AMAZON REVIEW SENTIMENT ANALYSIS

A PROJECT REPORT

Submitted by

Harshita Rani (20BCS9338)

Diksha Bhardwaz (20BCS9382)

Harsh Tandiya (20BCS9693)

Rounak Kumar (20BCS9649)

Mohammad Aman (20BCS2168)

In partial fulfilment for the award of the degree of

BACHELOR IN ENGINEERING

IN

COMPUTER SCIENCE & ENGINEERING

CHANDIGARH UNIVERSITY

NOVEMBER 2022

BONAFIDE CERTIFICATE

Certified that this project **"AMAZON REVIEW SENTIMENT ANALYSIS"** is the bonafide work of **"Harshita Rani, Diksha Bhardwaz, Harsh Tandiya, Rounak Kumar & Mohammad Aman"**, who carried out the project under our supervision.

SIGNATURE

Dr. Sandeep Singh Kang
HEAD OF THE DEPARTMENT

SIGNATURE

Er. Tanu Dhiman (E8141)
SUPERVISOR

Submitted for the project viva-voce examination held on _____

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We have put a lot of efforts for the completion of this project. We thank everyone that contributed to the project's eventual completion. Every bit of help and guidance had a big impact on the project and we are grateful for each one of them.

We are incredibly grateful for our supervisors Er. Tanu Dhiman & Er. Amritpal Kaur for their guidance and constant supervision and for providing necessary information about the project and their constant support towards the completion of the project.

Thank you.

TABLE OF CONTENTS

ABSTRACT

In the modern world of e-commerce, millions of products are being sold every minute of the day. These sales generate vast amount of user reviews data which are waiting to be made use of. The user review data can be made useful by finding out what they are saying about the product and what the customer sentiment is regarding the product. This *analysis of sentiments* of customer review data can be done through data analysis and different machine learning models which are specified for the task of sentiment analysis. The vast review data also requires to be extracted and stored in a format which will enable different analysis operations to be performed on it. This project deals with the analysis and the data extraction part of the sentiment analysis. The custom-made data scraper for this project does the task of scraping all the review data for any specific required product and export that data into a .csv or .xlsx file. This excel file is later put in a data frame, and different sentiment analysis operations can be performed on it. The model used for sentiment analysis can be correlated as an upgraded version of the BERT model (roBERTa Model). The model gives points to each review and classifies it as either positive and negative. All the data obtained by these operations are then visualized and presented back to the user.

CHAPTER 1

INTRODUCTION

In the recent years, there has been a significant increase in the shift in commercial sector shifting to a digital platform, giving rise to e-commerce. The biggest of the e-commerce platform is Amazon.com, with its millions of products. This trend of shift to e-commerce is accelerating, and more and more individuals choose internet shopping for reasons such as ease and cost savings. Customers rely on the product reviews to determine whether the product they are checking out is worth buying or not. Product reviews are a big and essential part of any e-commerce website, because a customer will trust another customer over their experience of using the product over any company advertisement.

Amazon website produces a large amount of review data and these review data can be made of use by the companies. The companies can use the reviews to investigate how their product is being perceived by their targeted user base and determine how the product is performing on the market. One of the primary methods by which the company can make use of review data is through sentiment analysis of the said reviews. This will enable company to get the general sentiment of users over any product. The company can then make necessary changes to the product and see the reaction in public through the reviews. More than that, the company can discontinue any product that is failing in the market.

In this project, we create a system through which we can extract the review data of any product with the help of the product's ASIN number and store it in a .csv or .xlms file with necessary data fields such as review title, star rating and the actual review text. We make use of the key aspects of this extracted data and perform sentiment analysis over this. The analyzed data is stored in dataframes and visualized for the easy understanding by the end-user.

1.1 Need Identification

E-commercial websites have many kinds of sellers, big and small. These sellers/vendors or big companies can make use of the sentiment analysis system to observe and track the performance of their products on Amazon's website. Different from big companies, which can afford to do their own data & sentiment analysis of their products, the small and moderate level businesses selling their products on the website do not generally get the means to perform data analysis over any of their products. The reason can be the ineligibility to afford it, or just being unaware about it.

