**A Project Report**

**On**

**“Sentiment Analysis”**

**SUBMITTED BY**

Varun Shah

Chinmay Solanki

Zeel Surani

**Under The Guidance of**

Prof. Ankit P Vaishnav

B. Tech. CE Semester – VI



Department Of Computer Engineering

Faculty of Technology

Dharmsinh Desai University

**DHARMSINH DESAI UNIVERSITY**

**NADIAD-387001, GUJARAT**



**CERTIFICATE**

**This is to certify that the project carried out in the subject of System Design Practices entitled “Sentimental Analysis” and recorded in this report is a work of**

|  |  |  |
| --- | --- | --- |
| **1) Varun Shah** | **ROLL NO: CE-131** | **ID: 14CEUOG092** |
| **2) Chinmay Solanki** | **ROLL NO: CE-122** | **ID: 14CEUBG032** |
| **3) Zeel Surani** | **ROLL NO: CE-133** | **ID: 14CEUON101** |

**Of Department of Computer Engineering, semester VI. They were involved in Project development during academic year 2016 -2017.**

***Prof. Ankit P. Vaishnav*** ***Dr .C.K. Bhensdadia***

Asst. Prof, Head,

Department of Computer Engineering, Department of Computer Engineering,

Faculty of Technology, Faculty of Technology,

Dharmsinh Desai University, Nadiad Dharmsinh Desai University,Nadiad

# Abstract

Sentiment Analysis or opinion mining is the computational study of people’s opinions, sentiments, attitudes, and emotions expressed in written language. It is one of the most active research areas in natural language processing and text mining in recent years. It has a wide range of applications because opinions are central to almost all human activities and are key influencers of our behaviours.

The project can be used to analyse polarity of a given statement. This is of high value to institutions which want to analyse the customer feedback or conduct market research. When used in this way, the domain is limited and the targeted characteristics are known beforehand. This can be fed to the project to greatly increase the accuracy of the results provided.

This report aims to provide reader with conceptual understanding of the system, and also act as a working guide.

**Contents**

1. **Introduction…...............................................................................................................1**
2. **Software Requirement Specification.............................................................................2**
   1. **Introduction ...................................................................................................2**
   2. **Overall Description ........................................................................................3**
   3. **System Features.............................................................................................4**
   4. **Individual Review Classification......................................................................5**
   5. **Nonfunctional Requirements..........................................................................6**
3. **System Diagrams .........................................................................................................7**
   1. **Use-case Diagram ..........................................................................................7**
   2. **Sequence Diagrams .......................................................................................8**
4. **Implementation...........................................................................................................9**
5. **Testing………………………………………………………………………………………………………….………….10**
6. **Screenshots……………………………………………………………………………………………………………..11**
   1. **CLI Version...................................................................................................11**
   2. **GUI Version……............................................................................................12**
7. **Conclusion……………………………………………………………………………………………………………....14**
8. **Limitations and Future enhancements ......................................................................14**
   1. **Limitations………………………………………………………………………………………………...14**
   2. **Future enhancements………………………………………………………………………….....…14**
9. **Bibliography .............................................................................................................15**
10. **Introduction**

This project on sentimental analysis aims to determine the polarity of a given set of statements (positive, negative or neutral).

This is achieved by lexicographic analysis of words in the sentence. A list of words segregated into positive and negatives is available. The system then looks for these known words, and associates them with other enhancers, diminishers or inverters, thus extracting the sentiment of the set of statements.

Further, the system tries to associate the sentiment with an appropriate noun, thus providing a more comprehensive analysis of statements provided.

The project is provided in two flavors: CLI and GUI version. The former can be used to see how the program analyzes the set of statements provided, while the latter provides a better visualization in form of graphs. The core program is implemented in python, while the GUI uses Django framework with Jinja template used in front-end. Google charts API is used to create graphs.

Also, the project has two variants. First is a generalized code, which depicts usefulness in non-predetermined scenario. Second is a specific code, which shows how by simply limiting the input domain and setting the target to analyze, the accuracy and relevancy of results can be increased many folds.

