CS2220: Intro to Computational Biology

Knowledge Discovery Essence

Wong Limsoon

© Copyright National University of Singapore. All Rights Reserved.

Outline

What knowledge discovery is

Training data gathering

Feature generation

To be covered in later lectures when specific comp bio problems are discussed

Feature integration by supervised learning

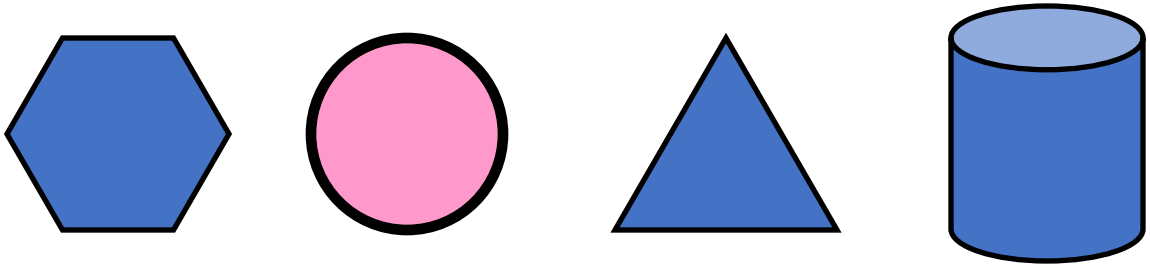
Classifier performance evaluation

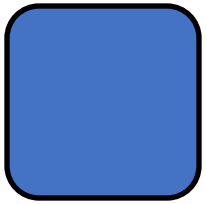
Feature selection

What knowledge discovery is

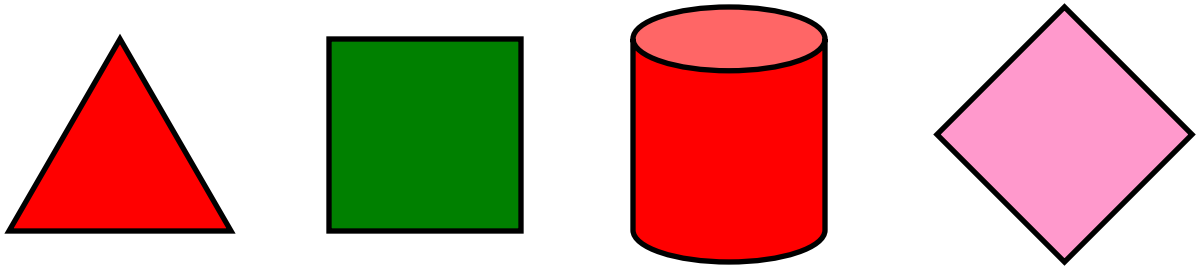
Two young children of a friend

Jonathan's blocks





Jessica's blocks



Whose block is this?

Jonathan's rules : Blue or Circle
Jessica's rules : All the rest

Going beyond “gut feeling”

Can you distinguish the men from the women?

Can you explain how?

Key steps of knowledge discovery

Training data gathering

Feature generation

k-grams, colour, texture, domain know-how, ...

Feature selection

Entropy, $\chi 2$, CFS, t-test, domain know-how...

Feature integration

SVM, ANN, PCL, CART, C4.5, kNN, ...

Training data gathering

Effective training datasets

Knowledge discovery begins with training data

Key requirements for effective training datasets

Relevant to the problem
Diverse & representative
Free from errors

} Highly domain specific

Other "good-to-have" properties

Large sample size
Few missing values
Balanced class distribution

Feature integration: Supervised learning

Supervised learning

A.k.a. classification

Learn from past experience, and use the learned knowledge to classify new data

Knowledge learned by intelligent algorithms

Data

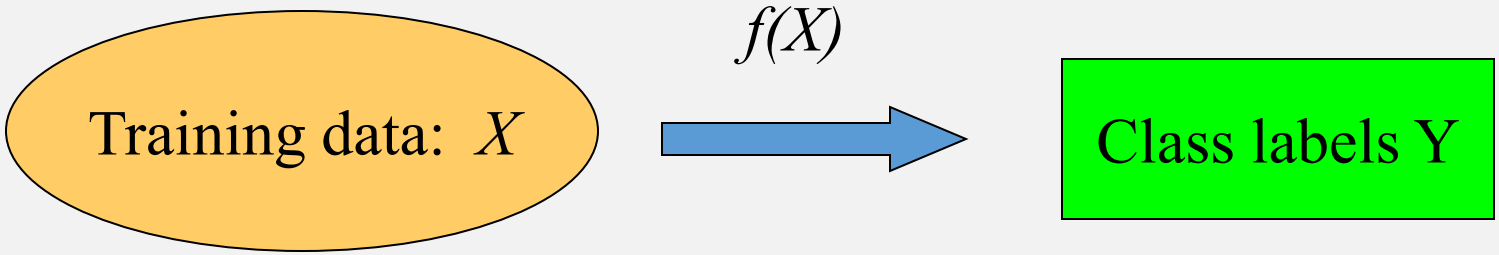
Classification application involves > 1 class of data

E.g., normal vs disease cells for a diagnosis problem

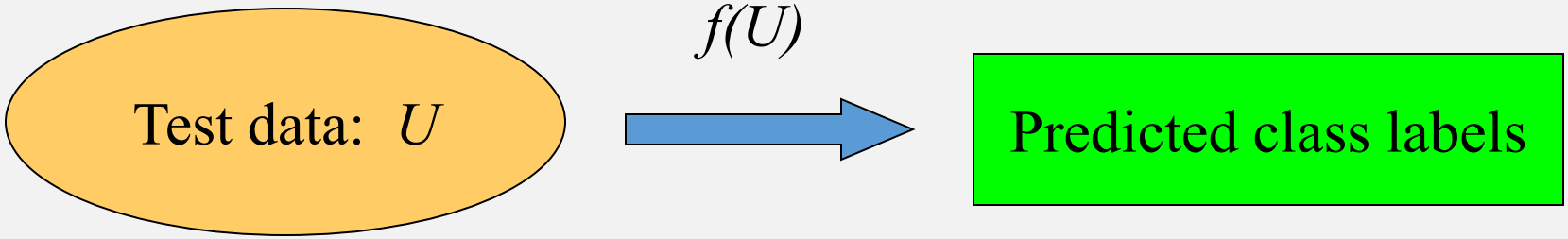
Training data is a set of instances (samples, points, etc.) with known class labels

Test data is a set of instances whose class labels are to be predicted

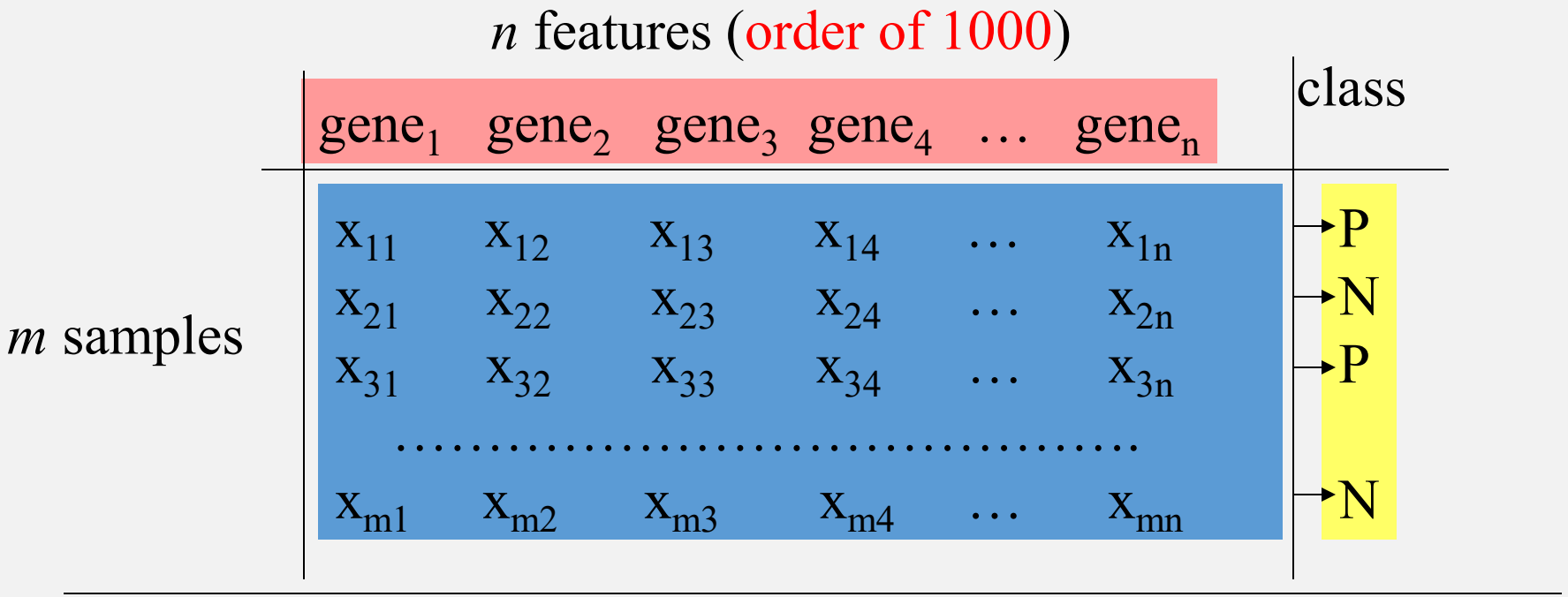
Process



A classifier, a mapping, a hypothesis



Relational representation of data



Features (a.k.a. attributes)

Categorical features

color = { red, blue, green }

Continuous or numerical features

gene expression

age

blood pressure

Discretization

age → { young, middle age, old }

Importance of rule-based methods

Systematic selection of a small number of features used for the decision making

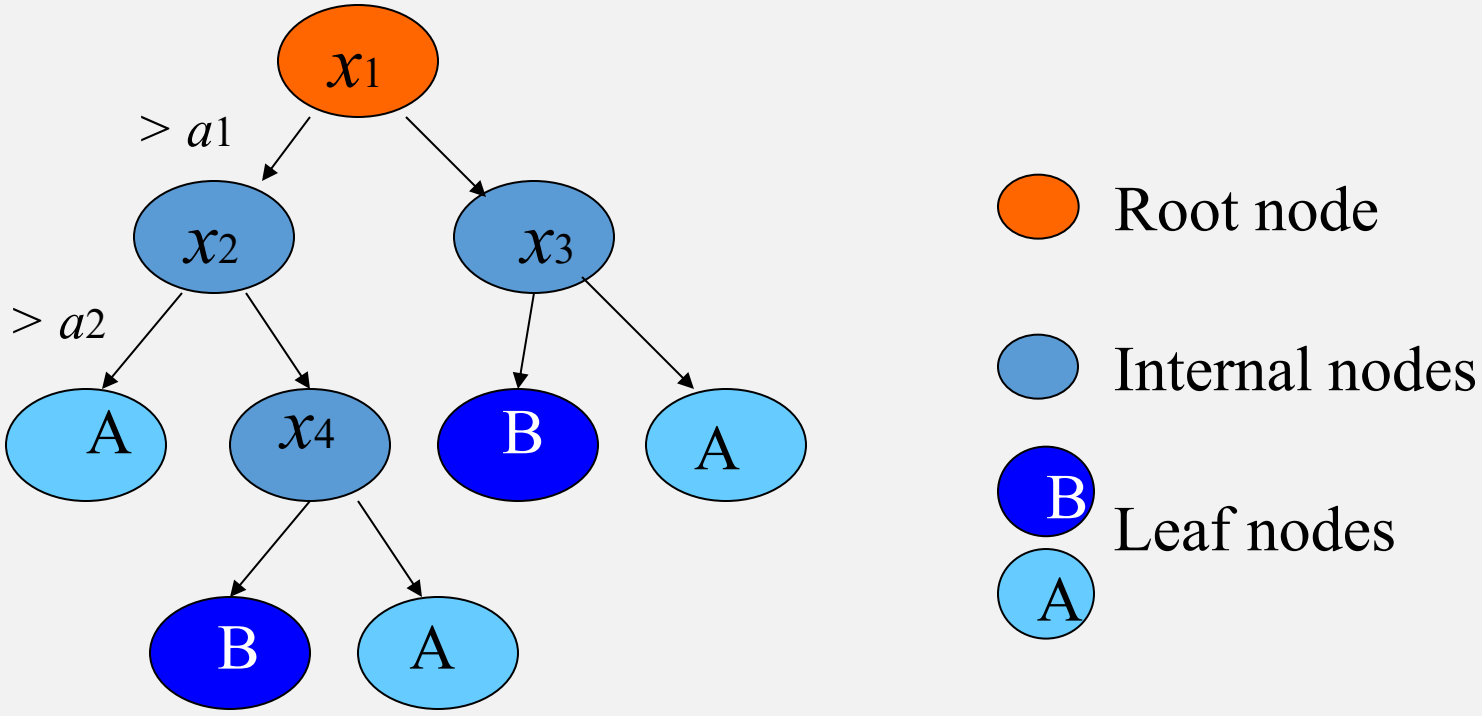
Increase comprehensibility of the knowledge patterns

