CLASSIFICATION

— EVALUATION METRICS :

- confusion matrix :

true \ predicted	Y	N
Y	TP	FN
N	FP	TN

- accuracy = $\frac{\text{true}}{\text{overall}}$
- precision : $p = \frac{TP}{TP+FP}$ = $\frac{\text{true positive}}{\text{classified positive}}$ ↑ FP↓
- recall : $r = \frac{TP}{TP+FN}$ = $\frac{\text{true positive}}{\text{positive}}$ ↑ FN↓
- F1-measure : $= \frac{2rp}{r+p}$ ↑ FP↓ FN↓
- sensitivity = $\frac{TP}{TP+FN} = r$ = true positive rate
- specificity = $\frac{TN}{TN+FP}$ = true negative rate

— PROBLEMS

- Bayes
- Decision tree

BAYES

$$\text{class} = \arg\max \; \mathbb{P}(y|\vec{x}) \overset{NB}{=} \arg\max \; \mathbb{P}(x_1|y) \cdots f(x_j|y) \cdots \mathbb{P}(x_n|y) \; \mathbb{P}(y)$$

Suppose the dataset :

Outlook	Temp	Humidity	Windy	Play
sunny/overcast/rainy	IR	IR	T/F	Y/N

play = target

we create :

Outlook

	Y	N
sunny	2 +1	3 +1
overcast	4 +1	0 +1
rainy	3 +1	2 +1

	Y	N
sunny	1/4	1/2
overcast	5/12	1/8
rainy	1/3	3/8

Temp

Y	N
64.68	65.71
69.70	72.80
72...	85...
...	...

	Y	N
	$\mu=73$ * $\sigma=6.2$	$\mu=75$ $\sigma=7.9$

Humidity

Y	N
67.70	70.85
70.78	90.91
80...	95...
...	...

	Y	N
	$\mu=79$ $\sigma=10.2$	$\mu=86$ $\sigma=9.2$

Windy

	Y	N
true	6 +1	2 +1
false	3 +1	3 +1

	Y	N
true	7/11	3/7
false	4/11	4/7

Play

Y	N
9	5

$\mathbb{P}(Y) = 9/14$

$\mathbb{P}(N) = 5/14$

→

Outlook	Temp	Humidity	Windy	Play
sunny	66	90	T	?

likelihood :

$$\ell(Y|\vec{x}) = \mathbb{P}(\text{outlook}=\text{sunny}|Y) \cdot f(\text{temp}=66|Y)^{*} \; f(\text{humidity}=90|Y) \; \mathbb{P}(\text{windy}=T|Y) \; \mathbb{P}(Y)$$

$$\ell(N|\vec{x}) = \mathbb{P}(\text{sunny}|N) \; f_T(66|N) \; f_H(90|N) \; \mathbb{P}(T|N) \; \mathbb{P}(N)$$

!!!! → Normalization :

$$\mathbb{P}(Y|\vec{x}) = \frac{\ell(Y|\vec{x})}{\ell(Y|\vec{x}) + \ell(N|\vec{x})} \quad ; \quad \mathbb{P}(N|\vec{x}) = 1 - \mathbb{P}(Y|\vec{x})$$

$$^{*} \; f(\text{temp}=60|Y) = \frac{1}{\sqrt{2\pi\,(6.2)^2}} \; e^{-\frac{(66-73)^2}{2\cdot(6.2)^2}} \qquad \frac{1}{\sqrt{2\pi\sigma^2}} \; e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

ASSOCIATION RULES

- Apriori
- Eclat
- FP-tree

} frequent itemset

- Rule generation
- Summarize itemsets
- Trawling
- Frequent sequences

$$support = \frac{\sigma(|X,Y|)}{\#\,transactions} \sim \mathbb{P}(X,Y)$$

$$confidence = \frac{\sigma(|X,Y|)}{\sigma(|X|)} \sim \mathbb{P}(Y|X) \sim \mathbb{P}(X \Rightarrow Y)$$

APRIORI

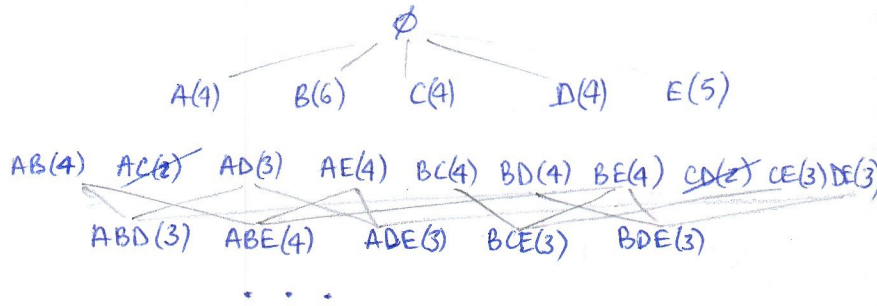
Treshold = 0.5

Transaction database

1	A	B		D	E
2		B	C		E
3	A	B		D	E
4	A	B	C		E
5	A	B	C	D	E
6		B	C	D	

(treshold = 3/6)

→



ECLAT

Treshold = 0.5

Transaction database

1	..
6	

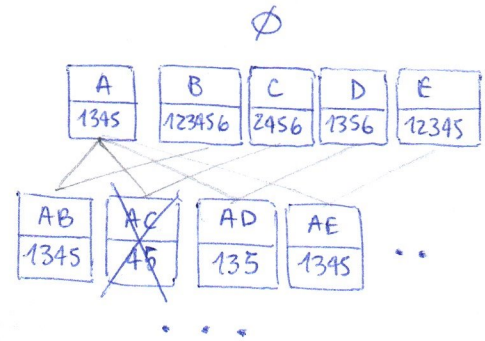
(↑) (treshold = 3/6)

