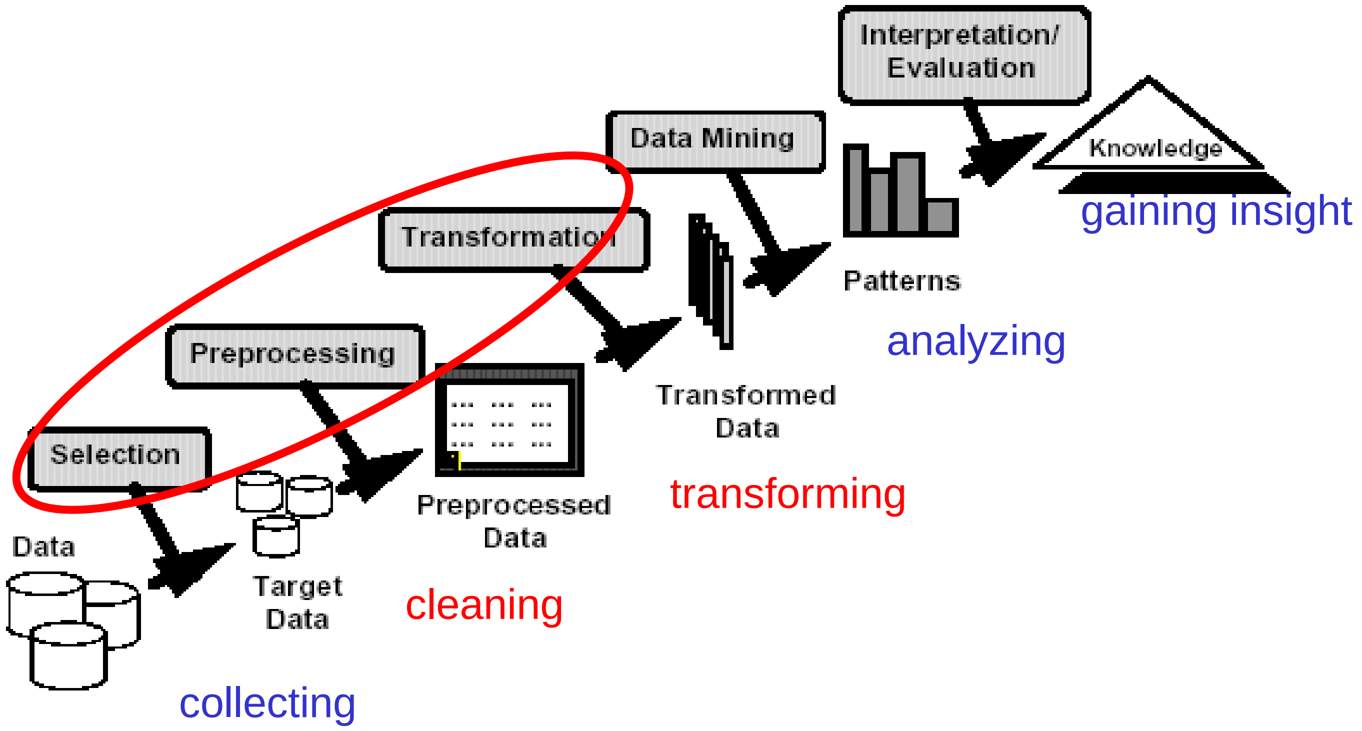
Lecture 2:

Data pre-processing

Outline

- Reminder: Data Mining workflow
- Feature extraction
- Types of features (attributes)
- Data cleaning
- Data reduction (attribute/instance selection)
- Feature transformation

Reminder: data mining workflow



Data mining workflow

A practical problem: an e-commerce retailer is interested to get insights on the behavior of its customers in order to generate recommendations and increase the selling of some products

Sources of data:

- web logs customer accesses + transaction info

```
98.206.207.157 - - [31/Jul/2013:18:09:38 -0700] "GET
/productA.htm HTTP/1.1" 200 328177 "-" "Mozilla/5.0 (Mac OS X)
AppleWebKit/536.26 (KHTML, like Gecko) Version/6.0
Mobile/10B329 Safari/8536.25” "retailer.net”
```

- demographic info collected during the registration of the users and stored in a data base (e.g. e-mail, phone, town, age category, professional category)

What can do the retailer?

Data mining workflow

What can do the retailer?

The retailer could use these date to find insights in the behaviour of the customers

This requires:

- To match the web logs to customers (not easy to do, it is a noisy process which might lead to some erroneous data -> it might need cleaning)

- To aggregate all logs corresponding to the same customer (not all information from a log are useful -> it might need selection)

- Integrate the information from both sources of data (it might need transformation)

Data mining workflow

Main steps

- Data collection (from various sources)
- Data pre-processing
  - Feature extraction (specific to the problem to be solved)
  - Data cleaning (e.g. remove erroneous records or fill in missing values)
  - Feature selection (ignore irrelevant, redundant or inconsistent attributes)
  - Data/feature/attribute transformation
    - Transform the values of an attribute:
      - from numerical to nominal/ordinal (e.g. the age value might be transformed in a category: very young, young, old, very old);
      - From nominal to logical/binary (e.g. for each possible value of a nominal attribute is created a binary attribute)
    - Transform a set of features in another set of features which are more informative (e.g. explain better the variability in data)
- Data analysis (extract knowledge from data)

Feature extraction

Aim:

- Extract meaningful features from raw data (which might come from different sources)

Particularity:

- The extraction process is highly dependent on the data domain and requires expertise in that domain

Examples: extract features from

- images
- documents (XML, PDF)
- web logs
- network traffic data

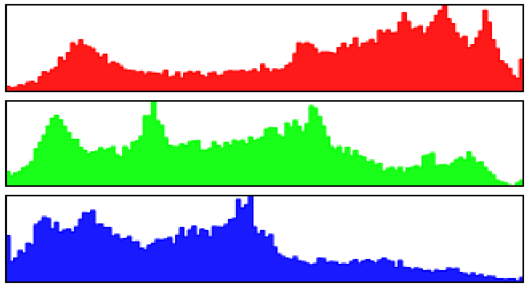
Feature extraction

Example: Extract texture features from an image:

Histogram –based approach:

- Construct the color histograms (for each color channel and for each image region)

  H(v)=number of pixels having the value v
- Compute statistical values:
  - Mean
  - Variance
  - Energy
  - Entropy
  - [Other statistical moments (skewness, kurtosis)]
- Remark: If the image is split in K^2 regions and for each region and each color channel are computed 4 statistical values then the image is transformed in a vector of 12 K^2 numerical features

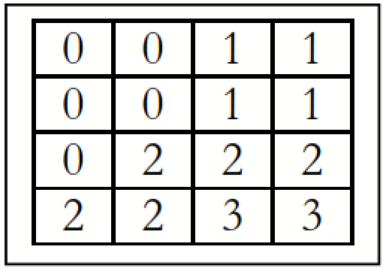


Feature extraction

Example: Extract texture features from an image:

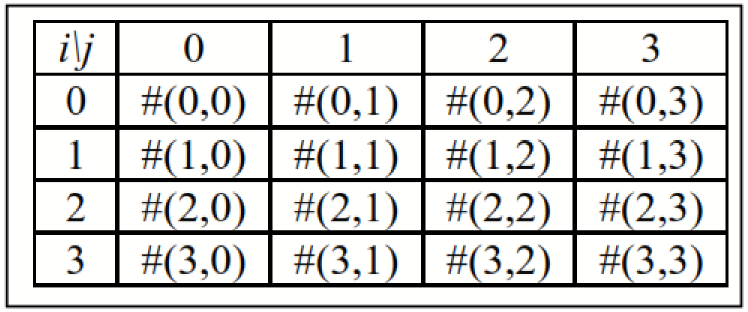
Other variants (see for instance [http://www.eletel.p.lodz.pl/programy/cost/pdf_1.pdf]):