C4.5 and CART are two commonly used rule induction algorithms---a.k.a. decision tree induction algorithms

Structure of decision trees

Every path from the root to a leaf forms a decision rule

E.g., If $x_1 > a_1$ & $x_2 > a_2$, then it's A class



A simple dataset

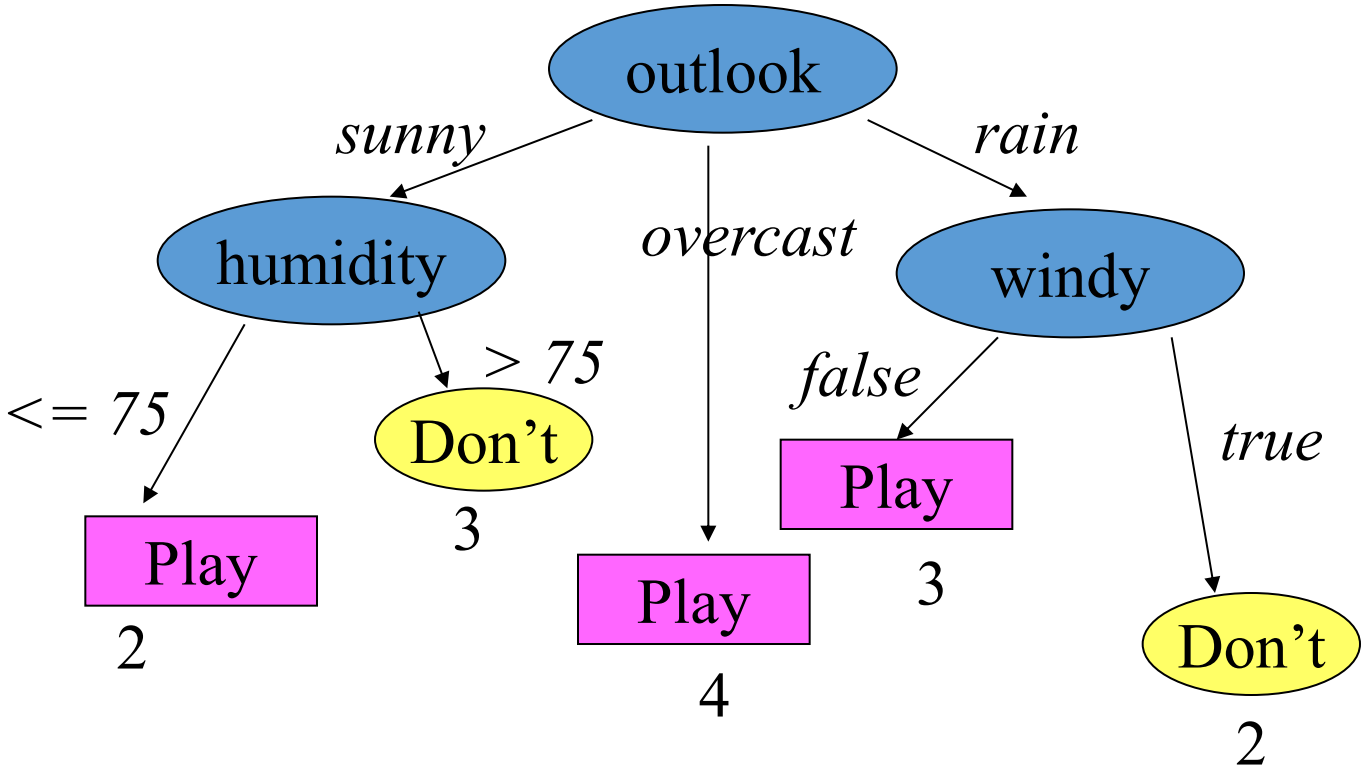
Outlook	Temp	Humidity	Windy	class
Sunny	75	70	true	Play
Sunny	80	90	true	Don't
Sunny	85	85	false	Don't
Sunny	72	95	true	Don't
Sunny	69	70	false	Play
Overcast	72	90	true	Play
Overcast	83	78	false	Play
Overcast	64	65	true	Play
Overcast	81	75	false	Play
Rain	71	80	true	Don't
Rain	65	70	true	Don't
Rain	75	80	false	Play
Rain	68	80	false	Play
Rain	70	96	false	Play

9 Play samples

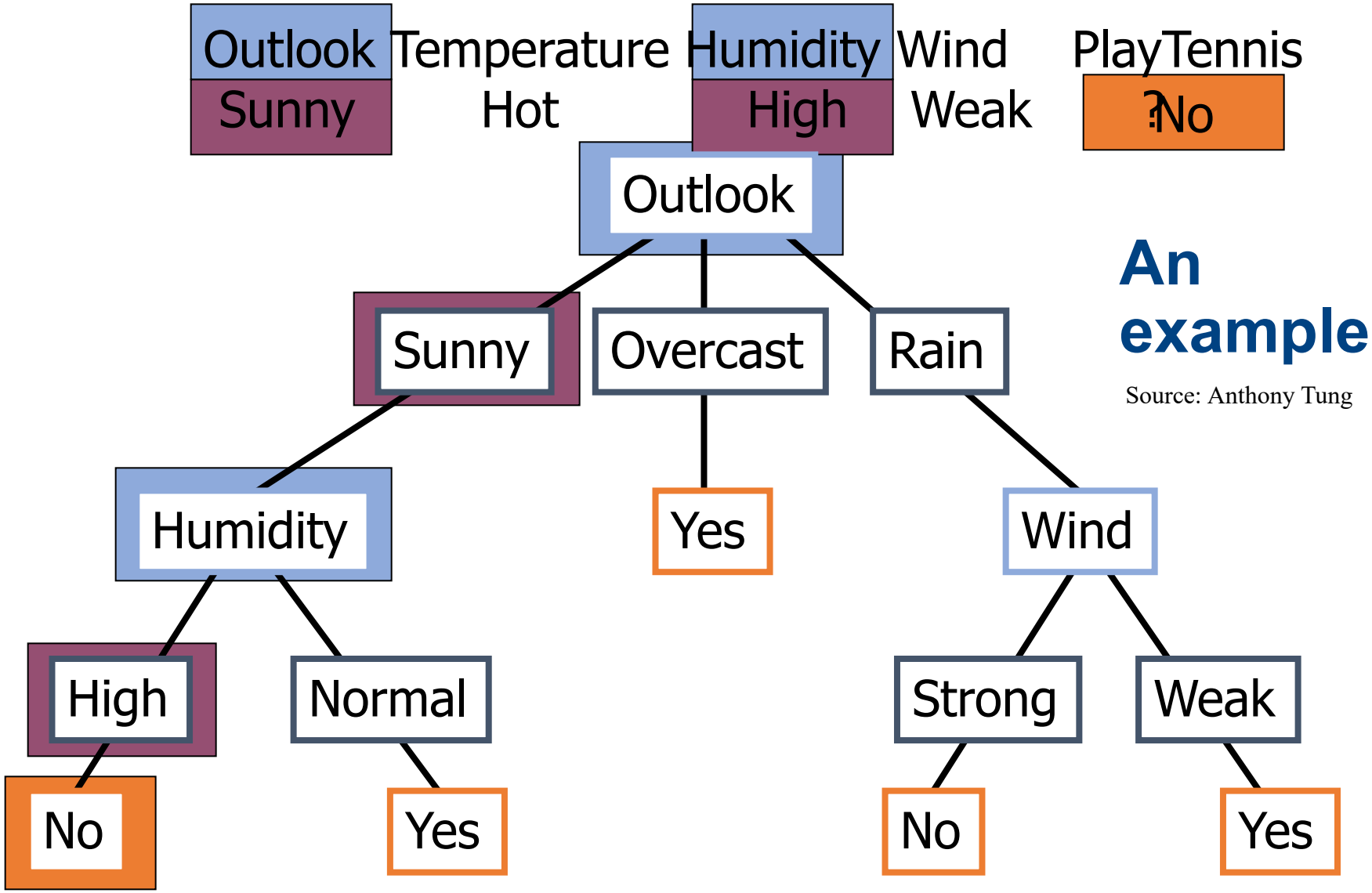
5 Don't

A total of 14.

A decision tree



Construction of a tree is equiv to determination of root node of the tree and root nodes of its sub-trees



An example

Source: Anthony Tung

Most discriminatory feature

Every feature can be used to partition the training data

If the partitions contain a pure class of training instances, then this feature is most discriminatory

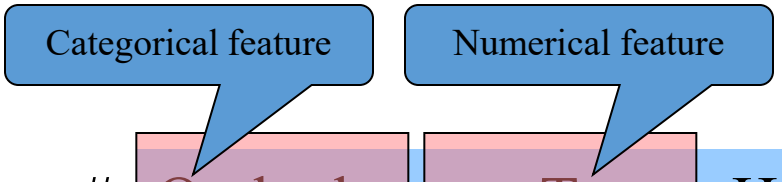
Example of partitions

Categorical feature

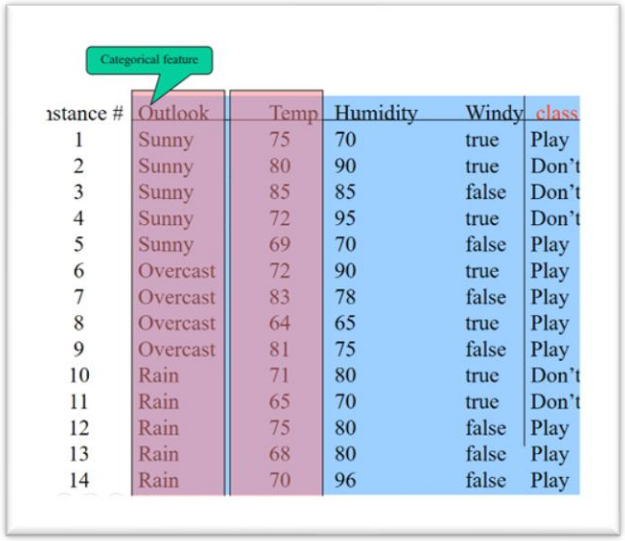
Number of partitions of the training data is equal to the number of values of this feature

Numerical feature

Two partitions



Instance #	Outlook	Temp	Humidity	Windy	class
1	Sunny	75	70	true	Play
2	Sunny	80	90	true	Don't
3	Sunny	85	85	false	Don't
4	Sunny	72	95	true	Don't
5	Sunny	69	70	false	Play
6	Overcast	72	90	true	Play
7	Overcast	83	78	false	Play
8	Overcast	64	65	true	Play
9	Overcast	81	75	false	Play
10	Rain	71	80	true	Don't
11	Rain	65	70	true	Don't
12	Rain	75	80	false	Play
13	Rain	68	80	false	Play
14	Rain	70	96	false	Play