→

Vertical database

A	B	C	D	E
1	1	2	1	1
3	2	4	3	2
4	3	5	5	3
5	4	6	6	4
	5			5
	6			

→



FP-TREE

Treshold = 0.5

Transaction database

1	..
6	

(↑) (treshold = 3/6)

→

Ordering:

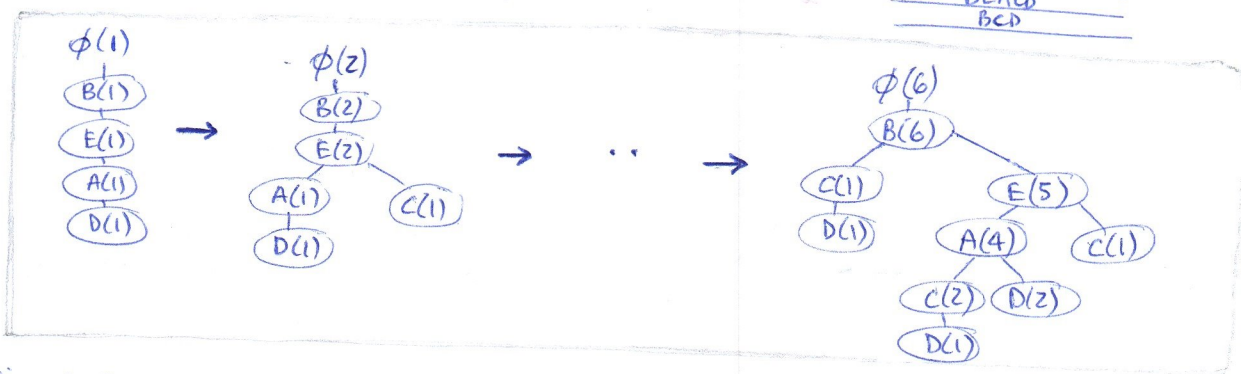
B(6)
E(5)
A(4)
C(4)
D(4)

→

New transaction database:

BEAD
BEC
BEAD
BEAC
BEACD
BCD

→

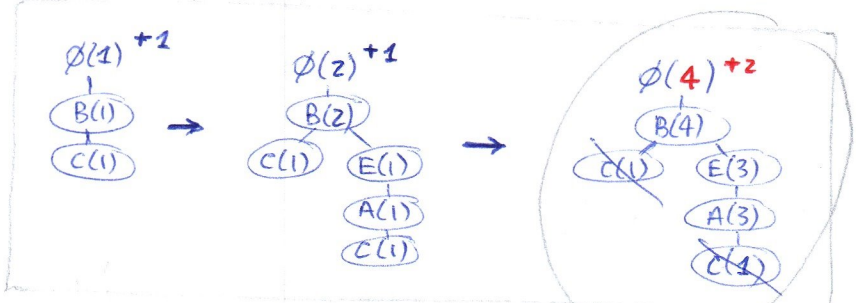


Conditional FP-tree for D:

Paths that end with D

BCD	count(D) = 1
BEACD	count(D) = 1
BEAD	count(D) = 2

→



here c(2), still not enough

→ Given D we have:

BEA, BE, EA, BE, B, E, A

(s(c) < minsup = 3)

→

Frequent itemsets

DBEA(3), DBE(3), DB(4)

DEA(3) DE(3)

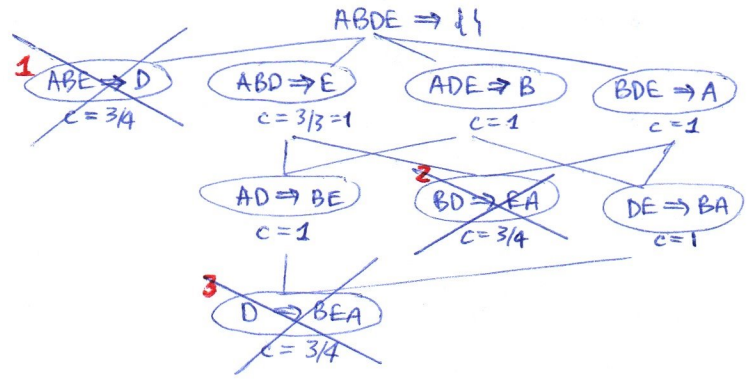
DBE(3) DA(3)

RULE GENERATION

min conf = 0.9

Support	(Frequent) itemsets
6	B
5	E, BE
4	A, C, D, AB, AE, BC, BD, ABE
3	AD, CE, DE, ABD, ADE, BCE, BDE, ABDE

