Introduction

Data Mining

- Discovering patterns in large data sets
- Extraction of patterns and knowledge from large amounts of data

Machine Learning

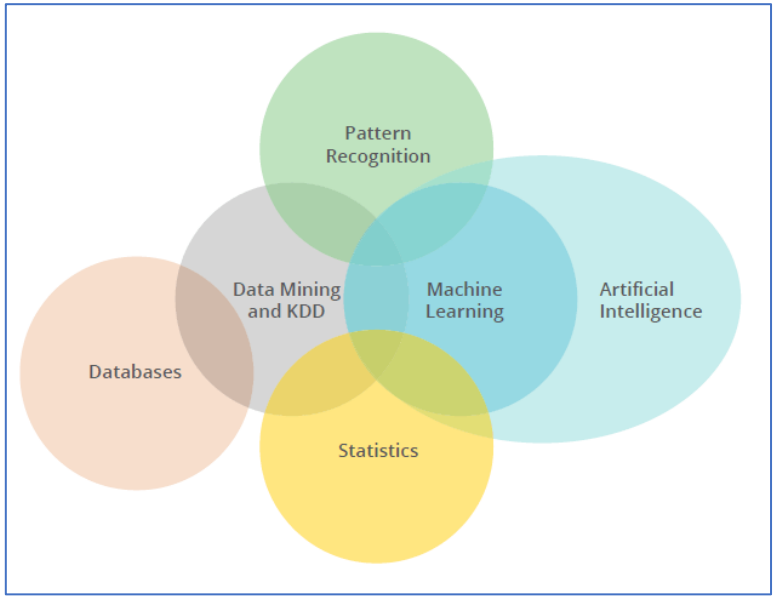
- Study of algorithms and statistical models without using explicit instructions, relying on patterns and inference

Deep Learning

- Uses multiple layers to progressively extract higher level features (attributes).

Reinforcement Learning (Trial and Error)

- Reinforcement Learning is concerned with how agents ought to take actions in an environment to maximize the notion of cumulative reward.



Machine Learning – Types

• Supervised	Labeled	predict class or value	SVM, Regression
• Unsupervised	Unlabeled	determine data patterns / groupings	K-Means, Hierarchical

Clustering

Clustering is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense) to each other than to those in other groups.

Regression

Regression is the problem of predicting and forecasting a concrete number based on a set of data containing observations whose category is known.

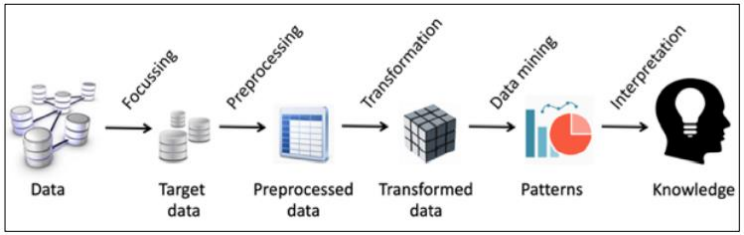
Data Processing

KDD (**K**nowledge **D**iscovery in **D**atabases)

Is the process of semi-automatic extraction of knowledge from databases which is…

- *valid*, *previously unknown,* and *potentially useful*.

Interactive and iterative process with continuous optimization of tasks.



Data Types

Categorial

• Nominal	No order, Scale (“labels”)	*Hair color, Gender, Race*
• Ordinal	Ordered	*Movie rating, Military rank*

Numerical

• Interval	Numeric scale, Ordered	*Time, Weight, Height*
• Ratio	Ordered with same difference	*Time, Weight, Height*

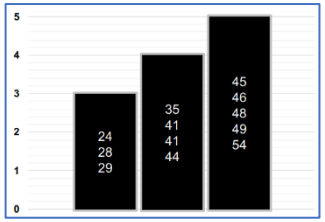
Data Cleaning

Process of improving the data quality by removing or improving incorrect or improperly formatted data.

- (near) duplicates
  - Compare the attributes of the tuple
  - Compare the content of the attributes
- missing values
  - Ignore the tuple
  - Fill in the missing value manually
  - Use a global constant such as -1
  - Use attribute mean
  - Use most probably value
- noisy data
  - binning
  - regression
  - clustering

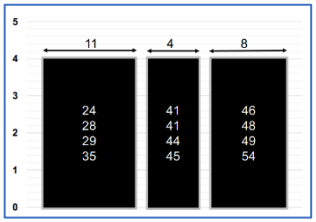
Equal-Width binning

- Intervals of equal size $= width$
- $width = (max - min)/N$



Equal-depth binning

- Intervals with approx. same number of records $= depth$
- $depth \approx Records_{tot}/N$



Data Processing

Linear Regression

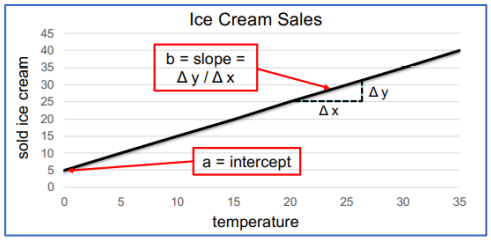
Discover the parameters of the straight-line equation that best fits the data points.

- $\bar{x}$ is the mean value of x
- $\bar{y}$ is the mean value of y

$$b = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}, \qquad a = \bar{y} - b\bar{x}, \qquad y = a + bx$$

Usage

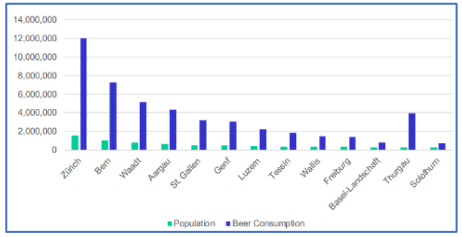
- Smooth out noise
- Fill in missing values
- Predict unknown values

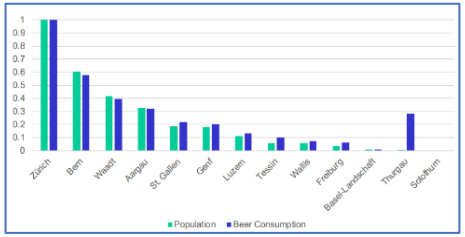


Data Normalization

Change the values of numeric columns to a common scale (e.g., between 0 and 1), without distorting differences in the ranges of values.

- Linear normalization $f_{lin}(v) = \frac{v - min}{max - min}$
- Square root normalization $f_{sq}(v) = \frac{\sqrt{v} - \sqrt{min}}{\sqrt{max} - \sqrt{min}}$
- Logarithmic normalization $f_{ln}(v) = \frac{ln(v) - ln(min)}{ln(max) - ln(min)}$



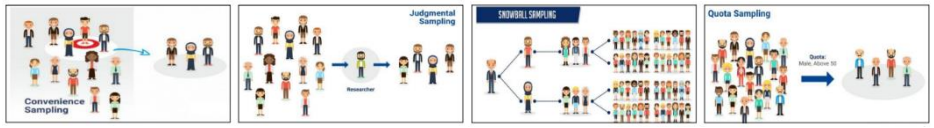


Data Sampling

Represent large dataset by smaller subset to speed up automatic calculations.

