Machine Learning & Applications

Nagur Shareef Shaik

What will you learn in these Sessions…?

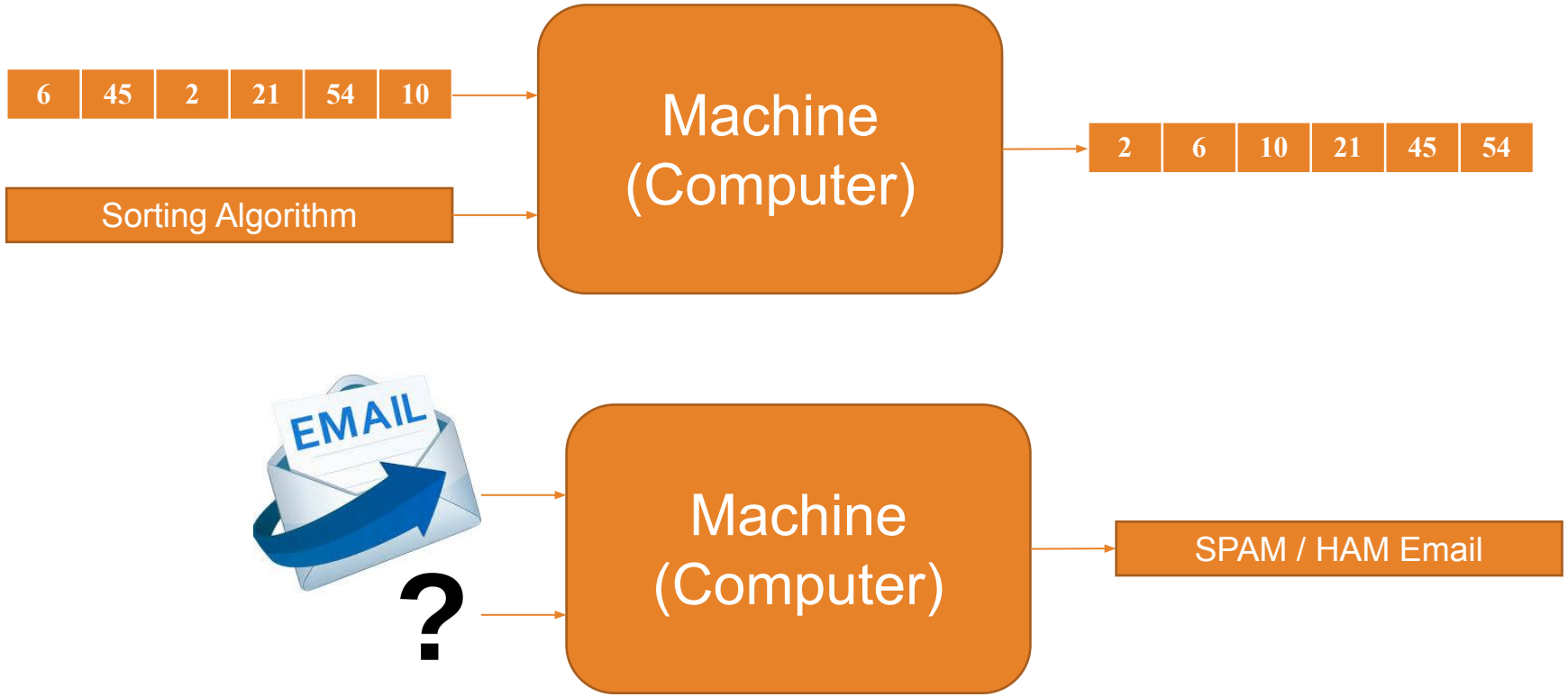
- Foundations of Machine Learning
- Classification Techniques (Basic to Advanced)
- Clustering Techniques (Basic to Advanced)
- Feature Engineering (Selection & Extraction)
- Handling Image / Video Data (Application Oriented Session)
- Handling Text Data (Application Oriented Session)
- Evolution of Artificial Neural Networks (Intuition to Deep Learning)

Foundations of Machine Learning

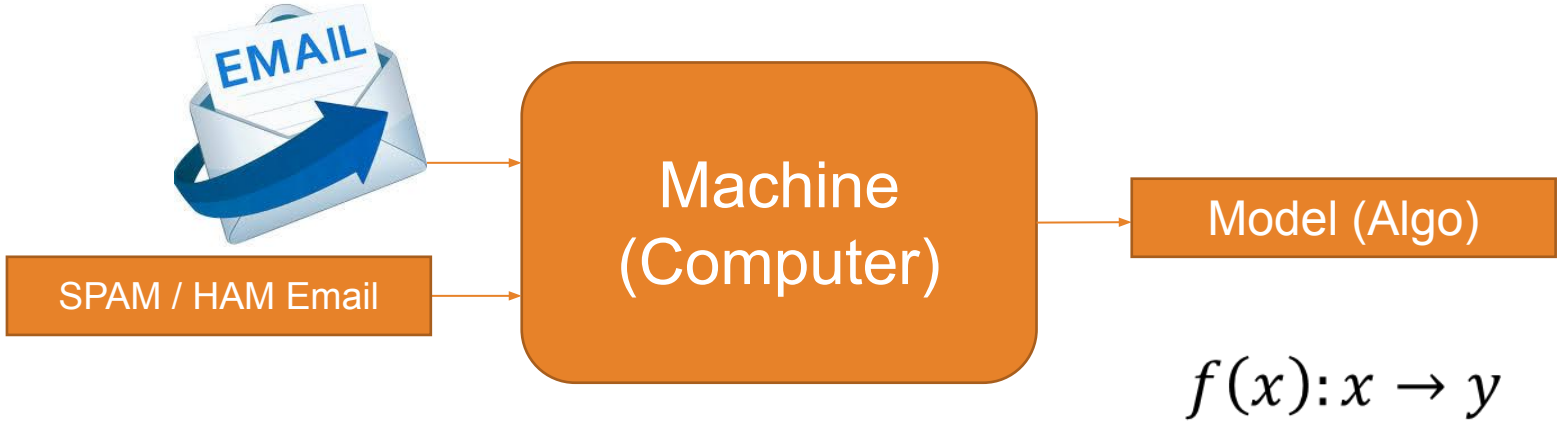
Agenda – ML Landscape

- ✔ What is Machine Learning..?
- ✔ Machine Learning Applications & Breakthroughs
- ✔ Types of Machine Learning
- ✔ Tools for Machine Learning
- ✔ Regression Analysis
  - ✔ Uni-Variate & Multivariate Linear Regression
  - ✔ Polynomial Regression
  - ✔ Case study

Programming Machines

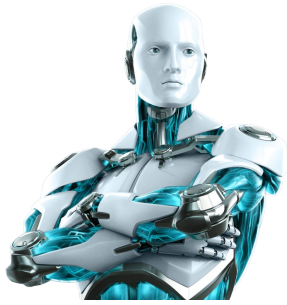


SPAM Filter



- First ML application that really became mainstream in the 1990s: *spam filter*.
- Spam filter is a Machine Learning program that can learn to flag spam given examples of spam emails (e.g., flagged by users) and examples of regular (non-spam, also called “ham”) emails.

Machine Learning



Machine Learning is the science (and art) of programming computers so they can *learn from data*.

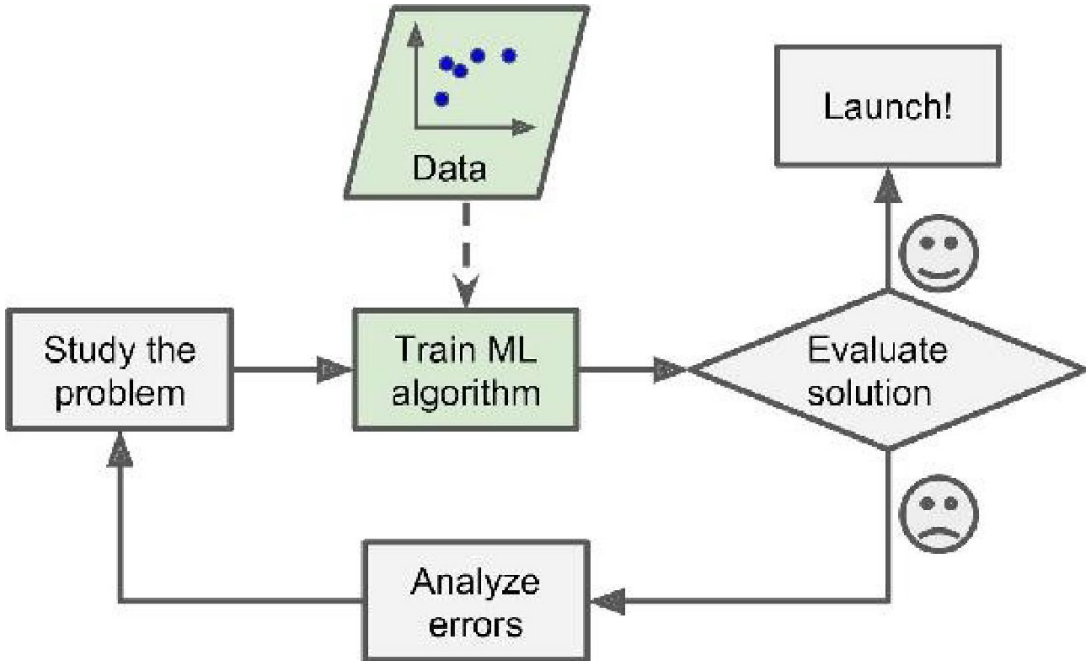
Here is a slightly more general definition:

- ✔ Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed. Arthur Samuel, *1959*

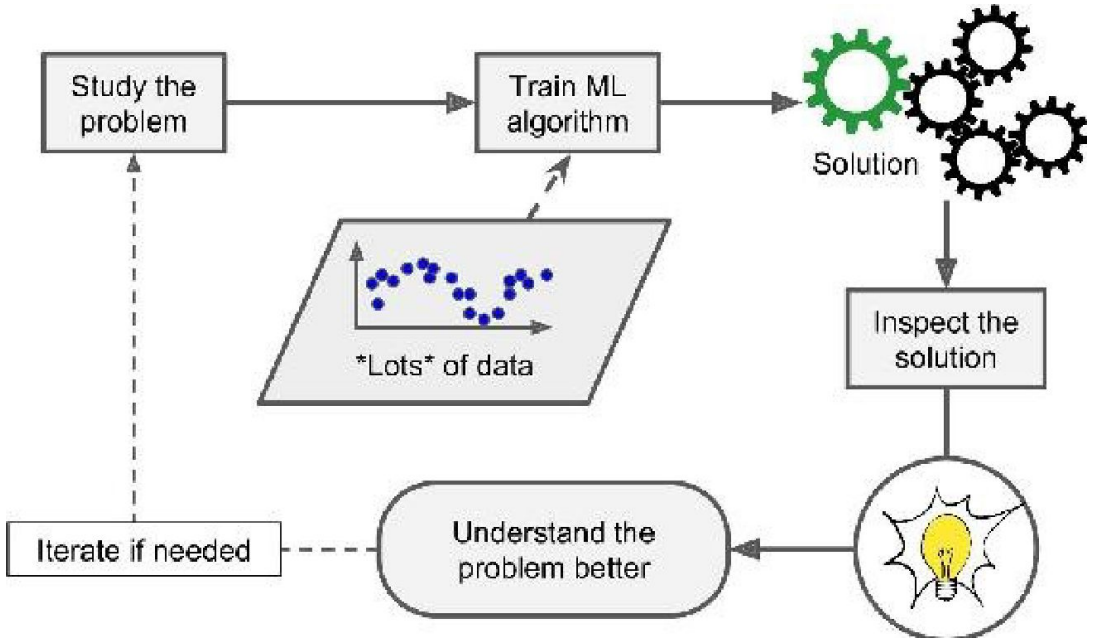
And a more engineering-oriented one:

- ✔ A computer program is said to learn from **experience E** with respect to some **task T** and some **performance measure P**, if its performance on T, as measured by P, improves with experience E. Tom Mitchell, *1997*

Machine Learning Approach

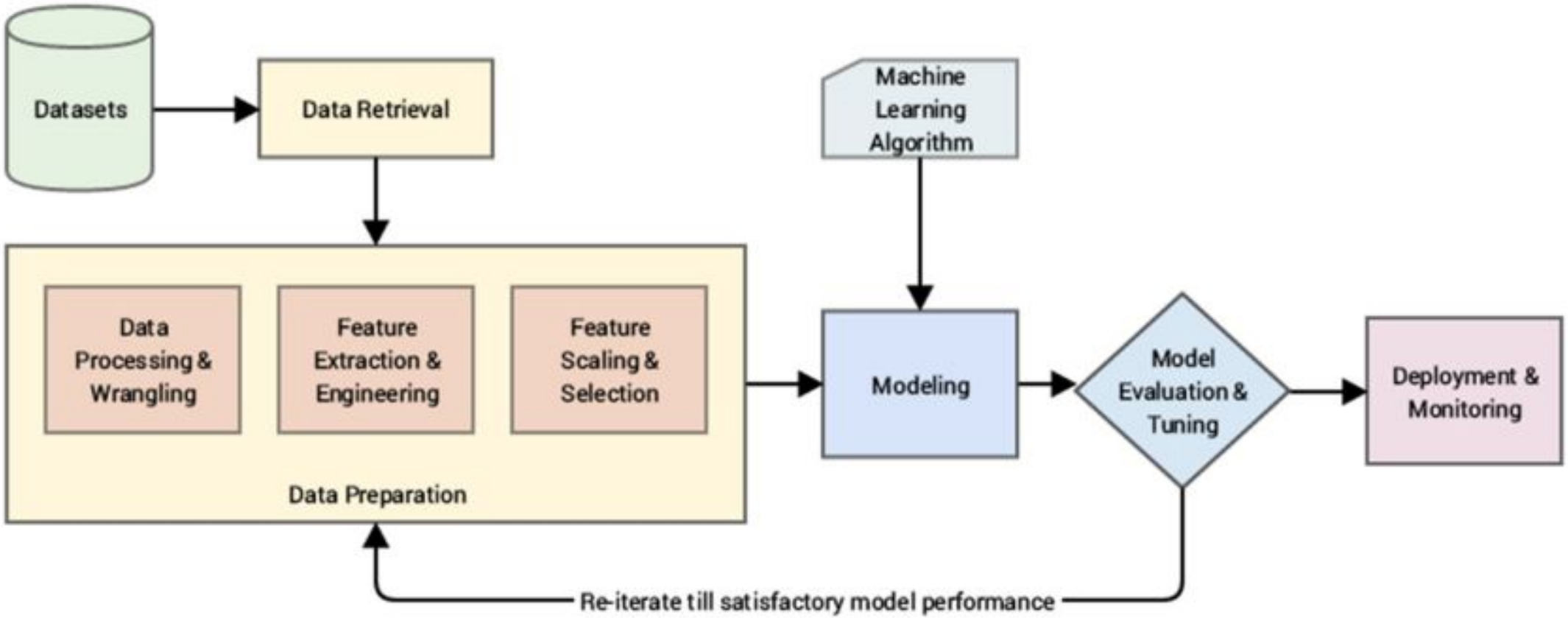


Case 1: Problem statement is clear

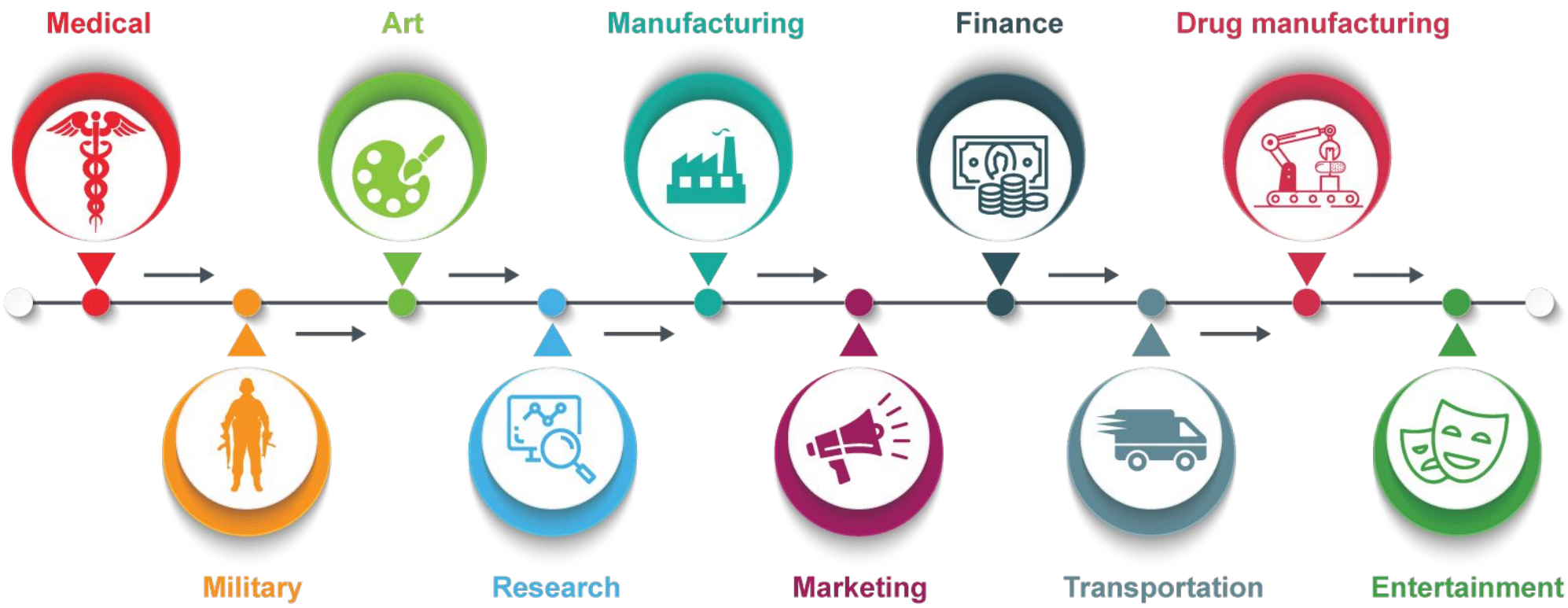


Case 2: Problem statement is not clear

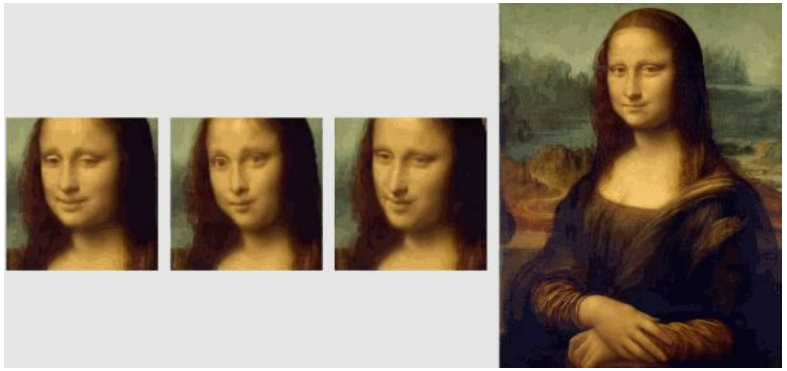
ML Pipeline Stages

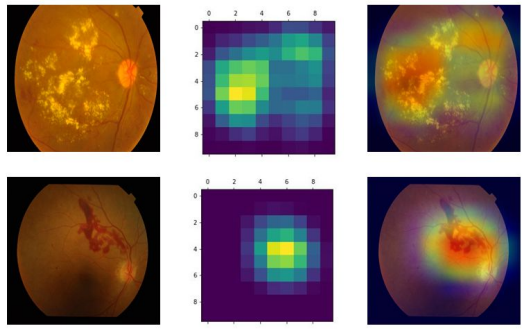


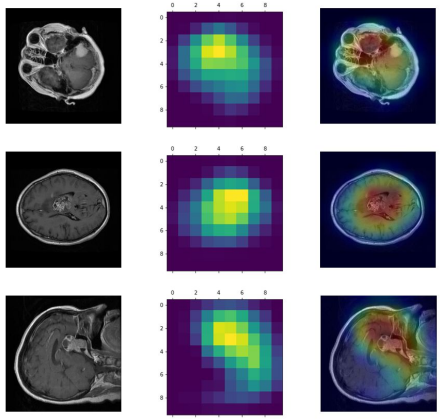
Importance of ML in various Domains…

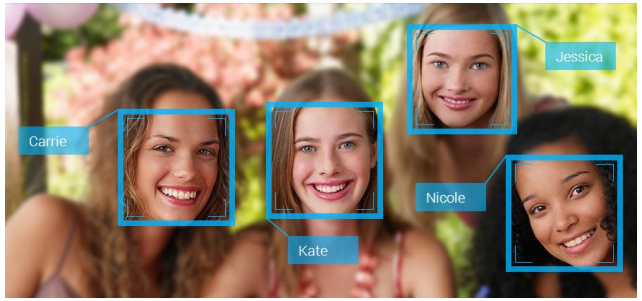


Breakthroughs of AI & ML (few..)

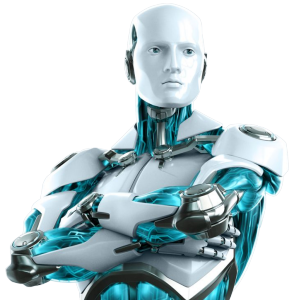






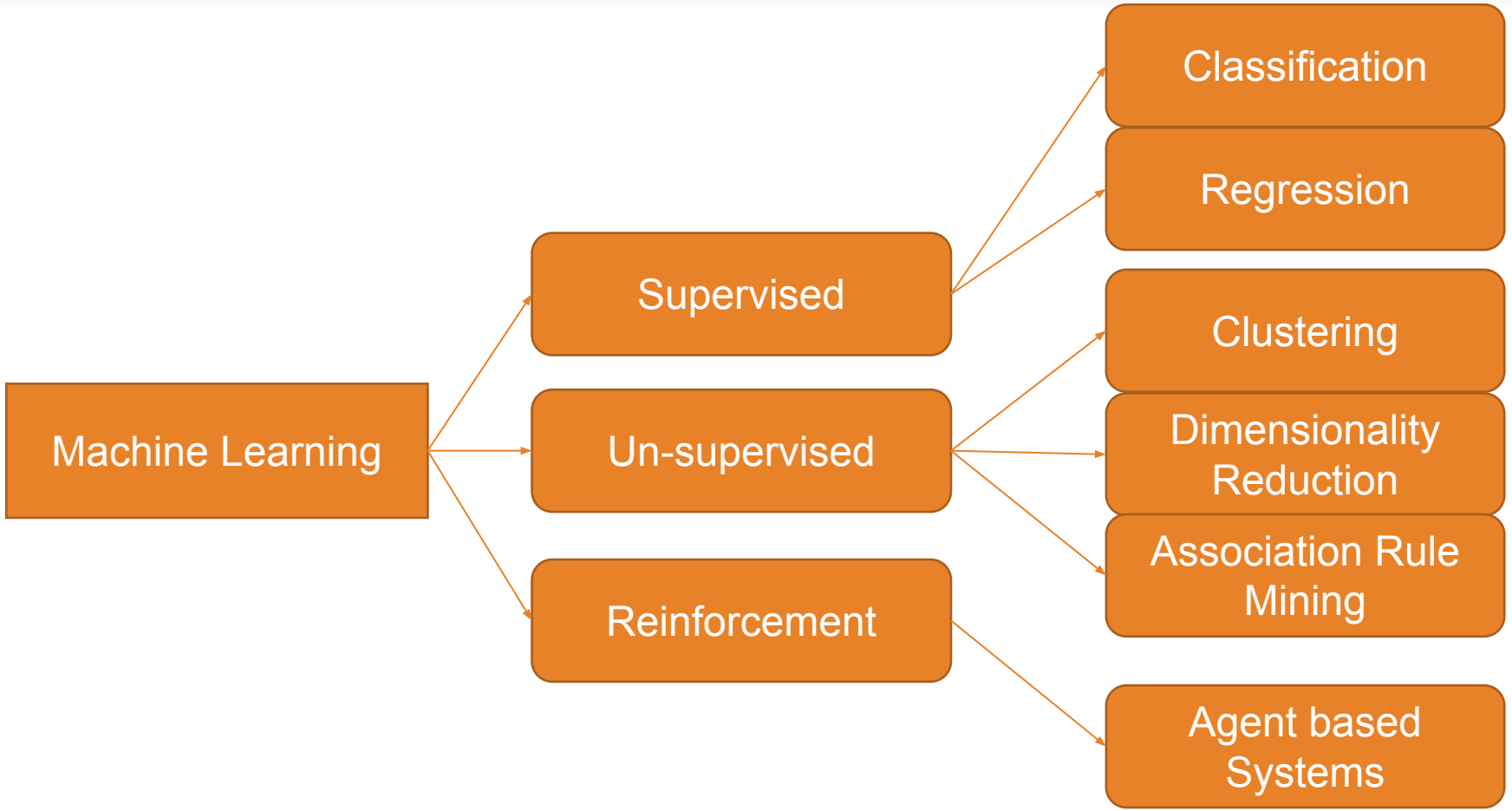


Types of Machine Learning

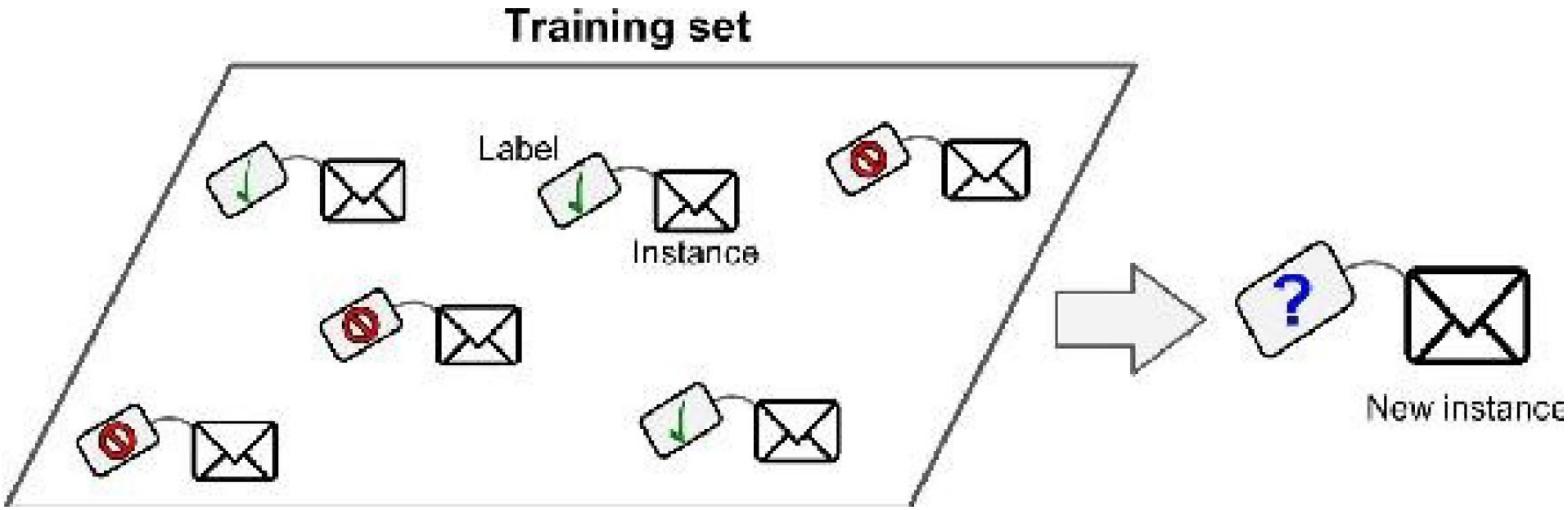


- Whether or not they are trained with human supervision:
  - Supervised, Unsupervised, and Reinforcement Learning
- Whether or not they can learn incrementally on the fly:
  - Online, Batch Learning
- Whether they work by simply comparing new data points to known data points, or instead detect patterns in the training data and build a predictive model, much like scientists do:
  - Instance-based versus Model-based learning

