CS 175, Project in Artificial Intelligence

Lecture 6: Project Proposals

Padhraic Smyth
Department of Computer Science
Bren School of Information and Computer Sciences
University of California, Irvine
Announcements

- Project Proposals
  - Due Friday 11pm
  - Submit to EEE dropbox
  - Template and instructions on class website

- Lectures and Office Hours this week
  - Today: discussion of projects and proposals
  - Tuesday: Office hours, 10 to 11
  - Wednesday: no lecture.
    - I will be available to meet students in my office (2 to 3:20) to discuss projects

- Examples of Project Proposals from previous years
  - Posted on Piazza
## Weekly Schedule (subject to change)

<table>
<thead>
<tr>
<th>Week</th>
<th>Monday</th>
<th>Wednesday</th>
<th>Deadlines</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan 9</td>
<td>Introduction and course outline</td>
<td>Basic concepts in automated text analysis</td>
<td></td>
</tr>
<tr>
<td>Jan 16</td>
<td>No class (university holiday)</td>
<td>Text classification</td>
<td>Assignment 1 (by Wednesday noon)</td>
</tr>
<tr>
<td>Jan 23</td>
<td>Text classification (continued)</td>
<td>Ideas for class projects</td>
<td>Assignment 2 (by Friday noon)</td>
</tr>
<tr>
<td>Jan 30</td>
<td><strong>Discussion of project proposals</strong></td>
<td>Office hours (no lecture)</td>
<td><strong>Project proposal (by Friday 11pm)</strong></td>
</tr>
<tr>
<td>Feb 6</td>
<td>Evaluation methods</td>
<td>Word prediction methods</td>
<td></td>
</tr>
<tr>
<td>Feb 13</td>
<td>Topic modeling algorithms</td>
<td>Office hours (no lecture)</td>
<td></td>
</tr>
<tr>
<td>Feb 20</td>
<td>No class (university holiday)</td>
<td>Office hours (no lecture)</td>
<td>Progress report (by Friday noon)</td>
</tr>
<tr>
<td>Feb 27</td>
<td>Office hours (no lecture)</td>
<td>Office hours (no lecture)</td>
<td></td>
</tr>
<tr>
<td>Mar 6</td>
<td>Project Presentations (in class)</td>
<td>Project Presentations (in class)</td>
<td></td>
</tr>
<tr>
<td>Mar 13</td>
<td>Office hours (no lecture)</td>
<td>Office hours (no lecture)</td>
<td></td>
</tr>
</tbody>
</table>
## Project Deliverables and Deadlines

<table>
<thead>
<tr>
<th>Deliverable</th>
<th>Grade Percentage</th>
<th>Date Due</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Proposal</td>
<td>20%</td>
<td>Friday Feb 3rd</td>
</tr>
<tr>
<td>Progress Report</td>
<td>20%</td>
<td>Friday Feb 24th</td>
</tr>
<tr>
<td>Presentation/Demo</td>
<td>10%</td>
<td>March 6(^{th})/8th</td>
</tr>
<tr>
<td>Final Report</td>
<td>30%</td>
<td>Monday March 20th</td>
</tr>
</tbody>
</table>
Rules for Projects

• 1 or 2 students per project
  – For 2-student projects I expect twice as much work as 1-student projects
  – Submit only one project proposal per team (include both team member names and IDs on the proposal)
  – Both team members will get the same grade for the proposal.

• Use of external code is allowed and encouraged
  – Such code needs to be acknowledged in your reports

• You must write at least some functionality on your own
  – What you implement is up to you
  – You will need to submit the code at the end of the quarter
Planning a Project

- Topic: select a general type of problem you are interested in, e.g.,
  - Classification, information extraction, summarization, synthesis, parsing, etc

- Do some background reading to learn more about the topic
  - e.g., from chapters in the books I recommended
  - e.g., from links on class Web site

- Define your problem precisely

- Determine at least 1 or 2 data sets you can use for your project

- Figure out how you will evaluate your results, e.g., to compare A v B
  - Experiments: classification accuracy, precision/recall, etc
  - User Studies: human users compare results from A and B
Project-Related Reference Material for CS 175

CS 175, Winter 2017
Below are links to suggested reading organized by topic. If you are doing a project on any of these topics (or interested in potentially doing a project on these topics) then these online resources should be helpful.