Non-Probabilistic

- Convenience — Easiest to obtain
- Judgement — Based on experts' knowledge and judgement
- Snowball — Purely based on referrals
- Quota — Based on attribute values



Probabilistic

- Simple Random Sampling — Just random
- Cluster Sampling — Randomly choose from clusters
- Stratified Random Sampling — Strategy based on attributes
- Systematic Sampling — Order → Every k_{th} element

Data Partitioning

Split data in training, test, and validation sets.

K-Fold cross validation

1. Split data into k folds
2. Train on $k - 1$ folds, evaluate on remaining 1 fold
3. Repeat for each fold
4. Calculate average errors

Evaluation

Standard error measure

$$E = \frac{1}{N}\sum_{i=1}^{N}\left(1 - id(\hat{y}_i, y_i)\right), \qquad id(a,b) = \begin{cases} 1 & if\ a = b \\ 0 & else \end{cases}$$

$$data_{size} = 9, \qquad corret = 6, \qquad wrong = 3$$

$$E = \frac{1}{9} \cdot 3 = 0.33$$

Error Types

- Type-I: False-Positive
- Type-II: False-Negative

$\downarrow y, \quad \hat{y} \rightarrow$	1	0
1	TP=**T**rue **P**ositive	FN=**F**alse **N**egative
0	FP=**F**alse **P**ositive	TN=**T**rue **N**egative

Evaluation measures

- $A =$ Accuracy — Standard measure (doesn't regard "costs" of errors)
- $R =$ Recall — How many relevant documents have been returned
- $P =$ Precision — How many of the returned documents are relevant
- $F =$ F-Measure — Harmonic mean of *recall* and *precision*

$$A = \frac{TP + TN}{TP + TN + FP + FN}, \qquad R = \frac{TP}{TP + FN}, \qquad P = \frac{TP}{TP + FP}$$

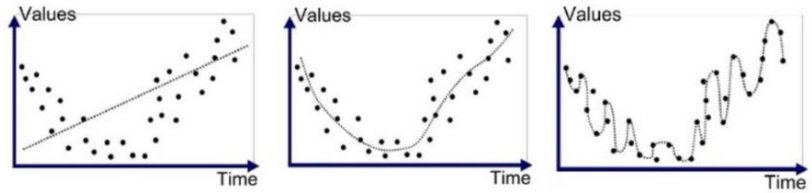
$$F_1 = 2 \cdot \frac{R \cdot P}{R + P}$$

Kappa calculation

- $P(A) =$ proportion of agreements of the raters
- $P(E) =$ agreement, which we could get randomly

$$K = \frac{P(A) - P(E)}{1 - P(E)}$$

Model Evaluation



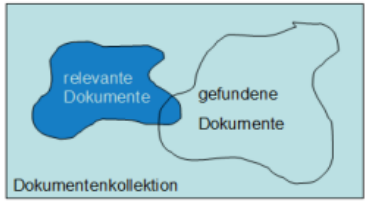
Underfitted | Good Fit/Robust | Overfitted

Model Evaluation – Clustering

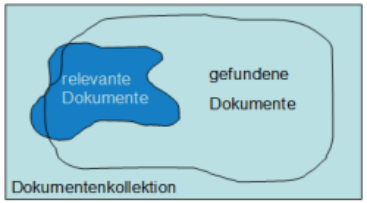
The Silhouette s coefficient indicates the following

1. $s \geq 1$ Sample is far away from the neighboring clusters
2. $s < 1$ Sample is close to the decision boundary
3. $s \approx 0$ Sample is at the decision boundary

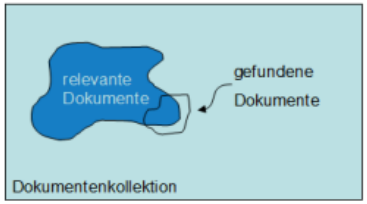
$$Recall = \frac{relevant_{found}}{relevant_{total}}, \qquad Precision = \frac{relevant_{found}}{total_{found}}$$



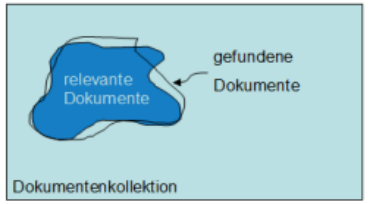
Geringe Ausbeute, geringe Präzision



Hohe Ausbeute, geringe Präzision



Geringe Ausbeute, hohe Präzision



Hohe Ausbeute, hohe Präzision

Recommender Systems

A good recommender system…

- Recommends *personalized and valuable* items within the current context
- Recommends items form all *different* and *possible areas of interest*
- Does **not** recommend items which are *already known*
- *Extends* the *areas of interest* (serendipity effect)

Collaborative Filtering

Analysis of behaviour patterns of user groups. Recommends items to a user based on comparison of their behaviour against users with a "similar" behaviour.

$$similarity = \cos(\theta) = \frac{A \cdot B}{\|A\| \cdot \|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \cdot \sqrt{\sum_{i=1}^{n} B_i^2}}$$

Challenges

- Many items to recommend, but only a few can be recommended
- Mostly sparse data per user
- Interest measure can be very diverse (*Explicit*: ratings, *Implicit*: clicks)
- No data for new users

Content-Based collaborative filtering

Recommendations are based on the content or the attributes of the objects instead of the user behaviour and ratings. There are explicit attributes and the content of an object.

Context-aware filtering

Filtering based on a dynamic collection of factors that describe the status of a user (time, location, social factor).

User-Based collaborative filtering

Calculates the *similarity* between the *users* based on their *rating* behaviour.

- $A = (4 \quad 3 \quad 0 \quad 0 \quad 5)^T$
- $B = (5 \quad 0 \quad 4 \quad 0 \quad 4)^T$
- $C = (4 \quad 0 \quad 5 \quad 3 \quad 4)^T$

$$sim_{AB} = \frac{(4 \cdot 5) + (3 \cdot 0) + \cdots + (5 \cdot 4)}{\sqrt{16 + 9 + 25} \cdot \sqrt{25 + 16 + 16}} = 0.75$$

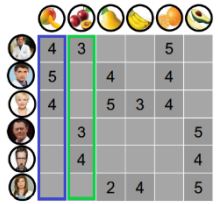
Recommend *items with a high rating* from these *similar users*.

$$\frac{0.75 \cdot 4 + 0.63 \cdot 5}{0.75 + 0.63}$$

Item-Based collaborative filtering

Calculates the *similarity* between the *items* based on their *rating*.