Based on Human Supervision

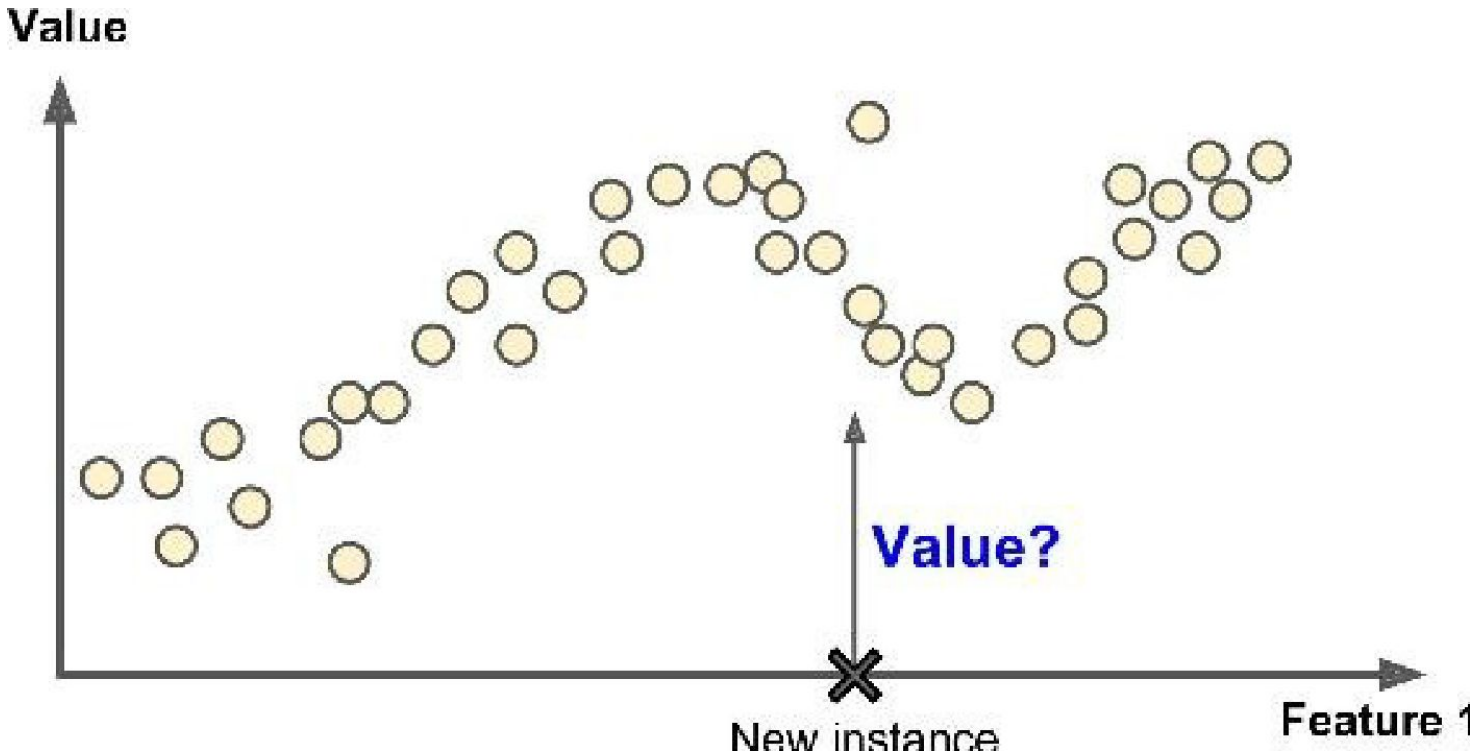


Supervised Learning



Labelled training set for supervised learning (e.g., spam classification)

Supervised Learning



Labelled training set for supervised learning (e.g., Regression)

Supervised Algorithms

- Linear Regression
- Logistic Regression
- K-Nearest Neighbours
- Naïve Bayes
- Decision Tree
- Support Vector Machines
- Ensemble Models
- Neural Networks

Unsupervised Algorithms

- K-Means
- K-Medoids
- Agglomerative
- Divisive
- DBSCAN
- BIRCH
- Gaussian Mixture Models

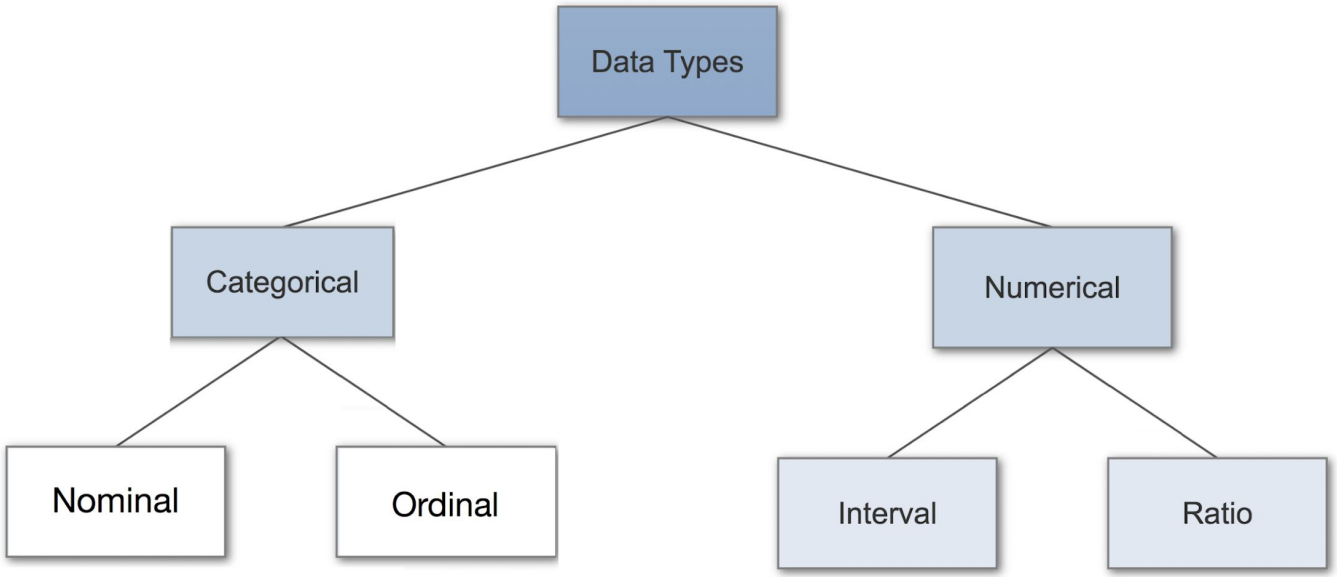
ML: Modules & Cloud Tools

- Scikit-learn, TensorFlow, Keras: Python Modules

- Azure Machine Learning Studio: Cloud based Platform

Types of Data

- Different Data Types : do proper exploratory data analysis (EDA)



Types of Data

- **Categorical** data represents characteristics
  - represent things like a person's gender, language etc.
  - can also take on numerical values (Example: 1 for female and 0 for male).

1. **Nominal** values represent discrete units and are used to label variables, that have no quantitative value.
2. **Ordinal** values represent discrete and ordered units.

- **Numerical** data contains both discrete and continuous data
  - Continuous Data represents measurements

1. **Interval** values represent *ordered units that have the same difference.*
2. **Ratio** values are *the same as interval values, with the difference that they do have an absolute zero*

Regression

Regression Analysis

- Regression analysis is a form of predictive modelling technique which determines the strength of the relationship between one dependent variable (usually denoted by Y) and a series of other changing variables (known as independent variables).
- Using the relationship between variables to find the best fit line or the regression equation that can be used to make predictions.
- Output is continuous.
- **Applications:**
  - Predicting the weather using the weather conditions in the past.
  - Predict sales for a company based on previous sales.
  - Predicting House Price based on some criteria viz. size, #rooms, location

Regression Analysis

- There are two types of Regression Analysis:
  - Linear Regression
  - Polynomial Regression

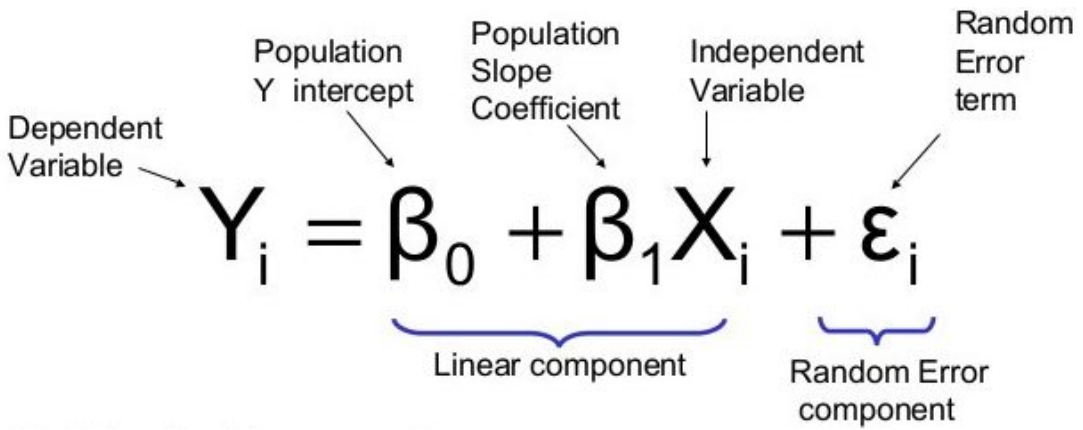
Linear Regression

- **A linear regression** refers to a regression model establishes a relationship between **dependent variable (Y)** and one or more **independent variables (X)** using a **best fit straight line**.
- The dependent variable is *continuous*, independent variable(s) can be *continuous or discrete*, and nature of regression line is *linear*.

- Types of Regression:
  1. Single Variable Linear Regression
  2. Multiple Linear Regression

Single Variable Linear Regression (SLR)

- It is a technique used to model the relationship between a single input independent variable (feature variable) and an output dependent variable using a linear model i.e. a line.

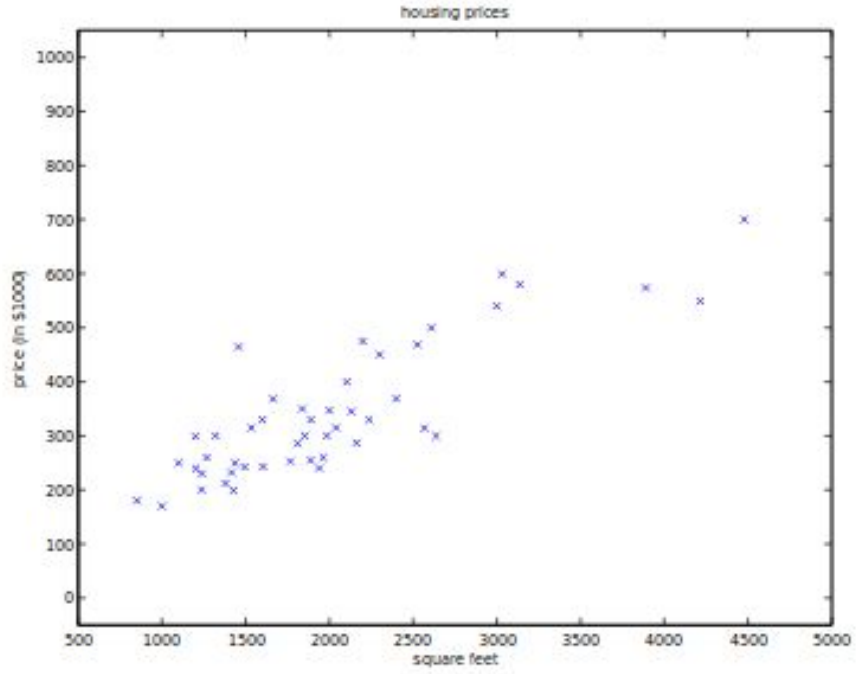


- *Y(predicted) = (β1*x + βo) + Error value*

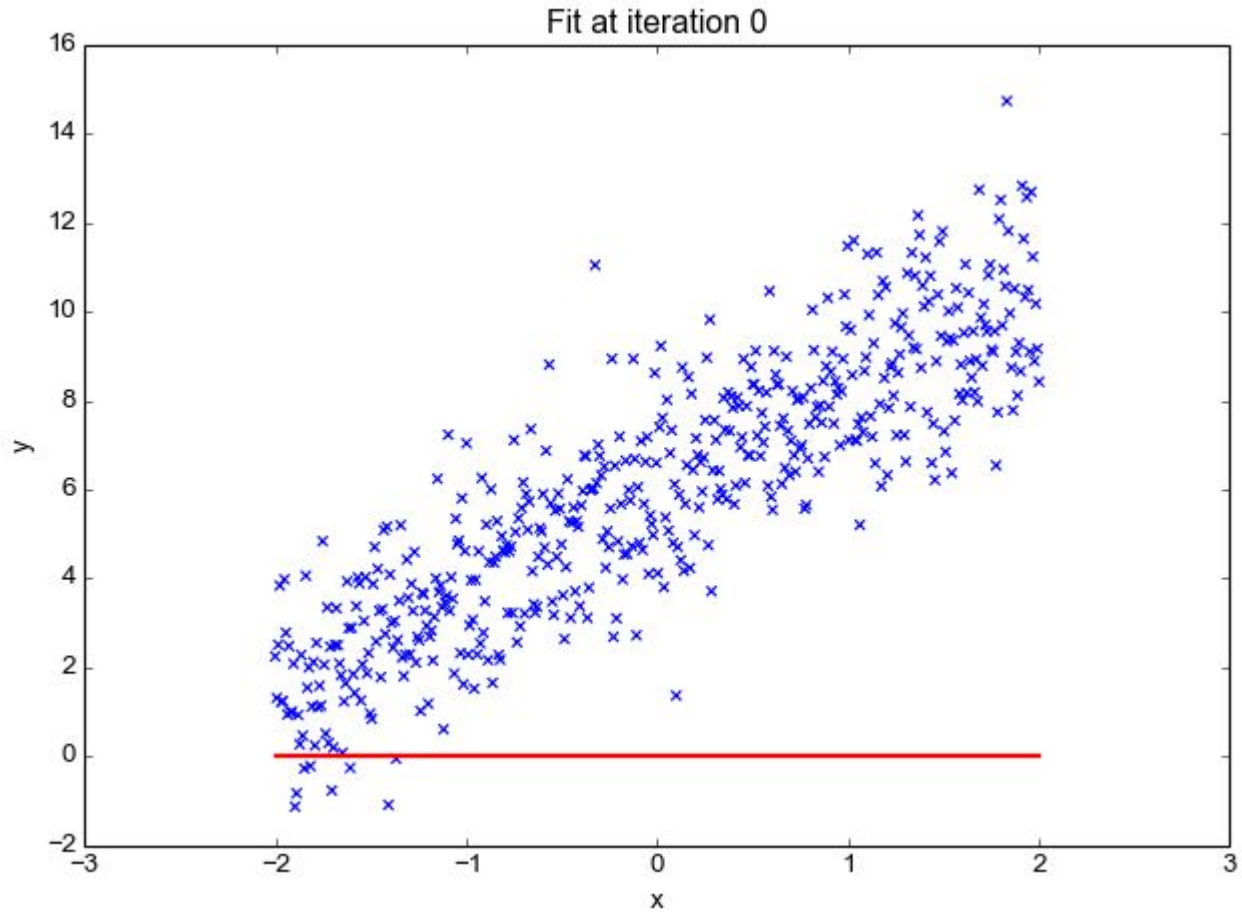
Single Variable Linear Regression (SLR)

- Estimate the “cost” of the house according to the “Living area” of the house in feet.

Living area (feet^2)	Price (1000$s)
2104	400
1600	330
2400	369
1416	232
3000	540
⋮	⋮



Single Variable Linear Regression (SLR)



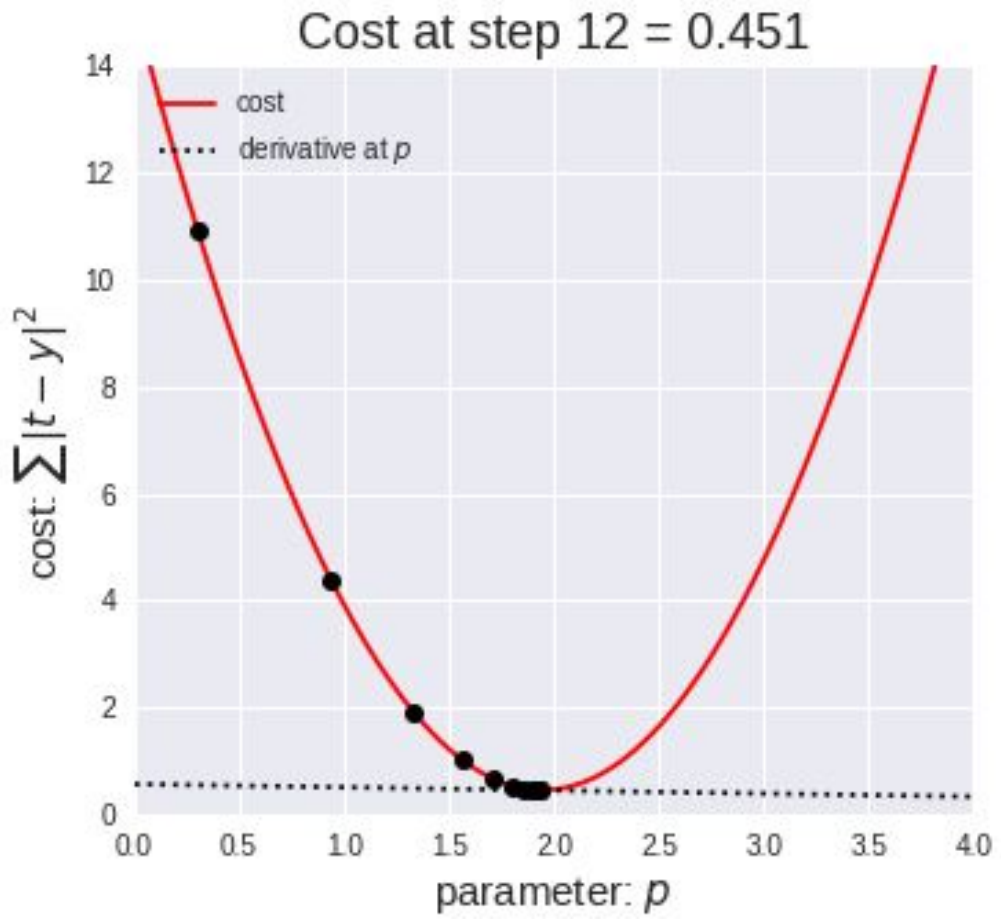
Single Variable Linear Regression (SLR)

- Calculate Error : Σ (actual output — predicted output)²

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2.$$

- Goal : **Find the values of *βo and β1 such that J(.) is minimum***
- *Use Gradient Descent Algorithm.*
- The main goal of Gradient descent is to **minimize the cost value.** i.e. min J(.)

Gradient Descent for Linear Regression



Multiple Linear Regression (MLR)

- Multiple linear regression has (>1) independent variables, whereas simple linear regression has only 1 independent variable.
- Output is a linear combination of the input variables.
- We can model a multi-variable linear regression as the following:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \ldots + \beta_3 X_3 + E$$

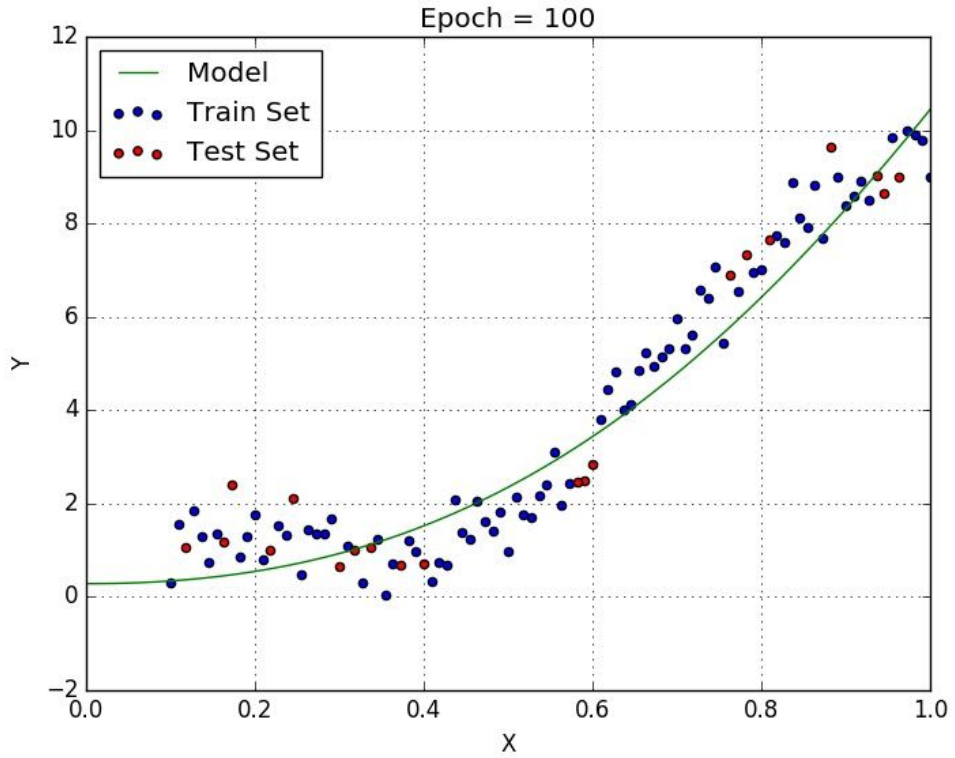
- The above function does not include any non-linearities and so is only suited for modelling linearly separable data.

Polynomial Regression

- Relationship doesn't look linear in many scenarios..
- Can do a Polynomial Regression on the data to fit a Polynomial Equation to it.
- We can model Polynomial Regression as the following:

$$Y = \beta_0 + \beta_1 X_1 + \beta_1 X_1^2 + \beta_2 X_2^3 + \beta_2 X_2^3 + \cdots + \beta_n X_n^{n-1}$$

Polynomial Regression

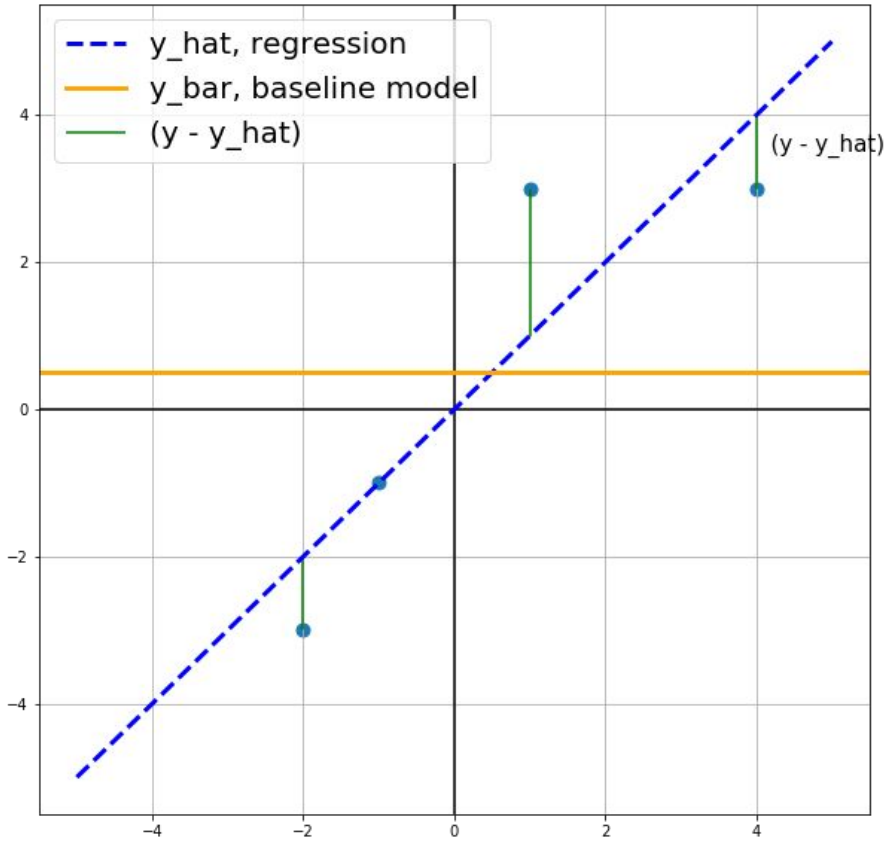


Evaluation Measures for Regression

- Evaluate the model performance using the metric **R-square**.

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y}_i)^2}$$

- Key points about Linear Regression:
  - Fast and easy to model.
  - Useful when the relationship to be modelled is not extremely complex
  - Very intuitive to understand and interpret.
  - Linear Regression is very sensitive to outliers.



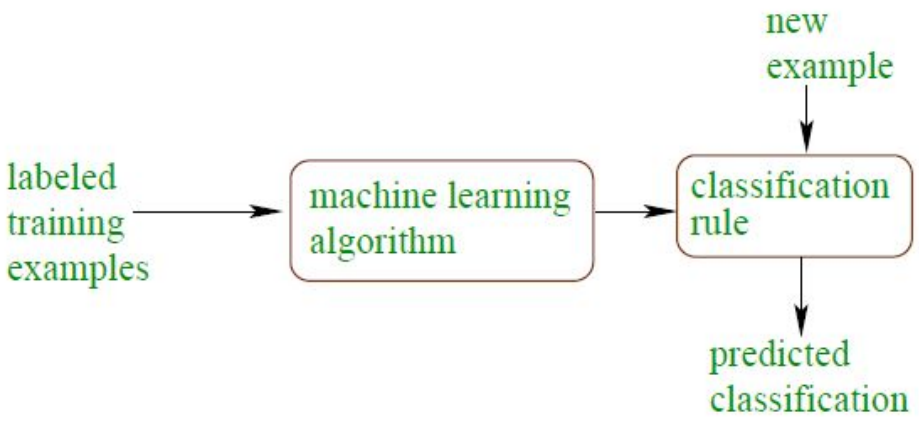
Regression Analysis using *scikit-learn*

- Simple and efficient tools for predictive data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license
- https://scikit-learn.org/

Classification

Classification…

- A process of categorizing a given set of data into classes.
- It specifies the class to which data elements belong to and is best used when the output has finite and discrete values.
- Performed on both structured or unstructured data.

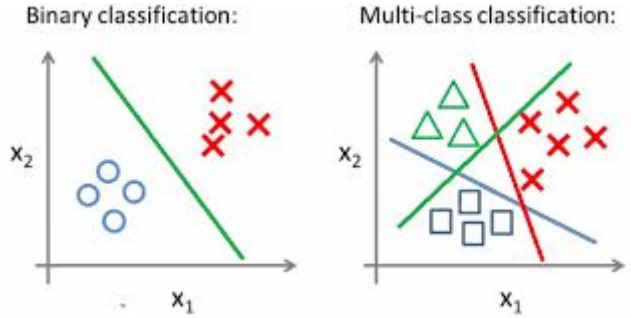


Applications

- Spam Mail Classification
- Heart Disease Prediction
- Brain Tumour Detection
- Bank customers Loan Prediction
- Sentiment Analysis
- Facial expression Recognition
- Plant Leaf Disease Prediction
- Vehicle Classification
- Face Recognition

Types of Classification

- Based on number of target variables available, Classification can be divided into two types:
  - Binary (or) Two Class Classification.
  - Multi – Class Classification.



- By analysing the health condition prediction - the person will get heart stroke or not **(Target : Yes or No)**
- Gender classification from hair length **(Target: Male or Female)**
- Classifying fruits from features like color, taste, size, weight **(Target: Apple, Orange, Cherry, Banana)**

Classification Terminologies In Machine Learning

- **Classifier** – It is an algorithm that is used to map the input data to a specific category.
- **Classification Model** – The model predicts or draws a conclusion to the input data given for training, it will predict the class or category for the data.
- **Feature** – A feature is an individual measurable property of the phenomenon being observed.
- **Binary Classification** – It is a type of classification with two outcomes, for eg – either true or false.
- **Multi-Class Classification** – The classification with more than two classes, in multi-class classification each sample is assigned to one and only one label or target.

Classification Terminologies In Machine Learning

- **Multi-label Classification** – This is a type of classification where each sample is assigned to a set of labels or targets.
- **Initialize** – It is to assign the classifier to be used for the
- **Train the Classifier** – Each classifier in sci-kit learn uses the fit(X, y) method to fit the model for training the train X and train label y.
- **Predict the Target** – For an unlabelled observation X, the predict(X) method returns predicted label y.
- **Evaluate** – This basically means the evaluation of the model i.e. classification report, accuracy score, etc.

