Data Transformation

Lecture 7

Self-study

Marina Santini

Acknowledgements
Slides borrowed and adapted from:
Data Mining by I. H. Witten, E. Frank and M. A. Hall

Outline

- Discretization
- From nominal to numeric
- From string to vector
- Data cleansing
- Anomalies

Just apply a learner? NO!

- Scheme/parameter selection
  treat selection process as part of the learning process
- Modifying the input:
  - Data engineering to make learning possible or easier

Attribute selection

- Adding a random **(i.e. irrelevant)** attribute can significantly degrade C4.5's performance
  - Problem: attribute selection based on smaller and smaller amounts of data
- Some algorithms are very susceptible to irrelevant attributes
  - Number of training instances required increases exponentially with number of irrelevant attributes
- Naïve Bayes doesn't have this problem
- *Relevant* attributes can also be harmful

Attribute discretization

- Avoids normality assumption in Naïve Bayes and clustering
- Some implementations apply discritization automatically. For ex:

- C4.5 performs *local* discretization

- *Global* discretization can be advantageous because it's based on more data

The converse of discretization

- Make nominal values into “numeric” ones

Transformation

- Simple transformations can often make a large difference in performance
- Example transformations (not necessarily for performance improvement):
  - Encoding cluster membership
  - Adding noise to data
  - Removing data randomly or selectively
  - Obfuscating the data
  - etc.

Text to attribute vectors

- Many data mining applications involve textual data (eg. string attributes in ARFF)
- Standard transformation: convert string into bag of words by *tokenization*
  - Attribute values are binary, word frequencies (f_{ij}), $\log(1+f_{ij})$, or TF × IDF:

$$f_{ij} \log \frac{\#\ documents}{\#\ documents\ that\ include\ word\ i}$$

- Only retain alphabetic sequences?
- What should be used as delimiters?
- Should words be converted to lowercase?
- Should *stopwords* be ignored?
- Should *hapax legomena* be included? Or even just the *k* most frequent words?

Automatic data cleansing

- To improve a decision tree:
  - Remove misclassified instances, then re-learn!
- Better (of course!):
  - Human expert checks misclassified instances
- Attribute noise vs class noise
  - Attribute noise should be left in training set *(don't train on clean set and test on dirty one)*
  - Systematic class noise (e.g. one class substituted for another): leave in training set
  - Unsystematic class noise: eliminate from training set, if possible

Detecting anomalies

- Visualization can help to detect anomalies
- Automatic approach:
  committee of different learning schemes
  - E.g.
    - decision tree
    - nearest-neighbor learner
    - linear discriminant function
  - Conservative approach: delete instances incorrectly classified by all of them
  - Problem: might sacrifice instances of small classes

The end