- $A = (4 \quad 5 \quad 4 \quad 0 \quad 0)^T$
- $B = (3 \quad 0 \quad 0 \quad 3 \quad 4)^T$



$$sim_{AB} = \frac{(4 \cdot 3)}{\sqrt{16 + 25 + 16} \cdot \sqrt{9 + 9 + 16}} = 0.27$$

Recommends the most similar items with the *highest ranking*.

Association Rules

The goal of **association rules** is to find correlations between multiple items.

Measures

- ***support count*** (σ) frequency of itemsets
- ***support*** (s) fraction of transactions that contain an itemset
- ***confidence*** c how often items in Y appear in transactions that contain X

$$c = \frac{\sigma(X \cup Y)}{\sigma(X)}$$

Lift measures for how likely item Y is purchased when item X is purchased, while controlling for how popular item Y is.

$$lift(x \rightarrow y) = \frac{support(x, y)}{support(x) \cdot support(y)}$$

- $lift \approx 1$ implies no association
- $lift < 1$ implies negative association
- $lift > 1$ implies positive association

Example

$$S_1 = \{bread, milk\}, \qquad S_2 = \{coke, chips\}$$

T#	items
1	bread, peanuts, milk, apple, honey
2	bread, honey, coke, chips, milk
3	steak, honey, coke, chips, bread
4	honey, coke, peanuts, milk, apple
5	honey, coke, chips, milk, bread
6	apple, coke, chips, milk
7	apple, coke, peanuts, milk, bread
8	apple, peanuts, cheese, milk, bread

support count

- $\sigma(S_1) = 5$
- $\sigma(S_2) = 4$

support

- $s(S_1) = 5/9$
- $s(S_2) = 4/9$

confidence

$$\{bread, milk\} \rightarrow \{honey\}$$

$$\frac{\sigma(\{bread, milk, honey\})}{\sigma(\{bread, milk\})} = \frac{3}{5} = 0.6$$

A **frequent itemset** is an itemset whose support is greater than or equal to a minimum support threshold.

An **association rule** is an implication expression of the from $X \rightarrow Y$ where X and Y are frequent itemsets.

$$\{milk, bread\} \rightarrow \{honey\}$$

Given a set of **transactions** T, the goal of association rule mining is to find all rules having:

$$s \geq \min_s threshold, \qquad c \geq \min_c threshold$$

One-Dimensional (Example)

$$buys(X, \text{IPhone}) \rightarrow buys(X, \text{charging cable})$$

$$s = 18, \qquad c = 63$$

- 18% of all transactions showed that an *IPhone* and a *charging cable* were bought together
- 63% of customers that bought an *IPhone* also bought a *charging cable*

Multi-Dimensional (Example)

$$age(X, [20, \ldots, 29]) \wedge income(X, [40k, \ldots, 49k]) \rightarrow buys(X, \text{IPhone})$$

$$s = 2, \qquad c = 60$$

Association Rules (Frequent Pattern Mining)

The possible number of frequent itemsets is huge. If an itemset is frequent, each of its subsets is frequent as well. For example, a frequent itemset of length 100 contains = $\binom{100}{1} = 100$ frequent 1-itemsets, $\binom{100}{2} = 100$ frequent 2-itemsets. Meaning that there $2^{100} - 1$ are possible frequent itemsets.

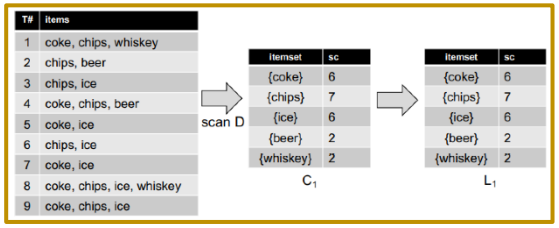
1. Reduce the number of candidates M by using pruning techniques
2. Reduce the number of transactions N by reducing the size of N as the size of itemset increases
3. Reduce the number of comparisons (NM) by using efficient data structures to store the candidates or transactions

Apriori-Algorithm to generate frequent patterns

Example with $\sigma_{min} = 2$

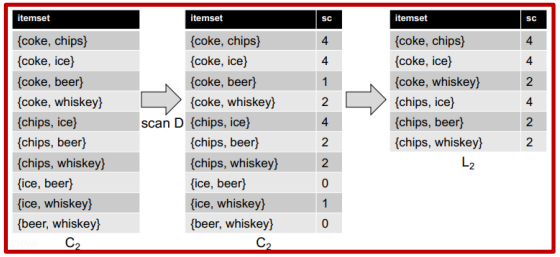
Step 1: Generate 1-itemset frequent pattern

- C_1 = All 1-itemsets
- L_1 = All 1-itemsets with $s \geq s_{min}$



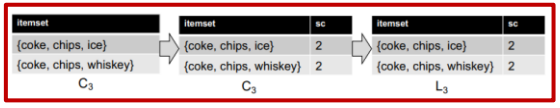
Step 2: Generate 2-itemset frequent pattern

- C_2 = All 2-itemsets ($L_1 \bowtie L_1$)
- L_2 = All 2-itemsets C_2 with $s \geq s_{min}$



Step 3: Generate 3-itemset frequent pattern

- C_3 = All 3-itemsets ($L_2 \bowtie L_2$)
- L_3 = All 3-itemsets with $s \geq s_{min}$



Step 4: Generate 4-itemset frequent pattern

- C_4 = All 4-itemsets ($L_3 \bowtie L_3$) $\rightarrow \{coke, chips, ice, whiskey\}$ $(s = 1)$
- $L_4 = \{\}$

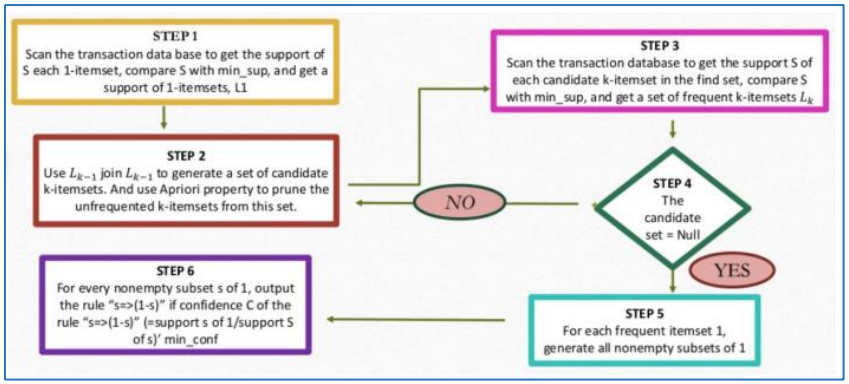
Step 5 + 6: Generate association rules from frequent itemsets