- Co-occurrence matrices



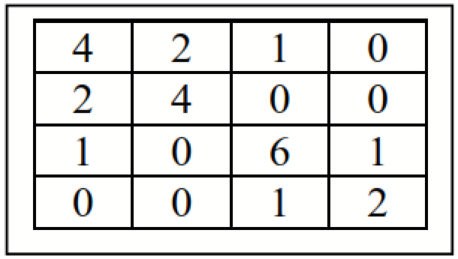
0	0	1	1
0	0	1	1
0	2	2	2
2	2	3	3

Image example



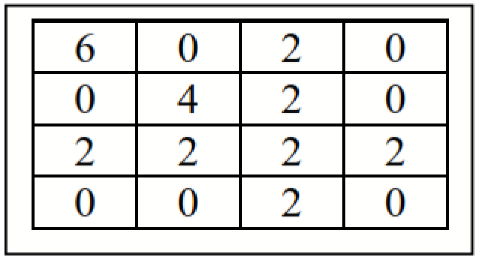
$i \backslash j$	0	1	2	3
0	#(0,0)	#(0,1)	#(0,2)	#(0,3)
1	#(1,0)	#(1,1)	#(1,2)	#(1,3)
2	#(2,0)	#(2,1)	#(2,2)	#(2,3)
3	#(3,0)	#(3,1)	#(3,2)	#(3,3)

Construction of co-occurrence matrix



4	2	1	0
2	4	0	0
1	0	6	1
0	0	1	2

$h_{1,0^\circ}$



6	0	2	0
0	4	2	0
2	2	2	2
0	0	2	0

$h_{1,90^\circ}$

Feature extraction

Example: Extract features from a document

1. XML - semistructured data

```
<PersonalData><PersonDescriptors><DemographicDescriptors><Natio
nality>francaise</Nationality>
</DemographicDescriptors>
<BiologicalDescriptors><DateOfBirth>1978-01-16</DateOfBirth>
<GenderCode>1</GenderCode>
</BiologicalDescriptors>
</PersonDescriptors>
</PersonalData>
```

…

By parsing, one can easily extract the demographic features:

Nationality	Date of birth	Gender
Francaise	1978-01-16	1

Feature extraction

Example: Extract features from a document

2. text file – unstructured data

Example (bag-of-words approach):

“In document classification, a bag of words is a sparse vector of occurrence counts of words; that is, a sparse histogram over the vocabulary. In computer vision, a bag of visual words is a vector of occurrence counts of a vocabulary of local image features.”

a) Remove the stop words

“In document classification, a bag of words is a sparse vector of occurrence counts of words; that is, a sparse histogram over the vocabulary. In computer vision, a bag of visual words is a vector of occurrence counts of a vocabulary of local image features.”

$\Downarrow$

“document classification bag words sparse vector occurrence counts words sparse histogram vocabulary computer vision bag visual words vector occurrence counts vocabulary local image features.”

Feature extraction

Extract features from a document – text file: bag-of-words approach

b) Ignore inflections (reduce the words to their stems) – stemming (Porter algorithm)

“document classification bag words sparse vector occurrence counts words sparse histogram vocabulariy computer vision bag visual words vector occurrence counts vocabulariy local image features”

[http://text-processing.com/demo/stem/]

“document classif bag word spars vector occurr count word spars histogram vocabulari comput vision bag visual word vector occurr count vocabulari local imag featur”

Feature extraction

Extract features from a document – text file: bag-of-words approach

c) Compute the frequencies

“document classif bag word spars vector occurr count word spars histogram vocabulari comput vision bag visual word vector occurr count vocabulari local imag featur”

The extracted features:

(bag,2), (classif,1), (comput,1), (count,2), (document,1), (featur,1), (histogram,1), (imag,1), (local,1), (occurr,2), (spars,2), (vector,2), (vision,1), (visual,1), (vocabulari,2), (word,3)

Feature extraction

Extract features from a web log:

```
192.168.198.92 - - [22/Dec/2002:23:08:37 -0400] "GET / HTTP/1.1" 200 6394
www.yahoo.com "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1...)" "-"
192.168.198.92 - - [22/Dec/2002:23:08:38 -0400] "GET /images/logo.gif HTTP/1.1"
200 807 www.yahoo.com "http://www.some.com/" "Mozilla/4.0 (compatible; MSIE
6...)" "-"
192.168.72.177 - - [22/Dec/2002:23:32:14 -0400] "GET /news/sports.html HTTP/1.1"
200 3500 www.yahoo.com "http://www.some.com/" "Mozilla/4.0 (compatible; MSIE
...)" "-"
```

By parsing the log one can extract:

Client IP address	Date	Time	Request command	etc.
192.168.198.92	22/Dec/2002	23:08:37	GET / HTTP/1.1	
192.168.198.92	22/Dec/2002	23:08:38	GET /images/logo.gif HTTP/1.1	
192.168.72.177	22/Dec/2002	23:32:14	GET /news/sports.htmlHTTP/1.1	

Feature extraction

Result of a feature extraction process:

- Data matrix: each row corresponds to an instance (record), each column correspond to a feature (attribute)

Example (CV data):

	Nationality	Date of birth	Gender
CV 1:	Francaise	1978-01-16	1
CV 2:	Roman	1965-09-01	2

….

- Set of instances, each instance = list of feature values (set of attribute – value pairs)

Example (text file):

File 1: (bag,2), (classif,1), (comput,1), (count,2), (document,1), (featur,1), (histogram,1), (imag,1), (local,1), (occurr,2), (spars,2), (vector,2), (vision,1), (visual,1), (vocabulari,2), (word,3)

File 2: …

Types of features/ attributes

- Numerical (quantitative, continuous)

Examples: age, weight, price, quantity, value, temperature

Characteristics:

- The values of such features are numbers:
  - integer (obtained by counting)
  - real (obtained by measurements)
- There exist an order over the set of values (i.e. one can compute the minimum, maximum, median and most/less frequent value and we can sort the values)
- One can apply arithmetical operations:
  - Compute the average, variance of the values in a data collection
  - Other relevant operations: addition, subtraction, multiplication, division etc (e.g. value = price*quantity)

Remark: a particular case is represented by date values (e.g. 1975-01-16); it makes sense to compare two dates or compute the difference between two dates but it does not make sense to multiply them

Types of features/ attributes

- Ordinal (discrete values from an ordered set)

Examples:

quality levels (e.g: unacceptable, acceptable, good, very good, excellent)

levels of a characteristic (e.g. very low, low, medium, high, very high)

Characteristics:

- The values of such features may be numbers, symbols, strings
- There exist an order over the set of values (i.e. one can compute the minimum, maximum, median and most/less frequent value and we can sort the values)
- One cannot apply arithmetical operations:
    - It does not make sense to add, subtract or multiply two ordinal values

Types of features/ attributes

- Nominal/ categorical (discrete values from a set on which it does not make sense to consider an order relationship)

Examples:

Gender (e.g: female, male)

Race (e.g. caucasian, mongolian, negroid, australoid)

Marital status

Characteristics:

- The values of such features may be numbers, symbols, strings
- It does not make sense to sort the values or apply arithmetical operations on them
- Operations on these values:
  - Equality check
  - Frequency count

Types of features/ attributes

- Binary (only two values: {0,1} or {False, True})
  - Used to encode the absence/presence of some characteristics
  - Allows to specify subsets (interpreted as indicator functions)