Start from the largest: ABDE



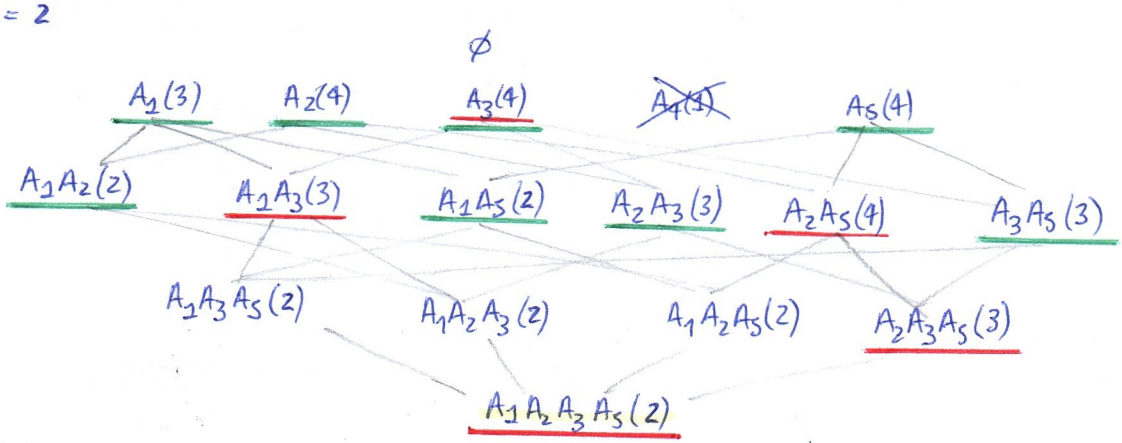
→ We don't need to investigate:
1. ABE, AB, BE, AE, A, B, E
2. BD, B, D
3. D

→ Association rules:
ABD ⇒ E, ADE ⇒ B, BDE ⇒ A, AD ⇒ BE, DE ⇒ BA

SUMMARIZING ITEMSETS

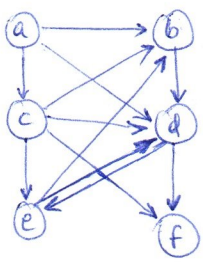
- CLOSED ITEMSET X: all super-itemsets containing X have < support ↑ ⇉
- MINIMAL GENERATOR X: all sub-sets of X have > support ↓ ⇉
- MAXIMAL FREQUENT ITEMSET X: no frequent subset containing X

minsup = 2



Remember:
1st line: if frequent ⇒ —
last line: if frequent ⇒ —

TRAWLING



→ Itemsets:
a = {b, c, d}
b = {d}
c = {b, d, e, f}
d = {e, f}
e = {b, d}
f = {}

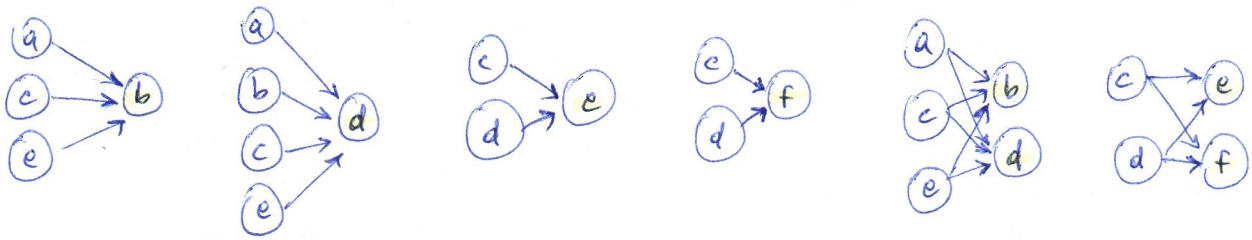
→ Frequent itemsets: minsup = 2

∅
b(3) ~~c(1)~~ d(4) e(2) f(2)
bd(3) ~~be(1)~~ ~~bf(1)~~ ~~de(1)~~ ~~df(1)~~ ef(2)

⇒ b(3), d(4), e(2), f(2), bd(3), ef(2)

Communities:

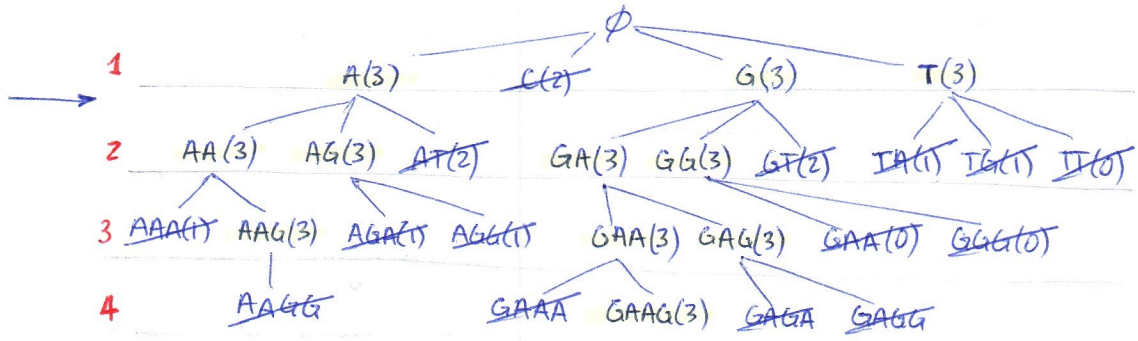
→



FREQUENT SEQUENCES

minsup = 3

S1	CAGAAGT
S2	TGACAG
S3	GAAGT



At level 1 dies C → we don't add it anywhere in 2,3,4
At level 2 dies T → we don't add it anywhere in 3,4
At level 4 dies A

