Title:	The Data - Pre–processing and dissimilarities
Course:	Machine Learning
Instructor:	Claudio Sartori
Date:	
Master:	Data Science and Business Analytics
Academic Year:	2022/2023

BOLOGNA BUSINESS SCHOOL

Alma Mater Studiorum Università di Bologna

Data pre–processing

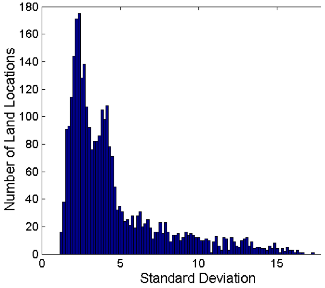
- Aggregation
- Sampling
- Dimensionality Reduction
- Feature subset selection
- Feature creation
- Discretization and Binarization Attribute Transformation

Aggregation

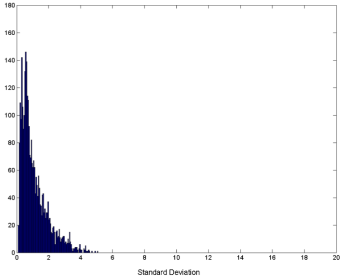
- Combining two or more attributes (or objects) into a single attribute (or object)
- Purpose
  - Data reduction
    - Reduce the number of attributes or objects
  - Change of scale
    - Cities aggregated into regions, states, countries, etc
  - *More stable* data
    - Aggregated data tends to have less variability

Aggregation – Example

Variation of precipitation in Australia



Standard Deviation of
Average Monthly Precipitation



Standard Deviation of
Average Yearly Precipitation

Sampling I

- For both preliminary investigation and final data analysis
- Statistician perspective
  - obtaining the entire data set could be impossibile or too expensive
- Data processing perspective
  - processing the entire data set could be too expensive or time consuming

Sampling II

1. using a sample will work almost as well as using the entire data sets, *if the sample is representative*
2. A sample is representative if it has approximately the same property (of interest) as the original set of data

Types of sampling

1. Simple random
   - a single random choice of an object with given probability distribution
2. With replacement
   - repetition of independent extractions of type 1
3. Without replacement
   - repetition of extractions, extracted element is removed from the population
4. Stratified
   - split data into several partitions according to some criteria, then draw the random samples from each partition
   - used when the data set is split into subsets with homogeneous characteristics
   - the representativity is guaranteed inside each subset

Sample size

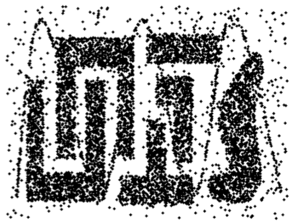
- Statistics provides techniques to assess
  - optimal sample size
  - sample significativity
- Tradeoff between data reduction and precision

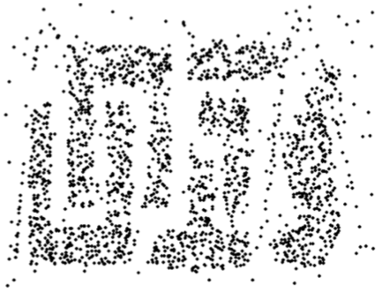
Sampling with/without replacement

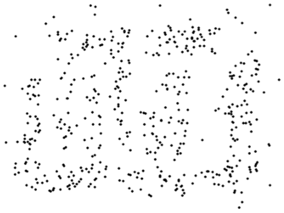
- they are nearly equivalent if sample size is a small fraction of data set size
- with replacement, in a small population a small subset could be underestimated
- sampling with replacement is
  - much easier to implement
  - much easier to be interpreted from a statistical point of view
    - extractions are statistically independent

Sample size – Example

Loss of information


8000 points


2000 points


500 points

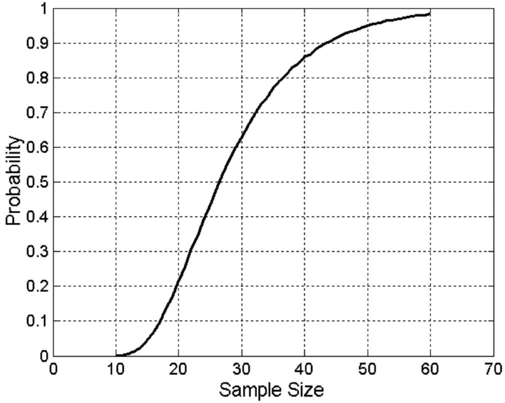
Sample size – Missing class I[1]

- Probability of sampling at least one element for each class (with replacement)
  - it is independent from the size of the data set!