Text Classification
Chapter on text classification and naive Bayes from Manning et al
Chapter on vector-based classification for text from Manning et al
Chapter on logistic regression for document classification from Jurafsky and Martin
Comprehensive survey paper on text classification algorithms by Aggarwal and Zhai (2012)
Overview of general principles in machine learning from Goodfellow et al (2016)
Tutorial paper on multi-label classification methods by de Carvalho and Freitas

Sentiment Analysis
Chapter on naive Bayes and sentiment classification from Jurafsky and Martin
Survey papers on opinion mining and sentiment analysis: by Pang and Lee (2008) and by Liu and Zhang (2012)

Sequential Classifiers
Chapter on recurrent and recursive neural networks from Goodfellow et al (2016)

Document Clustering
Chapters on flat clustering algorithms and hierarchical clustering algorithms for text documents, from Manning et al

Topic Modeling
Tutorial paper on topic modeling by Steyvers and Griffiths
David Blei’s page on topic modeling

Vector Embeddings
Chapter on dense vector representations for words from Jurafsky and Martin
Chapter on latent semantic indexing from Manning et al
Examples of Data Sets for Text Analysis

CS 175, Winter 2017

The links below point to just a few of the many data sets for text analysis that you can find on the Web, and should help you in terms of finding data sets to use for your projects.

Data Sets with Classification Labels or Ratings

Yelp Data Set Challenge (2.2M reviews of businesses from over 500k users in 10 cities)
(and here's a pointer to work from our own group at UCI that recently won the Round 5 Challenge)
Kaggle Data Sets. Contains multiple data sets with text content. Kaggle is a company that hosts data mining/prediction competitions
Movie review data for sentiment analysis, from Pang and Lee, Cornell
Product review data from Johns Hopkins University (goal is to predict ratings on scale of 1 to 5)
A variety of different text data sets from the UCI Machine Learning Repository (many already in the "bag of words" format)
Data Sets on "learning to rank" (for Web search)
All of Wikipedia (can be used to build classifiers using category labels or to provide additional information for other models such as n-gram statistics)
Various text and Web-related data sets from Yahoo! Labs (note that these data sets can also be used for unsupervised learning, such as clustering or topic detection by ignoring the class labels during training).
Document classification data sets (a large collection of different data sets used in text classification research)

Other Interesting Text Data Sets (often used for Clustering and other Exploratory Methods)

Enron email data set, from CMU (note that there are other "cleaner" versions available on the Web if you search...)
Python code for downloading IMDB (Internet Movie Database), with 425k titles and 1.7 million filmographies of cast and crew
A survey of data sets available for building data-driven dialogue systems
Book Summaries Corpus
Full text of US patents from 1980 to 2015, from the USPTO (US Patent and Trademark Office), hosted by Google
Very large data set of all Reddit submissions between 2006 and 2015

Data Sets used to build Language Models and Auto-complete Algorithms

Ngram data from Peter Norvig (Google), with an accompanying tutorial book chapter
Google ngrams, and Google syntactic ngrams over time, from Google books

Question-Answering Data Sets

WikiQA, a data set for "open-domain" question answering, from Microsoft Research
Question-Answering Data Sets from TREC (funding by the National Institute of Standards and Technology, NIST)
Question Answering Corpus from DeepMind (part of Google)
The Allen AI Science Challenge on Kaggle (competition ended in 2016)
The BioASQ data sets and challenge competitions on question answering for the biomedical domain
“everything you’d expect - but nothing more”