CLUSTERING

- DISTANCES :
  - Euclidean : $d(x,y) = \sqrt{\Sigma_i (x_i - y_i)^2}$
  - Manhattan : $d(x,y) = \Sigma_i |x_i - y_i|$
  - Jaccard : similarity : $J(x,y) = \frac{|x \cap y|}{|x \cup y|}$

    distance : $d(x,y) = 1 - J(x,y)$
  - Hamming : $d(x,y) = \frac{\#\text{mismatch}}{\#\text{overall}}$
  - Cosine : similarity $= \frac{\Sigma x_i y_i}{\sqrt{(\Sigma x_i^2)(\Sigma y_i^2)}}$

    distance : $d(x,y) = \arccos(\text{similarity})$
  - Edit : $d(x,y) = |x| + |y| - 2|LCS|$ (longest common subsequence)

x	0	0	1	0	0	2
y	0	0	1	0	1	1

✓ ✗ ✗ J
✓ ✓ ✓ ✓ ✗ ✗ H

$d_J = 1 - \frac{1}{3} = \frac{2}{3}$

$d_H = \frac{2}{6} = \frac{1}{3}$

- LINKAGE
  - single (min) : $d(A,B) = \min d(x_i^A, x_j^B)$
  - complete (max) : $d(A,B) = \max d(x_i^A, x_j^B)$
  - mean/centroid : $d(A,B) = d(\mu_A, \mu_B)$
  - group average : $d(A,B) = \text{avg } d(x_i^A, x_j^B)$
- PROBLEMS
  - Hierarchical clustering
  - DBSCAN
  - HDBSCAN
  - silhouette coefficient & k-means

HIERARCHICAL CLUSTERING

single linkage

Distance matrix : D =

	1	2	3	4	5
1	/				
2	2	/			
3	6	5	/		
4	10	9	4	/	
5	9	8	5	3	/

→ Merge {1},{2}, d = 2

$d(\{1,2\},\{3\}) = \min(d(1,3), d(2,3)) = \min(6,5) = 5$

$d(\{1,2\},\{4\}) = 9$

$d(\{1,2\},\{5\}) = 8$

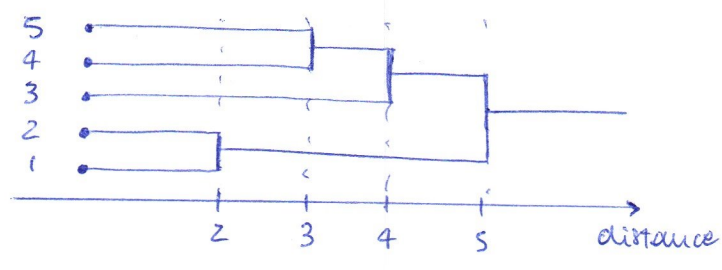
D =

	{1,2}	3	4	5
{1,2}	–			
3	5	–		
4	9	4	–	
5	8	5	3	–

→ Merge {4},{5}, d = 3

...

→ Dendrogram :

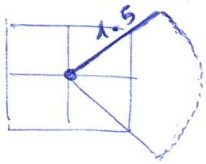


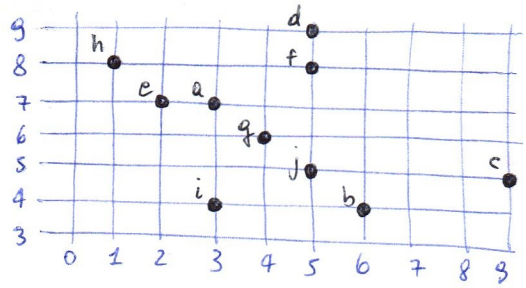
DBSCAN

- CORE POINT x : $|N_\varepsilon(x)| \geq minpts$
- BORDER POINT x : $|N_\varepsilon(x)| < minpts$, $x \in N_\varepsilon(\text{core point})$
- NOISE POINT x : $|N_\varepsilon(x)| < minpts$, $x \notin N_\varepsilon(\text{core point})$

! The point itself **is** part of the count

$\varepsilon = 1.5$, $minpts = 3$





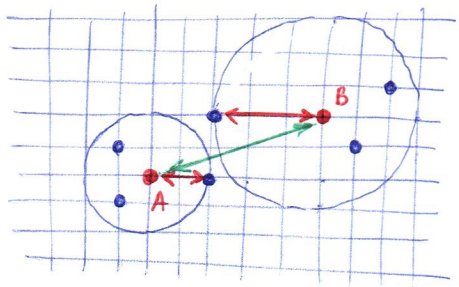
→

	$\#N_\varepsilon(\cdot)$	
a	3	core
b	2	border
c	1	noise
d	2	noise
e	3	core
f	2	noise
g	3	core
h	2	border
i	1	noise
j	3	core

HDBSCAN

- CORE DISTANCE OF x : max distance of x to its k-neaarest neig.
- MUTUAL REACHABILITY DISTANCE : $d_{mreach}(a,b) = \max\{core_k(a), core_k(b), d(a,b)\}$

$k = 3$



→

$core_3(A) = 2$

$core_3(B) = 3$

$d(a,b) = \sqrt{29} = 5.38$

$d_{mreach}(a,b) = \sqrt{29} = 5.38$

SILHOUETTE COEFFICIENT

↳ For every point x_i : a_i = average distance with points inside x_i's cluster
b_i = average distance with points of the closest cluster

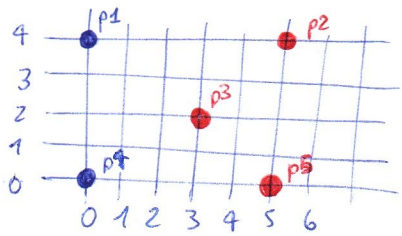
$$s_i = \frac{b-a}{\max\{a,b\}}$$

Then average on i.

