Text Classification of Reddit Posts

Jackie Gutman and Richard Nam

Introduction

Goal:

Create a system where a uncategorized reddit post can be suggested to be posted to the most relevant subreddit category

Corrolaries

I. Can features learned from one set of documents be used to infer features on a new set of documents?

II. Is the quality of the classification improved by incorporating information from the proxy measures of post quality/relevance (like the number of upvotes)?

Approach

- Apply different text classification models to the problem
  - Naive Bayes
  - Multinomial Logistic Regression
  - Support Vector Machines
  - Ensemble Methods
- Apply different feature extraction methods to generate input to the models
  - Bag-of-words unigram features
  - Bag-of-words n-gram (2-4) features
  - Weighted average Word2Vec embeddings
  - Doc2Vec (*Paragraph Vector*) embeddings

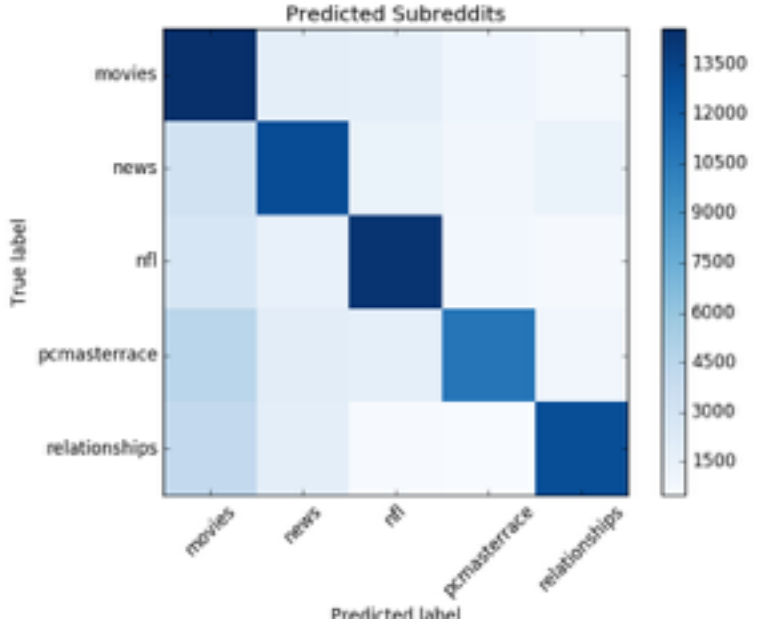
Data

- Reddit post data from Kaggle.com competition
- Originally 1.7 billion comments from May 2015
- Final dataset: 1 million comments drawn 5 popular subreddits
    - NFL, Relationships, News, Movies, PCMasterRace
- Distinct subset of available subreddit categories
- Posts that received more downvotes than upvotes removed
    - Should we train more heavily on higher-scoring posts (weighted bootstrap sample)?
    - Only helps if score is a good proxy for relevance
- Predict data from text only, metadata discarded

Baseline

- 3-gram Naive Bayes
- N-grams with less than 10 counts were dropped
- 494,609 dimensions
- Test accuracy: 0.656

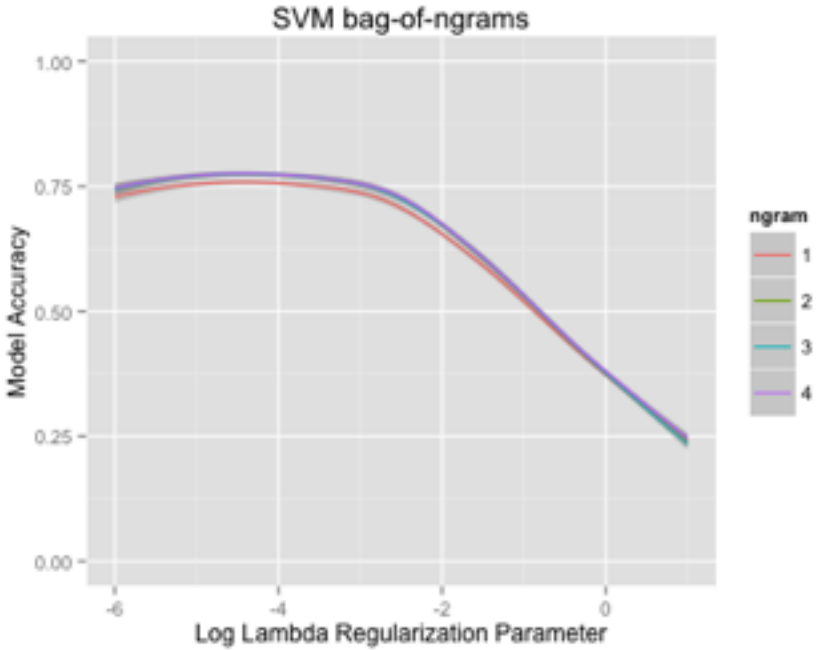
	precision	recall	f1-score
movies	0.50	0.73	0.59
news	0.65	0.65	0.65
nfl	0.72	0.72	0.72
pcmasterrace	0.75	0.54	0.63
relationships	0.78	0.65	0.71
avg / total	0.68	0.66	0.66