Data Pre-processing

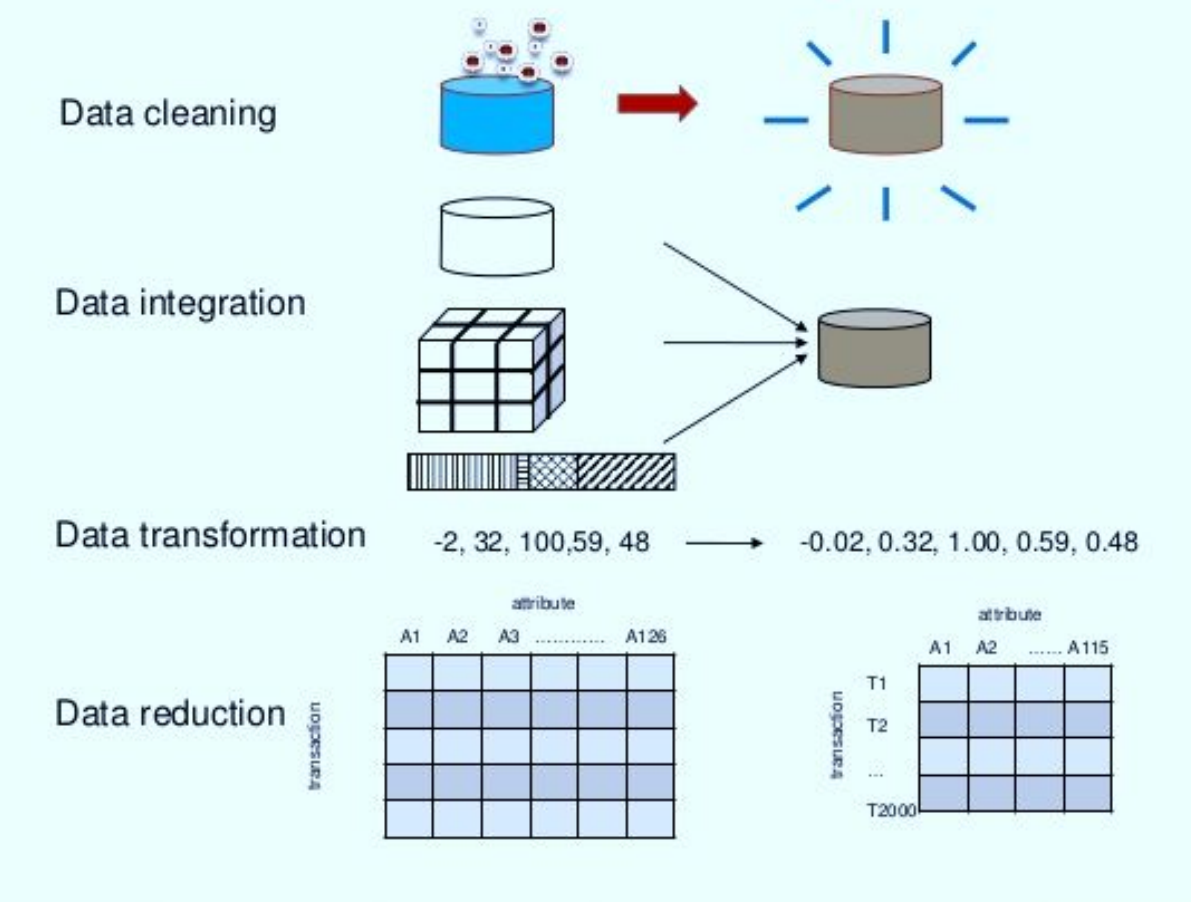
What is Data Preprocessing…?

Why Preprocess the Data…?

- Measures for data quality: A multidimensional view
  - Accuracy: correct or wrong, accurate or not
  - Completeness: not recorded, unavailable, …
  - Consistency: some modified but some not, dangling, …
  - Timeliness: timely update?
  - Believability: how trustable the data are correct?
  - Interpretability: how easily the data can be understood?

Major Tasks in Data Preprocessing

- **Data cleaning**
  - Fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies
- **Data integration**
  - Integration of multiple databases, data cubes, or files
- **Data reduction**
  - Dimensionality reduction
  - Data compression
- **Data Transformation**
  - Normalization



Types of Classification Algorithms

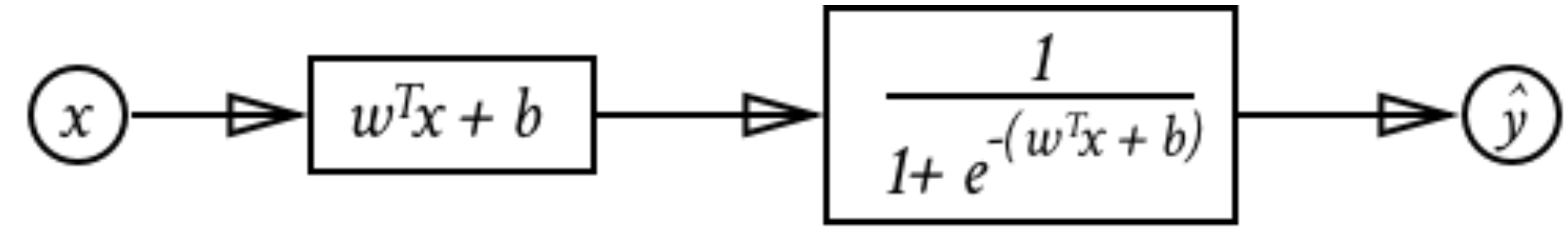
- Linear Models
  - Logistic Regression
  - Support Vector Machines

- Nonlinear models
  - K-Nearest Neighbours (KNN)
  - Kernel Support Vector Machines (SVM)
  - Decision Tree Classification

Logistic Regression

- This refers to a Regression model that is used for classification.
- The goal of Logistic Regression is to find a best-fitting relationship between the dependent variable and a set of independent variables.
- This method is widely used for binary classification problems.
- It can also be extended to multi-class classification problems.
- Here, the dependent variable is categorical: $y \epsilon \{0, 1\}$

Logistic Regression



input

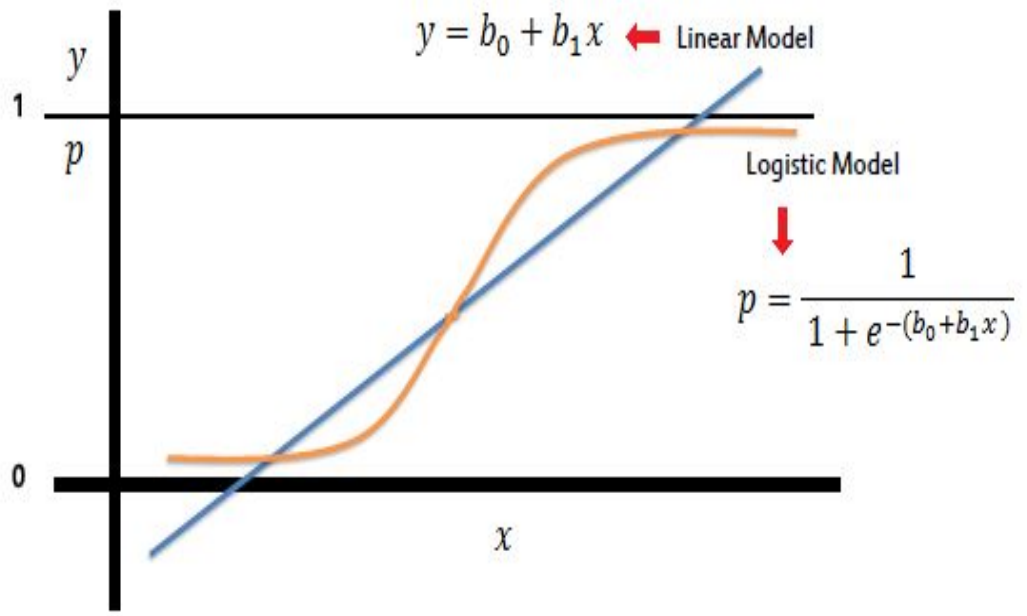
linear function,
w = weight
b = bias

sigmoid function ***a***
$a \in (0,1)$

output based on ***a****,*
1 if a >= 0.5
0 if a < 0.5

Linear Vs Logistic Regression

- A linear regression will predict values outside the acceptable range (e.g. predicting probabilities outside the range 0 to 1)
- A logistic regression will predict values in the acceptable range (e.g. predicting probabilities in the range 0 to 1)



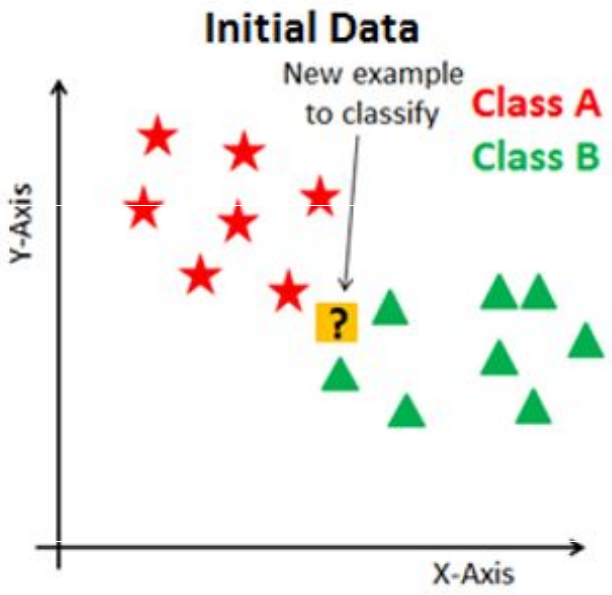
K-Nearest Neighbours (KNN)

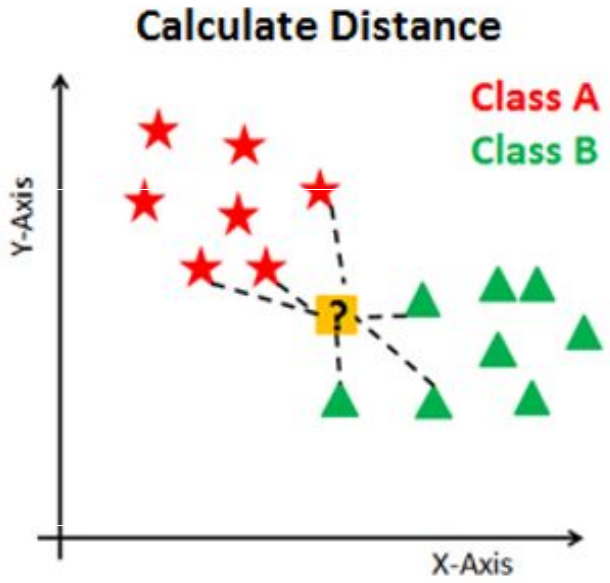
- KNN can be used for both classification and regression predictive problems.
- It is more widely used in classification problems in the industry.
- KNN algorithm fairs across all parameters of considerations.
- It is commonly used for its easy of interpretation and low calculation time.
- It is considered as lazy learner.

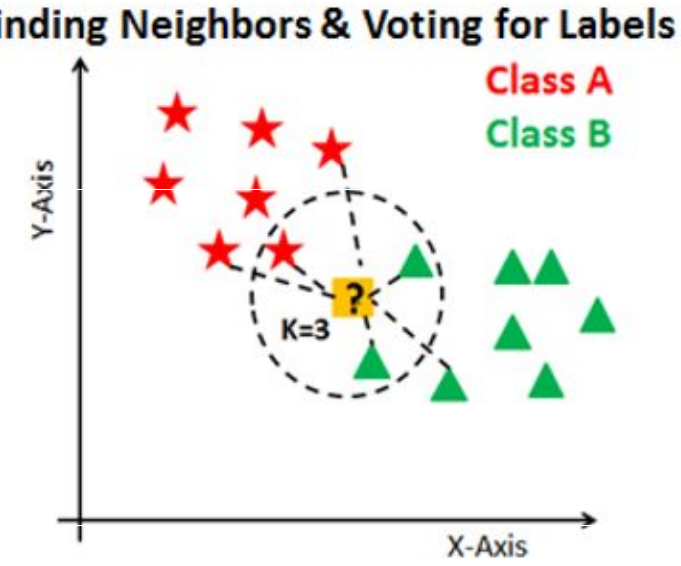
Algorithm

1. Load the data
2. Initialise the value of k
3. For getting the predicted class, iterate from 1 to total number of training data points
    a. Calculate the distance between test data and each row of training data. Here we will use Euclidean distance as our distance metric since it's the most popular method. The other metrics that can be used are Chebyshev, cosine, etc.
    b. Sort the calculated distances in ascending order based on distance values
    c. Get top k rows from the sorted array
    d. Get the most frequent class of these rows
    e. Return the predicted class

Algorithm - Example





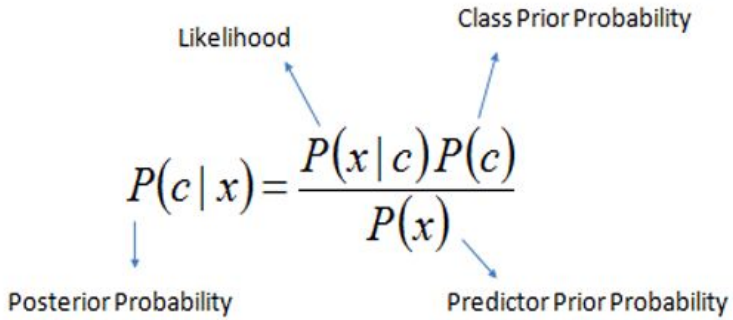


Algorithm - Example

Naïve Bayes

- Naïve Bayes algorithm is a supervised learning algorithm, which is based on **Bayes theorem** and used for solving classification problems.
- It is mainly used in *text classification* that includes a high-dimensional training dataset.
- Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
- **It is a probabilistic classifier, which means it predicts on the basis of the probability of an object**.
- Some popular examples of Naïve Bayes Algorithm are **spam filtration, Sentimental analysis, and classifying articles**.

Naïve Bayes – Rule & Steps



$$P(c \mid \mathrm{X}) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

1. Convert the given dataset into frequency tables.
2. Generate Likelihood table by finding the probabilities of given features.
3. Now, use Bayes theorem to calculate the posterior probability.

Naïve Bayes - Example

Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

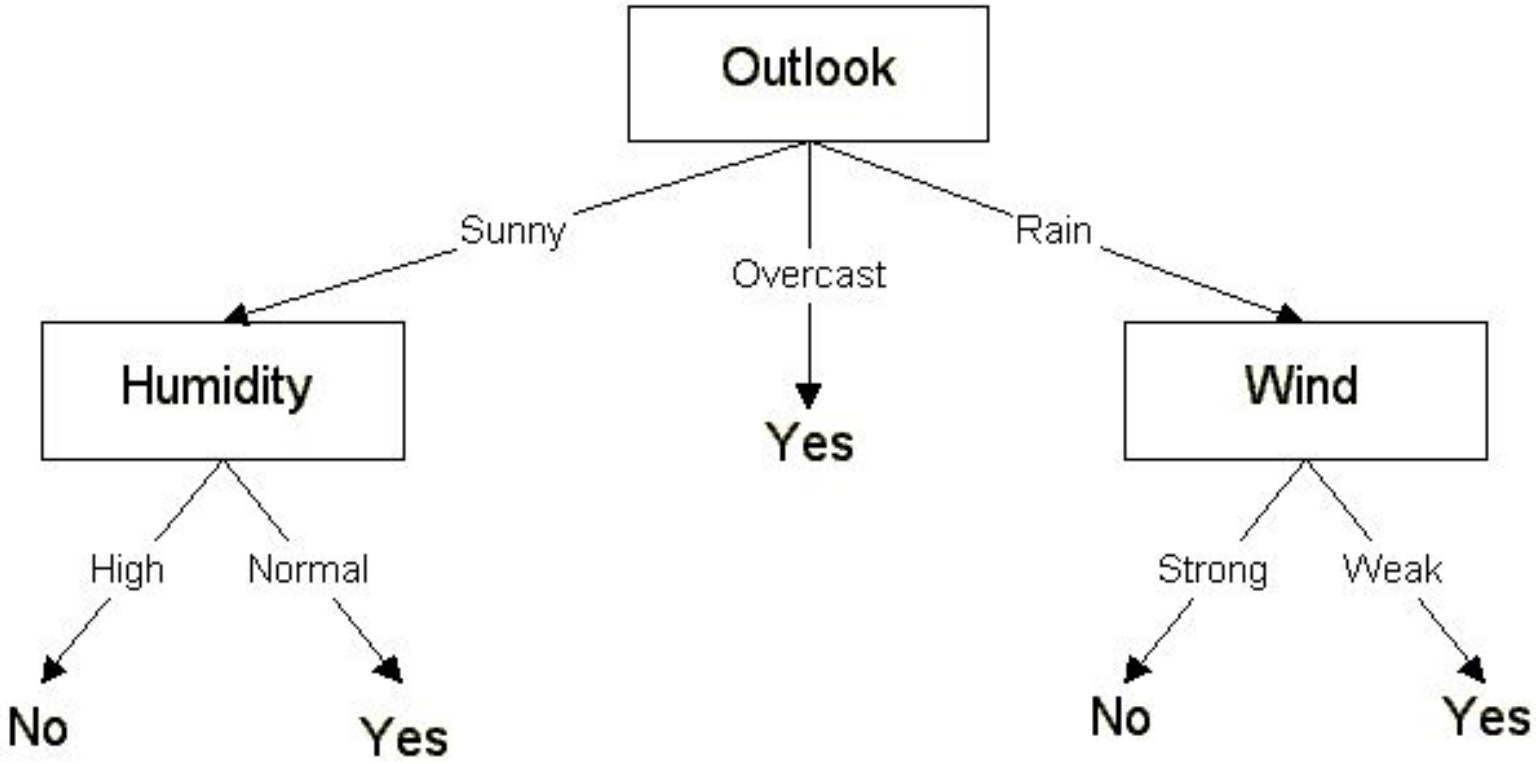
Frequency Table		
Weather	No	Yes
Overcast		4
Rainy	3	2
Sunny	2	3
Grand Total	5	9

Likelihood table				
Weather	No	Yes		
Overcast		4	=4/14	0.29
Rainy	3	2	=5/14	0.36
Sunny	2	3	=5/14	0.36
All	5	9		
	=5/14	=9/14		
	0.36	0.64		

Decision Trees

- Decision Trees (DT) can be used both for classification and regression.
- The advantage of decision trees is that they require very little data preparation.
- They do not require feature scaling at all.
- They are also the fundamental components of Random Forests, one of the most powerful ML algorithms.
- In the case of classification, the data is segregated based on a series of questions.
- Any new data point is assigned to the selected leaf node.

Simple Example

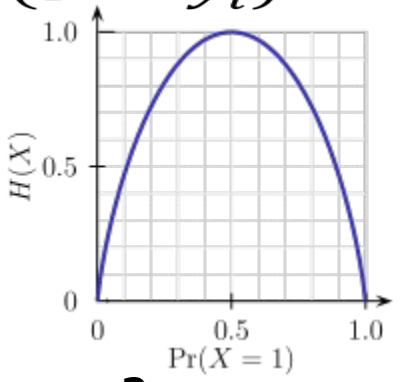


Algorithm for Decision Tree Induction

- Basic algorithm (a greedy algorithm)
  - Tree is constructed in a top-down recursive divide-and-conquer manner
  - At start, all the training examples are at the root
  - Attributes are categorical (if continuous-valued, they are discretized in advance)
  - Examples are partitioned recursively based on selected attributes
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain)
- Conditions for stopping partitioning
  - All samples for a given node belong to the same class
  - There are no remaining attributes for further partitioning – majority voting is employed for classifying the leaf
  - There are no samples left

Brief Review of Entropy

- Entropy (Information Theory)
  - A measure of uncertainty associated with a random variable
  - Calculation: For a discrete random variable Y taking m distinct values $\{y_1, \ldots, y_m\}$,
    - $H(Y) = -\sum_{i=1}^{m} p_i \log(p_i)$, where $p_i = P(Y = y_i)$
  - Interpretation:
    - Higher entropy => higher uncertainty
    - Lower entropy => lower uncertainty
- Conditional Entropy
  - $H(Y|X) = \sum_x p(x) H(Y|X = x)$

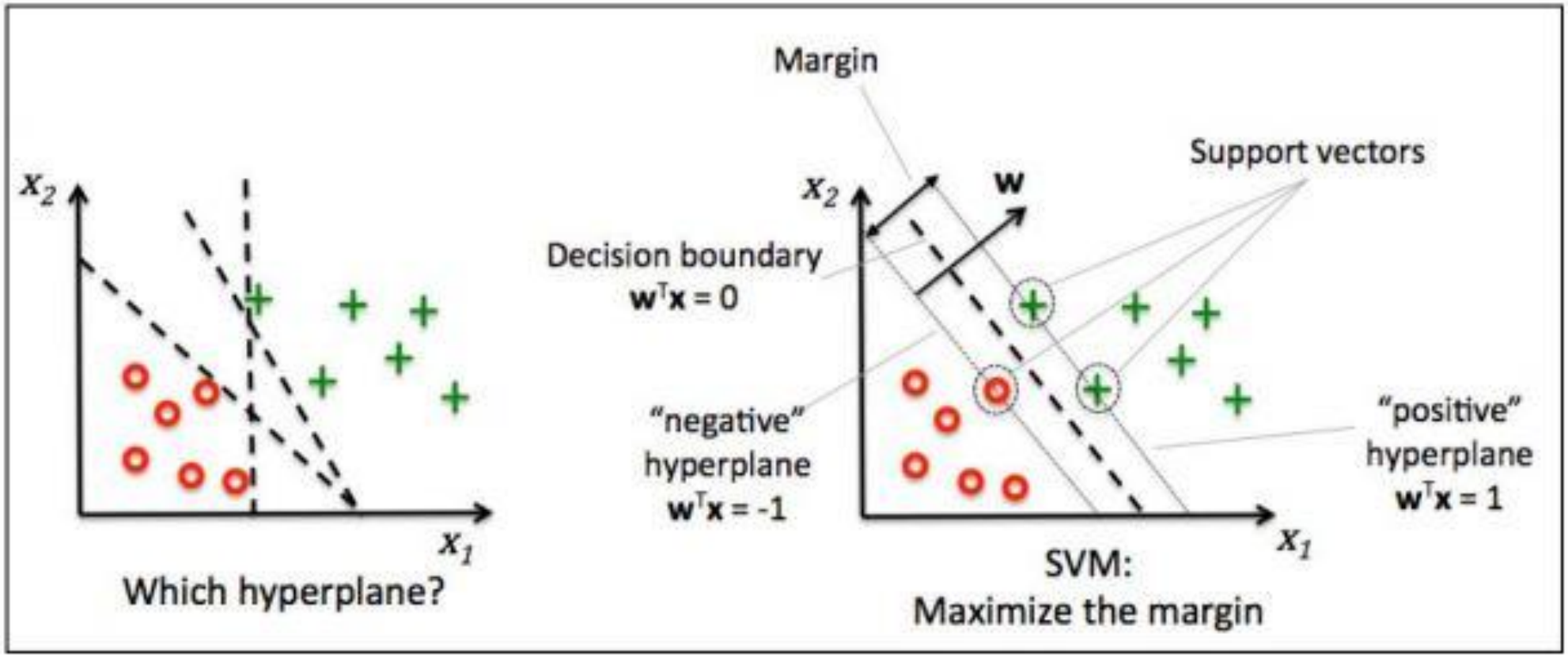


m = 2

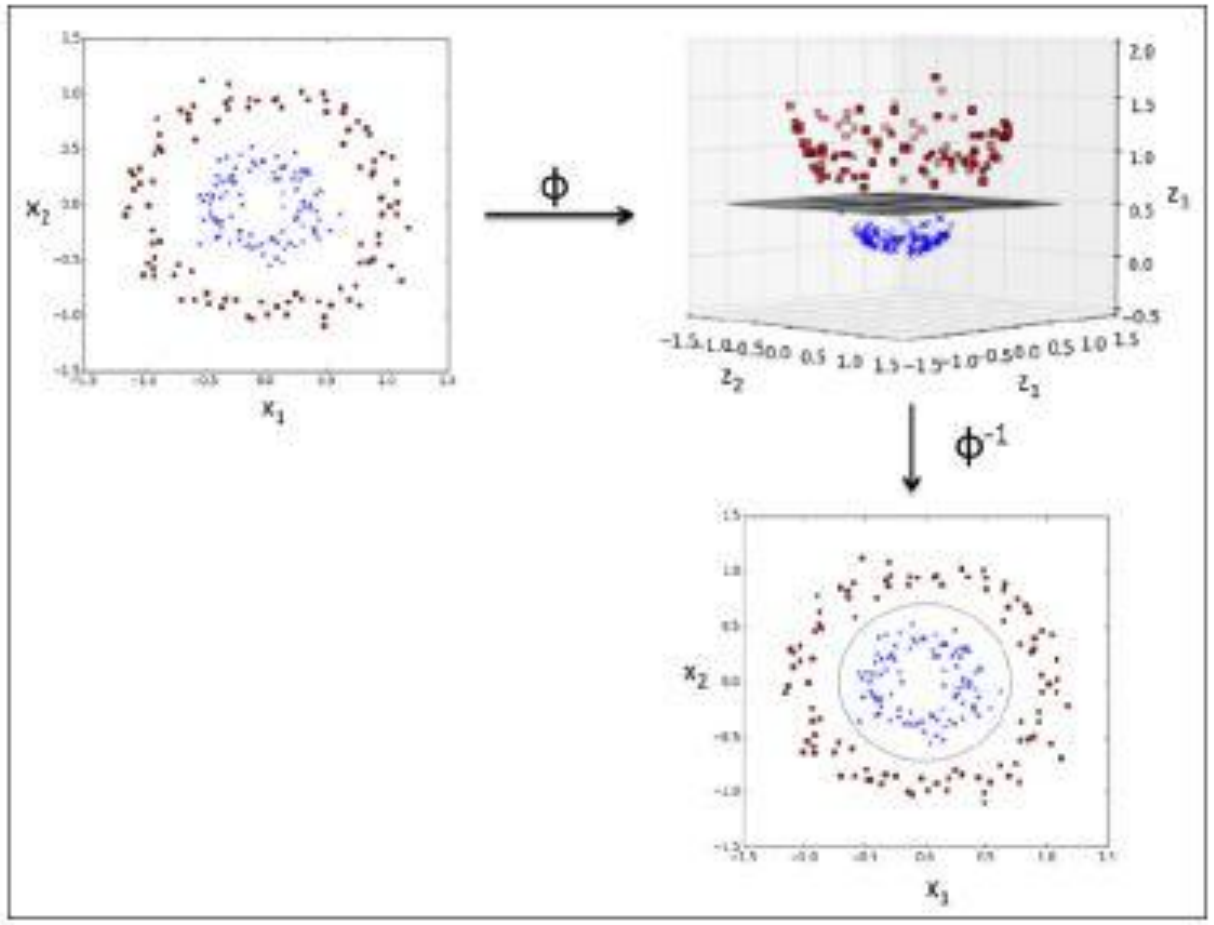
Support Vector Machines (SVM)

- SVMs are classification algorithms used to assign data to various classes.
- They involve detecting hyperplanes which segregate data into classes.
- SVMs are very versatile and are also capable of performing linear or nonlinear classification, regression, and outlier detection.
- Once ideal hyperplanes are discovered, new data points can be easily classified.
- The optimization objective is to find “maximum margin hyperplane” that is farthest from the closest points in the two classes (these points are called support vectors).