# 2. Software Requirement Specification

# Introduction

## Purpose

The system concludes the polarity of review based on lexicographic sentiment analysis to decide whether it is positive, negative or neutral.

## Product Scope

The system can identify polarity of any generalized opinion. Limiting the scope of opinion, however, improves accuracy (score) and relevancy (nouns) of the output produced.

System cannot identify language nuances (sarcasm, dual meanings)

System identifies nouns correctly provided there is at most one noun per sentence.

**2.1.3 Document Conventions**

IEEE format for software requirement specification*.*

# Overall Description

## Product Perspective

It is a new Self Contained software Product.

## Product Functions

The product is provided in two flavors. The GUI version presents the output via easy to understand graphs, while CLI version provides an option to see the in-depth processing details.

## User Classes and Characteristics

The Following class of user will be using the system:  
**Client** : client will be able to give review input or a review dump in a file and get analyzed output.

## Operating Environment

The system is built using Python and Django framework its requirements are as follows:

Software & Hardware Requirement

OS: windows 7, 8, 8.1, 10, Linux, MAC

RAM: 1 GB

# System Features

## Review Dump Classification

### Description and Priority

User can upload a review file (dump), and analyzed output graphs with a summed polarity will be provided.

High priority.

### Stimulus/Response Sequences

Stimulus: User uploads a file via “upload file” option.

Response: System takes the mentioned file, and calls the python code associated to it. The output of python code is processed appropriately to show final results: graphs in (GUI) version, and dictionary + processing (CLI)

### Functional Requirements

REQ-1: Upload File

Input: File provided by user

Processing: Fetch the file, call the python code to analyze the file.

Output: “File successfully uploaded”

REQ-2: Analysis of the review dump

Input: File provided by user

Processing: Analyze each review independently, provide summed and

individual analyzed outputs.

Output: Final dictionary containing summed and individual analysis.

CLI: Entire processing is displayed if debug is set.

REQ-3: (GUI only) Displaying the graphs

Input: Dictionary provided by code.

Processing: Create appropriate graphs (via Google chart API)

Output: Two graphs are also provided to visualize the result: one sums

up generalized review pattern, other shows the polarity about

concerned features (noun-graph)

# Individual Review Classification

### Description and Priority

User can ask the system to classify individual review. This provides for a much larger classification range, at the cost of relevancy.

### Stimulus/Response Sequences

Stimulus: User enters a single review.

Response: System sends the review to python code , and displays output received.

### Functional Requirements

REQ-1: Receive review

Input: Review provided by user

Processing: Fetch the review, call the python code to analyze it.

Output: NA

REQ-2: Analysis of the review dump

Input: Review provided by user

Processing: Analyze each the review.

Output: Final dictionary containing overall score and noun-based

score.

# Other Nonfunctional Requirements

## 2.5.1 Performance Requirements

System should be able to analyze up to 500 reviews, each with an average of 5 sentences of appreciable length in a reasonable time frame (<2 minutes)

## 2.5.2 Safety Requirements

System must be given appropriately formatted text file for it to work properly.

The file must contain only reviews in English, and two reviews must be separated by only a single line feed character (\n).

None of the reviews may contain any linefeeds as a part of their content.

Input review file must be reasonable in size, where the size depends on processing power and memory available on the host server running the main python code: primarily since python moves entire file to memory for reading.

## Software Quality Attributes

System contains two modules: CLI and GUI.

The GUI module can be used for extending the system, or using the system as a module in other system. It can be used as an API, and the output can be modified as per requirement.

System is guaranteed not to crash as long as safety requirements are met, with time-of-processing being dependent on the input size.

System processing can be seen in detail via CLI version, in case user wishes to find error or understand the processing.