Sentiment Analysis on Movie Reviews
Fri 28 Feb 2014 – Sat 28 Feb 2015 (23 months ago)

Data Files

<table>
<thead>
<tr>
<th>File Name</th>
<th>Available Formats</th>
</tr>
</thead>
<tbody>
<tr>
<td>test.tsv</td>
<td>.zip (470.84 kb)</td>
</tr>
<tr>
<td>sampleSubmission</td>
<td>.csv (582.06 kb)</td>
</tr>
<tr>
<td>train.tsv</td>
<td>.zip (1.21 mb)</td>
</tr>
</tbody>
</table>

The dataset is comprised of tab-separated files with phrases from the Rotten Tomatoes dataset. The train/test split has been preserved for the purposes of benchmarking, but the sentences have been shuffled from their original order. Each Sentence has been parsed into many phrases by the Stanford parser. Each phrase has a Phraselab. Each sentence has a SentenceID. Phrases that are repeated (such as short/common words) are only included once in the data.

- train.tsv contains the phrases and their associated sentiment labels. We have additionally provided a SentenceID so that you can track which phrases belong to a single sentence.
- test.tsv contains just phrases. You must assign a sentiment label to each phrase.

The sentiment labels are:

0 - negative  
1 - somewhat negative  
2 - neutral  
3 - somewhat positive  
4 - positive
Questions from Office Hours

Question:
We need to develop a crawler and parser to gather data for our project (e.g., crawling Web sites for resumes and job ads). Is the work on developing a crawler and parser be considered part of our project?

Answer:
Yes, this can be counted as part of your project work.

But it should not take too large a fraction of your time (e.g., 1 week or so at most)

Otherwise it means you are spending too much time just on collecting data rather than developing algorithms.
Questions from Office Hours

Question:
We are working on a challenging text classification problem where there are 10 different classes (e.g., different types of sentiment, different genres of movies or music, etc). We know from reading research papers on this topic that it can be difficult classify data into all 10 classes accurately. What do you suggest?

Answer:

It is fine (particularly when starting out) to select a subset of the easier classes to work with, rather than all of the classes. For example, in movies you could pick Horror versus Romance, and ignore all other classes.

This will allow you to test your classifier on a problem where you expect to be able to accuracy better than random

You can then work up to more complex problems, e.g.,

- 3 classes, e.g., the 2 original classes plus a 3rd class consisting of all of the other classes
Examples of Projects from past CS 175 Classes

• Automatically matching resumes to jobs

• Predicting what subreddit a Reddit post should go to or its popularity

• Predict whether a restaurant will close or not from review data

• Identifying the genre of a movie or song

• Simulating text using neural networks

• Rating the quality of answers to questions on StackOverflow

• Autocomplete for ICD codes for medical professionals

• Generating spatial maps of user happiness from Twitter
Project Proposals

- Your proposal should be 2 to 3 pages long
  - Required to use [project proposal template](#) (will be posted by this weekend)

- Project proposals will be graded like a homework assignment and receive a weight of 20% of your overall grade.

- Proposals will primarily be graded on
  - (a) clarity (is it clear what will be done in this project?) and
  - (b) completeness (does the proposal address all of the important aspects of the proposed project?)

- Note: if a project is too simple (or too complex!), or missing important details, it may be returned to you and a revision requested.

(Assignment for Project Proposals will be available on Webpage by end of day Friday)
Contents of Project Proposal

1. Project Summary

2. Problem Definition

3. Proposed Technical Approach

4. Data Sets

5. Experiments and Evaluation

6. Software

7. Milestones

8. (For Teams) Individual Student Responsibilities
1. Project Summary

A clear description (2 or 3 sentences) that summarizes your project: e.g., “This project will use XX methods to predict YY using the Z1 and Z2 data sets, with evaluation using classification accuracy and user studies.”

Examples:

• The goal of this project is to explore spam filtering by classifying SMS messages as spam or not spam using various machine learning techniques.