So, this sentiment analysis system can fill in those gaps. Doing sentiment analysis is easiest with our project. A user would only require the ASIN number of the product and all other things will be handled by the pipelines and model on our end.

1.2 Identification of Problem

The problem that is encountered is the lack of a pipeline/system through which a user can perform sentiment analysis on reviews of any desired product. There is a lack of accessibility to such a tool, which may scrape the review data and visualize the results of sentiment analysis done on them. There is also an idea of making the existing sentiment analysis model a lot better than it's current form.

1.3 Identification of tasks

This project has to cover many aspects leading to sentiment analysis of amazon reviews, and eventually achieving it. The task required are as followed:

- Researching on pre-existing NLPs and sentiment analysis models.
- Creating a Amazon Review Scraper from scratch.
- Ability to store the scraped reviews in the most efficient format to be analyzed.
- Creating and using the pre-trained sentiment analysis models.
- Performing sentiment analysis over the extracted review data.
- Storing the results of analysis in a suitable format.
- Visualizing the data to the end-user.

1.4 Organization of the Report

There are five total chapters in this report, and each chapter explains different aspects about the project. The five chapters are the following:

1. **Introduction:**

   The first chapter is the introduction part of the report. This chapter has all the literature to explain to the reader as to what the project is about. It introduces the project and explains why the project is needed in the real world. The clients that can supposedly make use of this project

are also listed down in the *Identification of Clients* section of the chapter. The overall tasks that are to be done are also listed down in this chapter.

2. **Literature Review:**

   Literature review is the second part of this report, and it generally focuses on the literature part of the project. Research paper which are relatable to the current project are thoroughly studied, and the key finding are related to the project at hand. The findings of current project are also listed down here, as well as a clear vision of goals and objective are also presented.

3. **Design Flow/Process:**

   This chapter of the report evaluates the features listed in the literature survey and lists down the features actually required in the project. The constraints that are faced while the development of project is also listed down in this chapter under the *Design Constraints* section. Flowcharts & user-flows are also covered in this chapter

4. **Result Analysis & Validation:**

   The result chapter has all the results that the project yields.

5. **Conclusion and Future Work:**

   This is the final chapter in this project report. It concludes the finding and the report of the project by summarizing the project as a whole. References section is also included in this chapter.

CHAPTER 2

LITERATURE SURVEY

2.1 Related Work

There have been many researches so far in the domain of sentiment analysis of product reviews. Some researches which are monumental to our report are listed below.

2.1.1 Sentiment Analysis on Large Scale Amazon Product Reviews [1]

Tanjim Ul Haque & Nudrat Nawal Saber & Faisal Muhammed Shah, Dept of Computer Science and Engineering, Ahsanullah University of Science & Technology, Dhaka, Bangladesh (2018)

The research analyses the Amazon Review dataset, which serves as an excellent reference point. They employed the Multinomial Naive Bayesian and SVM as the primary classifiers for reviewing reviews because the model chooses what data it learns from, using active learning (a semi-supervised learning technique) helps prevent bottleneck problems with unlabeled data. We find this research very interesting because it compares how well different algorithms (SVM, MNB, Stochastic Gradient Descent, Random Forest, and so on) performed for the specific dataset by comparing accuracy, precision, recall, and F1 score.

2.1.2 Sentiment Analysis of Product Reviews [2]

Najma Sultana & Pintu Kumar & Monika Rani Patra & Sourabh Chandra and S.K. Safikul Alam (2019)

This research paper intends to conduct sentiment analysis on customer evaluations and identify text as "Positive," "Negative," or "Neutral." The paper provides a theoretical approach to sentimental analysis and compares several algorithms and their accuracies. It also provides a quick overview of additional methods to sentiment analysis tools.

The solution described in the study entails building an ML model in three stages: data filtering, training model, and testing model. Pre-processing the text to remove undesired components and utilize only relevant textual content for the model is what data filtration entails. All feature words (verbs, adverbs, and adjectives) are extracted and categorized during training. To compare accuracies, a dataset is trained to classification techniques such as the Nave Bayes classification algorithm, the Linear Model

algorithm, the SVM algorithm, and the Decision tree. The user input (review) is mapped to the saved feature set during the testing phase. The frequency of occurrence is used to extract features.

