CMSC 25025 / STAT 37601

Machine Learning and Large Scale Data Analysis

Assignment 1

Due: Thursday, April 11, 2013

This assignment consists of four problems. Two of them are simple "pencil and paper" problems. Two involve some computation on the State of the Union data, using the course Amazon AWS infrastructure.

1. *Probability* (20 points)

   (a) Let X have a continuous, strictly increasing cdf F. Let $Y = F(X)$. Find the density of Y. Now let $U \sim \text{Uniform}\,(0,1)$ and let $X = F^{-1}(U)$. Show that $X \sim F$.

   (b) Let $X, Y \sim \text{Uniform}\,(0,1)$ be independent. Find the pdf for $Z = X - Y$ and $Z = \min\{X, Y\}$.

   (c) Let $X \sim N(0,1)$ and let $Y = e^X$. Find $\mathbb{E}(Y)$ and $\text{Var}(Y)$.

   (d) Prove that $\text{Var}(Y) = \mathbb{E}\,\text{Var}(Y \mid X) + \text{Var}\,\mathbb{E}(Y \mid X)$.

2. *Properties of the Hat Matrix* (10 points)

   In linear regression, the fitted values are defined to be $\widehat{\mathbf{y}} = \mathbf{X}\widehat{\beta}$ where $\widehat{\mathbf{y}} = \mathbf{H}\mathbf{y}$ and

   $$\mathbf{H} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T.$$

   The matrix $\mathbf{H}$ is called the "hat matrix." Define $\mathcal{L}$ to be the set of vectors that can be obtained as linear combinations of the columns of $\mathbf{X}$, which is an $n \times d$ matrix. Show that the hat matrix satisfies the following properties:

   (a) $\mathbf{H}\mathbf{X} = \mathbf{X}$.

   (b) $\mathbf{H}$ is symetric: $\mathbf{H} = \mathbf{H}^T$.

   (c) $\mathbf{H}$ is idempotent: $\mathbf{H}^2 = \mathbf{H}$.

   (d) $\widehat{\mathbf{y}} = \mathbf{H}\mathbf{y}$ is the projection of $\mathbf{y}$ onto the column space $\mathcal{L}$.

   (e) $\text{rank}(\mathbf{X}) = \text{tr}(\mathbf{H}) = d$.

3. *The Longest-Winded President?* (35 points)

   In this problem, you will analyze the distribution of sentence lengths in the State of the Union addresses. The way you should approach this problem is parallel to what Charles demo'ed in Thursday's class, when he fit a regression model to the length of the SOTU speeches.

(a) For each year, extract the sentence lengths from the SOTU address in that year. You will need to parse the addresses and find end-of-sentence markers. Don't worry about being too precise about sentence boundaries—as a first approximation, you could find words ending in a period. (But what about "Mr."?)

(b) For each year, compute the number of sentences in the address, and the mean sentence length for that year. Plot these data and two linear regressions, one plot for the number of sentences by year, another for the average sentence length by year. Describe the trends that you see, and give some explanation for them.

(c) Which Prez has the longest sentences on average? Which has the shortest sentences? Compute the median, 25% and 75% quantiles across all Presidents. What was the longest and shortest sentence ever spoken (or written) in a SOTU?

(d) Now, select about ten of the Presidents. For each one that you select, plot the empirical probability distribution of the sentence lengths in all of his SOTUs. You should bin the lengths to get a reasonably smooth histogram. Now, fit the maximum likelihood Poisson model to the sentence lengths. Plot this Poisson and compare it to the empirical distribution. (You will get a better visualization if you plot the Poisson as a curve.) Don't forget to label your axes (`http://xkcd.com/833/`). What can you say about the fit of the model to the data? Can you suggest a better model?

Hand in your solution as an iPython notebook that will run on a generic cluster in our AWS infrastructure.

4. *Lend Me Your Vectors* (35 points)

In this problem you will use the classic *vector space model* from information retrieval to find similar SOTU addresses.

In the vector space model, a document of words d is represented by a TF-IDF vector $\mathbf{w}(d) = (w_1(d), w_2(d), \ldots, w_V(d))$ of length V, where V is the total number of words in the vocabulary. The TF-IDF weights are given by

$$w_i(d) = n_i(d) \cdot \log\left(\frac{|\mathcal{D}|}{\sum_{d' \in \mathcal{D}} \mathbb{1}(t_i \in d')}\right)$$

where $n_i(d)$ is the number of times term t_i appears in document d, $\sum_{d' \in \mathcal{D}} \mathbb{1}(t_i \in d')$ is the number of documents that contain term t_i, and $|\mathcal{D}| = \sum_{d' \in \mathcal{D}} 1$ is the total number of documents in the collection $\mathcal{D}$. This weighting scheme favors terms that appear in few documents.

You will construct an iPython notebook that uses this representation to find similar SOTUs.

(a) Compute the TF-IDF vectors for each SOTU address. You should lower case all of the text, and remove punctuation. For example, you could use something like this:

```
s = s.lower().translate(string.maketrans("",""), string.punctuation)
```

You will have to make choices about the size of the term vocabulary to use—for example throwing out the 20 most common words, and words that appear fewer than, say, 50 times.

(b) A similarity measure between documents is

$$\text{sim}(d, d') = \frac{\mathbf{w}(d) \cdot \mathbf{w}(d')}{\|\mathbf{w}(d)\| \|\mathbf{w}(d')\|},$$

the cosine of the angle between the corresponding TF-IDF vectors. In terms of this measure, find the

- 50 most similar pairs of SOTUs given by different Presidents.
- 50 most similar pairs of SOTUs given by the same President.
- 25 most similar pairs of *Presidents*, averaging the cosine similarity over all pairs of their SOTUs.

When you read the above speeches, do they indeed seem similar to you? (You can read the speeches in a more reader-friendly format here: `http://www.presidency.ucsb.edu/sou.php`) Comment on what you find, and describe what is needed to construct a better similarity measure between documents.

Although the SOTU dataset is not very large, you should try to exploit parallelism whenever possible in order to become familiar with this paradigm.

Hand in your solution as an iPython notebook that will run on a generic cluster in our AWS infrastructure.