• Application and evaluation of multiple sentiment analysis classification algorithms using multiple datasets. Evaluation of correctness, uncertainty, and differences between the algorithms.

• Our project will be a poem generator that takes in a dictionary and outputs a poem that looks as if a human created the poem. We will be using various algorithms and libraries, such as NLTK and NodeBox, to identify different grammars and apply constraint satisfaction to make a poem that is clear and readable.
2. Problem Definition

Write a few sentences that clearly defines what problem you will be trying to solve. One way to describe this is to think about your project in terms of inputs and outputs: what will the inputs to your system be and what will it produce as output?

Example:

Sentiment analysis (or opinion mining) is the technique used to correctly classify subjective information through natural language processing. Our specific project will address the problem of performing sentiment analysis on Rotten Tomatoes movie reviews. Based on the phrases located in the movie reviews, we will attempt to assign a sentiment class label between 0 and 4, which represents how negative or positive their review is about the movie. Popular methods for sentiment analysis include support vector machines, “bag of words,” and neural networks and deep learning.
3. Proposed Technical Approach

- Write a paragraph with a clear description of the methods and algorithms you plan to use on the project.

- If the system you are building can be thought of as a pipeline with multiple components feel free to provide a figure that illustrates the pipeline with blocks for different components and brief descriptions of each component.
3. Proposed Technical Approach (Example)

We are going to divide the whole project into several stages.

• The first stage is to preprocess the movie review. We plan to use stopwords list from NLTK to remove stopwords, punctuations and non-alphabetic words. Secondly, we plan to obtain the base part of the word by using Snowball as a stemmer to remove morphological endings. The Porter Stemming Algorithm will be implemented in this step. Then, we plan to extract opinioned words by applying the positive words list from NLTK.

• The second stage is to classify the opinioned words list obtained from the first stage.

• The third stage is training our classifier to recognize the attitude of the reviews. We plan to try logistic regression, support-vector machines, and Multinominal Naive Bayes models for supervised training.
3. Proposed Technical Approach (Example)
3. Proposed Technical Approach (Example)

Our Algorithm

For each unique food product ID, we look at every sentence of individual reviews and remove all the stopwords (Python NLTK stopwords and our custom set of stopwords). This stopword removal step ensures that irrelevant, unhelpful key phrases do not get added to our set of key phrases. We then extract all n-grams that begin with an adjective and end with a noun (and vice versa) from the filtered reviews. We rank these key phrases based on their values of PF-IRF (phrase frequency - inverse review frequency), which is a variation of TF-IDF (term frequency - inverse document frequency). Finally, we can generate new sentences using only the key phrases with high rankings, along with a Markov model simulating common sentence structures.
4. Data Sets

• Briefly describe what data set(s) you plan to use in the project. Include references to the data (e.g., a URL) if you can. ......

• If you are able to access and take an initial look at your data, feel free to also include a figure or two in this section, e.g., a histogram of document lengths.

• You can change your data sets during the project if you need to, but you should have identified at least one data set to work with by the time you submit the proposal.
4. Data Sets (Example)