ıstance #	Outlook	Temp	Humidity	Windy	class
1	Sunny	75	70	true	Play
2	Sunny	80	90	true	Don't
3	Sunny	85	85	false	Don't
4	Sunny	72	95	true	Don't
5	Sunny	69	70	false	Play
6	Overcast	72	90	true	Play
7	Overcast	83	78	false	Play
8	Overcast	64	65	true	Play
9	Overcast	81	75	false	Play
10	Rain	71	80	true	Don't
11	Rain	65	70	true	Don't
12	Rain	75	80	false	Play
13	Rain	68	80	false	Play
14	Rain	70	96	false	Play

Total 14 training instances

Outlook = sunny

1,2,3,4,5
P,D,D,D,P

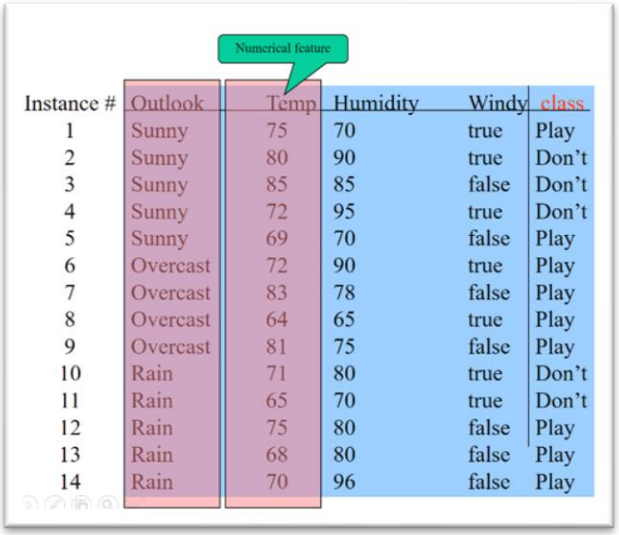
Outlook = overcast

6,7,8,9
P,P,P,P

Outlook = rain

10,11,12,13,14
D, D, P, P, P

A categorical feature is partitioned based on its number of possible values



Instance #	Outlook	Temp	Humidity	Windy	class
1	Sunny	75	70	true	Play
2	Sunny	80	90	true	Don't
3	Sunny	85	85	false	Don't
4	Sunny	72	95	true	Don't
5	Sunny	69	70	false	Play
6	Overcast	72	90	true	Play
7	Overcast	83	78	false	Play
8	Overcast	64	65	true	Play
9	Overcast	81	75	false	Play
10	Rain	71	80	true	Don't
11	Rain	65	70	true	Don't
12	Rain	75	80	false	Play
13	Rain	68	80	false	Play
14	Rain	70	96	false	Play

Total 14 training instances

Temperature <= 70

5,8,11,13,14
P,P, D, P, P

Temperature > 70

1,2,3,4,6,7,9,10,12
P,D,D,D,P,P,P,D,P

A numerical feature is generally partitioned by choosing a "cutting point"

Decision tree construction

Select “best” feature as root node of the whole tree

Partition dataset into subsets using this feature so that the subsets are as “pure” as possible

After partition by this feature, select the best feature (wrt the subset of training data) as root node of this sub-tree

Recursively, until the partitions become pure enough

Let's construct a decision tree

Outlook	Temp	Humidity	Windy	class
Sunny	75	70	true	Play
Sunny	80	90	true	Don't
Sunny	85	85	false	Don't
Sunny	72	95	true	Don't
Sunny	69	70	false	Play
Overcast	72	90	true	Play
Overcast	83	78	false	Play
Overcast	64	65	true	Play
Overcast	81	75	false	Play
Rain	71	80	true	Don't
Rain	65	70	true	Don't
Rain	75	80	false	Play
Rain	68	80	false	Play
Rain	70	96	false	Play

Ask the class to pick root node and construct the tree recursively with them… How good is that tree?

Which feature is “best”

Gini index

Information gain

Information gain ratio

Look up the last two up yourself

Gini index

Gini index is the expected value of ratio of the difference of two arbitrary specimens to the mean of all specimens

Closer to 1, feature is more "mixed" (not good)

Closer to 0, feature is more "pure" (good)

$$
\begin{aligned}
\text{gini}(S) &= \frac{\text{diff of two arbitrary specimen in } S}{2 \cdot \text{mean specimen in } S} \\
&= \text{prob(getting two specimen of diff class in } S) \\
&= 1 - \text{prob(getting two specimen of same class in } S) \\
&= 1 - \sum_i \text{prob(getting specimen of class } i \text{ in } S)^2
\end{aligned}
$$

Gini index

Let $\mathcal{U} = \{C_1, ..., C_k\}$ be all the classes. Suppose we are currently at a node and D is the set of those samples that have been moved to this node. Let f be a feature and $d[f]$ be the value of the feature f in a sample d. Let S be a range of values that the feature f can take. Then the Gini index for f in D for the range S is defined as

$$gini_f^D(S) = 1 - \sum_{C_i \in \mathcal{U}} \left(\frac{|\{d \in D \mid d \in C_i,\ d[f] \in S\}|}{|D|} \right)^2$$

The purity of a split of the value range S of an attribute f by some split-point into subranges S_1 and S_2 is then defined as

$$gini_f^D(S_1, S_2) = \sum_{S \in \{S_1, S_2\}} \frac{|\{d \in D \mid d[f] \in S\}|}{|D|} * gini_f^D(S)$$

we choose the feature f and the split-point p that minimizes $gini_f^D(S_1, S_2)$ over all possible alternative features and split-points.

Gini index of “Outlook”

Outlook	Temp	Humidity	Windy	class
Sunny	75	70	true	Play
Sunny	80	90	true	Don’t
Sunny	85	85	false	Don’t
Sunny	72	95	true	Don’t
Sunny	69	70	false	Play
Overcast	72	90	true	Play
Overcast	83	78	false	Play
Overcast	64	65	true	Play
Overcast	81	75	false	Play
Rain	71	80	true	Don’t
Rain	65	70	true	Don’t
Rain	75	80	false	Play
Rain	68	80	false	Play
Rain	70	96	false	Play

$$gini_f^D(S) = 1 - \sum_{C_i \in \mathcal{U}} \left(\frac{|\{d \in D \mid d \in C_i,\ d[f] \in S\}|}{|D|} \right)^2$$

$$gini_f^D(S_1, S_2) = \sum_{S \in \{S_1, S_2\}} \frac{|\{d \in D \mid d[f] \in S\}|}{|D|} * gini_f^D(S)$$

gini(Sunny) = 1 – $(2/5)^2$ – $(3/5)^2$ = 0.48

gini(Overcast) = 1 – $(4/4)^2$ – $(0/5)^2$ = 0

gini(Rain) = 1 – $(3/5)^2$ – $(2/5)^2$ = 0.48

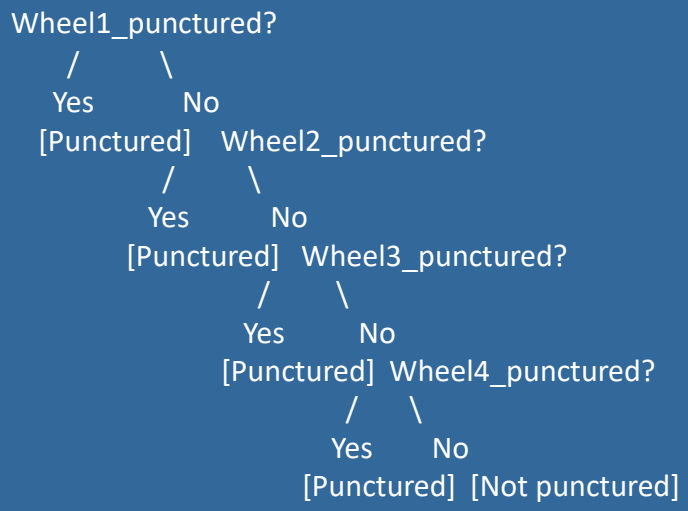
gini(Outlook) = 5/14 * 0.48 + 4/14 * 0 + 5/14 * 0.48 = 0.34

Exercise

How would you represent “car puncture if any of its wheel #1 to #4 is punctured” as a decision tree?

What do you think?

Irony



Loss of logical simplicity

Scattered logic

Difficulty spotting symmetry

Decision trees are considered interpretable models, but when modeling certain structures (e.g. disjunctions), they become less interpretable than a simple logical expression

Characteristics of C4.5/CART trees

Single coverage of training data (elegance)

Divide-and-conquer splitting strategy

Fragmentation problem

Locally reliable but globally insignificant rules

Miss many globally significant rules; mislead system

Decision tree ensembles

Motivating example

h_1, h_2, h_3 are classifiers w/ accuracy = 60%

C_1, C_2 are the only classes; t is a test instance in C_1

$h(t) = \text{argmax}_{C \in \{C1,C2\}} |\{h_j \in \{h_1, h_2, h_3\} \mid h_j(t) = C\}|$

Then prob($h(t) = C_1$)

$= \text{prob}(h_1(t)=C_1 \;\&\; h_2(t)=C_1 \;\&\; h_3(t)=C_1) +$
$\text{prob}(h_1(t)=C_1 \;\&\; h_2(t)=C_1 \;\&\; h_3(t)=C_2) +$
$\text{prob}(h_1(t)=C_1 \;\&\; h_2(t)=C_2 \;\&\; h_3(t)=C_1) +$
$\text{prob}(h_1(t)=C_2 \;\&\; h_2(t)=C_1 \;\&\; h_3(t)=C_1)$

$= 60\% * 60\% * 60\% + 60\% * 60\% * 40\% +$
$60\% * 40\% * 60\% + 40\% * 60\% * 60\%$

$= 64.8\%$

What if each doctor's success rate is 40%?

Exercise

Consider “1 man, 1 vote” as a political system

How do you know it is suitable for a given country?

What conditions are crucial for it to be successful besides the rule of law?