Example: transactions in market basket analysis

T1: {milk, bread, meat}

T2: {bread, water}

T3: {butter, meat}

T4: {water}

Trans.	bread	butter	meat	milk	water
T1	1	0	1	1	0
T2	1	0	0	0	1
T3	0	1	1	0	0
T4	0	0	0	0	1

- Remark: this is an example of data conversion (from nominal to binary)

Conversion between types

Converting numeric to categorical features (discretization)

- Motivation: some data mining methods can be applied directly only to categorical features (e.g. decision trees – see Lecture 3-4)
- Main idea:
  - Divide the range of values in a number of subranges
  - Assign a categorical value to each subrange

Example: let us consider the feature "age" taking values in [0,100]; one can transform this numerical feature in a categorical one by considering

Subrange	Nominal value
[0, 10)	1
[10,20)	2
[20,30)	3
[30,40)	4
…	
[90,100]	10

Conversion between types

Converting numeric to categorical features (discretization)

Remarks:

- Through discretization some information is lost
- A uniform discretization is not always the most appropriate approach (e.g. the interval [90,100] might contain a significantly smaller number of records than the other intervals)

Other variants:

- Equi-log: the range of values [a,b] is divided in K subranges $[a_1,b_1),[a_2,b_2),\ldots[a_K,b_K]$ such that $\log(b_i)-\log(a_i)$ is constant (instead of b_i-a_i)
- Equi-depth: each subrange has the same number of records
- Equi-label: each subrange contains values of instances belonging to the same class (it can be applied only in a supervised context – there is a training set which contains the class label)

Conversion between types

Converting numeric to categorical features (discretization)

- Equi-depth: each subrange has the same number of records
- Equi-label: each subrange contains values of instances belonging to the same class (it can be applied only in a supervised context – there is a training set which contains the class label)

Example (the values of the "age" attribute are increasingly sorted):

Age: 15, 16, 16, 20, 20, 20, 25,26,27,30,30,31

Class: c1, c2, c2, c1, c1, c1, c2,c2,c1, c2,c2,c1

Equi-depth: [15,18), [18,22.5),[22.5,28.5),[28.5,31)

Equi-label: [15,15.5), [15.5, 18), [18,22.5), [22.5,26.5), [26.5,28.5), [28.5,30.5), [30.5,31)

Conversion between types

Converting nominal to binary features (binarization)

(Remark: known as “OneHotEncoding”)

Motivation: there are data mining algorithms (e.g. neural networks) which cannot process directly nominal features

Procedure: a nominal attribute A which takes values in $\{v_1,v_2,\ldots,v_r\}$ is transformed in r binary attributes $Av_1,Av_2,\ldots,Av_r$ such that in a given data instance only one of this attributes will have value 1, the other ones having the value 0.

Example: let us consider the following attribute from the “nursery” dataset

@attribute social {nonprob, slightly_prob, problematic}

and let us consider the following instances:

	A_nonprob	A_slightly_prob	A_problematic
slightly_prob	0	1	0
nonprob	1	0	0
nonprob	1	0	0
problematic	0	0	1

Data cleaning

Aim: remove the errors and inconsistencies in the data

Type of errors:

- Erroneous values
- Missing values

Causes of errors:

- Device faults (e.g. for data collected from sensors)
- Human errors (e.g. misinterpretation of filling rules)

Erroneous value

Patient	**Age**	**Height [cm]**	**Weight[kg]**
P1	20	170	60
P2	10	1.30	30
P3	22	165	?
P4	8	190	80

Missing value

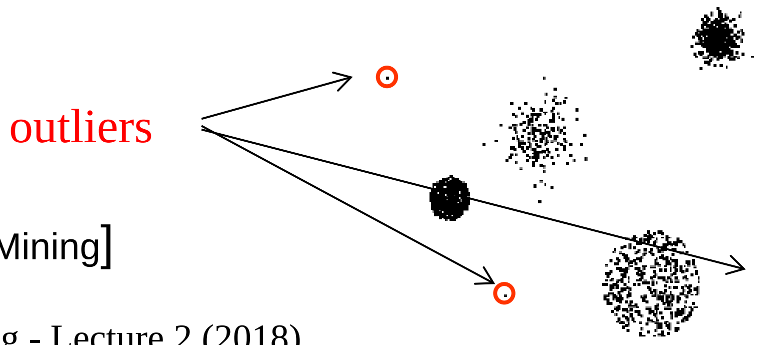
inconsistent values

Data cleaning - errors

Discovering and correcting erroneous values:

- By using domain knowledge (e.g. values outside normal ranges)
- By searching for inconsistencies between the values of the same feature provided by different data sources (e.g. the name of a person can be specified in different ways, “Ioan Popescu”, “I. Popescu”, “Ioan Popesu”; the race of a person appears with different values in different records)

- By using a statistical approach (e.g. supposing that there is a normal distribution of data and the exceptions are possible errors – it should be used with caution as exceptions might suggest relevant outliers)



[Tan, Steinbach, Kumar – Introduction to Data Mining]

Data cleaning – missing values

Reasons for missing values:

- Omission during data collection
- Information not provided (e.g. age or gender in a questionnaire, lack of answers cause by privacy issues)
- Information not applicable (e.g. annual income is not applicable to children)

Handling the missing values:

- Remove the record containing the missing value
- Consider it as a separate value (e.g. if the missing value is marked by 0 then 0 is considered a possible value for that feature)
- Estimate the missing value (such an approach is called imputation) using the corresponding values in “similar” records (the similarity concept depends on the data). For instance in the previous example one can use 60 (as P1 and P3 are similar with respect to the other features). If more than one “similar” records are used then the value to use for imputation is the average of the corresponding values in the similar records

Data cleaning – missing values

Imputation methods: the missing value can be replaced with:

- The mean/ median/ mode of the existing values for the same attribute
- The mean of the values of the attribute in other instances (most similar ones with respect to the other attributes)
- A value inferred by regression starting from the values of another attributed (as long as those two attributes are significantly correlated)

Attribute selection

Aim:

- reduce the data size
- improve the quality of the model extracted from data (by removing redundant features)

Examples:

- Irrelevant features for the data mining task (e.g. ID)
- Correlated features (e.g. BMI=weight/height^2)

Patient	Age	Height [m]	Weight [kg]	BMI	ID	Class
P1	20	1.70	60	20.8	111	normal
P2	15	1.30	30	17.8	222	underweight
P3	22	1.65	100	36.7	333	obese
P4	48	1.90	80	22.2	444	normal

Remark: in practice the relationship between features is hidden and the selection criteria are not obvious

Attribute selection

Aim:

- reduce the data size
- improve the quality of the model extracted from data (by removing redundant attributes)

Main elements of an attribute selection method:

- Selection criterion
- Search method (in the space of attribute subsets)

Remark:

- The attribute selection techniques (particularly the selection criterion) depend on the characteristics of the data mining task and on the available data

Variants:

- Unsupervised selection methods (e.g. used in the context of clustering)
- Supervised selection methods (e.g. used in the context of classification)

Attribute selection

Searching the attribute space:

- consider a data matrix with n attributes
- the search space (all possible subsets of attributes) has the size of order 2^n

Approaches:

- Exhaustive search: analyze the impact of each subset of attributes; it is feasible only when n is rather small
- Forward selection:
  - Start with an empty set of attributes
  - Add sequentially a new attribute (based on the selection criterion – the impact of adding each remaining attribute is analyzed and the best one is selected) – if no attribute improves the currently best performance than the search is stopped
- Backward elimination:
  - Start with the full set of attributes
  - Remove sequentially one of the attributes (the attribute to be removed is selected based on the largest gain in the performance)