distance : Euclidean

	x	y	label
p1	0	4	0
p2	5	4	1
p3	3	2	1
p4	0	0	0
p5	5	0	1

→



→

p1) $a = d(p1,p4) = 4$; $b = \frac{1}{3}(d(p1,p2) + d(p1,p3) + d(p1,p5)) = 5$

p2) $a = \frac{1}{2}(d(p2,p3) + d(p2,p5)) = 2+\sqrt{2} = 3.4$; $b = \frac{1}{2}(d(p2,p1) + d(p2,p4)) = 5.7$

p3) $a = \frac{1}{2}(d(p3,p2) + d(p3,p5)) = 2\sqrt{2} = 2.83$; $b = \frac{1}{2}(d(p3,p4) + d(p3,p1)) = 3.6$

p4) $a = a(p1)$; $b = b(p1)$

p5) $a = a(p2)$; $b = b(p2)$

→

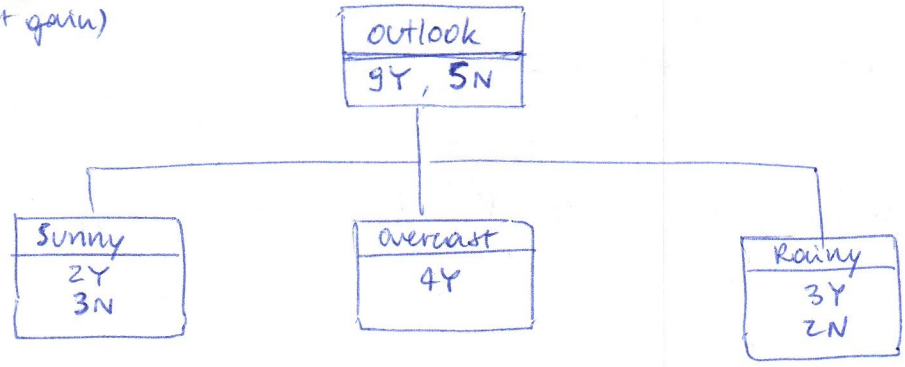
	a	b	s_i
p1	4	5	0.2
p2	3.4	5.7	0.4
p3	2.83	3.6	0.21
p4	4	5	0.2
p5	3.4	5.7	0.4

→ $s_{TOT} = \frac{\sum_i s_i}{n} = 0.282$

DECISION TREE

(gain → we take the attribute with highest gain)

$$entropy([p_1, \dots, p_n]) = -\sum_{j=1}^{n} p_j \log_2(p_j)$$



1. Entropy before splitting

$$info([9,5]) = entropy([9/14, 5/14]) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.94$$

2. Informations for each value:

sunny : $info([2,3]) = entropy([2/5, 3/5]) = -\frac{2}{5}\log_2\left(\frac{2}{5}\right) - \frac{3}{5}\log_2\left(\frac{3}{5}\right) = 0.971$

overcast : $info([4,0]) = entropy([1,0]) = 0$

rainy : $info([3,2]) = info([2,3]) = 0.971$

3. Entropy after the splitting

$$info(split) = \frac{5}{14} info_{sunny} + \frac{4}{14} info_{overcast} + \frac{5}{14} info_{rainy}$$

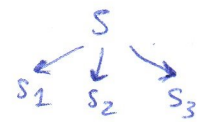
$$= \frac{5}{14} \cdot 0.971 + \frac{4}{14} \cdot 0 + \frac{5}{14} \cdot 0.971 = 0.693$$

4. Gain

$$gain(outlook) = entropy_{before} - entropy_{after} = 0.94 - 0.693 = 0.247$$

Intrinsic information of a split:

$$IntrinsicInfo(S, split) = -\sum_i \frac{|S_i|}{|S|} \log_2\left(\frac{|S_i|}{|S|}\right)$$



Information gain ratio:

$$GainRatio(S, split) = \frac{gain(S)}{IntrinsicInfo(S, split)}$$

$$gini(D) = 1 - \sum_{j=1}^{c} p_j^2$$

c = # classes in D

p_j = % of elems of class j

$$gini(split) = \sum \frac{|D_i|}{|D|} gini(D)$$

$$\Delta gini(split) = gini(D) - gini(split)$$

we look for the lower gini.

split : {sunny}, {overcast, rainy}

1. $gini(outlook) = 1 - (9/14)^2 - (5/14)^2 = 0.46$
2. $gini(sunny) = 1 - (2/5)^2 - (3/5)^2 = 0.48$

   $gini(\text{overcast}, \text{rainy}) = 1 - (7/9)^2 - (2/9)^2 = 0.364$
3. $gini(split) = \frac{5}{14} 0.48 + \frac{9}{14} 0.364 = 0.394$
4. $\Delta gini(split) = gini(outlook) - gini(split)$

   $= 0.46 - 0.394 = 0.066$

DECISION TREE – NUMERICAL

1. sort the values (with class labels)
2. check feasible cut points — those that have a change in the output
3. choose the one with best information gain

Ex.