A B C D E

F G H I J

Ten classes



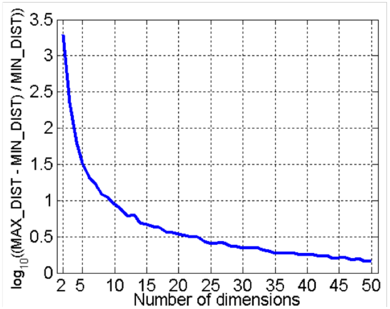
1 It is an instance of the Coupon Collector's Problem

Sample size – Missing class - II

- This aspect becomes relevant, for example, in a supervised dataset with a high number of different values of the target
- If the number data elements is not big enough, it can be difficult to guarantee a stratified partitioning in train/test split or in cross–validation split
- Example:
  - $N = 1000$, $C = 10$, test–set–size $= 300$, cross–validation–folds $= 10$
  - the probability of folds without adequate representation of some classes becomes quite high
- When designing the training processes it is necessary to consider those aspects
  - in the example, one could use only 3 folds in cross–validation

The curse of dimensionality

- When dimensionality is very high the occupation of the space becomes very sparse
- Discrimination on the basis of the distance becomes uneffective
- Experiment:
  - random generation of 500 points
  - plot the relative difference between the maximum and the minimum distance between pairs of points



Dimensionality reduction

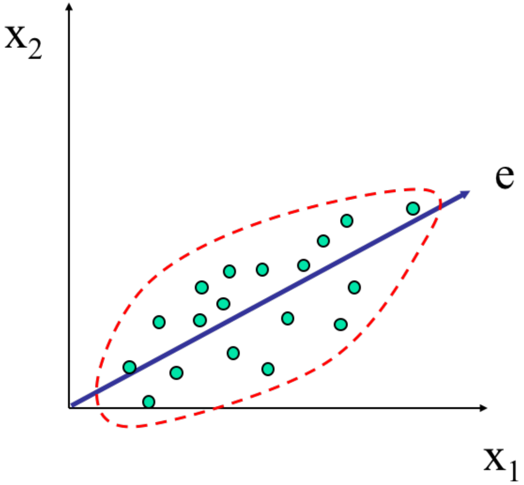
- Purposes:
  - avoid the *curse of dimensionality*
  - noise reduction
  - reduce time and memory complexity of the mining algorithms
  - visualization
- Techniques:
  - principal component analysis
  - singular values decomposition
  - supervised techniques
  - non–linear techniques

PCA

Find projections that capture most of the data variation

- Find the eigenvector of the covariance matrix
- the eigenvectors define the new space

The new dataset will have *only the attributes* which capture most of the data variation



