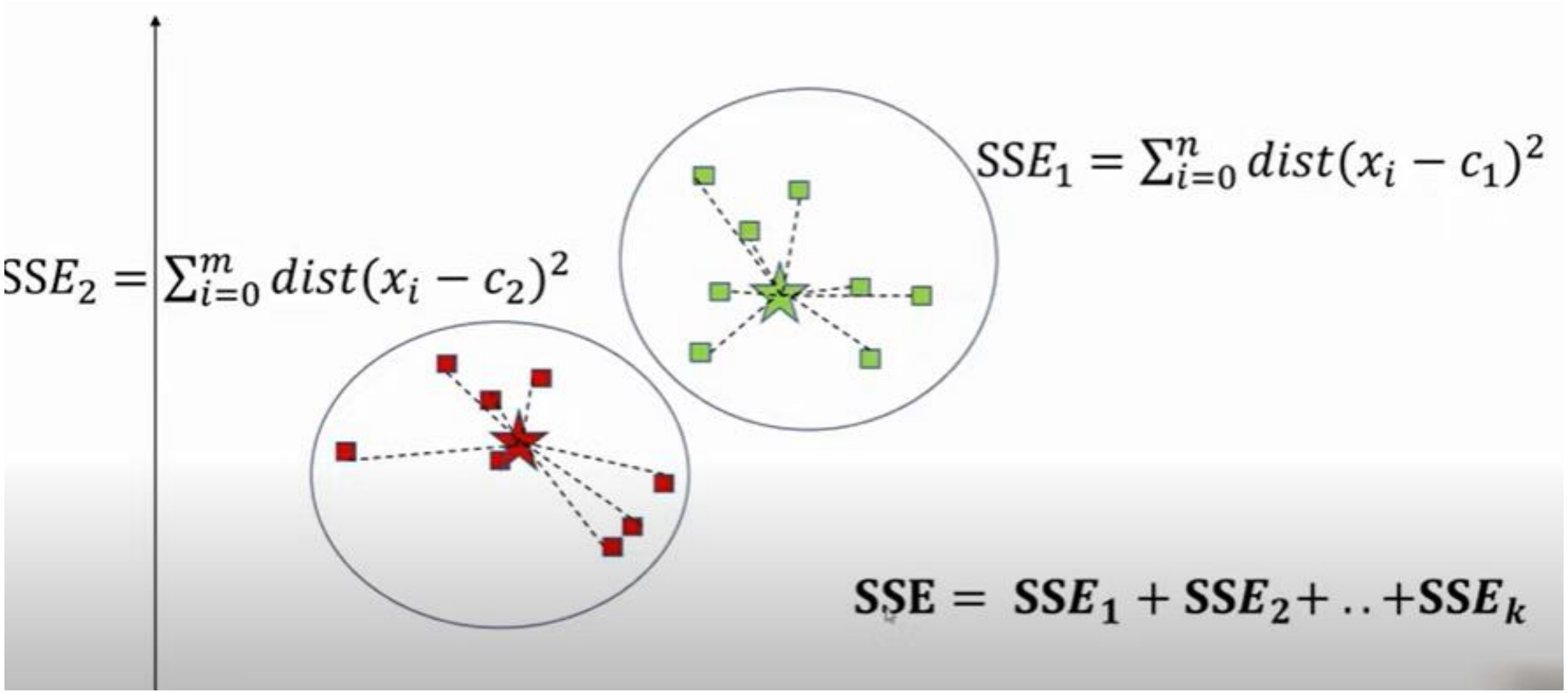
Assign Cluster Centroids
  Until Convergence: Cluster Assignment Step
  Re-assigning Centroid Step

To define, the number of Clusters:

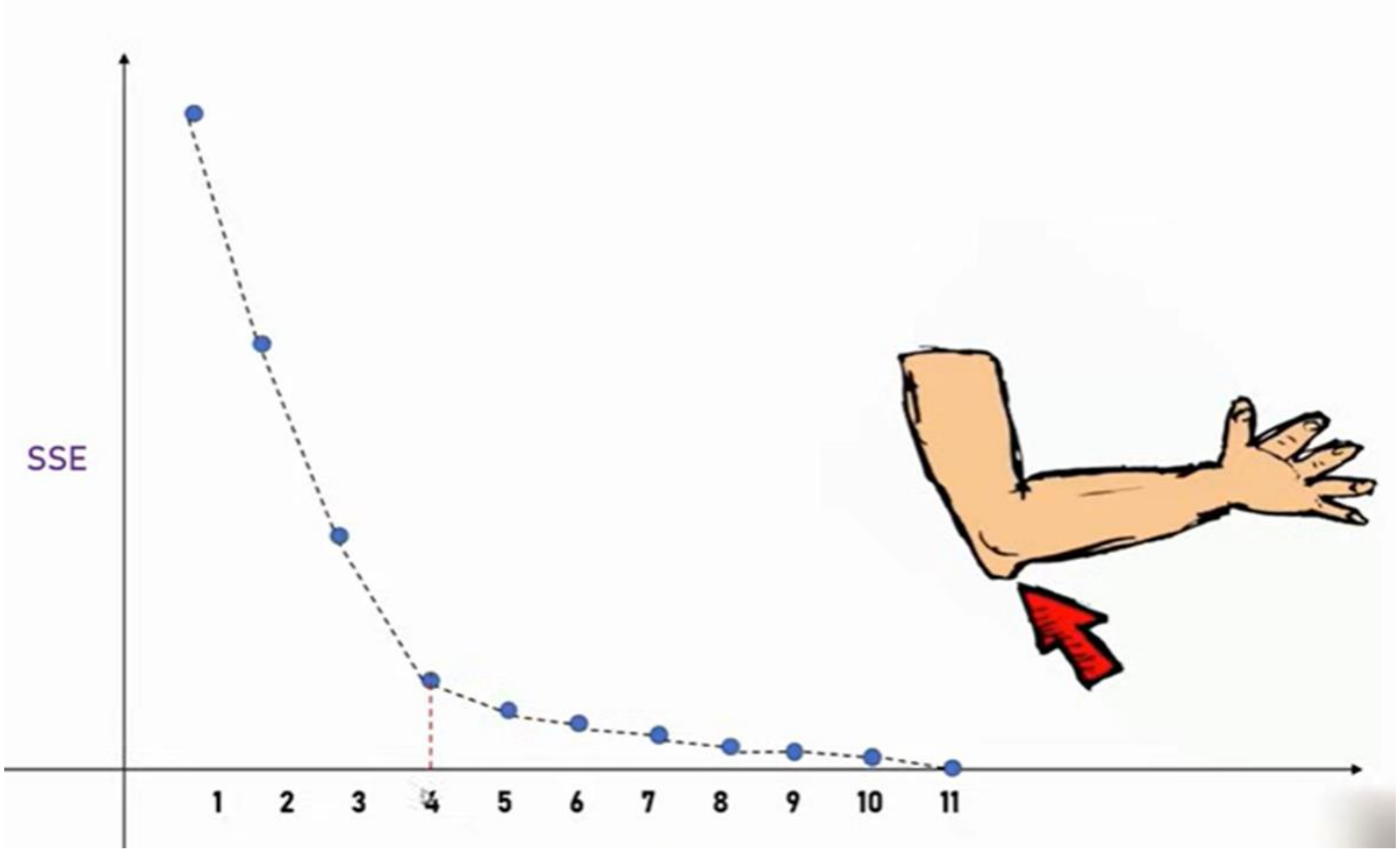
- Elbow Method
- Silhouette Method

Elbow Method:

- Assume no. of clusters k = 2, 3, 4,……..
- Calculate the sum of squared errors (SSE) for each k.
- Plot k versus SSE and find out elbow.



Elbow Method



K Means Case Study: Cluster the data (2,4,4,4,6,6) in 2 Clusters

The centroid of the points $(x_1, y_1), (x_2, y_2), (x_3, y_3), \ldots, (x_n, y_n)$ is

$$\left(\frac{x_1 + x_2 + x_3 + \cdots + x_n}{n}, \frac{y_1 + y_2 + y_3 + \cdots + y_n}{n}\right)$$

Manhattan $\sum_{i=1}^{k} |x_i - y_i|$

X	C1=4	C2=6	Cluser Labels	NewCentroid1 (3.5)	NewCentroid2(6)	Cluser Labels	
2	4-2=2	6-2=4	C1	3.5-2=1.5	6-2=4	C1	
4	4-4=0	6-4=2	C1	3.5-4=0.5	6-4=2	C1	
4	4-4=0	6-4=2	C1	3.5-4=0.5	6-4=2	C1	**NO CHANGE**
4	4-4=0	6-4=2	C1	3.5-4=0.5	6-4=2	C1	
6	4-6=2	6-6=0	C2	3.5-6=2.5	6-6=0	C2	
6	4-6=2	6-6=0	C2	3.5-6=2.5	6-6=0	C2	

NewCentroid1 = (2+4+4+4)/4 =3.5

NewCentroid1 = (6+6)/2 =6

Evaluating Model Performance

Confusion Matrix : Classifier Accuracy Measure

Actual class\Predicted class	C_1	$\neg C_1$
C_1	**True Positives (TP)**	**False Negatives (FN)**
$\neg C_1$	**False Positives (FP)**	**True Negatives (TN)**

Example of Confusion Matrix:

Actual class\Predicted class	buy_computer = yes	buy_computer = no	Total
buy_computer = yes	**6954**	**46**	7000
buy_computer = no	**412**	**2588**	3000
Total	7366	2634	10000

- Given *m* classes, an entry, $CM_{i,j}$ in a **confusion matrix** indicates # of tuples in class *i* that were labeled by the classifier as class *j*
- May have extra rows/columns to provide totals

A\P	C	¬C	
C	**TP**	**FN**	**P**
¬C	**FP**	**TN**	**N**
	P'	**N'**	**All**

- **Classifier Accuracy,** or recognition rate: percentage of test set tuples that are correctly classified

  Accuracy = (TP + TN)/All

- **Error rate:** *1 – accuracy*, or

  Error rate = (FP + FN)/All

- **Precision**: exactness – what % of tuples that the classifier labeled as positive are actually positive

$$precision = \frac{TP}{TP + FP}$$

- **Recall:** completeness – what % of positive tuples did the classifier label as positive?
- Perfect score is 1.0

$$recall = \frac{TP}{TP + FN}$$

- ***F* measure (*F_1* or *F-score*)**: harmonic mean of precision and recall,

$$F = \frac{2 \times precision \times recall}{precision + recall}$$

Thank You