Overpredicts movies as subreddit label

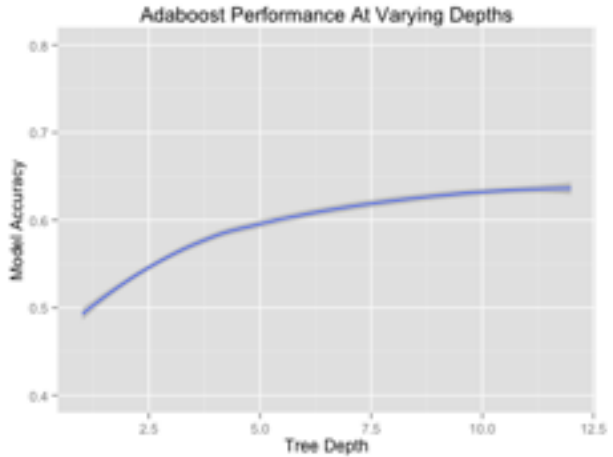
Logistic Regression and Support Vector Machine

- Logistic Regression
    - Multiple n-gram models 1-4
    - Weighted the training set
        - tuned on balance validation set
        - scored on balanced test set
- SVM
    - Multiple n-gram models 1-4
    - Test set score 0.78, 3rd order ngram
    - Tuning regularization hyperparameter C
    - L1, L2, hinge, squared-hinge
        - best: L2, hinge



Ensemble Methods : Adaboost

- Try non-linear approach
- Adaboost (incorrect classifications are weighted more heavily in updates)
- Decision Tree classifier



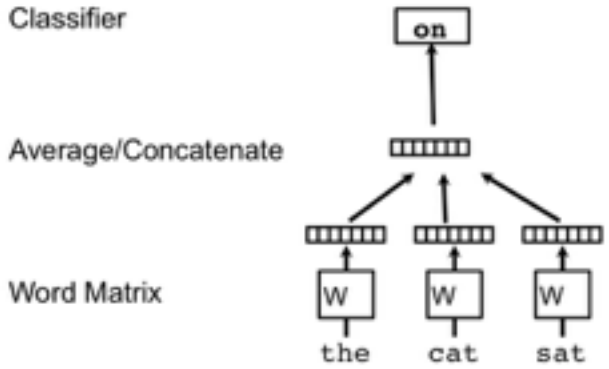
es model performance

xt data

- Performance on validation data has not yet leveled off with increasing depth
- Training may require a greater number of estimators
- Explicitly provide weights to predict balanced test set

Documents as Averaged Word Embeddings

- Train Word2Vec model over all words in all documents
  - Remove words that appear less than 10 times across all documents
- Take an (unweighted or weighted) average of all word vectors in each post
  - Weights are from TF-IDF model
  - Stopwords not included in average
- ovided as features to the classifier

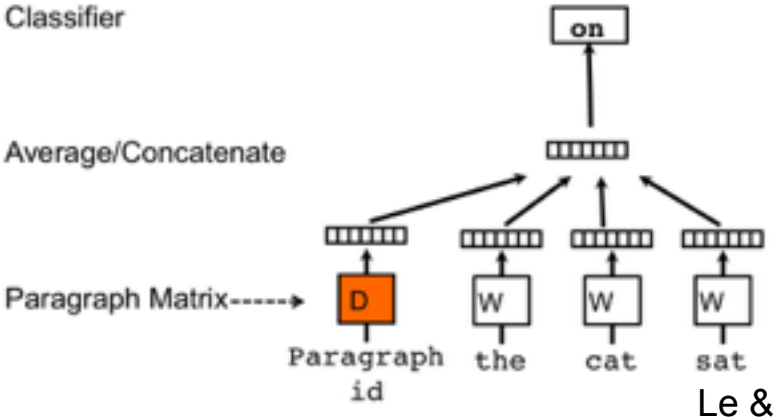


- Upweight vectors corresponding to high tf-idf word embeddings
- Downweight vectors corresponding to low tf-idf word embeddings
- Zero-weight vectors corresponding to English stopwords

Le & Mikolov, 2014

Document Embeddings: Paragraph Vector model

- CBOW Neural network learns two weight matrices simultaneously
  - Word embeddings for every word in the vocabulary, shared across all reddit posts
  - Document embeddings for every Reddit post in the corpus
- Two approaches for learning test set embeddings



Follow-up question:
Does concatenating the document and averaged word embeddings provide additional information beyond using either embedding alone?

Le & Mikolov, 2014

Preliminary Results

- N-gram models:
    - **Naive Bayes** *Accuracy:* .65, *Balanced Precision:* .68, *Macro F1: .66*
    - **Logistic Regression** *Accuracy:* .76, *Balanced Precision:* .78, *Macro F1:* .76
    - **SVM** *Accuracy:* .77, *Balanced Precision*: .77, *Macro F1:* .77
    - **Adaboost** *Accuracy* 0.64 (with depth 13 -- still training)
- Embedding models:
    - Use either inferred or separately learned neural networks on test data
    - Preliminary results don't look promising, but parameter tuning needed

Follow-up ideas

    - K-Means Clustering on document embeddings
    - Compute document similiarity within a particular subreddit