CS276B
Text Information Retrieval, Mining, and Exploitation

Lecture 5
23 January 2003
Recap
Today’s topics

- Feature selection for text classification
- Measuring classification performance
- Nearest neighbor categorization
Feature Selection: Why?

- Text collections have a large number of features
  - 10,000 – 1,000,000 unique words – and more
- Make using a particular classifier feasible
  - Some classifiers can’t deal with 100,000s of feat’s
- Reduce training time
  - Training time for some methods is quadratic or worse in the number of features (e.g., logistic regression)
- Improve generalization
  - Eliminate noise features
Recap: Feature Reduction

- Standard ways of reducing feature space for text
  - Stemming
    - Laugh, laughs, laughing, laughed -> laugh
  - Stop word removal
    - E.g., eliminate all prepositions
  - Conversion to lower case
  - Tokenization
    - Break on all special characters: fire-fighter -> fire, fighter
Feature Selection

- Yang and Pedersen 1997
- Comparison of different selection criteria
  - DF – document frequency
  - IG – information gain
  - MI – mutual information
  - CHI – chi square
- Common strategy
  - Compute statistic for each term
  - Keep n terms with highest value of this statistic
Information Gain

\[ G(t) = - \sum_{i=1}^{m} P_r(c_i) \log P_r(c_i) \]
\[ + P_r(t) \sum_{i=1}^{m} P_r(c_i \mid t) \log P_r(c_i \mid t) \]
\[ + P_r(\overline{t}) \sum_{i=1}^{m} P_r(c_i \mid \overline{t}) \log P_r(c_i \mid \overline{t}) \]
(Pointwise) Mutual Information

\[ I(t, c) = \log \frac{P_r(t \land c)}{P_r(t) \times P_r(c)} \]

\[ I_{\text{avg}}(t) = \sum_{i=1}^{m} P_r(c_i) I(t, c_i) \]

\[ I_{\text{max}}(t) = \max_{i=1}^{m} \{ I(t, c_i) \} \]
# Chi-Square

<table>
<thead>
<tr>
<th>Term present</th>
<th>Term absent</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>C</td>
<td>D</td>
</tr>
</tbody>
</table>

\[ X^2 = N(AD-BC)^2 / ( (A+B)(A+C)(B+D)(C+D) ) \]

Use either maximum or average \( X^2 \)

Value for complete independence?
Document Frequency

- Number of documents a term occurs in
- Is sometimes used for eliminating both very frequent and very infrequent terms
- How is document frequency measure different from the other 3 measures?
Yang & Pedersen: Experiments

- Two classification methods
  - kNN (k nearest neighbors; more later)
  - Linear Least Squares Fit
    - Regression method
- Collections
  - Reuters-22173
    - 92 categories
    - 16,000 unique terms
  - Ohsumed: subset of medline
    - 14,000 categories
    - 72,000 unique terms
- Ltc term weighting
Yang&Pedersen: Experiments

- Choose feature set size
- Preprocess collection, discarding non-selected features / words
- Apply term weighting -> feature vector for each document
- Train classifier on training set
- Evaluate classifier on test set
Figure 1: Average precision of kNN vs. unique word count
Discussion

- You can eliminate 90% of features for IG, DF, and CHI without decreasing performance.
- In fact, performance increases with fewer features for IG, DF, and CHI.
- Mutual information is very sensitive to small counts.
- IG does best with smallest number of features.
- Document frequency is close to optimal. By far the simplest feature selection method.
- Similar results for LLSF (regression).
Results

Table 1. Criteria and performance of feature selection methods in kNN & LLSF

<table>
<thead>
<tr>
<th>Method</th>
<th>DF</th>
<th>IG</th>
<th>CHI</th>
<th>MI</th>
<th>TS</th>
</tr>
</thead>
<tbody>
<tr>
<td>favoring common terms</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y/N</td>
</tr>
<tr>
<td>using categories</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>using term absence</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>performance in kNN/LLSF</td>
<td>excellent</td>
<td>excellent</td>
<td>excellent</td>
<td>poor</td>
<td>ok</td>
</tr>
</tbody>
</table>