Selection / ranking / weighting

- Sometimes is better to rank the attributes according to their relevance and let the user to decide which ones to keep

- The ranking criteria are similar to the selection criteria (aiming to measure the relevance of the attribute in the context of the data mining task)

- The ranking can be done by assigning weights to the attributes (a higher weight means that the attribute is more important)
  - Estimating the weights leads to the necessity of solving an optimization problem (e.g find the weights which minimize the loss of information or maximize the accuracy of the data mining task)
  - The weights are important in the case when the data mining tasks to be further applied are based on computing similarities/ dissimilarities (e.g. nearest neighbor classifiers, clustering)

Example: weighted Euclidean distance $d_w(x,y)=\sqrt{\sum_{i=1}^{n} w_i(x_i-y_i)^2}$

Attribute selection

Selection criterion - how to evaluate a subset of attributes (or the weights values)

- Filter based approaches
  - The selection is based only on the relationships between:
    - attributes (unsupervised context)
    - attributes and classes labels (supervised context)

- Wrapper based approaches
  - The quality of a subset of attributes is estimated using the performance of a classification or clustering model constructed based on that subset of attributes

Attribute selection

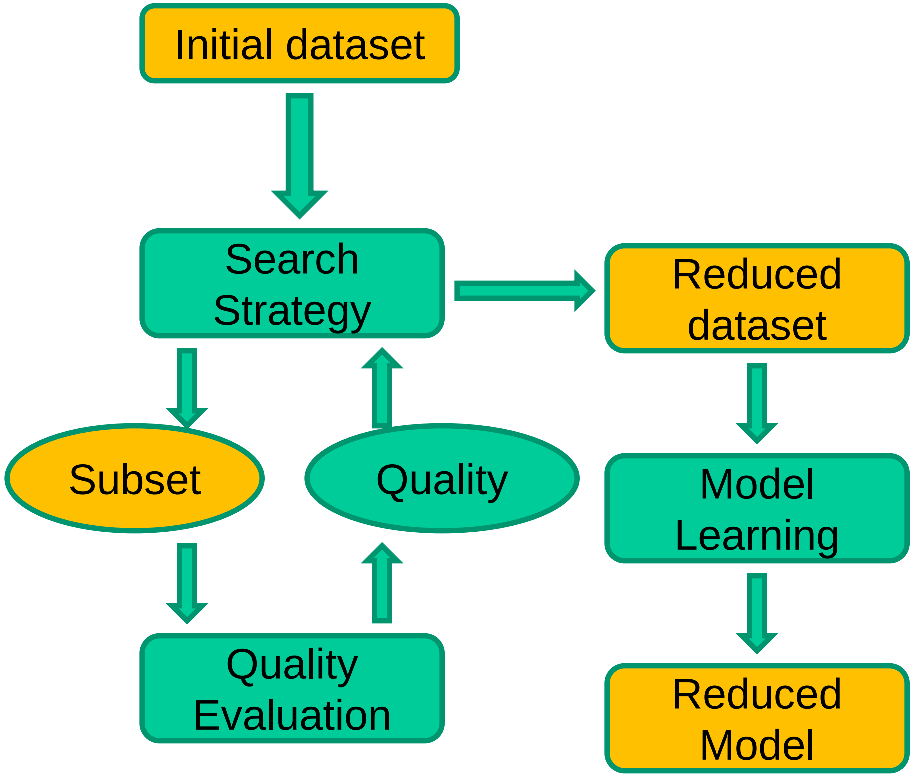
Filter based approach

Data based criteria

- Information gain/ loss
- Compactness (within-class)
- Separation (between-classes)
- Correlation between attributes and class labels
- Mutual information
- Symmetric uncertainty

Advantage: rather low evaluation cost

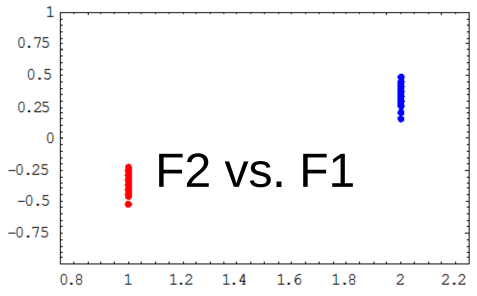
Disadvantage: ignore the impact of the reduced dataset on the model induction algorithm

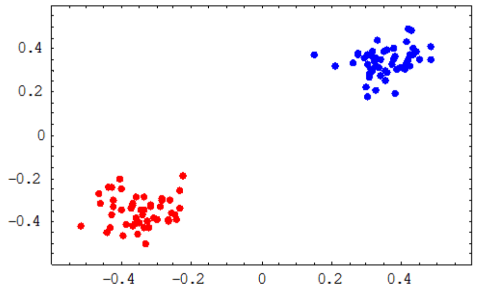


Attribute selection

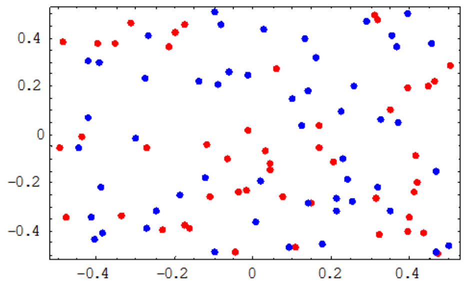
Example: Synthetic dataset: 10 attributes, 2 classes

- Attribute 1: identical with the class label
- Attributes 2-6: random values from $N(m_1,s_1)$ (class 1), $N(m_2,s_2)$ (class 2)
- Attributes 7,8: constant values for all instances
- Attributes 9,10: random values from $U(a,b)$ for all instances





F6 vs. F5



F10 vs. F9

Attribute selection

Unsupervised selection criteria (based only on data without knowledge of the class)

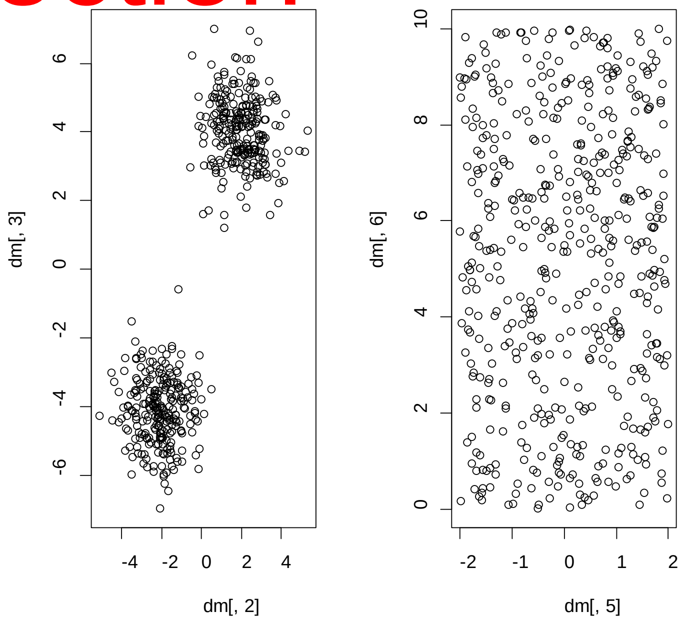
Notations:

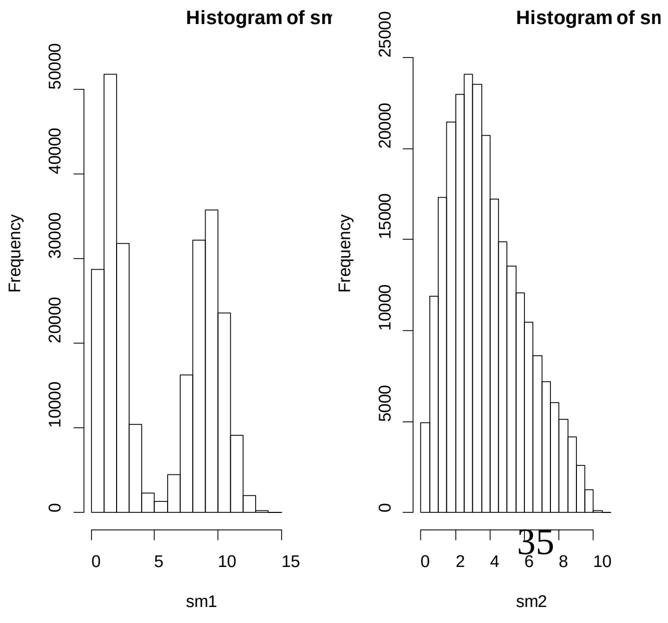
M={$x_1,x_2,\dots,x_N$} be a dataset of N instances, each one containing n attributes

A= set of attributes

Idea:

- Compute the similarities between the pairs of instances in the data set
- Compute the entropy associated to the similarity matrix (viewed as a measure of the information carried by the dataset)
- Analyze the impact of each attribute on the value of entropy and remove the attributes having the smallest influence on the entropy





Attribute selection

Unsupervised selection criteria

Similarity measures (computed based on the set of attributes A)

Numerical attributes

$$S_{ij}(A) = \exp(-\alpha d(x_i, x_j)), \qquad d(x_i, x_j) = \sqrt{\sum_{k=1}^{n} (x_{ik} - x_{jk})^2}$$

$$\alpha = \text{ct. (e.g. } 0.5)$$

Nominal/ordinal/binary attributes

$$S_{ij}(A) = \frac{1}{n}\sum_{k=1}^{n} I(x_{ik}, x_{jk}), \quad I(a,b) = 1 \text{ if } a = b;$$

$$I(a,b) = 0 \text{ if } a \neq b$$

Attribute selection

Unsupervised selection criteria

Entropy measure $E(S,A)=-\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}(S_{ij}(A)\ln(S_{ij}(A))+(1-S_{ij}(A))\ln(1-S_{ij}(A)))$

Remark: intuitively the entropy measures the unpredictability of information content or degree of disorder

Algorithm

Step 1. Start with the full set of attributes A
Step 2. For each attribute a_i in the current state compute $E(S,A-\{a_i\})$ and rank the attributes increasingly by $E(S,A)- E(S,A-\{a_i\})$
Step 3. Remove the first attribute from the sorted list (the attribute which can be removed with a minimal change in the entropy) and repeat Step2 – Step 3 until only one attribute remains in A (or until the loss of entropy by removing an attribute is higher than a threshold)

Attribute selection

Supervised selection criteria – discrete attributes

Gini index: it measures the discriminative power of an attribute

Notations:

$A_1, A_2, \ldots, A_n$ - attributes, $C_1, C_2, \ldots, C_K$ – classes to which the data belong

$v_{i1}, v_{i2}, \ldots, v_{ir}$ – possible values of attribute i (it works only for attributes with discrete values; r_i is the number of values of attribute A_i)

Gini index for attribute A_i

$$G(A_i) = \frac{1}{N}\sum_{j=1}^{r_i} n_{ij} G(v_{ij}),\ \ G(v_{ij}) = 1 - \sum_{k=1}^{K} p_{ijk}^2$$

n_{ij} = number of instances in the data set for which the value of A_i is v_{ij}

$$p_{ijk} = \frac{\text{number of instances in } C_k \text{ with } A_i = v_{ij}}{\text{number of instances with } A_i = v_{ij}}$$

Interpretation: smaller values of $G(A_i)$ suggest higher discriminative power of A_i

Attribute selection

Supervised selection criteria – continuous attributes

Fisher score: it measures the discriminative power of an attribute based on the ratio between inter-class and intra-class variances

Notations: $A_1, A_2, \ldots, A_n$ - attributes, $C_1, C_2, \ldots, C_K$ – classes to which the data belong;

$$F(A_i) = \frac{\sum_{k=1}^{K} n_k (\mu_{ik} - \mu_i)^2}{\sum_{k=1}^{K} n_k \rho_{ik}^2}$$

n_k = number of instances in class C_k

μ_{ik} = average of values of attribute A_i corresponding to the instances which belong to C_k

ρ_{ik}^2 = variance of the values mentioned above

μ_i = average of the values taken by attribute A_i

Interpretation: higher values of $F(A_i)$ suggest higher discriminative power of A_i

Attribute selection

Supervised selection/weighting criteria – continuous attributes

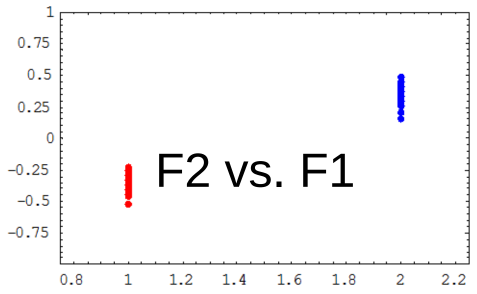
$\{x_i^c; i=\overline{1,N}\},\ x_i^c \in R^n, c\ \in \{1,\ldots,k\}$ class label

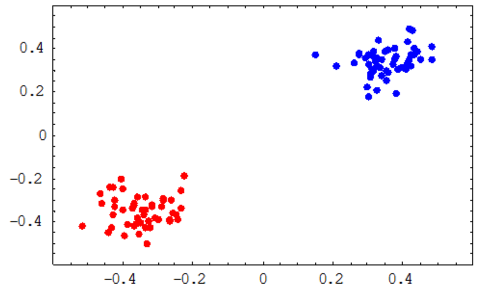
$w=(w_1,\ldots,w_n)$, weight vector

d_w - weighted dissimilarity measure

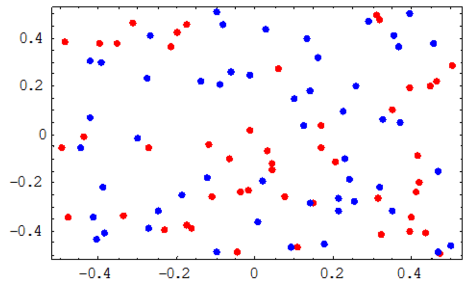
- Compactness (within-class)

$$C_1(w)=\frac{1}{N}\sum_{c=1}^{k}\sum_{i=1}^{n_r} d_w(x_i^c, m_c),\ m_c=\frac{1}{n_c}\sum_{i=1}^{n_c} x_i^c$$





F6 vs. F5



F10 vs. F9

Example:
Synthetic dataset: 10 attributes, 2 classes

- Attribute 1: identical with the class label
- Attributes 2-6: random values from N(m_1,s_1) (class 1), N(m_2,s_2) (class 2)
- Attributes 7,8: constant values for all instances
- Attributes 9,10: random values from U(a,b) for all instances

Attribute selection

Supervised selection/weighting criteria – continuous attributes

$\{x_i^c; i=\overline{1,N}\},\ x_i^c \in R^n, c \in \{1,\dots,k\}$ class label

$w=(w_1,\dots,w_n)$, weight vector

d_w - weighted dissimilarity measure

- Compactness (within-class)

$$C_1(w)=\frac{1}{N}\sum_{c=1}^{k}\sum_{i=1}^{n_c} d_w(x_i^c, m_c),\ m_c=\frac{1}{n_c}\sum_{i=1}^{n_c} x_i^c$$

