Content-Based Recommendation of RSS Feed Items

Michael Groble

Task Description

- Corpus
  - Approximately 17,000 items from my Google Reader history
  - Each labeled +1 if annotated as “starred”, -1 otherwise
- Attempt to learn classifier to predict next week’s starred items based on previous 16 weeks history

Performance Measure

- From corpus, only ~4% of items are marked starred
- Classification accuracy not the best metric
  - all -1 classification policy gives 96% accuracy
- Use F_{10} instead
  - Precision = correct hypothesized stars / all hypothesized stars
  - Recall = correct hypothesized stars / all true stars
  - F_{10} = 11 * Precision * Recall / (10* Precision + Recall)

Investigations

- Item Selection
  - Some items published to multiple feeds and get different labels (typically only label the first-read as starred)
  - “Unlabel” some non-starred items which are too similar to starred ones
    - Compare using all training data vs. filtering out unlabeled items vs. semi-supervised learning with unlabeled items
- Feature Selection
  - Compare unigram only vs. unigram + bigram features
  - Compare information gain vs. mutual information ranking
  - Consider latent topic probabilities
- Kernel Selection
  - Linear vs. Bhattacharyya kernels
- Algorithm Selection
  - Multinomial Naïve Bayes
  - Support Vector Machine

Example Latent Topics

- Most prevalent feed topic clusters (over all labels, April-August)

google	**iphone**	**video**	**microsoft**	**facebook**	**company**	**mobile**
search	**apple**	**content**	**open**	**platform**	**million**	**phone**
yahoo	**t**	**tv**	**source**	**users**	**quarter**	**service**
results	**jobs**	**youtube**	**software**	**myspace**	**cents**	**phones**
ebay	**store**	**videos**	**ibm**	**applications**	**says**	**wireless**
microsoft	**steve**	**live**	**community**	**developers**	**today**	**internet**
engine	**line**	**player**	**linux**	**application**	**stock**	**services**
internet	**stores**	**media**	**deal**	**friends**	**year**	**calls**
news	**ipod**	**television**	**patent**	**slide**	**billion**	**customers**
engines	**phone**	**channel**	**red**	**user**	**shares**	**new**
technorati	**new**	**channels**	**patents**	**social**	**share**	**verizon**
advertising	**sold**	**joost**	**license**	**growth**	**price**	**network**
data	**iphones**	**watch**	**open-source**	**apps**	**revenue**	**announced**
online	**launch**	**quality**	**hat**	**rockyou**	**market**	**devices**
ask	**today**	**movie**	**novell**	**app**	**sales**	**voice**

Results

- Best overall performance achieved through SVM with Bhattacharyya kernel on bigram features (F_{10} = 45.52%)
  - Filtered out 75% of items while retaining 70% of starred items
  - Both Unigram and Bigram Naïve Bayes with filtered similar items and top 5,000 features ranked by information gain produced very similar results to this best result
- Semi-supervised approaches did not improve F_{10}
- Latent topic features did improved accuracy and precision, but degraded recall and F_{10}