Why is selecting common terms a good strategy?
IG, DF, CHI Are Correlated.
Information Gain vs Mutual Information

- Information gain is similar to MI for random variables
- Independence?
- In contrast, pointwise MI ignores non-occurrence of terms
  - E.g., for complete dependence, you get:
  - \( \frac{P(AB)}{P(A)P(B)} = \frac{1}{P(A)} \) – larger for rare terms than for frequent terms
- Yang&Pedersen: Pointwise MI favors rare terms

\[
G(t) = \sum_{X \in \{t,t\}} \sum_{Y \in \{c_i\}} P_r(X,Y) \log \frac{P_r(X,Y)}{P_r(X)P_r(Y)}
\]
Feature Selection: Other Considerations

- Generic vs Class-Specific
  - Completely generic (class-independent)
  - Separate feature set for each class
  - Mixed (a la Yang & Pedersen)

- Maintainability over time
  - Is aggressive features selection good or bad for robustness over time?

- Ideal: Optimal features selected as part of training
Yang & Pedersen: Limitations

- Don’t look at class specific feature selection
- Don’t look at methods that can’t handle high-dimensional spaces
- Evaluate category ranking (as opposed to classification accuracy)
Feature Selection: Other Methods

- Stepwise term selection
  - Forward
  - Backward
  - Expensive: need to do $n^2$ iterations of training
- Term clustering
- Dimension reduction: PCA / SVD
Word Rep. vs. Dimension Reduction

- Word representations: one dimension for each word (binary, count, or weight)
- Dimension reduction: each dimension is a unique linear combination of all words (linear case)
- Dimension reduction is good for generic topics ("politics"), bad for specific classes ("ruanda"). Why?
- SVD/PCA computationally expensive
- Higher complexity in implementation
- No clear examples of higher performance through dimension reduction
## Word Rep. vs. Dimension Reduction

<table>
<thead>
<tr>
<th>classifier</th>
<th>input</th>
<th>precision for Topics 51–100</th>
<th>precision for Topics 101–150</th>
<th>average change</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>average</td>
<td>% change</td>
<td>at 100</td>
</tr>
<tr>
<td>baseline</td>
<td>expansion</td>
<td>0.3678</td>
<td>+0.0%</td>
<td>0.4710</td>
</tr>
<tr>
<td></td>
<td>LSI</td>
<td>0.3240</td>
<td>-11.9%</td>
<td>0.4210</td>
</tr>
<tr>
<td></td>
<td>200 terms</td>
<td>0.3789</td>
<td>+3.0%</td>
<td>0.4824</td>
</tr>
<tr>
<td></td>
<td>LSI + 200 terms</td>
<td>0.3359</td>
<td>-8.7%</td>
<td>0.4426</td>
</tr>
<tr>
<td>logistic regression</td>
<td>LSI</td>
<td>0.3980</td>
<td>+8.2%</td>
<td>0.5108</td>
</tr>
<tr>
<td></td>
<td>200 terms</td>
<td>0.3654</td>
<td>-0.7%</td>
<td>0.4788</td>
</tr>
<tr>
<td></td>
<td>LSI + 200 terms</td>
<td>0.3494</td>
<td>-5.0%</td>
<td>0.4652</td>
</tr>
<tr>
<td>LDA</td>
<td>LSI</td>
<td>0.4139</td>
<td>+12.5%</td>
<td>0.5166</td>
</tr>
<tr>
<td></td>
<td>200 terms</td>
<td>0.3966</td>
<td>+7.8%</td>
<td>0.4916</td>
</tr>
<tr>
<td></td>
<td>LSI + 200 terms</td>
<td>0.3973</td>
<td>+8.0%</td>
<td>0.5034</td>
</tr>
<tr>
<td>linear network</td>
<td>LSI</td>
<td>0.4098</td>
<td>+11.4%</td>
<td>0.5094</td>
</tr>
<tr>
<td></td>
<td>200 terms</td>
<td>0.4209</td>
<td>+14.4%</td>
<td>0.5044</td>
</tr>
<tr>
<td></td>
<td>LSI + 200 terms</td>
<td>0.4273</td>
<td>+16.2%</td>
<td>0.5180</td>
</tr>
<tr>
<td>non-linear network</td>
<td>LSI</td>
<td>0.4110</td>
<td>+11.7%</td>
<td>0.5090</td>
</tr>
<tr>
<td></td>
<td>200 terms</td>
<td>0.4210</td>
<td>+14.5%</td>
<td>0.5026</td>
</tr>
<tr>
<td></td>
<td>LSI + 200 terms</td>
<td>0.4251</td>
<td>+15.6%</td>
<td>0.5204</td>
</tr>
</tbody>
</table>
Measuring Classification Figures of Merit