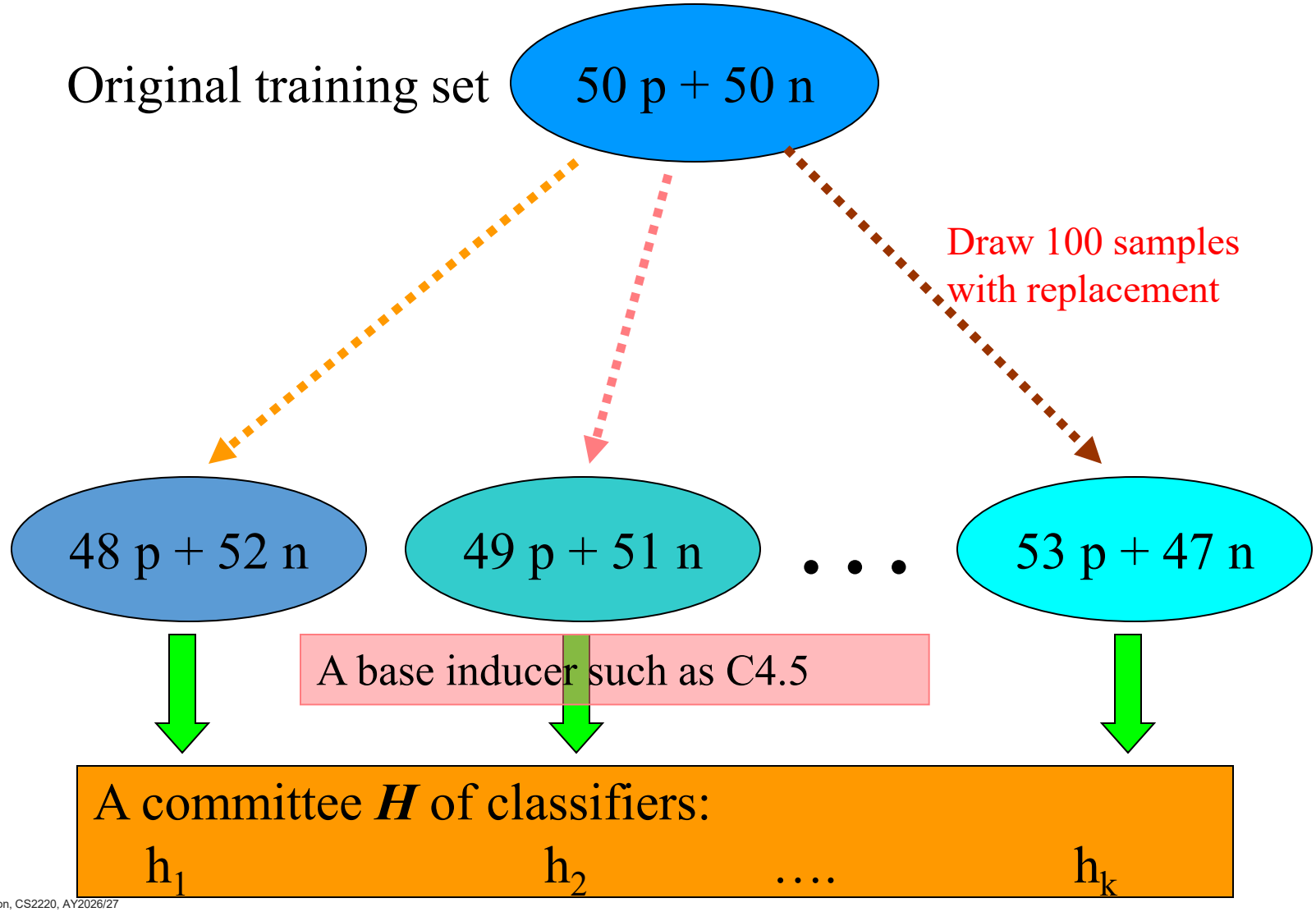
Bagging

Proposed by Breiman (1996)

Also called **B**ootstrap **agg**regat**ing**

Make use of randomness injected to training data

Main ideas


Original training set
50 p + 50 n
Draw 100 samples
with replacement
48 p + 52 n
49 p + 51 n
. . .
53 p + 47 n
A base inducer such as C4.5
A committee H of classifiers:
h1
h2
….
hk

Decision making by bagging

Given a new test sample T

$$bagged(T) = \text{argmax}_{C_j \in \mathcal{U}} |\{h_i \in \mathcal{H} \mid h_i(T) = C_j\}|$$

where $\mathcal{U} = \{C_1, \ldots, C_r\}$

K nearest neighbours, kNN

How kNN works

Given a new case

Find k “nearest” neighbours, i.e., k most similar points in the training data set

Assign new case to the same class to which most of these neighbours belong

A common “distance” measure between samples x and y is

$$\sqrt{\sum_f (x[f] - y[f])^2}$$

where f ranges over features of the samples

Illustration of kNN (k=8)

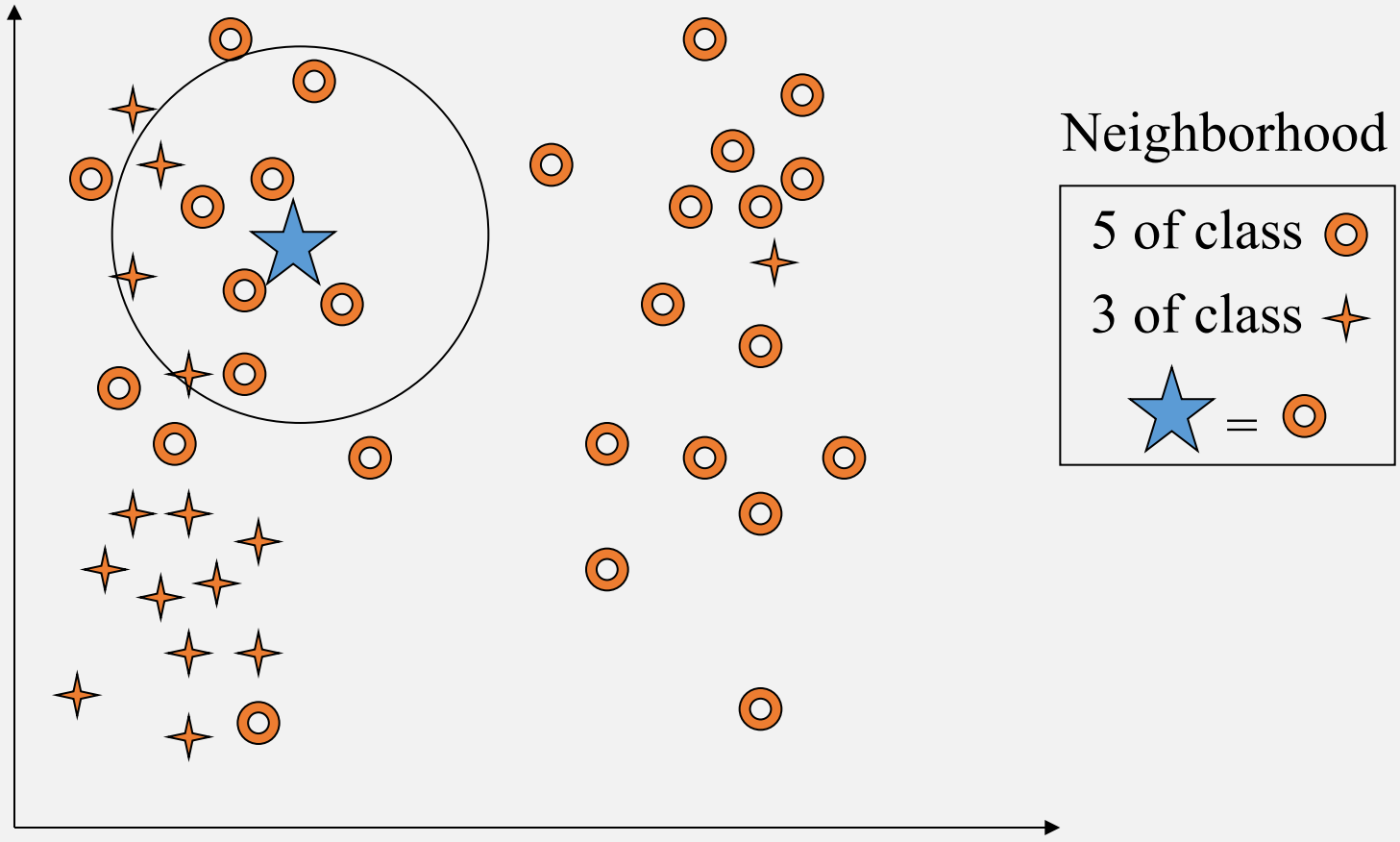


Image credit: Zaki

Some issues

Simple to implement

Must compare new case against all training cases

May be slow during prediction

No need to train

But need to design distance measure properly

Can’t explain prediction outcome

Can’t provide a model of the data

Naïve Bayes

Bayes theorem

$$P(h|d) = \frac{P(d|h) * P(h)}{P(d)}$$

P(h) = prior prob that hypothesis h holds

P(d|h) = prob of observing data d given h holds

P(h|d) = posterior prob that h holds given observed data d

Bayesian approach

Let H be all possible classes. Given a test instance w/ feature vector $\{f_1 = v_1, \ldots, f_n = v_n\}$, the most probable classification is given by

$$\text{argmax}_{h_j \in H} P(h_j | f_1 = v_1, \ldots, f_n = v_n)$$

Using Bayes Theorem, rewrites to

$$\text{argmax}_{h_j \in H} \frac{P(f_1 = v_1, \ldots, f_n = v_n | h_j) * P(h_j)}{P(f_1 = v_1, \ldots, f_n = v_n)}$$

Since denominator is independent of h_j, this simplifies to

$$\text{argmax}_{h_j \in H} P(f_1 = v_1, \ldots, f_n = v_n | h_j) * P(h_j)$$

Naïve Bayes

Estimating $P(f_1 = v_1, \ldots, f_n = v_n \mid h_j)$ accurately may not be feasible unless training data set is large

"Solved" by assuming $f_1, \ldots, f_n$ are conditionally independent of each other. Then,

$$\text{argmax}_{h_j \in H} P(f_1 = v_1, \ldots, f_n = v_n | h_j) * P(h_j)$$

$$= \text{argmax}_{h_j \in H} \prod_i P(f_i = v_i | h_j) * P(h_j)$$

where $P(h_j)$ and $P(f_i = v_i \mid h_j)$ can often be estimated reliably from typical training data set

Exercise: How do you estimate $P(h_j)$ and $P(f_j=v_j|h_j)$?

Exercise

Independence

$P(A,B) = P(A) \cdot P(B)$

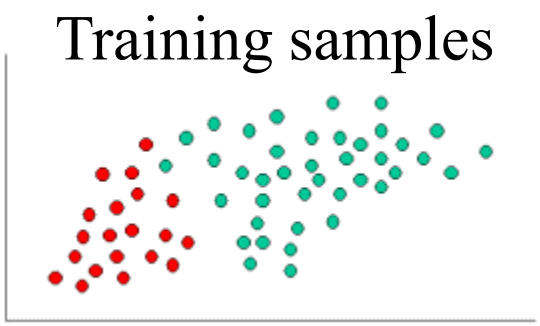
Conditional Independence

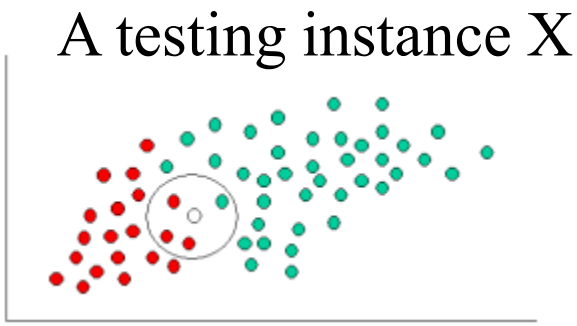
$P(A,B|C) = P(A|C) \cdot P(B|C)$

Does independence imply conditional independence?

Or the other way round?

Exercise



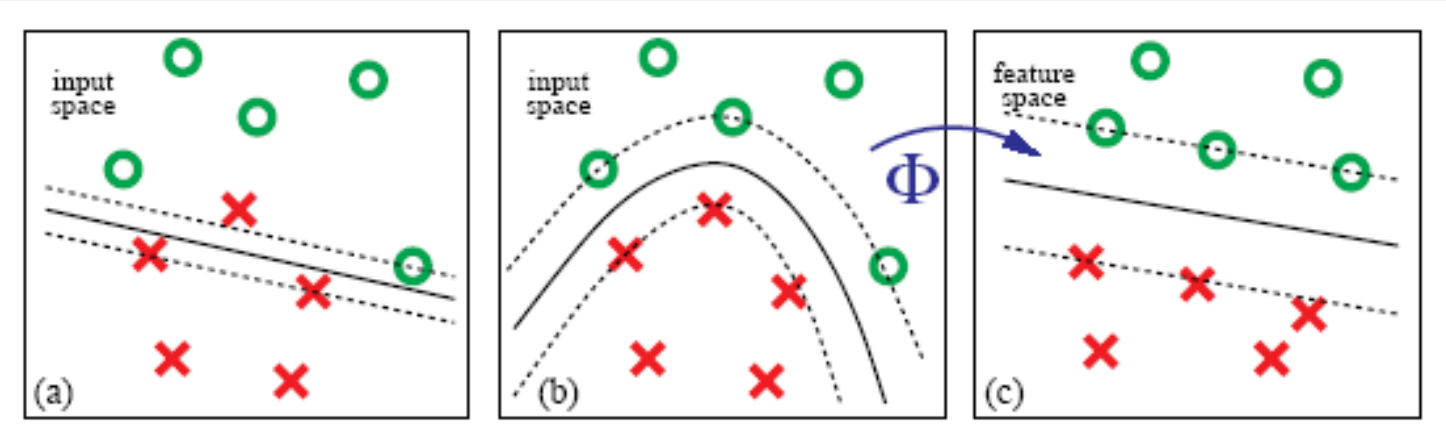


Should be X classified as red or green?

