Data preparation

Mario Martin

Universitat Politcnica de Catalunya

mmartin@cs.upc.edu

September 30, 2019

Overview

1. Data preparation
2. Data processing: columns
   - Analysis of columns: Uni-variate and Bi-variate
   - Finding errors
   - Finding outliers
   - Missing data
   - Enriching the data set: Joining tables
   - Enriching the data set: Building new variables
   - Transforming values of columns
   - Reducing number of columns: feature selection and feature extraction
3. Data processing: rows
   - Reducing the number of rows
   - Finding outliers
4. Observations about the project

Data preparation

Data preparation

- When facing a data mining problem, a great effort has to be done in the preparation of your data for the machine learning algorithms.
- Acquisition of data usually involves database, scrapping or retrieval techniques needed to obtain the data.
- We will focus on how to pre-process the raw data obtained from these techniques.
- Usually the raw data contains errors, redundant information, poor description of data, etc. Data has be *tidied*.

Data processing: columns

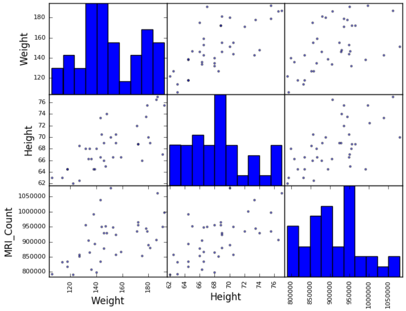
Analysis of columns: Uni-variate and Bi-variate

Analysis of columns: Uni-variate

- First step consist in *understanding* your variables, so you have to study them.
- Uni-variate analysis of a column consist in finding basic statistic description of each variable.
- For quantitative variables, max, min, median, mode, standard deviation, distribution (plot), (boxplot), etc.
- For qualitative variables: Number of modalities, histogram.
- Useful to detect outliers, errors, etc.

Analysis of columns: Uni-variate

```
>>> import pandas
>>> data = pandas.read_csv('examples/brain_size.csv', sep=';', na_values=".")
>>> data
   Unnamed: 0  Gender  FSIQ  VIQ  PIQ  Weight  Height  MRI_Count
0           1  Female   133  132  124     118    64.5     816932
1           2    Male   140  150  124     NaN    72.5    1001121
2           3    Male   139  123  150     143    73.3    1038437
3           4    Male   133  129  128     172    68.8     965353
4           5  Female   137  132  134     147    65.0     951545
...
>>> from pandas.tools import plotting
>>> plotting.scatter_matrix(data[['Weight', 'Height', 'MRI_Count']])
```



Analysis of columns: bi-variate

- Consist in plotting one variable against other
- If both variables are quantitative, scatterplot, regression and correlation value
- If one variable is qualitative and the other quantitative, boxplot for each modality
- If both variables are qualitative, heatmap of each pair of modalities and χ^2 comparison

Analysis of columns: bi-variate

- Consist in plotting one variable against other
- If both variables are quantitative, scatterplot, regression and correlation value
- If one variable is qualitative and the other quantitative, boxplot for each modality
- If both variables are qualitative, heatmap of each pair of modalities and χ^2 comparison
- Useful to detect correlations, anti-correlations, redundancies, and so understand your data

Analysis of columns: bi-variate

- Consist in plotting one variable against other
- If both variables are quantitative, scatterplot, regression and correlation value
- If one variable is qualitative and the other quantitative, boxplot for each modality
- If both variables are qualitative, heatmap of each pair of modalities and χ^2 comparison
- Useful to detect correlations, anti-correlations, redundancies, and so understand your data
- Theoretically this can be extended to more than bi-variate analysis, but number of comparisons grows exponentially with number of columns compared and also become harder to interpret and visualize

Finding errors

Finding errors

- Errors consist in data that does not corresponds to reality. Different possible causes (in measure, transcription, etc.).