Feature subset selection I

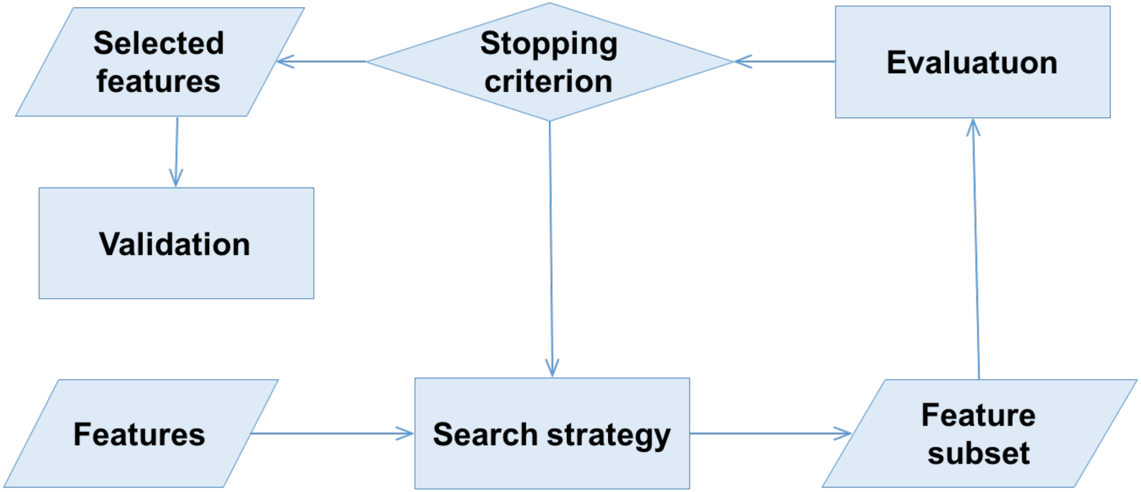
A *local* way to reduce dimensionality

- Redundant attributes
  - duplicate most of the information contained in other attributes
    - e.g. price and v.a.t. amount
- Irrelevant attributes
  - do not contain any information useful for analysis
    - e.g. SSN is not relevant to predict wealth

Feature subset selection II

1. Brute force
   - try all possible feature subsets as input to data mining algorithm and measure the effectiveness of the algorithm with the reduced dataset
2. Embedded approach
   - Feature selection occurs naturally as part of the data mining algorithm
     - e.g. decision trees
3. Filter approach
   - Features are selected before data mining algorithm is run
4. Wrapper approaches
   - A data mining algorithm can choose the best set of attributes
     - try as in 1, but without exhaustive search

An Architecture for Feature Subset Selection

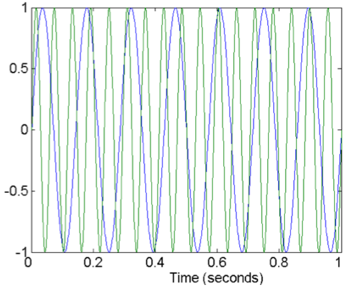


Feature creation

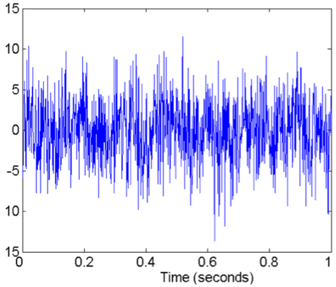
New features can capture more efficiently data characteristics

- Feature extraction
  - pixel picture with a face $\Rightarrow$ eye distance, ...
- Mapping to a new space
  - e.g. signal to frequencies with Fourier transform
- New features
  - e.g. volume and weight to density

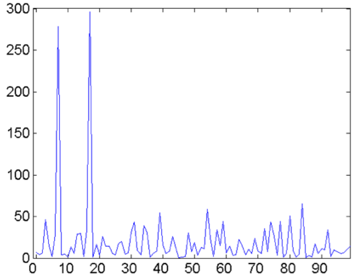
Space transformation example



Two sinusoids



Two sinusoids with noise



Transformation in the domain of frequencies

Why do we need type conversion?

- Many algorithms require numeric features
  - categorical features must be transformed into numeric
  - ordinal features must be transformed into numeric, and the order must be preserved
- Classification requires a target with nominal values
  - a numerical target can be discretised
- Discovery of association rules require boolean features
  - a numerical feature can be discretised and transformed int a series of boolean features

Binarization of discrete attributes

Attribute d allowing V values $\Rightarrow$ V binary attributes.

Quality
Awful
Poor
Ok
Good
Great

Quality–Awful	*Quality–Poor*	*Quality–OK*	*Quality–Good*	*Quality–Great*
1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

Nominal to numeric

- One–Hot–Encoding
  - a feature with V unique values is substituted by V binary features each one corresponding to one of the unique values
  - if object x has value $v$in feature d then the binary feature corresponding to v has `True` for x, all the other binary features have value `False`
  - `True` and `False` are represented as 1 and 0, therefore can be processed by also by procedures working only on numeric data, as is the case for the estimators available in `scikit-learn`
- `sklearn.preprocessing.OneHotEncoder`

link to manual page

Ordinal to numeric

- The ordered sequence is transformed into consecutive integers
  - by default the lexicographic order is assumed
  - The user can specify the proper order of the sequence
- `sklearn.preprocessing.OrdinalEncoder`

link to manual page

Numeric to binary with threshold

- Not greater than the threshold becomes zero
- Greater than the threshold becomes one
- `sklearn.preprocessing.Binarizer`

link to manual page

Discretization/Reduction of the number of distinct values

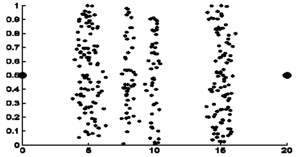
- Some algorithms work better with categorical data
- A small number of distinct values can let patterns emerge more clearly
- A small number of distinct values let the algorithms to be less influenced by noise and random effects

