**Homework 1: Tweet Clustering**

You can work in pairs or individually on this homework. If working in pairs, only one of you should submit the code and result files. Also, name your partner in the README and in the text file containing your analysis.  The other person should submit a text file containing the information of both partners (full names and ucsc usernames).

In this homework, we will experiment with different features for doing clustering on tweets. We will use scikit-learn toolkit for feature generation and clustering. It is a powerful toolkit with a range of features and capabilities. We will use it to preprocess the data, create features, do feature selection and run algorithms.  
The first experiment creates a set of baseline classifiers for the dataset. The second experiment tests the effect of varying the numbers of features by feature selection using LSA. For clustering, we will use Kmeans.

**Dataset: We will run the same code on two different datasets and report the results for both the datasets.**The data we will be using comes from Twitter scrapes executed during the months leading up to the general election, mainly in the months of September and October. It contains hashtags related to Hillary Clinton and Donald Trump.

Dataset: There is a zip file in the location Files/Homework/Homework1/cs245-asg1-data.zip. It consists of two files, each containing 50,000 tweets.

* clinton-50k.csv
* trump-50k.csv

**Your tasks**:

**Task1: Pre-processing, tokenizing (3 points)**

The tweets, as given, are not in a form suitable for feature extraction for clustering. Therefore, the first step is to write a Python program named preprocess.py (tweet tokenize and clean), in accordance with the “General specifications” section below, that will take a tweet file in its provided form and convert it to a normalized form where:

1. All html tags and attributes (i.e., /<[^>]+>/) are removed.
2. All URLs (i.e., tokens beginning with http or www) are removed.
3. The first character in Twitter user names (@) and trailing hash tags (#) are removed.
   1. Remove only trailing hash tags (we don’t want to remove the ones embedded in the tweet text). Example:

**initial text:** "@HillaryClinton stands with #LGBT youth #SpiritDay #Hillary2016"

**processed text:** "HillaryClinton stands with #LGBT youth"

For preprocessing and tokenization, you may use Stanford Parser or standard Python libraries such as nltk and sklearn.

**Task 2:** **Feature Extraction (7 points)**

Feature extraction is basically the process of analyzing the preprocessed data in terms of variables that are indicative or discriminative of the clusters. The second step is to write a Python program feature\_generation.py that takes clean tweets from Task 1 and creates features that will be used to cluster tweets in Task 3.

**The Feature Sets:** The feature representations you should implement are defined as follows.

* Feature set 1: unigrams features (baseline).
* Feature set 2: unigrams using TFIDF features.
* Feature set 3: Perform text normalization:
  + Word stemming
  + Lowercase
  + Stop word removal
  + Strip any proper nouns (NNP) or plural proper nouns (NNPS) from the text
    - you will first need to run part-of-speech tagging on the tweet text. Since we are working with tweets, using [POS tagger for twitter from CMU](http://www.cs.cmu.edu/~ark/TweetNLP/).
  + Then, create unigrams using TFIDF features using these processed tweet texts.

**Feature Selection: (3 points)**

Feature selection is the process of selecting a subset of the terms occurring in the training set and using only this subset as features for clustering.

We will perform feature selection using [Latent Semantic Analysis.](http://scikit-learn.org/stable/modules/generated/sklearn.decomposition.TruncatedSVD.html) LSA transforms the original data in a different space so that two documents/words about the same concept are mapped close (so that they have higher cosine similarity). LSA achieves this by Singular Value Decomposition (SVD) of term-document matrix.

**Task3:** Perform feature selection using LSA on your feature set 3 giving feature set 4 and call it lsi.py.

**Task4: Clustering and Evaluation (7 points)**

**Clustering:**Write a python source file called clustering.py. Perform clustering using KMeans for all the four feature sets you obtained and write the results to a csv file. (Vary K between 5-10) and observe the clusters you get. Set K to a value you think gives better clusters by examining the output you get. You can do it by sorting the tweets by cluster label in your csv file and then looking at the semantic similarity of tweets grouped in the same cluster.

**Task 5:**

Have a Main.py that calls all the above tasks in sequence.

**Evaluation:**

Evaluating the performance of a clustering algorithm is not as trivial as counting the number of errors or the precision and recall of a supervised classification algorithm.  Measures such as Adjusted Rand index, V-measure require the knowledge of the ground truth classes, which is almost never available in practice. Since the ground truth labels are not known for this dataset, evaluation must be performed with a qualitative analysis for each of the four feature sets for both the datasets.

**Submission:**

1. Result files that contain the clustering results.  This result file should contain the tweet id and 4 additional columns, one column for each type of feature set. This added column should contain the cluster id indicating the cluster of each tweet. You should have two result files, one for each type of dataset.
2. A text file containing the qualitative analysis of the clusters you obtained. For each of the two datasets, {trump, clinton}, one or two paragraphs where you discuss your choice of K and your observations on the clusters you created. In your opinion, how effectively have the tweets been sorted into groups of related tweets? Can you identify any commonalities across the tweets of a given cluster?
3. The source code to finish this task (you should have the following four files at least: preprocess.py, feature\_generation.py, lsi.py, clustering.py)
4. A README file that briefly explains the main idea and implementation of your algorithm and also the instructions to run and test your code. If there is anything that the grader should know about your code, please include it in the README.

**We should be able to run your code for all the types of algorithms and generate all the result files.**

To get full marks, you must implement the algorithms in a modular way allowing for easy addition of new features with minimal modification.

**Please submit a zipped file containing all of the following files.**

* Source code: 4 files
* Qualitative Analysis
* 2 Result Files for each of the two datasets
* A readme

**Grading Rubric**

* 5/5: code is readable and runs, all scripts and output files are included, and explanation of results is meaningful
* 4/5: same as above, but does not easily run
* 3/5: code is provided but results analysis is superficial
* 2/5: missing components from code, still some effort shown
* 1/5: messy/incomplete code and output files, very limited analysis
* 0/5: no effort demonstrated

Top of Form