For every itemset in $L_n = (L_1, L_2, \dots)$

1. Generate rules
2. Check if $\frac{\sigma(I)}{\sigma(S)} = c \geq \min_c$ $\qquad min_c = 0.7 = 70\%$

Example: $I = \{coke, chips, whiskey\}$

- Rule1: $S = \{coke, chips\} \rightarrow \{whiskey\}$
  - $c = \frac{\sigma(I)}{\sigma(S)} = \frac{2}{4} \rightarrow 0.5 = 50\% \rightarrow rejected$
- Rule2: $S = \{coke, whiskey\} \rightarrow \{chips\}$
  - $c = \frac{\sigma(I)}{\sigma(S)} = \frac{2}{2} \rightarrow 1 = 100\% \rightarrow accepted$



FP-Tree

ID	Transaction Items					List
T1	coke	chips			ice	{A, B, E}
T2		chips		beer		{B, D}
T3		chips	whiskey			{B, C}
T4	coke	chips		beer		{A, B, D}
T5	coke		whiskey			{A, C}
T6		chips	whiskey			{B, C}
T7	coke		whiskey			{A, C}
T8	coke	chips	whiskey		ice	{A, B, C, E}
T9	coke	chips	whiskey			{A, B, C}

Step 1: Calculate min support count min_σ

$$\min_s = 22\% \rightarrow min_\sigma = 0.22 \cdot 9 = 1.98 \rightarrow 2$$

Step 2: Find frequency of occurrence σ of each item

- A: Coke $\sigma = 6$
- B: Chips $\sigma = 7$
- C: Whiskey $\sigma = 6$
- D: Beer $\sigma = 2$
- E: Ice $\sigma = 2$

Item	Freq	Prio
coke	6	2
chips	7	1
whiskey	6	3
beer	2	4
ice	2	5

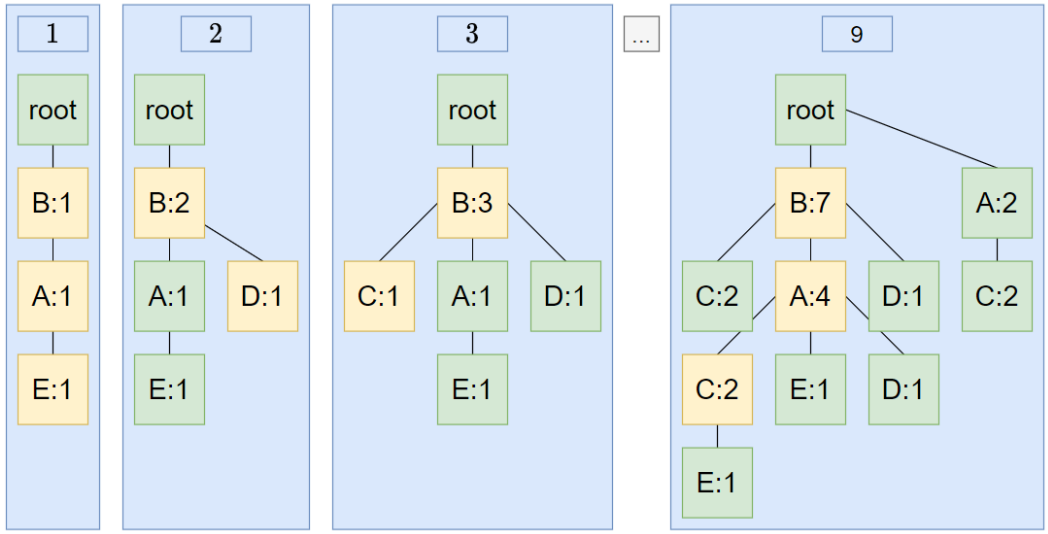
Step 3: Order the items according to priority

- $\{A, B, E\} \rightarrow \{B, A, E\}$
- ...

ID	Transaction Items					Ordered List
T1	coke	chips			ice	{B, A, E}
T2		chips		beer		{B, D}
T3		chips	whiskey			{B, C}
T4	coke	chips		beer		{B, A, D}
T5	coke		whiskey			{A, C}
T6		chips	whiskey			{B, C}
T7	coke		whiskey			{A, C}
T8	coke	chips	whiskey		ice	{B, A, C, E}
T9	coke	chips	whiskey			{B, A, C}

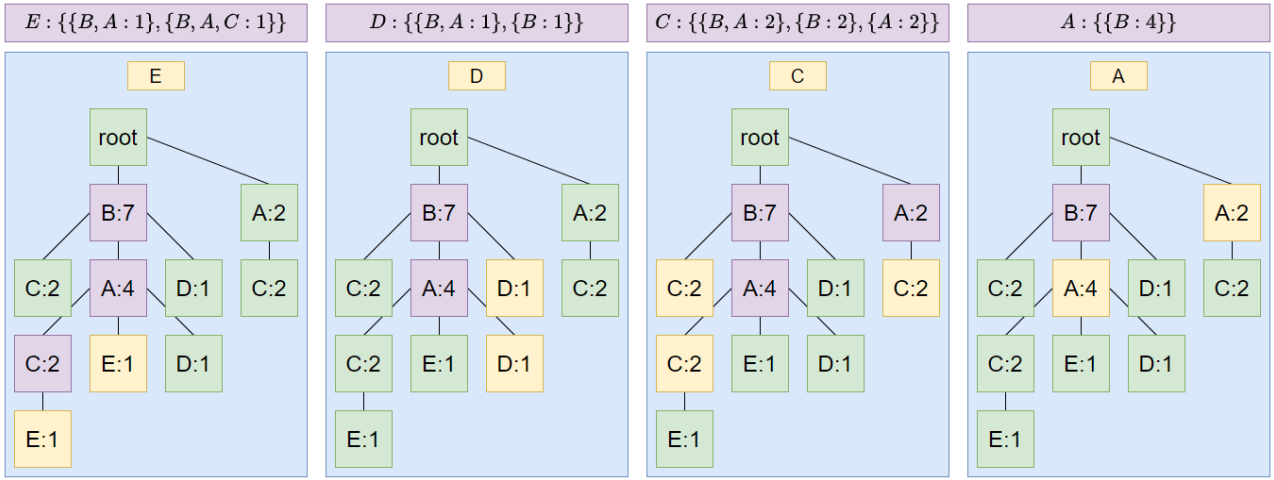
Step 4: Build the FP-Tree

1	$\{B, A, E\}$
2	$\{B, D\}$
3	$\{B, C\}$
4	$\{B, A, D\}$
5	$\{A, C\}$
6	$\{B, C\}$
7	$\{A, C\}$
8	$\{B, A, C, E\}$
9	$\{B, A, C\}$



Step 5: Find *conditional pattern base* and *conditional FP-Tree* for each item

$E: \{B, A: 2\}, \quad D: \{B: 2\}, \quad C: \{B: 4, A: 2\}, \{A: 2\}, \quad A: \{B: 4\}$



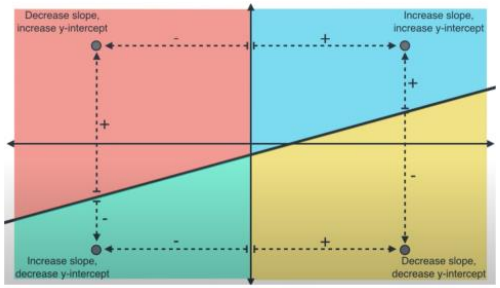
Linear Regression ( )

$$b = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

Gradient Descent Method

Adjust the slope with rotation and the y-intercept with translation.

