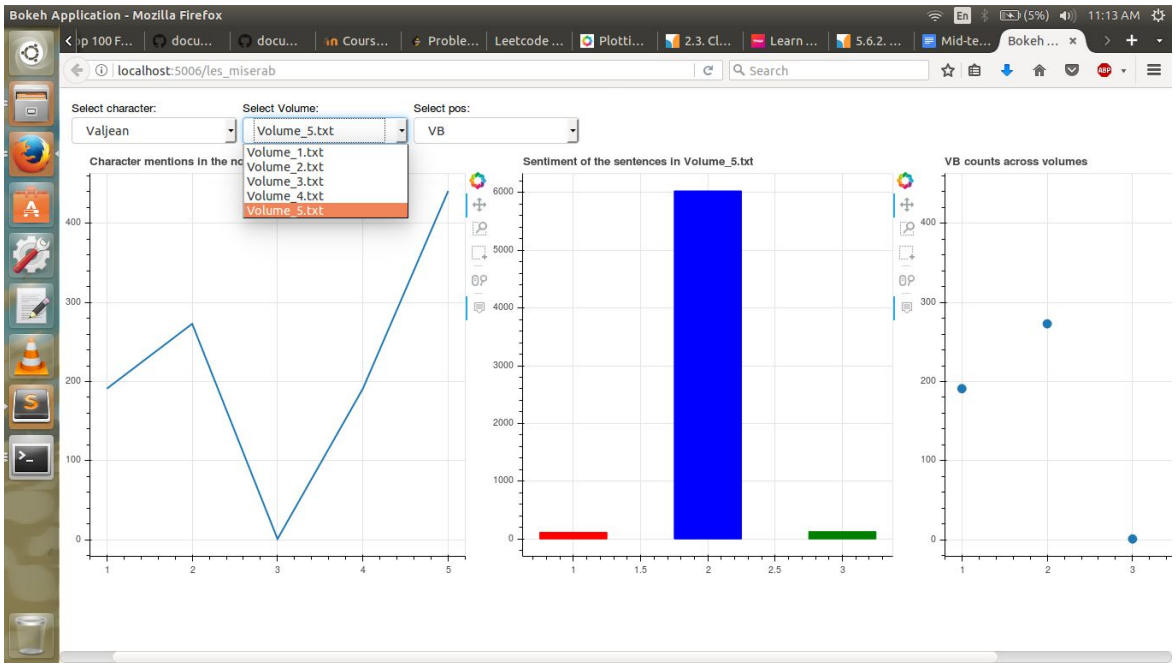
Mid-term Visual Analytics

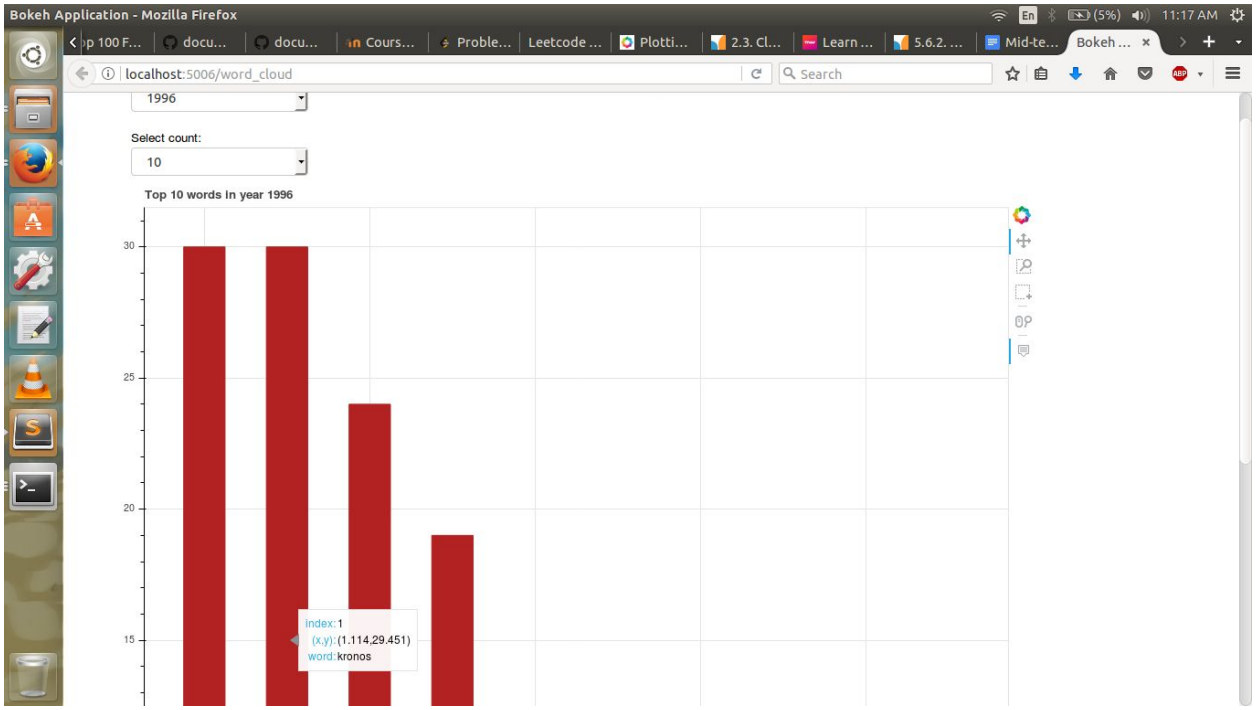
Sanjay Reddy Satti (31315112), Aditya Agrawal (31221087)

