ATTRIBUTE SELECTION
ATTRIBUTE TRANSFORMATION

M.L. PIPELINE / METHODOLOGY:

- Construct the data matrix
- Training:
  - Hyper-parameter tuning
  - Final model training
- Model evaluation (e.g. train/test or crossvalidation)
- Use of the model

?

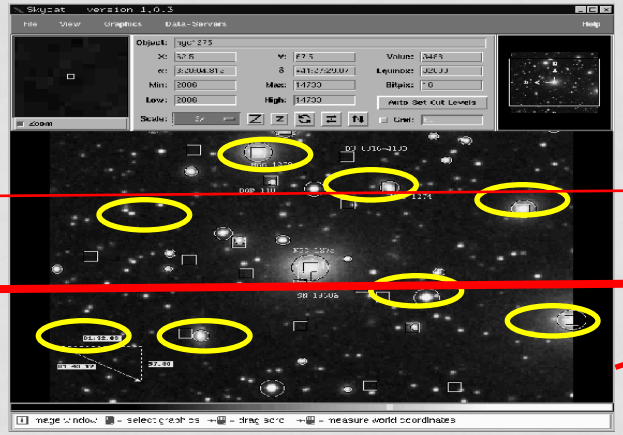
Use of the model

Available data

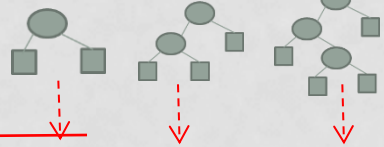
Training

Validation

Test

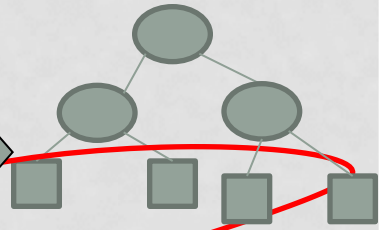


Hyper-parameter tuning



60% 92% 70%

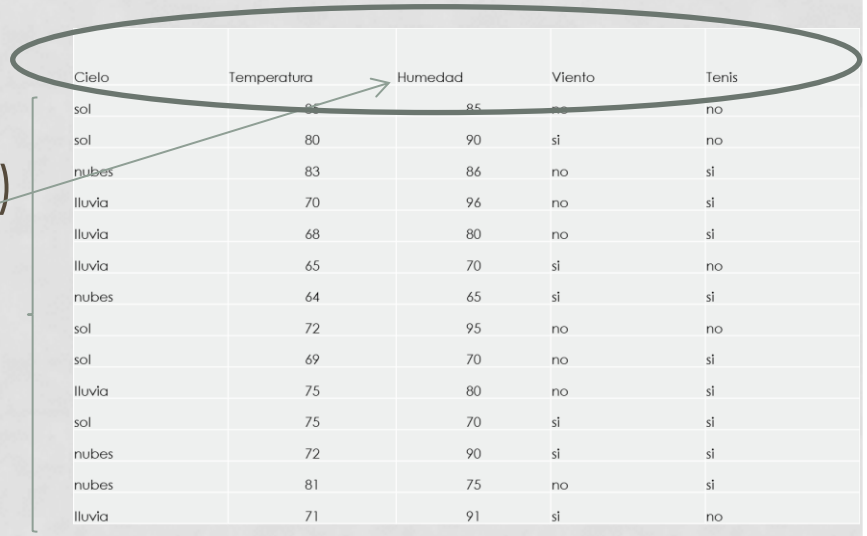
E and V
And depth=2



Evaluation 83%

Constructing the data matrix

- Constructing the data matrix:
  - Get raw data
  - Feature extraction if necessary (e.g. bag of words for text mining)
  - **Preprocessing:**
    - **Instances**
    - **Attributes**



Cielo	Temperatura	Humedad	Viento	Tenis
sol	85	85	no	no
sol	80	90	si	no
nubes	83	86	no	si
lluvia	70	96	no	si
lluvia	68	80	no	si
lluvia	65	70	si	no
nubes	64	65	si	si
sol	72	95	no	no
sol	69	70	no	si
lluvia	75	80	no	si
sol	75	70	si	si
nubes	72	90	si	si
nubes	81	75	no	si
lluvia	71	91	si	no

PREPROCESSING

- Instances:
  - Remove outliers (strange instances)
  - Sampling (in order to have a smaller but representative training dataset)
  - Sampling in order to balance classes in imbalanced problems
- Attributes:
  - Normalization
  - Dummy variables, one-hot encoding: for some algorithms, every categorical / discrete variable must be transformed into several binary variables
  - Imputation (what to do with missing values?)
  - **Attribute selection**
  - **Attribute transformation**

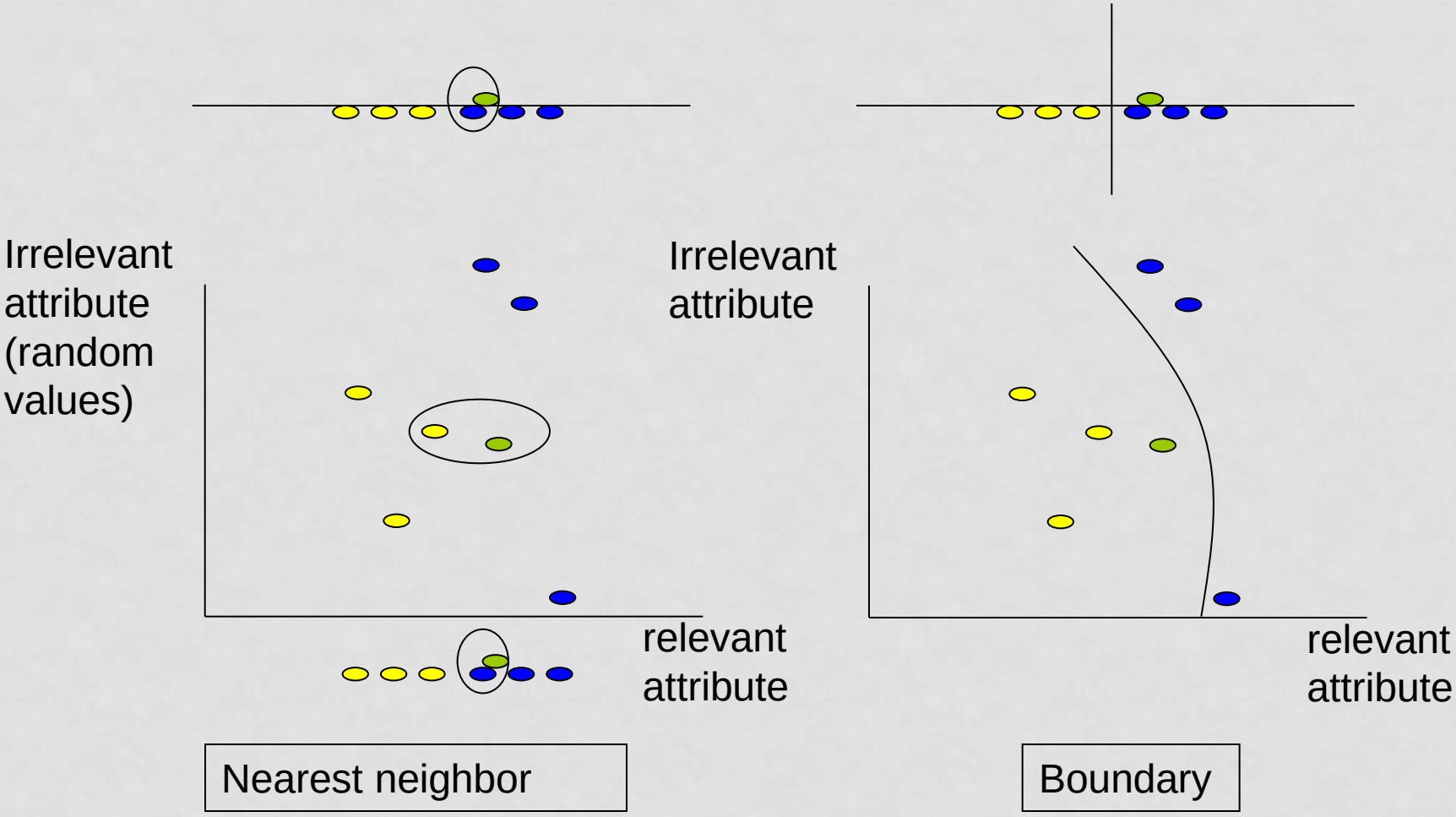
Attribute selection: motivation

- Some attributes can be **redundant** to some extent (such as "salary" and "social class")
- Some attributes can be **irrelevant** (such as "eye color" in order to predict payment of a loan)
- **Curse of dimensionality**: The number of required instances for learning can grow exponentiallly with the number of dimensions

- Learning is slower if there are many attributes (e.g.: C4.5 is $O(m*n^2)$ SVM is $O(m*n)$)
- Some classifiers can get confused by redundant attributes (e.g. Naive Bayes and KNN)
- Having too many attributes (specially irrelevant ones) may result in **overfitting**, because it provides the model with extra arbitrary degrees of freedom to fit the data (i.e. it increases the complexity of the model)

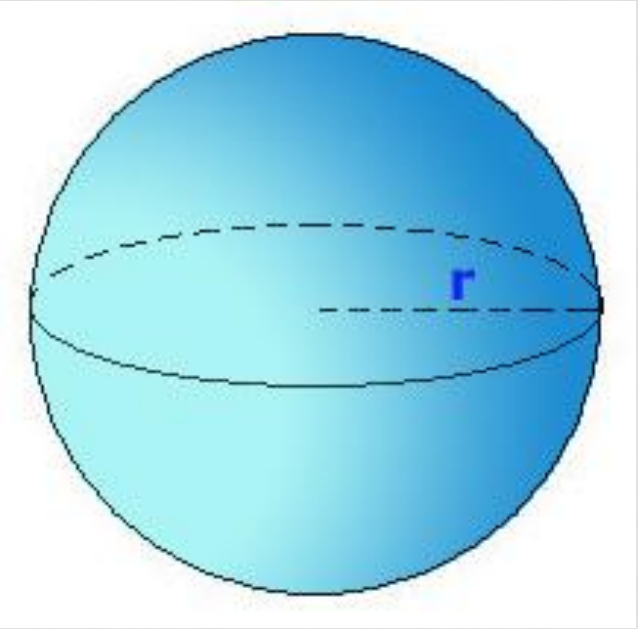
- Sometimes it is useful to know which attributes are relevant (e.g. which genes are able to predict cancer?)
- The fewer attributes, the easier is to interpret the model