Support Vector Machines (SVM)



Kernel - Support Vector Machines (SVM)



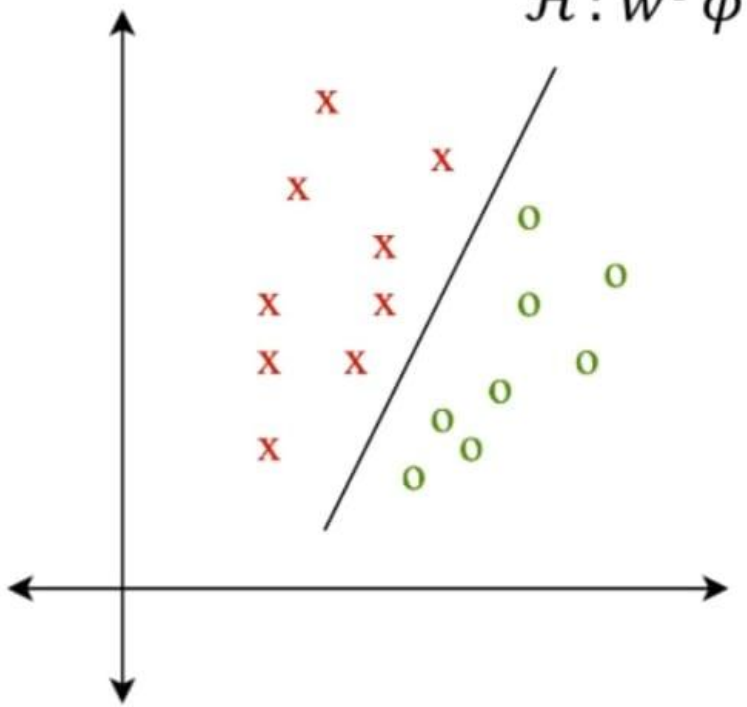
Support Vector Machines

1. What is a *Point*? $x \in \mathbb{R}^D$

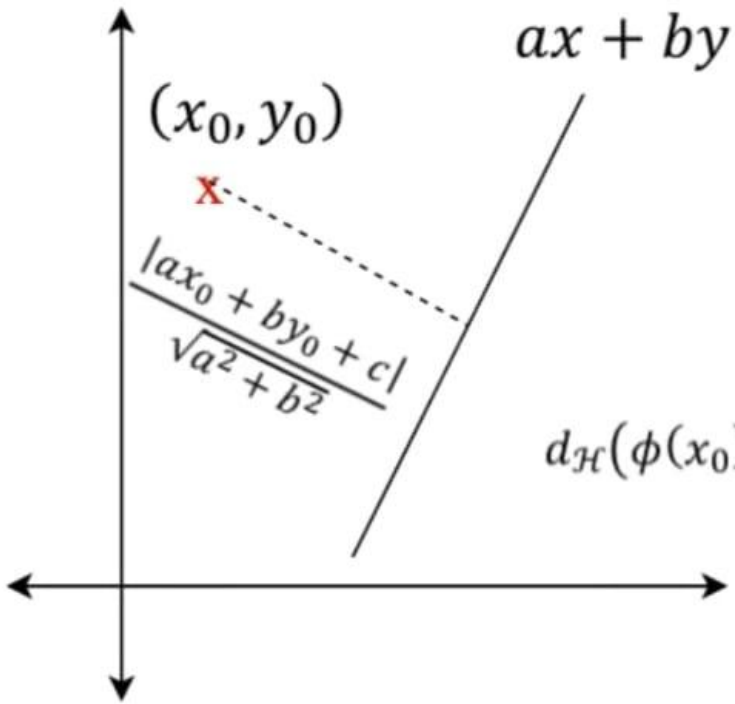
$\phi: \mathbb{R}^D \rightarrow \mathbb{R}^M \quad \phi(x) \in \mathbb{R}^M$

2. What is a *Decision Boundary?*

$\mathcal{H}: w^T\phi(x) + b = 0$



3. What is the *Distance Measure?*

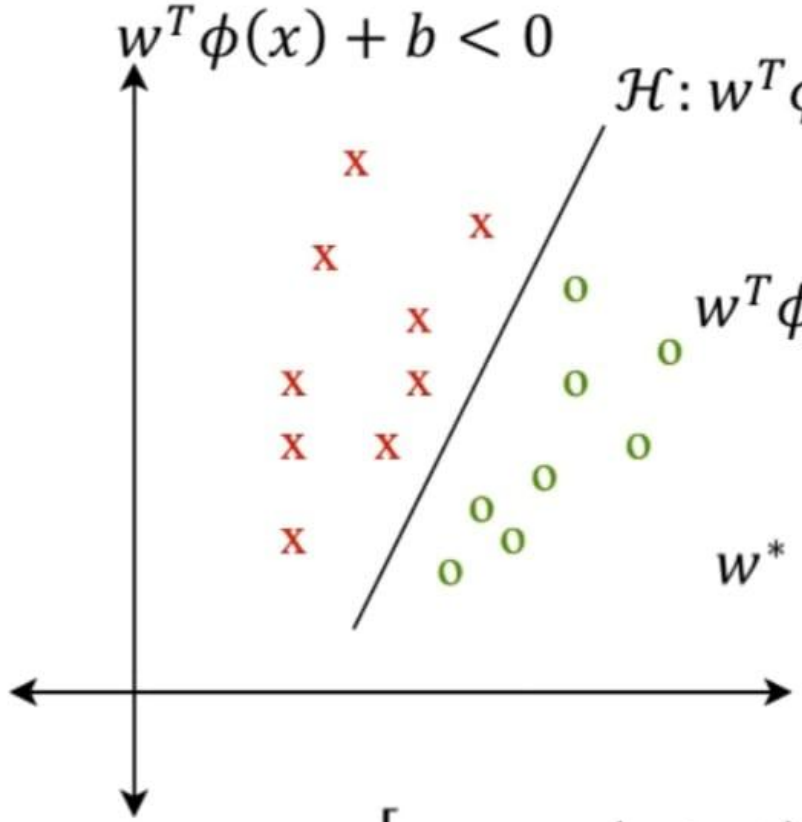


$$d_{\mathcal{H}}(\phi(x_0)) = \frac{|w^T\phi(x_0) + b|}{\|w\|_2}$$

Case 1: Perfect Separation

Support Vector Machines

4. What are we *optimizing*?



$$w^* = \arg\max_w \left[\min_n d_{\mathcal{H}}\left(\phi(x_n)\right)\right]$$

$$y_n[w^T\phi(x)+b] = \begin{cases} \geq 0, & correct \\ < 0, & incorrect \end{cases}$$

$$w^* = \arg\max_w \left[\min_n \frac{|w^T\phi(x_n)+b|}{\|w\|_2}\right]$$

$$w^* = \arg\max_w \left[\min_n \frac{y_n[w^T\phi(x_n)+b]}{\|w\|_2}\right] \quad \because perfect\ separation$$

$$w^* = \arg\max_w \frac{1}{\|w\|_2}\left[\min_n y_n[w^T\phi(x_n)+b]\right]$$

Distance of closest point to $\mathcal{H}$

Case 1: Perfect Separation

Support Vector Machines

4. What are we *optimizing*?

$$w^* = \arg\max_w \frac{1}{\|w\|_2}\left[\min_n y_n[w^T\phi(x_n) + b]\right]$$

Let $\min_n y_n[w^T\phi(x_n) + b] = 1$

$$(cw)^T\phi(x_n) + (cb) = c(w^T\phi(x_n) + b) = 0 \qquad \begin{matrix} w \leftarrow cw \\ b \leftarrow cb \end{matrix}$$

Primal Form of SVM

$$w^* = \arg\max_w \frac{1}{\|w\|_2}$$

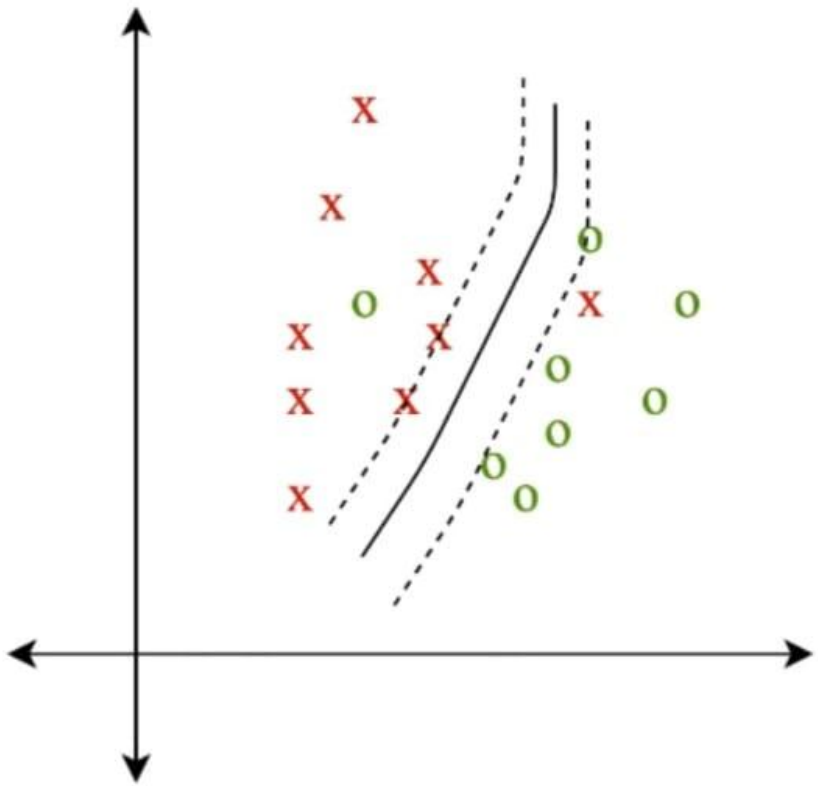
$$s.t \ \min_n y_n[w^T\phi(x_n) + b] = 1$$

$$\min_w \frac{1}{2}||w||_2$$

$$s.t \ y_n[w^T\phi(x_n) + b] \geq 1 \ \forall n$$

Case 2: Non-Perfect Separation

Support Vector Machines



$$\cancel{y_n[w^T\phi(x_n)+b] > 0 \ \forall n}$$

$$y_n[w^T\phi(x_n)+b] \leq 0 \ \exists n$$

New Primal Form of SVM

$$\min_{w,b,\{\xi_n\}} \frac{1}{2}||w||_2 + C\sum_n \xi_n$$

$$s.t \ \ y_n[w^T\phi(x_n)+b] \geq 1-\xi_n \ \forall n$$

$$\xi_n \geq 0 \ \forall n$$

C = 0: Less complex Boundary

C = inf: More complex Boundary

Case 2: Non-Perfect Separation

Support Vector Machines

Method of Lagrange Multipliers

1. Obtain Primal form

$$\min_x f(x)$$

$$g_i(x) \leq 0$$

2. Rewrite as Lagrangian

$$L(x, \{\lambda_i\}) = f(x) + \sum_{i=1}^{n} \lambda_i g_i(x)$$

$$\lambda_i \geq 0$$

3. Solve $\frac{\delta L}{\delta x} = 0$

4. We have a set of $x = h(\{\lambda_i\})$

5. Substitute back in Lagrangian.

$$L(\{\lambda_i\}) = f(h(\{\lambda_i\})) + \sum_{i=1}^{n} \lambda_i g_i(h(\{\lambda_i\}))$$

6. Rewrite constraints

$$\max_{\{\lambda_i\} \geq 0} \min_x L(x, \{\lambda_i\})$$

Case 2: Non-Perfect Separation # Support Vector Machines

Determine Dual Form for SVM

1. Obtain Primal form (we did this)

$$\min_{w,b,\{\xi_n\}} \frac{1}{2}||w||_2 + C\sum_n \xi_n$$

$$s.t\ \ y_n[w^T\phi(x_n)+b] \geq 1-\xi_n\ \ \forall n$$

$$\xi_n \geq 0\ \forall n$$

2. Determine Lagrangian

$$L(w,b,\{\xi_n\},\{\lambda_n\},\{\alpha_n\})$$
$$= \left[\frac{1}{2}||w||_2 + C\sum_n \xi_n\right] + \sum_n [\alpha_n\{1-\xi_n - y_n[w^T\phi(x_n)+b]\}] + \sum_n \lambda_n(-\xi_n)$$

3, 4. Express primals in form of duals

$$\frac{\delta L}{\delta w} = w - \sum_n \alpha_n y_n \phi(x_n) = 0 \qquad w = \sum_n \alpha_n y_n \phi(x_n)$$

$$\frac{\delta L}{\delta b} = \sum_n \alpha_n y_n = 0 \qquad \sum_n \alpha_n y_n = 0$$

$$\frac{\delta L}{\delta \xi_n} = C - \alpha_n - \lambda_n = 0 \qquad C - \alpha_n - \lambda_n = 0$$

Support Vector Machines

Case 2: Non-Perfect Separation

Determine Dual Form for SVM

5. Substitute back in Lagrangian

$$L(w,b,\{\xi_n\},\{\lambda_n\},\{\alpha_n\}) = \left[\frac{1}{2}||w||_2 + C\sum_n \xi_n\right] + \sum_n [\alpha_n\{1-\xi_n - y_n[w^T\phi(x_n)+b]\}] + \sum_n \lambda_n(-\xi_n)$$

$$L(w,b,\{\xi_n\},\{\lambda_n\},\{\alpha_n\}) = \left[\frac{1}{2}\left(\sum_m \alpha_m y_m \boldsymbol{\phi}(x_m)\right)^T\left(\sum_n \alpha_n y_n \boldsymbol{\phi}(x_n)\right) + C\sum_n \xi_n\right] + \sum_n\left[\alpha_n\left\{1-\xi_n - y_n\left[\left(\sum_m \alpha_m y_m \boldsymbol{\phi}(x_m)\right)^T \phi(x_n)+b\right]\right\}\right] + \sum_n \lambda_n(-\xi_n)$$

$$L(w,b,\{\xi_n\},\{\lambda_n\},\{\alpha_n\}) = \left[\frac{1}{2}\sum_n \alpha_m\alpha_n y_m y_n \boldsymbol{\phi}^T(x_m)\boldsymbol{\phi}(x_n) + C\sum_n \xi_n\right] + \sum_n \alpha_n - \sum_n \alpha_n\xi_n + \sum_n \lambda_n(-\xi_n)$$

$$L(w,b,\{\xi_n\},\{\lambda_n\},\{\alpha_n\}) = \sum_n \alpha_n + \frac{1}{2}\sum_n \alpha_m\alpha_n y_m y_n \boldsymbol{\phi}^T(x_m)\boldsymbol{\phi}(x_n) + \sum_n \alpha_n(C-\xi_n-\lambda_n)$$

$$L(w,b,\{\xi_n\},\{\lambda_n\},\{\alpha_n\}) = \sum_n \alpha_n + \frac{1}{2}\sum_n \alpha_m\alpha_n y_m y_n \boldsymbol{\phi}^T(x_m)\boldsymbol{\phi}(x_n)$$

$$g(\{\lambda_n\},\{\alpha_n\}) = \sum_n \alpha_n + \frac{1}{2}\sum_n \alpha_m\alpha_n y_m y_n \boldsymbol{\phi}^T(x_m)\boldsymbol{\phi}(x_n)$$

Dual Form for SVM

$$\max_{\{\alpha_i\}} g(\{\lambda_n\},\{\alpha_n\}) = \sum_n \alpha_n + \frac{1}{2}\sum_n \alpha_m\alpha_n y_m y_n \boldsymbol{\phi}^T(x_m)\boldsymbol{\phi}(x_n)$$

$$\alpha_n, \lambda_n \geq 0 \;\forall n$$

$$\sum_n \alpha_n y_n = 0$$

$$C - \alpha_n - \lambda_n = 0$$

Case 2: Non-Perfect Separation

Support Vector Machines

Kernelization

Function with 2 properties

1. Symmetry $k(x_m, x_n) = k(x_n, x_m)$
2. Positive Semi-Definite $\sum_m \sum_n v_m v_n k(x_m, x_n) \geq 0$

$\boldsymbol{\phi}^T(x_m)\boldsymbol{\phi}(x_n) = k(x_m, x_n)$

$$\max_{\{\alpha_i\}} g(\{\lambda_n\}, \{\alpha_n\}) = \sum_n \alpha_n + \frac{1}{2} \sum_n \alpha_m \alpha_n y_m y_n \boldsymbol{\phi}^T(x_m)\boldsymbol{\phi}(x_n)$$

$$\max_{\{\alpha_n\},\{\lambda_n\}} g(\{\lambda_n\}, \{\alpha_n\}) = \sum_n \alpha_n + \frac{1}{2} \sum_n \alpha_m \alpha_n y_m y_n k(x_m, x_n)$$

such that $\alpha_n, \lambda_n \geq 0 \; \forall n$

$$\sum_n \alpha_n y_n = 0$$

$$C - \alpha_n - \lambda_n = 0$$

No ϕ terms!

Making Predictions

$$w^T\phi(x) = \left(\sum_n \alpha_n y_n \phi(x_n)\right)^T \phi(x)$$

$$w^T\phi(x) = \sum_n \alpha_n y_n \phi^T(x_n)\,\phi(x)$$

$$w^T\phi(x) = \sum_n \alpha_n y_n k(x_n, x)$$

RBF Kernel

$$k(x_m, x_n) = \exp\left(-\frac{||x_m - x_n||^2}{2\sigma^2}\right)$$

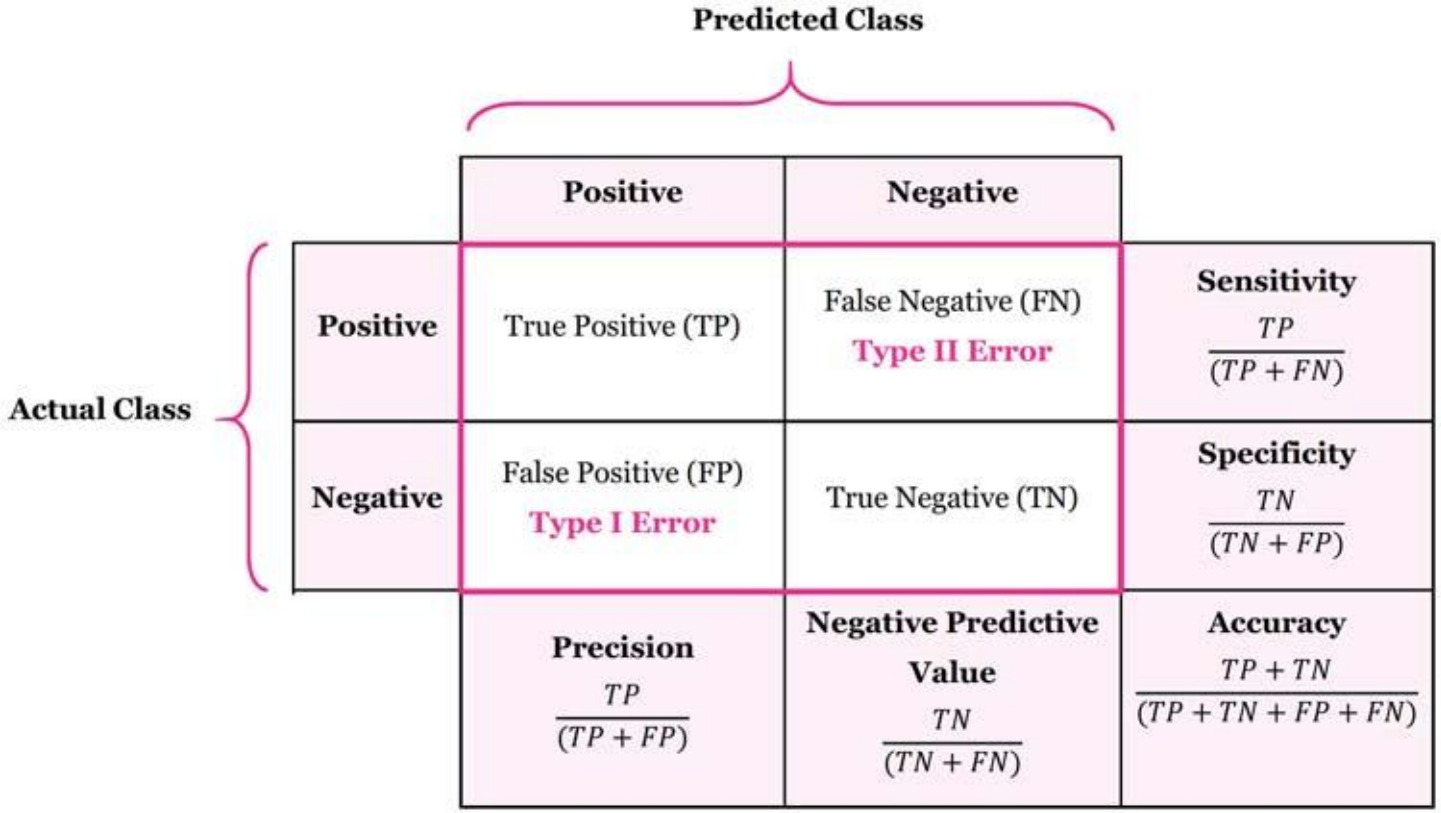
Kernel - Support Vector Machines (SVM)

- Common kernel functions for SVM
  - linear $k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1 \cdot \mathbf{x}_2$
  - polynomial $k(\mathbf{x}_1, \mathbf{x}_2) = (\gamma\, \mathbf{x}_1 \cdot \mathbf{x}_2 + c)^d$
  - Gaussian or radial basis $k(\mathbf{x}_1, \mathbf{x}_2) = \exp\left(-\gamma \left\|\mathbf{x}_1 - \mathbf{x}_2\right\|^2\right)$
  - sigmoid $k(\mathbf{x}_1, \mathbf{x}_2) = \tanh(\gamma\, \mathbf{x}_1 \cdot \mathbf{x}_2 + c)$

Classification Metrics

- **Accuracy**: Proportion of the total number of predictions that were correct.
- **Positive Predictive Value (or) Precision**: Proportion of positive cases that were correctly identified.
- **Negative Predictive Value** : Proportion of negative cases that were correctly identified.
- **Sensitivity (or) Recall** : Proportion of actual positive cases which are correctly identified.
- **Specificity** : Proportion of actual negative cases which are correctly identified.
- **F1 Score** : Harmonic mean of precision and recall values.

Confusion Matrix



$$F_1 = \left(\frac{\text{recall}^{-1} + \text{precision}^{-1}}{2} \right)^{-1} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$

Classification using *scikit-learn*

- Fruit Classification
  - Logistic Regression
  - KNN
  - Decision Tree
  - SVM
- Heart Disease Prediction Case Study with SVM Classifier

Ensemble Learning

What is Ensemble Learning..?

- Ensemble is the art of combining diverse set of learners (individual models) together to improvise on the stability and predictive power of the model.
- Diverse models can be created by:
  - Difference in population
  - Difference in hypothesis
  - Difference in modeling technique
  - Difference in initial seed

Simple Ensemble Techniques

- Max Voting
  - The max voting method is generally used for classification problems.
  - In this technique, multiple models are used to make predictions for each data point. The predictions by each model are considered as a ‘vote’.
  - The predictions which we get from the majority of the models are used as the final prediction.
- Averaging: Take Average of all Predictions
- Weighted Averaging: Take Weighted Average of Predictions

Python Script for Max Voting

```
» from sklearn.ensemble import VotingClassifier
» model1 = LogisticRegression(random_state=1)
» model2 = tree.DecisionTreeClassifier(random_state=1)
» model = VotingClassifier(estimators=[('lr', model1), ('dt', model2)],
  voting='hard')
» model.fit(x_train,y_train)
» model.score(x_test,y_test)
```

Advanced Ensemble techniques

- Stacking
- Blending
- Bagging
- Boosting

Stacking

- Stacking is an ensemble learning technique that uses predictions from multiple models (for example decision tree, knn or svm) to build a new model.
- This model is used for making predictions on the test set.

Stacking: Algorithm

1. The train set is split into 10 parts.
2. A base model (suppose a decision tree) is fitted on 9 parts and predictions are made for the 10th part. This is done for each part of the train set.
3. The base model (decision tree) is then fitted on the whole train dataset.
4. Using this model, predictions are made on the test set.
5. Steps 2 to 4 are repeated for another base model (say knn) resulting in another set of predictions for the train set and test set.
6. The predictions from the train set are used as features to build a new model.
7. This model is used to make final predictions on the test prediction set.

Blending

- Blending follows the same approach as stacking but uses only a holdout (validation) set from the train set to make predictions.
- In other words, unlike stacking, the predictions are made on the holdout set only.
- The holdout set and the predictions are used to build a model which is run on the test set.

Blending: Algorithm

1. The train set is split into training and validation sets.
2. Model(s) are fitted on the training set.
3. The predictions are made on the validation set and the test set.
4. The validation set and its predictions are used as features to build a new model.
5. This model is used to make final predictions on the test and meta-features.

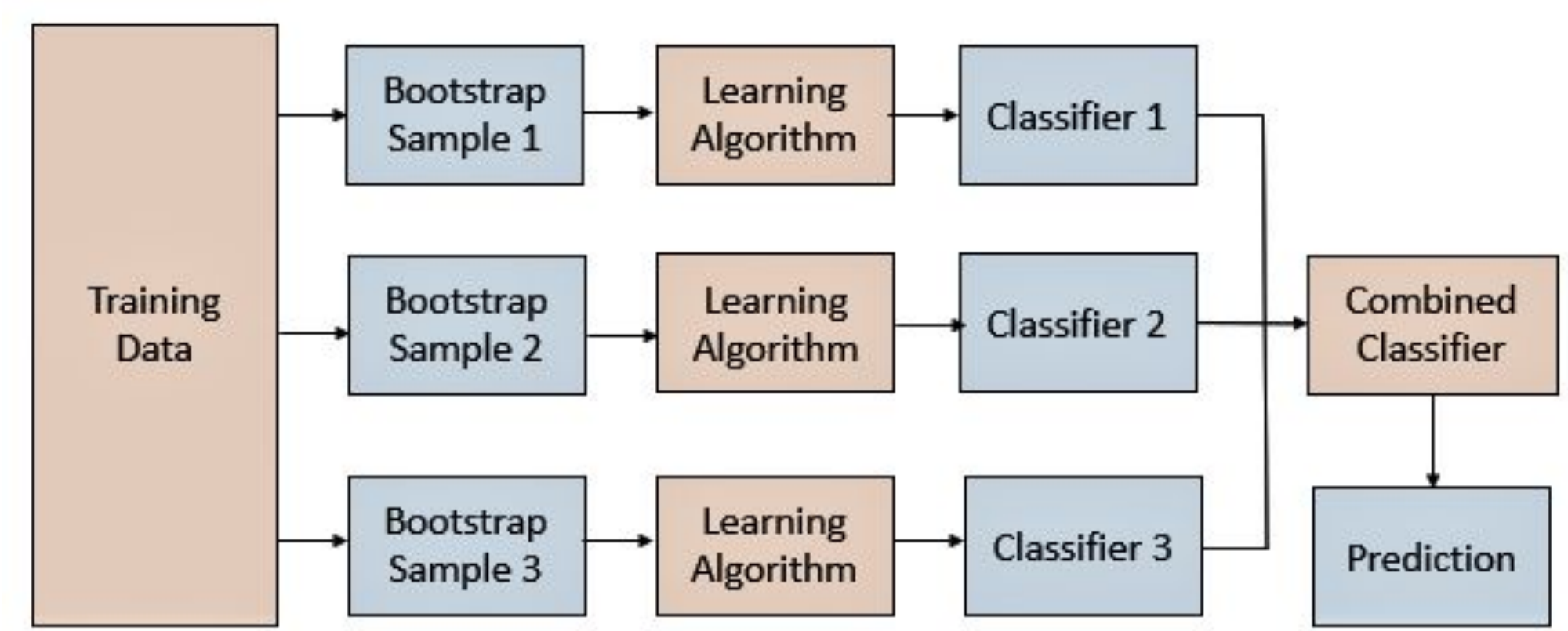
Question…?

- If we create all the models on the same set of data and combine it, will it be useful…?
- There is a high chance that these models will give the same result since they are getting the same input.
- So how can we solve this problem…?

Bagging

- One of the techniques is Bagging (or Bootstrap Aggregating).
- The idea behind bagging is combining the results of multiple models (for instance, all decision trees) to get a generalized result.
- Bootstrapping is a sampling technique in which we create subsets of observations from the original dataset, with replacement.
- The size of the subsets is not as same as the size of the original set.
- Bagging technique uses these subsets (bags) to get a fair idea of the distribution (complete set).
- The size of subsets created for bagging may be less than the original set.

Bagging


Training Data
Bootstrap Sample 1
Bootstrap Sample 2
Bootstrap Sample 3
Learning Algorithm
Learning Algorithm
Learning Algorithm
Classifier 1
Classifier 2
Classifier 3
Combined Classifier
Prediction

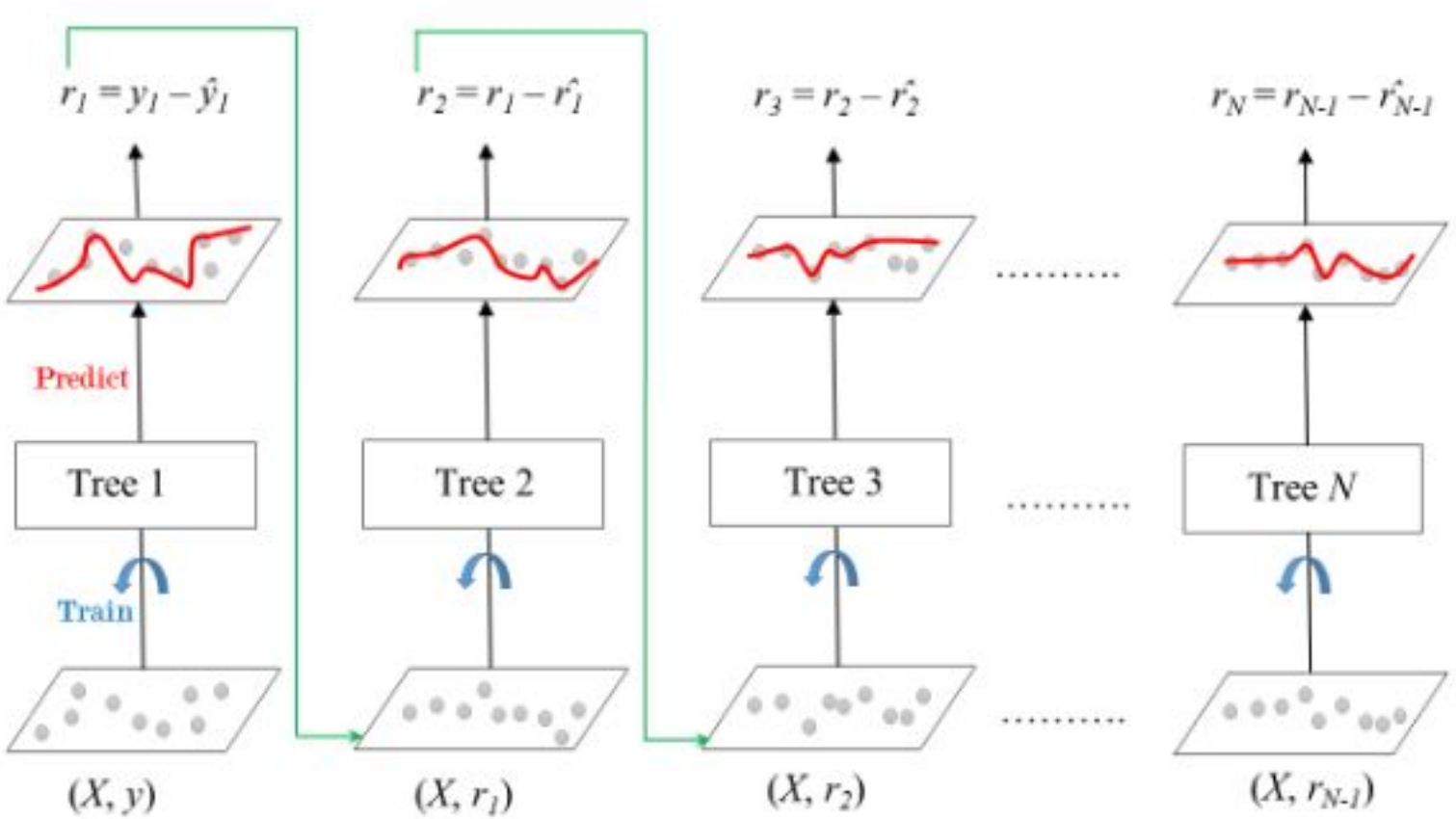
Another Question..?