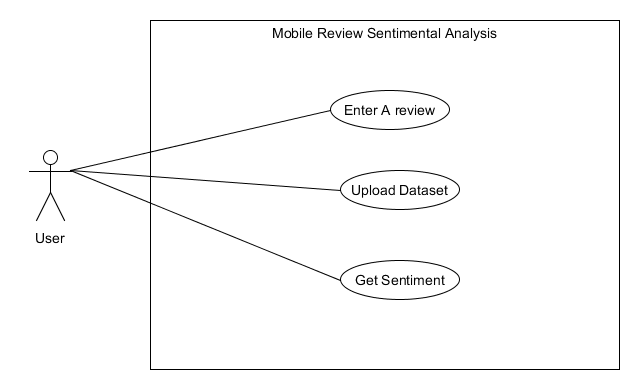
System’s main code can be deployed on any system which can process python and Django, and the output client can be hosted on any system capable of processing and rendering jinja template, along with javascript and html-css support.

System can be used anywhere where analysis of sentiments is required.

Accuracy and relevancy of results vary inversely with the size of domain used for analysis.

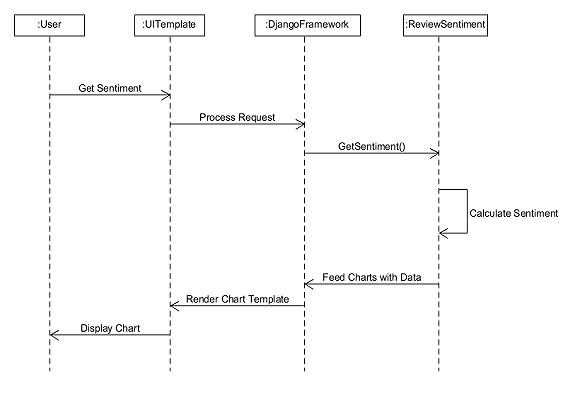
**3) Diagrams**

**3.1 Use-case Diagrams**

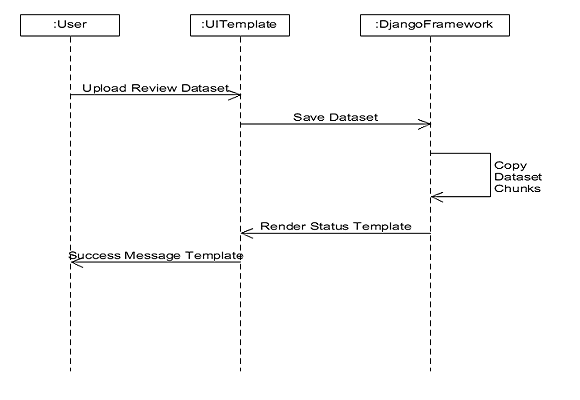
****

**3.2 Sequence Diagrams**

**Get Sentiment**

****

**Upload Dataset**

****

# Implementation Details

The program consists of two flavors (CLI and GUI), each flavor having two functionalities(single review and review dump analysis) , each functionality having two files: One contains the ProcessReview class, and other contains main method.

However, the core logic remains same across all the classes. Main method varies slightly to accommodate slight nuances between each type.

The Main method (\_Main files) is responsible for calling process review class, providing it the input that it expects (A single review) and format the output according to what is required.

The ProcessReview class is where all the analysis happens. All the important function prototypes along with a brief description of each is provided below:

1. \_\_init\_\_(self) : Void

--Constructor--

Responsible to bring in the lists of positive and negative words into python’s running memory. Also brings in noun list in \_List programs. Segregates the contents of files to make them iterable.

1. getSentiment(self,review) : Dictionary

The function called by the main method. Input is a single review (may consist of several statements.). Separates sentences from the review, and operates on each sentence. It calls all other functions needed to analyze the review, stores answer in a dictionary, and returns this dictionary.

1. preprocess(self,sentence) : Void

Alters case of the sentence, and performs optimization by removing unnecessary words.

1. scoreSent(self,sent) : Void

Scores each sentence. It tokenizes the sentence, tags them and uses the tags to further optimize. Also factors in words that enhance the meaning of a word, and inverters.