Irrelevant attributes



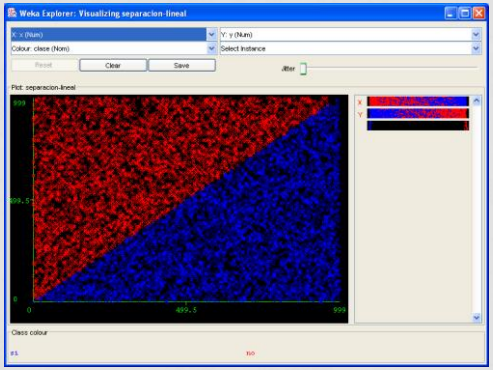
THE CURSE OF DIMENSIONALITY

- Volumes grow exponentially to the number of dimensions d.
- Example: surface of a sphere:
  - 2D: $2\pi r = 10$ instances
  - 3D: $4\pi r^2 = 100$ instances
  - 4D: $2\pi^2 r^3 = 1000$ instances
  - 50D: $= 10^{50}$ instances
  - dD: $O(r^{d-1})$



THE CURSE OF DIMENSIONALITY IN A LINEAR CLASSIFIER

- Two-class classification problem with **1000 attributes**
- Let's solve it with a linear classifier (i.e. the boundary is a hyper-plane)
- Let's assume we have **1001 training instances** (and 10000 test instances)
- What would be the training accuracy?
- What would be the test accuracy?



$$A_1*X_1 + A_2*X_2 + A_3*X_3 + \ldots + A_{1000}*X_{1000} > A_0$$

THE CURSE OF DIMENSIONALITY

- Conclusion: if there is not a good relation of available data to the number of attributes, classification may not be accurate, **even if all the attributes are relevant**

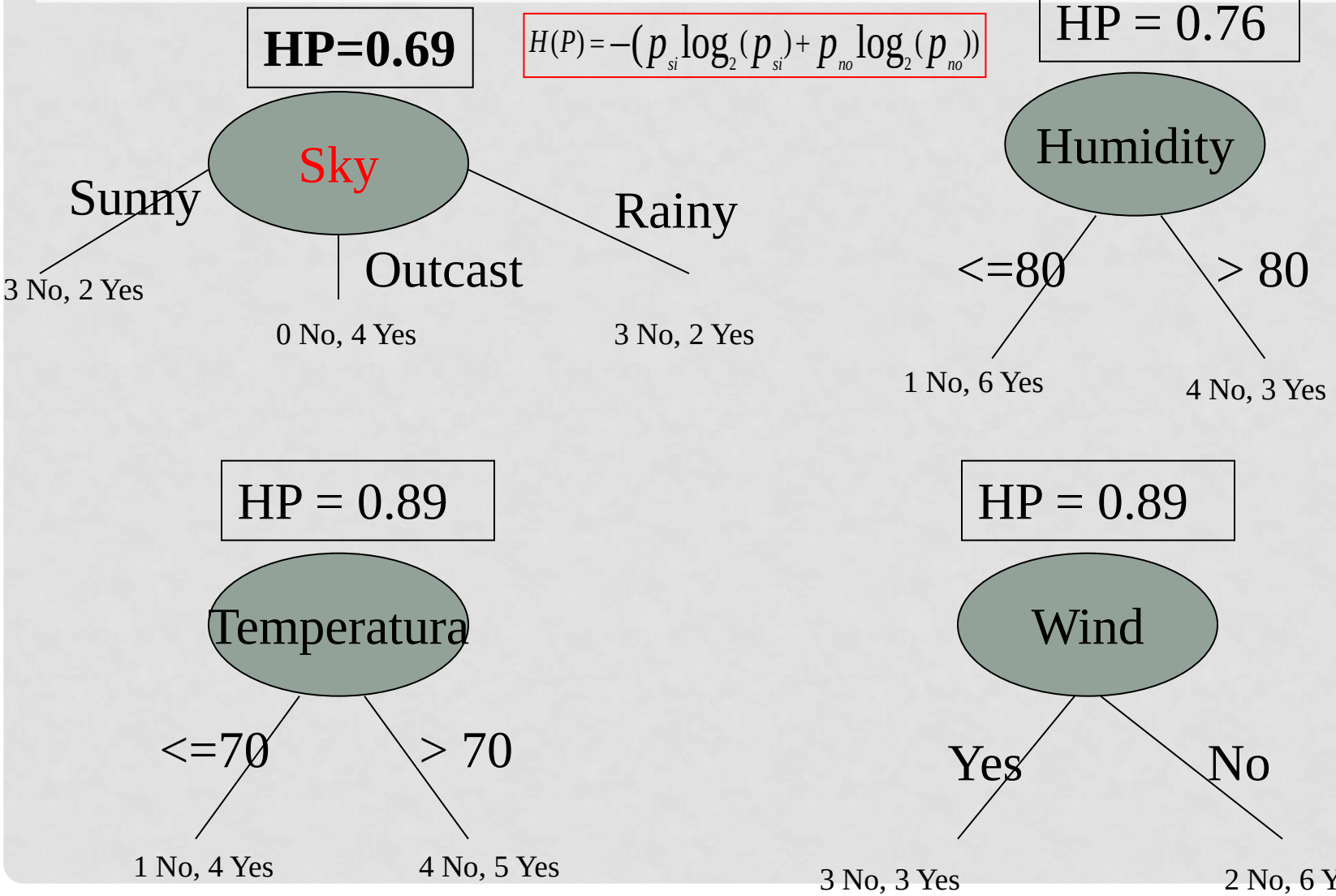
ADVANTAGES OF ATTRIBUTE / FEATURE SELECTION

- Alleviate the curse of dimensionality
- Improve generalization of the classifier (removing irrelevant and redundant attributes)
  - However, bear in mind that some classifier learning algorithms are able to deal with irrelevant attributes indirectly via hyper-parameters. For instance, shallow decision trees indirectly force the algorithm to choose the most relevant attributes. In other algorithms such as neural networks and SVMs, this is called "regularization"
- Speed up the learning process (but it is necessary to include the time for the attribute selection phase)
- Improve the interpretability of the model (by reducing the complexity / size of the model)

Ranking

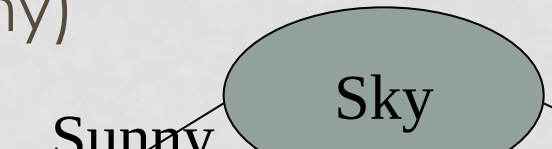
- Given input attributes A_1, A_2, ..., A_n, each A_i is evaluated by itself, computing its correlation with the class, independently of the rest of attributes (i.e. attributes are considered individually, rather than subsets)
- An attribute A_1 is correlated with the class, if knowing its value implies that the class can be predicted more accurately
  - For instance, car speed is correlated with having an accident. But the Social Security Number of the driver is not.
  - For instance, salary is correlated with credit default
- How to evaluate / rank attributes (attribute/class correlation):
  - Entropy (information gain), like in decision trees
  - Chi-square
  - Mutual information
  - ...
- Once evaluated and ranked, the worse attributes are removed (according to a threshold)

Entropy / Information Gain for ranking attributes



Entropy / Information Gain for ranking attributes

Sky generates as many partitions as values (3: sunny, outcast, rainy)



S	T	H	W	Ten
Sunny	85	85	No	No
Sunny	80	90	Yes	No
Sunny	72	95	No	No
Sunny	69	70	No	Yes
Sunny	75	70	Yes	Yes

"3 No, 2 Yes"

"no" majority

S	T	H	W	Ten
Outcast	83	86	No	Yes
Outcast	64	65	Yes	Yes
Outcast	72	90	Yes	Yes
Outcast	81	75	No	Yes

"0 No, 4 Yes"

Perfect partition

S	T	H	W	Ten
Rainy	70	96	No	No
Rainy	68	80	No	Yes
Rainy	75	80	No	Yes
Rainy	65	70	Yes	No
Rainy	71	91	Yes	No

"3 No, 2 Yes"

"no" majority

RANKING WITH MUTUAL INFORMATION

If x and y are independent,
p(x,y)=p(x)*p(y)

$$I(x,y) = \sum_i \sum_j p(x=i, y=j) \log \left[\frac{p(x=i, y=j)}{p(x=i)p(y=j)} \right]$$

- i means the values of attribute x, j means the values of class y
- I(x,y)=0 if x and y are independent (log(1) = 0)
- I(x,y)>= 0 (the more correlated, the larger is mutual information)
- If x or y are continuous, discretize them

RANKING WITH CHI SQUARE

- Chi square is a statistical test that measures the association strength between two variables
- Liu, H., & Setiono, R. (1995, November). Chi2: Feature selection and discretization of numeric attributes. In *tai* (p. 388). IEEE.

Ranking

- Advantages: fast
- Disadvantages:
  - Redundant attributes are not removed
  - Attributes are evaluated individually. Therefore, attribute interaction is not detected
  - Attribute interaction: subsets of attributes that work well together but not individually. In fact, they are likely to be discarded.

WHAT IS ATTRIBUTE INTERACTION?

- Sometimes, two attributes are not predictive separately, but they are if they are used together (**attribute interaction**)
- Example:
  - Classification problem into two classes: computer science and antrhopology
  - Binary attributes "intelligence" and "artificial" which are true if these words appear in the text and false otherwise (remember what we learned about feature extraction)
  - Separately, they do not allow to differenciate between computer science and anthropology, because both words appear in both types of books:

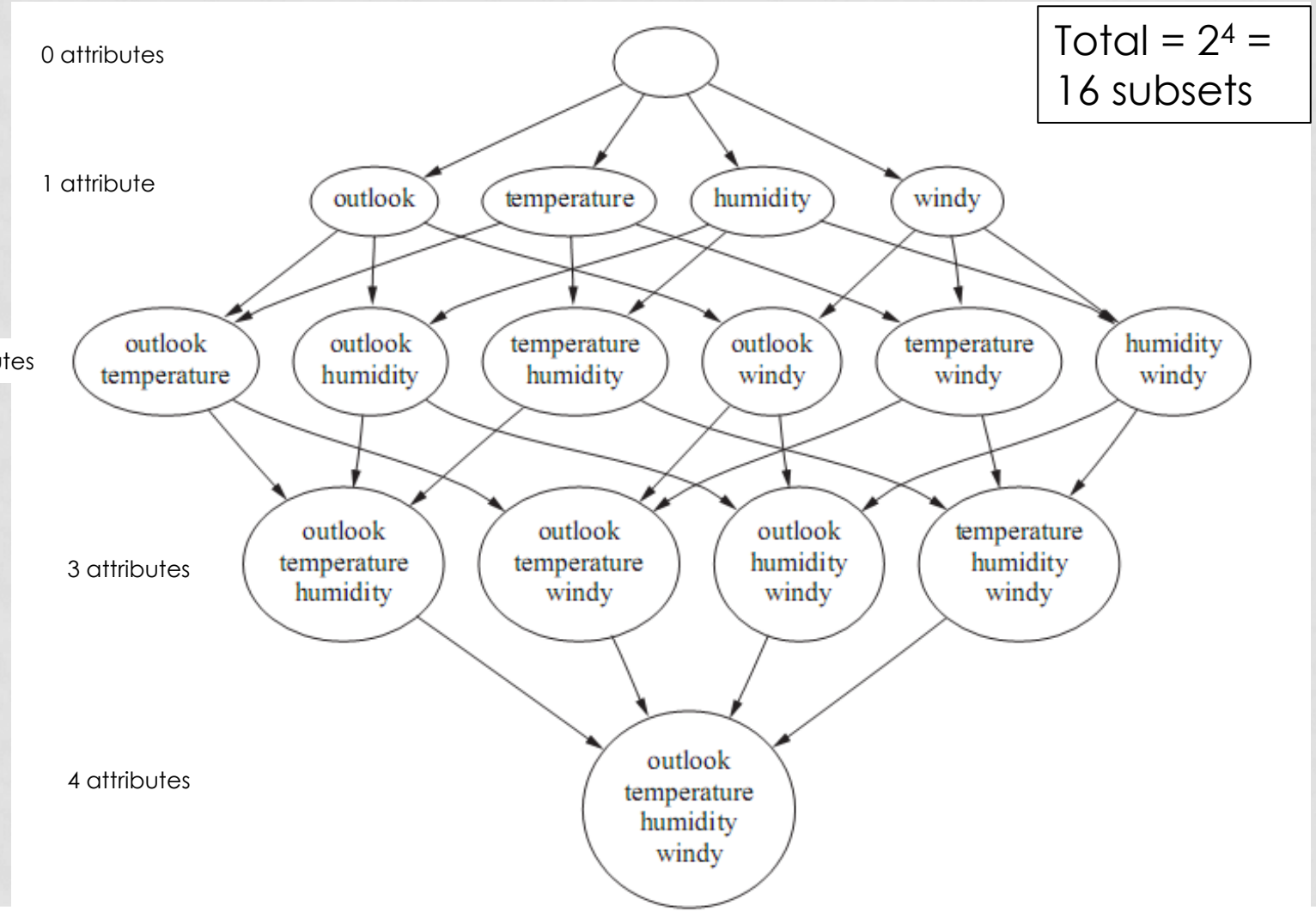
    IF intelligence=yes **THEN** ?; **IF** artificial=yes **THEN** ?
  - But together they can

    IF artificial=yes **AND** intelligence=yes **THEN** "computer science"
- Therefore, in the general case, the aim of attribute selection is to find the smallest **subset** of attributes that "work well" together. In this case, the subset would be {"artificial", "intelligence"}

EXHAUSTIVE SEARCH

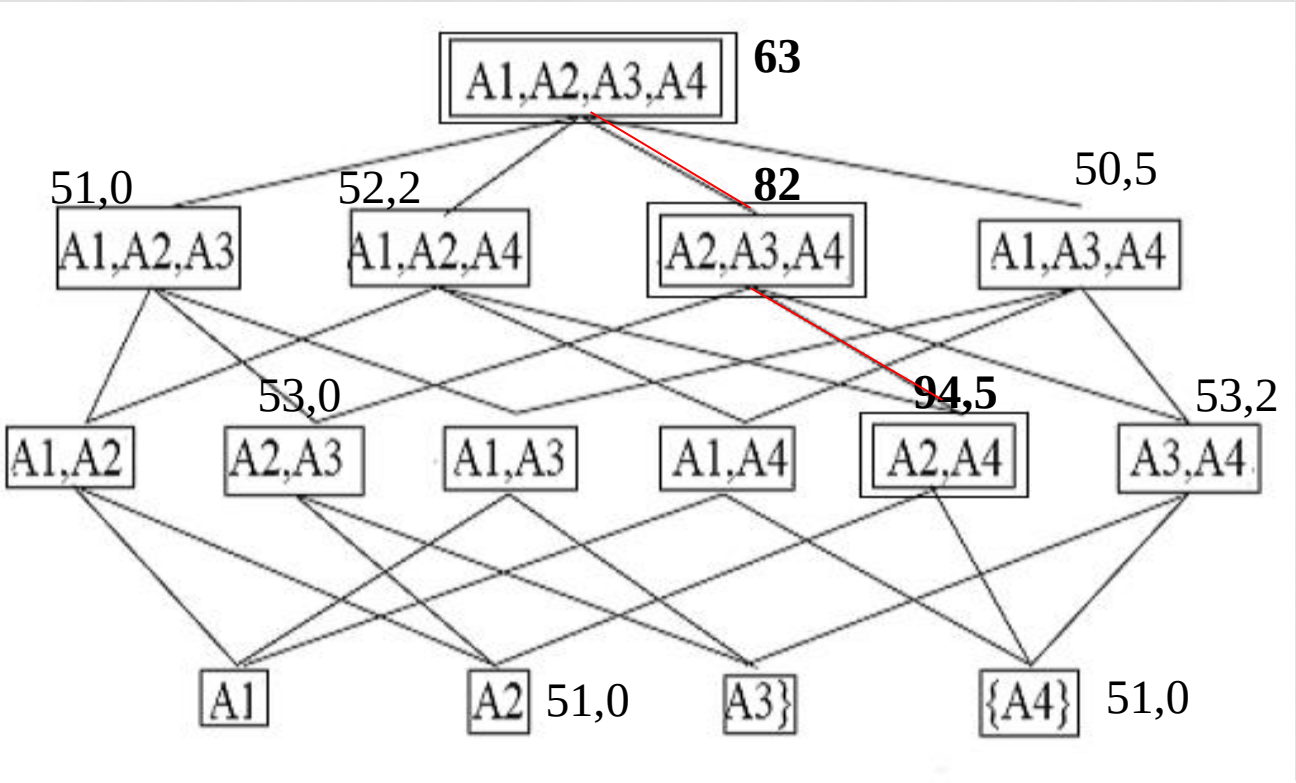
- Test all possible subsets of attributes
- If there are 4 input attributes A, B, C, D
- The list of possible subsets to try is 2^4=16: {A, B, C, D}, {A, B, C}, {A, B, D}, {B, C, D}, {A, C, D}, {A, B}, {A, C}, ..., {A}, {B}, {C}, {D}
- For n large, this is not feasible:
  - n = 10 => 2^{10} = 1024 subsets
  - n = 20 => 2^{20} = 1048576 subsets
  - n = 30 => 2^{30} = 1073741824 subsets
  - …

Define the space of subsets of attributes



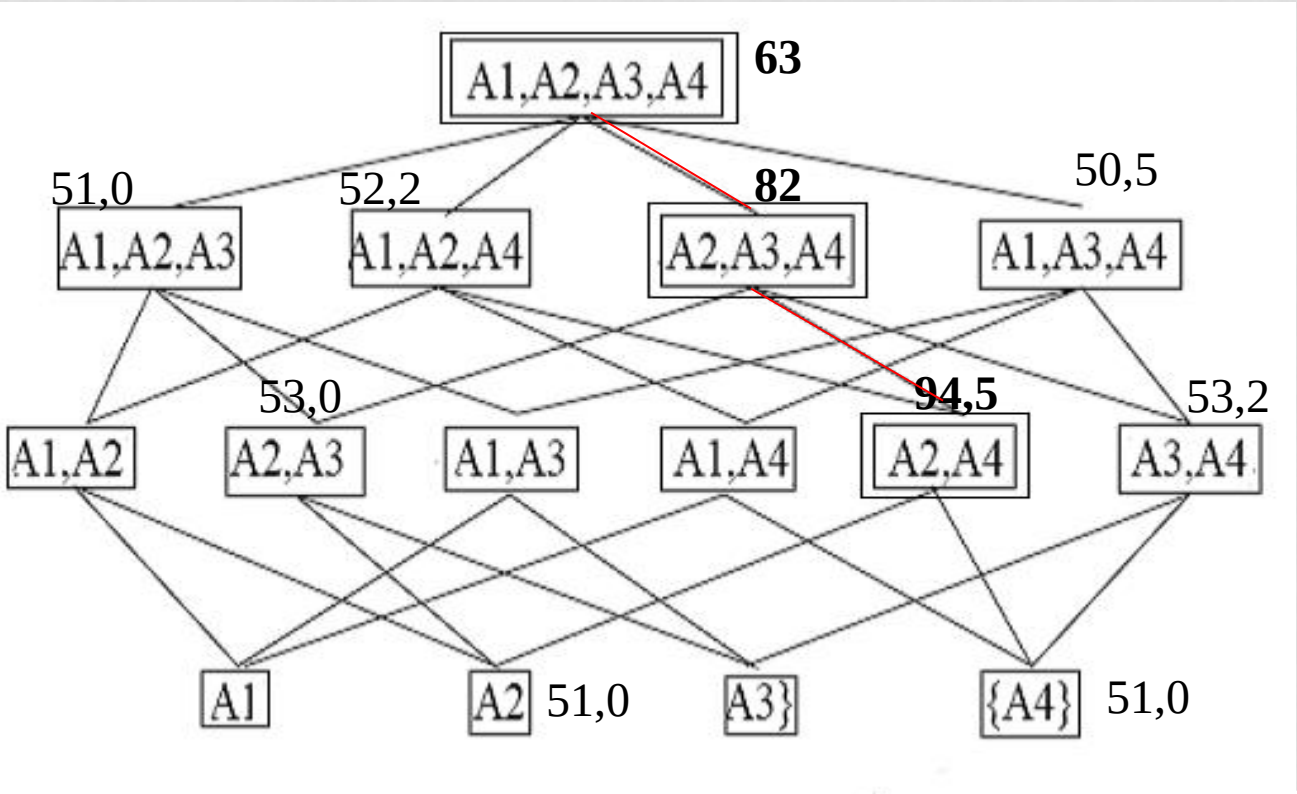
SUBSET SELECTION

- Subsets are evaluated (rather than individual attributes)
- But given that exhaustive search is not feasible,
- only a few subsets are evaluated
- Two ways of traversing subset space: backwards and forwards