Discretization

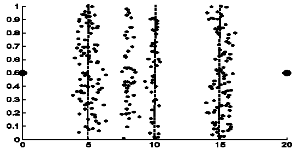
- Continuous ⇒ Discrete
  - thresholds
    - many options
  - binarization ⇒ single threshold
- Discrete with many values ⇒ Discrete with less values
  - guided by domain knowledge

Continuous ⇒ Discrete

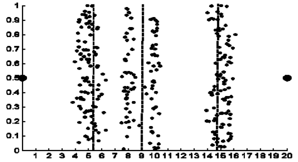
Boundaries on x axis – Unsupervised



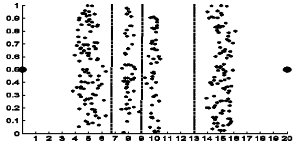
Data



Equal width



Equal frequency



K-means

Numeric to k values

- The numbers are discretised into a sequence of integers 0 to $k-1$
- Several strategies are available
  - `{'uniform','quantile','means'}`
- `sklearn.preprocessing.KBinsDiscretizer`

link to manual page

Similarity and dissimilarity

- Similarity
    - Numerical measure of how alike two data objects are
    - Is higher when objects are more alike
    - Often falls in the range [0,1]
- Dissimilarity
    - Numerical measure of how different are two data objects
    - Lower when objects are more alike
    - Minimum dissimilarity is often 0
    - Upper limit varies
- Proximity refers to a similarity or dissimilarity

Similarity and Dissimilarity by Attribute type

p and q are the values of an attribute for two data objects

Attribute type	Dissimilarity	Similarity
Nominal	$d = \begin{cases} 0 \text{ if } p = q \\ 1 \text{ if } p \neq q \end{cases}$	$s = \begin{cases} 1 \text{ if } p = q \\ 0 \text{ if } p \neq q \end{cases}$
Ordinal Values mapped to integers 0 to V-1	$d = \frac{\lvert p-q \rvert}{V-1}$	$s = 1 - \frac{\lvert p-q \rvert}{V-1}$
Interval or Ratio	$d = \lvert p - q \rvert$	$s = \frac{1}{1+d}$ or $s = 1 - \frac{d - \min(d)}{\max(d) - \min(d)}$

Euclidean distance – L_2

$$\text{dist} = \sqrt{\sum_{d=1}^{D} (p_d - q_d)^2}$$

- Where D is the number of dimensions (attributes) and p_d and q_d are, respectively, the d-th attributes (components) of data objects p and q
- Standardization/Rescaling is necessary if scales differ

Minkowski distance – L_r

$$\text{dist} = \left(\sum_{d=1}^{D} |p_d - q_d|^r \right)^{\frac{1}{r}}$$

- Where D is the number of dimensions (attributes) and p_d and q_d are, respectively, the d-th attributes (components) of data objects p and q
- Standardization/Rescaling is necessary if scales differ
- r is a *parameter* which is chosen depending on the data set and the application

Minkowski distance – Cases

$r = 1$ also named *city block*, *Manhattan*, L_1 norm

- it is the best way to discriminate between zero distance and *near zero* distance
- a ϵ change on any coordinate causes a ϵ change in the distance
- works better than euclidean in very high dimensional spaces

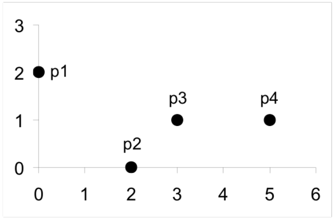
$r = 2$ euclidean, L_2 norm

$r = \infty$ also named Chebyshev, *supremum*, L_{max} norm, L_∞ norm

- considers only the dimension where the difference is maximum
- provides a simplified evaluation, disregarding the dimensions with lower differences

$$\mathrm{dist}_\infty = \lim_{r \to \infty} \left(\sum^{D} |p_d - q_d|^r \right)^{\frac{1}{r}} = \max_d |p_d - q_d|$$