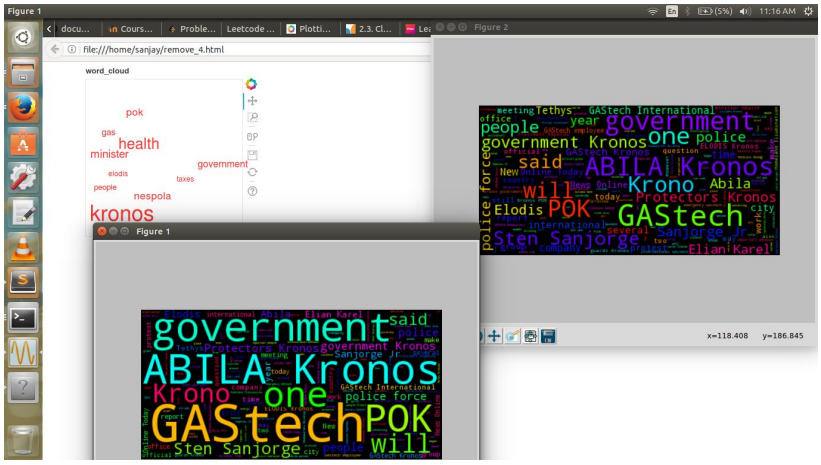
Seeing that the aim was "*exploratory visualization, to get familiar and insights into the data sets*" We wanted to try out many different datasets (from different domains), so that we become very comfortable with NLP tasks.
Unfortunately, not all of these are tested on Python-3 (It was giving me few errors on my laptop while using Python-3). For each dataset, clear instructions on how to run are present. Sorry for any inconvenience. Needs nltk (along with datasets) to be installed.

1) Dataset1: Les Misérables by Victor Hugo (Dataset is a "Novel")
    - We need to run bokeh server to get this running. Tested on Python-3. Run the program in the folder where the 5 volumes of the book are present. (~/Mid_Sem/les_miserab$ bokeh serve --show les_miserab.py)
    - The first visualization is the character mentions throughout the novel. The main characters were selected from Wikipedia. As expected Jean Valjean (the protagonist) has very high word count. In particular, the volume 5 (titled 'Valjean') obviously has many mentions of him.
    - The second visualization utilizes sentiment analysis (of nltk library) and reports the number of sentences which are positive, neutral and negative in each volume. As expected, neutral sentences are much more common that positive and negative ones. Although we didn't expect such a large skew.
    - The third visualization is a scatter plot displaying different parts of speech in sentences across the volumes.
    - All three are interactive visualizations