- If a data point is incorrectly predicted by the first model, and then the next (probably all models), will combining the predictions provide better results?

Boosting

- Boosting is a sequential process, where each subsequent model attempts to correct the errors of the previous model.
- The succeeding models are dependent on the previous model.

Boosting


$r_1 = y_1 - \hat{y}_1$
$r_2 = r_1 - \hat{r_1}$
$r_3 = r_2 - \hat{r_2}$
$r_N = r_{N-1} - \hat{r}_{N-1}$
Predict
Tree 1
Tree 2
Tree 3
Tree N
Train
(X, y)
(X, r_1)
(X, r_2)
(X, r_{N-1})

Boosting: Algorithm

1. A subset is created from the original dataset.
2. Initially, all data points are given equal weights.
3. A base model is created on this subset.
4. This model is used to make predictions on the whole dataset.
5. Errors are calculated using the actual values and predicted values.
6. The observations which are incorrectly predicted, are given higher weights.
7. Another model is created and predictions are made on the dataset. (This model tries to correct the errors from the previous model)
8. Similarly, multiple models are created, each correcting the errors of the previous model.
9. The final model (strong learner) is the weighted mean of all the models (weak learners).

Bagging & Boosting Algorithms

- Bagging algorithms:
  - Bagging meta-estimator (BaggingClassifier & BaggingRegressor)
  - Random forest
- Boosting algorithms:
  - AdaBoost (AdaBoostClassifier & AdaBoostRegressor)
  - GBM (GradientBoostingClassifier & GradientBoostingRegressor)
  - XGBM (xgboost. XGBClassifier & XGBRegressor)
  - Light GBM
  - Cat Boost (catboost.CatBoostClassifier & CatBoostRegressor)

Reference: https://www.analyticsvidhya.com/blog/2018/06/comprehensive-guide-for-ensemble-models/

Unsupervised Learning

Unsupervised Learning

- Unsupervised learning is where you only have input data (X) and no corresponding output variables.
- The goal for unsupervised learning is to model the underlying structure or distribution in the data in order to learn more about the data.
- These are called unsupervised learning because unlike supervised learning there are no class labels and there is no teacher.
- Algorithms are left to their own devises to discover and present the interesting structures in the data.

Types of Unsupervised Learning

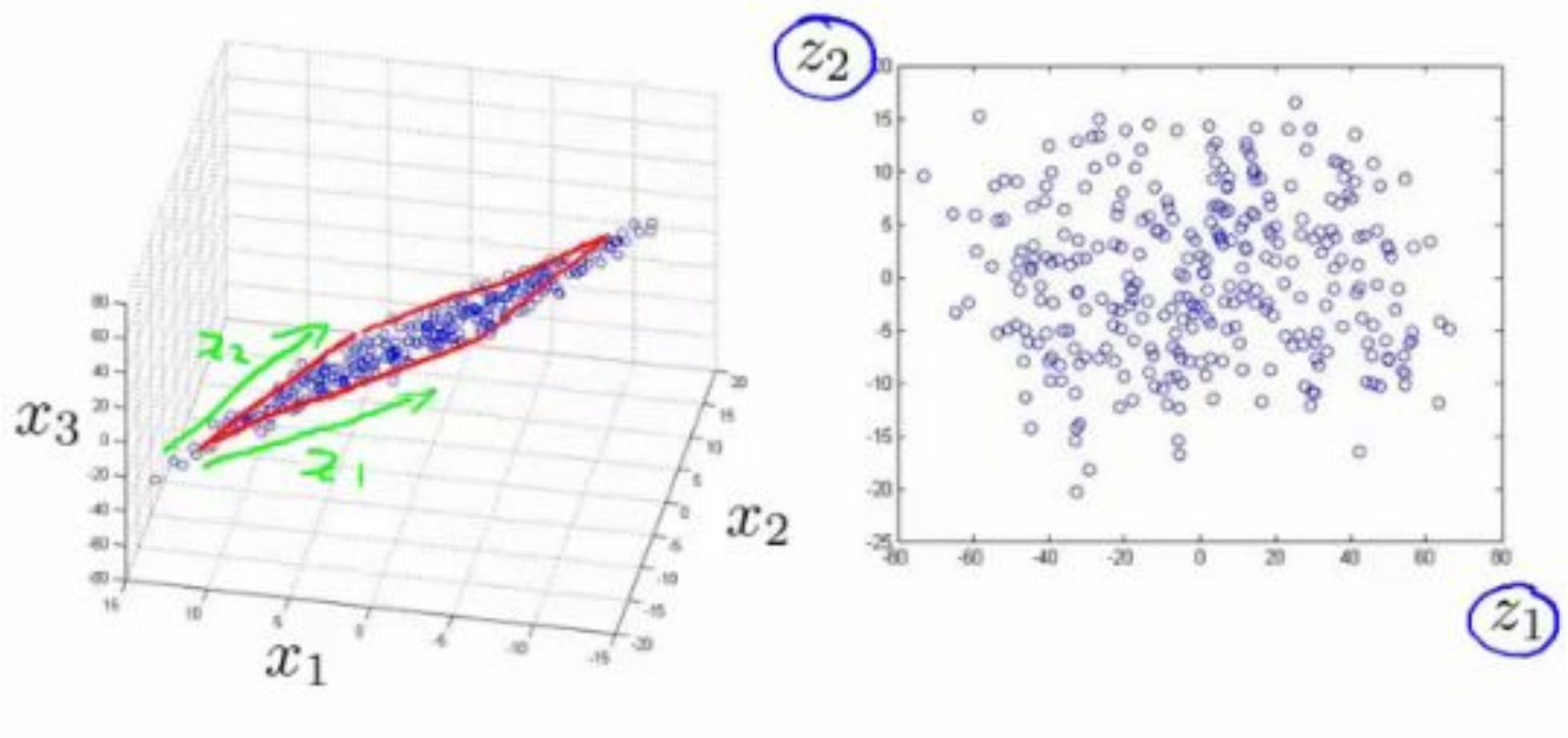
- Unsupervised learning problems can be further grouped into:
  - Dimensionality Reduction
  - Clustering

- **Dimensionality Reduction** - It means reducing the number of features (or) dimensions present in the data.
- **Clustering**: A clustering problem is where you want to discover the inherent groupings in the data.

What is Dimensionality Reduction…?

- Reducing the number of random variables under consideration by obtaining a set of principal variables.
- Removing features that may be redundant or correlated or overlapping.

Dimensionality Reduction

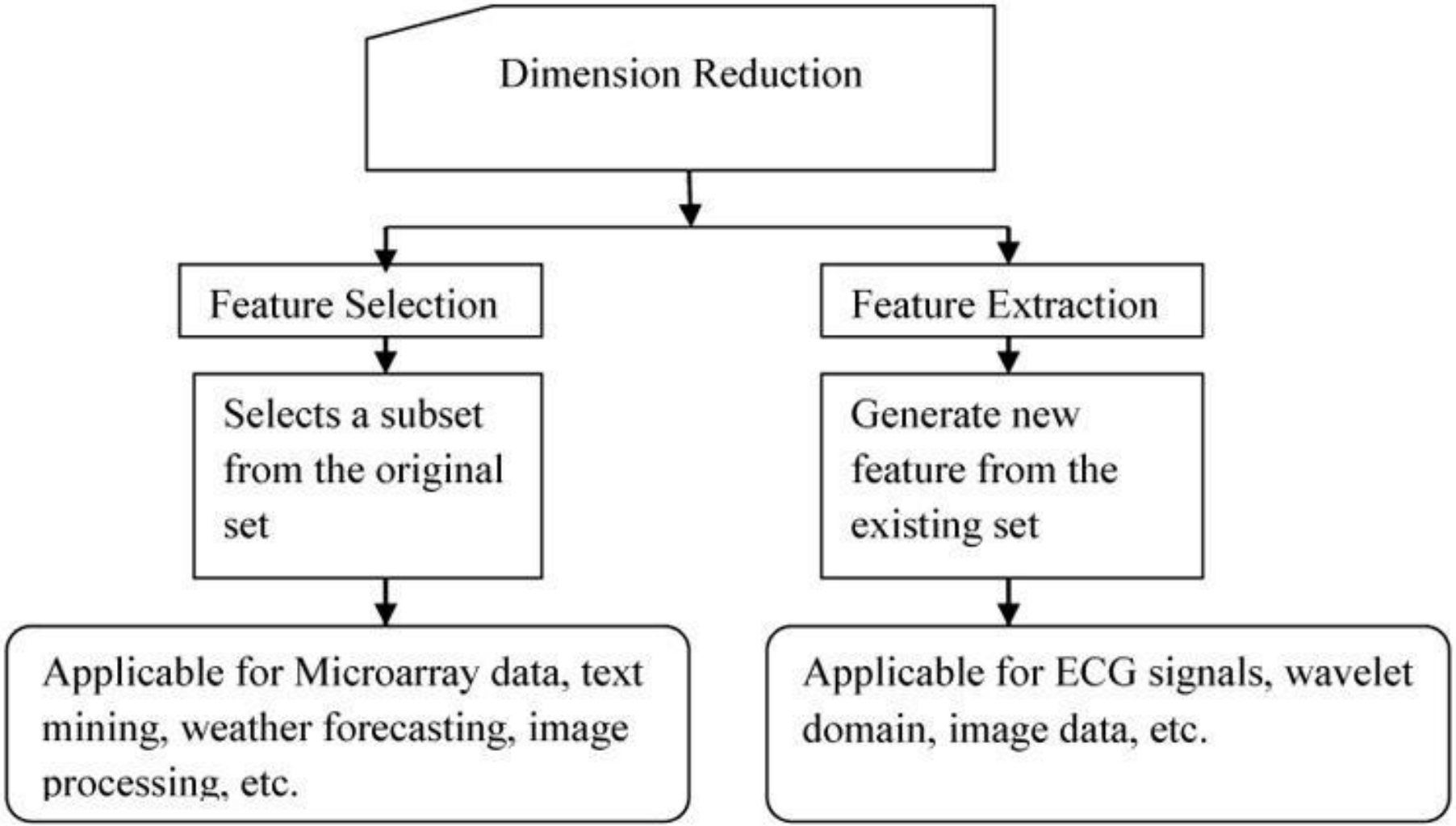


Why Dimensionality Reduction…?

- **Less Features**:
  - Easy to interpret
  - Less likely to overfit
  - Low prediction accuracy

- **More Features**:
  - Difficult to interpret
  - More likely to overfit
  - High prediction accuracy

Types of Dimensionality Reduction



Feature Selection

- Select a subset of input features from the dataset.
- It is common to use correlation type statistical measures between input and output variables as the basis for feature selection.
- The choice of statistical measures is highly dependent upon the variable data types.
- Common data types include numerical and categorical.
- Each may be further subdivided such as integer and floating point for numerical variables, and Boolean, ordinal, or nominal for categorical variables.

Why Feature Selection…?

- To improve performance in terms of
  - Speed
  - predictive power
  - simplicity of the model.
- To visualize the data for model selection.
- To reduce dimensionality and remove noise.

Process of Feature Selection

- Find Predictive power of each feature with respect to target
- Individual features are ranked according to specific statistical measure.
- The top N features are then selected.
- **Issue:** Different types of Input and Output Data.

Feature\Response	Continuous	Categorical
Continuous	Pearson's Correlation	LDA
Categorical	Anova	Chi-Square

Statistical Measure

- **Pearson's Correlation:** It is used as a measure for quantifying linear dependence between two continuous variables X and Y.

$$\rho_{X,Y} = \frac{\operatorname{cov}(X,Y)}{\sigma_X \sigma_Y}$$

- **LDA**: It is used to find a linear combination of features that characterizes or separates two or more classes of a categorical variable.

$$J(\mathbf{w}) = \frac{\mathbf{w}^T S_B \mathbf{w}}{\mathbf{w}^T S_W \mathbf{w}}$$

Statistical Measure

- **ANOVA:** It provides a statistical test of whether the means of several groups are equal or not.

$$F_{\text{stat}} = \frac{MS_{between}}{MS_{within}}$$

- **Chi-Square:** applied to the groups of categorical features to evaluate the likelihood of correlation or association between them.

$$\chi_c^2 = \sum \frac{(O_i - E_i)^2}{E_i}$$

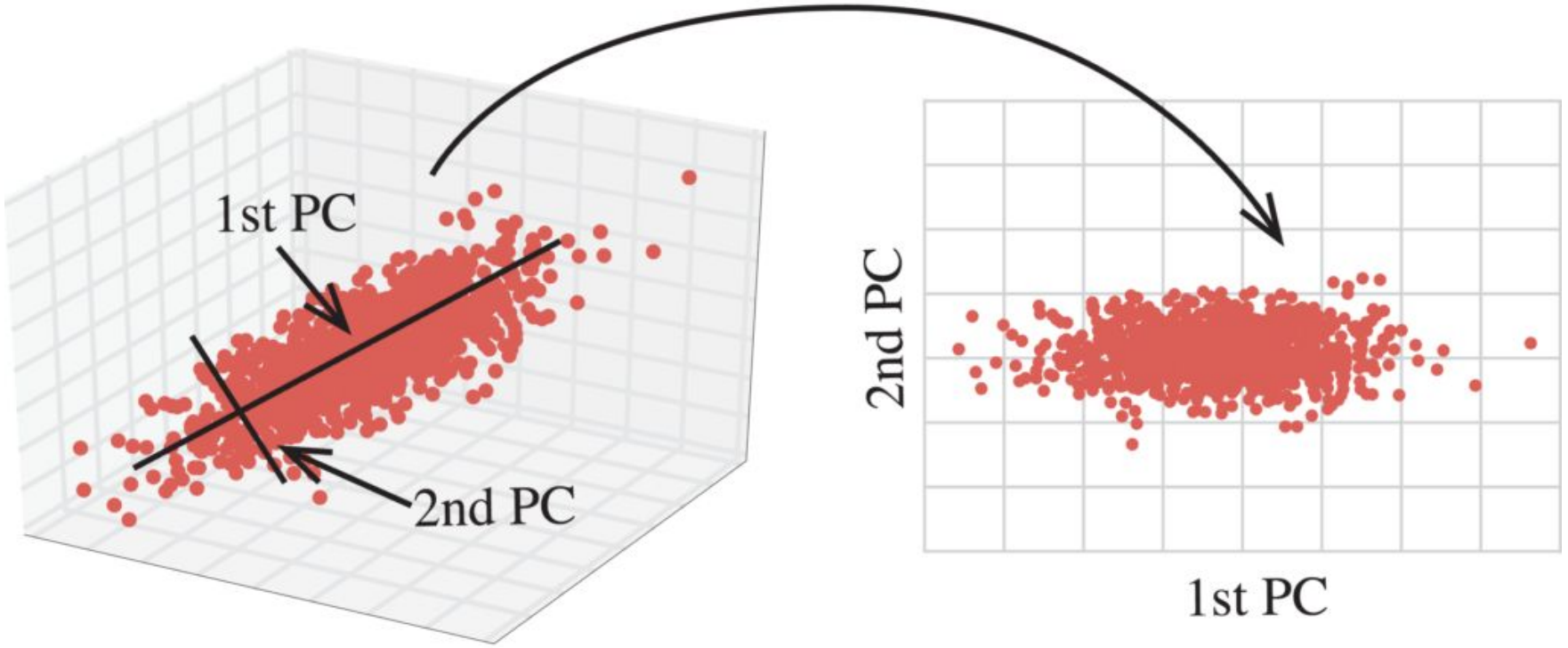
Feature Selection : sci-kit learn

- The scikit-learn library provides an implementation of most of the useful statistical measures.
  - Pearson's Correlation Coefficient: f_regression()
  - ANOVA: f_classif()
  - Chi-Squared: chi2()
  - Mutual Information: mutual_info_classif() and mutual_info_regression()

Feature Extraction

- The process of projecting the data of many dimensions to fewer dimensions, i.e. projecting to lesser dimension space.
- PCA is widely used for projecting higher dimensions to lesser dimension space.

Feature Extraction

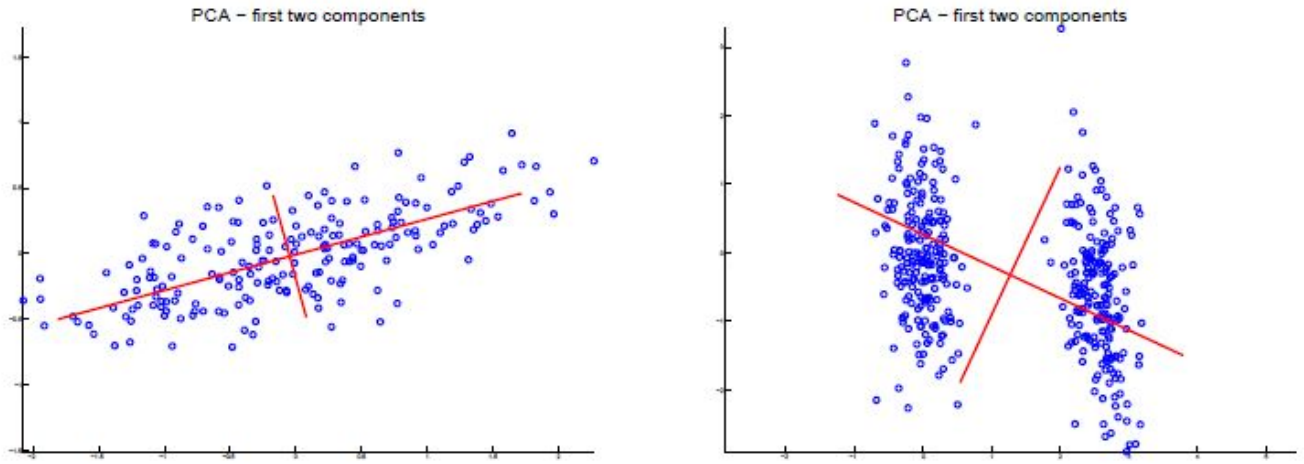


Principal Components Analysis (PCA)

- This is one of the most famous technique for dimensionality reduction.
- Also known as Karhonen-Loeve (KL) transform.
- It is linear dimensionality reduction. It is unsupervised i.e. with no information about classes.
- PCA searches for directions in the data that have largest variance and subsequently project the data onto it.
- Max variance direction can be obtained by the eigen vectors of the covariance matrix of the original data.

Principal Components Analysis (PCA)

- **Goal** – reduce the dimensionality of data while preserving the variation present in dataset.
- If there are d dimensions for given dataset then using PCA we can project from 1 to d dimensions.



1. Compute the mean feature vector

$$\mu = \frac{1}{p}\sum_{k=1}^{p} x_k$$, where, x_k is a pattern ($k = 1$ to p), p = number of patterns, x is the feature matrix

2. Find the covariance matrix

$$C = \frac{1}{p}\sum_{k=1}^{p}\{x_k - \mu\}\{x_k - \mu\}^T$$ where, T represents matrix transposition

3. Compute Eigen values λ_i and Eigen vectors v_i of covariance matrix

   $Cv_i = \lambda_i v_i$ $(i = 1, 2, 3,....q)$, q = number of features

4. Estimating high-valued Eigen vectors

   (i) Arrange all the Eigen values (λ_i) in descending order

   (ii) Choose a threshold value, θ

   (iii) Number of high-valued λ_i can be chosen so as to satisfy the relationship

$$\left(\sum_{i=1}^{s}\lambda_i\right)\left(\sum_{i=1}^{q}\lambda_i\right)^{-1} \geq \theta$$, where, s = number of high valued λ_i chosen

   (iv) Select Eigen vectors corresponding to selected high valued λ_i

5. Extract low dimensional feature vectors (principal components) from raw feature matrix.

   $P = V^T x$, where, V is the matrix of principal components and x is the feature matrix

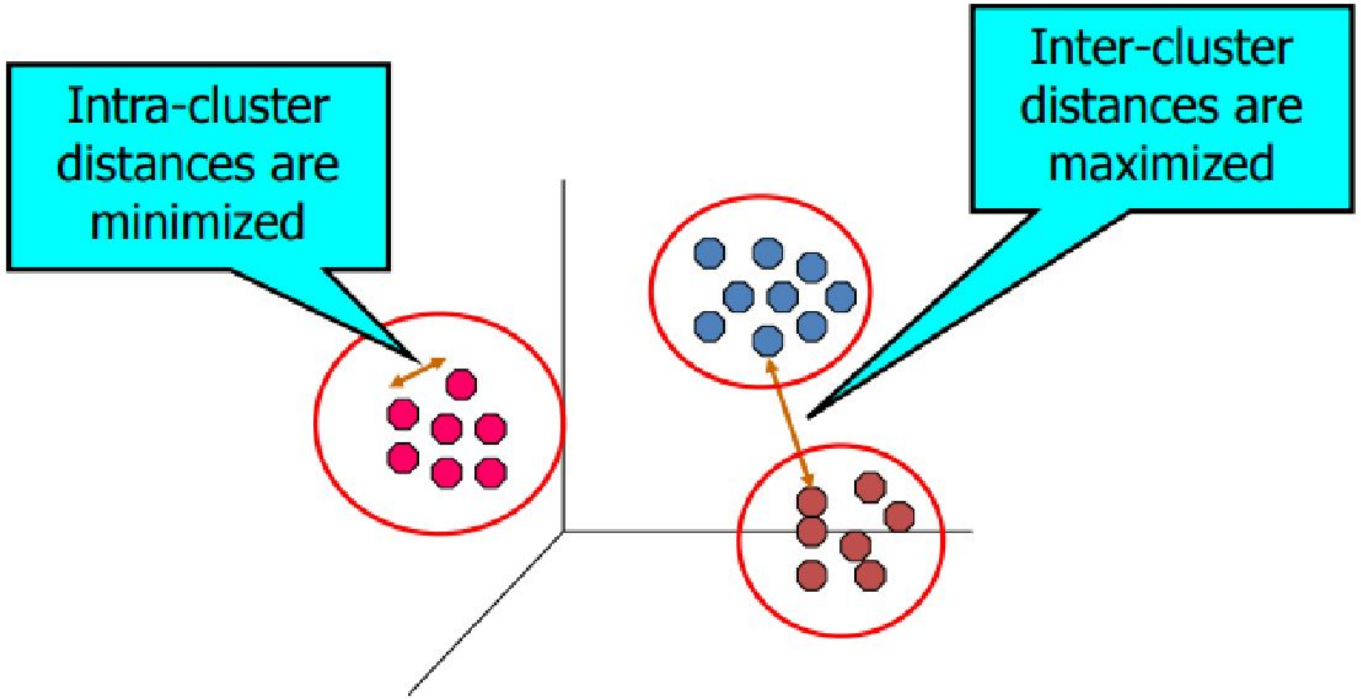
Other Feature Extraction Methods

- Linear Discriminant Analysis
- Quadratic Discriminant Analysis
- Independent Component Analysis
- T-SNE

Clustering

What is Clustering…?

Finding groups of objects such that the objects in a group will be similar (or related) to one another and different from (or unrelated to) the objects in other groups



Why Clustering…?

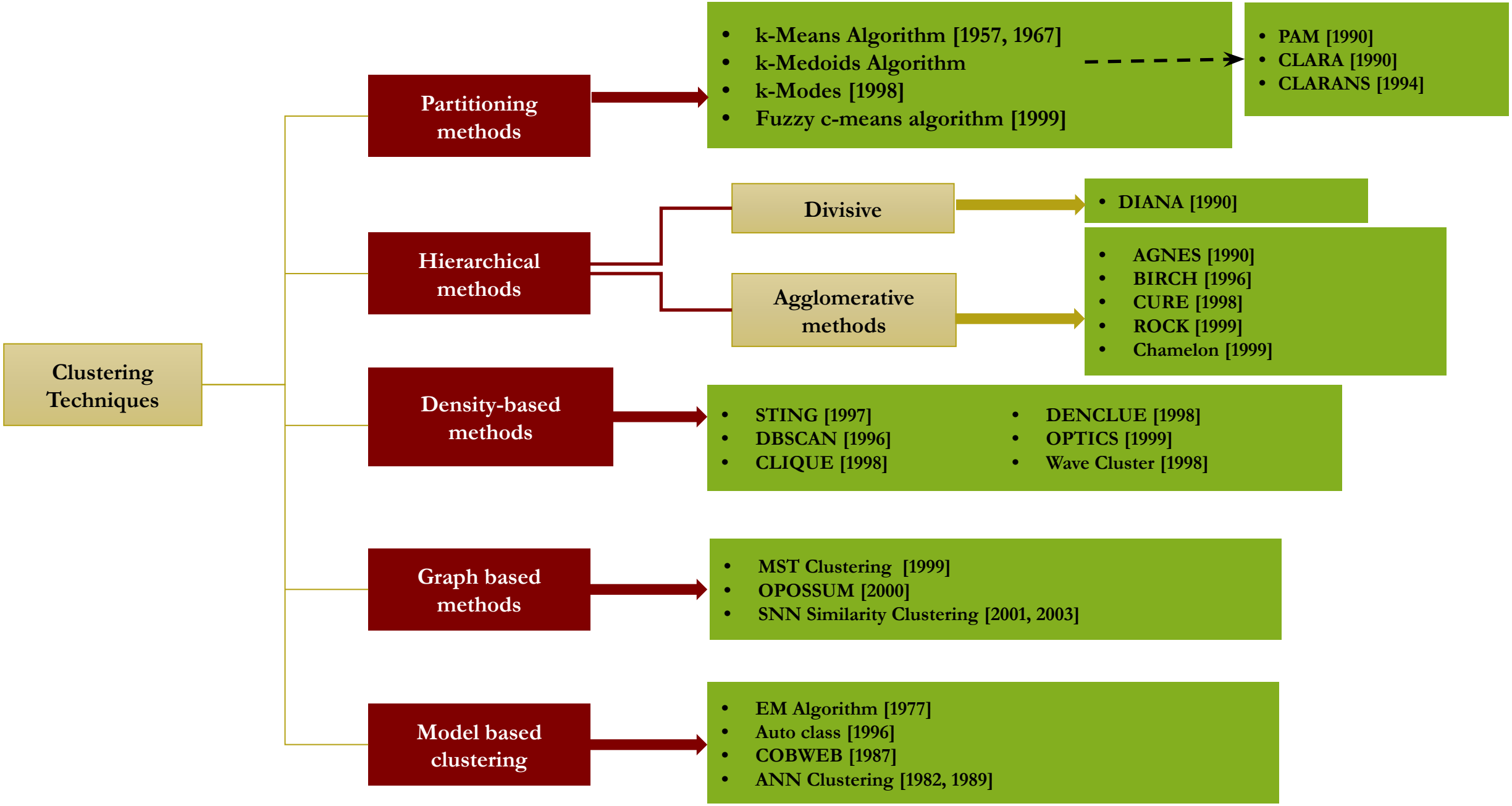
- Labelling a large set of sample patterns can be costly
- The contents of the database may not be known
- Clustering can be used for finding features that will later be useful for categorization
- It may help to gain insight into the nature of the data
- It may lead to discovery of distinct subclasses or similarities among patterns

Applications

- **Marketing:** finding groups of customers with similar behavior given a large database of customer data containing their properties and past buying records;
- **Biology:** classification of plants and animals given their features;
- **Libraries:** book ordering; • Insurance: identifying groups of motor insurance policy holders with a high average claim cost; identifying frauds;
- **City-planning:** identifying groups of houses according to their house type, value and geographical location;
- **Earthquake studies:** clustering observed earthquake epicentres to identify dangerous zones;
- **WWW:** document classification; clustering weblog data to discover groups of similar access patterns.