Minkowski distances – Example



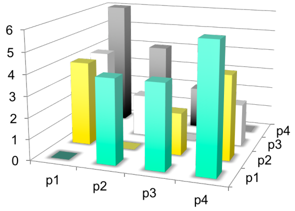
point	*x*	*y*
p1	0	2
p2	2	0
p3	3	1
p4	5	1

L_1	p1	p2	p3	p4
p1	0	4	4	6
p2	4	0	2	4
p3	4	2	0	2
p4	6	4	2	0

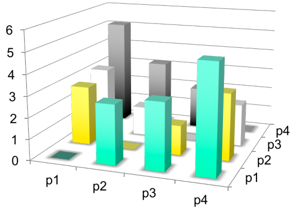
L_2	p1	p2	p3	p4
p1	0	2.828	3.162	5.099
p2	2.828	0	1.414	3.162
p3	3.162	1.414	0	2
p4	5.099	3.162	2	0

L_∞	p1	p2	p3	p4
p1	0	2	3	5
p2	2	0	1	3
p3	3	1	0	2
p4	5	3	2	0

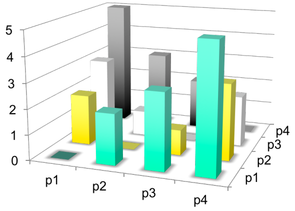
Comparison



L_1



L_2



L_∞

Mahalanobis Distance

Optional

- Considers data distribution
- The Mahlanobis distance between two points p and q decreases if, keeping the same euclidean distance, the segment connecting the points is stretched along a direction of greater variation of data
- The distribution is described by the covariance matrix of the data set

$$\Sigma_{ij} = \frac{1}{N-1}\sum_{k=1}^{N}(x_{ki}-\bar{x}_i)(x_{kj}-\bar{x}_j)$$

$$\mathrm{dist}_m = \sqrt{(p-q)\Sigma^{-1}(p-q)^T}$$

Mahalanobis Distance – Example OPTIONAL

$$\Sigma = \begin{bmatrix} 0.3 & 0.2 \\ 0.2 & 0.3 \end{bmatrix}$$

$$A = (0.5, 0.5)$$

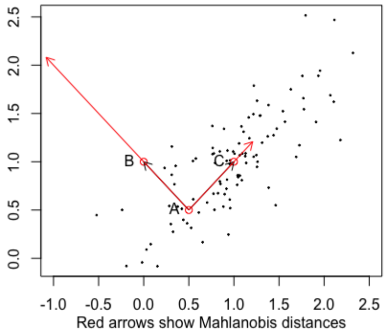
$$B = (0, 1)$$

$$C = (1, 1)$$

The euclidean distances AB and AC are the same

$$\mathrm{dist}_m(A, B) = 2.236068$$

$$\mathrm{dist}_m(A, C) = 1$$



Covariance matrix

OPTIONAL

- Variation of pairs of random variables
- The summation is over all the observations
- The main diagonal contains the variances
- The values are positive if the two variables grow together
- If the matrix is diagonal the variables are non–correlated
- If the variables are standardised the diagonal contains "one"
- If the variables are standardised and non correlated, the matrix is the identity and the Mahalanobis distance is the same as the euclidean

Common properties of a distance

1. Positive definiteness: $\text{Dist}(p, q) \geqslant 0 \ \forall p, q$
   and $\text{Dist}(p, q) = 0$ if and only if $p = q$
2. Symmetry: $\text{Dist}(p, q) = \text{Dist}(q, p)$
3. Triangle inequality: $\text{Dist}(p, q) \leqslant \text{Dist}(p, r) + \text{Dist}(r, q) \forall p, q, r$

A distance function satisfying all the properties above is called a metric

Common properties of a Similarity

1. $\text{Sim}(p, q) = 1$ only if $p = q$
2. $\text{Sim}(p, q) = \text{Sim}(q, p)$

Similarity between binary vectors

- Consider the counts below

M_{00}	the number of attributes where p is 0 and q is 0
M_{01}	the number of attributes where p is 0 and q is 1
M_{10}	the number of attributes where p is 1 and q is 0
M_{11}	the number of attributes where p is 1 and q is 1