2.1.3 Amazon Reviews Sentiment Analysis: A Reinforcement Learning Approach [3]

Roshan Pramod Samineedi Joseph(2020)

Using reinforcement learning and a pre-trained BERT model, the researcher hoped to categorize Amazon Feedback as binary or multi-school. Amazon.com is used by the researcher to acquire product-based data. The study intends to evaluate and forecast the sentiment behind the analysis utilizing algorithms such as BERT and LSTM, as well as to improve learning by categorizing it as positive, negative, or neutral. LSTM (Long-Short-Term Memory) is a type of RNN that avoids the problem of long-term reliance. It processes data and forwards information as it propagates. The paper attempts to conclude why this model is an appropriate approach for Sentiment Analysis.

2.2 Review Summary

The previous studies/research papers done on Sentiment Analysis have made use of various techniques to increase the accuracy of the analysis in a review. They have gone ahead to even give a whole new model, which is better. In the 2009 sentiment analysis research paper of *Najma Sultana, Pintu Kumar, Monika Rani Patra, Sourabh Chandra, and S.K. Safikul Alam;* they have marked the reviews as either ‘Positive’, ‘Negative’ or ‘Neutral’. This exact same practice has been used in the implementation of models in our project. The *roBERTa model* that has been implemented in this project, has returned the ‘positive’, ‘neutral’ and ‘negative’ points to individual review text and upon these scores the review has been remarked respectively. This project has also utilized the usage of pretrained models as by *Roshan Pramod* in their research paper.

2.3 Problem Definition

The previous iterations of studies done on this topic go into a lot of detail on Sentiment Analysis and the usage of different models to get accurate results. With our project, we focus mainly on three things, that are: *gathering data for analysis, making sentiment analysis more accurate & visualizing meaningful data back to the user.* No other project has figured out a way to actually do data scraping

and analysis in one go. Our project tries to create a system/pipeline through which the user can swiftly perform sentiment analysis on a review of any product, they desire.

For this, a scraper is made from the scratch and is implemented to scrape all the reviews of the required product. The user is only required to give the ASIN number of the product, which is easily available on Amazon page of the product in question. The scraper extracts the reviews in .csv or .xlsx format for further analysis.

The sentiment analysis model utilized in this project is the pre-trained roBERTa model, which has been explained in the later sections of this report. The outputs of this model are stored in data frames and visualized back to the user. This enables a normal person to make sense of the review data. Visualization makes it easy to understand the rather complex numerical outputs of the model.

2.4 Objectives & Goals

The objectives for our project are as follow:

- Scrape/extract review data of any required product off from Amazon's website.
- Store the extracted data in a useful format.
- Run review dataset through pre-trained roBERTa model, and get the output scores.
- Store analysis data into a suitable format for visualization.
- Visualize the data and showcase to the user.

CHAPTER 3

DESIGN FLOW / PROCESS

3.1 Evaluation & Selection of Specialization / Features:

Our Sentiment Analysis model has the following features:

- **Built-in scraper** to scrape Amazon reviews of any required product.
- Utilization of the pre-trained **roBERTa Model & Hugging Face Pipeline** for accurate sentiment analysis.
- Visualization of the output of the following models, and showcasing them to the user.

3.1.1 Scraping Product Review from Amazon

For the sentiment analysis to be performed, we require datasets on which it would be performed on. In this project, we assume that the user has their desired product listed on Amazon over which they wish to perform sentiment analysis. A custom data scraper has been made for the purpose of extracting review data of the required product over from Amazon's website and store it in a .csv or .xlms file, which can be later used for performing data analysis.

The scraper asks an ASIN number from the user. ASIN number is a unique id code which is assigned by Amazon to each and every product listed on their website. With the help of ASIN number, the program gets each website html structure, using the *requests* and *BeautifulSoup* libraries. From this html code, the desired information are extracted from their respective tags, using specific attributes to locate them.

