EECS 4312

SOFTWARE REQUIREMENTS

INSTRUCTOR: Maleknaz Nayebi

RELEASE DATE: March 21st, 2022

CONTEXT

The course project defines 20% of your total grade in EECS4312. This project is designed in two different phases each having a separate deadline:

PHASE 1: Applying design principles, Deadline: April 8th, 2021.

PHASE 2: Reviewing the applied design of one of your classmates, Deadline: April 15th, 2021.

In this project we work on real world projects and based on the concepts in Experimental Education and Hands-on Learning to enable you to utilize your critical thinking and reflect on what you have learned. This is based on our strong belief that "one size does not fit all".

Please note that, Students without a valid submission in PHASE 1 would not be eligible to submit PHASE 2.

OBJECTIVE OF THIS DOCUMENT

Software requirements is often recorded in a natural language. Understanding software requirements and retrieving information is a key to many phases of software lifecycle. This document is the project description for the students attending EECS 4312 in Winter 2022.

QUESTIONS

For any questions, please attend the labs and lectures.

If emailing questions, please anticipate at least one business day needed to receive a reply.

ABBREVIATIONS & DEFINITIONS

The terms in this document might be unfamiliar for you as a beginner, in case needed please refer to GitHub Glossary and the Machine Learning Glossary.

In this document we consider the “ASSIGNED APP” as the app that have been assigned to you for Assignment #2 and Assignment #3 from F-Droid and by the professor.

EECS 4312

SOFTWARE REQUIREMENTS

INSTRUCTOR: Maleknaz Nayebi

RELEASE DATE: March 21st, 2022

AMENDMENTS

So far so good.

POLICIES

Your (submitted or un-submitted) solution to this assignment (which is not revealed to the public) remains the property of the EECS department. Do not distribute or share your code in any public media (e.g., a non-private GitHub repository) in any way, shape, or form before you get permission from your instructors.

- You are required to work on your own for this lab. No group partners are allowed.
- When you submit your solution, you claim that it is solely your work. Therefore, it is considered a violation of academic integrity if you copy or share any parts of your code or documentation.
- When assessing your submission, the instructor and TA may examine your doc/code, and suspicious submissions will be reported to the department/faculty if necessary. We do not tolerate academic dishonesty, so please obey this policy strictly.
- Emailing your solutions to the instruction or TAs will not be acceptable.
- For any extension request please refer to the course syllabus.

BACKGROUND

- Lab of week 6 and Lab of Week 11,
- Lecture Week 1 to Week 11.

RUBRIC

ID	Item	Percentage
1	TASK 1	5%
2	TASK 2	3%
3	TASK 3	35%
4	TASK 4	40%
5	TASK 5	10%
6	Complete submission of all the files and reporting	7%

The assignment includes three <u>BONUS TASKS</u> with overall of <u>16 Points</u>.

EECS 4312

SOFTWARE REQUIREMENTS

INSTRUCTOR: Maleknaz Nayebi

RELEASE DATE: March 21st, 2022

DUE DATE

As for the peer review process, this project has two due dates:

PHASE 1: April 8th, 2022, to submit reports via easy chair.

PHASE 2: April 15th, 2022.

Make sure to submit ALL the files requested in the DELIVERABLES section.

DELIVERABLES

You need to submit a total of **eight Files**:

1- A CSV file of all the app descriptions named “MyAppDescriptions.csv”.
2- A CSV file of all the manually extracted features from your app description from Assignment 3 named “MyAppFeatures.csv”.
3- A CSV file for each ASSIGNED APP (five files in total) including the scrapped reviews for the assigned app and the competitor and similar apps you identified is Assignment 2.
4- A Jupyter Notebook with all the requested functionalities.

SUBMISSION

Please submit all the eight files in designated folders in eclass. All the discussion and interpretations should be written in your notebook and **no PDF files** can be submitted.

THE PROJECT

We are going to review

TASK 1: GATHERING REVIEWS

In Assignment 2, you have identified similar and competitor apps for each of the five apps assigned to you (we call these five apps the ASSIGNED APPS). For each of the ASSIGNED APPS:

I. Create a CSV file with these header *<Package name, Reviewer name, Review, Rating>*.
II. Crawl/Scrap reviews from app store and pars them to extract Reviewer name, Review’s text, and the rating.

III. In your CSV file, each row represents one review for one app.

↳ As the result you should have five CSV files each named in the format of "*[Assigned App's package name]*.csv" and contains reviews of the app's you were assigned and the competitors and similar apps you identified in your Assignment 2.

NOTES:

1- There is a limit in the number of reviews you can get from the app store. If your app has less than 1,000 (ish) reviews that should be ok. If the app has more, gather as much possible and add a note to your assignment, stating this limitation.

2- You may use one of the existing web scrappers for the app stores (for example 1, 2, 3, 4) or write your own crawler for gathering the reviews.

3- The code for this scrapper should not be submitted. You only submit the data as part of your submission package.

4- You cannot change the set of apps (competitor or similar) comparing to your Assignment #2.

TASK 2: PREPROCESS YOUR TEXT

Before entering any machine learning task, we need to make the text readable for the machine. Pre-process each review text using the below steps you have already implemented in Assignment #3.