Major Clustering Approaches

- Partitioning-based Algorithms
- Hierarchical Algorithms
- Density-based Algorithms
- Graph based Algorithms
- Model-based Algorithms

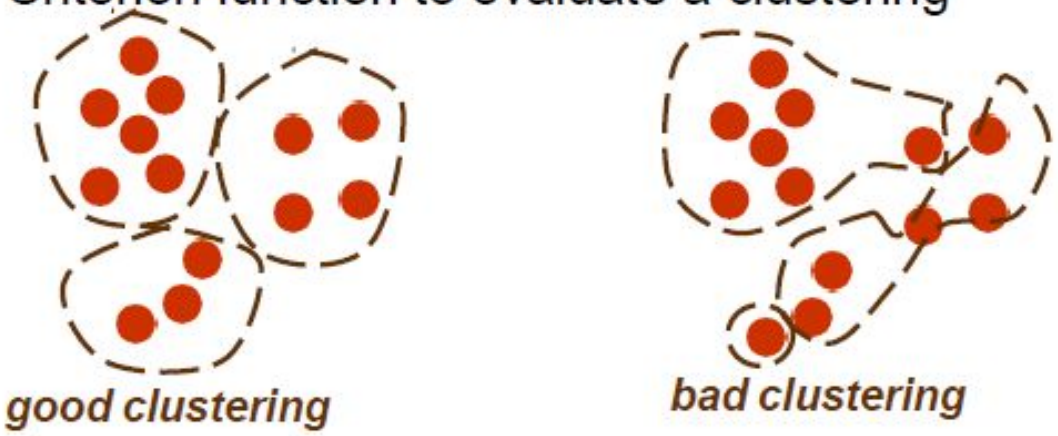


Nagur Shareef Shaik (1800420)

What we Need for Clustering

1. Proximity measure, *either*
   - similarity measure $s(x_i,x_k)$: large if x_i,x_k are similar
   - dissimilarity(or distance) measure $d(x_i,x_k)$: small if x_i,x_k are similar

2. Criterion function to evaluate a clustering

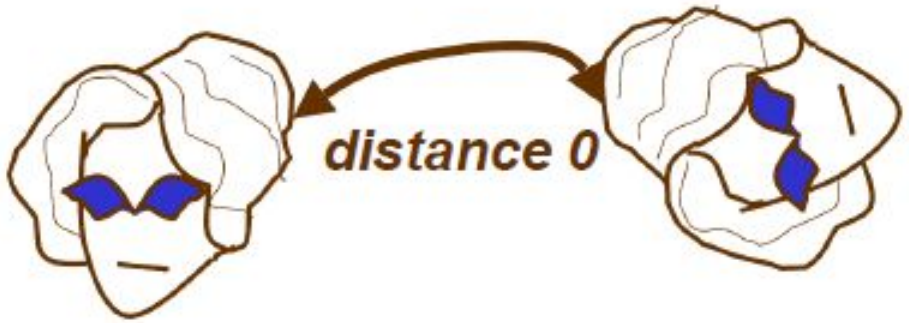


3. Algorithm to compute clustering
   For example, by optimizing the criterion function

Proximity Measures

good proximity measure is application dependent

- Clusters should be invariant under the transformations "natural" to the problem
- For example for object recognition, should have invariance to rotation



- For character recognition, no invariance to rotation

Distance on Numeric Data: Minkowski Distance

- *Minkowski distance*: A popular distance measure

$$d(i, j) = \sqrt[h]{|x_{i1} - x_{j1}|^h + |x_{i2} - x_{j2}|^h + \cdots + |x_{ip} - x_{jp}|^h}$$

where $i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ and $j = (x_{j1}, x_{j2}, \ldots, x_{jp})$ are two p-dimensional data objects, and h is the order (the distance so defined is also called L-h norm)

- Properties
  - d(i, j) > 0 if i ≠ j, and d(i, i) = 0 (Positive definiteness)
  - d(i, j) = d(j, i) (Symmetry)
  - d(i, j) ≤ d(i, k) + d(k, j) (Triangle Inequality)
- A distance that satisfies these properties is a metric

Special Cases of Minkowski Distance

- *h* = 1: Manhattan (city block, L_1 norm) distance
  - E.g., the Hamming distance: the number of bits that are different between two binary vectors

$$d(i,j)=|x_{i_1}-x_{j_1}|+|x_{i_2}-x_{j_2}|+...+|x_{i_p}-x_{j_p}|$$

- *h* = 2: (L_2 norm) Euclidean distance

$$d(i,j)=\sqrt{(|x_{i_1}-x_{j_1}|^2+|x_{i_2}-x_{j_2}|^2+...+|x_{i_p}-x_{j_p}|^2)}$$

- $h \rightarrow \infty$. "supremum" (L_{max} norm, L_∞ norm) distance.
  - This is the maximum difference between any component (attribute) of the vectors

$$d(i,j)=\lim_{h\rightarrow\infty}\left(\sum_{f=1}^{p}|x_{if}-x_{jf}|^h\right)^{\frac{1}{h}}=\max_{f}^{p}|x_{if}-x_{jf}|$$

Cosine Similarity

- A **document** can be represented by thousands of attributes, each recording the *frequency* of a particular word (such as keywords) or phrase in the document.

Document	*team*	*coach*	*hockey*	*baseball*	*soccer*	*penalty*	*score*	*win*	*loss*	*season*
Document1	5	0	3	0	2	0	0	2	0	0
Document2	3	0	2	0	1	1	0	1	0	1
Document3	0	7	0	2	1	0	0	3	0	0
Document4	0	1	0	0	1	2	2	0	3	0

Cosine Similarity

- Other vector objects: gene features in micro-arrays, …
- Applications: information retrieval, biologic taxonomy, gene feature mapping, ...
- Cosine measure: If d_1 and d_2 are two vectors (e.g., term-frequency vectors), then

$$\cos(d_1, d_2) = (d_1 \cdot d_2) / ||d_1|| \, ||d_2|| ,$$

where · indicates vector dot product, $||d||$: the length of vector d

Proximity Measure for Binary Attributes

- A contingency table for binary data

Object *i* \ Object *j*	1	0	sum
1	q	r	$q+r$
0	s	t	$s+t$
sum	$q+s$	$r+t$	p

- Distance measure for symmetric binary variables:

$$d(i,j) = \frac{r+s}{q+r+s+t}$$

- Distance measure for asymmetric binary variables:

$$d(i,j) = \frac{r+s}{q+r+s}$$

- Jaccard coefficient (*similarity* measure for *asymmetric* binary variables):

$$sim_{Jaccard}(i,j) = \frac{q}{q+r+s}$$

- Note: Jaccard coefficient is the same as "coherence":

$$coherence(i,j) = \frac{sup(i,j)}{sup(i)+sup(j)-sup(i,j)} = \frac{q}{(q+r)+(q+s)-q}$$

Dissimilarity between Binary Variables

- Example

Name	Gender	Fever	Cough	Test-1	Test-2	Test-3	Test-4
Jack	M	Y	N	P	N	N	N
Mary	F	Y	N	P	N	P	N
Jim	M	Y	P	N	N	N	N

- Gender is a symmetric attribute
- The remaining attributes are asymmetric binary
- Let the values Y and P be 1, and the value N be 0

$$d(jack, mary) = \frac{0+1}{2+0+1} = 0.33$$

$$d(jack, jim) = \frac{1+1}{1+1+1} = 0.67$$

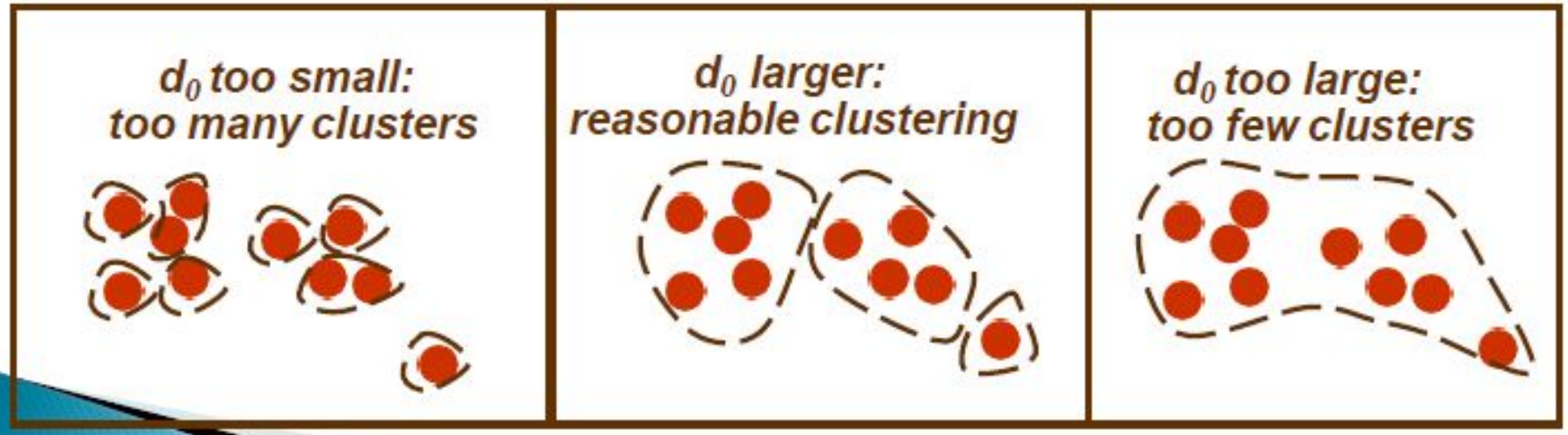
$$d(jim, mary) = \frac{1+2}{1+1+2} = 0.75$$

Simplest Clustering Algorithm

Having defined a distance function

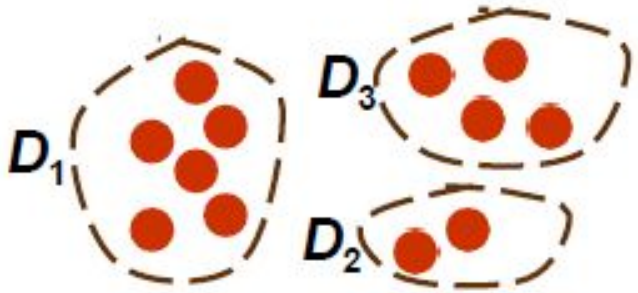
- go over all sample pairs, and put them in the same cluster if the distance between them is less then some threshold distance d_0 (or if similarity is larger than s_0)

- Pros: simple to understand and implement
- Cons: very dependent on d_0 (or s_0), automatic choice of d_0 (or s_0)is not an easily solved issue



Criterion Functions for Clustering

- Have samples $x_1,\ldots,x_n$
- Suppose partitioned samples into c subsets $D_1,\ldots,D_c$



- There are approximately $c^n/c!$ distinct partitions
- Can define a criterion function $J(D_1,\ldots,D_c)$ which measures the quality of a partitioning $D_1,\ldots,D_c$
- Then the clustering problem is a well defined problem
  - the optimal clustering is the partition which optimizes the criterion function

Challenges

✔ What cost function to use?

✔ What underlying structure to assume?

✔ How to measure similarity?

✔ How to decide on the number of clusters?

- Different answers to these questions may lead to different clustering algorithms and different clustering outcomes.
- Common objective: generalize well.

Clustering examples

Image segmentation

Goal: Break up the image into meaningful or perceptually similar regions

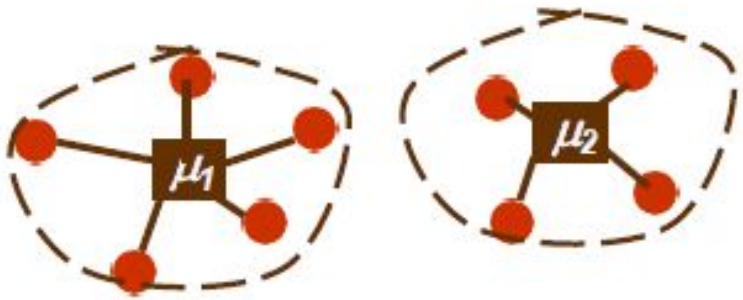
SSE Criterion Function

- Let n_i be the number of samples in D_i, and define the mean of samples in in D_i

$$\mu_i = \frac{1}{n_i} \sum_{x \in D_i} x$$

- Then the sum-of-squared errors criterion function (to minimize) is:

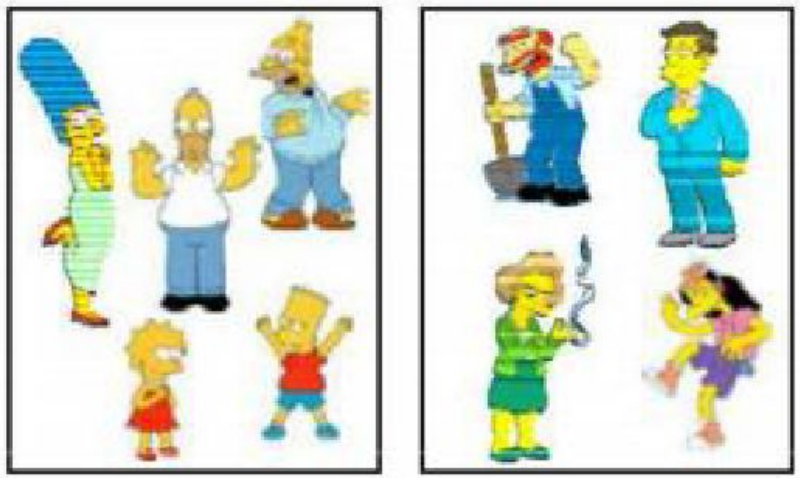
$$J_{SSE} = \sum_{i=1}^{c} \sum_{x \in D_i} || x - \mu_i ||^2$$



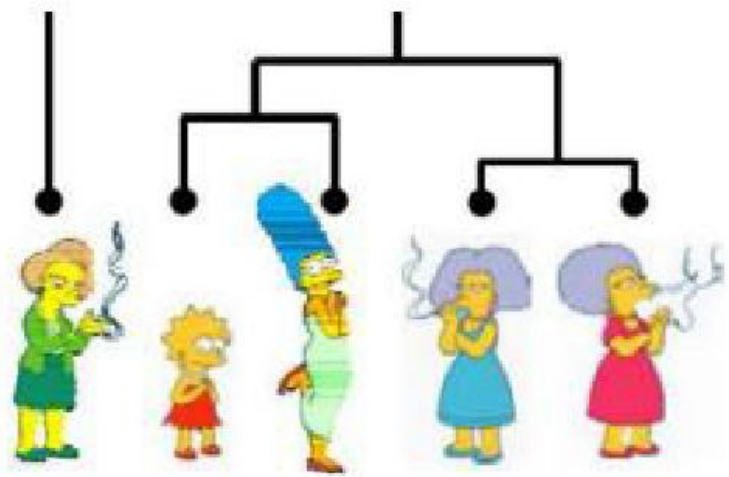
- Note that the number of clusters, c, is fixed

Clustering algorithms

- Partition algorithms (Flat)
  - K-means
  - Mixture of Gaussian
  - Spectral Clustering

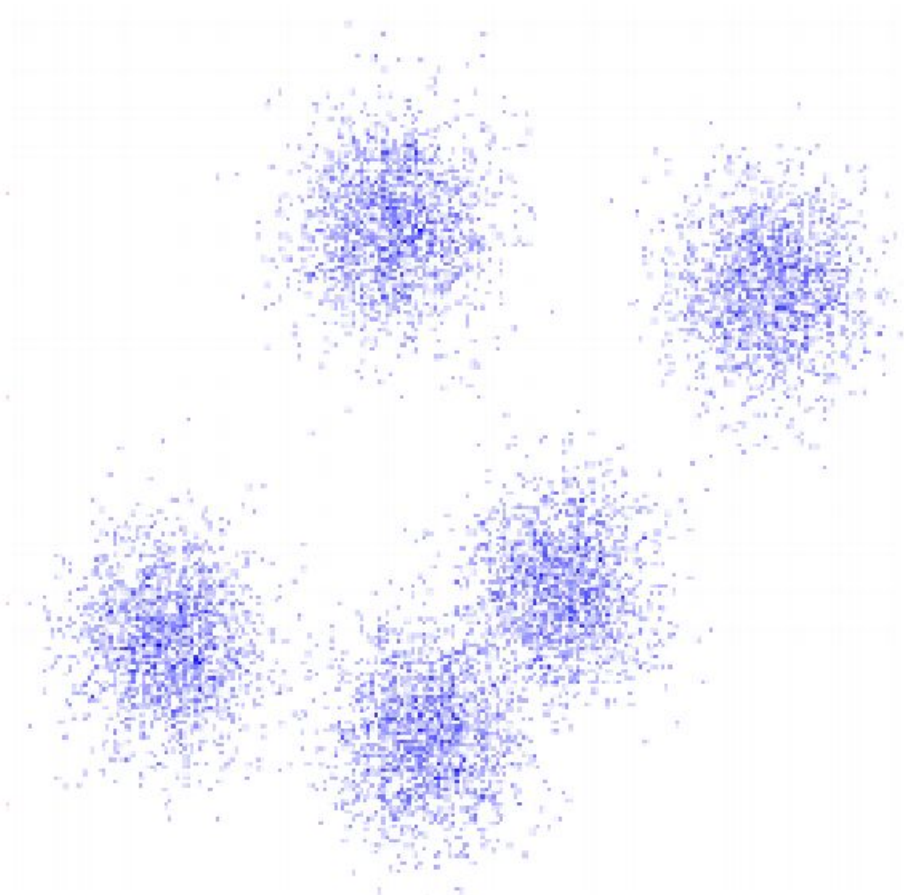


- Hierarchical algorithms
  - Bottom up – agglomerative
  - Top down – divisive



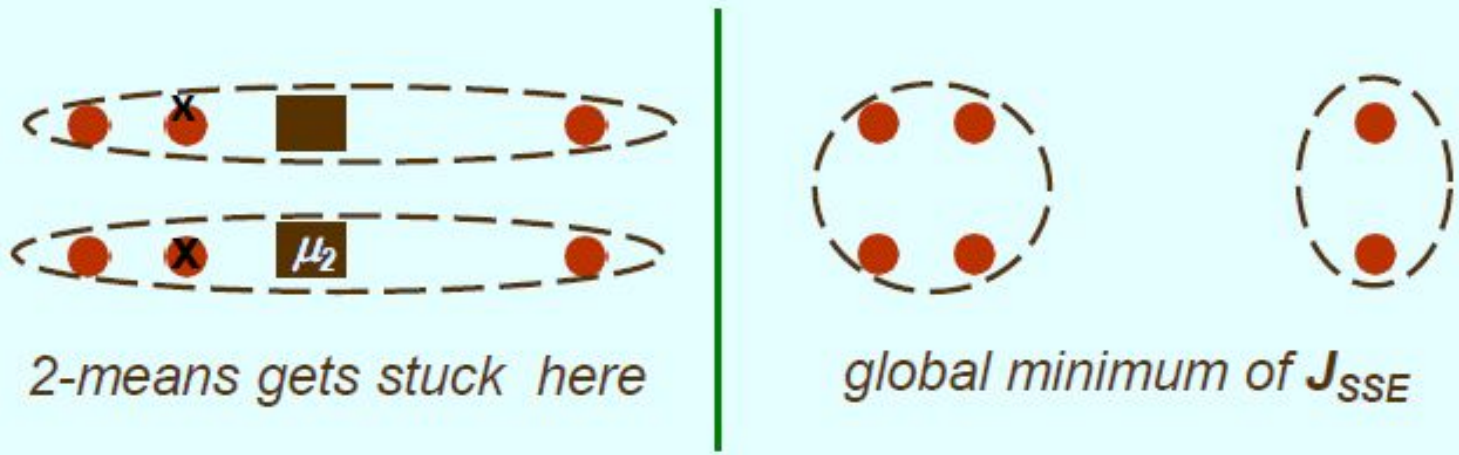
K-Means

- An iterative clustering algorithm
  - Initialize: Pick *K* random points as cluster centers
  - Alternate:
    1. Assign data points to closest cluster center
    2. Change the cluster center to the average of its assigned points
  - Stop when no points' assignments change



K-means Clustering

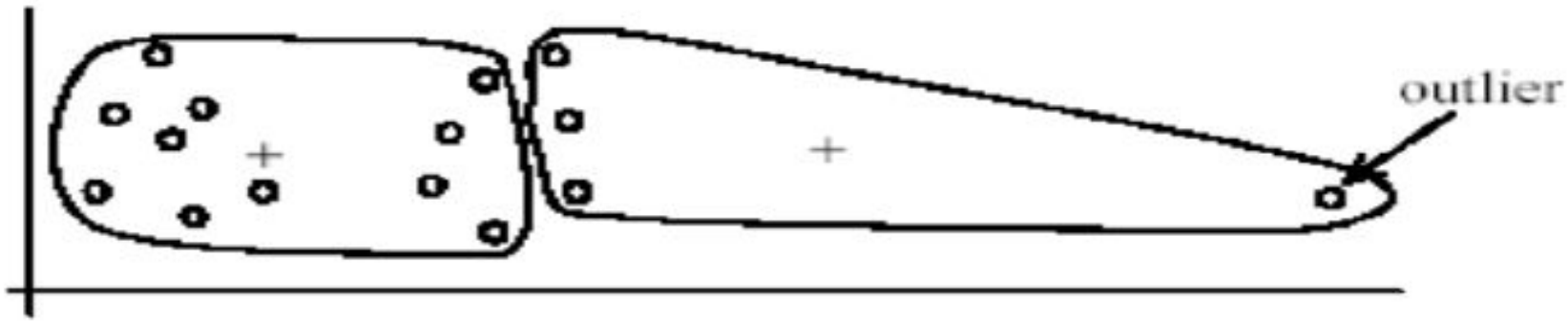
- Simple: easy to understand and to implement
- Requires initial cluster centers – It does matter what you pick!
- However the algorithm is not guaranteed to find a global minimum
- The algorithm is sensitive to outliers



2-means gets stuck here

global minimum of J_{SSE}

Outliers



(A): Undesirable clusters

(B): Ideal clusters

Example: K-Means for Segmentation

K=2

Goal of Segmentation is to partition an image into regions each of which has reasonably homogenous visual appearance.

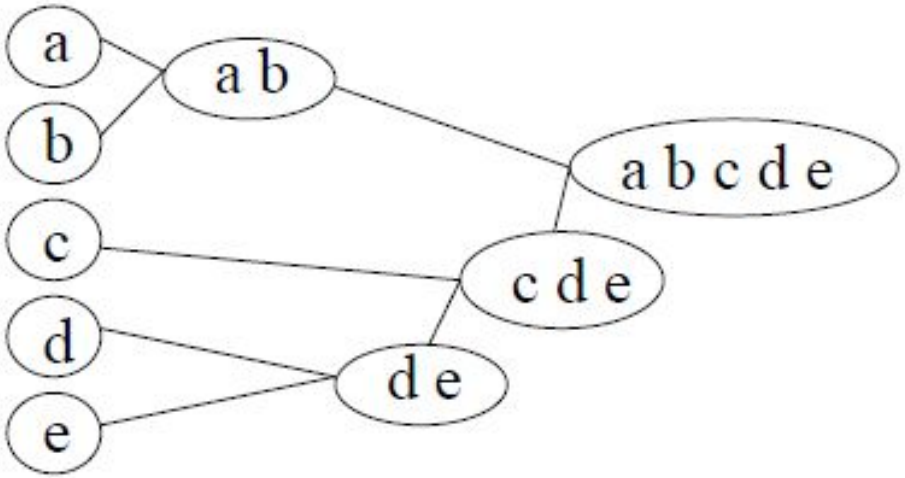
Original

Hierarchical Clustering Algorithms

- Two main types of hierarchical clustering
  - **Agglomerative:**
    - Start with the points as individual clusters
    - At each step, merge the closest pair of clusters until only one cluster (or **k** clusters) left
  - **Divisive:**
    - Start with one, all-inclusive cluster
    - At each step, split a cluster until each cluster contains a point (or there are **k** clusters)
- Traditional hierarchical algorithms use a similarity or distance matrix
  - Merge or split one cluster at a time

Hierarchical Clustering

- Agglomerative approach



Initialization:

Each object is a cluster

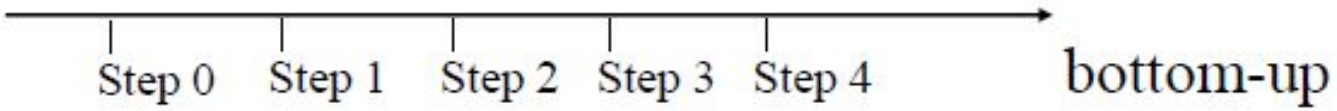
Iteration:

Merge two clusters which are

most similar to each other;

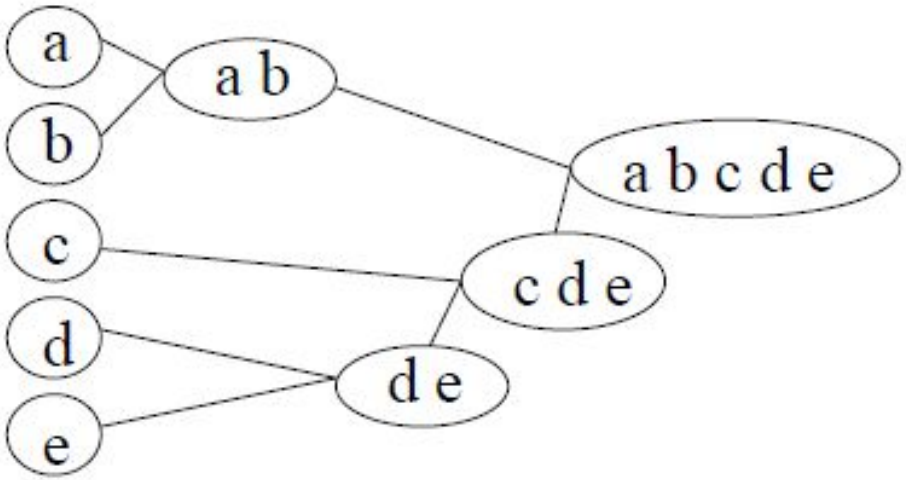
Until all objects are merged

into a single cluster



Hierarchical Clustering

- Divisive Approaches



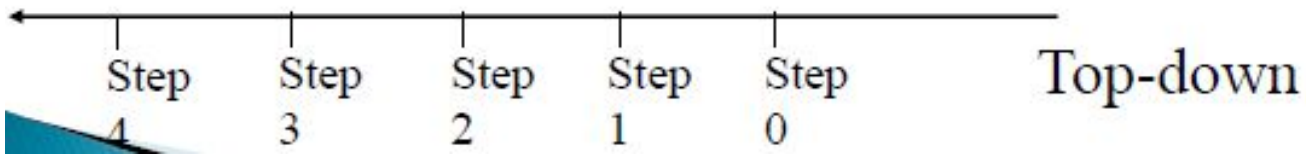
Initialization:

All objects stay in one cluster

Iteration:

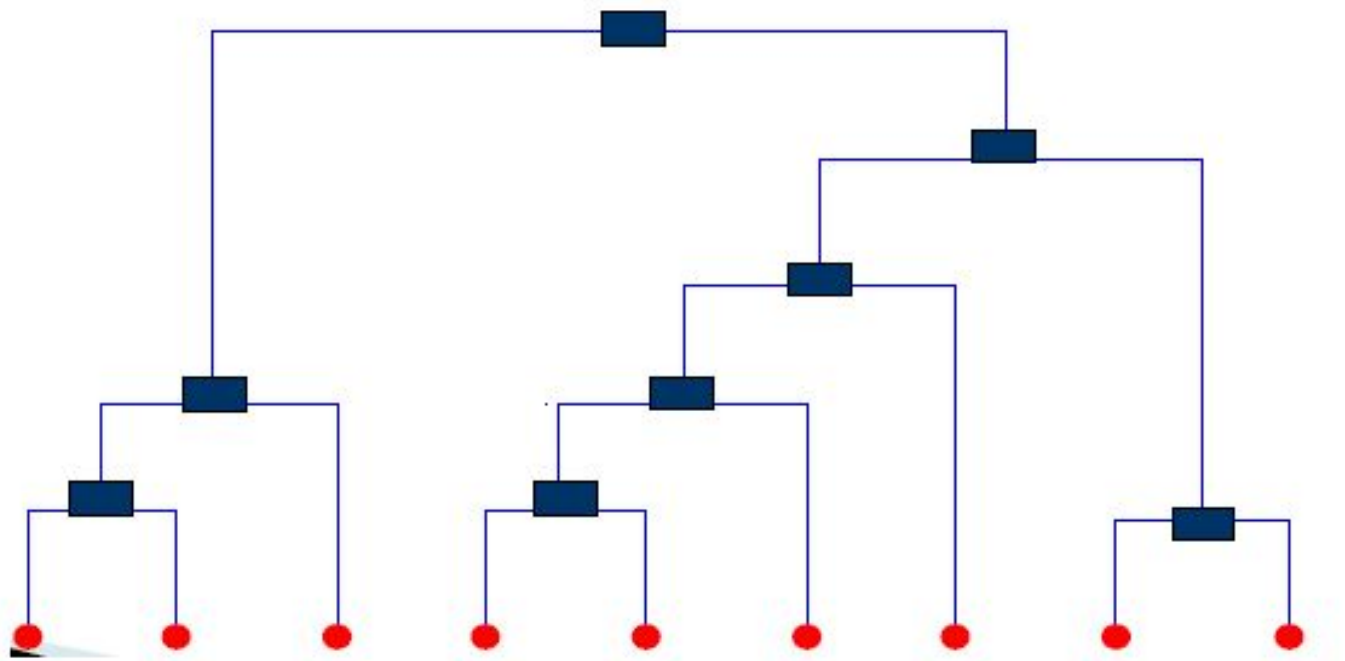
Select a cluster and split it into two sub clusters

Until each leaf cluster contains only one object



Dendrogram

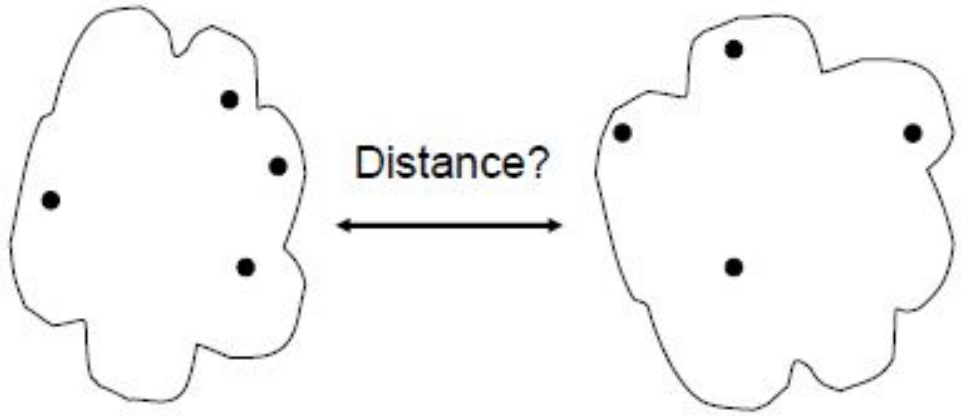
- A binary tree like structure that shows how clusters are merged/split hierarchically
- Each node on the tree is a cluster; each leaf node is a singleton cluster



How to Merge Clusters?