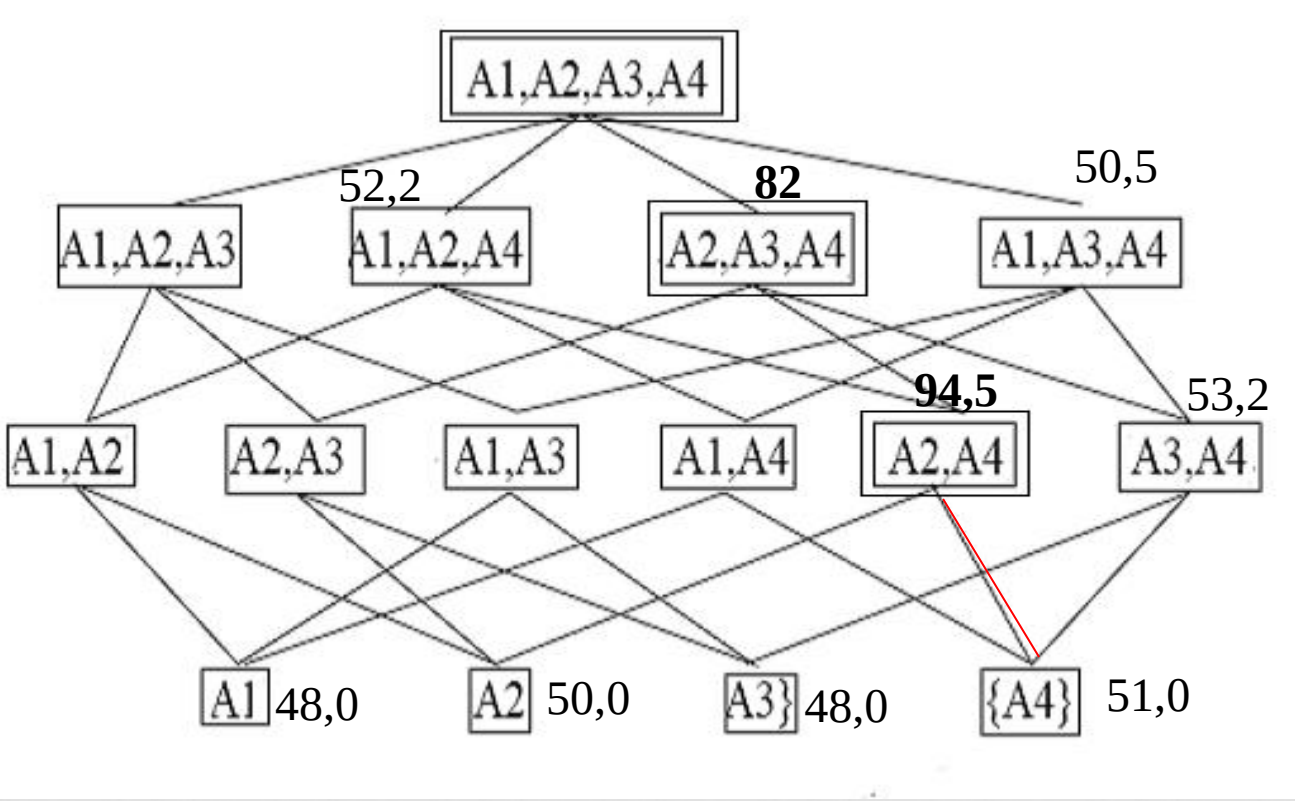
SEQUENTIAL BACKWARD SELECTION

1. Start with the whole set of attributes
2. Remove the attribute that produces the best reduced subset
3. Go to 2 until no improvements are found (or the size of the subset is small enough)



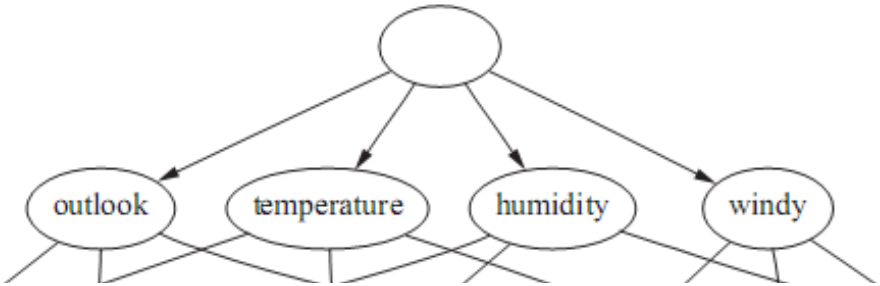
SEQUENTIAL FORWARD SELECTION

1. Start with the empty set of attributes
2. Add the attribute that best works with the current attribute set
3. Go to 2 until no improvements are found