1	1	2	3	3	3	3	4	4	4	4	4
N	Y	N	Y	N	N	Y	N	Y	Y	Y	N

(1) (2) (3)

6Y
6N

1. Entropy before:

$$info(D) = info([6,6]) = entropy([1/2, 1/2]) = 1$$

2. Informations of the values

(1)
$$info(\leq 1) = info([1,1]) = entropy([1/2,1/2]) = 1$$
$$info(>1) = info([5,5]) = entropy([1/2,1/2]) = 1$$

(2)
$$info(\leq 2) = info([1,2]) = entropy([1/3,2/3]) = -1/3\log_2(1/3) - 2/3\log_2(2/3) = 0.918$$
$$info(>2) = info([5,4]) = entropy([5/9,4/9]) = 0.991$$

(3)
$$info(\leq 3) = info([3,4]) = entropy([3/7,4/7]) = 0.985$$
$$info(>3) = info([3,2]) = entropy([3/5,2/5]) = 0.971$$

3. Entropy after

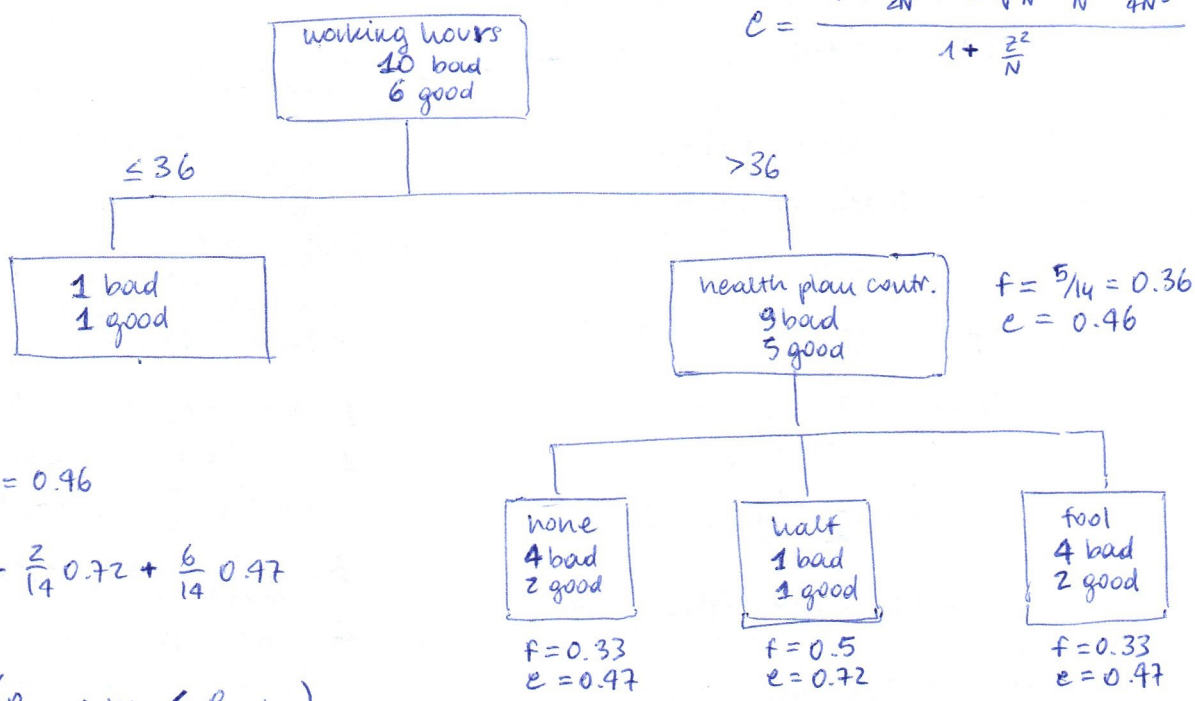
$$info((1)) = \frac{1}{6}\cdot 1 + \frac{1}{6}\cdot 1 = 1$$
$$info((2)) = \frac{1}{4}\cdot 0.918 + \frac{3}{4}\cdot 0.991 = 0.973$$
$$info((3)) = \frac{7}{12}\cdot 0.985 + \frac{5}{12}\cdot 0.971 = 0.979$$

4. Gain

$$gain((1)) = 1 - 1 = 0$$
$$gain((2)) = 1 - 0.973 = 0.027 \leftarrow$$
$$gain((3)) = 1 - 0.979 = 0.021$$

PRUNING

$$e = \frac{f + \frac{z^2}{2N} + z\sqrt{\frac{f}{N} - \frac{f^2}{N} + \frac{z^2}{4N^2}}}{1 + \frac{z^2}{N}}$$



- no split: $e = 0.46$
- split:

$$e = \frac{6}{14}0.47 + \frac{2}{14}0.72 + \frac{6}{14}0.47 = 0.51$$

$\Longrightarrow$ prune! ($e_{no\_split} < e_{split}$)

ADABOOST (ENSEMBLE)

$$\varepsilon_i = \sum_j w_j \, \mathbb{1}\{y_j \neq h_i(x_j)\}$$

$$\alpha_i = \frac{1}{2} \ln\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$$

$$H(x) = \text{sign}\left(\sum_i \alpha_i h_i(x)\right)$$

Ex.

Dataset :

x	0	1	2	3	4	5	6	7	8	9
y	+1	+1	+1	-1	-1	-1	-1	-1	+1	+1

Initial weights → uniform :

w	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
x	0	1	2	3	4	5	6	7	8	9
y	+1	+1	+1	-1	-1	-1	-1	-1	+1	+1

We bootstrap the data with the assigned weights.
We generate h_1 that returns +1 if $x \leq 2$

w	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
x	0	1	2	3	4	5	6	7	8	9
y	+1	+1	+1	-1	-1	-1	-1	-1	+1	+1
y-pred	+1	+1	+1	-1	-1	-1	-1	-1	-1	-1
									×	×

$\Rightarrow \varepsilon_1 = \frac{2}{10}$, $\alpha_1 = 0.69$

The new weights are : $w_2 = w_1 e^{-\alpha_1}$ if correct, $w_2 = w_1 e^{\alpha_1}$ if wrong

We bootstrap and we generate h_2 that returns +1 if $x \geq 8$

w	0.0625							0.0625	0.25	0.25
x	0	1	2	3	4	5	6	7	8	9
y	+1	+1	+1	-1	-1	-1	-1	-1	+1	+1
y-pred	-1	-1	-1	-1	-1	-1	-1	-1	+1	+1
	×	×	×							

$\Rightarrow \varepsilon_2 = 0.0625 + 0.0625 + 0.0625 = 0.1875$

$\alpha_2 = 0.7332$

The new weights are : ...