Finding errors

- Errors consist in data that does not corresponds to reality. Different possible causes (in measure, transcription, etc.).
- Easy to detect wrong values when they are impossible (f.i. negative price, too high age, etc.)
- This kind of errors can be detected because they are out of range, so they can be detected using uni-variate analysis of columns

Finding errors

- Other erroneous values can be detected because they are unlikely considering the information of another correlated variable (f.i. age considering also number of descendants)
- This kind of errors can be detected because they are out of the regression line for both variables, so they can be detected using bi-variate analysis of correlated columns

Finding errors

- Other errors are hard to find because the wrong value is plausible (so don't expect to obtain a completely clean dataset: your algorithms will have to deal with noisy data.)
- When an error is found we can do the following:
  1. Nothing
  2. Remove the row or the column with the error (specially when there are a lot of errors in row or column)
  3. Replace the value using missing data techniques

Finding outliers

Finding outliers

- Outliers consist in right data so different from normality that seems an error
- Detection can be done using same tools like error detection.
  But, what to do with them? You should keep them, because they are actual data, but...
- ... some data mining algorithms can be fooled because of this outliers (this also happens with basics statistic measures, for instance the mean / mode measures)
- So, in most cases is better to remove the row (be cautious)

Missing data

Missing data

- Frequently, data for one column and row is not defined. There are possible causes:
  1. Data not applicable
  2. Data not recorded or lost

Missing data

- Frequently, data for one column and row is not defined. There are possible causes:
    1. Data not applicable
    2. Data not recorded or lost
- There are several ways to deal with missing data:
    1. Substitute the data with an special value (unknown or 0)
    2. Substitute the data with something reasonable: *imputation*
    3. Remove the column or row (caution: it's last choice)

Missing data: imputation methods

- Popular ways to do imputation:
  1. Substitute the missing data with the average of the column or the most frequent modality of the column
  2. Impute the missing value with the value of the most similar case in training data (it can be extended to the average of k most similar cases)
  3. When the row with the missing data is known to be linearly correlated with another column, impute the missing data with the corresponding regression equation between both columns

Enriching the data set: Joining tables

Enriching the data set: Joining tables

- It is common to have information stored in data bases
- Databases, for instance relational methods, store information in separate tables.
- So you can complete (enrich) your dataset with corresponding data from another table

Enriching the data set: Joining tables

- It is common to have information stored in data bases
- Databases, for instance relational methods, store information in separate tables.
- So you can complete (enrich) your dataset with corresponding data from another table

- For instance think in a dataset about pulmonary diseases. It has several columns plus the identifier of the patient.
- In other dataset you have information about each patient. For instance residence
- Joining both tables you now can relate features of the first dataset with residence, for instance.

Enriching the data set: Building new variables

Enriching the data set: Building new variables

- Sometimes your data is expressed in columns that it is difficult to manipulate with your data mining algorithm. So you add define new columns from the old ones
- For instance think in a dataset about taxis, with columns like departure point, destination point, duration of trip.
- You can add to to the data set new columns: length of trip, speed of taxi
- So you can complete (enrich) your dataset with columns that ease the finding of a good model

Transforming values of columns

Transforming values of columns

- From qualitative to numerical

Transforming values of columns

- From qualitative to numerical: Not right because it implies different distances between modalities

	...	*color*	...
o_1	...	red	...
o_2	...	blue	...
o_3	...	brown	...
o_4	...	red	...
o_5	...	green	...
...	...	...	...

$\not\Rightarrow$

	...	*color*	...
o_1	...	1	...
o_2	...	2	...
o_3	...	3	...
o_4	...	1	...
o_5	...	4	...
...	...	...	...

Transforming values of columns

- From qualitative to numerical: Create *dummy* variables, one for modality

	...	*color*	...
o_1	...	red	...
o_2	...	blue	...
o_3	...	brown	...
o_4	...	red	...
o_5	...	green	...
...	...	...	...

$\Longrightarrow$

	...	*red?*	*blue?*	*brown?*	*green?*	...
o_1	...	1	0	0	0	...
o_2	...	0	1	0	0	...
o_3	...	0	0	1	0	...
o_4	...	1	0	0	0	...
o_5	...	0	0	0	1	...
...	...	...	...	...	...	...

- When to many modalities, for some algorithms, you have to define weight $1/|mods|$ for each dummy variable to reduce the impact of this attribute in computation of distances.