1. Start with a random line (e.g. $y = 2x + 3$)
2. Pick a large number of repetitions R
3. Pick a small number for the learning rate LR
4. Pick a random point (repeat R times)
   - Slope Add $LR \cdot (x - x_{new}) \cdot (y - y_{new})$
   - Intercept Add $LR \cdot (y - y_{new})$



Mean Square Error (MSE)

1. Find the linear regression line
2. Insert your X values into the linear regression equation to find new Y values Y'
3. Subtract the new Y' value from the original Y to get the error
4. Square the errors, add up the errors and calculate the mean

The following regression lines are given

$$y_1 = 9.2 + 0.8x, \qquad y_2 = 9.1 + 0.75x$$

Calculate the MSE for the given X and Y values

$$MSE_1 = 6.08 = \frac{6.67 + 0.36 + 14.44 + 1 + 7.84}{5}$$

$$MSE_2 = 11.61 = \frac{0.12 + 8.41 + 37.8 + 11.56 + 0.12}{5}$$

X	Y	Y1'	e1	$e1^2$	Y2'	e2	$e2^2$
43	41	43.6	-2.6	6.76	41.35	0.35	0.12
44	45	44.4	0.6	0.36	42.1	-2.9	8.41
45	49	45.2	3.8	14.44	42.85	-6.15	37.8
46	47	46	1	1	43.6	-3.4	11.56
47	44	46.8	-2.8	7.84	44.35	0.35	0.12

Regressions

Multivariate Linear Regression

Linear regression also works on data sets with multiple features.

Example: Find regression equation $\hat{y} = b_0$ … to predict test score, based on IQ and the number of study hours.

person	score	IQ	study hours
1	100	110	40
2	90	120	30
3	80	100	20
4	70	90	0
5	60	80	10

- b_n = Regression coefficients
- x_n = Features
- $\hat{y} = b_0 + b_1 x_1 + b_2 x_2$

To find the regression coefficients b we need solve the following equation

$$\vec{b} = (X'X)^{-1}X'Y$$

$$X' = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 110 & 120 & 100 & 90 & 80 \\ 40 & 30 & 20 & 0 & 10 \end{bmatrix} * Y = \begin{bmatrix} 100 \\ 90 \\ 80 \\ 70 \\ 60 \end{bmatrix} = \begin{bmatrix} 400 \\ 40900 \\ 8900 \end{bmatrix}$$

$$(X'X)^{-1} = \begin{bmatrix} 101/5 & -7/30 & 1/6 \\ -7/30 & 1/360 & -1/450 \\ 1/6 & -1/450 & 1/360 \end{bmatrix} * \begin{bmatrix} 400 \\ 40900 \\ 8900 \end{bmatrix} = \begin{bmatrix} 20 \\ 0.5 \\ 0.5 \end{bmatrix}$$

$$b = \begin{bmatrix} b_0 \\ b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} 20 \\ 0.5 \\ 0.5 \end{bmatrix} \Rightarrow \hat{y} = 20 + 0.5x_1 + 0.5x_2$$

Logistic Regression

Predicts if something is true or false. It Provides probabilities p and classifies new samples using continuous and discrete measurements.

$$p = \frac{1}{1 + e^{logit}}$$

Example

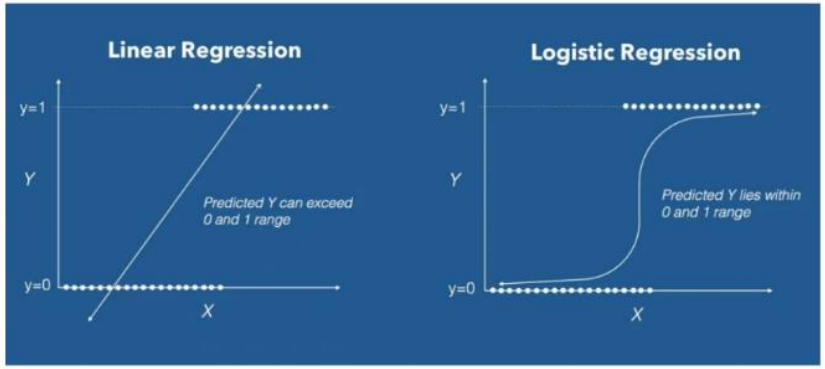
$$p = \frac{1}{1 + e^{-(b_0 + b_1 x)}}$$

Probability p of passing exam

$$\frac{1}{1 + e^{-(-4.0777 + 1.5048 \cdot Hours)}}$$

- Studying 2 hours → $p = 0.26$
- Studying 4 hours → $p = 0.87$

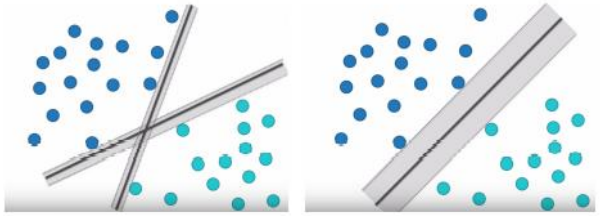
	Coefficient	Std.Error	z-value	P-value (Wald)
Intercept	−4.0777	1.7610	−2.316	0.0206
Hours	1.5046	0.6287	2.393	0.0167



Support Vector Machines

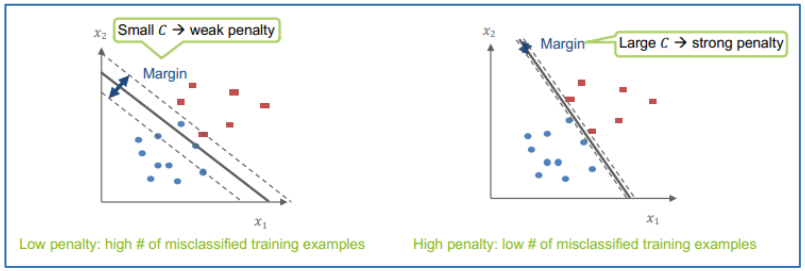
Find a *hyperplane* that separates the two classes of input points with maximum margin.