- We plan to work with the Rotten Tomatoes movie review dataset publicly provided on Kaggle.com as the basis for a machine learning competition. The data set is a collection of sentences from reviews that are parsed into phrases by the Stanford Parser. The data set is preprocessed with a predefined vocabulary that simply removes repeated common or short phrases. We plan to improve this simple vocabulary through our own method of preprocessing as discussed in the next section.
## 4. Data Sets (Example)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Length</th>
<th>Type</th>
<th>Classified</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yelp RSS feed</td>
<td><a href="http://www.yelp.com/rss">http://www.yelp.com/rss</a></td>
<td>25 reviews for 5 cities per day (125) Need to collect (1000+)</td>
<td>Short: around 300 characters</td>
<td>Review Yes: each has a rating (N/5)</td>
</tr>
<tr>
<td>Song Lyrics</td>
<td><a href="http://www.songlyrics.com/">http://www.songlyrics.com/</a></td>
<td>Need to collect</td>
<td>Medium: Around 2,500 characters</td>
<td>Song, Poetry No</td>
</tr>
<tr>
<td>Twitter Tweets</td>
<td><a href="http://www.sananalytics.com/lab/twitter-sentiment/">http://www.sananalytics.com/lab/twitter-sentiment/</a></td>
<td>5513 tweets Pre-packaged</td>
<td>Short: max 140 characters</td>
<td>Message Yes: hand classified</td>
</tr>
<tr>
<td>Movie Reviews</td>
<td><a href="http://www.cs.cornell.edu/people/pabo/movie-review-data/">http://www.cs.cornell.edu/people/pabo/movie-review-data/</a></td>
<td>polarity_dataset_v2.0 1000 positive, 1000 negative reviews Pre-Packaged</td>
<td>Medium-Long</td>
<td>Review Yes: positive/negative</td>
</tr>
</tbody>
</table>
4. Data Sets (Example)
4. Data Sets (Example)
5. Experiments and Evaluation

- Provide a brief and clear description of how you will evaluate the results of your project, e.g., accuracy for classification, precision-recall for document ranking.

- Aspects to consider
  - Single metrics: classification accuracy
  - Curves: precision-recall
  - Test sets and cross-validation
  - User studies
5. Experiments and Evaluation (Examples)

- The data set comes in a set of two files: one train.tsv file and one test.tsv file. We will essentially use cross validation to split our training data set into a validation set and a testing set in order to evaluate our models. We could also use our models on the test.tsv file and upload a submission file onto Kaggle, which will evaluate our models and give us a corresponding score on the leaderboard. The Kaggle leaderboard currently has about 700 teams, who are ranked by how well their model performs on the test data provided in test.tsv - this can give us an accurate indication on how well our model performs in a more realistic setting.
6. Software

• Provide a list of the major pieces of project software that you expect to use, divided into 2 sets:
  – (1) publicly-available code, and
  – (2) code will write yourself.

• This list will probably be incomplete at this point (which is fine) since you may not know yet about all of the publicly-available software that might be relevant to your project
6. Software (Example)

Publicly-available code:

- **NLTK**: provides a list of stop words and build-in naive Bayes classifier.
- **PyEnchant**: provides spell checking and spelling suggestions.
- More later.

Code to be written ourselves in Python:

- **Tokenizer** to parse SMS message.
- Generate a feature list using tokenized message.
- Group misspelled words using PyEnchant suggestions.
- Track metadata such as number of misspellings in a message.
- Bernoulli naive Bayes to analyze features and classify messages.
6. Software (Example)

<table>
<thead>
<tr>
<th>Publicly-Available Code</th>
<th>Code We Will Write/Test</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Programming Languages:</strong></td>
<td><strong>Ranking Algorithms:</strong></td>
</tr>
<tr>
<td>• Python 3.5 and libraries such as NLTK, NumPy, SciPy, and Matplotlib</td>
<td>• Determine the phrase rank for a product by computing the value of PF-IRF measure for each phrase. Phrases with high PF-IRF ranks are selected and included in our summary.</td>
</tr>
<tr>
<td>• SQLite3</td>
<td>• <strong>Comparison algorithm</strong> to evaluate similarity between RAKE’s phrases and our algorithm’s phrases.</td>
</tr>
<tr>
<td><strong>Keywords Extraction:</strong></td>
<td></td>
</tr>
<tr>
<td>• RAKE (Rapid Automatic Keyword Extraction) using NLTK</td>
<td></td>
</tr>
<tr>
<td><strong>Evaluation Software:</strong></td>
<td><strong>Phrase Extraction Algorithm:</strong></td>
</tr>
<tr>
<td>• ROUGE software package to automate the evaluation of our results.</td>
<td>• Extract frequent n-grams from all reviews of a product.</td>
</tr>
<tr>
<td></td>
<td>• Extract key phrases from the frequent n-grams, phrases that contain adjectives followed by nouns (and vice versa) with stopwords eliminated.</td>
</tr>
<tr>
<td></td>
<td><strong>Sentence Generation Algorithm:</strong></td>
</tr>
<tr>
<td></td>
<td>• Uses sentence structure and a <strong>Markov chain</strong> to generate readable sentences containing descriptive phrases.</td>
</tr>
</tbody>
</table>
7. Milestones