Transforming values of columns

- From qualitative to numerical
- Reducing number of values:
  - From numerical to ranges (f.i. age: 0-18,19-40,40-65,67-120)
  - From numerical to qualitative (f.i. age: young, adult, senior, old)
  - Grouping categorical values when there is a hierarchical structure over them (f.i. kind of illness)
- Normalization or Standarization (p. 28)
- Change of units
- Log of the column

Transforming values of columns

- From qualitative to numerical
- Reducing number of values:
- Normalization or Standarization:
  - Mandatory for some algorithms like *k-NN* or *SVM*
  - Normalization $[0-1]$:

$$v(i) = \frac{v(i) - min(v)}{max(v) - min(v)}$$

  - Standarization $N(0,1)$:

$$v(i) = \frac{v(i) - mean(v)}{std(v)}$$

Transforming values of columns

- From qualitative to numerical
- Reducing number of values:
- Normalization or Standarization
- Other changes:
  - Change of units
  - Log of the column

Reducing number of columns: feature selection and feature extraction

Removing variables

- Very common in DM
- Why dimensionality reduction?
    1. Reduced Computing Time
    2. Increased predictive/descriptive accuracy
    3. Better representation of the data-mining model
    4. The intrinsic dimension may be small.
- Some data mining techniques may not be effective for high-dimensional data (they suffer the *Curse of dimensionality*: accuracy and efficiency may degrade rapidly as the dimension) increases.

Reducing number of columns

Two ways to reduce the number of columns:

1. **Feature selection**: A process that chooses an optimal subset of features according to an objective function
    - Feature ranking algorithms, and
    - Minimum subset algorithms.
2. **Feature extraction**: refers to the mapping of the original high-dimensional data onto a lower-dimensional space.
    - Descriptive setting: minimize the information loss
    - Predictive setting: maximize the class discrimination

Removing variables: Feature selection

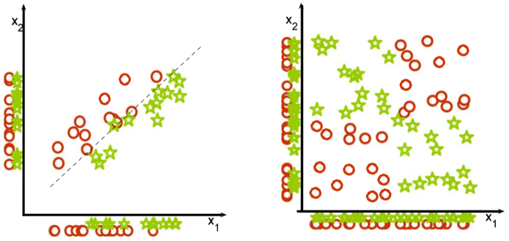
Feature selection: Filter and feature ranking algorithms.

- Remove columns without enough variability ($std(v) < threshold$)
- Remove variables that are not enough correlated with label ($corr(v, label) < threshold$)
- Remove variables without enough Information Gain ($IG(v) < threshold$)
- Remove variables without enough Relief [Paper] ($Relief(v) < threshold$)

All these methods implicitly build a ranking of features using a given measure. After ranking, we use a threshold to select the interesting features.

Removing variables: Feature selection

Rank and cut methods could miss joint effects of variables:

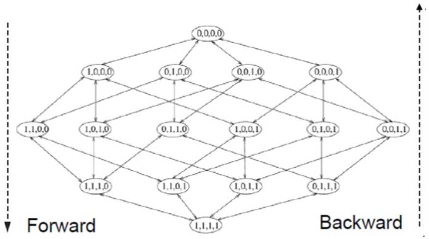


Guyon-Elisseeff, JMLR 2004; Springer 2006

Removing variables: Feature selection

Instead of evaluating features isolated, we will evaluate sets of features:

- It can be viewed as a search problem (2^n sets of features to consider)



- An optimal search is not feasible, and simplifications are made to produce acceptable and timely reasonable results:
  - heuristic criteria
  - bottom-up approach
  - top-down approach

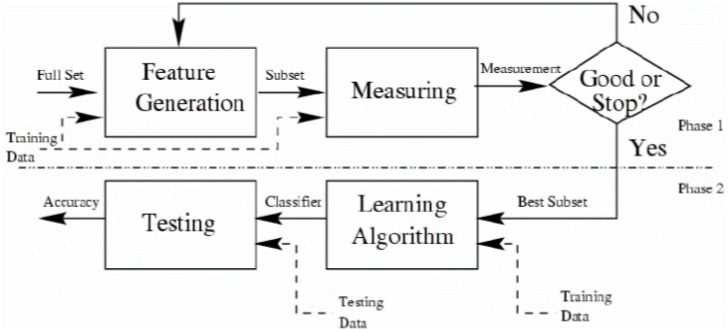
Removing variables: Feature selection

Advanced feature selection methods have the following elements:

- A Feature set generator that proposes a set of features to evaluate
- An evaluation method. Two kinds of evaluation methods:
  1. *Filter*: Uses an evaluation metric that is independent of the learning mechanism that will be used (f.i. the ones we have seen in the previous slides)
  2. *Wrapper* method: Uses a measure of the target learning method to evaluate the set of features (f.i. in SVM, the margin of the classifier obtained with that set of features)
- A criteria to stop the search for the set of features

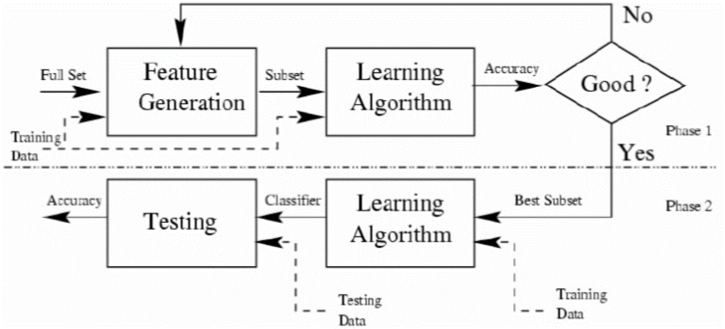
Removing variables: Feature selection

Filter method:



Removing variables: Feature selection

Wrapper method:



Reducing number of columns

Two ways to reduce the number of columns:

1. **Feature selection**: A process that chooses an optimal subset of features according to an objective function
    - Feature ranking algorithms, and
    - Minimum subset algorithms.
2. **Feature extraction**: refers to the mapping of the original high-dimensional data onto a lower-dimensional space.
    - Descriptive setting: minimize the information loss
    - Predictive setting: maximize the class discrimination

Reducing number of columns: Feature extraction

- **Feature extraction**: refers to the mapping of the original high-dimensional data onto a *lower-dimensional space*.
- The new dimensions (columns) are a *new* and should contain the information of the original dataset in a lower number of columns.
- Examples:
  - **PCA** (descriptive setting): minimize the information loss embedding the dataset in a space of new columns that are build as linear combinations of the original columns. Columns are sorted by inertia, and we keep only the *n* most informative columns.
  - **LDA** (predictive setting): does like in the case of PCA an embedding in a dimensional space of columns that are build as linear combinations of the original columns, but in this case the goal is to maximize the class discrimination instead of minimal information loss.

Data processing: rows

Reducing the number of rows

Reducing the number of rows

- Most DM algorithms have time complexity cost $O(n^2)$ (f.i. kNN) or even $O(n^3)$ (f.i. SVM), where n is the number of rows in the dataset.
- Some algorithms need to store matrices of distances between elements, so we need $O(n^2)$ space.
- So, in some datasets we could have too many rows to build a model in reasonable time
- ... but maybe not all rows are necessary to build a model.
- So we could try to reduce the number of elements using a *sampling* procedure

Reducing the number of rows

Some questions appear when considering sampling:

- How many examples should I sample?
- Which kind of sampling should I use
- Am I sure that the sampling was successful?

Sampling procedure

Which kind of sampling should I use? Different kinds of sampling

- *Systematic sampling*: get every k-th sample in the dataset. For instance if I want 50% sampling, get one and the next one no.
- **Problem!**: when there are regularities in data set (data set sorted, for instance)

Sampling procedure

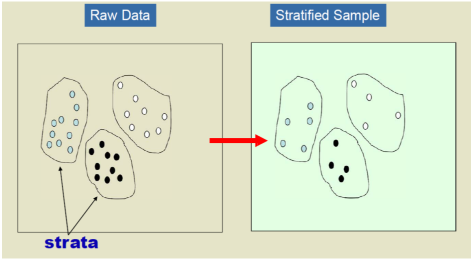
Which kind of sampling should I use? Different kinds of sampling

- *Systematic sampling*: get every k-th sample in the dataset. For instance if I want 50% sampling, get one and the next one no.
- **Problem!**: when there are regularities in data set (data set sorted, for instance)
- *Random sampling*: Two different choices
  - random sampling without replacement,
  - random sampling with replacement.
- **Problem!**: Is my sampling representative of the complete dataset?