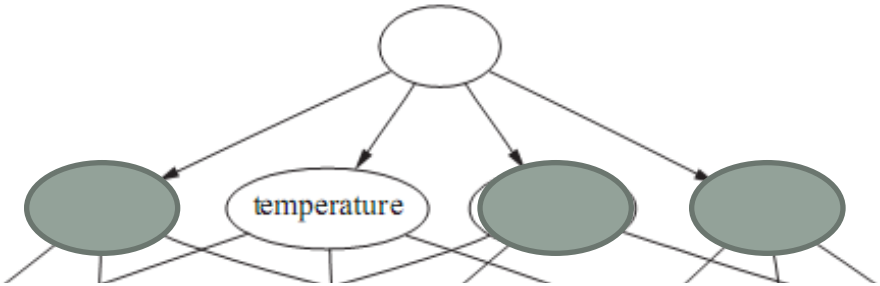


SEQUENTIAL FORWARD SELECTION

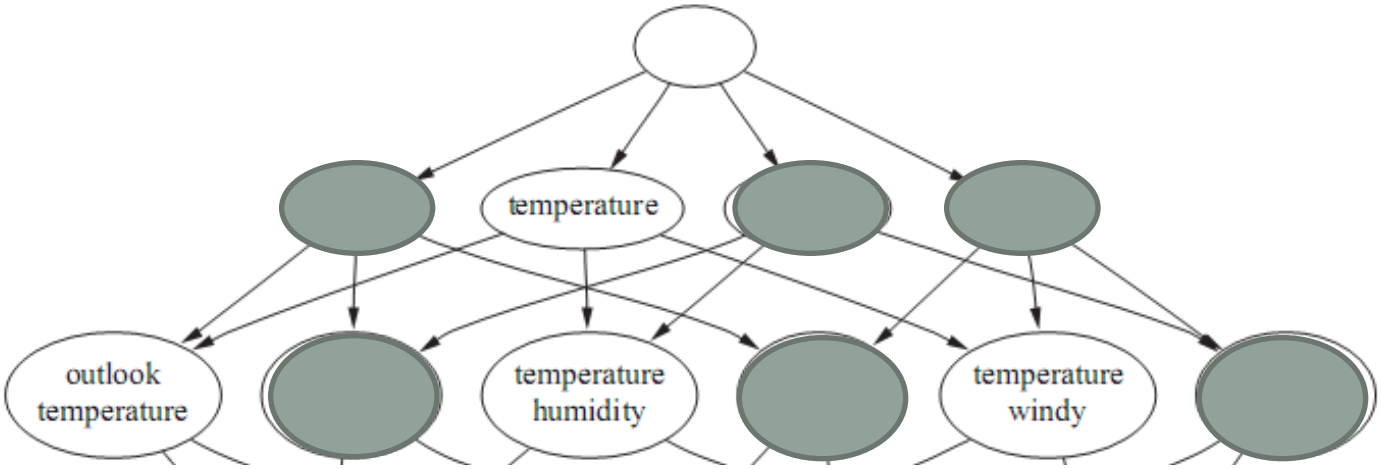
SEQUENTIAL FORWARD SELECTION



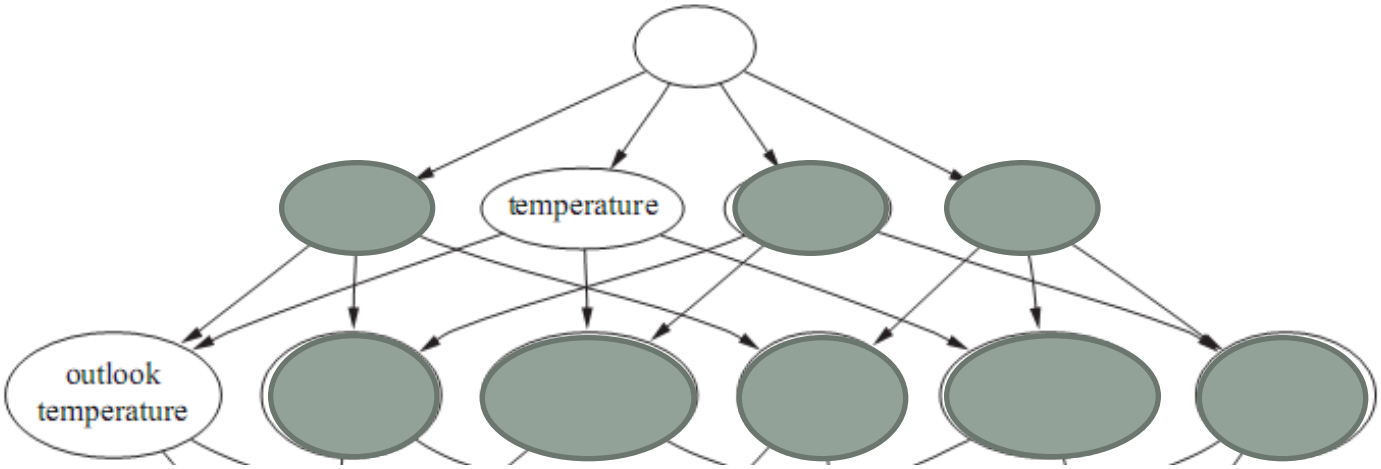
SEQUENTIAL FORWARD SELECTION



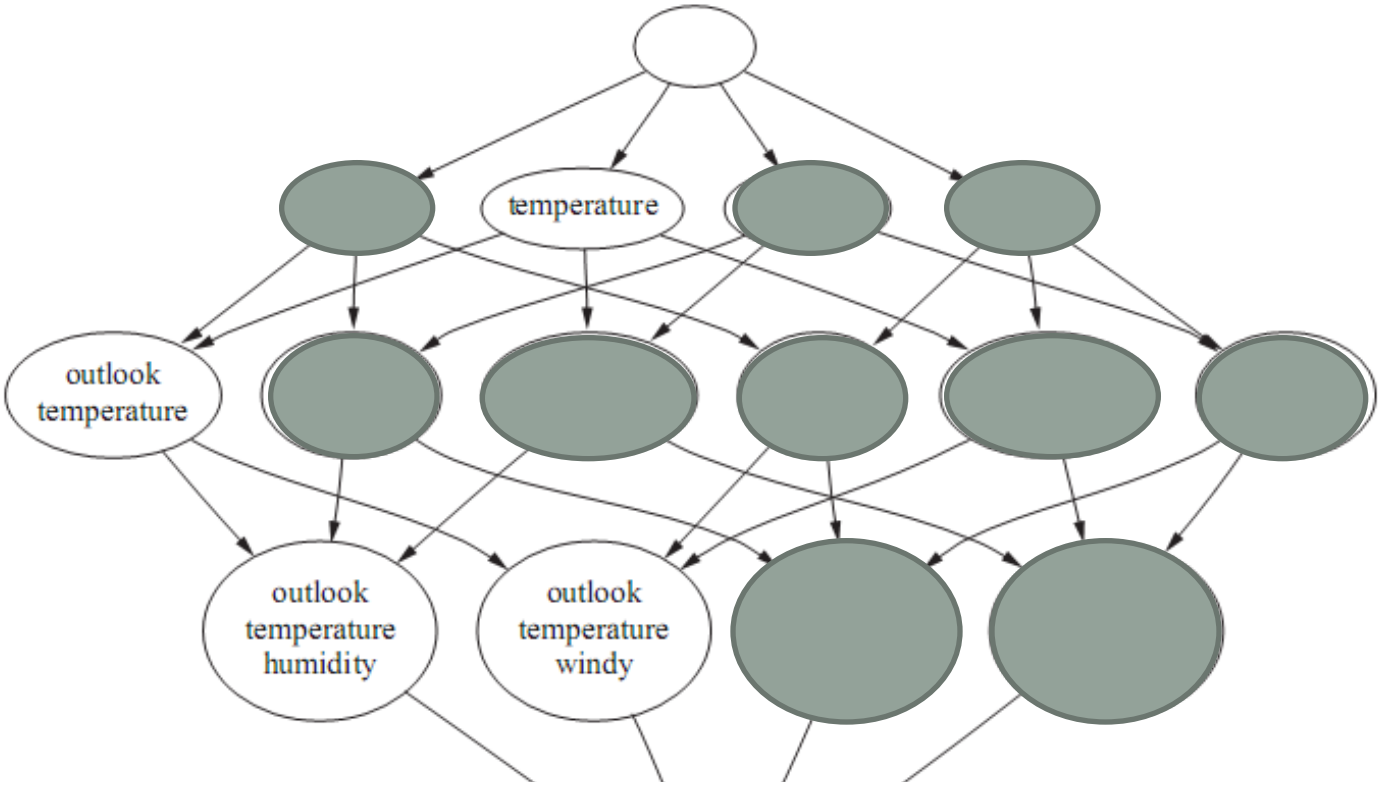
SEQUENTIAL FORWARD SELECTION



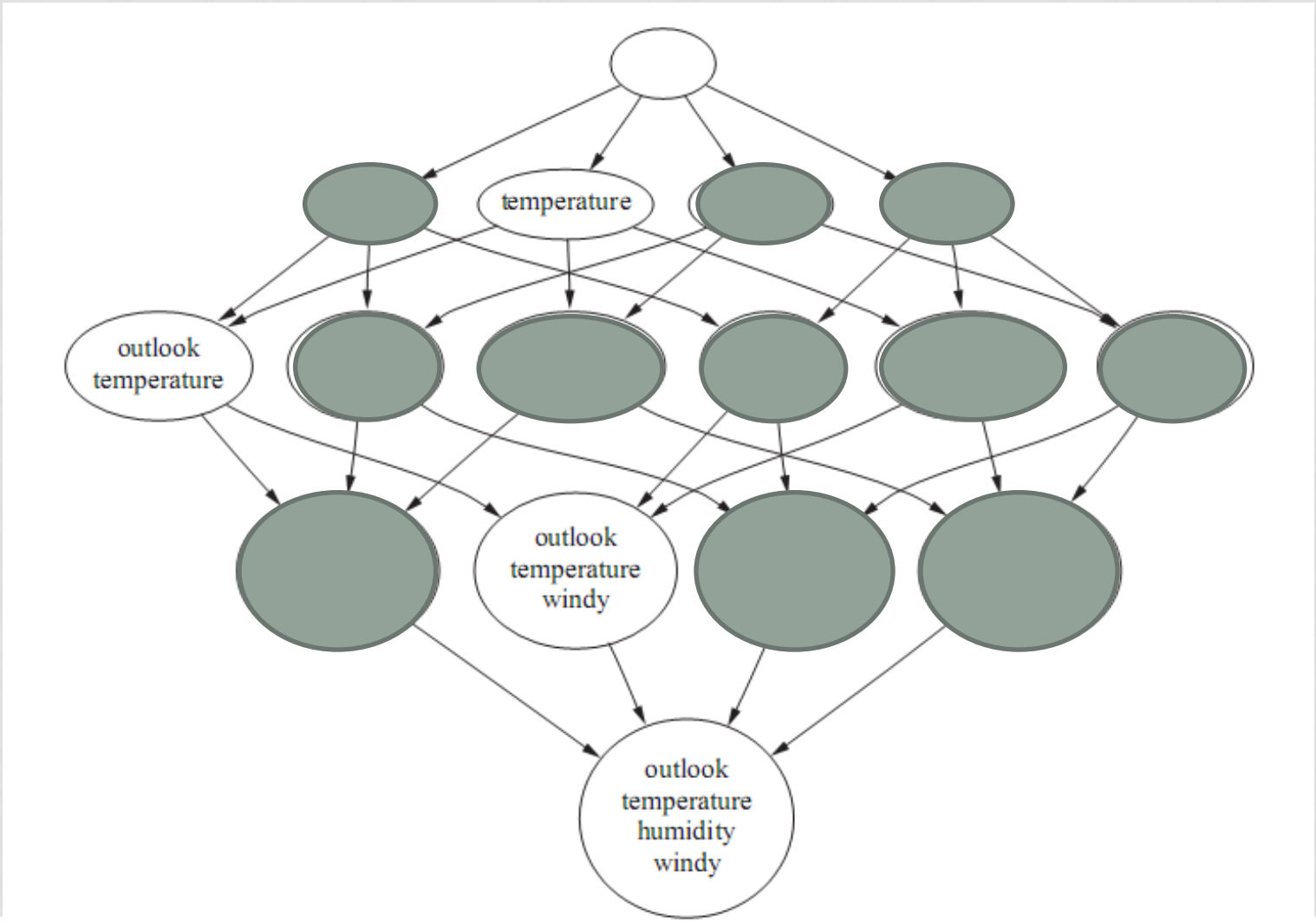
SEQUENTIAL FORWARD SELECTION



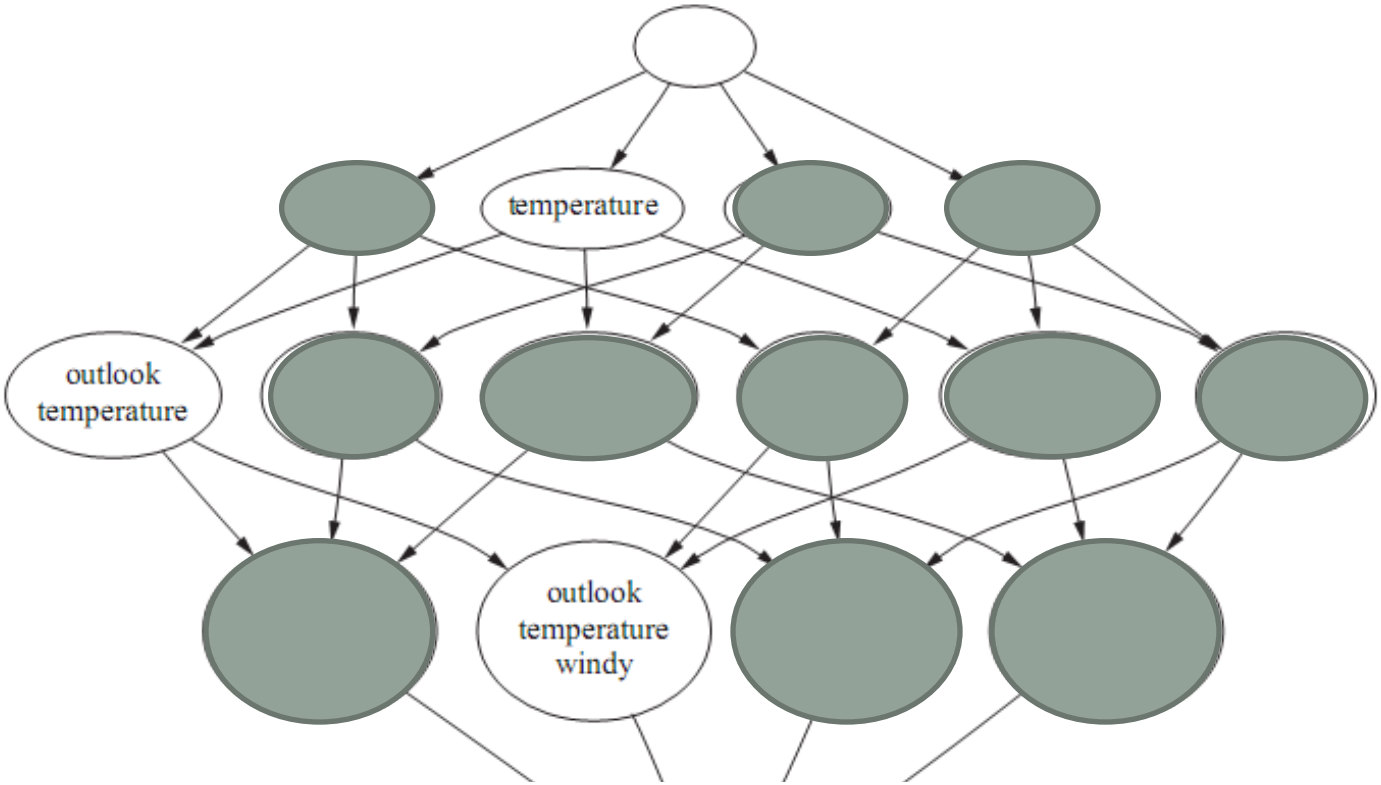
SEQUENTIAL FORWARD SELECTION



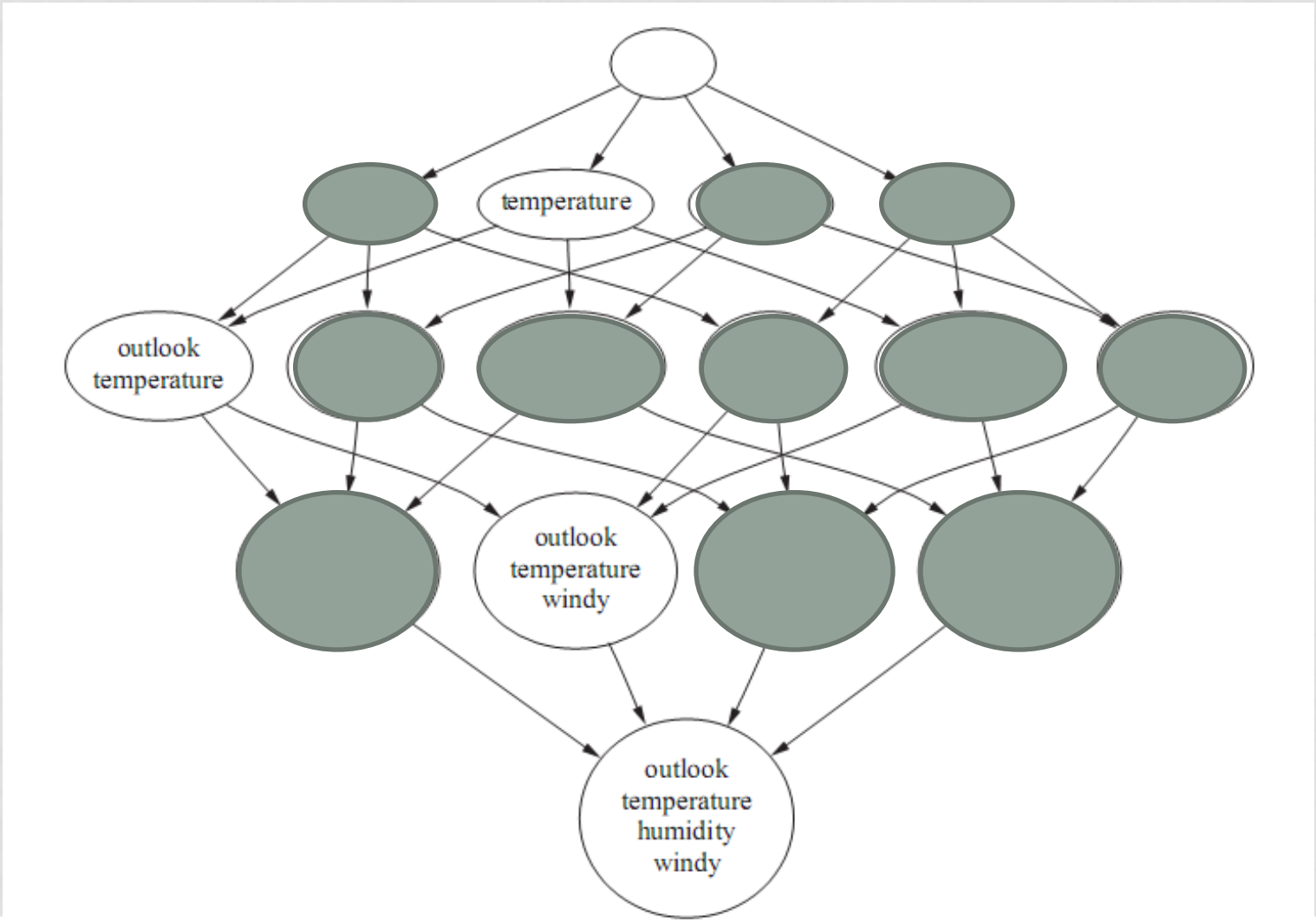
SEQUENTIAL FORWARD SELECTION



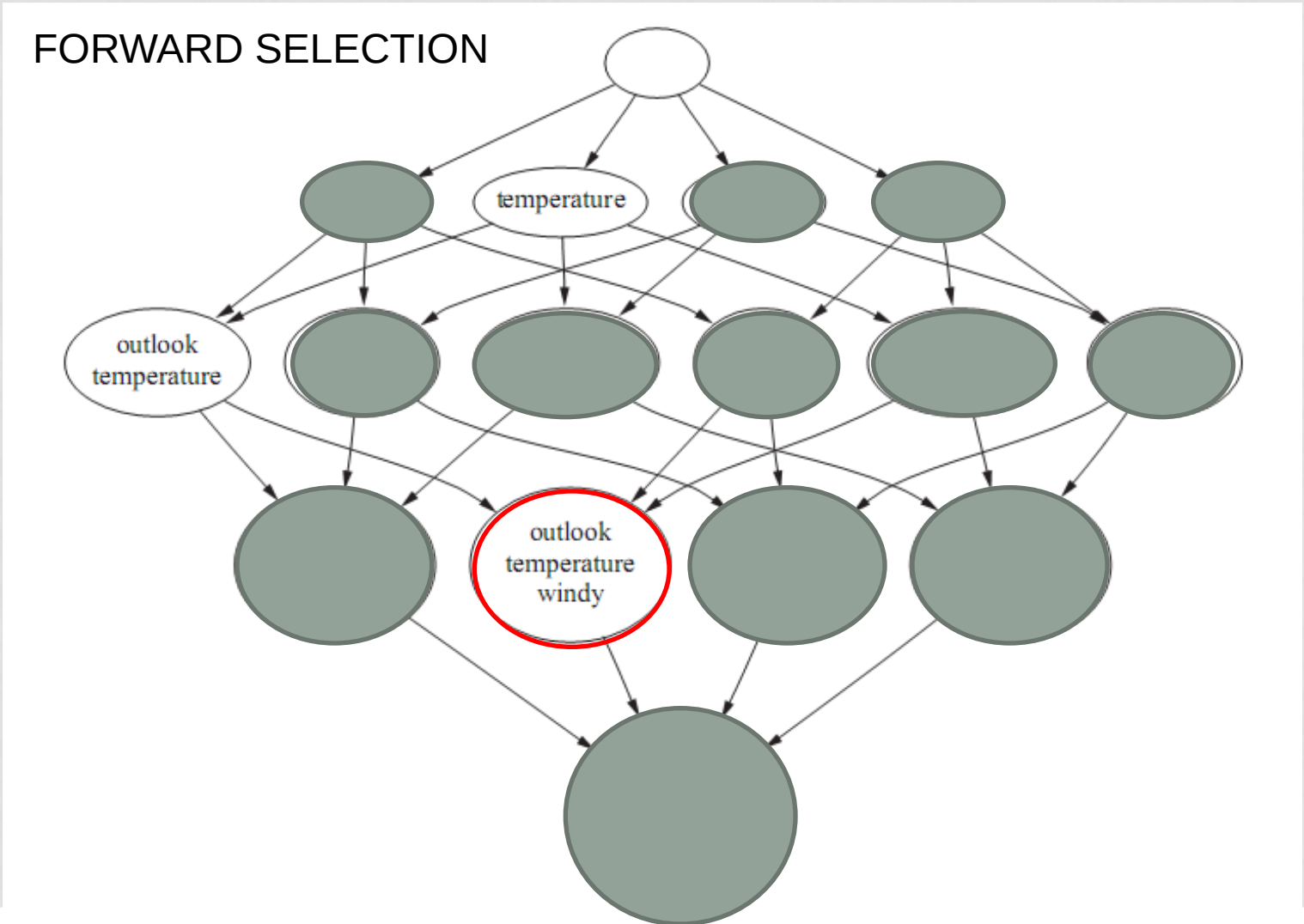
SEQUENTIAL FORWARD SELECTION



SEQUENTIAL FORWARD SELECTION



SEQUENTIAL FORWARD SELECTION



SEQUENTIAL FORWARD FLOATING SELECTION (SFFS)

- If attributes {A, B, D} have been selected so far, and attribute F is added, it might be the case that A, B, or D are no longer so relevant.
- We should check if they can be removed

1. Start with the empty set of attributes
2. Add the attribute that works best with the current attribute set
3. Do Sequential Backward Selection until no more attributes can be removed
4. Go to 2 until no improvements are found

SUBSET EVALUATION

- How to evaluate subsets of attributes?
- We know how to evaluate a single attribute. E.g., Mutual Information in Ranking: $MI(A_3, Class)$
- What about subset $\{A_2, A_7\}$?
  - $MI(A_2, Class) + MI(A_7, Class)$
- Problems:
  - It does not penalize subsets containing redundant attributes
  - It does not detect attribute interaction

SUBSET EVALUATION: Correlation Feature Selection (CFS)

- *CFS* evaluates a subset of attributes computing:
  - The average of each input attribute-class correlation
  - The correlations between input attributes. If two input attributes are correlated, that means they have some degree of redundancy

$$\text{Evaluation}(A_i) = \frac{\text{Average of correlation with the class}}{\text{correlations between input attributes}} = \sum_j U(A_j, C) \Big/ \sqrt{\sum_i \sum_j U(A_i, A_j)}$$