- Accuracy of classification
  - Main evaluation criterion in academia
  - More in a moment
- Speed of training statistical classifier
- Speed of classification (docs/hour)
  - No big differences for most algorithms
  - Exceptions: kNN, complex preprocessing requirements
- Effort in creating training set (human hours/topic)
  - More on this in Lecture 9 (Active Learning)
Measures of Accuracy

- Error rate
  - Not a good measure for small classes. Why?
- Precision/recall for classification decisions
- $F_1$ measure: $1/F_1 = \frac{1}{2} (1/P + 1/R)$
- Breakeven point
- Correct estimate of size of category
  - Why is this different?
- Precision/recall for ranking classes
- Stability over time / concept drift
- Utility
Precision/Recall for Ranking Classes

- Example: “Bad wheat harvest in Turkey”
- True categories
  - Wheat
  - Turkey
- Ranked category list
  - 0.9: turkey
  - 0.7: poultry
  - 0.5: armenia
  - 0.4: barley
  - 0.3: georgia
- Precision at 5: 0.1, Recall at 5: 0.5
Precision/Recall for Ranking Classes

- Consider problems with many categories (>10)
- Use method returning scores comparable across categories (not: Naïve Bayes)
- Rank categories and compute average precision recall (or other measure characterizing precision/recall curve)
- Good measure for interactive support of human categorization
- Useless for an “autonomous” system (e.g. a filter on a stream of newswire stories)
Concept Drift

- Categories change over time
- Example: “president of the united states”
  - 1999: clinton is great feature
  - 2002: clinton is bad feature
- One measure of a text classification system is how well it protects against concept drift.
- Feature selection: good or bad to protect against concept drift?
Micro- vs. Macro-Averaging

- If we have more than one class, how do we combine multiple performance measures into one quantity?
- Macroaveraging: Compute performance for each class, then average.
- Microaveraging: Collect decisions for all classes, compute contingency table, evaluate.
## Micro- vs. Macro-Averaging: Example

<table>
<thead>
<tr>
<th>Class 1</th>
<th>Class 2</th>
<th>Micro.Av. Table</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Truth:</strong></td>
<td><strong>Truth:</strong></td>
<td><strong>Truth:</strong></td>
</tr>
<tr>
<td><strong>yes</strong></td>
<td><strong>yes</strong></td>
<td><strong>yes</strong></td>
</tr>
<tr>
<td><strong>no</strong></td>
<td><strong>no</strong></td>
<td><strong>no</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Classifier:</th>
<th>10</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>yes</strong></td>
<td>10</td>
<td>970</td>
</tr>
<tr>
<td><strong>no</strong></td>
<td>10</td>
<td>890</td>
</tr>
</tbody>
</table>

- **Macroaveraged precision:** \((0.5 + 0.9)/2 = 0.7\)

- **Microaveraged precision:** \(100/120 = 0.83\)

- **Why this difference?**
Reuters 1

- Newswire text
- Statistics (vary according to version used)
  - Training set: 9,610
  - Test set: 3,662
  - 50% of documents have no category assigned
  - Average document length: 90.6
  - Number of classes: 92
  - Example classes: currency exchange, wheat, gold
  - Max classes assigned: 14
  - Average number of classes assigned
    - 1.24 for docs with at least one category
Reuters 1

- Only about 10 out of 92 categories are large
- Microaveraging measures performance on large categories.
Factors Affecting Measures