• Provide a brief list of milestones. For example, since the project will span 6 weeks of the class (weeks 5 to 10), you could break your milestones into a list of 3 intermediate phases:
  – Weeks 5 and 6
  – Weeks 7 and 8
  – Weeks 9 and 10

• For example, much of the data gathering and preprocessing and coding (development and test) could happen in the earlier weeks, and much of the experimentation and evaluation in the later weeks. Note that you have a progress report due at the end of week 7.
7. Milestones (Example)

Weeks 5 and 6:
- Search for additional data sets
- Write basic tokenizer and generate feature list using most common words.
- Test accuracy of scikit-learn Bernoulli naive Bayes and multinomial naive Bayes using the basic tokenizer. (If multinomial turns out to be more accurate, then we will implement multinomial instead).

Weeks 7 and 8:
- Write Bernoulli naive Bayes classifier, and test accuracy of basic algorithm
- Compare accuracy with scikit-learn Bernoulli naive Bayes and Multinomial naive Bayes.
- Improve tokenizer and feature list generation.
- Use PyEnchant to group misspelled words.

Weeks 9 and 10:
- Improve tokenizer and feature list generation.
- Explore tracking different metadata features.
8. Individual Student Responsibilities (Teams Only)

Summarize briefly what each student will be primarily responsible for in the project. For example, you might write something like this

- **Name 1**: will write and test the code for Algorithms 1 and 2, will integrate components A and B in the pipeline, will assist in doing experiments and interpreting results, will assist in writing project reports

- **Name 2**: will acquire the data sets to test the algorithms, will preprocess the text data (e.g., define the vocabulary for the algorithms), will implement Algorithm 3 and integrate all the components into a pipeline, will write the scripts for evaluating the accuracy of the algorithms, will assist in writing project reports.

[Note these are just suggestions – you can and should organize responsibilities in whatever way makes sense.....]
8. Individual Student Responsibilities (Example)

**Student 1:**

*will write and test the code for our PF-IRF Ranking Algorithm, contribute the rest of his time generating grammatically correct and logical sentences for our summaries, and assist in writing project reports.*

**Student 2:**

*will acquire the relevant data from our dataset to feed into our algorithms, preprocess the text data, write the scripts for evaluating the accuracy of the algorithms, write and test the code for our Sentence Generator using ROUGE, and assist in writing project reports.*
9. References & Links


Planning and Organization of Projects
Project Tips: Goals

• Be clear in your goals
  – e.g., “will systematically evaluate the accuracy of logistic regression and neural network classifiers on the Reuters data set and 2 other data sets”
  – Ok to not to have all the details of how you will get there, but important to know what the goal is
  – Ok if goals are updated/changed as you learn more about the problem

• Team members should agree on the goals
  – To be effective the team needs to be clear about the goals
  – If there is any doubt about what the goal is, ask questions and discuss
Project Tips: Plan in Stages

Plan your project in stages so that the overall project is not dependent on the riskier elements working.

Example:

PHASE 1

Original Documents \(\rightarrow\) Standard Bag of Words \(\rightarrow\) Standard Logistic Regression \(\rightarrow\) Cross-Validation Experiments
Project Tips: Plan in Stages

Plan your project in stages so that the overall project is not dependent on the riskier elements working.

Example:

PHASE 1

Original Documents → Standard Bag of Words → Standard Logistic Regression → Cross-Validation Experiments

PHASE 2

Bag of Phrases (ngrams) → Standard Logistic Regression
Plan your project in stages so that the overall project is not overly dependent on the riskier elements.