The final model is :

$$H(x) = \text{sign}\left(0.69\, h_1(x) + 0.73\, h_2(x)\right)$$

TEXT MINING

how many d_1

$$f(q,d) = q_1 d_1 + \dots + q_n d_n = q_1(w_1 d_1) + \dots + q_n(w_n d_n)$$

q = "news about presidential campaign"

d_1 = .. news about ..

d_2 = .. news about .. campaign ..

d_3 = .. news .. presidential campaign ..

d_4 = .. news .. presidential campaign .. presidential ..

d_5 = .. news .. campaign .. campaign .. campaign .. campaign

$f(q,d_1) = 2, 2$

$f(q,d_2) = 3, 3$

$f(q,d_3) = 3, 3$

$f(q,d_4) = 3, 4$

$f(q,d_5) = 2, 5$

count of matchings

Count of matchings + frequency

To penalize words that frequently occur in many documents:

$$IDF(w) = \log_2\left(\frac{M}{k}\right)$$

M = # documents in the corpus

k = # documents that contain word w

$c(w_i, d)$ = frequency of word w_i in d

$$d_i = c(w_i, d) \cdot IDF(w_i)$$

$$q_i = c(w_i, q)\, IDF(w_i)$$

$$f(q,d) = \sum_{i=1}^{N} d_i q_i = \sum_{w \in (q \cap d)} c(w,d)\, c(w,q)\, (IDF(w))^2$$

We are given 40 documents:

- in 20 there is the word "set"
- in 10 there is the word "computing"
- in 5 there is the word "mining"

Which keyword is the most important according to IDF?

$IDF(\text{"set"}) = \log_2\left(\frac{40}{20}\right) = 1$

$IDF(\text{"computing"}) = \log_2\left(\frac{40}{10}\right) = 2$

$IDF(\text{"mining"}) = \log_2\left(\frac{40}{5}\right) = 3$ → most important since it appears in fewer documents

FEATURES SELECTION

- embedded approaches: feature selection is part of the algorithm (e.g. Lasso)
- filter approaches: features are selected ⊥ specific problem
- wrapper approaches: apply data mining algorithms as black box to find the best subset of features

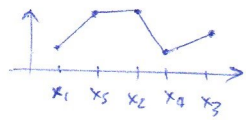
FILTER APPROACHES:

- reduced variance feature selection:
  eliminate the variables that have a variance < threshold
  (low variance → similar feature values → not informative (?!)
  (however a low-variance feature may be important)
- univariate feature selection:
  select the best features based on a statistical test
  (stat. test: how much does a variable explain the target (~ supervised method))
- PCA (normalize!):
  features that best capture the variance of the data
  (remember: we use only variables, not the target)
- recursive features elimination:
  we start with all and eliminate one-at-the time the ones that bring less informations about the target

WRAPPER APPROACHES: (wrapped around target)

- random forests:
  random forest has a function that, based on the forest, returns a score for each feature/variable (importance)
- hill climbing:
  - start from a random set of features
  - provide an evaluation of the performance (algorithm-dependent)
  - generate a perturbation in the solution
  - evaluate the performance of the perturbed sol. ↻
  - take the best
- genetic algorithms:
  as hill climbing but with "genetic variation"
- permutation importance
  (how important is a single feature)
  - train the model on a dataset
  - shuffle the values in a single column
  - apply the model to both datasets
  - feature importance = loss in performance in shuffled data

  } how much a feature is important for the model, not the target
- partial dependence plot
  (how the target depends on average on one feature)
  - variable $\mathbf{X} = \{x_1, \dots, x_n\}$
  - train a model
  - for each $x_i \in \mathbf{X}$:
    substitute x_i in all the column
    predict the output of the model for each sample
    average the outputs
  - plot:



DISCRETIZATION? continuous → nominal

- supervised: use informations about the target to generate intervals that preserve relevant informations (e.g. decision trees)
- unsupervised: do not ↑
  - equal width intervals
  - equal frequency intervals
  - k-means (unsupervised clustering)

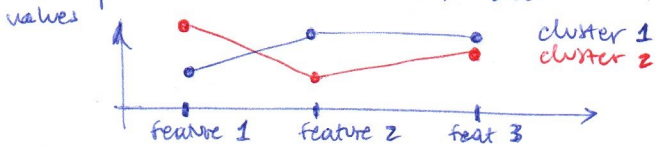
PCA? Standardize (StandardScaler().fit_transform(X)) — Standardize almost always (PCA, clustering, regression, ...)