Sampling procedure

Which kind of sampling should I use? Different kinds of sampling

- *Stratified sampling*: Split data set into non-overlapping subsets (strata) and combine strata results.
- Used in general to ensure same proportion of each class of examples in the sample.



- **Problem!**: Ok. Class values are representative, but What happens with other columns?

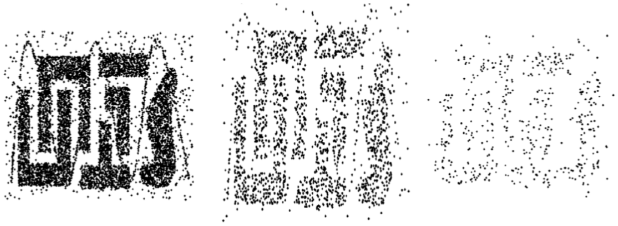
Sampling procedure

Which kind of sampling should I use? Different kinds of sampling

- *Multi-Stratified sampling*: Do a Stratified sampling, and after that, *verify* that for each column we have a similar statistical structure to the original dataset (histogram of modalities in qualitative features, mean for numerical values). If the sampled set does not pass the test, throw it and do another sampling until one that passes the test is found.

Number of examples

How many examples should I sample?



8000 points **2000 Points** **500 Points**

- Error should be not different than using complete dataset.
- One solution could be **incremental sampling**: In consists in starting with a small sample, measure the accuracy with the learning algorithm and increase the size of the sampling until no improvement in accuracy is found.

Number of examples

How many examples should I sample?

- Data should be representative of the original dataset.
- **Inverse sampling**: Used when features with rare frequencies of values (skewed distributions): Consists in the dynamic increase of examples sampled until some condition about feature are satisfied (f.i. all values appear at least t times)

Finding outliers

Finding outliers

- Some outliers can be found with uni-variate and bi-variate analysis of columns, but some others escape of this analysis.
- Outliers can also be found analyzing rows (instances) instead of columns.
- Key concept is that one example is one outlier when its *far away of the other examples* or is located in a *zone of low density*.
- Two kind of methods:
  - **Distance approach.** An instance o in a data set S is an outlier if the its *k-th nearest neighbor* is at a too high distance when compared with rest of examples of S.
  - **Density approach**: An instance o in a data set S is an outlier if at least a fraction p of the samples in S lies at a distance greater than d

Observations about the project

Observations about the project (I)

- In the project you have to test some algorithm for which the best parameters have to be found.
- Remember that you should divide your dataset into developement and validation.
- Development dataset will be used for finding the best parameters using, f.i. a 10-fold cross-validation (that automatically separates data into training and test).
- After finding the best parameters, you should build a classifier training with *all* the data in the developer dataset and later apply the model learnt to the validation dataset. You should show the results in that validation dataset.
- To do a fair comparison of algorithms, you should test all them on the same partition of development and validation.

Observations about the project (II)

- In some case you will have an unbalanced dataset (the amount of examples of one class is a lot higher than the other class.
- Remember that in this case you should not report results of accuracy, but results about precision, recall and F-measure on the smaller class.

Observations about the project (II cont.)

- Still with unbalanced datasets, in some cases is extremely difficult to obtain good results because the smaller class is a lot underrepresented. In this case you can apply one of the following tricks:
    - Use a cost function when building the classifier to penalize different kinds of errors in the confusion matrix
    - Force a more similar representation of the smaller class (by removing examples of the more populated class or by repeating in the dataset several times the examples of the less common class).
    - Generate new examples of the smaller class. For instance, choose randomly two examples of the smaller class and compute the average between them. This average is included as a example of the negative class. Repeat until you have enough examples of the smaller class.