Example:

PHASE 1: Original Documents  →  Standard Bag of Words  →  Standard Logistic Regression  →  Cross-Validation Experiments

PHASE 2: Bag of Phrases (ngrams)  →  Deep Neural Network

PHASE 3
Project Tips: Evaluation Methods

- Very important to have a clear idea of how you will evaluate your system

- For some tasks, such as document classification, there are well-defined metrics that are straightforward
  - E.g., cross-validated classification accuracy

- For other tasks, such as clustering, you will have to do some research to figure out what metrics are appropriate
  - For some projects, some user evaluation may be necessary

- Always include a baseline method in your experiments
  - E.g., for classification your baseline could be a Naïve Bayes classifier
Project Tips: Revision/Source Control for Code

• Teams should consider using a collaborative revision control system
  – e.g., Github accounts (freely available)
  – If you are not familiar with these tools, this is a good time to learn

• Revision control
  – Provides a systematic way for a team to develop code, scripts, documents, etc
  – Individuals can “check out” code, work on it, and then “commit”
  – Earlier versions of code can be recovered
    • Useful when you want to go back to an earlier version without a bug
Project Tips: Revision Control for Experiments

• You are likely to conduct many experiments over the course of the project, comparing versions of preprocessing, parameter settings, algorithms
  – You want to be able to keep track of your experiments and results in a systematic way

• Recommendations:
  – Organize and document your code/scripts for experiments
  – Use time-stamps, give your scripts interpretable names, use comments

• Consider using tools such as IPython Notebook, with github, for documentation and collaboration
  – just like a “lab notebook” in a science lab

• Reproducibility is important
  – Scripts will allow to you to regenerate results from earlier in the project
  – Also allows team members to share results efficiently
General Project Tips: Writing

• Write clearly
  – Try to put yourself in the mind of the person reading it
  – Make sure you don’t leave out important details and concepts
  – Use spell-checkers, grammar-checkers, etc
  – Target audience = a random student in the class

• Work collaboratively
  – Teams can use a shared document with version control, e.g., Google Docs
  – Have each member of the team edit, read, comment on the document
  – Work on your documents collaboratively: will lead to better documents

• Use figures and tables when you can
  – “A picture is worth a 1000 words”
Examples of Projects from Past Years
## Examples of Types of Projects

<table>
<thead>
<tr>
<th>Type</th>
<th>Goal</th>
<th>Examples of Data Sets</th>
<th>Evaluation Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classification</td>
<td>Predict sentiment (pos/neg) or stars from text</td>
<td>Twitter, Yelp, Product or Movie Reviews</td>
<td>Classification accuracy, Precision-recall, ...</td>
</tr>
<tr>
<td></td>
<td>Predict dialog acts in transcribed conversations</td>
<td>Switchboard corpus</td>
<td></td>
</tr>
<tr>
<td>Text Summarization</td>
<td>Summarize a set of documents</td>
<td>Product or Movie Reviews</td>
<td>BLEU scores, User studies</td>
</tr>
<tr>
<td>Information Extraction</td>
<td>Product or restaurant reviews</td>
<td>Product Reviews News Articles</td>
<td>Accuracy (if labeled), otherwise user studies</td>
</tr>
<tr>
<td>Text Synthesis</td>
<td>Generate new text in the style of an author</td>
<td>Articles/books/songs by different authors</td>
<td>User studies</td>
</tr>
<tr>
<td>Question Answering</td>
<td>Generate an answer to a question</td>
<td>Q&amp;A data sets for research projects</td>
<td>Accuracy, precision-recall</td>
</tr>
</tbody>
</table>
## Projects on Document Classification