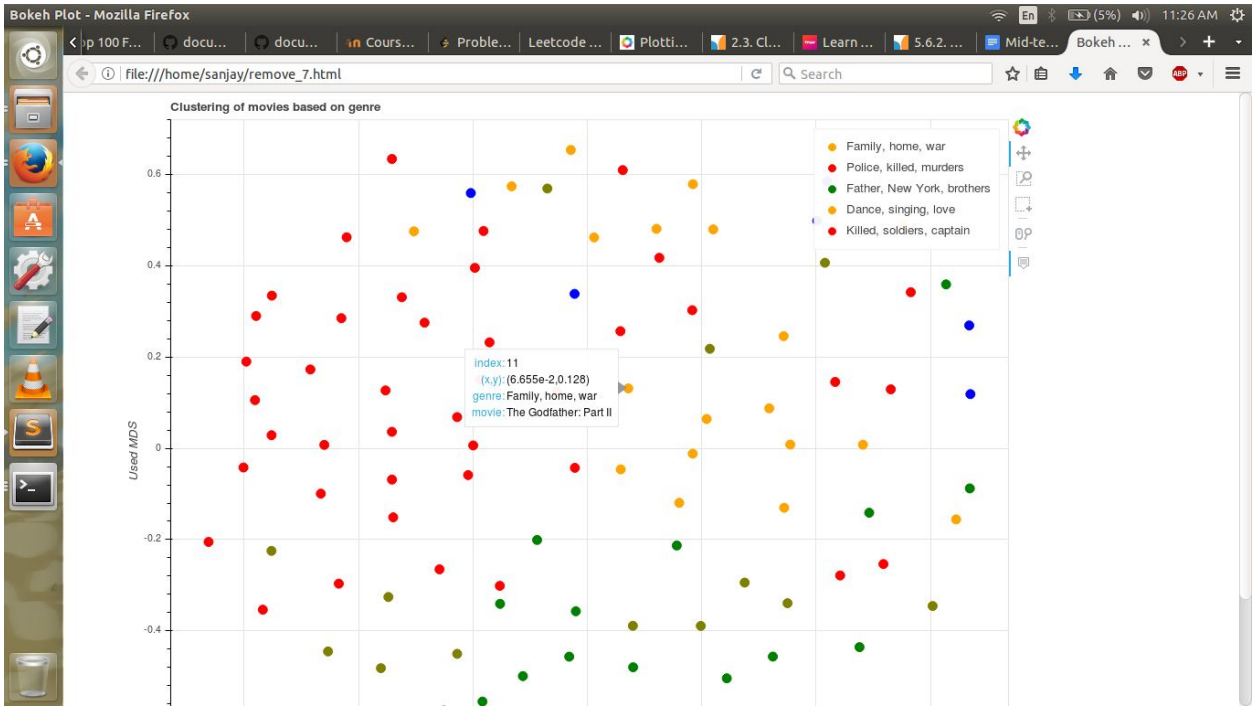


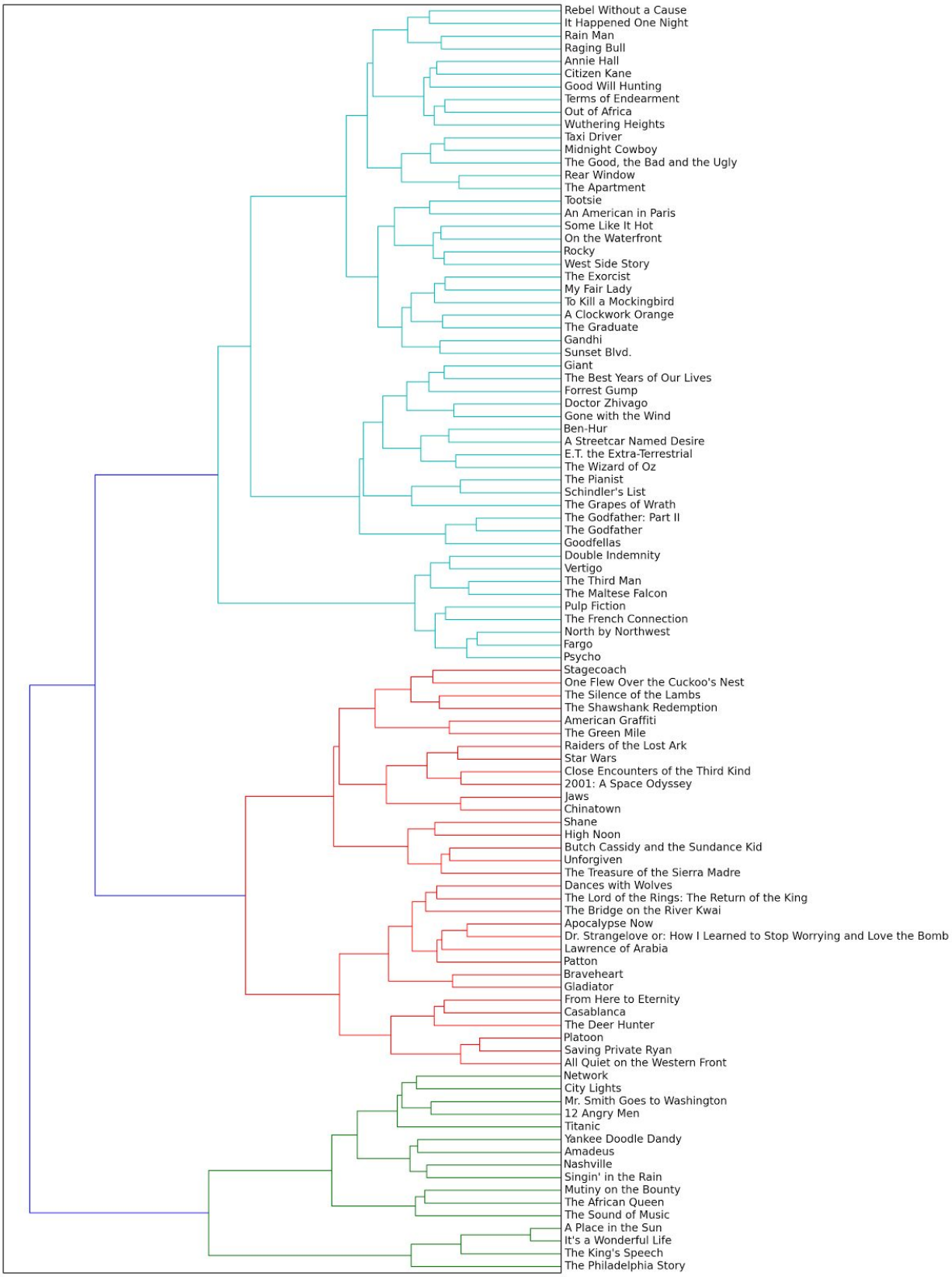
2) Dataset2: VAST Challenge 2014 MC-1 (Disappearance of GASTech employees) (Dataset are “emails”)
    - We did not use any NLP techniques on this dataset. We just thought this dataset would provide good interactive visualizations because it has a time component to it. It can obviously be replaced by a similar dataset and the results should be the same.
    - The first visualization is an interactive histogram chart, where in you can filter using the year and also the number of top words. (Inspiration being: Google NGrams). We need to run bokeh server to get this running. Tested on Python-3. Change the path to the articles folder in the .py file. (I put it in the folder so it should work)
      ($ echo ‘Check the path to articles folder accordingly in word_cloud.py & _2.py’
      $ ~/Mid_Sem/Word_Cloud$ bokeh serve --show word_cloud.py)
    - The second visualization is a word-cloud created using a python library and in bokeh (We specifically wanted to create one in bokeh. Although it’s not so good as one created by other libraries, we got one working). This one, we tested using Python-2. It requires no bokeh server, just run the command using Python-2. The rendered html and saved pictures have been included in the folder.
      (~/Mid_Sem/Word_Cloud$ python2 word_cloud_2.py)