- Points on the *decision boundary* are called *support vectors*
- Only some points influence the decision boundary



Idea of the C penalty factor on misclassifications

- soft margins allow for some misclassification
- data may contain untypical or mislabeled samples
- the number of misclassifications is controlled by a penalty factor C



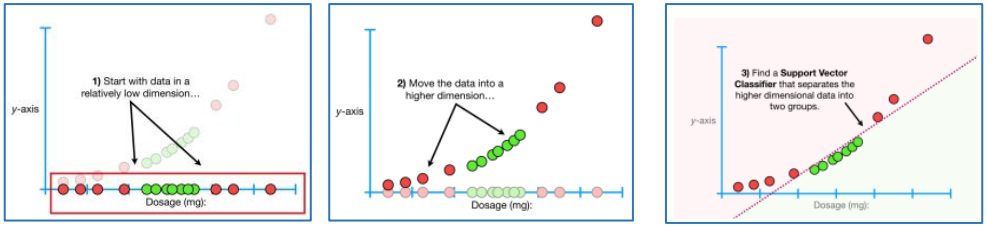
Family of classifiers

- (max. margin) hyperplane classifier for linearly separable data
- Support vector classifier for almost linearly separable data
- Support vector machine for non-linearly separable data

Maximal margin classifiers and *support vector classifiers* can handle outliers and overlapping classifications, but what if we have tons of them? → SVM

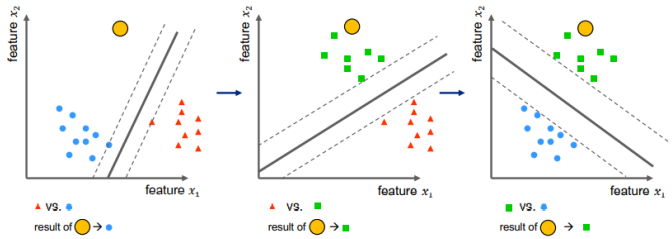
Support Vector Machine

1. Start with data in a relatively low dimension
2. Transform data into a higher dimension
3. Find a Support Vector Classifier that separates the transformed data



Multiclass Classification

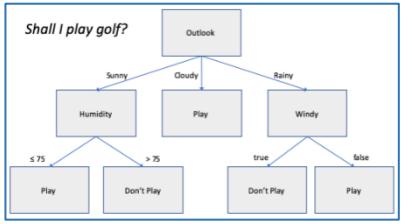
SVM is a binary classifier so if it must separate more than two classes it does a pairwise comparison. Classify unknown instance by majority vote among all pairwise comparisons.



Decision Tree

Decision Tree

A decision tree is a flowchart-like structure



- Node test on an attribute
- Branch outcome of the test
- Leaf class label
- Path classification rule

Measures

Impurity (Gini) measures how often a randomly chosen element from the set would be incorrectly labelled.

$$Gini = 1 - p(\text{yes})^2 - p(\text{no})^2$$

$$Total_{Gini} = \left(\frac{True}{True + False} \cdot Gini_{True}\right) + \left(\frac{False}{True + False} \cdot Gini_{False}\right)$$

Advantages

- Simple to understand and interpret
- Have value even with little hard data
- Support to determine worst, best, and expected values for different scenarios
- Can be combined with other decision techniques and to form random forests

Disadvantages

- They are unstable, a small change in the data can lead to a large change in the structure of the optimal decision tree
- They are often relatively inaccurate, other predictors perform better with similar data
- Calculations can get very complex, particularly if many values are uncertain and/or if many outcomes are linked

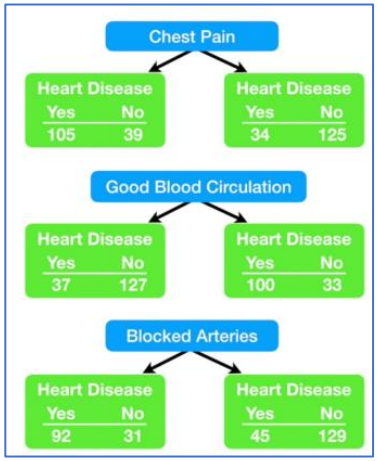
Step 1: Calculate Gini Impurity

$$T = 1 - \left(\frac{105}{144}\right)^2 - \left(\frac{39}{144}\right)^2 = 0.395, \qquad F = 1 - \left(\frac{34}{159}\right)^2 - \left(\frac{125}{159}\right)^2 = 0.336$$

$$Tot_{Weighted} = \left(\frac{144}{144 + 159} \cdot 0.395\right) + \left(\frac{159}{144 + 159} \cdot 0.336\right) = 0.364$$

Gini coefficients

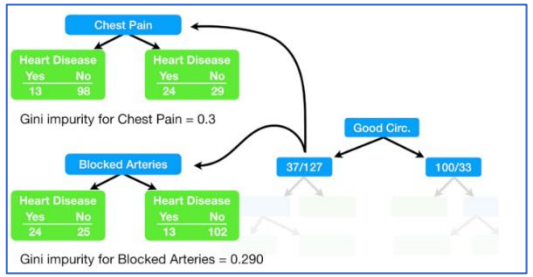
- Chest Pain $= 0.364$
- Good Blood Circulation $= 0.360$
- Blocked Arteries $= 0.381$



Step 2: Lowest Impurity = Root

- Root = Good Blood Circulation

Step 3: Calculate Gini Impurity for Branches



Naive Bayes

Naive Bayes

Naïve Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naïve) independence assumptions between the features.

Bayes theorem

- X is a data tuple
- H is some hypothesis, such as X belongs to a specified class C
- $P(H|X)$ is the posteriori probability of H conditioned on X
- $P(C_i|X)$ is the probability that X belongs to C_i, given that we know the attribute description of X

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

Advantages

- Fast to train and classify
- Performance is similar to decision trees and neural networks
- Easy to implement
- Handles numeric and categorical data
- Useful for very large data sets

Disadvantages

- Assumes class conditional independence, therefore, loss of accuracy
- Model is difficult to interpret

Example

$$C_1 = buys_{computer} = yes, \qquad C_2 = buys_{computer} = no$$

$$X = (age = youth; income = med; student = yes; credit_{rating} = fair)$$

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

1: Compute $P(C_i)$

$$P(C_1) = 0.643, \qquad P(C_2) = 0.357$$

2: Compute $P(X|C_i)$

- $P(X|C_1) = P(age = youth|C_1) \cdot P(income = med|C_1) \ldots = 0.044$
- $P(X|C_2) = P(age = youth|C_2) \cdot P(income = med|C_2) \ldots = 0.019$

3: Compute $P(X|C_i) \cdot P(C_i)$

- $P(X|C_1) \cdot P(C_1) = 0.044 \cdot 0.643 = 0.028$
- $P(X|C_2) \cdot P(C_2) = 0.019 \cdot 0.357 = 0.007$

Partitioning Clustering

Clustering is the task of grouping a set of objects in such a way that objects in the same group (cluster) are more similar to each other than to those in other groups.

Clustering Models

- Connectivity models — For example, *hierarchical clustering* builds models based on distance connectivity
- Centroid models — For example, the *k-means* algorithm represents each cluster by a single mean vector
- Distribution models — Clusters are modelled using statistical distributions, such as multivariate normal distributions
- Density models — For example, *DBSCAN* and *OPTICS* defines clusters as connected dense regions in the data space

Cluster Properties

- Clusters may have different sizes, shapes, and densities
- Clusters may form a hierarchy
- Clusters may be overlapping or disjoint

K-means clustering

Aim to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean.

