Feature selection

LING 572
Fei Xia

Creating attribute-value table

	f_1	f_2	…	f_K	y
x_1					
x_2					
…					

- Choose features:
  - Define feature templates
  - Instantiate the feature templates
  - Dimensionality reduction: feature selection

- Feature weighting
  - Global feature weighting: weight the whole column
  - Local feature weighting: weight for a cell

Feature Selection Example

- Task: Text classification
- Feature template definition:
  - Word – just one template
- Feature instantiation:
  - Words from training data
- Feature selection:
  - Stopword removal: remove top K (~100) highest freq
    - Words like: the, a, have, is, to, for,…
- Feature weighting:
  - Apply tf*idf feature weighting
    - tf = term frequency; idf = inverse document frequency

The Curse of Dimensionality

- Think of the instances as vectors of features
  - # of features = # of dimensions

- Number of features potentially enormous
  - e.g., # words in corpus continues to increase w/corpus size

- High dimensionality problematic:
  - Leads to difficulty with estimation/learning
    - Hard to create valid model
    - Hard to predict and generalize – think kNN
    - More dimensions → more samples needed to learn model

  - Leads to high computational cost

Breaking the Curse

- Dimensionality reduction:
  - Produce a representation with fewer dimensions
    - But with comparable performance

  - More formally, given an original feature set r,
    - Create a new set r' |r'| < |r|, with comparable performance

Outline

- Dimensionality reduction
- Some scoring functions **
- Chi-square score and Chi-square test

In this lecture, we will use "term" and "feature" interchangeably.

Dimensionality reduction (DR)

Dimensionality reduction (DR)

- What is DR?
  - Given a feature set r, create a new set r', s.t.
    - r' is much smaller than r, and
    - the classification performance does not suffer too much.

- Why DR?
  - ML algorithms do not scale well.
  - DR can reduce overfitting.

Dimensionality Reduction

- Given an initial feature set r,
  - Create a feature set r’ such that |r| < |r’|

- Approaches:
  - r’: same for all classes (a.k.a. global), vs
  - r’: different for each class (a.k.a. local)

  - Feature selection/filtering
  - Feature mapping (a.k.a. extraction)

Feature Selection

- Feature selection:
  - r' is a subset of r
  - How can we pick features?
    - Extrinsic 'wrapper' approaches:
      - For each subset of features:
        - » Build, evaluate classifier for some task
      - Pick subset of features with best performance

    - Intrinsic 'filtering' methods:
      - Use some intrinsic (statistical?) measure
      - Pick features with highest scores

Feature Selection

- Wrapper approach:
  - Pros:
    - Easy to understand, implement
    - Clear relationship between selected features and task performance.
  - Cons:
    - Computationally intractable: $2^{|r'|}$*(training + testing)
    - Specific to task, classifier
- Filtering approach:
  - Pros: theoretical basis, less task, classifier specific
  - Cons: Doesn't always boost task performance

Feature selection by filtering

- Main idea: rank features according to predetermined numerical functions that measure the “importance” of the terms.

- It is fast and classifier-independent.

- Scoring functions:
  - Information Gain
  - Mutual information
  - chi square
  - …

Feature Mapping

- Feature mapping (extraction) approaches
  - r’ represents combinations/transformations of features in r
    - Ex: many words near-synonyms, but treated as unrelated
    - Map to new concept representing all
      - big, large, huge, gigantic, enormous → concept of ‘bigness’

  - Examples:
    - Term classes: e.g. class-based n-grams
      - Derived from term clusters
    - Latent Semantic Analysis (LSA/LSI)
      - Result of Singular Value Decomposition (SVD) on matrix produces ‘closest’ rank r’ approximation of original

Feature Mapping

- Pros:
  - Data-driven
  - Theoretical basis – guarantees on matrix similarity
  - Not bound by initial feature space

- Cons:
  - Some ad-hoc factors:
    - e.g., # of dimensions
  - Resulting feature space can be hard to interpret

Quick summary so far

- DR: to reduce the number of features
  - Local DR vs. global DR
  - Feature extraction vs. feature selection

- Feature extraction:
  - Feature clustering
  - Latent semantic indexing (LSI)

- Feature selection:
  - Wrapping method
  - Filtering method: different functions

Feature scoring measures

Basic Notation, Distributions

- Assume binary representation of terms, classes
- t_k : term in T; c_i: class in C

- $P(t_k)$: proportion of documents in which t_k appears
- $P(c_i)$: proportion of documents of class c_i
  - Binary so have

$$P(\overline{t}_k), P(\overline{c}_i)$$

$$P(t_k, c_i), P(\overline{t}_k, c_i), etc....$$

Calculating basic distributions

	$\bar{c_i}$	c_i
$\bar{t_k}$	a	b
t_k	c	d

$$P(t_k, c_i) = d/N$$

$$P(t_k) = (c+d)/N, P(c_i) = (b+d)/N$$

$$P(t_k|c_i) = d/(b+d)$$

where $N = a + b + c + d$

Feature selection functions

- Question: What makes a good feature?

