ASSIGNMENT 1

COMP 550, Fall 2018

Due: Tuesday, September 25^{th}, 2018, 11:59pm.

You must do this assignment individually. You may consult with other students orally, but may not take notes or share code, and you must complete the final submission on your own.

Question 1:	30 points
Question 2:	20 points
Question 3:	50 points
	100 points total

Assignment

Question 1: Find Cases of Ambiguity (30 points)

Search for naturally occurring sources of linguistic ambiguity on the web. You must find five cases of linguistic ambiguity each falling into a different domain of language (e.g., phonological, lexical, syntactic orthographic, pragmatic, other).

Write a *short* paragraph for each sample that answers the following questions. a) What is the ambiguity, and what are the different possible interpretations? b) What in the passage specifically causes this ambiguity? c) What sort of knowledge is needed for a natural language understanding system to disambiguate the passage, whether the system is human or machine? Be more specific than simply saying "contextual knowledge." State the source of your sample (a URL and the name of the website will suffice.)

Question 2: FST for Spanish Verbal Conjugation (20 points)

Develop a FST to perform morphological analysis for the following Spanish verbal conjugation table, which shows verbs conjugated in the present tense:

Infinitive	**1 Sg**	**2 Sg**	**3 Sg**	**1 Pl**	**2 Pl**	**3 Pl**
Regular verbs						
andar (to walk)	ando	andas	anda	andamos	andáis	andan
contestar (to answer)	contesto	contestas	contesta	contestamos	contestáis	contestan
beber (to drink)	bebo	bebes	bebe	bebemos	bebéis	beben
correr (to run)	corro	corres	corre	corremos	corréis	corren
vivir (to live)	vivo	vives	vive	vivimos	vivís	viven
recibir (to receive)	recibo	recibes	recibe	recibimos	recibís	reciben
Irregular verbs						
ser (to be)	soy	eres	es	somos	sois	son
haber (to have)	he	has	ha	hemos	habéis	han

The morphological analyzer should provide the infinitive form of the verb, which we will take to be its lemma, along with its POS, person and number agreement. For example, feeding "*he#*" as input to the final FST should result in the output "*haber* +V +1 +Sg".

Your response should include three components:

- A schematic transducer in the style of Figure 3.13 in J&M (page 61)
- A lexicon table as in the top half of Figure 3.14 in J&M (page 62)
- A "fleshed-out" FST in the format of the bottom half of Figure 3.14 for the lexical items presented above

Question 3: Sentiment Analysis (50 points)

In this question, you will train a simple classifier that classifies a sentence into either a positive or negative sentiment. These sentences come from a movie review dataset constructed by the authors of this paper:

Bo Pang and Lillian Lee, Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales, *Proceedings of ACL 2005*.

The goal of this question is to give you experience in using existing tools for machine learning and natural language processing to solve a classification task. Before you attempt this question, you will need to install Python 2 or 3 on the machine you plan to work on, as well as the following Python packages and their dependencies:

- NLTK: `http://www.nltk.org/`
- NumPy: `http://www.numpy.org/`
- scikit-learn: `http://scikit-learn.org/stable/`

Download the corpus of text available in the attached file. This corpus is a collection of movie review sentences that are separated into positive and negative polarity. Your task is to train a sentence classifier to distinguish them.

Data storage and format

The raw text files are stored in *rt-polarity.neg* for the negative cases, and *rt-polarity.pos* for the positive cases.

Preprocessing and feature extraction

Preprocess the input documents to extract feature vector representations of them. Your features should be N-gram counts, for N $\leq$ 2. You may also use scikit-learn's feature extraction module. You should experiment with the complexity of the N-gram features (i.e., unigrams, or unigrams and bigrams), and whether to remove stop words. NLTK contains a list of stop words in English. Also, remove infrequently occurring words and bigrams as features. You may tune the threshold at which to remove infrequent words and bigrams. You can also experiment with the amount of smoothing/regularization in training the models to achieve better results. Read scikit-learn's documentation for more information on how to do this.

Setting up the experiments

Design and implement an experiment that correctly compares the model variants, so that you can draw reasonable conclusions about which model is the best for generalizing to similar unseen data. Compare the logistic regression, support vector machine (with a linear kernel), and Naive Bayes algorithms. Also, compare against the expected performance of a random baseline, which just guesses positive or negative with equal probability.

Report

Write a *short* report on your method and results, carefully document i) the problem setup, ii) your experimental procedure, iii) the range of parameter settings that you tried, and iv) the results and conclusions. It should be no more than one page long. Report on the performance in terms of accuracy, and speculate on the successes and failures of the models. Which machine learning classifier produced the best performance? For the overall best performing model, include a confusion matrix as a form of error analysis.

Submitting code

Submit your code in a file named "a1q3p2.py" if you are using Python 2, or "a1q3p3.py" if you are using Python 3.

What To Submit

Submit your solutions to Questions 1 to 2, as well as the report part of Question 3 as a single pdf on myCourses. For the programming part of Question 3, you should submit one zip file with your source code. All work should be submitted to myCourses under the Assignment 1 folder.