$$\text{Example of evaluation of subset } \{A1, A3, A10\} = \frac{U(A1,C)+U(A3,C)+U(A10,C)}{\text{sqrt}(U(A1,A3)+U(A1,A10)+U(A3,A10)}$$

SUBSET EVALUATION : Correlation Feature Selection (CFS)

- Advantage:
  - Quick
  - It penalizes subsets with redundant attributes
- Disadvantages: it does not detect attribute interactions (i.e. it can remove attributes that are correlated with the class together, but not individually. e.g. “inteligence” and “artificial”)

SUBSET EVALUATION: Wrapper

- *Wrapper* methods evaluate a subset of attributes by building a model (e.g. a decision tree) that uses only those attributes and then computing its expected performance (e.g. success rate)

- E.g. in order to evaluate subset {A1, A3, A10}, a model M is trained that uses only those attributes. The evaluation of the subset is the accuracy obtained by model M.

SUBSET EVALUATION: Wrapper

- Advantages:
  - They obtain subsets of attributes for particular machine learning algorithms (like decision trees)
  - They actually evaluate subsets of attributes
- Disadvantages:
  - They obtain subsets of attributes which are too dependent of the particular machine learning algorithm used
  - Very slow (testing different attribute subsets involves building many models from training sets)
  - Although they are based on a good idea, they sometimes overfit

TYPES OF ATTRIBUTE SELECTION METHODS

	Filter	Wrapper
Ranking (individual attributes)	Entropy (Information Gain), Chi-square, …	
Subset selection	Correlation Feature Selection (CFS)	Wrapper

Ranking: evaluation and ranking of attributes individually. Remove the less relevant attributes, below a threshold

Subset selection : search for the most relevant subset

Filter: evaluate attributes by using a simple expression

Wrapper: evaluate attributes by learning a model and testing its performance

- Search methods: different ways of traversing the space of attribute subsets
  - Greedy stepwise:
    - Sequential Forward Selection
    - Sequential Backward Selection
  - Best first (this is an artificial
  - Genetic algorithms
  - …