Input

- $D = p_1, \dots, p_n$ Points
- k Number of clusters

Output

- $C = c_1, \dots, c_k$ Cluster centroids
- $D \to 1, \dots, k$ Cluster membership

Distance functions $dist(x, y)$

$$euclid = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2}, \qquad manhatten = \sqrt{\sum_{i=1}^{d} |x_i - y_i|}$$

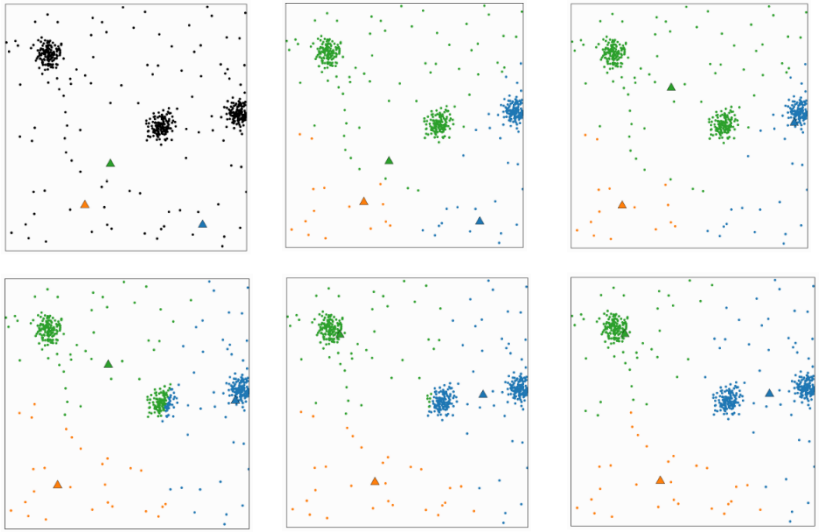
K-Means Clustering

Step 1: Choose k objects as initial cluster centers.

Step 2: Assign each data point to the cluster which has the closest mean point (centroid) under chosen distance metric.

Step 3: When all data points have been assigned, recalculate the positions of k centroids (mean points).

Step 4: Repeat steps 2 and 3 until the centroids do not change anymore.



Partitioning Clustering

Clustering evaluation without labels

- Elbow method
  a heuristic used in determining the number of clusters in a data set
- Silhouette coefficient
  measure for cluster validity/consistency
- Dendrogram
  visual inspection of distances and balancing in hierarchical clustering

Clustering evaluation with labels

- Purity a simple and transparent measure of correctness of clustering
- Rand index compares two clusterings
- Precision and Recall can also be applied
- Misclassification rate MR
  - $N =$ number of samples
  - $C =$ number of true clusters
  - $e_j =$ number of wrongly assigned samples of true cluster j

$$MR = \frac{1}{N}\sum_{j=1}^{c} e_j$$

Silhouette coefficient: measure for cluster validity / consistency

The silhouette score falls within the range $[-1,1]$

- $1 \rightarrow$ clusters are very dense and nicely separated.
- $0 \rightarrow$ clusters are overlapping
- $< 0 \rightarrow$ cluster data may be wrong/incorrect

The silhouette plots can be used to select the most optimal value of K in *k-means clustering*.

ISODATA (*k-means refinement*)

It doesn't need to know the number of clusters. Clusters are merged if either the number of members in a cluster is less than a certain threshold or if the centers of two clusters are closer than a certain threshold. The clusters are split into two different clusters if the clusters standard deviation exceeds a predefined value and the number of members is twice the threshold for the minimum number of members. The user must provide several additional parameters.

1. Choose k objects as initial cluster centers
2. (Re)assign each object to the cluster to which the object is the most similar based on the mean value of the objects in the cluster
3. Recalculate the centroid by taking the average of all points in the cluster.
4. Reassignment of data points do not change clusters. Clusters are splitted, merged or eliminated, if requirements are not met.
5. Repeat steps 2 and 3 until the centroids do not change anymore.

Hierarchical Density Clustering

Hierarchical clustering builds models based on distance connectivity and merging of clusters with minimum distance.

Bottom-up strategy: initially each data object is in its own atomic cluster. Then merge these atomic clusters into larger and larger clusters.

Method

Step 1: Form initial clusters consisting of a single object and compute the distance between each pair of clusters

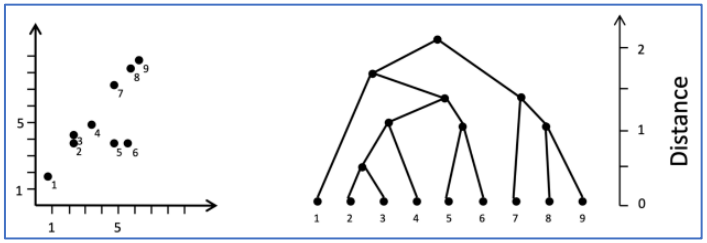
Step 2: Merge the two clusters having minimum distance

Step 3: Calculate the distance between the new cluster and all other clusters

Step 4: Repeat step 2 and 3 until there is only one cluster remaining

A **Dendrogram** is a tree of nodes representing clusters, satisfying the following properties:

- *Root* represents the whole data set
- *Leaf* nodes represent clusters containing a single object
- *Inner* nodes represent the union of all objects contained in its corresponding subtrees



Single Linkage: Shortest distance single link (min) between an element in one cluster and an element in the other, i.e., $d(C_i, C_j) = \min\{d(x_{ip}, x_{jq})\}$.

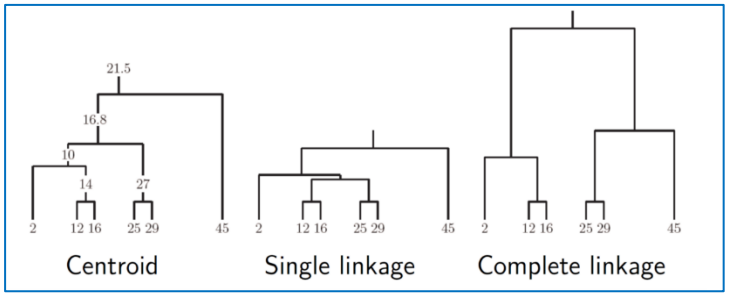
Complete Linkage: Largest distance between an element in one cluster complete link (max) and an element in the other, i.e., $d(C_i, C_j) = \max\{d(x_{ip}, x_{jp})\}$.

Average Linkage: Average distance between average elements in one cluster and elements in the other, i.e. $d(C_i, C_j) = avg\{d(x_{ip}, x_{jp})\}$.