Updates the score on each sentence iteration, thus providing final score.

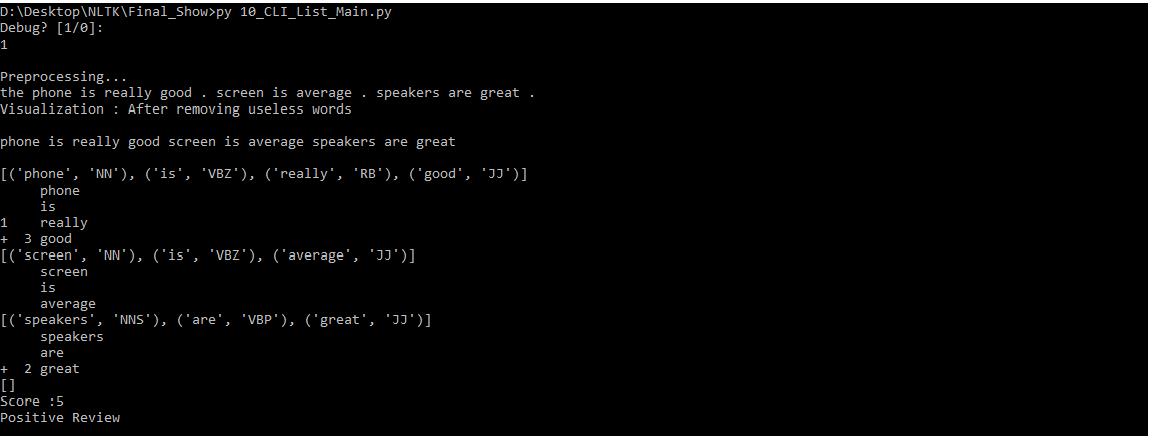
# Testing

# The program has been tested on several types of input datasets, to determine how well it is able to guess the sentiment.

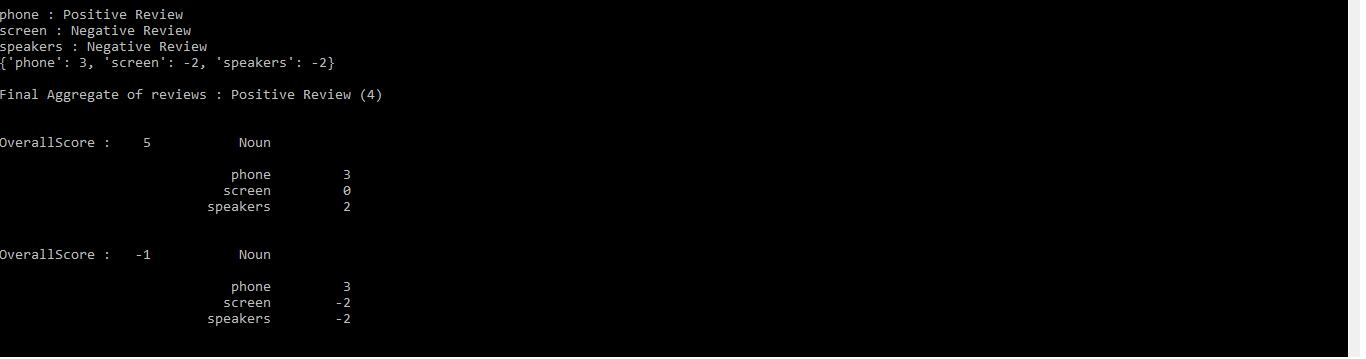
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr No. | Test Case | Output | Expected | Result |
| 1 | The movie was great. I really liked the movie. | Positive | Positive | Pass |
| 2 | The phone has great screen, but I did not like its speakers. | Neutral | Neutral | Pass |
| 3 | I did not like his acting. | Negative | Negative | Pass |
| 4 | Call quality is good, but the quality of features it provides as per price point is very bad. | Negative | Negative | Pass |
| 5 | The phone is very fast. | Neutral | Positive | Fail |
| 6 | He has such good oratory skills, that everyone within the room is asleep within 5 minutes. | Positive | Negative | Fail |