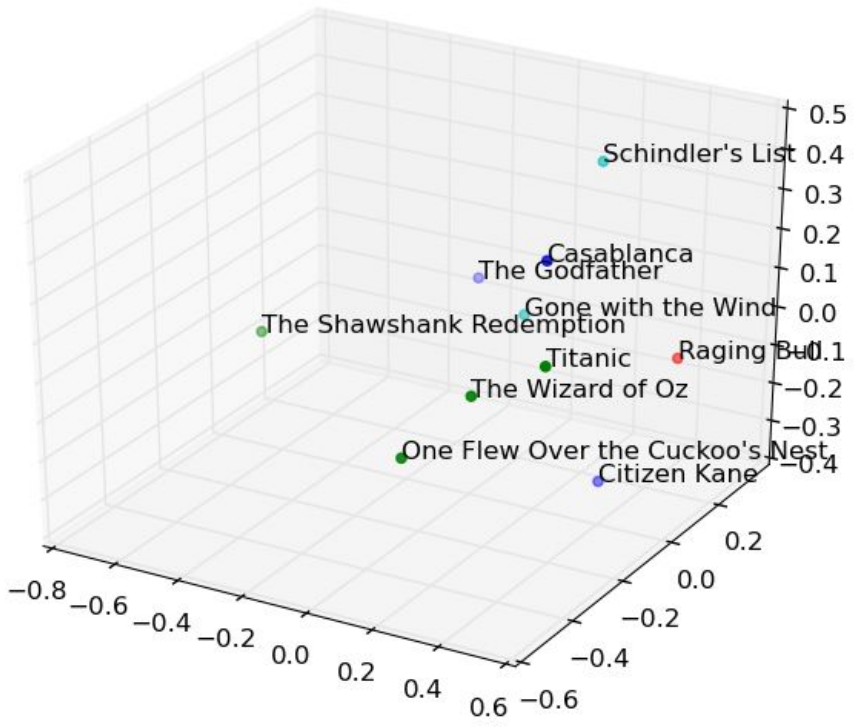


3) Dataset3: IMDB data (top 100 movies list) (Dataset are “synopses” written by users)
   - We use lot of NLP techniques (like removing stop words, stemming, parts-of-speech tagging) to analyze the synopsis of the movies and cluster them in 5 genres. Using tf-idf, the text was converted to vector space. By hovering over the points we can see which movie and which genre it belongs to. Tested on Python 2.7. We used MDS (Multidimensional Scaling) to bring down the number of dimensions
     ($ ~/Mid_Sem/IMDB$ python2 imdb_cluster.py)
   - For another clustering method, we used Dendrogram (& agglomerative clustering) and classified the movies into bins. Since we were having trouble integrating the dendrogram methods with bokeh, the plot has been generated in matplotlib. Run the program in the same folder as the files are present. Tested on Python-2.7.