The following libraries/modules are involved in the scraper:

- **Requests (requests)**
- **BeautifulSoup**

a) Requests module:

Requests library is one of the integral part of Python for making HTTP requests to a specified URL. Whether it be REST APIs or Web Scrapping, requests is must to be learned for proceeding further with these technologies. When one makes a request to a URI, it returns a response. Python requests provides inbuilt functionalities for managing both the request and response.[4]

The scraper uses requests module to request the review web pages of the required product, and then the data is extracted in the later steps.

b) BeautifulSoup module:

Beautiful Soup is a Python library for pulling data out of HTML and XML files. It works with your favorite parser to provide idiomatic ways of navigating, searching, and modifying the parse tree. It commonly saves programmers hours or days of work.[5]

In the scraper, BeautifulSoup has been used to pull the HTML files of the review pages. We have then used the ***find*** & ***findAll*** functions of the BeautifulSoup library to find the desired tags with the respective attributes. From these tags, we have then extracted the text/data enclosed inside them. Each scraped review text, star rating and review title are stored in a data frame with help of the pandas library, and then exported to an .xlsx file.

3.1.2 Utilizing RoBERTa model & Hugging Face Pipelines

RoBERTa Model [6]

RoBERTa is an acronym that stands for Robustly Optimized BERT Pre-training Approach. It was provided by Facebook and Washington University researchers. The purpose of this work was to optimize BERT architecture training so that it took less time during pre-training.

RoBERTa has a nearly identical architecture to BERT, however the authors made some small design adjustments in its architecture and training technique to improve the outcomes on BERT architecture. These modifications are as follows:

- Removing the Next Sentence Prediction (NSP) goal: In next sentence prediction, the model is trained to predict whether the observed document segments are from the same or other documents using an auxiliary Next Sentence Prediction (NSP) loss. The authors tested multiple versions by removing and adding NSP loss and concluded that deleting the NSP loss matches or slightly improves downstream job performance.
- Training with larger batch sizes and longer sequences: Originally, BERT was trained for 1M steps with 256 sequences in a batch. The authors trained the model with 125 steps of 2K sequences and 31K steps of batch size 8k sequences in this work. This has two benefits: large

batches enhance perplexity on the masked language modelling objective and end-task accuracy. Large batches can also be parallelized more easily using distributed parallel training.

- Changing the masking pattern dynamically: In the BERT architecture, masking is done only once during data preprocessing, resulting in a single static mask. To avoid using a single static mask, training data is duplicated and masked ten times over 40 epochs, each time with a different mask strategy, resulting in four epochs with the same mask. This method contrasts with dynamic masking, in which different masking is generated each time input is passed into the model.

Hugging Face Pipelines[7]

Hugging Face is a data science and community platform that offers tools for building, training, and deploying machine learning (ML) models based on open source (OS) code and technologies. It is a community for data scientists, researchers, and machine learning engineers to share ideas, get support, and contribute to open-source projects.

In our project, we have used the **Twitter-roBERTa-base for Sentiment Analysis[8] model** to perform sentiment analysis on the Amazon reviews. This is a roBERTa-based model that has been fine-tuned for sentiment analysis using the TweetEval benchmark after being trained on 58 million tweets. This model is appropriate for English.

It is imported using the *transformer module.* This model outputs the following labels: 0-> negative, 1-> neutral, 2-> positive. The example of it's output is shown below:

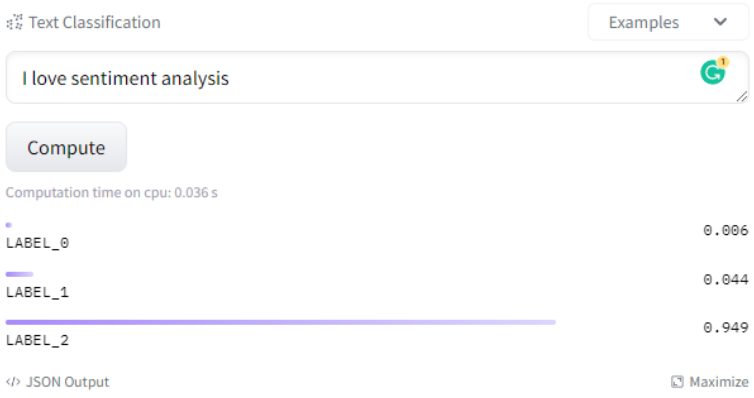


Fig: Analysis of a sentence on Huggingface website (i)