# Screenshots

**6.1** CLI version:

****

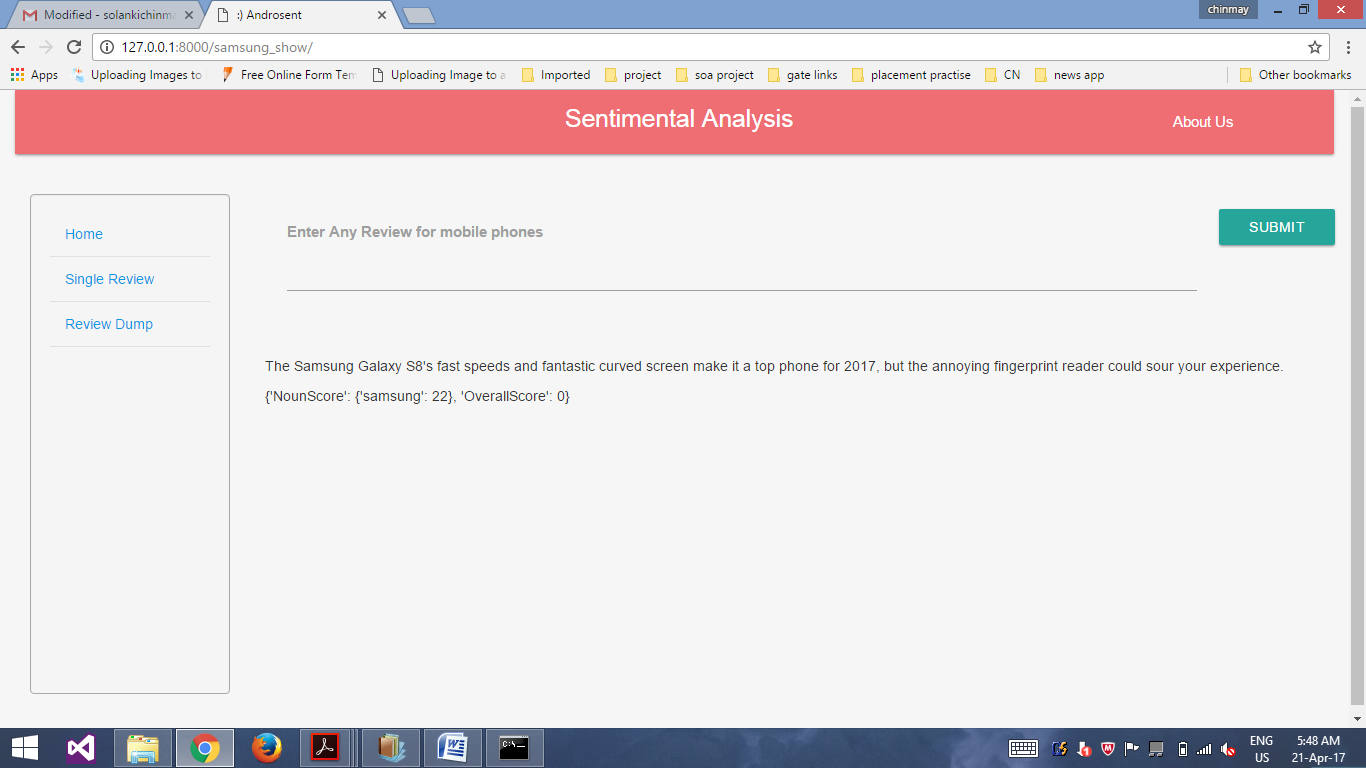
Analysis when Debug is set. Shows preprocessing, tokenization, tagging, scoring and result for each sentence.