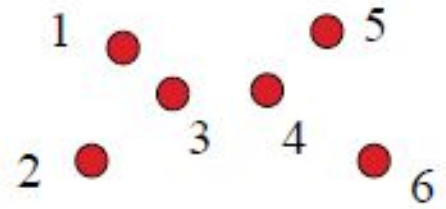
- How to measure the distance between clusters?

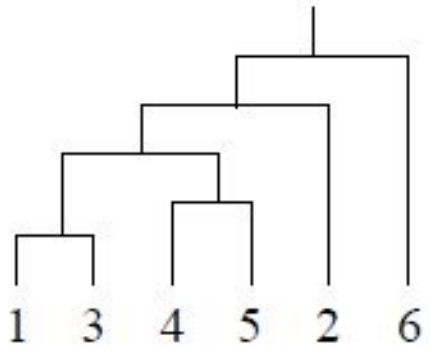
Single-link
Complete-link
Average-link
Centroid distance



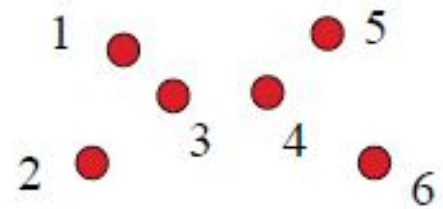
Hint: *Distance between clusters* is usually defined on the basis of *distance between objects.*

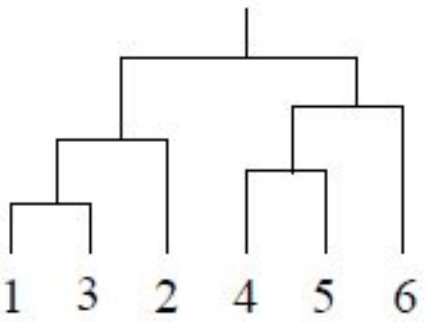
Result of the Single-Link algorithm





Result of the Complete-Link algorithm





Measuring Clustering Quality

- Two methods: extrinsic vs. intrinsic
- Extrinsic: supervised, i.e., the ground truth is available
  - Compare a clustering against the ground truth using certain clustering quality measure
  - Ex. Purity, precision and recall metrics, normalized mutual information
- Intrinsic: unsupervised, i.e., the ground truth is unavailable
  - Evaluate the goodness of a clustering by considering how well the clusters are separated, and how compact the clusters are
  - Ex. Silhouette coefficient

BIRCH : Zhang, Ramakrishnan & Livny, SIGMOD'96

- **BIRCH :** Integration of hierarchical & distance-based clustering
- **BIRCH** stands for **B**alanced **I**terative **R**educing and **C**lustering using **H**ierarchies.

What Birch algorithm tries to solve?

- ✔ Most of the existing algorithms DO NOT consider the case that **datasets** can be **too large** to **fit** in main **memory**
- ✔ They DO NOT concentrate on **minimizing** the number of **scans** of the dataset
- ✔ **I/O costs** are very high

- The complexity of BIRCH is O(n) where n is the number of objects to be clustered.

BIRCH

- Incrementally construct a CF (Clustering Feature) tree, a hierarchical data structure for multiphase clustering

Phase 1: scan DB to build an initial in-memory CF tree (a multi-level compression of the data that tries to preserve the inherent clustering structure of the data)

Phase 2: use an arbitrary clustering algorithm to cluster the leaf nodes of the CF-tree

- *Scales linearly*: finds a good clustering with a single scan and improves the quality with a few additional scans
- *Weakness:* handles only numeric data, and sensitive to the order of the data record

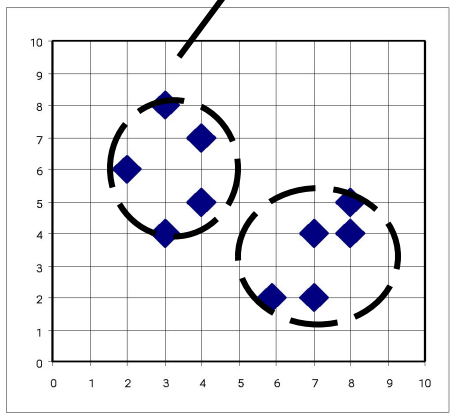
Clustering Feature Vector in BIRCH

Clustering Feature (CF): ***CF = (N, LS, SS)***

N: **Number of data points**

LS: linear sum of N points: $\sum_{i=1}^{N} X_i$

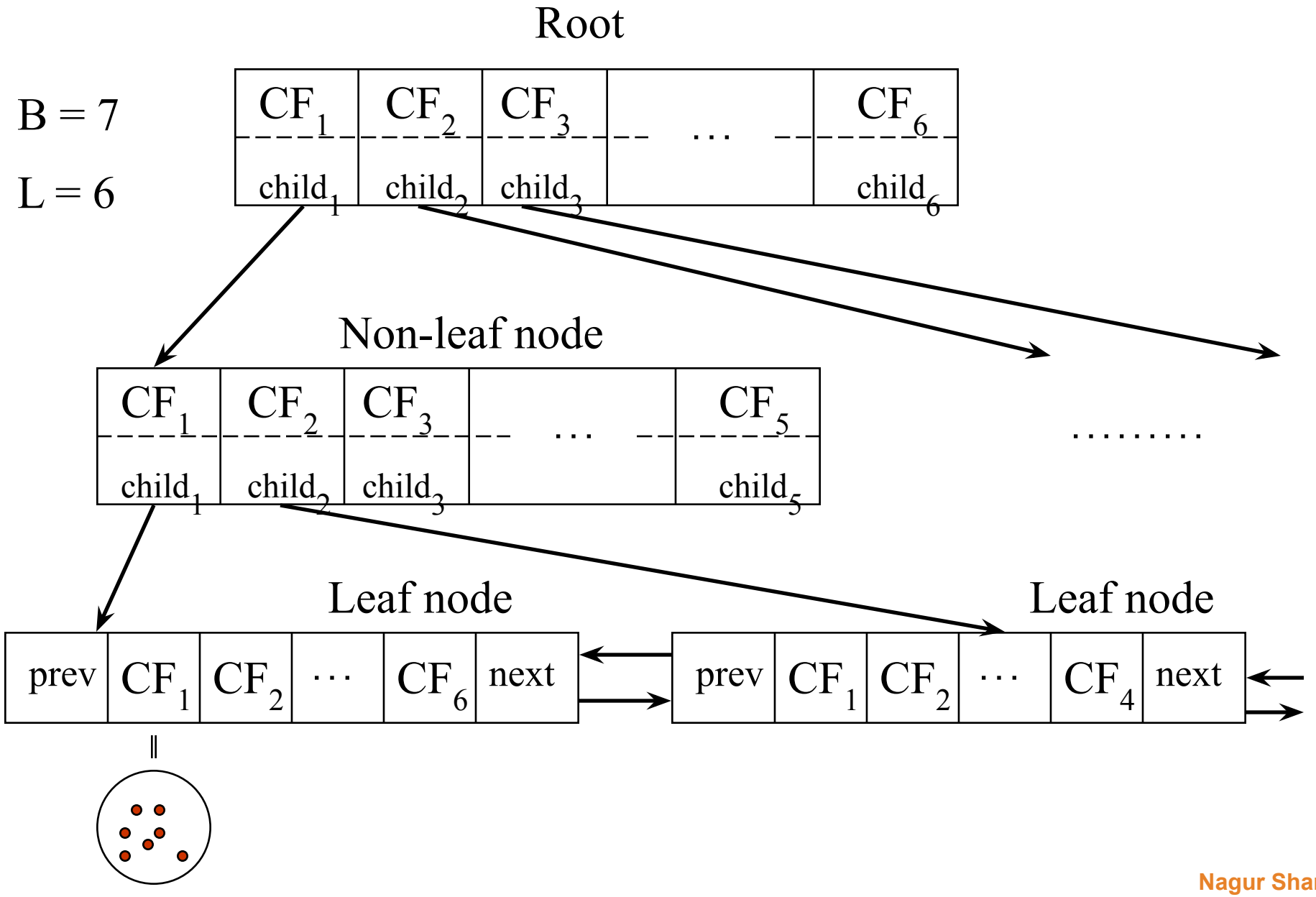
SS: square sum of N points $\sum_{i=1}^{N} X_i^2$



CF-Tree in BIRCH

- Clustering feature:
  - Summary of the statistics for a given subcluster: the 0-th, 1st, and 2nd moments of the subcluster from the statistical point of view
  - Registers crucial measurements for computing cluster and utilizes storage efficiently
- A CF tree is a height-balanced tree that stores the clustering features for a hierarchical clustering
  - A nonleaf node in a tree has descendants or “children”
  - The nonleaf nodes store sums of the CFs of their children
- A CF tree has two parameters
  - Branching factor: max # of children
  - Threshold: max diameter of sub-clusters stored at the leaf nodes

The CF Tree Structure

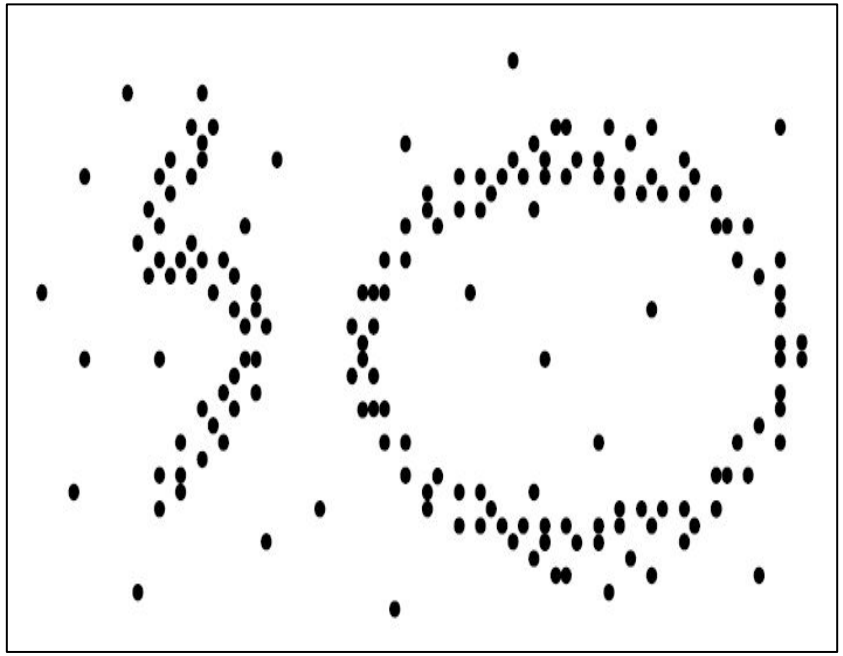


The Birch Algorithm

- Cluster Diameter $$\sqrt{\frac{1}{n(n-1)}\sum(x_i - x_j)^2}$$
- For each point in the input
  - Find closest leaf entry
  - Add point to leaf entry and update CF
  - If entry diameter > max_diameter, then split leaf, and possibly parents
- Algorithm is O(n)
- Concerns
  - Sensitive to insertion order of data points
  - Since we fix the size of leaf nodes, so clusters may not be so natural
  - Clusters tend to be spherical given the radius and diameter measures

Why Density-Based Clustering Methods ?

- Partitioning and hierarchical methods are designed to find spherical-shaped clusters.
- They have difficulty finding clusters of arbitrary shape such as the “S” shape and oval clusters.
- To find clusters of arbitrary shape, alternatively, we can model clusters as dense regions in the data space, separated by sparse regions.
- Basic techniques of density-based clustering by studying three representative methods, namely, **DBSCAN**, OPTICS and DENCLUE.



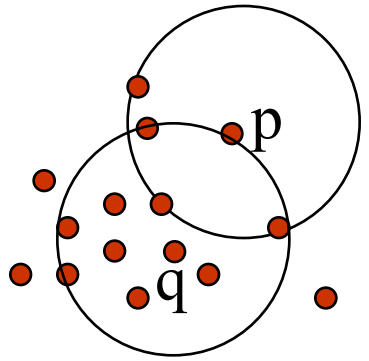
Key Features

- Can find arbitrarily shaped clusters
- Clusters are dense regions of objects in space that are separated by low-density regions
- Cluster density: Each point must have a minimum number of points within its “neighbourhood”
- May filter out outliers

Density-Based Clustering: Basic Concepts

- Two parameters*:*
  - *Eps*: Maximum radius of the neighbourhood
  - *MinPts*: Minimum number of points in an Eps-neighbourhood of that point
- $N_{Eps}(p)$: {q belongs to D | dist(p,q) ≤ Eps}
- Directly density-reachable: A point *p* is directly density-reachable from a point *q* w.r.t. *Eps*, *MinPts* if
  - *p* belongs to $N_{Eps}(q)$
  - core point condition:

$$|N_{Eps}(q)| \geq MinPts$$

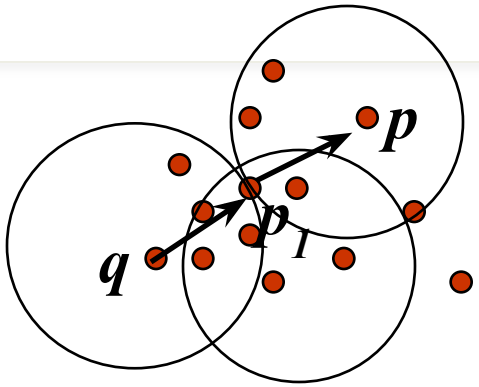


MinPts = 5

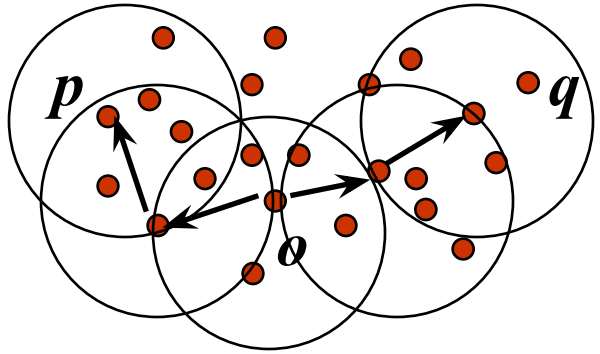
Eps = 1 cm

Density-Reachable and Density-Connected

- Density-reachable:
  - A point *p* is density-reachable from a point *q* w.r.t. *Eps*, *MinPts* if there is a chain of points p_1, …, p_n, $p_1 = q$, $p_n = p$ such that p_{i+1} is directly density-reachable from p_i

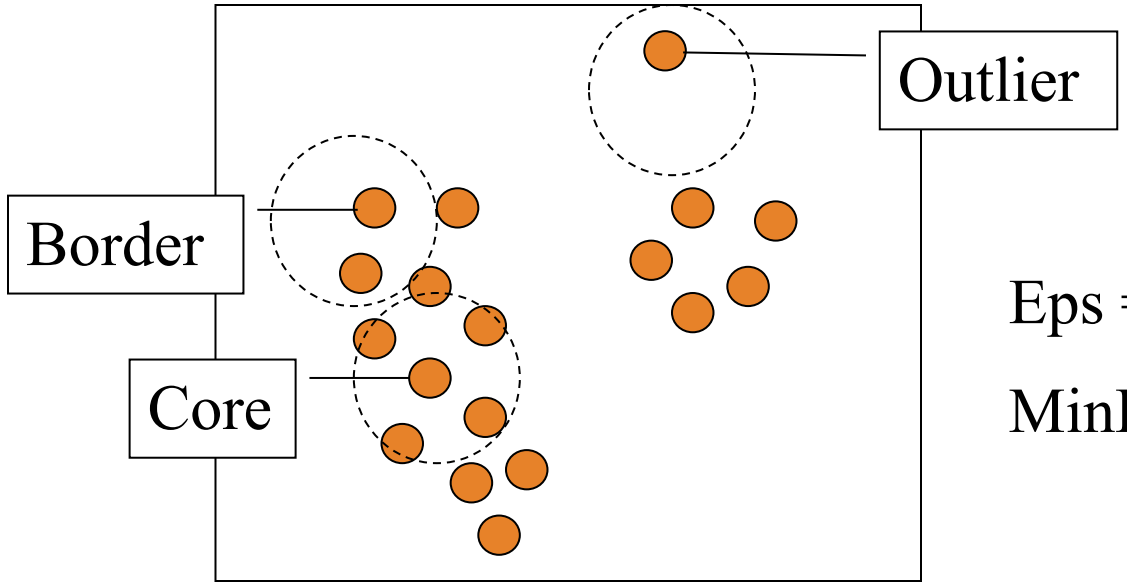


- Density-connected
  - A point *p* is density-connected to a point *q* w.r.t. *Eps*, *MinPts* if there is a point *o* such that both, *p* and *q* are density-reachable from *o* w.r.t. *Eps* and *MinPts*



DBSCAN: Density-Based Spatial Clustering of Applications with Noise

- Relies on a *density-based* notion of cluster: A *cluster* is defined as a maximal set of density-connected points
- Discovers clusters of arbitrary shape in spatial databases with noise



DBSCAN: The Algorithm

- Arbitrary select a point *p*
- Retrieve all points density-reachable from *p* w.r.t. *Eps* and *MinPts*
- If *p* is a core point, a cluster is formed
- If *p* is a border point, no points are density-reachable from *p* and DBSCAN visits the next point of the database
- Continue the process until all of the points have been processed

DBSCAN: Sensitive to Parameters

Figure 8. DBScan results for DS1 with MinPts at 4 and Eps at (a) 0.5 and (b) 0.4.

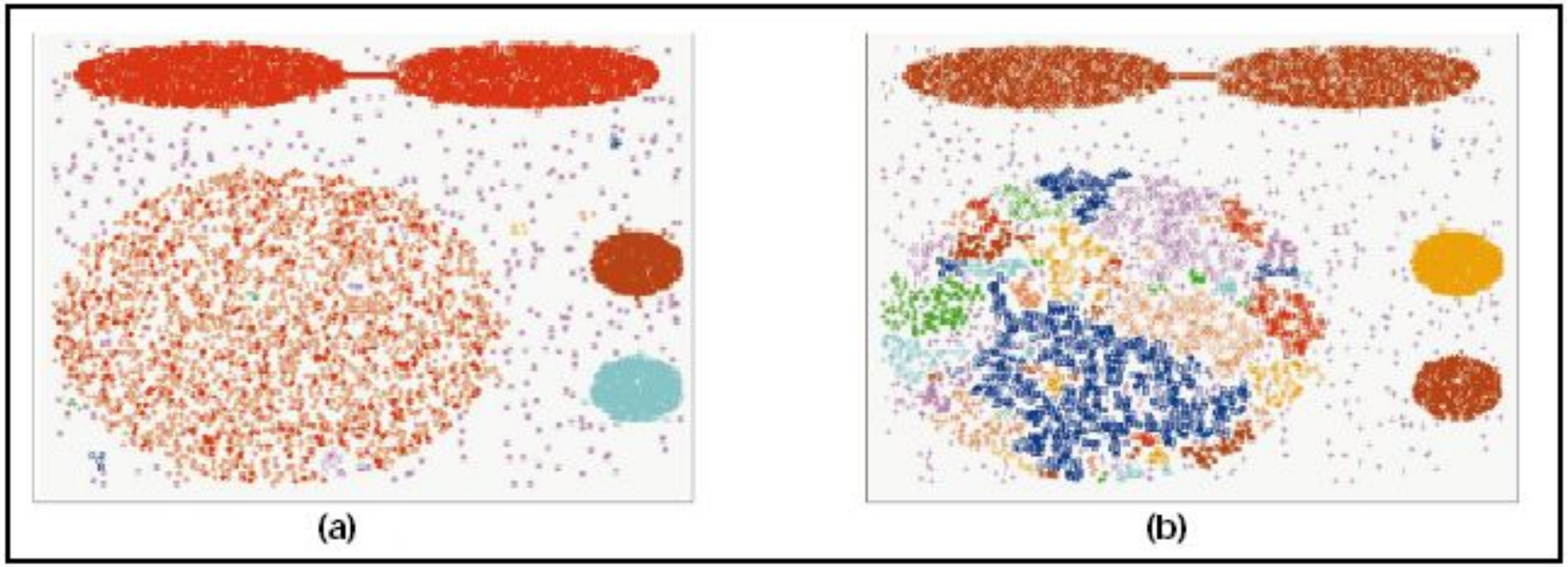
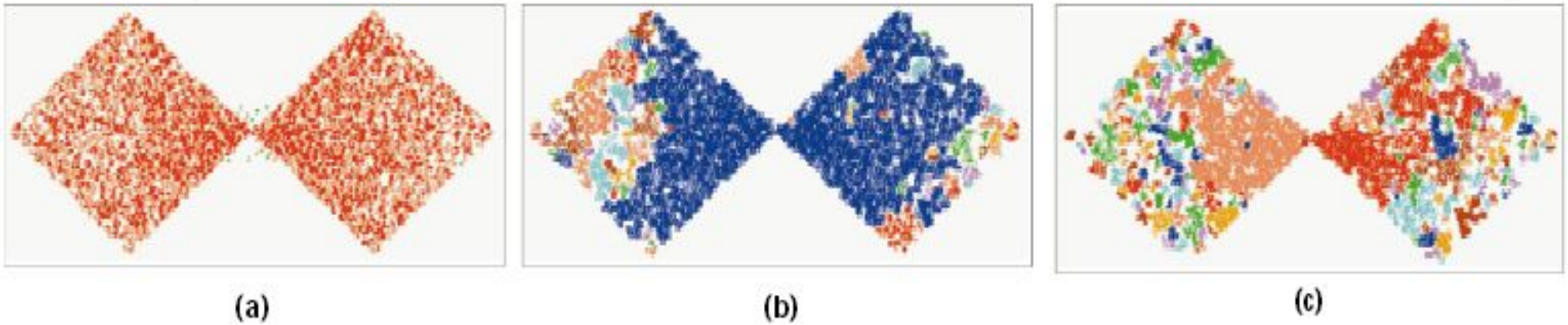


Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.

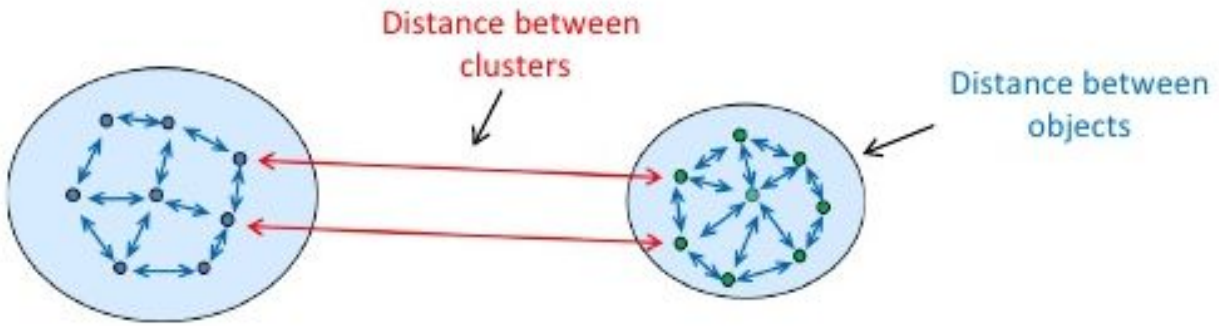


Graph based Clustering

Idea

- Objects are represented as nodes in a complete or connected graph.
- Assign a weight to each branch between the two nodes x and y.
  The weight is defined by the distance d(x,y) between the nodes.

Clustering

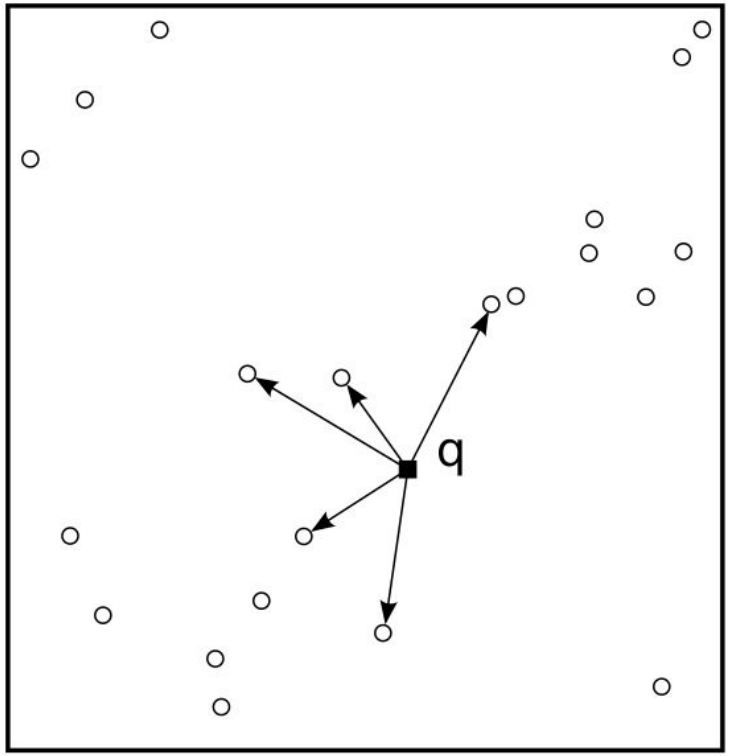


Graph Construction Methods

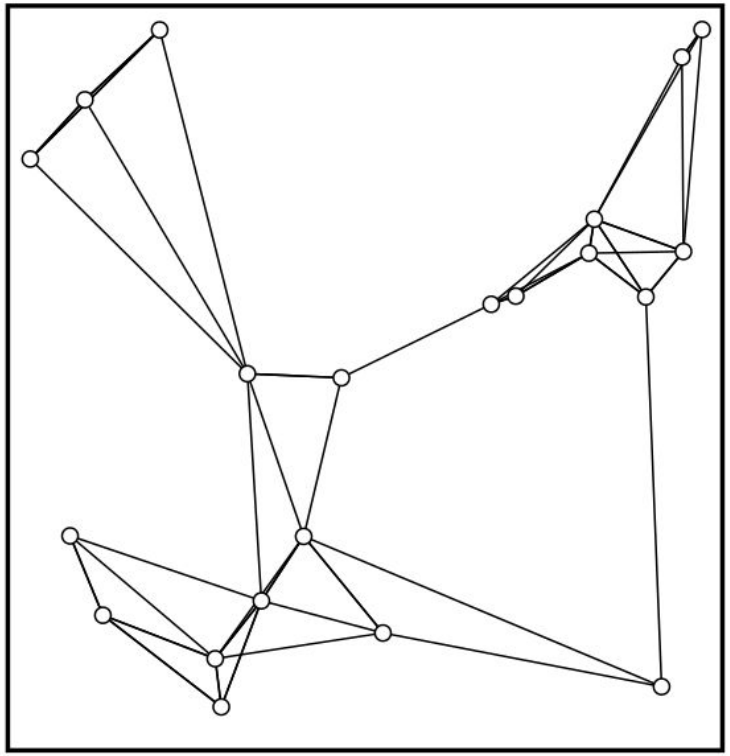
1. K-NN Graph
2. Epsilon neighbourhood Graph

Nearest neighbor problems

k-nearest neighbors, $k = 5$



k nearest neighbors graph ($k = 3$)



Notation: k = number of neighbors, not clusters (k-means).