Support vector machines

Basic idea

Image credit: Zien



(a) Linear separation not possible w/o errors

(b) Better separation by nonlinear surfaces in input space

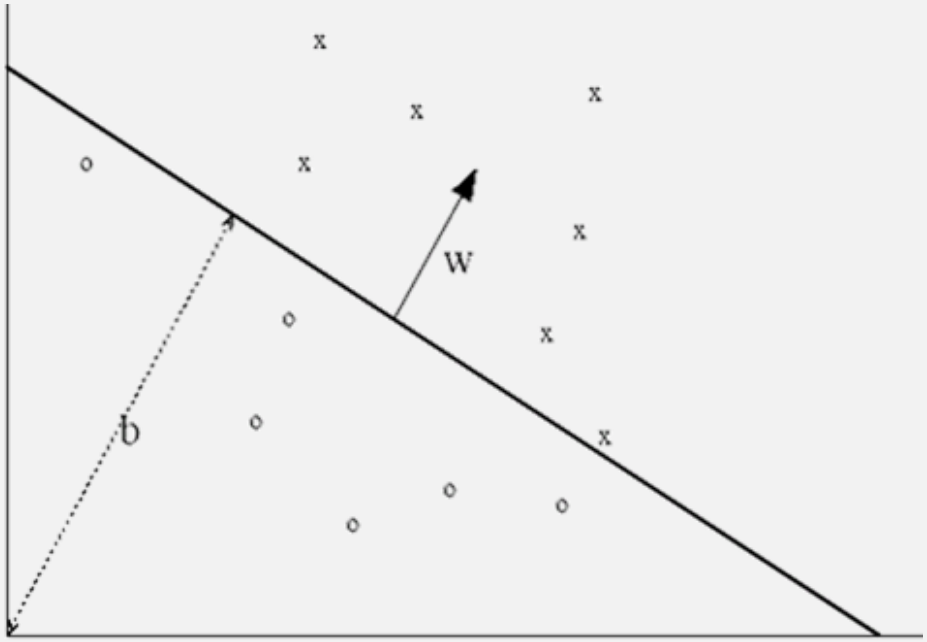
(c) Nonlinear surface corresponds to linear surface in feature space. Map from input to feature space by "kernel" function Φ

$\Rightarrow$"Linear learning machine" + kernel function as classifier

Linear learning machines

Hyperplane separating the x's and o's points is given by $(W \bullet X) + b = 0$, with $(W \bullet X) = \Sigma_j W[j] \cdot X[j]$

$\Rightarrow$ Decision function is $\mathrm{Ilm}(X) = \mathrm{sign}((W \bullet X) + b)$



Linear learning machines

Solution is a linear combination of training points X_k with labels Y_k

$$W = \Sigma_k \alpha_k \cdot Y_k \cdot X_k,$$

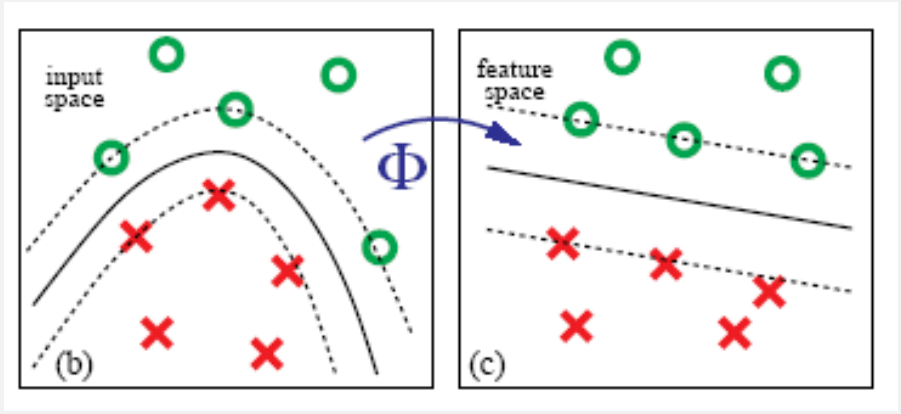
$$\text{with } \alpha_k > 0, \text{ and } Y_k = \pm 1$$

$$\Rightarrow \text{llm}(X) = \text{sign}(\Sigma_k \alpha_k \cdot Y_k \cdot (X_k \bullet X) + b)$$

"data" appears only in dot product!

Kernel function

$\text{Ilm}(X) = \text{sign}(\sum_k \alpha_k \cdot Y_k \cdot (X_k \bullet X) + b)$



$\text{svm}(X) = \text{sign}(\sum_k \alpha_k \cdot Y_k \cdot (\Phi X_k \bullet \Phi X) + b)$

$\Rightarrow \text{svm}(X) = \text{sign}(\sum_k \alpha_k \cdot Y_k \cdot K(X_k, X) + b)$

where $K(X_k, X) = (\Phi X_k \bullet \Phi X)$

Kernel function

svm(X) = sign($\Sigma_k \alpha_k \cdot Y_k \cdot K(X_k, X) + b$)

$\Rightarrow$ K(A,B) can be computed w/o computing Φ

In fact, replace it w/ lots of more "powerful" kernels besides (A • B):

$K(A,B) = (A \bullet B)^d$

$K(A,B) = exp(- || A\ B||^2 / 2\sigma)$, ...

How SVM works

$svm(X) = sign(\Sigma_k \alpha_k \cdot Y_k \cdot K(X_k, X) + b)$

To find α_k is a quadratic programming problem

max: $\Sigma_k \alpha_k - 0.5 \Sigma_k \Sigma_h \alpha_k \cdot \alpha_h \cdot Y_k \cdot Y_h \cdot K(X_k, X_h)$

subject to: $\Sigma_k \alpha_k \cdot Y_k = 0$

and for all α_k , $C \geq \alpha_k \geq 0$

To find b, estimate by averaging

$Y_h - \Sigma_k \alpha_k \cdot Y_k \cdot K(X_h, X_k)$

for all $\alpha_h \geq 0$

Classifier performance evaluation

Accuracy

	predicted as positive	predicted as negative
positive	TP	FN
negative	FP	TN

$$\text{Accuracy} = \frac{\text{No. of correct predictions}}{\text{No. of predictions}}$$

$$= \frac{TP + TN}{TP + TN + FP + FN}$$

Balanced population

classifier	TP	TN	FP	FN	Accuracy
A	25	25	25	25	50%
B	50	25	25	0	75%
C	25	50	0	25	75%
D	37	37	13	13	74%

Clearly, B, C, D are all better than A

Is B better than C, D?

Is C better than B, D?

Is D better than B, C?

Unbalanced population

classifier	TP	TN	FP	FN	Accuracy
A	25	75	75	25	50%
B	0	150	0	50	75%
C	50	0	150	0	25%
D	30	100	50	20	65%

Clearly, D is better than A

Is B better than A, C, D?

What might B's prediction strategy be?

Sensitivity, a.k.a. recall

	predicted as positive	predicted as negative
positive	TP	FN
negative	FP	TN

$$\text{Sensitivity}_{\text{wrt positives}} = \frac{\text{No. of correct positive predictions}}{\text{No. of positives}}$$

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

Sensitivity wrt negatives is termed specificity

Precision

	predicted as positive	predicted as negative
positive	TP	FN
negative	FP	TN

$$\text{Precision}_{\text{wrt positives}} = \frac{\text{No. of correct positive predictions}}{\text{No. of positives predictions}}$$

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

Exercise

classifier	TP	TN	FP	FN	Accuracy	Sensitivity	Precision
A	25	75	75	25	50%	50%	25%
B	0	150	0	50	75%		
C	50	0	150	0	25%		
D	30	100	50	20	65%	60%	38%

What are the sensitivity and precision of B and C?

Is B better than A, C, D?

Abstract model of a classifier

Given a test sample S

Compute scores p(S), n(S)

Predict S as negative if $p(S) / n(S) < t$

Predict S as positive if $p(S) / n(S) \geq t$

t is the decision threshold of the classifier

Changing t affects the performance of the classifier

Example

S	P(S)	N(S)	Actual Class	Predicted Class @ t = 3	Predicted Class @ t = 2
2	0.961252	0.038748	P	P	P
3	0.435302	0.564698	N	N	N
6	0.691596	0.308404	P	N	P
7	0.180885	0.819115	N	N	N
8	0.814909	0.185091	P	P	P
10	0.887220	0.112780	P	P	P
			accuracy	5 / 6	6/6
			recall	3 / 4	4/4
			precision	3 / 3	4/4

Recall that …

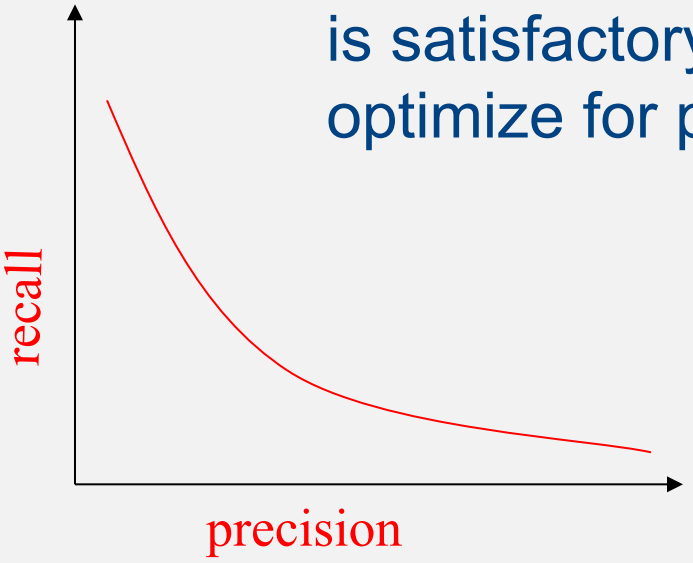
Predict *S* as negative if $p(S) / n(S) < t$

Predict *S* as positive if $p(S) / n(S) \geq t$

Precision-recall trade-off

A predicts better than B if A has better recall and precision than B

There is a trade-off between recall and precision



In some apps, once precision is satisfactory, you optimize for recall

In some apps, once recall is satisfactory recall, you optimize for precision

Comparing prediction performance

Accuracy

Convey the right intuition only when the positive and negative populations are roughly equal in size

Recall & precision together

What do you do when A has better recall than B and B has better precision than A?