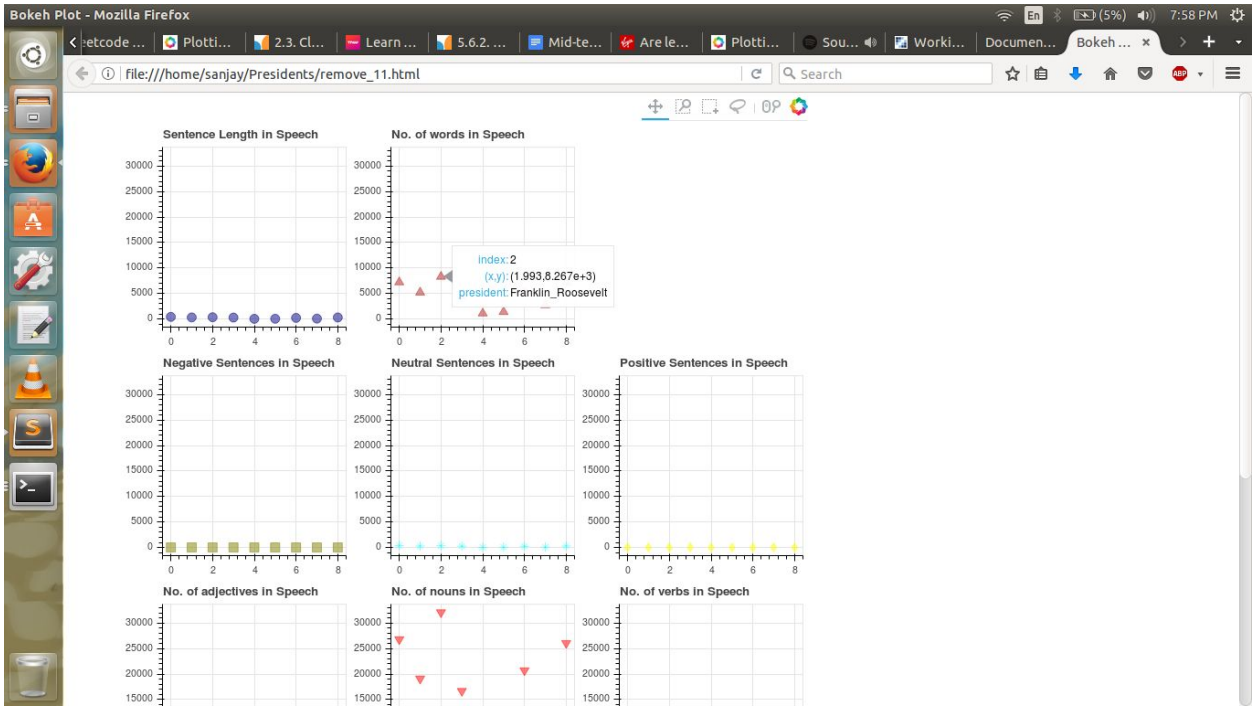

Bokeh Plot - Mozilla Firefox
file:///home/sanjay/remove_7.html
Search
Clustering of movies based on genre
Family, home, war
Police, killed, murders
Father, New York, brothers
Dance, singing, love
Killed, soldiers, captain
index: 11
(x,y): (6.655e-2,0.128)
genre: Family, home, war
movie: The Godfather: Part II
Used MDS
0.6
0.4
0.2
0
-0.2
-0.4


Rebel Without a Cause
It Happened One Night
Rain Man
Raging Bull
Annie Hall
Citizen Kane
Good Will Hunting
Terms of Endearment
Out of Africa
Wuthering Heights
Taxi Driver
Midnight Cowboy
The Good, the Bad and the Ugly
Rear Window
The Apartment
Tootsie
An American in Paris
Some Like It Hot
On the Waterfront
Rocky
West Side Story
The Exorcist
My Fair Lady
To Kill a Mockingbird
A Clockwork Orange
The Graduate
Gandhi
Sunset Blvd.
Giant
The Best Years of Our Lives
Forrest Gump
Doctor Zhivago
Gone with the Wind
Ben-Hur
A Streetcar Named Desire
E.T. the Extra-Terrestrial
The Wizard of Oz
The Pianist
Schindler's List
The Grapes of Wrath
The Godfather: Part II
The Godfather
Goodfellas
Double Indemnity
Vertigo
The Third Man
The Maltese Falcon
Pulp Fiction
The French Connection
North by Northwest
Fargo
Psycho
Stagecoach
One Flew Over the Cuckoo's Nest
The Silence of the Lambs
The Shawshank Redemption
American Graffiti
The Green Mile
Raiders of the Lost Ark
Star Wars
Close Encounters of the Third Kind
2001: A Space Odyssey
Jaws
Chinatown
Shane
High Noon
Butch Cassidy and the Sundance Kid
Unforgiven
The Treasure of the Sierra Madre
Dances with Wolves
The Lord of the Rings: The Return of the King
The Bridge on the River Kwai
Apocalypse Now
Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb
Lawrence of Arabia
Patton
Braveheart
Gladiator
From Here to Eternity
Casablanca
The Deer Hunter
Platoon
Saving Private Ryan
All Quiet on the Western Front
Network
City Lights
Mr. Smith Goes to Washington
12 Angry Men
Titanic
Yankee Doodle Dandy
Amadeus
Nashville
Singin' in the Rain
Mutiny on the Bounty
The African Queen
The Sound of Music
A Place in the Sun
It's a Wonderful Life
The King's Speech
The Philadelphia Story