Graph based Clustering

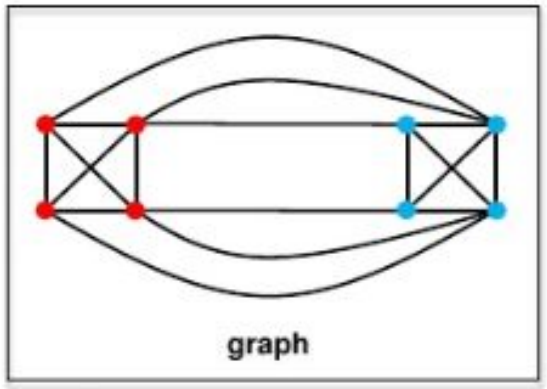
Graph Based Clustering

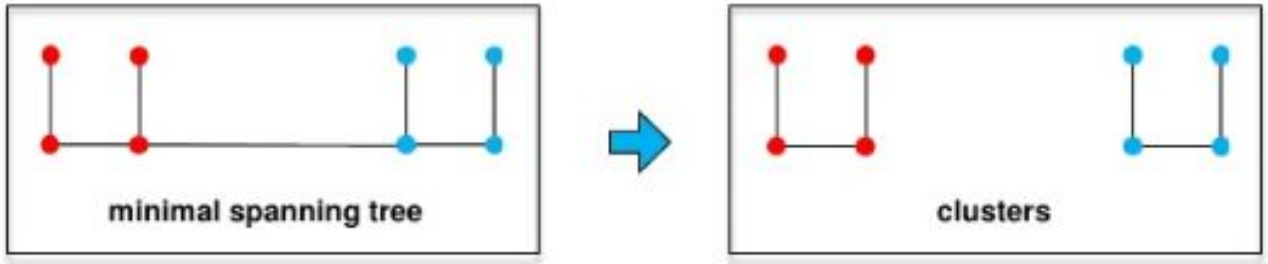
Hierarchical method

(1) Determine a minimal spanning tree (MST)

(2) Delete branches iteratively

New connected components = Cluster





Deleting Branches: Different Strategies

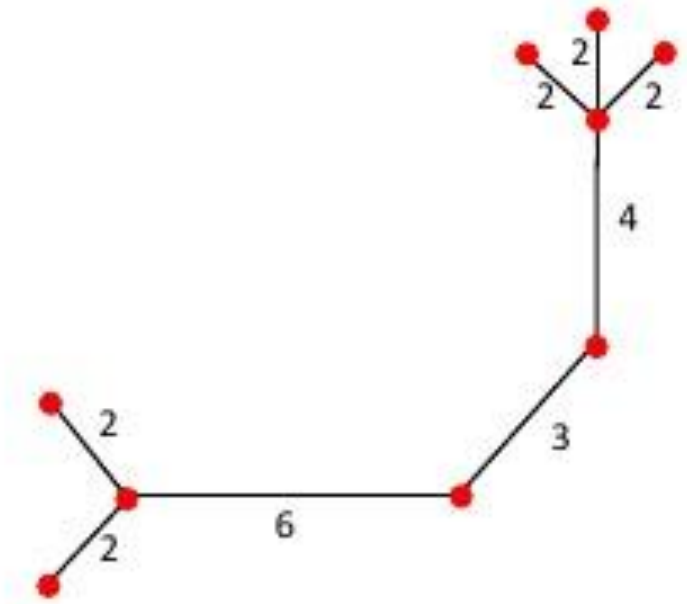
(1) Delete the branch with maximum weight.

(2) Delete inconsistent branches.

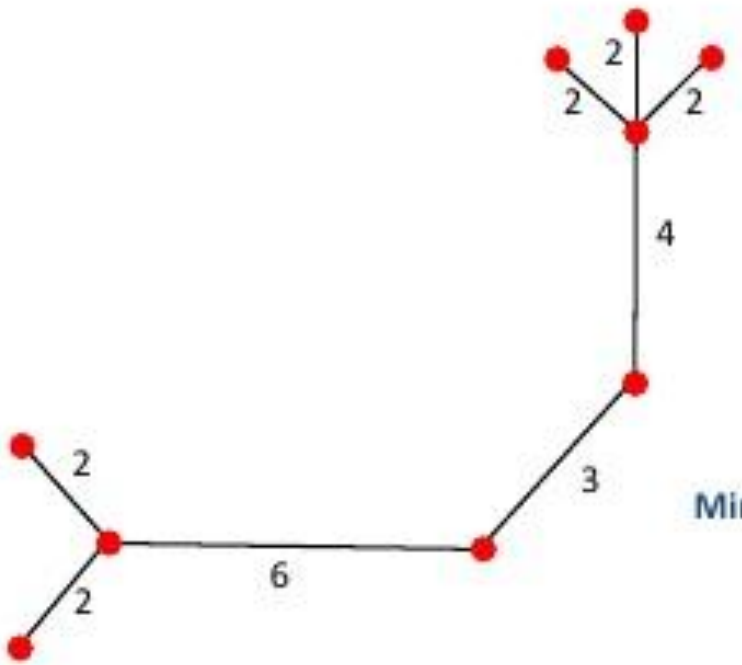
(3) Delete by analysis of weights.

(1) Delete the branch with maximum weight

- In each step, create two new clusters by deleting the branch with maximum weight.
- Repeat until the given number of clusters is reached.



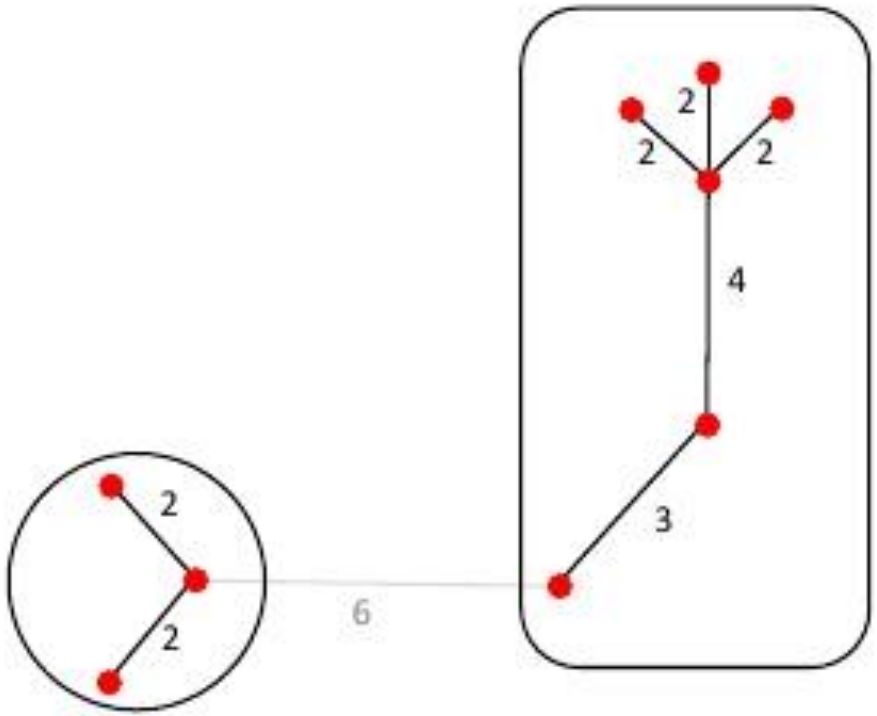
Example: Delete the branch with maximum weight



Minimum spanning tree

Ordered weights of branches: 6, 4, 3, 2, 2, 2, 2, 2.

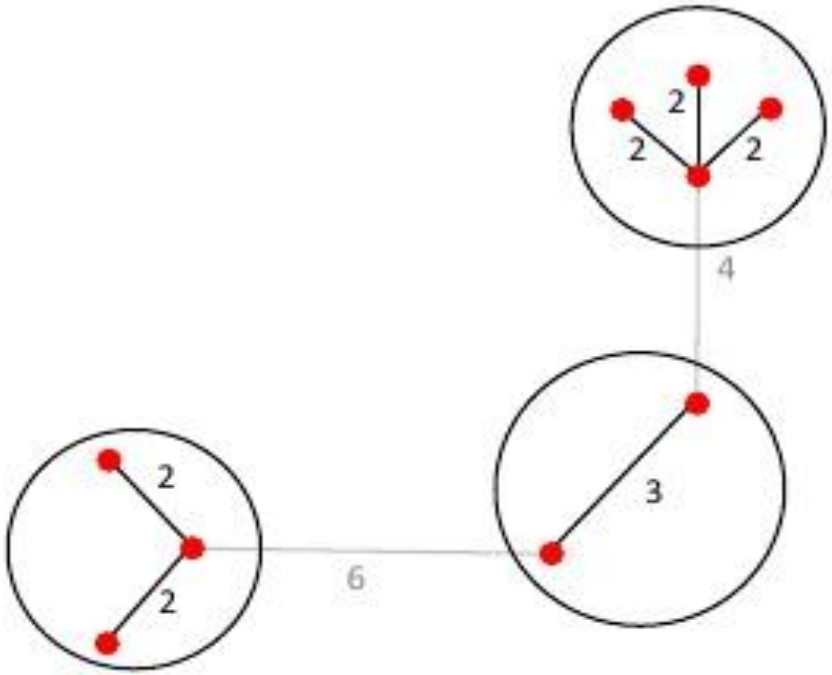
Example: Delete the branch with maximum weight



Ordered weights of branches: ~~6~~, 4, 3, 2, 2, 2, 2, 2.

Step 1: Delete branch (weight 6) $\Rightarrow$ 2 clusters

Example: Delete the branch with maximum weight



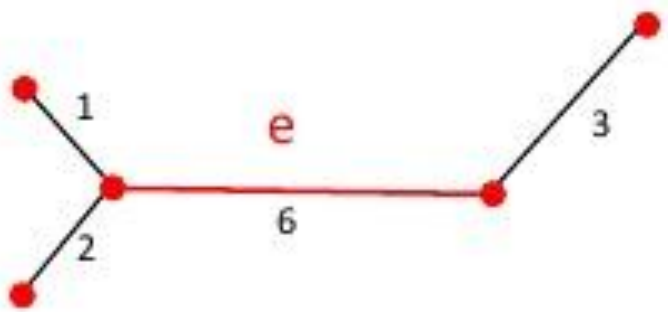
Ordered weights of branches: ~~∅~~, ~~4~~, 3, 2, 2, 2, 2, 2.

Step 1: Delete branch (weight 6) ⇒ 2 clusters

Step 2: Delete branch (weight 4) ⇒ 3 clusters

(2) Delete inconsistent branches

- A branch e is inconsistent, if the corresponding weight d_e is (much) larger than a reference value $\bar{d}_e$.
- The reference value $\bar{d}_e$ can be defined by the average weight of all branches adjacent to e.



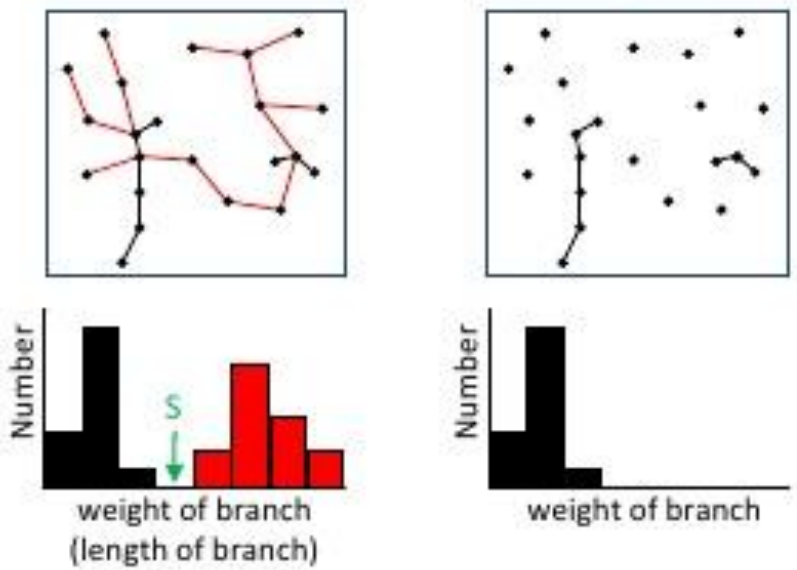
$$\bar{d}_e = \frac{3+2+1}{3} = 2$$

$$d_e = 6 > 2 = \bar{d}_e$$

$$\Rightarrow \text{ e inconsistent}$$

(3) Delete by analysis of weights

- Perform an "analysis" of all weights of branches in the MST.
  Determine a threshold S.
- The threshold can be estimated by
  histograms on the weights of branches (= length of branches).
- Delete a branches, if the corresponding weight higher than the threshold S.



Intuition to Deep Learning

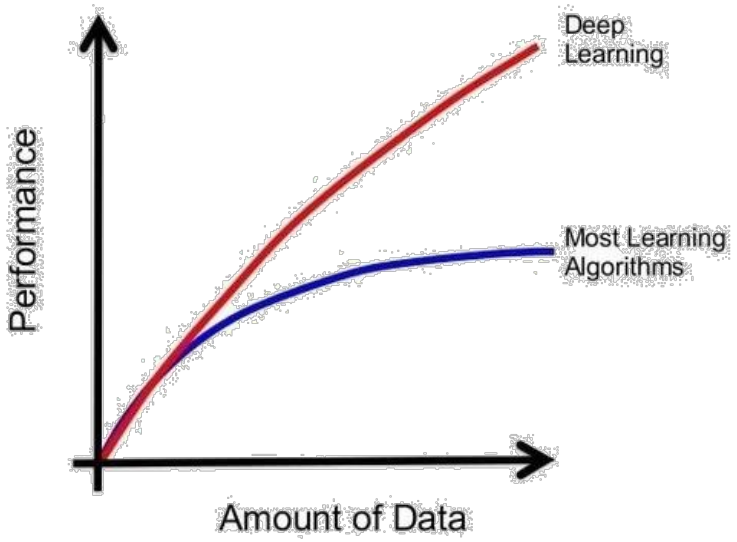
What is Deep Learning…?

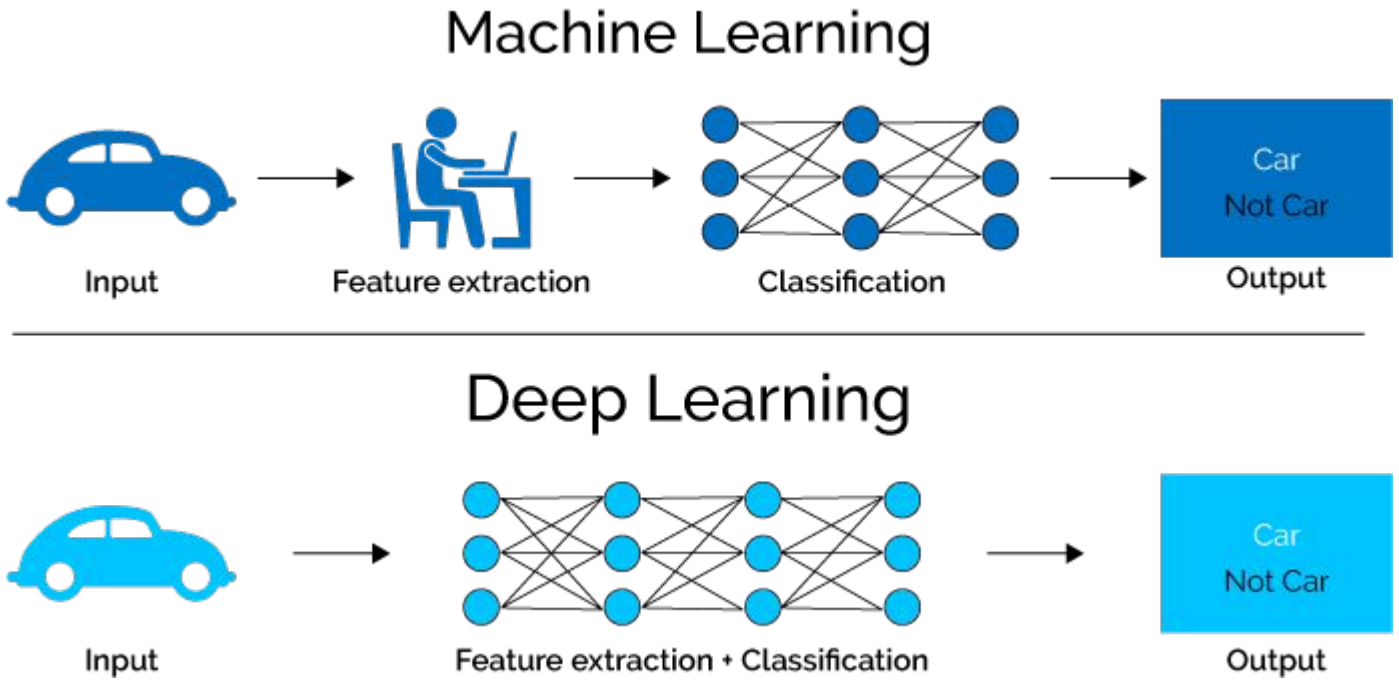
- A Machine Learning subfield of learning representations of data.
- Exceptional effective at learning patterns.
- Deep learning algorithms attempt to learn (multiple levels of) representation by using a hierarchy of multiple layers.
- If you provide the system tons of information, it begins to understand it and respond in useful ways.

Why Deep Learning..?

- Manually designed features are often over-specified, incomplete and take a long time to design and validate
- Learned Features are easy to adapt, fast to learn
- Deep learning provides a very flexible, (almost?) universal, learnable framework for representing world, visual and linguistic information.
- Can learn both unsupervised and supervised
- Effective end-to-end joint system learning
- Utilize large amounts of training data

ML vs DL


Deep
Learning
Most Learning
Algorithms
Performance
Amount of Data


Machine Learning
Input
Feature extraction
Classification
Car
Not Car
Output
Deep Learning
Input
Feature extraction + Classification
Car
Not Car
Output

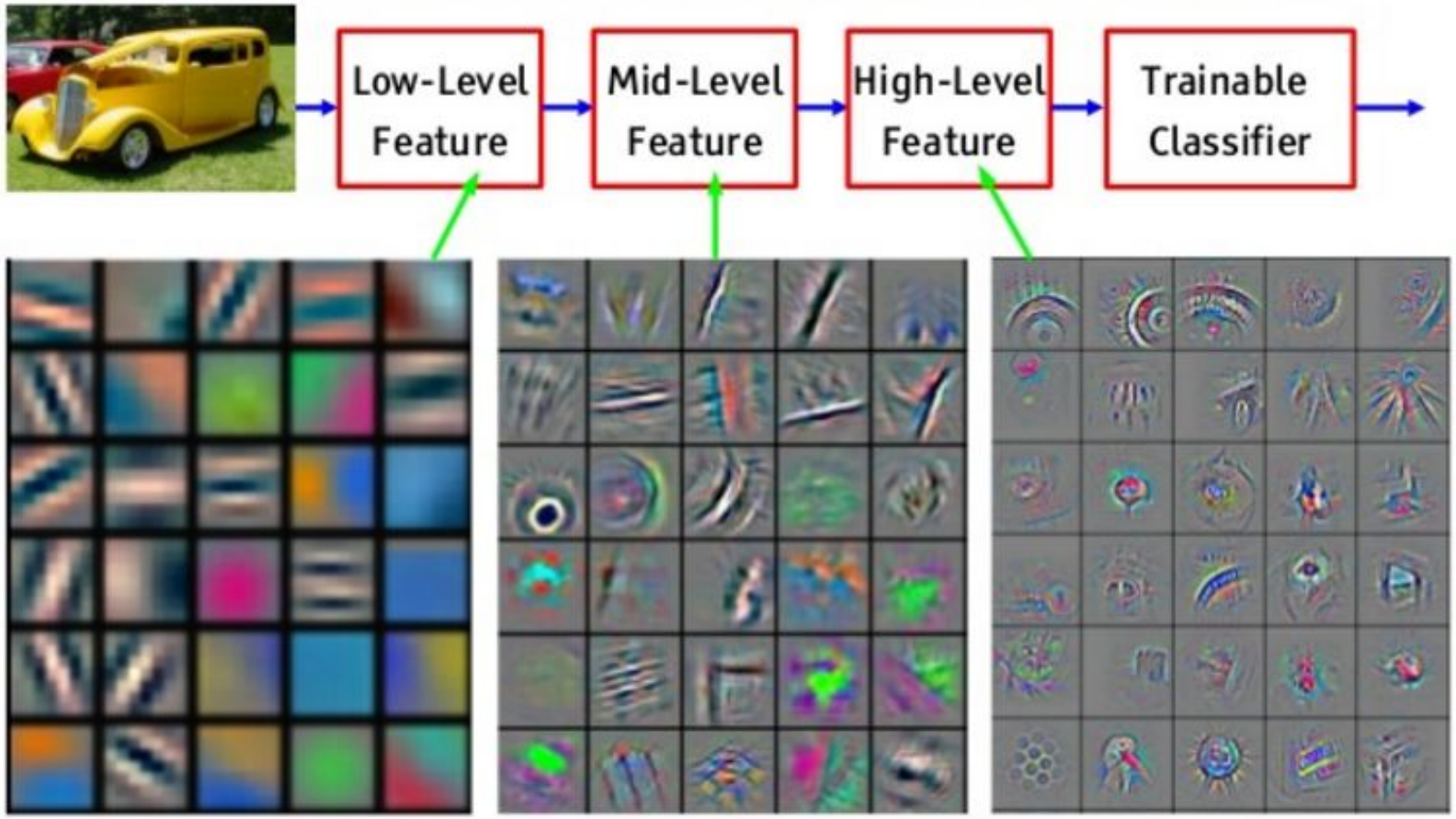
Evolution of Artificial Neural Networks

MP Neuron to Transformers (1943 to 2021)

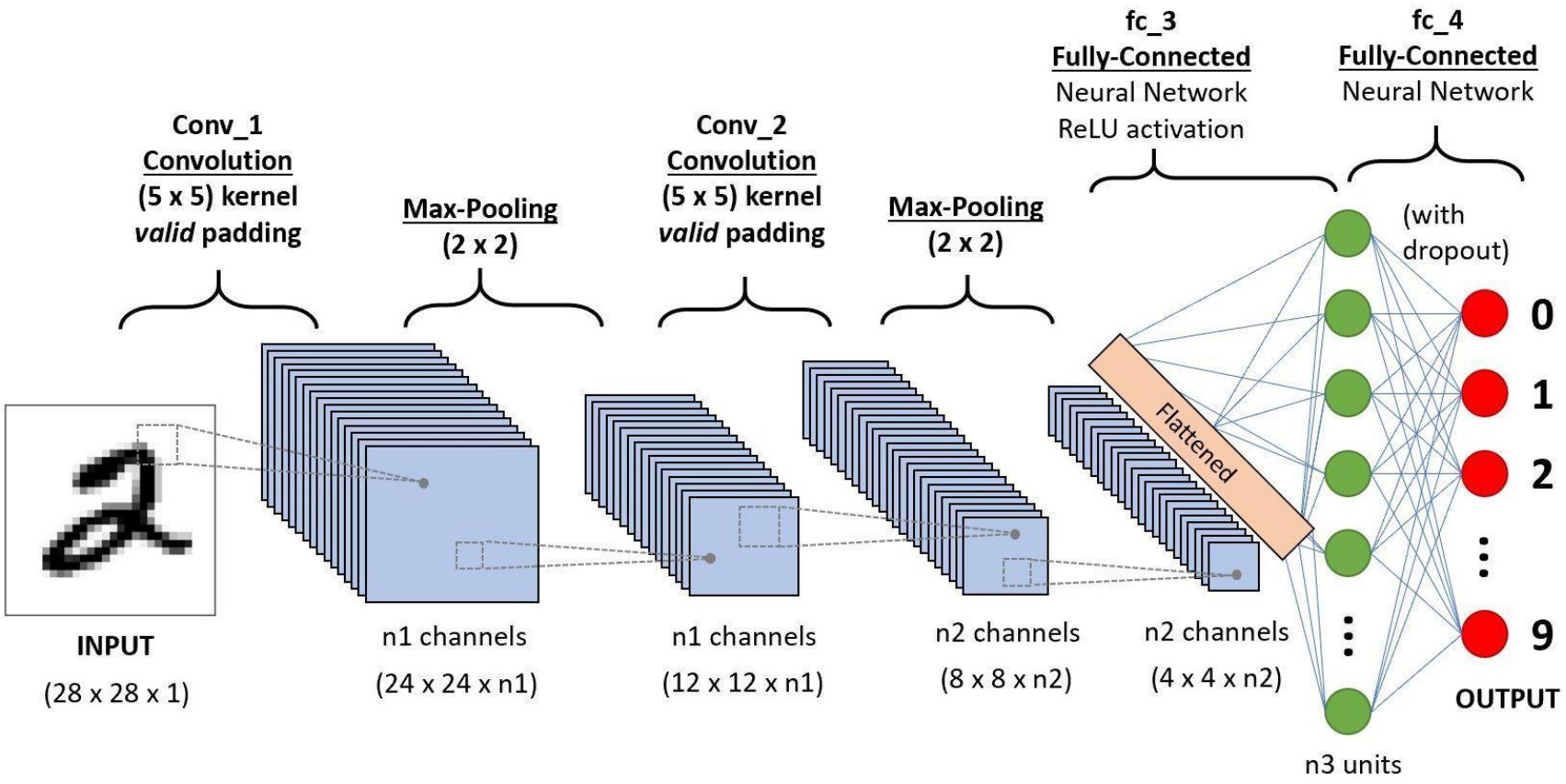
- MP Neuron
- Perceptron
- Multi Layered Perceptron
- Boltzmann Machine
- Restricted Boltzmann Machine
- Deep Belief Networks
- Deep Neural Network
- Convolutional Neural Network
- Recurrent Neural Network
- Long Short Term Memory
- Gated Recurrent Unit
- Auto-encoder
- Generative Adversarial Networks
- Attention Mechanism
- Transformer
- Visual Transformer

Nagur Shareef Shaik (1800420)

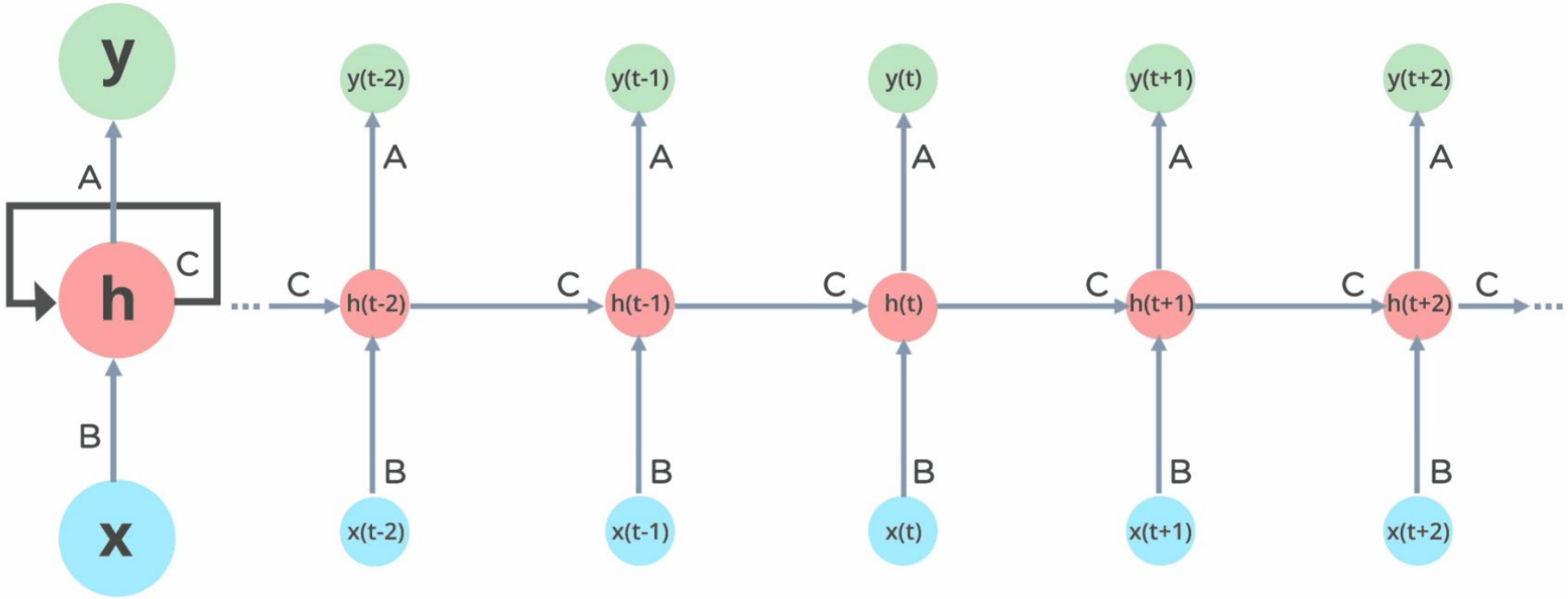
What NNs Learn



Architecture of CNN



RNN: Sequence Modelling



Thank You