Text Classification

Dr. Ahmed Rafea
Supervised learning

- Learning to assign objects to classes given examples
- Learner (classifier)

A typical supervised text learning scenario.
Difference with texts

- M.L classification techniques used for structured data
- Text: lots of features and lot of noise
- No fixed number of columns
- No categorical attribute values
- Data scarcity
- Larger number of class label
- Hierarchical relationships between classes less systematic unlike structured data
Techniques

- **Nearest Neighbor Classifier**
  - Lazy learner: remember all training instances
  - *Decision on test document:* distribution of labels on the training documents most similar to it
  - Assigns large weights to rare terms

- **Feature selection**
  - Removes terms in the training documents which are statistically uncorrelated with the class labels,

- **Bayesian classifier**
  - Fit a generative term distribution $\Pr(d|c)$ to each class $c$ of documents $\{d\}$.
  - *Testing:* The distribution most likely to have generated a test document is used to label it.
Other Classifiers

- **Maximum entropy classifier:**
  - Estimate a direct distribution $Pr(c|d)$ from term space to the probability of various classes.

- **Support vector machines:**
  - Represent classes by numbers
  - Construct a direct function from term space to the class variable.

- **Rule induction:**
  - Induce rules for classification over diverse features
  - E.g.: information from ordinary terms, the structure of the HTML tag tree in which terms are embedded, link neighbors, citations
Other Issues

- **Tokenization**
  - *E.g.* replacing monetary amounts by a special token

- **Evaluating text classifier**
  - **Accuracy**
  - Training speed and scalability
  - Simplicity, speed, and scalability for document modifications
  - Ease of diagnosis, interpretation of results, and adding human judgment and feedback
Benchmarks for accuracy

- **Reuters**
  - 10,700 labeled documents
  - 10% documents with multiple class labels

- **OHSUMED**
  - 348,566 abstracts from medical journals

- **20NG**
  - 18,800 labeled USENET postings
  - 20 leaf classes, 5 root level classes

- **WebKB**
  - 8,300 documents in 7 academic categories.

- **Industry**
  - 10,000 home pages of companies from 105 industry sectors
  - Shallow hierarchies of sector names
Measures of accuracy

- Assumptions
  - Each document is associated with exactly one class.
  - OR
  - Each document is associated with a subset of classes.

- Confusion matrix (M)
  - For more than 2 classes
  - $M[i; j]$ : number of test documents belonging to class $i$ which were assigned to class $j$
  - *Perfect classifier:* diagonal elements $M[i; i]$ would be nonzero.
Evaluating classifier accuracy

- **Two-way ensemble**
  - To avoid searching over the power-set of class labels in the subset scenario
  - Create positive \( (C_d) \) and negative classes \( (\overline{C_d}) \) for each document \( d \) (E.g.: “Sports” and “Not sports” (all remaining documents))

- **Recall and precision**
  - \( 2 \times 2 \) contingency matrix per \( (d,c) \) pair
    - \( M_{d,c}[0,0] = [ c \in C_d \text{ and classier outputs } c ] \)
    - \( M_{d,c}[0,1] = [ c \in C_d \text{ and classier does not output } c ] \)
    - \( M_{d,c}[1,0] = [ c \notin C_d \text{ and classier outputs } c ] \)
    - \( M_{d,c}[1,1] = [ c \notin C_d \text{ and classier does not output } c ] \)
Evaluating classifier accuracy (contd.)

- micro averaged contingency matrix
  \[ M_\mu = \sum_{d,c} M_{d,c} \]
- micro averaged contingency matrix
  \[ M_c = \frac{1}{|C|} \sum_c \sum_d M_{c,d} \]
- micro averaged precision and recall
  - Equal importance for each document
    \[ M_\mu (\text{precision}) = \frac{M_\mu \left[ 0,0 \right]}{M_\mu \left[ 0,0 \right] + M_\mu \left[ 1,0 \right]} \quad M_\mu (\text{recall}) = \frac{M_\mu \left[ 0,0 \right]}{M_\mu \left[ 0,0 \right] + M_\mu \left[ 0,1 \right]} \]

- Macro averaged precision and recall
  - Equal importance for each class
    \[ M_c (\text{precision}) = \frac{M_c \left[ 0,0 \right]}{M_c \left[ 0,0 \right] + M_c \left[ 1,0 \right]} \quad M_c (\text{recall}) = \frac{M_c \left[ 0,0 \right]}{M_c \left[ 0,0 \right] + M_c \left[ 0,1 \right]} \]
Evaluating classifier accuracy (contd.)