(To be minimized)

- $C_1(1,1,1,1,1,1,1,1,1,1,1)=0.88$
- $C_1(1,1,1,1,1,1,1,1,1,0,0)=0.78$
- $C_1(1,1,1,1,1,1,1,0,0,0,0)=0.78$
- $C_1(1,1,1,0,0,0,0,0,0,0,0)=0.49$
- $C_1(1,1,0,0,0,0,0,0,0,0,0)=0.34$
- $C_1(1,0,0,0,0,0,0,0,0,0,0)=0$

Example:

Synthetic dataset: 10 attributes, 2 classes

- Attribute 1: identical with the class label
- Attributes 2-6: random values from $N(m_1,s_1)$ (class 1), $N(m_2,s_2)$ (class 2)
- Attributes 7,8: constant values for all instances
- Attributes 9,10: random values from U(a,b) for all instances

Attribute selection

Supervised selection/weighting criteria – continuous attributes

$\{x_i^c; i=\overline{1,N}\},\ x_i^c \in R^n, c \in \{1,\dots,k\}$ class label

$w=(w_1,\dots,w_n)$, weight vector

d_w - weighted dissimilarity measure

- Separation (between-class)

$$C_2(w)=\frac{1}{N}\sum_{c=1}^{k} n_c d_w(m_c,m),\ m=\frac{1}{N}\sum_{c=1}^{k} n_c m_c$$

(To be maximized)

- C_2(1,1,1,1,1,1,1,1,1,1,1)=0.51
- C_2(1,1,1,1,1,1,1,1,1,0,0)=0.50
- C_2(1,1,1,1,1,1,1,0,0,0,0)=0.50
- C_2(1,1,1,0,0,0,0,0,0,0,0)=0.49
- C_2(1,1,0,0,0,0,0,0,0,0,0)=0.49
- C_2(1,0,0,0,0,0,0,0,0,0,0)=0.49
- C_2(1,0,0,0,0,0,0,0,0,1,1)=0.50

Example:
Synthetic dataset: 10 attributes, 2 classes

- Attribute 1: identical with the class label
- Attributes 2-6: random values from N(m_1,s_1) (class 1), N(m_2,s_2) (class 2)
- Attributes 7,8: constant values for all instances
- Attributes 9,10: random values from U(a,b) for all instances

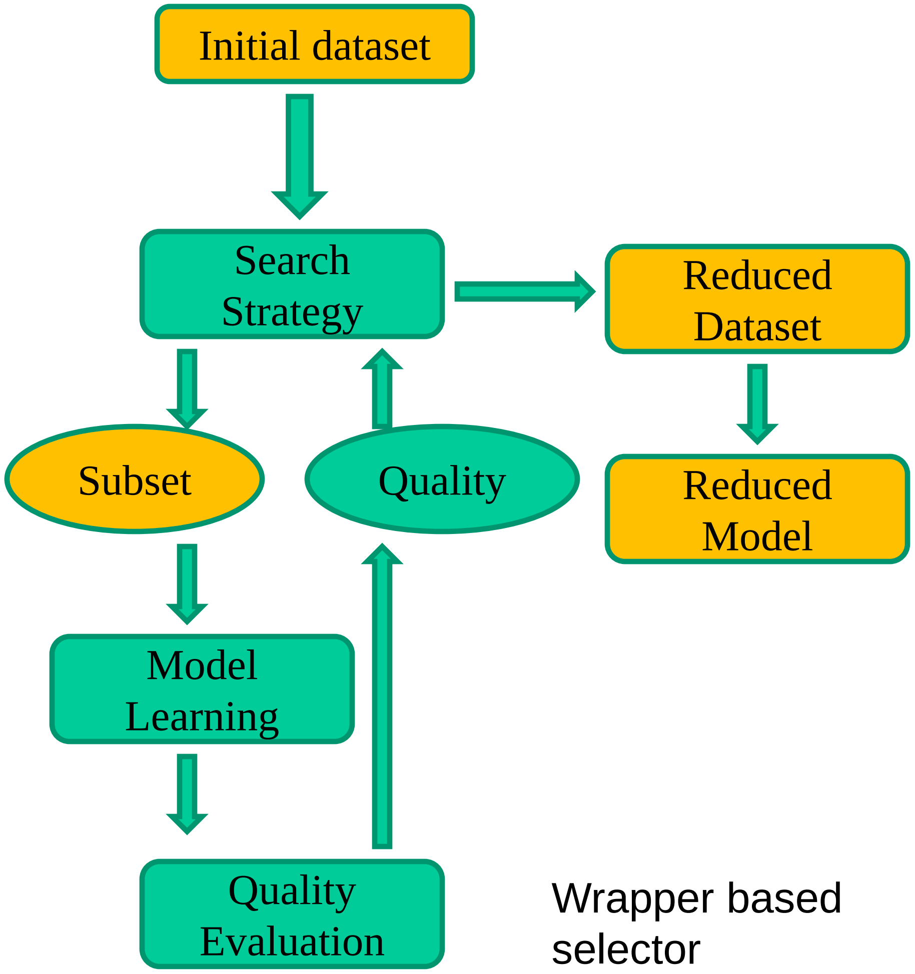
Attribute selection

Wrapper based approach

- Accuracy = number of correctly classified data/ total number of data
- The evaluation of each potential subset needs the full training of the model

Advantage: it uses the impact of the reduced dataset on the classifier performance

Disadvantage: the evaluation cost is rather high



Wrapper based selector

Instance selection

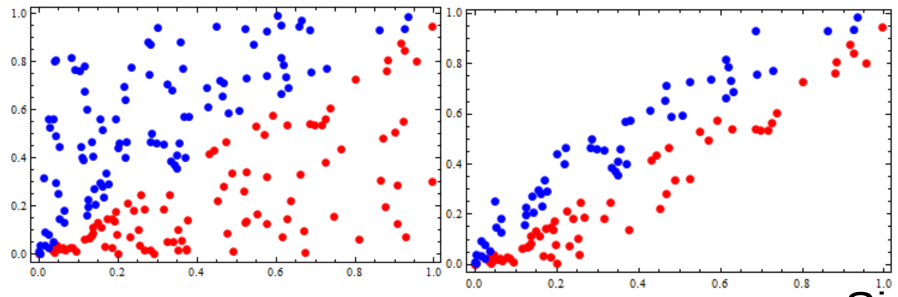
Selection can be applied not only to features but also to instances.