Both alternatives are not good if the distribution of positives & negatives in the test set differs from real world

F-measure (Used in info extraction)

Take the harmonic mean of recall and precision

$$F = \frac{2 * \text{recall} * \text{precision}}{\text{recall} + \text{precision}} \quad \text{(wrt positives)}$$

classifier	TP	TN	FP	FN	Accuracy	F-measure
A	25	75	75	25	50%	33%
B	0	150	0	50	75%	undefined
C	50	0	150	0	25%	40%
D	30	100	50	20	65%	46%

Does not accord with intuition:
C predicts everything as +ve, but still rated better than A

Adjusted accuracy

Weigh by the importance of the classes

$$\text{Adjusted accuracy} = \alpha * \text{Sensitivity} + \beta * \text{Specificity}$$

$$\text{where } \alpha + \beta = 1$$

$$\text{typically, } \alpha = \beta = 0.5$$

classifier	TP	TN	FP	FN	Accuracy	Adj Accuracy
A	25	75	75	25	50%	50%
B	0	150	0	50	75%	50%
C	50	0	150	0	25%	50%
D	30	100	50	20	65%	63%

Geometric mean of sensitivity & specificity

Geometric mean = sqrt (Sensitivity * Specificity)

classifier	TP	TN	FP	FN	Accuracy	Geometric mean
A	25	75	75	25	50%	50%
B	0	150	0	50	75%	0%
C	50	0	150	0	25%	0%
D	30	100	50	20	65%	63%

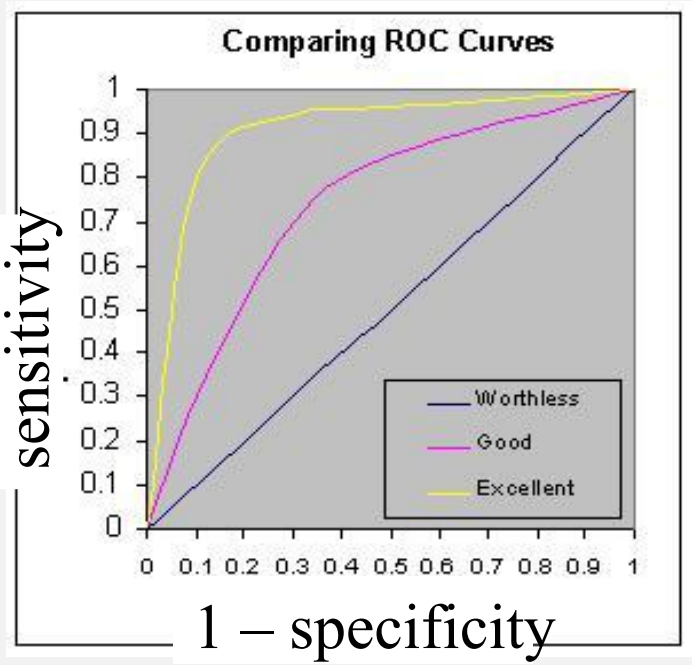
ROC curves

By changing t, we get a range of sensitivities and specificities of a classifier

A predicts better than B if A has better sensitivities than B at most specificities

Leads to ROC curve of sensitivity vs. (1 – specificity)

The larger the area under the ROC curve, the better



Exercise

You have a classifier. On a test set with 20% +ve & 80% -ve cases, its recall & precision are both 80%

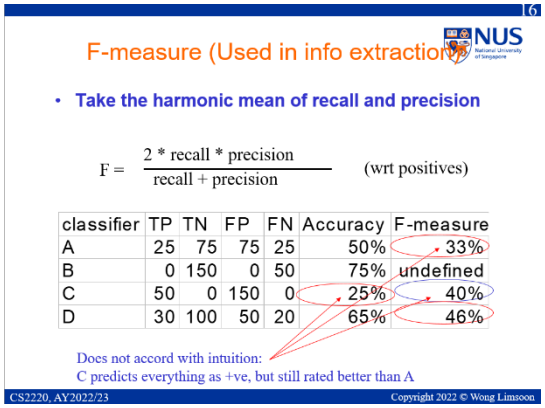
Suppose you test it on a new test set with 80% +ve & 20% -ve cases. What do you expect its accuracy to be?

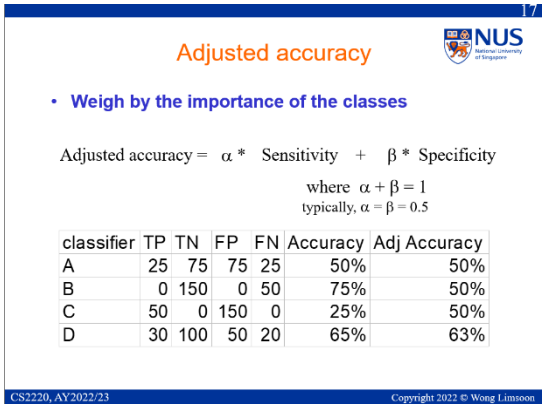
You may assume that the +ve (resp. –ve) instances in both test sets are sufficiently representative of the +ve (resp. –ve) real-world population

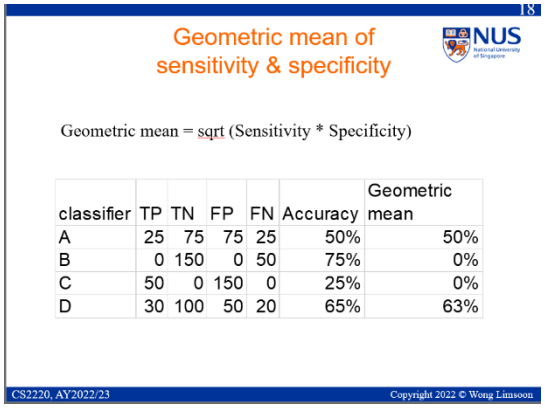
What lesson have you learned?

Exercise

Consider F-measure, adjusted accuracy, and geometric mean of sensitivity & specificity



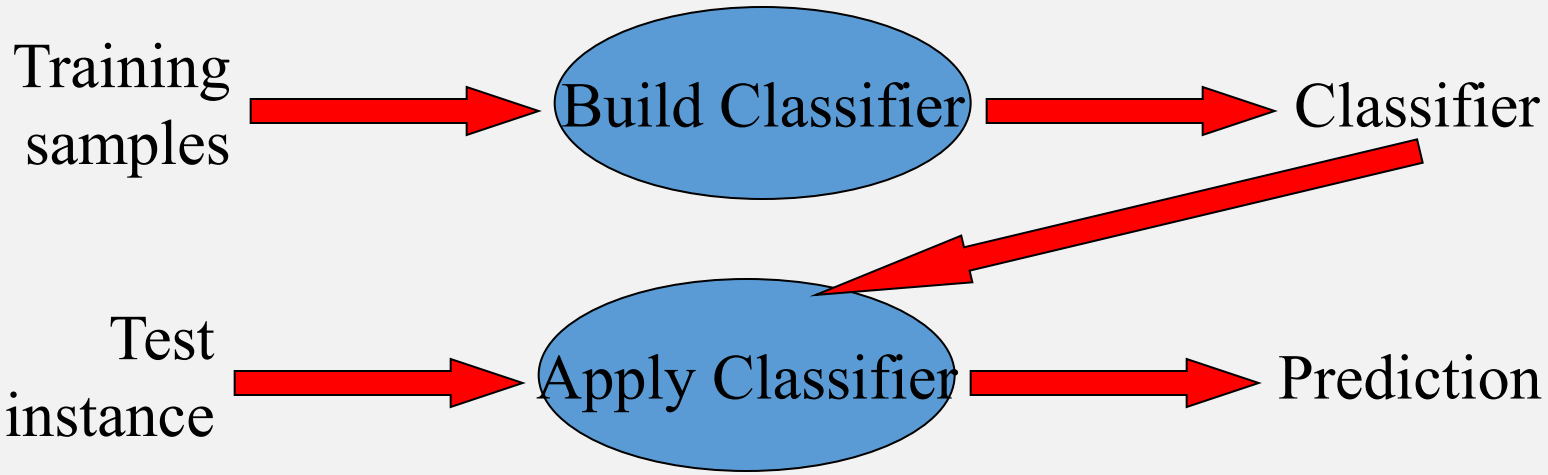




Which among these is likely the most robust measure of a classifier's performance?

Cross validation

Construction of a classifier



Recall kNN classifier

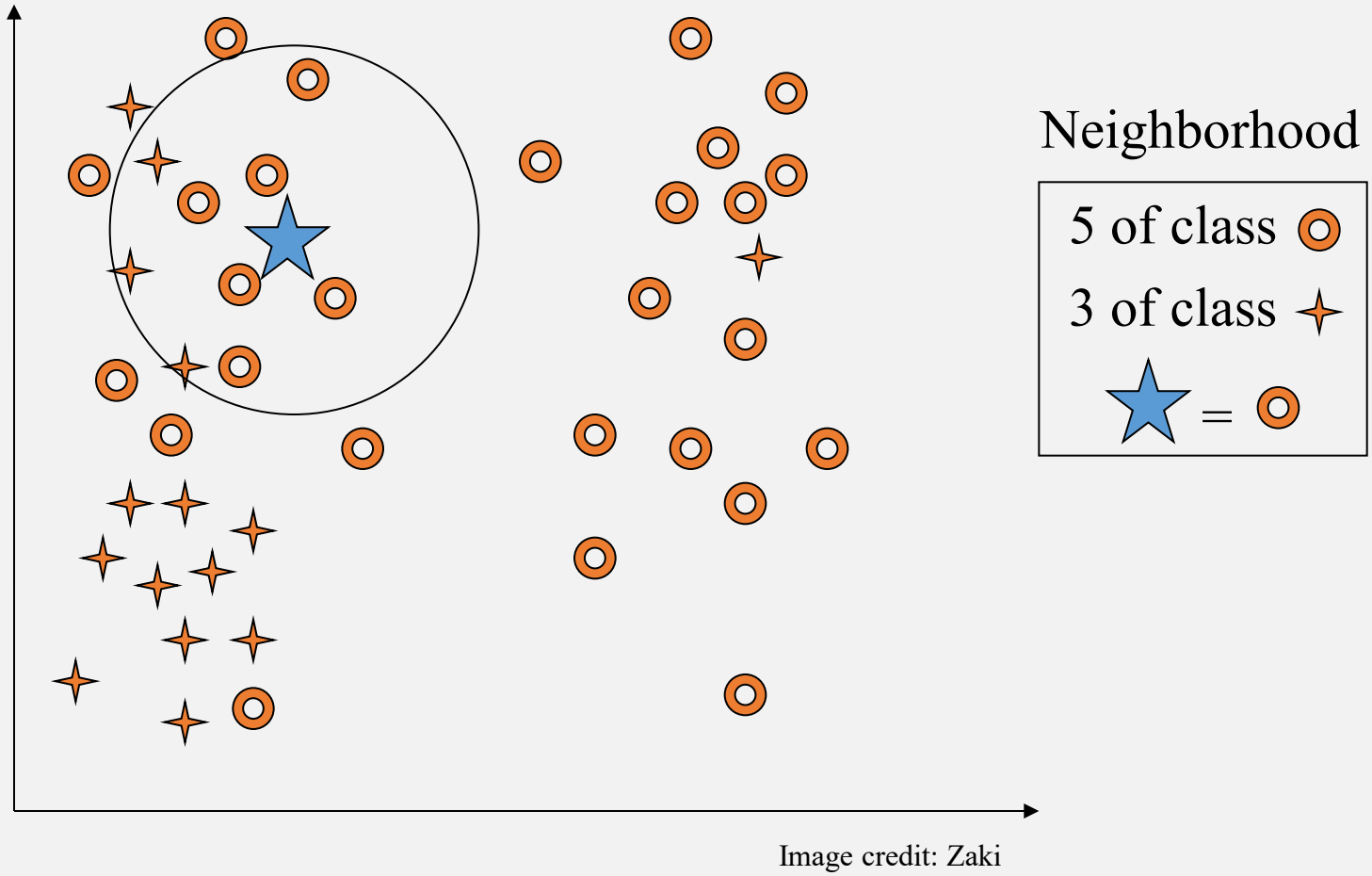
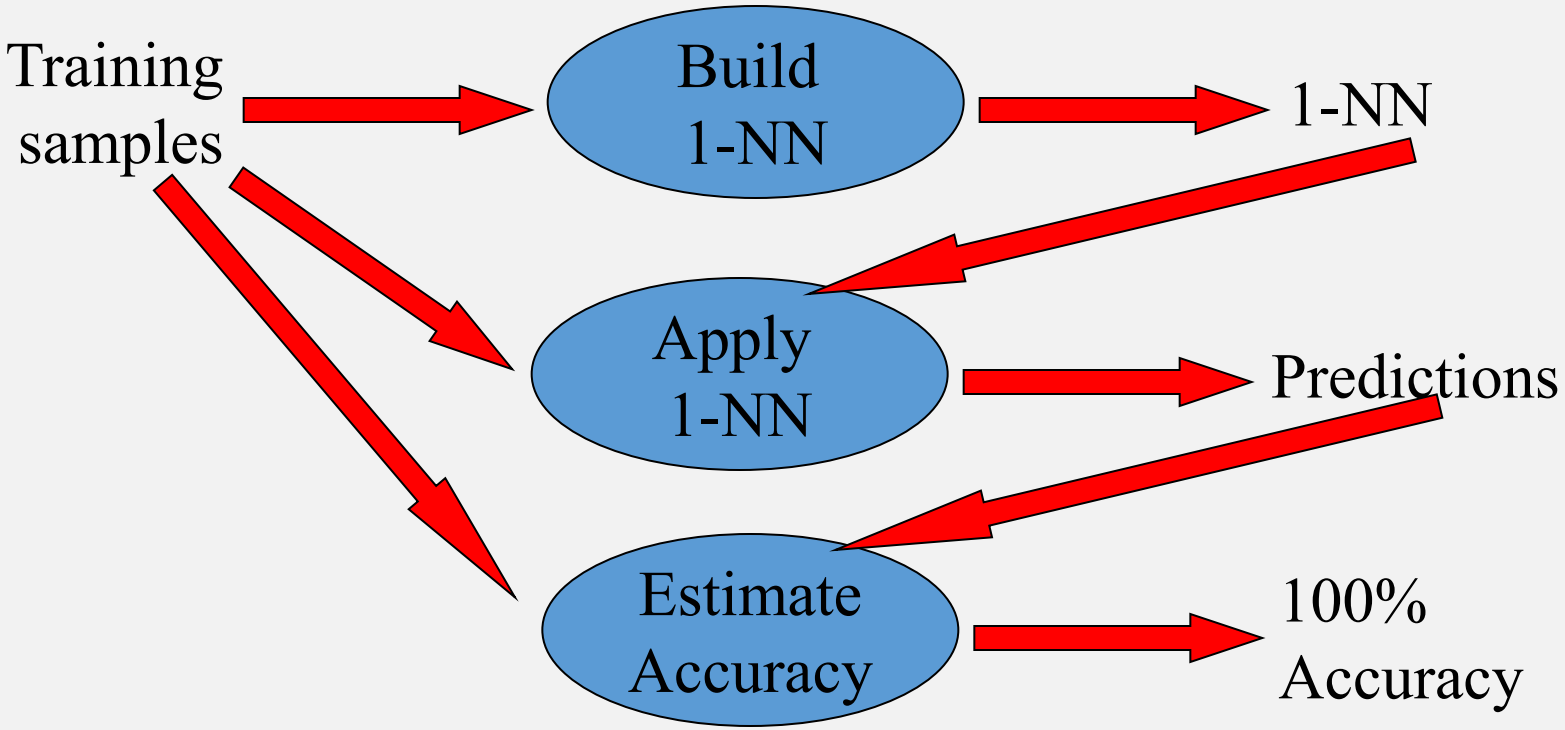