- Precision – Recall tradeoff
  - Plot of precision vs. recall: Better classifier has higher curvature
  - Harmonic mean: Discard classifiers that sacrifice one for the other

\[
F_1 = \frac{2 \times \text{recall} \times \text{precision}}{\text{recall} + \text{precision}}
\]
Nearest Neighbor classifiers (1/7)

- Intuition
  - similar documents are expected to be assigned the same class label.
  - Vector space model + cosine similarity
  - Training:
    - Index each document and remember class label
Nearest Neighbor classifiers (2/7)

• Testing:
  ♦ Fetch “k” most similar document to given document
    – Majority class wins
    – **Alternative:** Weighted counts – counts of classes weighted by the corresponding similarity measure
      \[
      s(d_q,c) = \sum_{d_c \in kNN(d_q)} s(d_q,d_c)
      \]
    – **Alternative:** per-class offset \( b_c \) which is tuned by testing the classifier on a portion of training data held out for this purpose.
      \[
      s(d_q,c) = b_c + \sum_{d_c \in kNN(d_q)} s(d_q,d_c)
      \]
Nearest Neighbor classifiers (3/7)

Nearest neighbor classification

Test document

Training documents

Nearest neighbor classification
Nearest Neighbor classifiers (4/7)

- Pros
  - Easy availability and reuse of inverted index
  - Collection updates trivial
  - Accuracy comparable to best known classifiers
Nearest Neighbor classifiers (5/7)

- Cons
  - Iceberg category questions
    - involves as many inverted index lookups as there are distinct terms in $d_q$,
    - scoring the (possibly large number of) candidate documents which overlap with $d_q$ in at least one word,
    - sorting by overall similarity,
    - picking the best $k$ documents,
  - Space overhead and redundancy
    - Data stored at level of individual documents
    - No distillation
Nearest Neighbor classifiers (6/7)

- Workarounds
  - To reducing space requirements and speed up classification
    - Find clusters in the data
    - Store only a few statistical parameters per cluster.
    - Compare with documents in only the most promising clusters.
  - Again….
    - Ad-hoc choices for number and size of clusters and parameters.
    - $k$ is corpus sensitive
Nearest Neighbor classifiers (7/7)

- TF-IDF
  - TF-IDF done for whole corpus
  - Interclass correlations and term frequencies unaccounted for
  - Terms which occur relatively frequent in some classes compared to others should have higher importance
  - Overall rarity in the corpus is not as important.
Feature selection (1/11)

- **Data sparsity:**
  - Term distribution could be estimated if training set larger than number of features, however this is not the case.
  - Vocabulary $W \Rightarrow 2^{|W|}$ documents.
  - For Reuters, that number would be $2^{30,000} \approx 10^{10,000}$ but only about 10300 documents are available.

- **Over-fitting problem**
  - Joint distribution may fit training instances.
  - But may not fit unforeseen test data that well.
Feature selection (2/11)

- Marginal rather than joint
  - Marginal distribution of each term in each class
  - Empirical distributions may not still reflect actual distributions if data is sparse
  - Therefore feature selection is needed

  - Purposes:
    - Improve accuracy by avoiding over fitting
    - maintain accuracy while discarding as many features as possible to save a great deal of space for storing statistics

  - Heuristic, guided by linguistic and domain knowledge, or statistical.
Feature selection (3/11)

- **Perfect feature selection**
  - goal-directed
  - pick all possible subsets of features
  - for each subset train and test a classifier
  - retain that subset which resulted in the highest accuracy.
  - COMPUTATIONALLY INFEASIBLE

- **Simple heuristics**
  - Stop words like “a”, “an”, “the” etc.
  - Empirically chosen thresholds (task and corpus sensitive) for ignoring “too frequent” or “too rare” terms
  - Discard “too frequent” and “too rare terms”

- **Larger and complex data sets**
  - Confusion with stop words
  - Especially for topic hierarchies

- **Two basic strategies**
  - Starts with the empty set and includes good features (Greedy inclusion algorithm)
  - Starts from complete feature set and exclude irrelevant features (Truncation algorithm)
Feature selection(4/11)

- Greedy inclusion algorithm
  (most commonly used in the text domain)
  1. Compute, for each term, a measure of discrimination amongst classes.
  2. Arrange the terms in decreasing order of this measure.
  3. Retain a number of the best terms or features for use by the classifier.