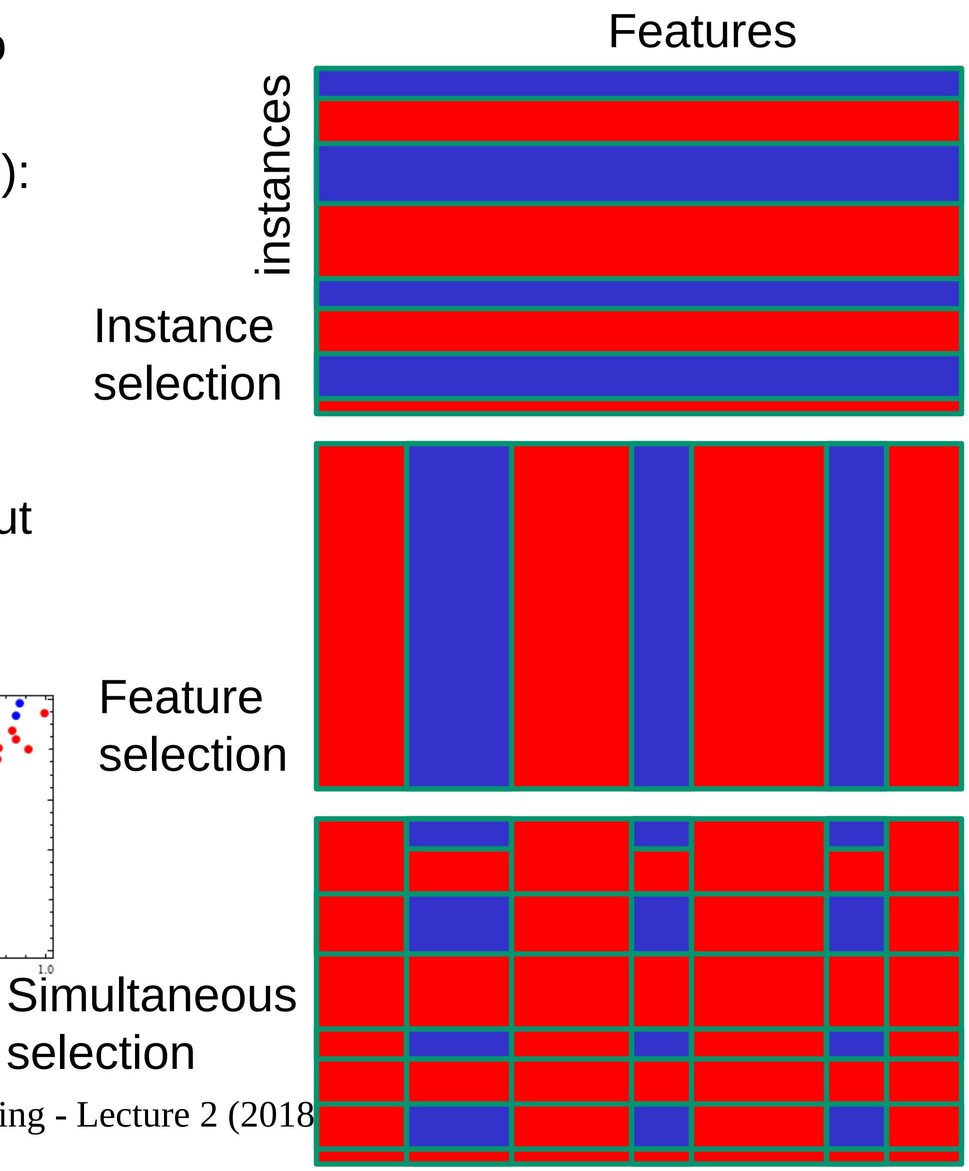
Example (classification in 2 classes):

It would be enough to use only the instances near the border

Approaches:

- Random sampling (with or without replacement)
- Stratified sampling





Attribute transformation

Aim:

- improve the quality of the model extracted from data by removing the bias induced by various scales among features or by the correlation between them

Variants:

- Scaling
- Standardization
- Normalization
- Projection – Principal Component Analysis

Remark: these transformations can be applied only to numerical features

Normalization

Scaling (range normalization):

- Linear scaling: subtract from the value of each feature the minimum and divide by the range
- Not robust with respect to exceptions

Standardization (standard score normalization):

- Subtract the mean and divides by the standard deviation
- More robust than linear scaling

Euclidean normalization:

- Divide the vector by its norm (e.g. the Euclidean norm)

Linear scaling:

$$z_i^j = \frac{x_i^j - \min_j}{\max_j - \min_j}, \quad i=\overline{1,n} \quad j=\overline{1,d}$$

Standardization:

$$z_i^j = \frac{x_i^j - m(X^j)}{s(X^j)}, \quad i=\overline{1,n} \quad j=\overline{1,d}$$

$$m(X^j) = \frac{1}{n}\sum_{i=1}^{n} x_i^j, \quad s(X^j) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i^j - m(X^j))^2}$$

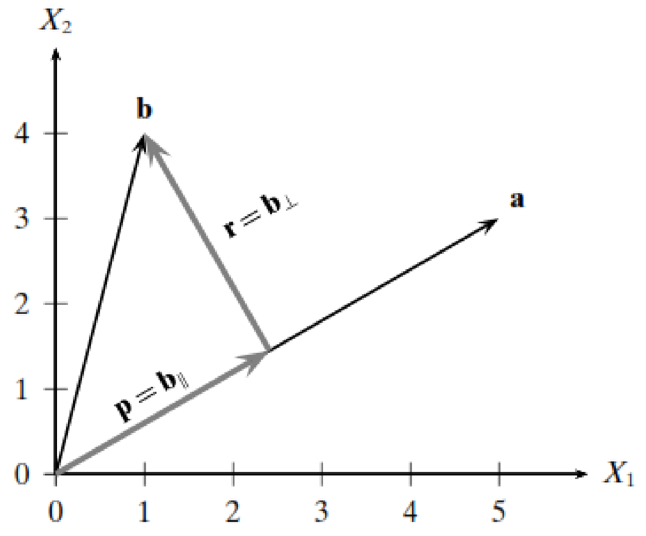
Normalization:

$$Z_i = X_i / \|X\|, \quad \|X\| = \sqrt{\sum_{j=1}^{d}(x_i^j)^2}, \quad i=\overline{1,n}$$

Principal Component Analysis

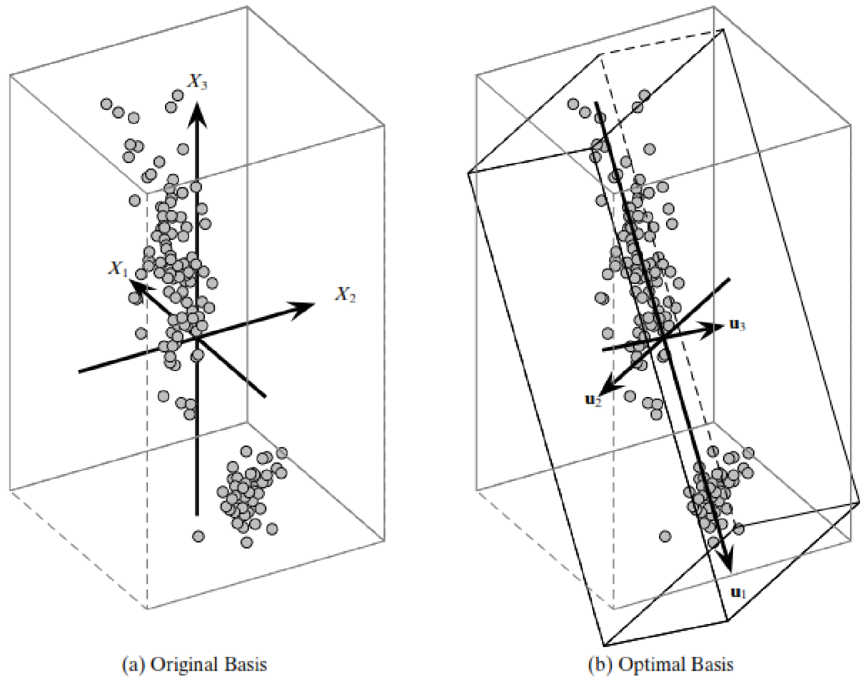
Principal Component Analysis (PCA):

- Project the data on the directions characterized by the largest variability



X_2
X_1
b
a
$\mathbf{r} = \mathbf{b}_\perp$
$\mathbf{p} = \mathbf{b}_\parallel$
0
1
2
3
4
5

Orthogonal projection [Zaki, 2014]