I. Remove punctuations,
II. Remove special characters and emojis,
III. Turn numbers into text,
IV. Remove extra white spaces,
V. Turn all words into lowercase,
VI. Remove stop words,
VII. Lemmatize the reviews.
VIII. Output 15 sample pre-processed reviews.

↳ As the result you should have an intermediate set of preprocessed reviews to be used in all the following tasks. You also should have the (i) pre-processing code snippet, and (ii) a sample of 15 Pre-processed reviews outputted in your Notebook.

BONUS of TASK 2 (5 Points): NLTK has a list of stop words which can be customized or replaced. Does the choice of stop words has any impact on the analysis of the reviews? What is the risk of using a non-customized list of stop words in analyzing mobile app reviews for software requirements elicitation?

NOTE:

1- The only difference in the pre-processing step in this task and what you have done in Assignment #3 is that we only lemmatize the text and one does not need to apply Stemming.

TASK 3: SENTIMENT ANALYSIS

We are going to use and compare two different tools based to calculate app review sentiments.

I. Use Textblob for calculating Reviews Sentiment;

II. The output should be a table with headers in the format of <*App's package name, Review, Positive, Negative, Neutral*>.

III. Use Vader for calculating Reviews Sentiment;

IV. The output should be a table with headers in the format of <*App's package name, Review, Polarity, Positive, Negative, Neutral*>.

V. How does the sentiments retrieved by Textblob and Vader compare with each other? How do you interpret the similarity/difference? Which one is the best option for review analysis of your apps? Why?

↳ As the result you should have the (i) two code snippets for sentiment analysis and (ii) two output tables in your Jupyter Notebook followed by (iii) any type of discussion or numerical comparison for part V.

NOTE:

1- If too many reviews make your Notebook heavy, you can restrict the sample output table into 250 reviews (rows).

TASK 4: TOPIC MODELING Using LDA

It is important to understand user feedback when developing a software iteratively and plan for changes accordingly. One way to condense and synthesize user reviews in a large-scale dataset is topic modeling which is an unsupervised learning model. It is known that users talk about app features/functionalities in their feedback, so we are going to further connect these reviews to the features you extracted from your apps.

I. Among your five ASSIGNED APPs, choose the one with the highest number of reviews.

II. From its reviews, extract minimum of 10 LDA topics [1, 2] each being described by at least 7 Words.

III. How does these review topics (i.e. summarized user feedback) relate to the software features (functionalities) you extracted from app description in Assignment 1 (manual) and/or Assignment 3 (automated bi-grams/tri-grams)?

IV. For this chosen ASSIGNED APP, extract topics of reviews (minimum of 5 Topics each described by 7 Words) for each of the competitor and similar apps (separately).

V. How does the topics compare to the topics extracted from reviews of your ASSIGNED APP?

↳ As the result you should have (i) topic modeling code snippet, (ii) a table of topics for the reviews of your ASSIGNED APP, (iii) any discussion or snippet for connecting the extracted topics to the app features, (iv) code snippet and (v) a table of topics for the reviews of competitor and similar apps, (vi) code and/or tables of comparison between topics of your chosen ASSIGNED APP and its competitor and similar apps.

BONUS for TASK 4 (8 Points): FINDING THE DOMINANT TOPIC FOR EACH REVIEW

We would need to trace each review to a topic for better understanding the details of the user feedback.

I. Retrieve the topic for each review of your ASSIGNED APP. This should be done by finding the topic that has the highest percentage contribution in that Review.

II. Provide a list of 10 Reviews that you believe developers should take care of with high priority. Discuss your logic for this selection.

III. Add the code and the discussion under "BONUS for TASK 4" in your Notebook.

NOTES:

1. If you have apps with equal (high) number of reviews, choose the one you wish.
2. The topic training can be non-deterministic and the final set might change by running or rerunning the model. Make sure to save your output in the notebook.
3. For comparisons, any appropriate comparison is acceptable (even manual analysis). One might also consider calculating the Cosine Similarity [1, 2].

TASK 5: RECOMMENDATION

The apps that have been assigned to you are open source and their code are hosted on GitHub.

I. Look into the F-Droid page, GitHub Repository, and if available the issue tracking system, Wikis, and website of your chosen ASSIGNED APP (from TASK 4).

II. Recommend two new features or enhancements (could be bug reports) for the app based on your above analysis that have not been done in the project.

↳ As the result you should have two suggestions for app improvement followed by the discussion on how and why you came up with these recommendations.

<u>BONUS of TASK 5 (3 Points):</u> Open a new issue in the issue tracking system and suggest the two recommend changes to the developers. Describe your logic and the data you analyzed. List two URLS to the reported suggestions in your Notebook under the section titled "Bonus of Task 5".

NOTES:

1- To ensure that the recommendation you are having are new (have not been done in past or not being introduced by anyone else), you might search topics and sub topics or the keywords within the project. One way is to search commit messages [1, 2, 3], release notes, pull requests, etc.

THE PEER REVIEW

The peer review will be done in eclass. Detailed process will be announced later.