- Greedy because
  - measure of discrimination of a term is computed independently of other terms
  - Over-inclusion: mild effects on accuracy
Feature selection (5/11)

• Measure of discrimination depends on:
  • model of documents
  • desired speed of training
  • ease of updates to documents and class assignments.

• Observations
  • Although different measures will result in somewhat different term ranks, the sets included for acceptable accuracy tend to have large overlap.
  • Therefore, most classifiers will be insensitive to specific choice of discrimination measures.
Feature selection (6/11)

- The $\chi^2$ test
  - Build a 2 x 2 contingency matrix per class-term pair
    
    $k_{i,0} = \text{number of documents in class } i \text{ not containing term } t$
    $k_{i,1} = \text{number of documents in class } i \text{ containing term } t$
  
  - Under the independence hypothesis
    - Aggregates the deviations of observed values from expected values
    - Larger the value of $\chi^2$, the lower is our belief that the independence assumption is upheld by the observed data.
Feature selection (7/11)

• The $\chi^2$ test

$$\chi^2 = \sum_{l,m} \frac{k_{l,m} - n \Pr(C = l) \Pr(I_t = m)}{n \Pr(C = l) \Pr(I_t = m)} = \frac{n(k_{11}k_{00} - k_{10}k_{01})^2}{(k_{11} + k_{10})(k_{01} + k_{00})(k_{11} + k_{01})(k_{10} + k_{00})}$$

• Feature selection process
  • Sort terms in decreasing order of their $\chi^2$ values,
  • Train several classifier with varying number of features
  • Stopping at the point of maximum accuracy.
Feature selection (8/11)

• Truncation algorithms
  • Start from the complete set of terms $T$
    1. Keep selecting terms to drop
    2. Till you end up with a feature subset
    3. Question: When should you stop truncation?
  • Two objectives
    ◆ minimize the size of selected feature set $F$.
    ◆ Keep the distorted distribution $Pr(C|F)$ as similar as possible to the original $Pr(C|T)$
Feature selection (9/11)

• Truncation Algorithms: Example
  • Kullback-Leibler (KL)
    ✷ Measures similarity or distance between two distributions
  • Markov Blanket
    ✷ Let $X$ be a feature in $T$. Let $M \subseteq T \setminus \{X\}$
    ✷ The presence of $M$ renders the presence of $X$ unnecessary as a feature => $M$ is a Markov blanket for $X$
    ✷ Technically
      • $M$ is called a Markov blanket for $X \in T$ if $X$ is conditionally independent of $(T \cup C) \setminus (M \cup \{X\})$ given $M$
      • eliminating a variable because it has a Markov blanket contained in other existing features does not increase the KL distance between $\Pr(C|T)$ and $\Pr(C|F)$. 
Feature selection (10/11)

• Finding Markov Blankets
  • Absence of Markov Blanket in practice
  • Finding approximate Markov blankets
    ♦ Purpose: To cut down computational complexity
    ♦ search for Markov blankets $M$ to those with at most $k$ features.
    ♦ given feature $X$, search for the members of $M$ to those features which are most strongly correlated (using tests similar to the $X^2$ or MI tests) with $X$.

• Example: For Reuters dataset, over two-thirds of $T$ could be discarded while increasing classification accuracy
Feature selection (11/11)

• General observations on feature selection
  • The issue of document length should be addressed properly.
  • Choice of association measures does not make a dramatic difference
  • Greedy inclusion algorithms scale nearly linearly with the number of features
  • Markov blanket technique takes time proportional to at least

• Advantage of Markov blankets over greedy inclusion
  ♦ Greedy may include features with high individual correlations even though one subsumes the other
  ♦ Features individually uncorrelated could be jointly more correlated with the class
    • This rarely happens