****

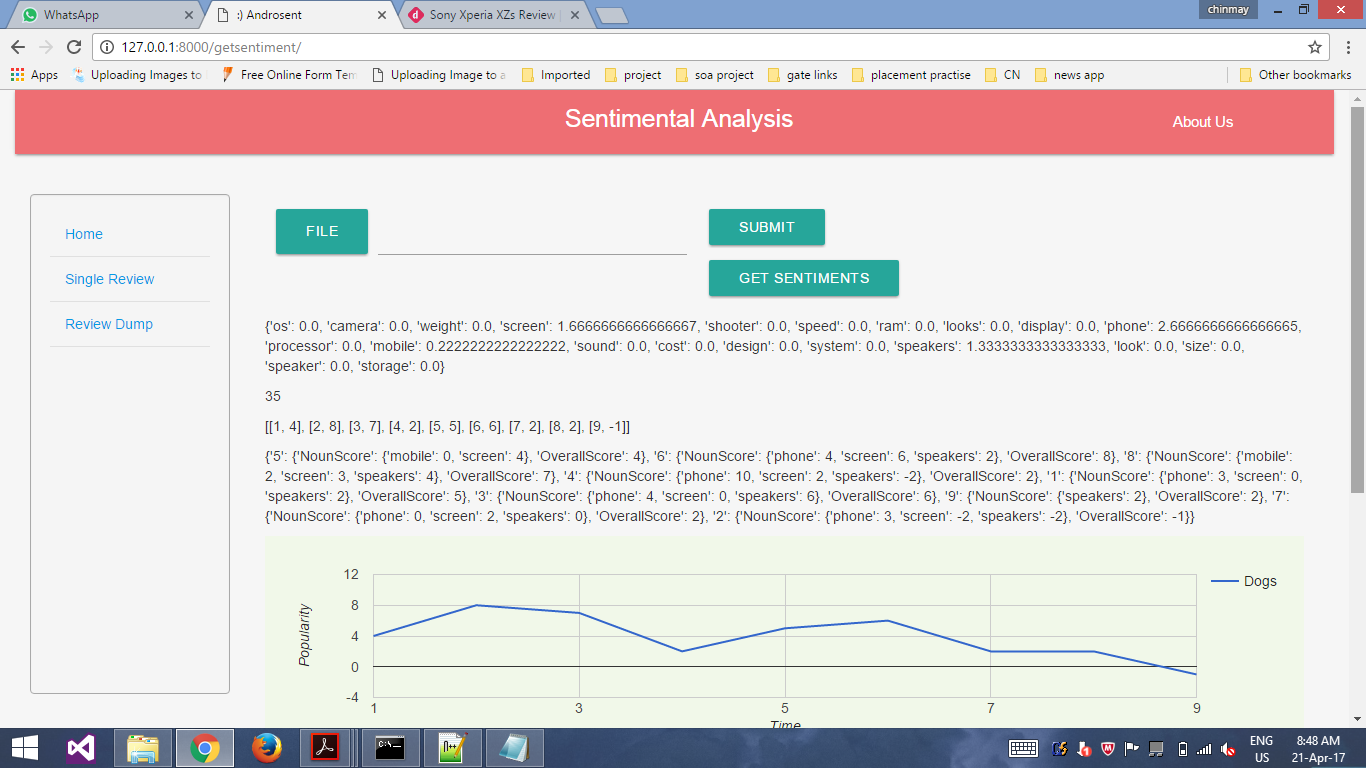
Final Output

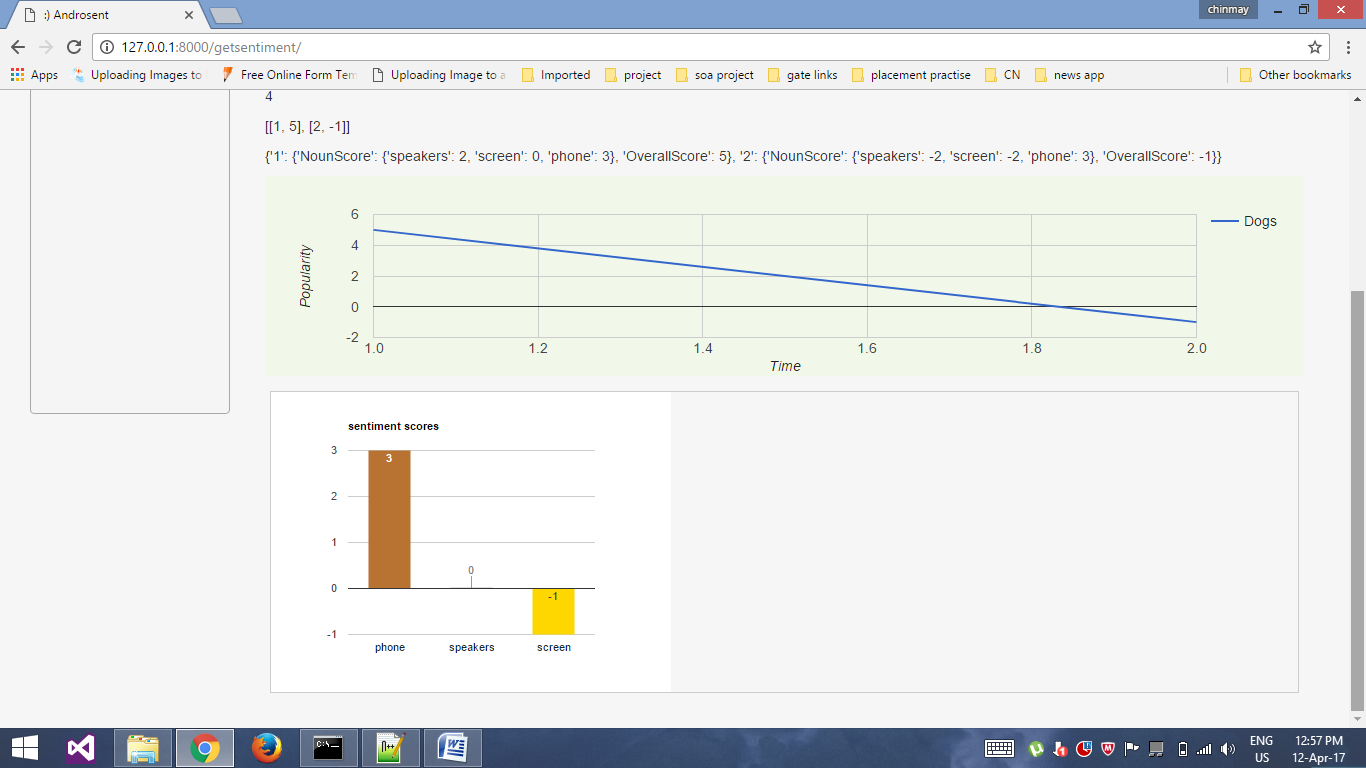
**6.2 GUI version**

Sentiment for single review

****

Sentiment for review dataset





# Conclusion

The system successfully classifies review sentiments. It also provides proper graphs as output.

# Limitations and Future Extension

## 8.1 Limitation:

The System cannot identify regional or context-specific words (“fast” can be positive or negative based on context in which it is used), higher level abstractions of opinions (“The screenplay was on a whole another level”), or sarcasms.

System identifies nouns properly iff there is a single noun per sentence. Otherwise, the first noun is considered as the subject for the sentence. Also, conjunctions are not identified.

## 8.2 Future Extension:

The system’s accuracy and relevancy can be improved if more refined noun-lists are used, or if context-specific rules are added.

To identify language nuances, sarcasms or abstract opinions, Machine Learning based approaches are required. It involves training the system on a pre-classified dataset and then system guesses a review’s polarity based on its learning.

# Bibliography

## Websites:

## [www.nltk.org](http://www.nltk.org)

## [www.pythonprogramming.net](http://www.pythonprogramming.net)

## [www.google-developers.appspot.com](http://www.google-developers.appspot.com)

## nlp.stanford.edu

## docs.djangoproject.com

## [www.learnpython.org](http://www.learnpython.org)