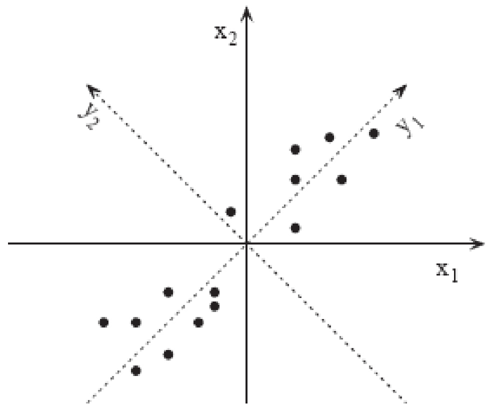
PCA visualization:
http://setosa.io/ev/principal-component-analysis/

Iris dataset – 3D bases [Zaki, 2014]

Principal Component Analysis

Principal Component Analysis (PCA)

Project the data on the directions which capture the most of the variability in the dataset



Input: data set containing N instances having n numerical attributes (data matrix D containing N rows and n columns)

Output: data matrix with N instances having m<n attributes (such that most of the variability from the original dataset is preserved)

Remark:

- PCA concentrates the information about the differences between instances in a small number of attributes
- PCA is used to project a dataset from a n-dimensional space in a dataset from a m-dimensional space such that the attributes in the new space are uncorrelated and most of the data variability is preserved

Principal Component Analysis

Principal Component Analysis (PCA)

Main steps:

- Compute the covariance matrix C (nxn matrix having as elements: C(i,j)=cov(D(i),D(j)), where D(i) is the column i of data matrix D);
- Compute the eigenvalues and the eigenvectors of C and sort the list of eigenvectors decreasingly by their corresponding eigenvalues
- Select the eigenvectors corresponding to the m largest eigenvalues
- Project the dataset D on the hyperspace defined by the m largest eigenvalues

Principal Component Analysis

Principal Component Analysis (PCA) – some statistics and linear algebra

Covariance matrix $C = (c_{ij})_{i=\overline{1,n},\, j=\overline{1,n}}$

$$c_{ij} = \frac{1}{N}\sum_{k=1}^{N}(x_{ki}-\mu_i)(x_{kj}-\mu_j) = \frac{1}{N}\sum_{k=1}^{N}x_{ki}x_{kj} - \mu_i\mu_j,\ \ i=\overline{1,n},\ j=\overline{1,n}$$

μ_i = mean of the values of attribute i, c_{ii} = variance of attribute i

C has n eigenvectors $v_1, v_2, \ldots, \mathrm{v}_{\mathrm{n}}$ corresponding

to n eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_{\mathrm{n}}$

$Cv_i = \lambda_i v_i$ (C transforms its eigenvectors only by scaling)

Remark: C is a semi-positive definite matrix

$(\mathrm{x}^{\mathrm{T}}Cx \geq 0$ for any vector $x) \Rightarrow$ all its eigenvectors are real and positive values

Principal Component Analysis

Principal Component Analysis (PCA) – some statistics and linear algebra

Projection of a dataset on a space defined by one eigenvector (transformation into one-dimensional data)

- If v is a direction (e.g. one of the eigenvectors) then the dataset of one-dimensional projections of the dataset D on v is Dv (product between a matrix with N rows and n columns and a column vector with n elements)
- The covariance of the new data set (it is in fact the variance as the data are one-dimensional)

$$\frac{(Dv)^T(Dv)}{N} - (\mu v)^2 = v^T C v = v^T \lambda v = \lambda \|v\|^2 = \lambda,$$

$$\text{Remark: the eigenvectors are orthonormal}: v_i^T v_j = 0, \ \|v\| = 1$$

- The variance of the one-dimensional projection on an eigenvector is equal to the corresponding eigenvalue, thus in order to capture as most as possible of the variability it should be chosen the largest eigenvalue

Principal Component Analysis

Principal Component Analysis (PCA) – some statistics and linear algebra

Projection of a dataset on a space defined by several eigenvectors

- An important result of linear algebra is:

$C = P\Lambda P^T$ (C can be decomposed based on the eigenvector matrix)

P contains the eigenvectors as columns

P is an orthogonal matrix : $PP^T = I_{n\times n}$

Λ is a diagonal matrix which contains the eigenvalues

- As the eigenvectors are orthogonal they define a new coordinate system
- The projection of the dataset D on the new coordinate system is D'=DP
- Question: which is the covariance matrix corresponding to this new dataset?

Principal Component Analysis

Principal Component Analysis (PCA) – some statistics and linear algebra

Projection of a dataset on a space defined by several eigenvectors

- Question: which is the covariance matrix corresponding to projected dataset?

$$D' = DP, \quad X_k' = P^T X_k$$

$$C' = \frac{1}{N}\sum_{k=1}^{N}(P^T X_k - P^T M)(P^T X_k - P^T M)^T =$$

$$= \frac{1}{N}\sum_{k=1}^{N} P^T (X_k - M)(X_k - M)^T P =$$

$$= P^T C P = P^T P \Lambda P^T P = \Lambda = diag(\lambda_1, \lambda_2, \dots, \lambda_n)$$

Thus the projected data are uncorrelated (the covariance matrix is a diagonal one) and the variance for attribute i is the i-th eigenvalue

Principal Component Analysis

Principal Component Analysis (PCA) – some statistics and linear algebra

Projection of a dataset on a space defined by several eigenvectors

- Question: what happens if we keep only m attributes from the transformed data?
- By keeping m attributes only a fraction from the data variability will be preserved
- Assumption: the eigenvalues are decreasingly sorted
- Procedure: compute the ratio of variances (R) and choose m such that R>threshold (e.g. R>0.95)
- Result: The new dataset (with m attributes obtained by projection on the eigenvectors corresponding to the m largest eigenvalues) captures the most of data variability (e.g. 95%)

$$\text{Proportion of variance} : R = \frac{\sum_{i=1}^{m} \lambda_i}{\sum_{i=1}^{n} \lambda_i}$$

Principal Component Analysis

Example: iris dataset

4 numerical attributes:

A1=sepallength, A2=sepalwidth, A3=petallength, A4=petalwidth,

3 classes

150 instance

Covariance matrix (in R: covMatrix <- cov(iris[,1:4])

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
Sepal.Length	0.6856935	-0.0424340	1.2743154	0.5162707
Sepal.Width	-0.0424340	0.1899794	-0.3296564	-0.1216394
Petal.Length	1.2743154	-0.3296564	3.1162779	1.2956094
Petal.Width	0.5162707	-0.1216394	1.2956094	0.5810063

Principal Component Analysis

Example: iris dataset

4 numerical attributes:

A1=sepallength, A2=sepalwidth, A3=petallength, A4=petalwidth,

3 classes

150 instance

Eigenvalues (in R: eigen(covMatrix)$values)

4.23 0.24 0.078 0.02

the variance explained by the first two components: 4.47/ 4.57 -> 97.7%

Eigenvectors (in R: eigen(covMatrix)$vectors)

	[,1]	[,2]	[,3]	[,4]
[1,]	0.36	-0.65	-0.58	0.31
[2,]	-0.08	-0.73	0.59	-0.31
[3,]	0.85	0.17	0.07	-0.47
[4,]	0.35	0.07	0.54	0.75

New attributes:

tA1: 0.36*A1-0.08*A2+0.85*A3+0.35*A4

tA2: -0.65*A1-0.73*A2+0.17*A3+0.07*A4

Next lecture

Classification models:

- Basic concepts
- Classifiers
  - Simple voting (ZeroR)
  - Simple classification rules (OneR)
  - Decision trees
  - Instance based classifiers (kNN)
- Performance measures