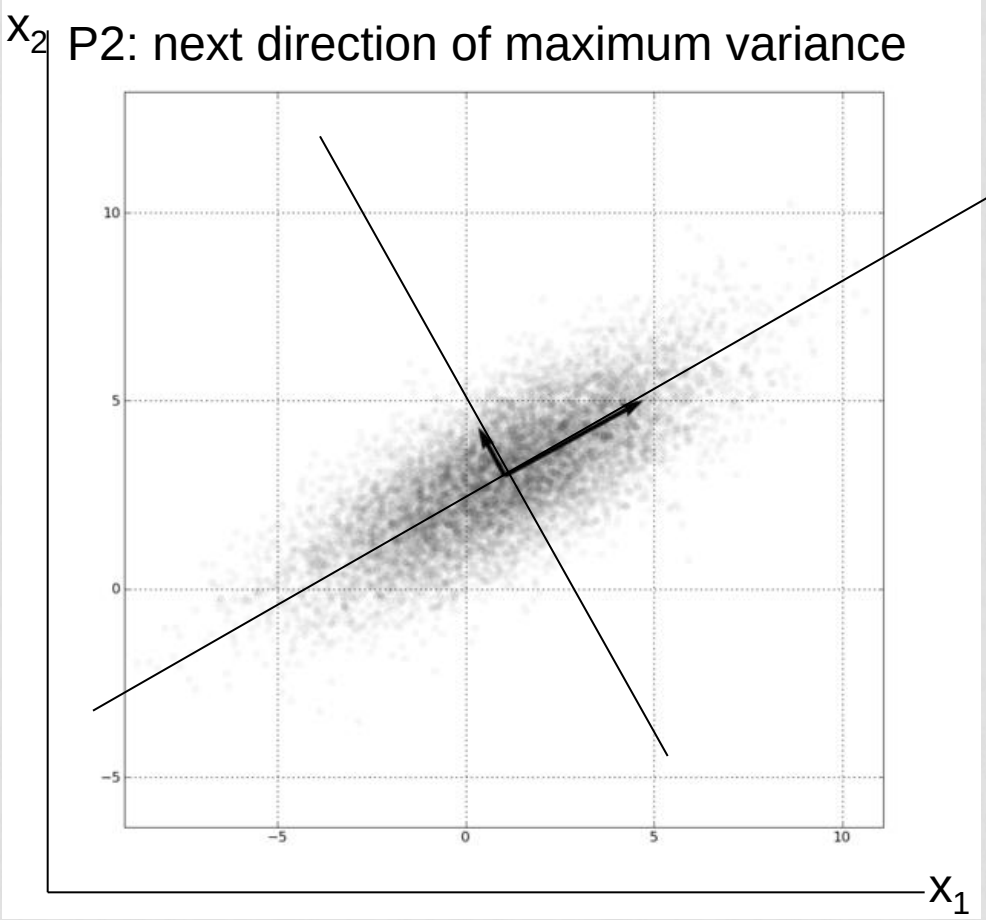
TRANSFORMATION (+ SELECTION) OF ATTRIBUTES

- Principal Component Analysis (PCA)
- Random Projections

TRANSFORMATION WITH PRINCIPAL COMPONENT ANALYSIS (PCA)

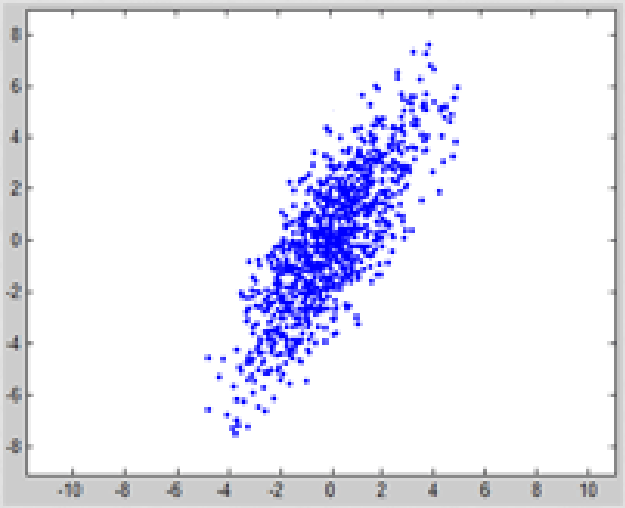
- This method constructs new attributes, as a linear combination of the original input attributes
- The new attributes are sorted by the variance of the new attributed (explained variance)
- Dimensionality can be reduced by choosing the attributes with more variance

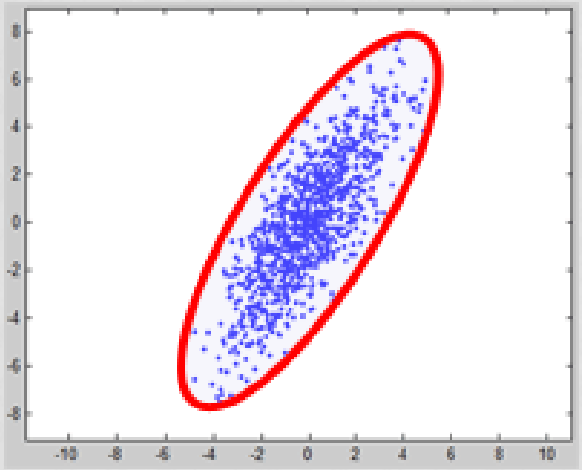
PCA

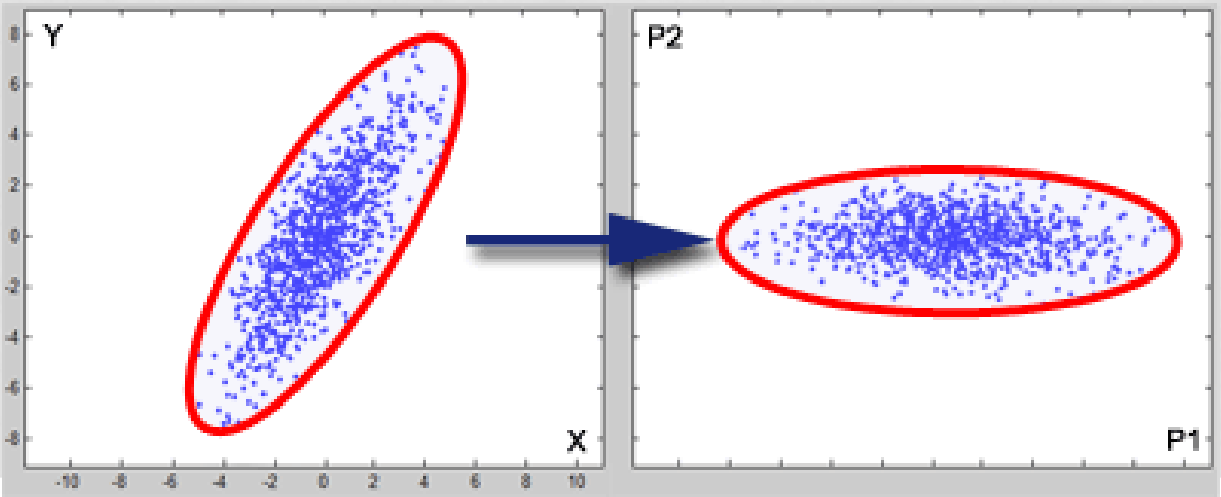


Two new attributes: P1 and P2

PCA TRANSFORMATION







- Linear transformations
- It removes redundancy from attributes (correlation)

$P_1 = k_{11}*x_1+k_{12}*x_2$

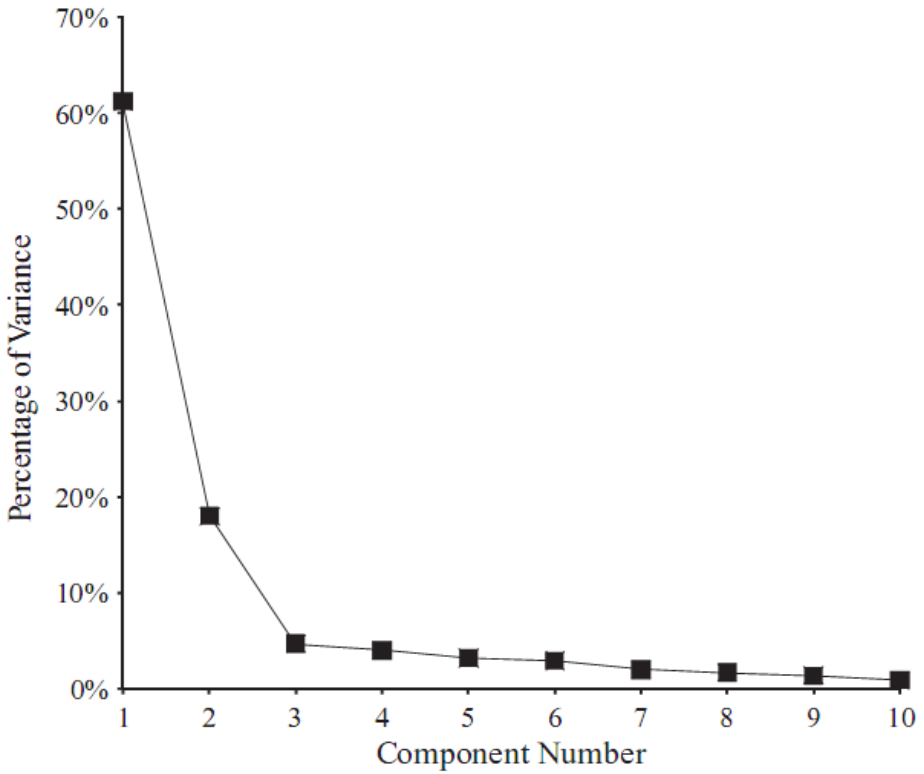
$P_2 = k_{21}*x_1+k_{22}*x_2$

$P = \mathbf{X}*k$

PCA: TRANSFORMATION AND SELECTION

Axis	Variance	Cumulative
1	61.2%	61.2%
2	18.0%	79.2%
3	4.7%	83.9%
4	4.0%	87.9%
5	3.2%	91.1%
6	2.9%	94.0%
7	2.0%	96.0%
8	1.7%	97.7%
9	1.4%	99.1%
10	0.9%	100.0%

(a)

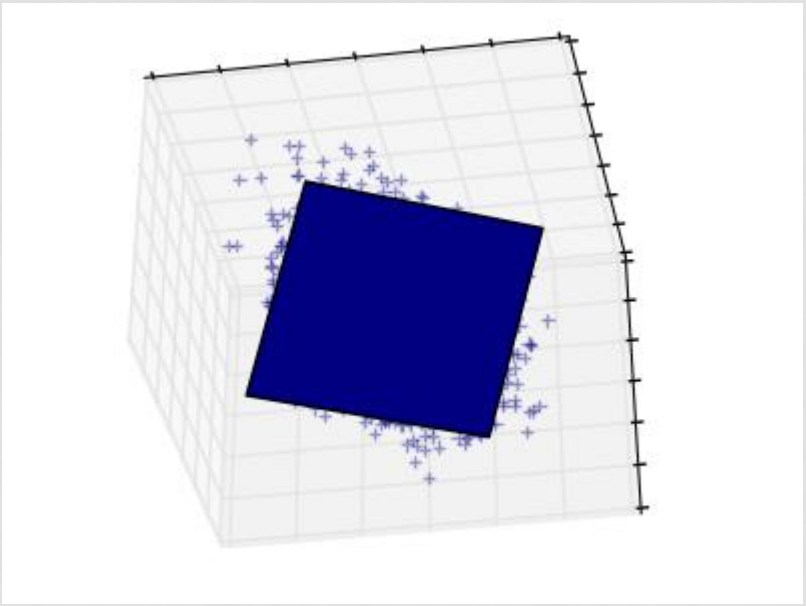


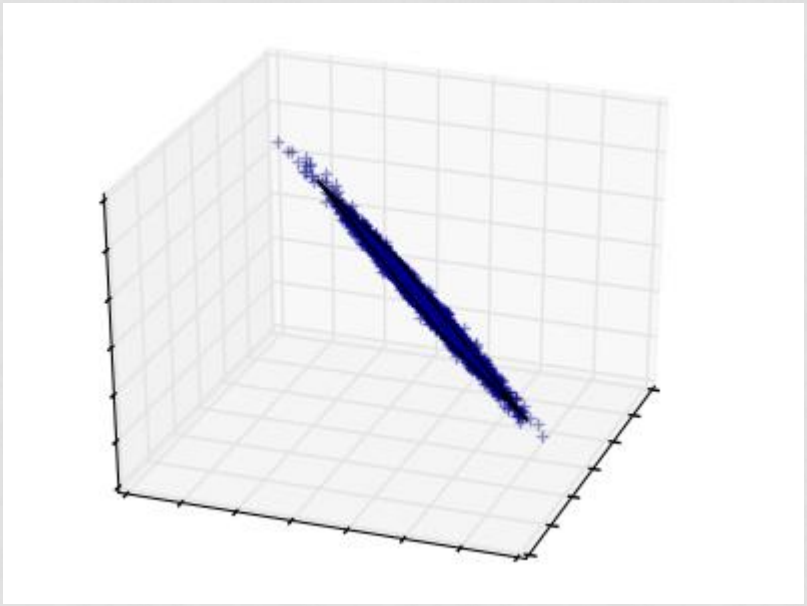
(b)