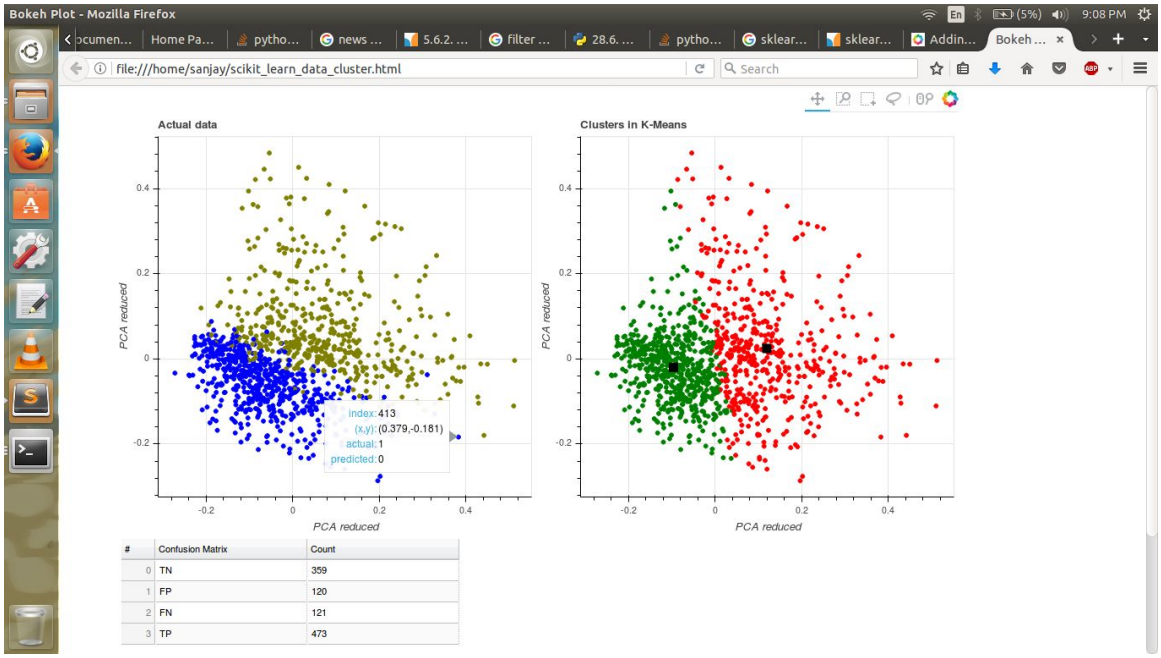
- In order to get a clustering in 3D space, we used ‘matplot’ to plot a graph (and to keep it visually simple, we restricted ourselves to only 10 movie titles)



4) Dataset4: State of the Union (President’s speeches)
   - Here we took 9 president’s speeches (at random) and made various graphs relating to number of words, sentences, types of words used etc. All of the graphs are linked. Tested on Python-3. Just run the program in the folder containing the speeches.
     ($~/Mid_Sem/Presidents$ python3 presidents.py)
   - The sentences used are very complex and hence the basic nltk sentiment analyzer is not able to accurately predict the sentiments of the sentences (Hence majority are low).
   - Earlier the speeches used to be very short (George Washington, John Adams) but as the time progresses, they got longer (There’s almost a 7 fold increase in the speech lengths of Washington and Obama)
   - Pronouns and adverbs are something not used frequently in the speeches but nouns and adjectives are used heavily. Another observation, verbs also have been steadily increasing as a function of time (One obvious explanation is the increase in speech length but one can also argue that the presidents of modern times have to do more work/action [hence more verbs]).



5) [Dataset5](): Scikit-Learn data (20 newsgroup data) (Dataset is “news articles”)
   - We wanted to cluster a more basic data and try PCA analysis on it. (Earlier we used MDS). So we chose scikit-learn dataset itself and clustered based on two classes. For plotting, we brought the dimensions back to 2 (using PCA) and plotted the points (along with the cluster centers). Uses tf-idf method again. The dataset should automatically be downloaded by the program (It might take 5 min to download the 31mb file). Runs on Python 3. Both the clustering plots are linked
     ($ ~/Mid_Sem/SKLearn_Cluster$ python3 tf_idf_cluster.py)



References:

1) http://neoformix.com/2013/NovelViews.html
2) http://brandonrose.org/top100
3) https://www.theguardian.com/world/interactive/2013/feb/12/state-of-the-union-reading-level
4) http://lotrproject.com/statistics/books/sentimentanalysis
5) https://github.com/amueller/word_cloud/blob/master/examples/simple.py
6) http://www.nltk.org/howto/sentiment.html
7) https://de.dariah.eu/tatom/working_with_text.html
8) http://jonathanzong.com/blog/2013/02/02/k-means-clustering-with-tfidf-weights
9) https://www.cs.cornell.edu/people/pabo/movie-review-data/review_polarity.tar.gz
10) https://appliedmachinelearning.wordpress.com/2017/02/12/sentiment-analysis-using-tf-idf-weighting-pythonscikit-learn/
11) https://stackoverflow.com/questions/28160335/plot-a-document-tfidf-2d-graph
12) Bokeh
13) Scikit-learn