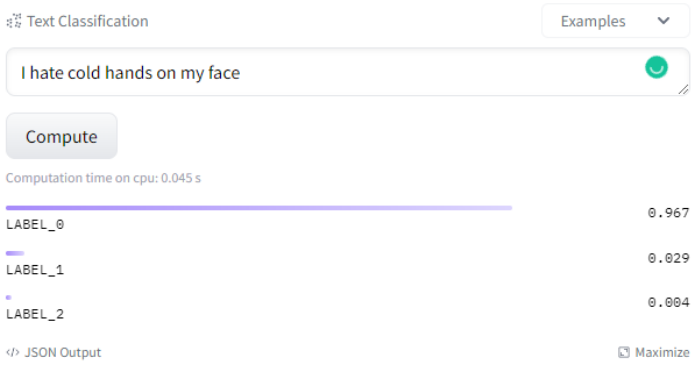


Fig: Analysis of a sentence on Huggingface website (ii)

3.2 Constraints Identification:

The following constraints were faced during the project development:

- **Time:** There was a lingering time constraint over the projects. Because of the many functions of it which had to be made from scratch, there was a time constraint that was faced. The project was finished with it being as good as it could be, but if provided extra time, it could've been even more polished.
- **Quality:** Quality is another constraint that the system has. The quality constraint focuses on the characteristics of the deliverable or product. In general, the quality of the project will be evaluated by how closely the outcome matches the expectations set in the planning stages.
- **Scraper Constraint:** Due to the scraper acting like a bot which fetches information from the website, it is more than likely that Amazon flags it down as bot and blocks the traffic from the IP that it is using. Due to this, we cannot use the scraper & model repetitively. We have to give significant cool down time for the scraper to be not blocked.
- **Resources:** Resources is one of the constraints that we face in designing the project. To avoid the effects of this constraint our team takes help of Google, teachers, friends and also collects the resources from different other websites.