- Simple Matching Coefficient

$$\text{SMC} = \frac{\text{number of matches}}{\text{number of attributes}} = \frac{M_{00} + M_{11}}{M_{00} + M_{01} + M_{10} + M_{11}}$$

- Jaccard Coefficient

$$\text{JC} = \frac{\text{number of 11 matches}}{\text{number of non–both–zero attributes}} = \frac{M_{11}}{M_{01} + M_{10} + M_{11}}$$

Cosine similarity

- It is the cosine of the angle between two vectors

$$\cos(p, q) = \frac{p \cdot q}{\|p\| \|q\|}$$

- Example

$$\begin{aligned}
p &= 3\ 2\ 0\ 5\ 0\ 0\ 0\ 2\ 0\ 0 \\
q &= 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 2 \\
p \cdot q &= 3*1 + 2*0 + 0*0 + 5*0 + 0*0 + 0*0 + 0*0 + 2*1 + 0*0 + 0*2 = 5 \\
\|p\| &= \sqrt{3*3 + 2*2 + 0*0 + 5*5 + 0*0 + 0*0 + 0*0 + 2*2 + 0*0 + 0*0} = 6.481 \\
\|q\| &= \sqrt{1*1 + 0*0 + 0*0 + 0*0 + 0*0 + 0*0 + 0*0 + 1*1 + 0*0 + 2*2} = 2.245 \\
\cos(p, q) &= .3150
\end{aligned}$$

Extended Jaccard Coefficient (Tanimoto) OPTIONAL

- Variation of Jaccard for continuous or count attributes
  - reduces to Jaccard for binary attributes

$$\mathrm{T}(p, q) = \frac{p \cdot q}{\|p\|^2 + \|q\|^2 - p \cdot q}$$

Choose the right proximity measure

It depends on data

- Dense, continuous
  - a metric measure, such as the euclidean distance
- Sparse, asymmetric data
  - cosine, jaccard, extended jaccard

Correlation of quantitative data (Pearson's) OPTIONAL

Measure of the linear relationship between a pair of attributes

- Standardize the values
- For two given attributes p and q, consider as vectors the ordered lists of the values over all the data records
- Compute the dot product of the vectors

$$\mathbf{p} = [p_1, \dots, p_N] \xRightarrow{\textit{standardize}} \mathbf{p}'$$

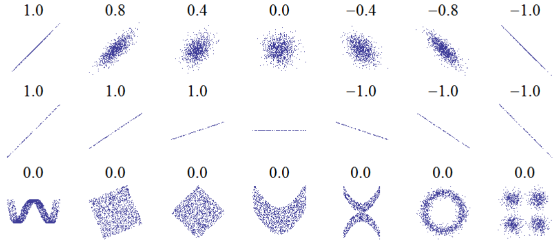
$$\mathbf{q} = [q_1, \dots, q_N] \xRightarrow{\textit{standardize}} \mathbf{q}'$$

$$\mathrm{corr}(p, q) = \mathbf{p}' \bullet \mathbf{q}'$$

There is an alternative definition based on covariance and variances

Correlation – Discussion OPTIONAL

- Independent variables ⇒ correlation is zero
    - the inverse is not valid *in general*
- Correlation zero ⇒ absence of *linear relationship* between the variables
- Positive values imply positive linear relationship



From “Correlation and Dependence” on Wikipedia

Correlation between nominal attributes OPTIONAL

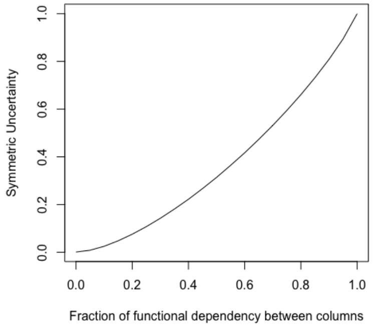
Symmetric Uncertainty [Witten et al.(2011)Witten, Frank, and Hall]

$$U(p, q) = 2\frac{H(p) + H(q) - H(p, q)}{H(p) + H(q)}$$

- where $H()$ is the entropy of a single attribute while $H(,)$ is the joint entropy, computed from the joint probabilities
- is always between 0 and 1