- Variability of data
  - Document size/length
  - quality/style of authorship
  - uniformity of vocabulary
- Variability of “truth” / gold standard
  - need definitive judgement on which topic(s) a doc belongs to
    - usually human
  - Ideally: consistent judgements
Accuracy measurement

- Confusion matrix

This \((i, j)\) entry means 53 of the docs actually in topic \(i\) were put in topic \(j\) by the classifier.
Confusion matrix

- Function of classifier, topics and test docs.
- For a perfect classifier, all off-diagonal entries should be zero.
- For a perfect classifier, if there are $n$ docs in category $j$ than entry $(j,j)$ should be $n$.
- Straightforward when there is 1 category per document.
- Can be extended to $n$ categories per document.
Confusion measures (1 class / doc)

- Recall: Fraction of docs in topic $i$ classified correctly:
  \[
  \frac{c_{ii}}{\sum_{j} c_{ij}}
  \]

- Precision: Fraction of docs assigned topic $i$ that are actually about topic $i$:
  \[
  \frac{c_{ii}}{\sum_{j} c_{ji}}
  \]

- “Correct rate”: (1 - error rate) Fraction of docs classified correctly:
  \[
  \frac{\sum_{i} c_{ii}}{\sum_{j} \sum_{i} c_{ij}}
  \]
Principled approach to training

- Optimize the measure that performance is measured with
- $s$: vector of classifier decision, $z$: vector of true classes
- $h(s, z) = \text{cost of making decisions } s \text{ for true classes}$
Utility / Cost

- One cost function \( h \) is based on contingency table.
- Assume identical cost for all false positives etc.
- Cost \( C = 111 * A + 112 * B + 121 * C + 122 * D \)
- For this cost \( c \), we get the following optimality conditions:

\[
p_i > \frac{(\lambda_{12} - \lambda_{22})}{(\lambda_{21} - \lambda_{11}) + (\lambda_{12} - \lambda_{22})}.
\]

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Truth: yes</th>
<th>Truth: no</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classifer: yes</td>
<td>Cost: ( \lambda_{11} )</td>
<td>Cost: ( \lambda_{12} )</td>
</tr>
<tr>
<td>Count: A</td>
<td>Count: B</td>
<td></td>
</tr>
<tr>
<td>Classifier: no</td>
<td>Cost: ( \lambda_{21} )</td>
<td>Cost: ( \lambda_{22} )</td>
</tr>
<tr>
<td>Count: C</td>
<td>Count: D</td>
<td></td>
</tr>
</tbody>
</table>
Utility / Cost

\[
p_i > \frac{(\lambda_{12} - \lambda_{22})}{(\lambda_{21} - \lambda_{11}) + (\lambda_{12} - \lambda_{22})}.
\]

Most common cost: 1 for error, 0 for correct. \( P_i > ? \)

Product cross-sale: high cost for false positive, low cost for false negative.

Patent search: low cost for false positive, high cost for false negative.

<table>
<thead>
<tr>
<th>Classifer: yes</th>
<th>Classifer: no</th>
</tr>
</thead>
<tbody>
<tr>
<td>Truth: yes</td>
<td>( \lambda_{11} )</td>
</tr>
<tr>
<td>Truth: no</td>
<td>( \lambda_{21} )</td>
</tr>
</tbody>
</table>
Are All Optimal Rules of Form $p > \theta$?

- In the above examples, all you need to do is estimate probability of class membership.
- Can all problems be solved like this?
- No!
- Probability is often not sufficient
- User decision depends on the distribution of relevance
- Example: information filter for terrorism
Naïve Bayes
Vector Space Classification
Nearest Neighbor Classification
Recall Vector Space Representation

- Each doc $j$ is a vector, one component for each term (= word).
- Normalize to unit length.
- Have a vector space
  - terms are axes
  - $n$ docs live in this space
  - even with stemming, may have 10000+ dimensions, or even 1,000,000+
Classification Using Vector Spaces

- Each training doc a point (vector) labeled by its topic (= class)
- Hypothesis: docs of the same topic form a contiguous region of space
- Define surfaces to delineate topics in space
Topics in a vector space