<table>
<thead>
<tr>
<th>Project Name</th>
<th>Data Sets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis of financial journalism for market prediction</td>
<td>Reuters articles</td>
</tr>
<tr>
<td>Evaluating the accuracy of three distinct single-label text classification models</td>
<td>Reuters articles, and possibly Wikipedia</td>
</tr>
<tr>
<td>Reddit miner</td>
<td>Reddit posts</td>
</tr>
<tr>
<td>Document reading level classification</td>
<td>Reddit posts</td>
</tr>
<tr>
<td>Wikipedia pages and what pages are about cats</td>
<td>Wikipedia pages</td>
</tr>
<tr>
<td>Classification of documents by author</td>
<td>Gutenberg Books</td>
</tr>
</tbody>
</table>
## Projects on Text Generation/Simulation

<table>
<thead>
<tr>
<th>Project Name</th>
<th>Data Sets</th>
</tr>
</thead>
<tbody>
<tr>
<td>The botinator chatbot system</td>
<td>IMDB</td>
</tr>
<tr>
<td>Author simulator 2015</td>
<td>General text</td>
</tr>
<tr>
<td>Poetry generation</td>
<td>General text</td>
</tr>
<tr>
<td>Text simulation</td>
<td>General text</td>
</tr>
<tr>
<td>Grammar-learning and sentence-generating AI</td>
<td>General text</td>
</tr>
<tr>
<td>Project Name</td>
<td>Data Sets</td>
</tr>
<tr>
<td>------------------------------------------------------------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>Rotten or fresh? An exploration on the freshness of movies</td>
<td>Rotten Tomatoes movie reviews</td>
</tr>
<tr>
<td>Rotten tomatoes movie review classification with machine learning and NLP</td>
<td>Rotten Tomatoes movie reviews</td>
</tr>
<tr>
<td>Analysis of different algorithms in classifying reviews</td>
<td>Product reviews</td>
</tr>
<tr>
<td>Reach for the stars: prediction of product review star ratings</td>
<td>Product reviews</td>
</tr>
<tr>
<td>Foodiecity</td>
<td>Facebook, Yelp, Twitter</td>
</tr>
</tbody>
</table>
## Projects on Sentiment Analysis

<table>
<thead>
<tr>
<th>Project Title</th>
<th>Data Sets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Confidence and sentiment analysis algorithms</td>
<td>Rotten Tomatoes movie reviews</td>
</tr>
<tr>
<td>News and stock price relationship via sentiment analysis and linear regression</td>
<td>News articles</td>
</tr>
<tr>
<td>Technology stock market predictions based on twitter trends</td>
<td>Twitter</td>
</tr>
<tr>
<td>Tonal analysis of tweets</td>
<td>Twitter</td>
</tr>
<tr>
<td>Sentiment analysis</td>
<td>Twitter</td>
</tr>
<tr>
<td>Sentiment analysis of trends on microblogs</td>
<td>Twitter and/or Facebook</td>
</tr>
<tr>
<td>BladeRunner, sentiment analysis on Twitter tweets</td>
<td>Twitter</td>
</tr>
</tbody>
</table>
## Projects on Other Topics

<table>
<thead>
<tr>
<th>Project Title</th>
<th>Data Sets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Twitter trend detection</td>
<td>Twitter</td>
</tr>
<tr>
<td>Exploring spam filtering by applying various classifiers with SMS spam data sets</td>
<td>SMS texts</td>
</tr>
<tr>
<td>AutoCorrect</td>
<td>Any text</td>
</tr>
<tr>
<td>AlgorithmicQuestGenerator</td>
<td>Gutenberg Books</td>
</tr>
</tbody>
</table>
Announcements

• Project Proposals
  – Due Friday 11pm
  – Submit to EEE dropbox
  – Template and instructions on class website

• Lectures and Office Hours this week
  – Today: discussion of projects and proposals
  – Tuesday: Office hours, 10 to 11
  – Wednesday: no lecture.
    • I will be available to meet students in my office (2 to 3:20) to discuss projects

• Examples of Project Proposals from previous years
  – Posted on Piazza