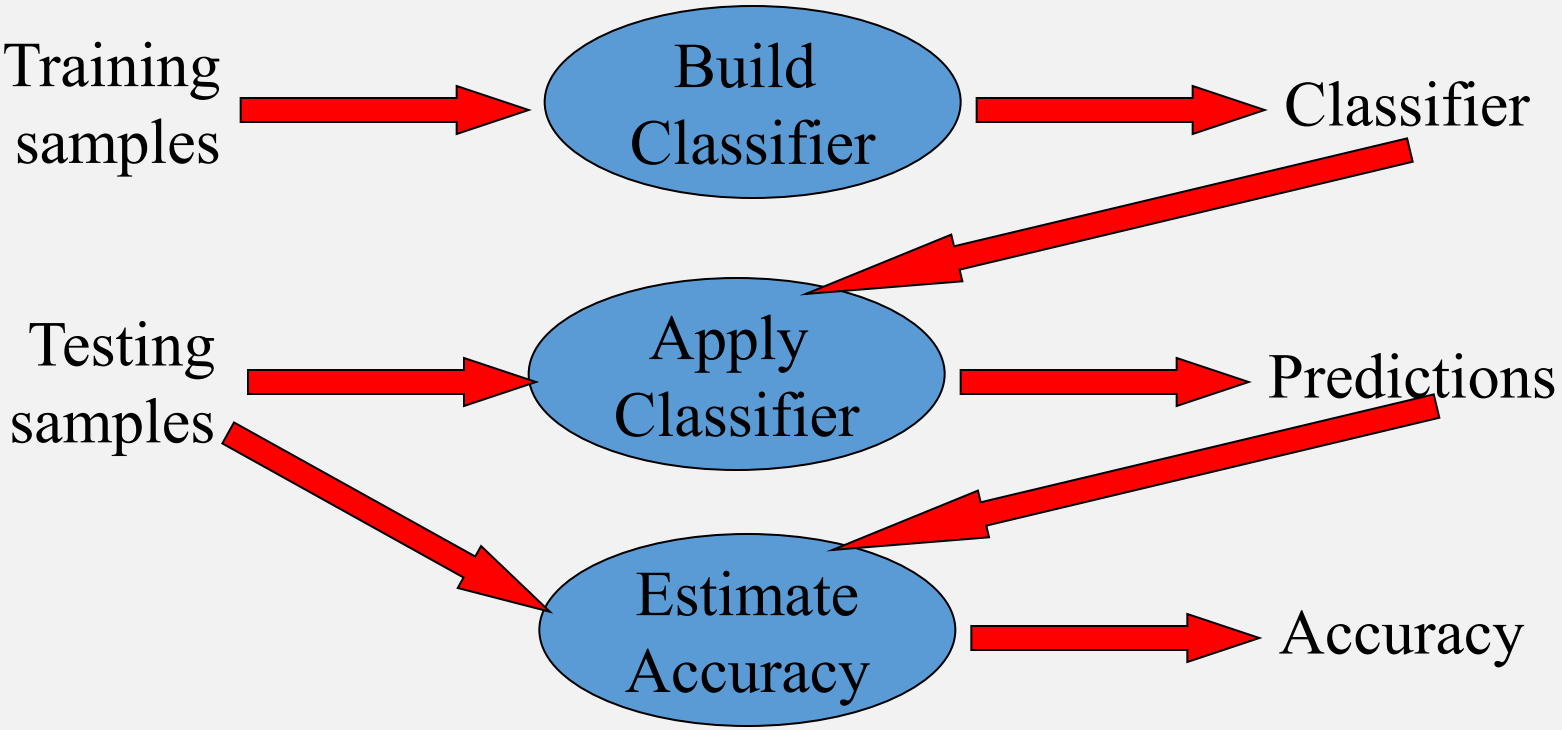
Image credit: Zaki

Estimate accuracy: Wrong way



k-NN (k = 1) has 100% accuracy in the "accuracy estimation" above. Does this accuracy generalize to new test instances?

Estimate accuracy: Right way



Testing samples are NOT to be used during "Build Classifier"

Exercise

The real-world population is highly imbalanced in terms of positive and negative instances

Is it a good idea to class-balance the training set?

Is it a good idea to class-balance the test set?

Cross validation

Divide samples into k roughly equal parts

Each part has similar proportion of samples from different classes

Use each part to test other parts

Average up accuracy

Exercise

What is the logical basis of cross validation?

Hint: Central limit theorem

What / whose accuracy does it really estimate?

Can it be a substitute for independent testing using new datasets?

Exercise

Coudray et al. report that common mutations in lung cancers can be predicted from histopath images using deep learning

Is this claim sound based purely on their results?

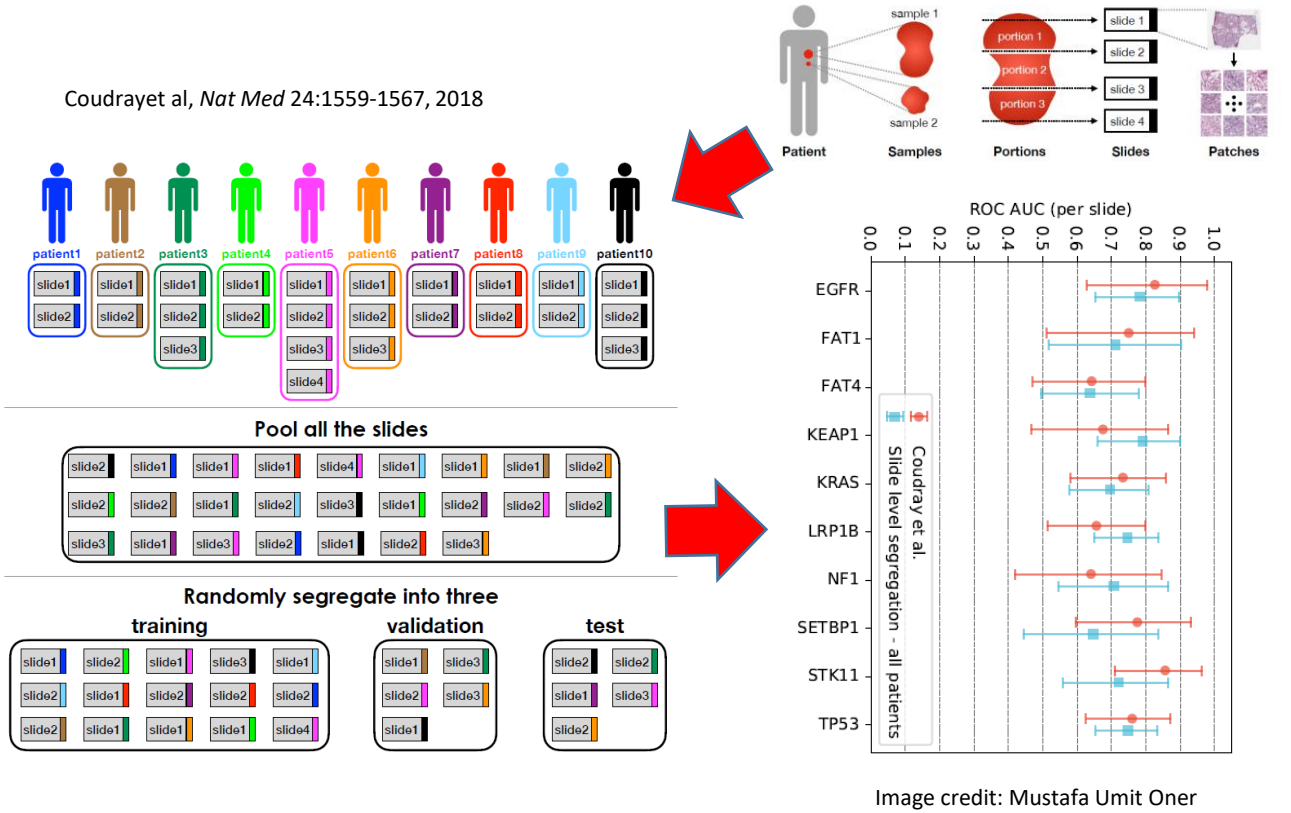
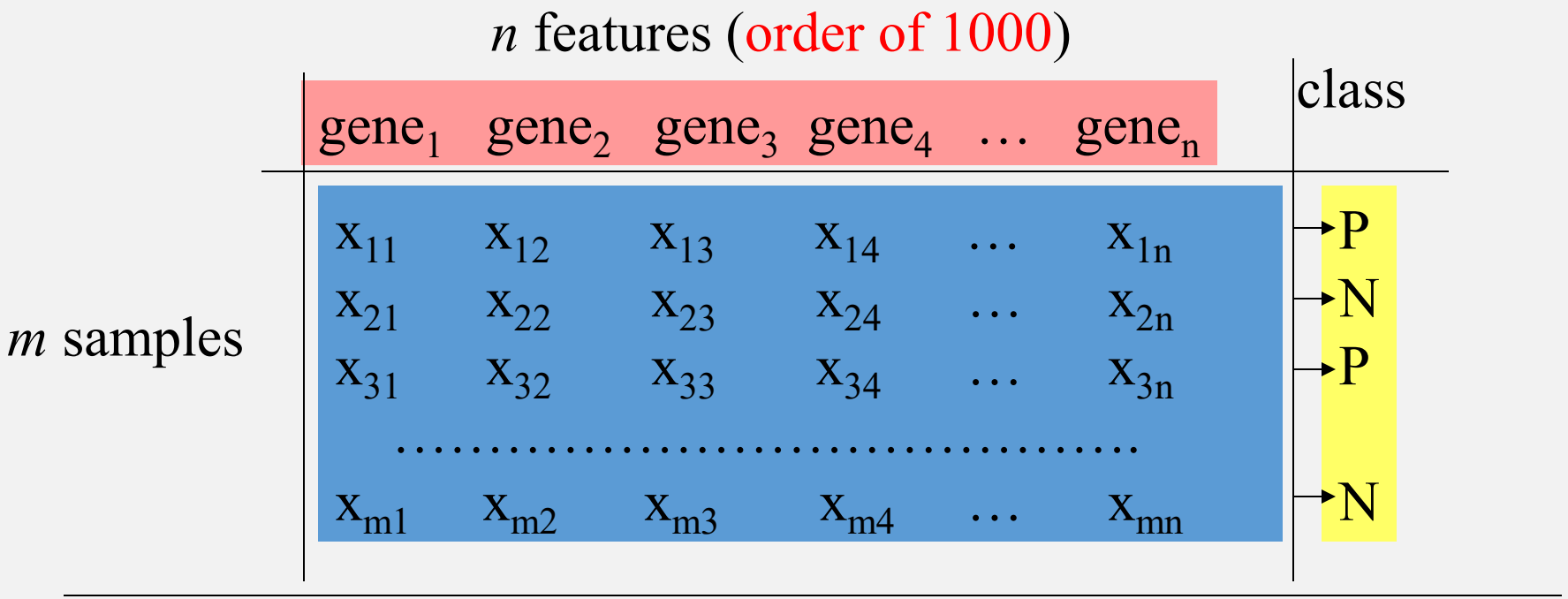