Exploration?
- barplot for nominal
- heatmap for correlation: darker → + correlated (~1) (.heatmap())
- clustermap: - annot=True → clusters of columns (attributes)
  - ∅ → clusters of both attributes and values (hierarchical)
- distplot: histogram (if kde=False: no kernel density estimation) / map()
- scatterplots (.pairplot())
- scatterplots + histograms (.jointplot())
- boxplot (if jitter=true → points) / violinplot
- missing values (drop/mean/"None")
- skewness
- one-hot-encoding categorical / ordinal instead of categorical
- feature creation from datestamps
- data.describe() → mean, std, min, 25%, 50%, 75%, max

Clusters?
- inconsistency = $\frac{h - avg}{std}$ — compares the current high h of the merge with the average hight of the previous merges
- acceleration: we can search the highest acceleration of merge distance growth (in terms of distance)

Internal measures → WSS/BSS / External measures → already existing labels

Represent clusters?
Snake plots of standardized values



K-MEANS ASSUMPTIONS
problems with:
- wrong k
- different variance
- unevenly sized clusters
- different densities

Connectivity constraints
constraints that we can put on hierarchical agg. clustering.
For example we can say:
"a point can be merged only with points in the 10 nearest neig."
This prevents to merge clusters that are too far away.

SILHOUETTE
measures cohesion and separation, it's based on the difference between the average distance to points in the closest cluster and to points in the same cluster

$s_i = \frac{b_i - a_i}{\max\{b_i, a_i\}}$

b_i = avg d(x_i, points of the closest cluster)
a_i = avg d(x_i, points in the same cluster)

$s_i \sim \begin{cases} +1 & x_i \text{ close to its cluster points more than the closer cluster's points} \\ 0 & x_i \text{ close to boundary} \\ -1 & x_i \text{ closest to the closest cluster's points than its own} \end{cases}$

Ensambles
- generate models difficult to analize
- provide ways to score the variables used by the models (.feature_importances_)

TIME SERIES

- trend : long term increase/decrease
- seasonal pattern : if ∃ seasonal factors
- cyclic : rises/falls that are not of fixed frequency

} plot a series against time

- time series : additive / multiplicative
- .head(), .describe(), .describe(include=[np.objects]), .info(), .plot()
  (.describe() : numerical ; .describe(include=[np.objects]) : numerical + categorical)
- sample at a given frequency (! N
- filling strategies :
  - pad/ffill : use last valid obs
  - backfill/bfill : use next valid obs
  - fill_value : we put the value
- shift :
  - .shift : shifting ahead of 1 value in the index
  - .shift(n) : shift ahead of n (if < 0 ⇒ shift back)
  - .asfreq('3D').shift(n) : shift ahead of n but take the frequence of 3days
- percentage change : $\frac{x_t - x_{t-1}}{x_{t-1}}$ (.pct_change())
- correlation between pairs of series (CF)
  measures how much two series vary together

$$r_{xy} = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sigma_x \sigma_y}$$

  - high positive → (↓↓), (↑↑)
    high negative → (↓↑), (↑↓)
    low → weak association

autocorrelation : (.autocorr()) (ACF)

$r_k = corr(x_t, x_{t+k})$: how a series is correlated with a lagged copy of itself
(any substantial non-zero autocorrelation implies that series can be forecast from past)
→ we can plot it with correlograms

- random noise : $\mathbb{E} = 0$, $\sigma^2 = 1$, zero autocorrelation
- random walk : $p_{t+1} = p_t + \varepsilon_t$
  with drift : $p_{t+1} = \mu + p_t + \varepsilon_t$

} unpredictable

statistical test :

$p_{t+1} = \alpha + \beta p_t + \varepsilon_t$ { $H_0 : \beta = 1$ → random walk ; $H_1 : \beta < 1$ → not

$p_{t+1} - p_t = \alpha + \beta p_t + \varepsilon_t$ { $H_0 : \beta = 0$ → r.w. ; $H_1 : \beta < 0$ → not (Dickey-Fuller)

- stationary : statistical properties are constant over time
  (zero trend, const variance, const autocorrelation)
  non-stationary ? → diff (first difference) (r.w. non-stat, its diff. stationary)
  statistical test : augmented Dickey-Fuller : H_0 : **non**-stat.
- ARMA(p,q) : $R_t = \mu + \phi R_{t-1} + \varepsilon_t + \vartheta \varepsilon_{t-1}$ (p=q=1)

  today_return = mean + ϕ yesterday_return + today_noise + ϑ yesterday_noise
- ARMAX(1,1) : $R_t = \mu + \phi R_{t-1} + \varepsilon_t + \vartheta \varepsilon_{t-1} + \gamma d_{t-1}$

  external variable to bring extra (⊥ series) info
- ARIMA(1,1,1) : $\Delta R_t = \alpha \Delta R_{t-1} + \varepsilon_t + \beta \varepsilon_{t-1}$ ← (✗ non-stat)

  $\Delta R_t = R_t - R_{t-1}$
- Diagnostic : analysis of residuals (real values − predicted) ~ white noise ?
  - plot
  - histogram
  - qqplot gaussian
  - correlogram

- Model selection :
  - partial autocorrelation function : evaluates the benefit of adding lags
  - Bayesian information criterion :

$$BIC = T \log\left(\frac{SSE}{T}\right) + (k+2)\log(T)$$

T = # obs for the estimate
k = # model params

  - auto.arima(data)
- BOX-JENKINS
  1. identification : stationary? → transform, AR/MA ..?
  2. estimation : data → estimate params
  3. diagnostic
  4. decision : good model?
- Evaluation metrics :
  - Single error : $e_t = y_t - \hat{y}_t$
  - Mean Absolute Error : $MAE = \frac{1}{N}\sum |e_t|$
  - Root Mean Square Error : $RMSE = \sqrt{\frac{1}{N}\sum e_t^2}$
  - Mean Absolute percentage error :

$$MAPE = \frac{1}{N}\sum |p_t| \quad , \quad p_t = 100\,\frac{e_t}{y_t}$$