- Government
- Science
- Arts
Given a test doc

- Figure out which region it lies in
- Assign corresponding class
Binary Classification

- Consider 2 class problems
- How do we define (and find) the separating surface?
- How do we test which region a test doc is in?
Separation by Hyperplanes

- Assume *linear separability* for now:
  - in 2 dimensions, can separate by a line
  - in higher dimensions, need hyperplanes
- Can find separating hyperplane by *linear programming* (e.g. perceptron):
  - separator can be expressed as $ax + by = c$
Linear programming / Perceptron

Find $a, b, c$, such that $ax + by \geq c$ for red points
$ax + by \leq c$ for green points.
Relationship to Naïve Bayes?

Find $a, b, c$, such that

$ax + by \geq c$ for red points

$ax + by \leq c$ for green points.
Linear Classifiers

- Many common text classifiers are linear classifiers
- Despite this similarity, large performance differences
  - For separable problems, there is an infinite number of separating hyperplanes. Which one do you choose?
  - What to do for non-separable problems?
Which hyperplane?

In general, lots of possible solutions for $a, b, c$. 
Support Vector Machine (SVM)

- Quadratic programming problem
- The decision function is fully specified by subset of training samples, the support vectors.
- Text classification method du jour
- Topic of lecture 9
Category: Interest

- Example SVM features
  \[ w_i \quad t_i \quad w_i \quad t_i \]

- 0.70 prime
- 0.67 rate
- 0.63 interest
- 0.60 rates
- 0.46 discount
- 0.43 bundesbank
- 0.43 baker
- -0.71 dlr
- -0.35 world
- -0.33 sees
- -0.25 year
- -0.24 group
- -0.24 dlr
- -0.24 january
More Than Two Classes

- Any-of or multiclass classification
  - For n classes, decompose into n binary problems
- One-of classification: each document belongs to exactly one class
  - How do we compose separating surfaces into regions?
- Centroid classification
- K nearest neighbor classification
Composing Surfaces: Issues
Separating Multiple Topics

- Build a separator between each topic and its complementary set (docs from all other topics).
- Given test doc, evaluate it for membership in each topic.
- Declare membership in topics
  - One-of classification:
    - for class with maximum score/confidence/probability
  - Multiclass classification:
    - For classes above threshold
Negative examples

- Formulate as above, except negative examples for a topic are added to its complementary set.

○ Positive examples
□ Negative examples
Centroid Classification

- Given training docs for a topic, compute their centroid
- Now have a centroid for each topic
- Given query doc, assign to topic whose centroid is nearest.
- Exercise: Compare to Rocchio
k Nearest Neighbor Classification

- To classify document d into class c
- Define k-neighborhood N as k nearest neighbors of d
- Count number of documents l in N that belong to c
- Estimate $P(c|d)$ as $l/k$
Cover and Hart 1967

- Asymptotically, the error rate of 1-nearest-neighbor classification is less than twice the Bayes rate.
- Assume that query point coincides with a training point.
- Both query point and training point contribute error $\rightarrow$ 2 times Bayes rate
kNN vs. Regression

- kNN has high variance and low bias.
- Linear regression has low variance and high bias.
**kNN: Discussion**

- Classification time linear in training set
- Training set generation
  - incompletely judged set can be problematic for multiclass problems
- No feature selection necessary
- Scales well with large number of categories
  - Don’t need to train n classifiers for n classes
- Categories can influence each other
  - Small changes to one category can have ripple effect
- Scores can be hard to convert to probabilities
- No training necessary
  - Actually: not true. Why?
Number of neighbors
References

- Trevor Hastie, Robert Tibshirani and Jerome Friedman, "Elements of Statistical Learning: Data Mining, Inference and Prediction" Springer-Verlag
Kappa Measure

- Kappa measures
  - Agreement among coders
  - Designed for categorical judgments
  - Corrects for chance agreement
- Kappa = \[ \frac{P(A) - P(E)}{1 - P(E)} \]
- P(A) – proportion of time coders agree
- P(E) – what agreement would be by chance
- Kappa = 0 for chance agreement, 1 for total agreement.