Image credit: Mustafa Umit Oner

Feature selection

Feature space

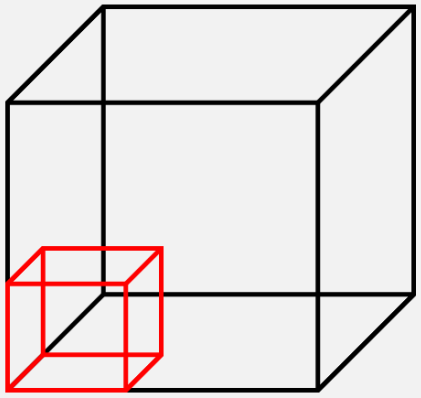


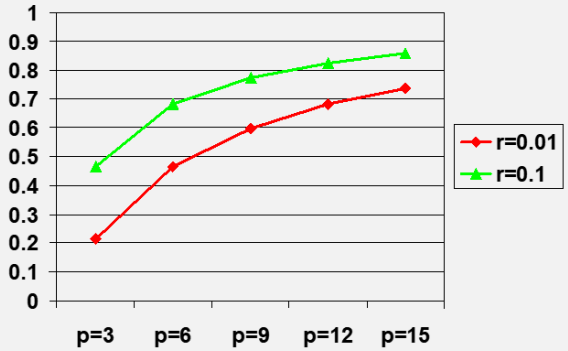
Curse of dimensionality

How much of each dimension is needed to cover a proportion r of a p-dimensional sample space?

Calculate by $e_p(r) = r^{1/p}$. Why?

So, to cover 10% of a 15-D space, need 85% of each dimension!





Hard to cover high-dimensional feature space well

Many features are irrelevant or misleading any way

Differences between samples become meaningless

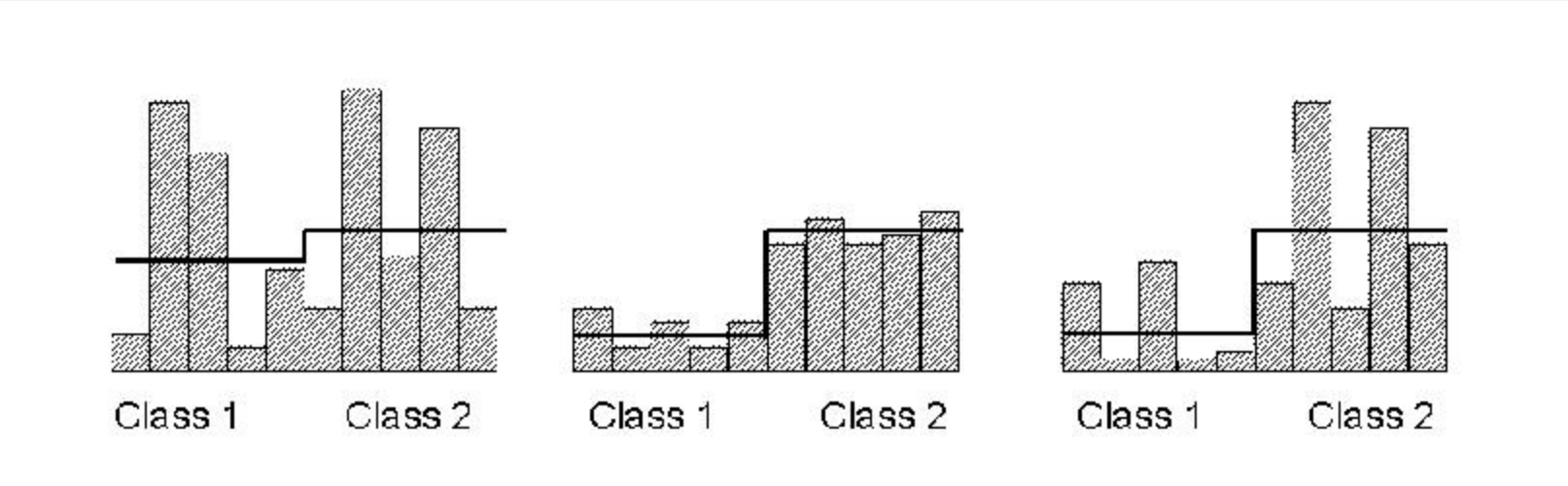
Tackling the curse

Given a feature space of p dimensions

It is possible that some dimensions are irrelevant

Find ways to separate those dimensions (a.k.a. features) that are relevant (a.k.a. signals) from those that are irrelevant (a.k.a. noise)

Signal selection (Basic idea)



Choose a feature w/ low intra-class distance

Choose a feature w/ high inter-class distance

Signal selection (e.g., t-statistics)

The t-stats of a signal is defined as

$$t = \frac{|\mu_1 - \mu_2|}{\sqrt{(\sigma_1^2/n_1) + (\sigma_2^2/n_2)}}$$

where σ_i^2 is the variance of that signal in class i, μ_i is the mean of that signal in class i, and n_i is the size of class i.

Exercise

How is the t-statistic typically used?

What are the assumptions required for this way of using the t-statistic?

Self-fulfilling oracle

Construct an artificial dataset with 100 samples, each with 100,000 randomly generated features and randomly assigned class labels

Select 20 features with the best t-statistics (or other methods)

Evaluate accuracy by cross validation using the 20 selected features

The resulting accuracy can be ~90%

But the true accuracy should be 50%, as the data were derived randomly

Exercise

What went wrong?

Self-fulfilling oracle

Construct artificial dataset with 100 samples, each with 100,000 randomly generated features and randomly assigned class labels

Select 20 features with the best t-statistics (or other methods)

Evaluate accuracy by cross validation using the 20 selected features

The resulting accuracy can be ~90%

But the true accuracy should be 50%, as the data were derived randomly

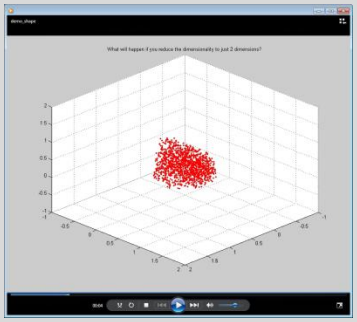
© Eamonn Keogh

While **dimensionality reduction** is an important tool in machine learning/data mining, we must always be aware that it can distort the data in misleading ways.

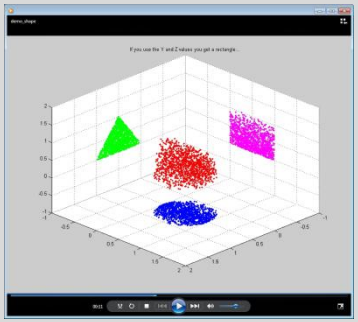
Above is a two-dimensional projection of an intrinsically three-dimensional world….

© Eamonn Keogh

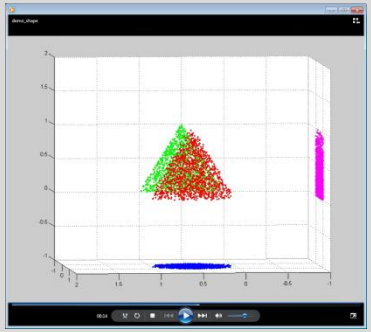
A cloud of points in 3D



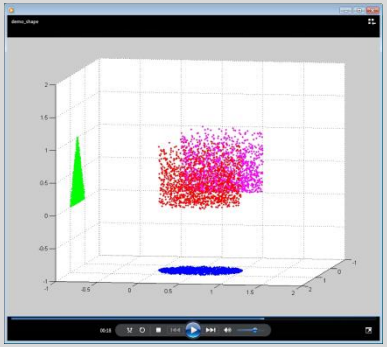
Can be projected into 2D
XY or XZ or YZ



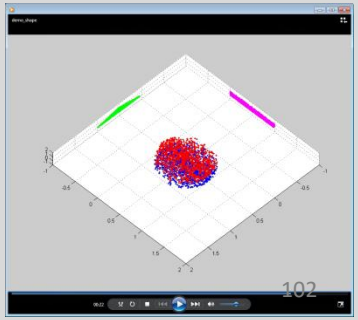
In 2D XZ we see
a triangle



In 2D YZ we see
a square



In 2D XY we see
a circle



Screen dumps of a short video from
www.cs.gmu.edu/~jessica/DimReducDanger.htm

Concluding remarks

What we have learned

Methodologies of knowledge discovery

Feature generation, feature selection, feature integration

Evaluation of classifiers

Accuracy, sensitivity, specificity, precision

Independent test set, cross validation

Curse of dimensionality

Feature selection

Self-fulfilling oracle

http://www.cs.waikato.ac.nz/ml/weka

Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization

Acknowledgements

The “young children” and “men vs women” slides were given to WLS by Tan Ah Hwee

The three slides on the dangers of dimensionality reduction were created by Eamonn Keogh

Most of the slides used in this ppt came from a tutorial that WLS gave with Jinyan Li at the PKDD 2004

The “independent vs conditional independent” example came from Choi Kwok Pui

Good to read

Hastie et al., *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer, 2001. Chapters 1, 7

Li et al., Data mining techniques for the practical bioinformatician, *The Practical Bioinformatician*, Chapter 3, pages 35-70, WSPC, 2004
https://www.comp.nus.edu.sg/~wongls/psZ/practical-bioinformatician/ch3-wlsdatamining/ch3-wlsdatamining.pdf

Ho et al. Avoid oversimplifications in machine learning: Going beyond the class-prediction accuracy. *Patterns*, 1(2):100025, 2020

Ho et al. Extensions of the external validation for checking learned model interpretability and generalizability. *Patterns*, 1(8):100129, 2020

Good to read

L. Breiman, et al. *Classification and Regression Trees*. Wadsworth and Brooks, 1984

L. Breiman, Bagging predictors, *Machine Learning*, 24:123-140, 1996

J. R. Quinlan, Induction of decision trees, *Machine Learning*, 1:81-106, 1986

C. Gini, Measurement of inequality of incomes, *The Economic Journal*, 31:124-126, 1921