- Intuition: for a category c_i , the most valuable feature are those that are distributed most <u>differently</u> in the sets of positive and negative examples of c_i.

Term Selection Functions: DF

- Document frequency (DF):
  - Number of documents in which t_k appears
- Applying DF:
  - Remove terms with DF below some threshold
- Intuition:
  - Very rare terms won't help with categorization
    - or not useful globally
- Pros: Easy to implement, scalable
- Cons: Ad-hoc, low DF terms 'topical'

Term Selection Functions: MI

- Pointwise Mutual Information (MI)

$$MI(t_k, c_i) = \log \frac{P(t_k, c_i)}{P(t_k)P(c_i)}$$

- MI(t,c)=0 if t and c are independent

- Issue: Can be heavily influenced by marginal probability
  - Problem comparing terms of differing frequencies

Term Selection Functions: IG

- Information Gain:
  - Intuition: Transmitting Y, how many bits can we save if both sides know X?
  - IG(Y,X) = H(Y)-H(Y|X)

$$IG(t_k, c_i) = P(t_k, c_i)\log\frac{P(t_k, c_i)}{P(t_k)P(c_i)} + P(\overline{t}_k, c_i)\log\frac{P(\overline{t}_k, c_i)}{P(\overline{t}_k)P(c_i)}$$

Global Selection

- Previous measures compute class-specific selection
- What if you want to filter across ALL classes?
  – an aggregate measure across classes
    - Sum:

$$f_{sum}(t_k) = \sum_{i=1}^{|C|} f(t_k, c_i)$$

    - Average:

$$f_{avg}(t_k) = \sum_{i=1}^{|C|} f(t_k, c_i) P(c_i)$$

    - Max:

$$f_{\max}(t_k) = \max_{i=1}^{|C|} f(t_k, c_i) P(c_i)$$

|C| is the number of classes

Which function works the best?

- It depends on
  - Classifiers
  - Type of data
  - …

- According to (Yang and Pedersen 1997):

$$\{OR, NGL, GSS\} > \{\chi^2_{max}, IG_{sum}\}$$
$$> \{\#_{avg}\} >> \{MI\}$$

Feature weighting

Feature weights

- Feature weight $\in$ {0,1}: same as DR

- Feature weight $\in$ R: iterative approach:
  - Ex: MaxEnt

➔ Feature selection is a special case of feature weighting.

Feature values

- Binary features: 0 or 1.
- Term frequency (TF): the number of times that t_k appears in d_i.
- Inversed document frequency (IDF): log |D| /$d_{k,}$ where d_k is the number of documents that contain t_k.
- TFIDF = TF * IDF
- Normalized TFIDF: $$w_{ik} = \frac{tfidf(d_i, t_k)}{Z}$$

Summary so far

- Curse of dimensionality ➔ dimensionality reduction (DR)

- DR:
  - Feature extraction
  - Feature selection
    - Wrapping method
    - <u>Filtering method</u>: different functions

Summary (cont)

- Functions:
  - Document frequency
  - Information gain
  - Gain ratio
  - Chi square
  - …

Additional slides

Information gain**

$$\boxed{\sum_i IG(t_k, c_i)}$$

$$= \sum_{c \in C} \sum_{t \in \{t_k, \bar{t_k}\}} P(t,c) log \frac{P(t,c)}{P(c)P(t)}$$

$$= \sum_{c \in C} \sum_t P(t,c) log P(c|t)$$

$$- \sum_c \sum_t P(t,c) log P(c)$$

$$= -H(C|T) - \sum_c ((log P(c)) \sum_t P(t,c))$$

$$= -H(C|T) + H(C) = \boxed{IG(C,T)}$$

More term selection functions**

Relevancy score:

$$RS(t_k, c_i) = log\frac{P(t_k|c_i)+d}{P(\bar{t_k}|\bar{c_i})+d}$$

Odds Ratio:

$$OR(t_k, c_i) = \frac{P(t_k|c_i)P(\bar{t_k}|\bar{c_i})}{P(\bar{t_k}|c_i)P(t_k|\bar{c_i})}$$

More term selection functions**

GSS coefficient:

$$GSS(t_k, c_i) = P(t_k, c_i)P(\bar{t_k}, \bar{c_i}) - P(t_k, \bar{c_i})P(\bar{t_k}, c_i)$$

NGL coefficient: N is the total number of docs

$$NGL(t_k, c_i) = \frac{\sqrt{N}\ GSS(t_k, c_i)}{\sqrt{P(t_k)P(\bar{t_k})P(c_i)P(\bar{c_i})}}$$

Chi-square: (one of the definitions)

$$\chi^2(t_k, c_i) = NGL(t_k, c_i)^2 = \frac{(ad-bc)^2 N}{(a+b)(a+c)(b+d)(c+d)}$$