- Typically, a threshold is set so that the explained variance is larger than 95% (7 in this case)
- If only a few attributes explain most of the variance, the rest can be removed (e.g. imagine two dimensional data embedded in 20 dimensions)

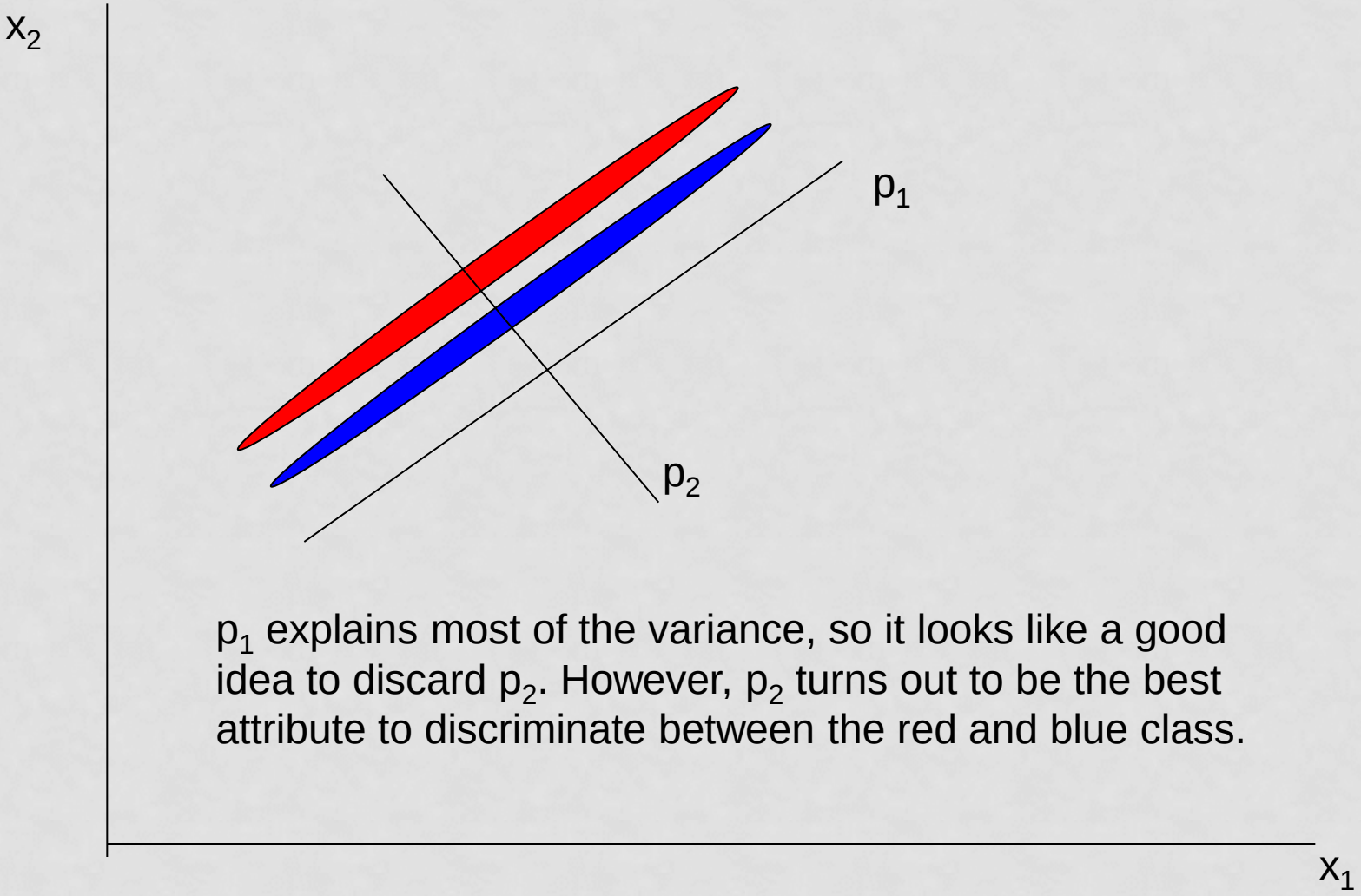
PCA AND ACTUAL DIMENSION OF DATA

- A two dimensional dataset embedded in three dimensions





BEWARE, PCA IS NOT SUPERVISED



p_1 explains most of the variance, so it looks like a good idea to discard p_2. However, p_2 turns out to be the best attribute to discriminate between the red and blue class.

ADVANTAGES / DISADVANTAGES OF PCA

- Advantage: it may find out the actual dimensionality of data
  - E.g.: let's imagine instances in 2D with an ellipsoid shape, but embedded in 20 dimensions. PCA will easily identify that only 2 dimensions are required.
- Advantage: decorrelates attributes (removes redundancy between attributes)
- Disadvantage: PCA is **not supervised**, so there is guarantee that it will find out the attributes that best discriminate between the classes.
- Disadvantage: Slow if lots of attributes.

RANDOM PROJECTIONS

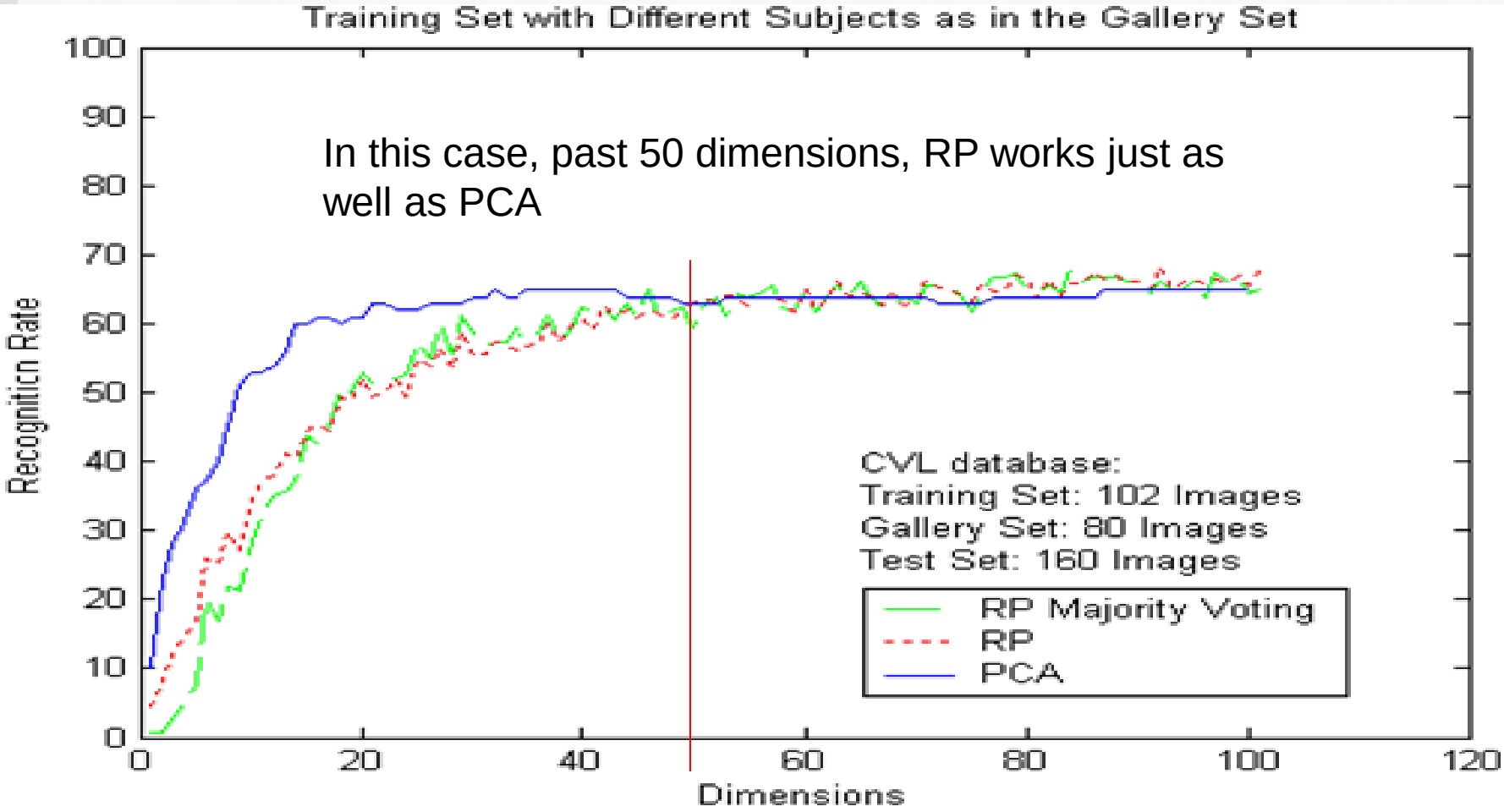
- Projecting data to smaller dimensions by means of random matrices. They can usually obtain similar results to PCA but quickly, as far as the number of projected dimensions is not too small.
- $X' = X*R$
  - Dim(X) = num. instances x d
  - Dim(R) = dxd' ; d' << d
  - Dim(X') = num. instances x d'
- It can be shown that X' maintains to some extent the structure of instances in X. That is, distances between instances are approximately maintained

RANDOM PROJECTIONS

- Steps:
  1. Generate a matrix R with gaussian random numbers: Normal(0,1)
  2. Orthogonalize R (that is, R's columns become orthogonal vectors), by Gram-Schmidtt method (for instance)
  3. Normalize R's columns to unity

- Step 2 can be removed, because in high dimensions random vectors are almost orthogonal

RANDOM PROJECTIONS



(c)