1) An overview of the function of the code (i.e., what it does and what it can be used for).

We created a chrome extension that analyzes the sentiment of comments on YouTube videos. A user can navigate to a Youtuber's video page and use the extension to color-annotate the videos based on the positivity/negativity of the comments on each video. For example, there will be a green box highlighting videos that are liked by video commenters and a red box around videos that are disliked by video commenters. The lightness of the green/red varies depending on how positive or negative the video comments are.

2) Documentation of how the software is implemented with sufficient detail so that others can have a basic understanding of your code for future extension or any further improvement.

We have two main components of code. The first component relates to the chrome extension setup. The chrome extension portion can be found in the chrome extension folder of the project Github. This folder contains all the relevant code to set up our Youtube Sentiment Analyzer extension. The contents of this folder are largely inspired by Chrome's extension getting started guide: https://developer.chrome.com/docs/extensions/mv3/getstarted/. This webpage can be consulted to add our extension to your own Google Chrome. The most relevant file to our project is in popup.js, where we have a button listener that executes sentiment analysis when clicked. The code first extracts the video urls from the Youtube page's HTML. We split these urls into reasonably sized batches and send them to our sentiment analysis server using an HTTP POST request. The server will respond with a list of percentages representing the positivity/negativity of the video's comments. We use these percentages to color the respective video. This process then repeats for a new batch until all the videos are colored.

The sentiment analyzer code sits inside a basic server created using python's BaseHTTPRequestHandler. All this code can be found within comments_scraping.py. This server was largely inspired by the following: https://pythonbasics.org/webserver/. Our code starts up a local server capable of receiving POST requests. Within this post request, the server will obtain the list of video urls to analyze. It will run our sentiment analyzer, receive a list of positivity/negativity percentages, and send this data to our chrome extension that made the request.

The last part related to the actual scraping, cleaning, and sentiment analysis of the youtube videos and comments.

In addition to all this, we also optimized our code in both the server and comment scraping end. We made post requests in batches on the server side and also implemented threading and parallelization with the sentiment analysis so that each video is being processed at the same time. This change decreased our runtime significantly, as processing 5 videos without threading originally took around 90 seconds, while using threading resulted in around 60 seconds for 5 videos.

3) Documentation of the usage of the software including either documentation of usages of APIs or detailed instructions on how to install and run a software, whichever is applicable

Please see the README.

4) Brief description of contribution of each team member in case of a multi-person team.

We split into two teams. Christina and Cindia worked on scraping comments given a YouTube link, cleaning those comments, finding the overall sentiment of the comments, and optimizing processing for a video. Teresa and Amy worked on creating the Chrome Extension that can scrape all the video urls given a Youtube page, connecting to the sentiment analyzer backend via local python server, and using the results from the analyzer to display the color annotations on the Youtube page.