ML in IR: We’ve talked about:

- Classification
  - Topic categorization
  - Sentiment analysis
- Clustering
  - Topic detection
- And a few others
Types of ML

- Supervised learning
  - Classification
  - Regression
- Unsupervised learning
  - Clustering
  - Anomaly detection
- Pattern Discovery
  - Association rules, ...

Regression

- Goal: Learn to predict a numeric score
- Approaches
  - Linear Regression
    - $Ax + By + Cz = \text{output}$
    - Choose A+B+C to minimize error
  - CART (Classification and Regression Tree)
    - Piecewise linear regression
  - Neural networks
    - With some caveats
Regression for Retrieval Models

• Challenge: Training Data
  – What would training data look like?

<table>
<thead>
<tr>
<th>Query</th>
<th>Word</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regression</td>
<td>Retrieval</td>
<td>4</td>
</tr>
<tr>
<td>Information</td>
<td>Model</td>
<td>3</td>
</tr>
<tr>
<td>Retrieval</td>
<td>Training</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>Data</td>
<td>2</td>
</tr>
</tbody>
</table>

Score: 0.73

• Is this feasible?

Where is Regression Appropriate?

• Advertising?
  – Predict revenue from an ad?

• Search engine revenue models
  – Pay per view (rather simple)
  – Pay per click (need probability of click)

• Advertiser bidding
  – Revenue expectation per ad: Conversion rate
Pattern Discovery

- Identify recurring patterns in the data
- Examples:
  - Correlation (Information and Retrieval occur together frequently)
  - Association rules (A & B $\rightarrow$ C)
- Where is this useful in IR?
  - Summarization
  - Topic identification / naming
  - Feature selection

Feature Selection Example:

Word2vec

- Word Embedding: Map a word to a vector of numbers
  - Words with similar means should have similar vectors
- Similar goal to Latent Semantic Indexing
  - But word level rather than document level
- Word2vec: Neural Network approach
Word2vec: Basic Idea
(Mikolov et al. 2013)

- Neural networks for word embedding not new
  - Mikolov et al. point to Morin & Bengio ’86
- Key contribution: Scale
  - Google has a lot more data
  - Authors figured out how to train network much more efficiently
- Increasing the amount of data dramatically improved results
TopCat: Data Mining for Topic Identification

Chris Clifton
Robert Cooley

16 September, 1999

Goal: Automatically Identify Recurring Topics in a News Corpus

- Started with a user problem: Geographic analysis of news
- Idea: Segment news into ongoing topics/stories
  How do we do this?
- What we need:
  - Topics
  - “Mnemonic” for describing/remembering the topic
  - Mapping from news articles to topics
- Other goals:
  - Gain insight into collection that couldn’t be had from skimming a few documents
  - Identify key players in a story/topic
User Problem: Geographic News Analysis

TopCat identified separate topics for U.S. embassy bombing and counter-strike.

Bombing

Counter-strike

A Data Mining Based Solution

Idea in Brief

- A topic often contains a number of recurring players/concepts
  - Identified highly correlated named entities (frequent itemsets)
  - Can easily tie these back to the source documents
  - But there were too many to be useful
- Frequent itemsets often overlap
  - Used this to cluster the correlated entities
  - But the link back to the source documents is no longer clear
- Evaluated against manually-categorized “ground truth” set
  - Data for Topic Detection and Tracking (TDT2) program
  - Used “topic” (list of entities) as a query to find relevant documents to compare with known mappings
Preprocessing

- Identify named entities (person, location, organization) in text
  - Alembic Natural Language Processing system
- Data Cleansing:
  - Coreference Resolution
    - Used intra-document coreference from NLP system
    - Heuristic to choose "global best name" from different choices in a document
  - Eliminate composite stories
    - Heuristic - same headline monthly or more often
  - High Support Cutoff (5%)
    - Eliminate overly frequent named entities (only provide "common knowledge" topics)

Named Entities vs. Full Text

- Corpus contained about 65,000 documents.
- Full text resulted in almost 5 million unique word-document pairs vs. about 740,000 for named entities.
- Prototype was unable to generate frequent itemsets at support thresholds lower than 2% for full text.
  - At 2% support, one week of full text data took 30 times longer to process than the named entities at 0.05% support.
- For one week:
  - 91 topics were generated with the full text, most of which aren't readily identifiable.
  - 33 topics were generated with the named-entities.
Frequent Itemsets

- **Query Flocks** association rule mining technique
  - 22894 frequent itemsets with 0.05% support
- Results filtered based on strength of correlation and support
  - Cuts to 3129 frequent itemsets
- Ignored subsets when superset with higher correlation found
  - 449 total itemsets, at most 12 items (most 2-4)

<table>
<thead>
<tr>
<th>Itemset</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Israel State</td>
<td>627390806</td>
</tr>
<tr>
<td>Iraq State</td>
<td>479</td>
</tr>
<tr>
<td>Israel Jerusalem</td>
<td>4989413</td>
</tr>
<tr>
<td>Gaza Netanyahu</td>
<td>39</td>
</tr>
<tr>
<td>Ramallah Authority</td>
<td>19506</td>
</tr>
<tr>
<td>Iraq Israel U.N.</td>
<td>39</td>
</tr>
</tbody>
</table>

Clustering

- **Cluster similar associations**
  - Hypergraph clustering based on hMETIS graph partitioning algorithm (adapted from (Han et. al. 1997))
  - Groups entities that may not appear together in a single broadcast, but are still closely related
    \[
    \sum_{j \in P} \text{Weight}(\text{cut_edges}_j) = \sum_{i \in P} \text{Weight}(\text{original_edges}_i)
    \]
TopCat Evaluation

- Tested on Topic Detection and Tracking Corpus
  - Six months of print, video, and radio news sources
  - 65,583 documents
  - 100 topics manually identified (covering 6941 documents)
- Evaluation results (on evaluation corpus, last two months)
  - Identified over 80% of human-defined topics
  - Detected 83% of stories within human-defined topics
  - Misclassified 0.2% of stories
- Results comparable to “official” Topic Detection and Tracking participants
  - Slightly different problem - retrospective detection
  - Provides “mnemonic” for topic (TDT participants only produce list of documents)