Correlation between nominal attributes – Experiment OPTIONAL

- Behavior of SU for two independent uniformly distributed discrete attributes, say p and q
    - in a variable fraction of records the value of p is copied to q
- from complete independence (left) to complete biunivocal correspondence (right)
- when there is independence, the joint entropy is the sum of the individual entropies, and SU is zero
- when there is complete correspondence, the individual entropies and the joint entropy are equal and SU is one



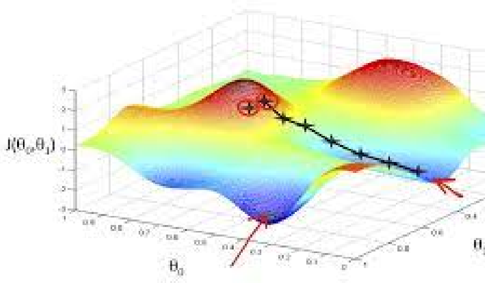
Why Data Transformation

- the features may have different scales
  - this can alterate the results of many learning techniques
  - some machine learning algorithms are sensitive to feature scaling while others are virtually invariant to it
- there can be outliers

Gradient descent

Machine learning algorithms that use *gradient descent* as an optimization technique require data to be scaled

- e.g. linear regression, logistic regression, neural network, etc.
- The presence of feature value X in the formula will affect the step size of the gradient descent
- The difference in ranges of features will cause different step sizes for each feature.
- Similar ranges of the various features ensure that the gradient descent moves smoothly towards the minima and that the steps for gradient descent are updated at the same rate for all the features



Attribute transformation

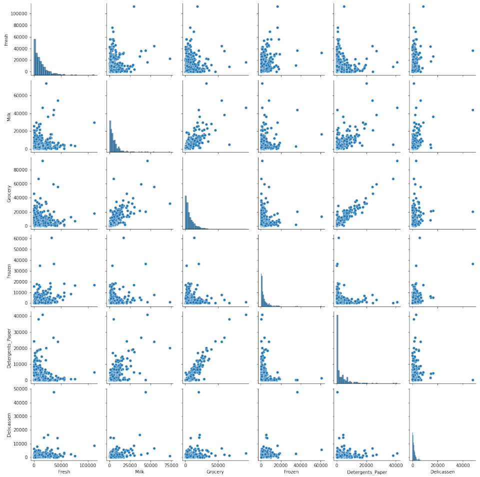
- Map the entire set of values to a new set according to a function
  - $x^k, \log(x), e^x, |x|$
  - in general they change the *distribution of values*
- Standardization: $x \rightarrow \frac{x-\mu}{\sigma}$
  - if the original values have a *gaussian* distribution, the transformed values will have a *standard* gaussian distribution $(\mu = 0, \sigma = 1)$
  - translation and shrinking/stretching, no change in distribution
- MinMax scaling (a.k.a. Rescaling): the domains are mapped to standard ranges

$$x \rightarrow \frac{x - x_{min}}{x_{max} - x_{min}} \quad \text{(0 to 1)} \qquad\qquad x \rightarrow \frac{x - \frac{x_{max}+x_{min}}{2}}{\frac{x_{max}-x_{min}}{2}} \quad \text{(-1 to 1)}$$

  - translation and shrinking/stretching, no change in distribution

Attribute transformation – Example I

Data with skewed distribution

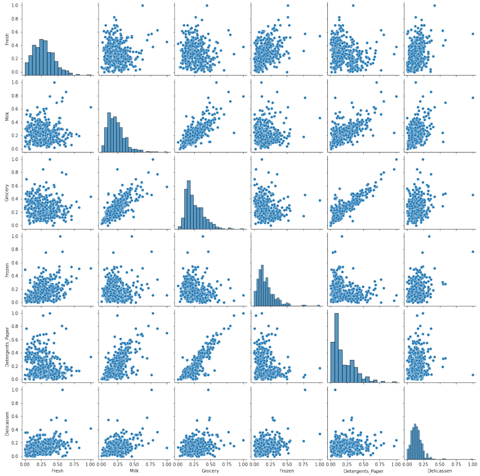


Attribute transformation – Example II

Python code

```
from sklearn.preprocessing
      import PowerTransformer
pt = PowerTransformer(method='box-cox')
X = pd.DataFrame(pt.fit_transform(X0)
                  , columns = X0.columns)
```

After the transformation the data are *less skewed*



Distance–based algorithms

- KNN, K–Means, SVM, . . .
- distances between points are used to determine their similarity