Example:

Clustering of the 1-dimensional data set $\{2, \mathbf{12}, \mathbf{16}, \mathbf{25}, \mathbf{29}, 45\}$

1. Find lowest distance $\rightarrow (10, \mathbf{4}, 9, \mathbf{4}, 45)$ and build clusters
   $\{2, (12,16), (25,29), 45\}$
2. Find lowest distance based on metrics and build clusters
   - Centroid $\{2, (12,16), (25,29), 45\} \rightarrow (\mathbf{12}, 13,18)$
     $\{(\mathbf{2}, \mathbf{12}, \mathbf{16}), (25,29), 45\}$
   - Single $\{2, (12,16), (25,29), 45\} \rightarrow (10, \mathbf{9}, 16)$
     $\{2, (\mathbf{12}, \mathbf{16}, \mathbf{25}, \mathbf{29}), 45\}$
   - Complete $\{2, (12,16), (25,29), 45\} \rightarrow (\mathbf{14}, 17,16)$
     $\{(\mathbf{2}, \mathbf{12}, \mathbf{16}), (25,29), 45\}$

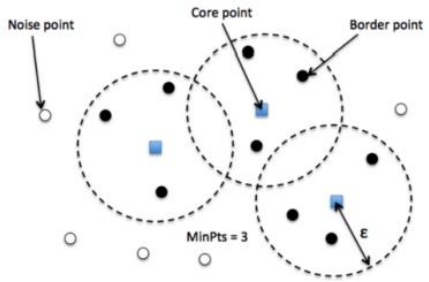


Hierarchical Density Clustering (DBSCAN)

Density based Clustering

Density based clustering defines clusters as connected dense regions in the data space. There is no need to specify the number of clusters.

DBSCAN (**D**ensity-**B**ased **S**patial **C**lustering of **A**pplications with **N**oise)



- min_{points} minimum number of points clustered together for a region to be considered dense
- ε a distance measure that will be used to locate the points in the neighbourhood of any point
- *core* a point that has at least m points within distance n from itself
- *border* a point that has at least one core point at a distance n
- *noise* a point that is neither core nor a border with less than m points within distance n from itself.

Method

1. It starts with a random unvisited point. All points within a distance ε classify as neighborhood points
2. It needs a minimum number of points min_{points} within the neighbourhood to start the clustering process. Otherwise, the point gets labeled as Noise.
3. All points within the distance ε become part of the same cluster. Repeat the procedure for all the new points added to the cluster group. Continue till it visits and labels each point within the ε neighborhood of the cluster.
4. On Completion of the process, it starts again with a new unvisited point thereby leading to the discovery of more cluster or noise. At the end of the process each point is marked.

Advantages

- No need to specify the number of clusters
- Able to find arbitrarily shaped clusters
- Able to detect noise

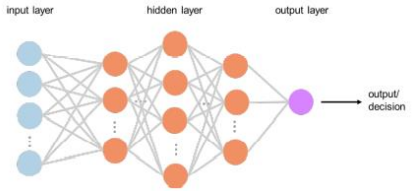
Disadvantages

- Cannot cluster data sets well with large differences in densities

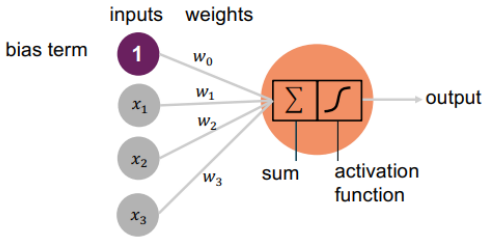
Neural Networks

A **Neural Network** has an *Input Layer*, one or several *Hidden Layers* and an *Output Layer*.

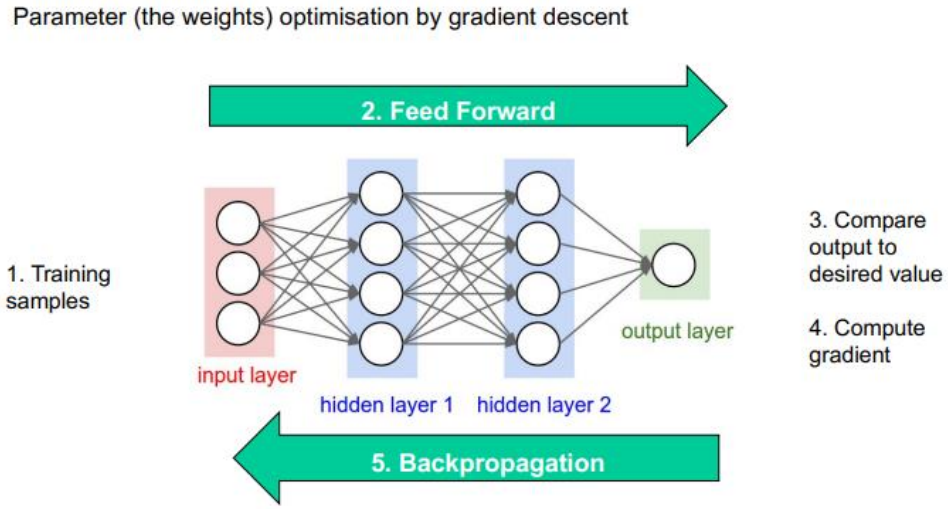
- Each neuron in one layer has directed connections to the neurons of the subsequent layer
- No connections between nodes within the same layer
- Parameters: Weights connecting the layer



A Neuron applies an activation function to the weighted sum of its inputs



Training in **Feedforward Neural Networks**

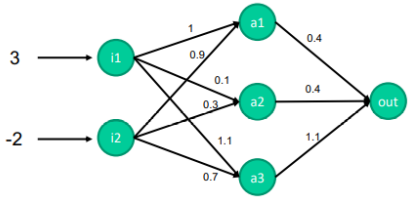


Training process

1. Choose one sample from our dataset. We only operate on one sample at a time.
2. Calculate all the partial derivates of loss with respect to weights or biases.
3. Use the update equation to update each weight and bias.
4. Go back to step 1.

A Simple Example

- w_0 = bias for all nodes = 0.2
- $act(z) = \begin{cases} z & if z \geq 0 \\ 0 & otherwise \end{cases}$



$$z_1 = i_1 \cdot 1 + i_2 \cdot 0.9 + w_{a1,0} = 3 \cdot 1 - 2 \cdot 0.9 + 0.2 = 1.4$$

$$a_1 = act(1.4) = 1$$

$$z_2 = i_1 \cdot 0.1 + i_2 \cdot 0.3 + w_{a2,0} = 3 \cdot 0.1 - 2 \cdot 0.3 + 0.2 = -0.1$$

$$a_2 = act(-0.1) = 0$$

$$z_3 = i_1 \cdot 1.1 + i_2 \cdot 0.7 + w_{a3,0} = 3 \cdot 1.1 - 2 \cdot 0.7 + 0.2 = 2.1$$

$$a_3 = act(2.1) = 1$$