Example

Original data		
Student	**CGPA**	**Salary**
A	3	60
B	3	40
C	4	40
D	4.5	50
E	4.2	52

Scaled data		
Student	**CGPA**	**Salary**
A	-1.18431	1.520013
B	-1.18431	-1.100699
C	0.41612	-1.100699
D	1.21635	0.209657
E	0.736212	0.471728

Distances before and after scaling

$$
\begin{aligned}
distance(A, B) &= \sqrt{(40-60)^2 + (3-3)^2} = 20 \\
distance(B, C) &= \sqrt{(40-40)^2 + (4-3)^2} = 1 \\
distance(A_s, B_s) &= \sqrt{(1.1+1.5)^2 + (1.18-1.18)^2} = 2.6 \\
distance(B_s, C_s) &= \sqrt{(1.1-1.1)^2 + (0.41+1.18)^2} = 1.59
\end{aligned}
$$

Before the scaling the two distances seemed to be very different, due to the a big numeric difference in the `Salary` attribute, now they are comparable

Range–based scaling and standardization

operate on single features

- Range–based scaling stretches/shrinks and translates the range, according to the range of the feature (there are some variants)
    - good when we know that the data are not gaussian, or we do not make any assumption on the distribution
    - the base variant, the MinMax scaler, remaps to $0, 1$
- Standardization subtracts the mean and divides by the standard deviation
    - the resulting distribution has mean *zero* and *unitary* standard deviation
    - good when the distribution is gaussian
    - `StandardScaler`

Range–based scalers in `Scikit-Learn`

affine transformations: linear transformation plus translation

- `MinMaxScaler` – remaps the feature to $[0, 1]$
- `RobustScaler` – centering and scaling statistics is based on percentiles
  - not influenced by a few number of very large marginal outliers
  - the resulting range of the transformed feature values is larger than the one given by `MinMaxScaler` and `StandardScaler`

Normalization

- Normalization is mentioned sometimes with different meanings
  - frequently it refers to `MinMaxScaler`
- in `Scikit-learn` the `Normalizer` normalizes each data row to unit norm

Workflow

1. transform the features as required both for the train and test data
2. fit and optimize the model(s)
3. test
4. possibly, use the original data to plot relevant views (e.g. to plot cluster assignments)

Imbalanced data in classification

- The performance minority class (classes) has little impact on standard performance measures
- The optimised model could be less effective on minority class (classes)
- Some estimators allow to weight classes
- Some performance measures allow to take into account the contribution of minority class (classes)

Cost Sensitive learning

Already introduced in *Machine-Learning-03-classification*

- several classifiers have the parameter `class_weight`
- it changes the cost function to take into account the imbalancing of classes
- in practice it is equivalent to oversampling the minority class, (repeating random examples) in order to produce a balanced training set

Undersampling

- Obtains a balanced training set by randomly reducing the number of examples of the majority class
- Obviously part of the knowledge embedded in the training set is dropped out

Oversampling with SMOTE

Synthetic Minority Oversampling Technique – a type of data augmentation

let the minority class be c_{min}, synthesise new examples of class c_{min}

- choose from the *training set* random example x_r of class c_{min}
- find in the training set the k nearest neighbours of x_r whose class is c_{min}
- choose randomly one of the neighbours, say x_{rn} found above and *create* a new data element chosen randomly from the segment connecting x_r x_{rn} in the feature space $m = r * (x_r + x_{rn})/2$

Theory developed in SMOTE: Synthetic Minority Over-sampling Technique[Bowyer et al.(2011)Bowyer, Chawla, Hall, and Kegelmeyer]

Workflow for *undersampling/oversampling*

- resample the training set
- fit and optimise the estimator
- test the fitted estimator on the test set (untouched)

Bibliography I

- Kevin W. Bowyer, Nitesh V. Chawla, Lawrence O. Hall, and W. Philip Kegelmeyer.
  SMOTE: synthetic minority over-sampling technique.
  CoRR, abs/1106.1813, 2011.
  URL http://arxiv.org/abs/1106.1813.
- Ian H. Witten, Eibe Frank, and Mark Hall.
  Data Mining – Practical Machine Learning Tools and Techniques.
  Morgan Kaufman, 2011.
  ISBN 978-0-12-374856-0.