3.3 User-flow Diagram

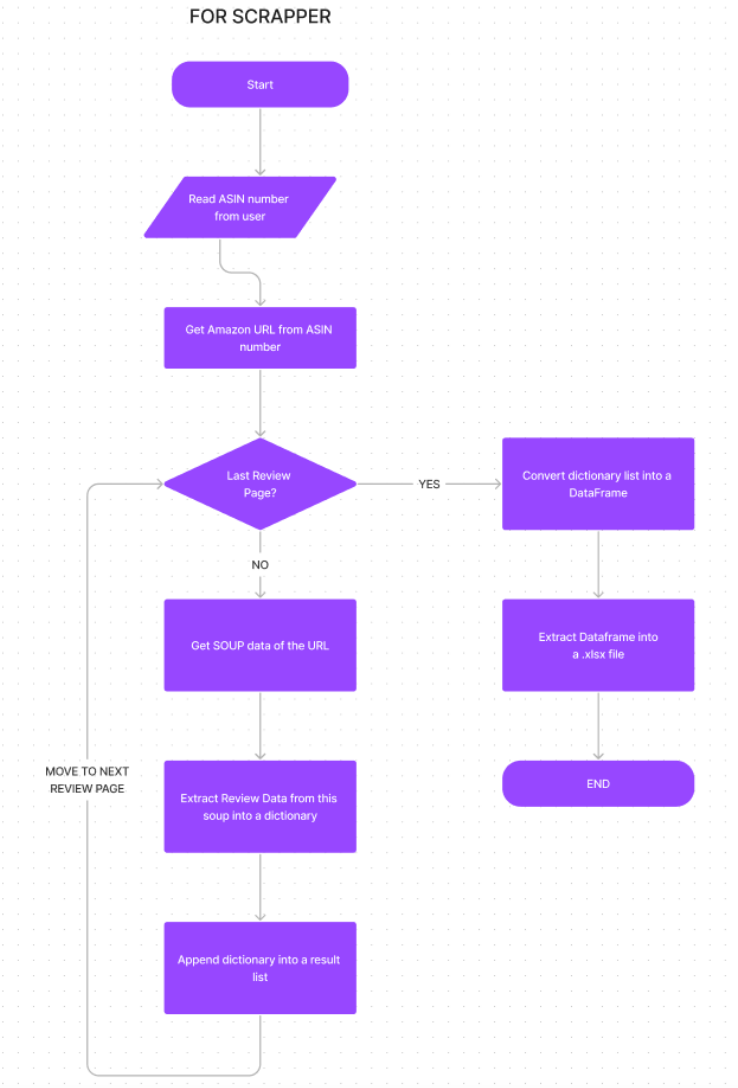


Fig: User flow for review scraper

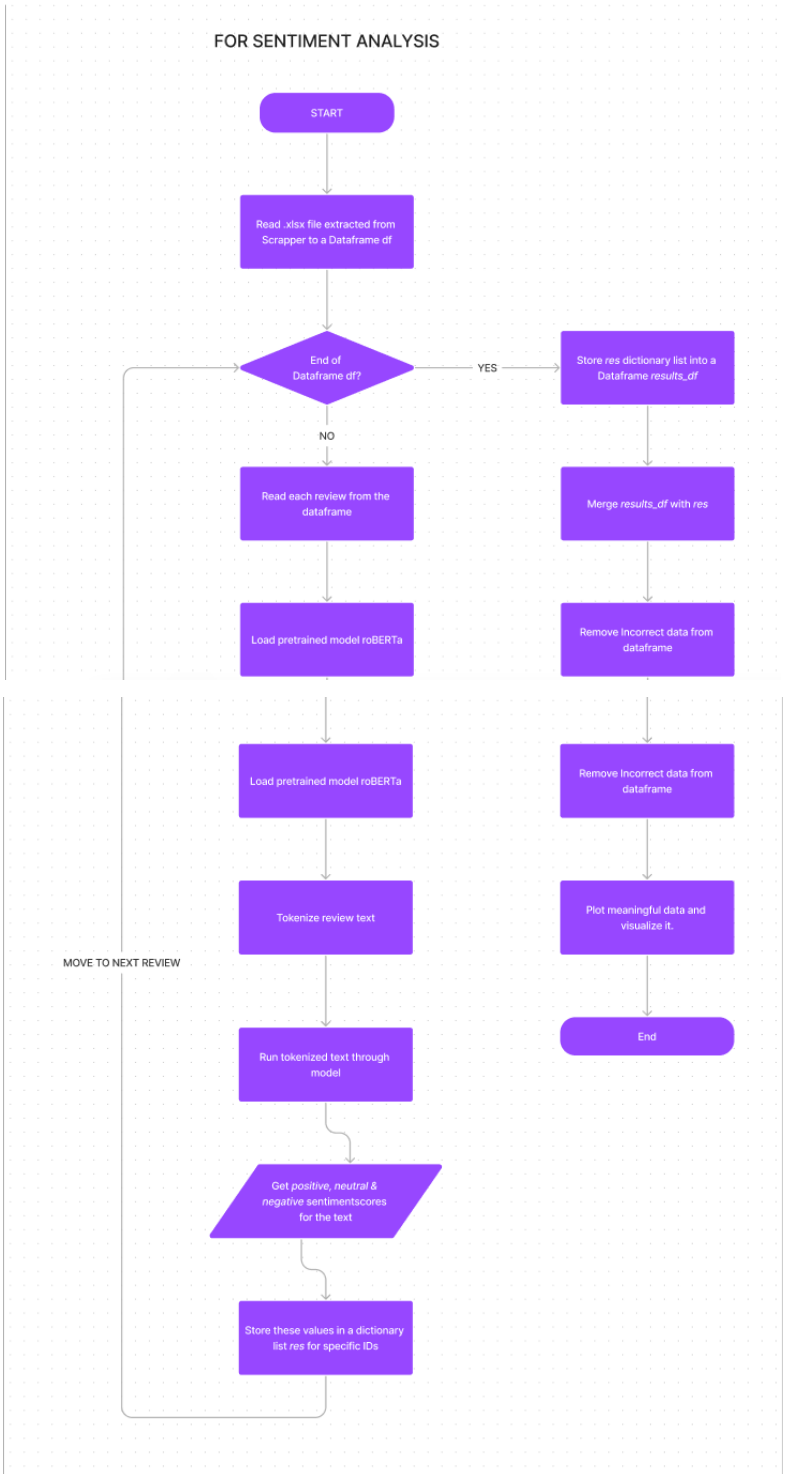


Fig: User flow for Sentiment Analysis Model

3.4 E-R Diagram

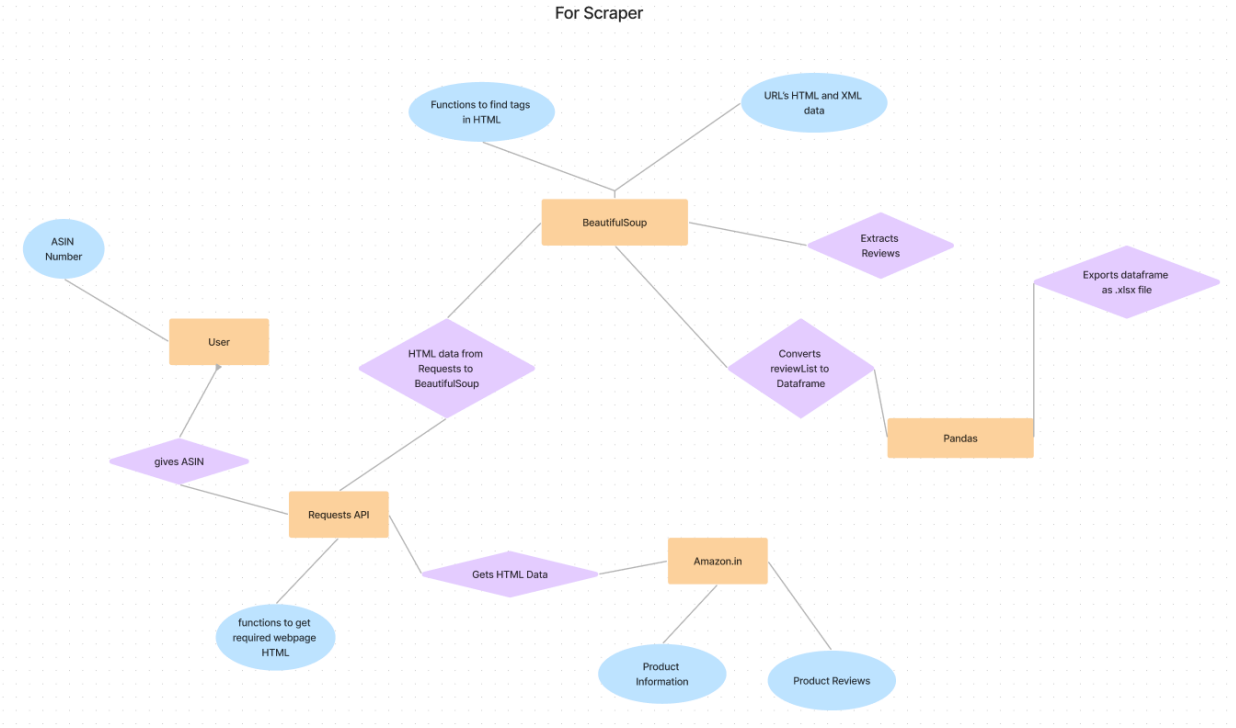


Fig: E-R diagram for review scraper

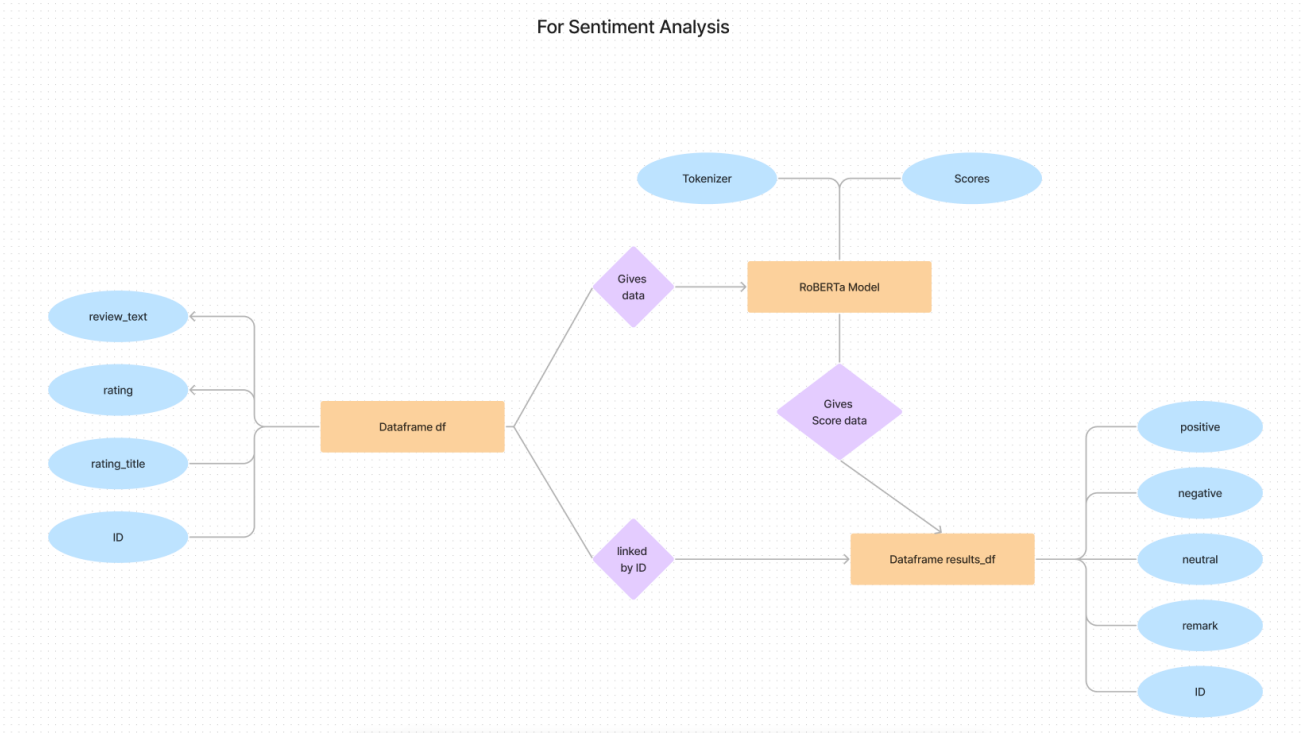


Fig: E-R diagram for sentiment analysis

CHAPTER 4

RESULTS AND VALIDATION

4.1 Implementation of solutions

The following model and tools are used to implement the desired solution:

- **Jupyter Notebook**

Fig: Jupyter Notebook logo

We have utilized a *Jupyter Notebook* for the implementation of our project. We have executed all the codes and functions in a single Jupyter Notebook.

The Jupyter Notebook is an incredibly powerful tool for interactively developing and presenting data science projects. A notebook is a document that mixes graphics, narrative prose, mathematical formulae, and other rich media with code and its output. In other words, it's a single page in which you can execute code, view the results, and add explanations, formulas, and charts to make your work more transparent, intelligible, repeatable, and shared.

Notebooks are now an important element of the data science workflow at firms all over the world. If you want to work with data, using a Notebook will help you get there faster and make it easier to communicate and share your findings.

- **Python**

Fig: Python logo

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.[9]

Python is extensively used in Data Science as it provides a vast number of libraries which can be made use of, for comprehensive data analysis. In this project also, many of the said libraries are used for the goal to be achieved.

- **Requests Module**

Fig: Request API icon

Requests allows you to send HTTP/1.1 requests extremely easily. There's no need to manually add query strings to your URLs, or to form-encode your POST data. Keep-alive and HTTP connection pooling are 100% automatic, thanks to urllib3. [10]

We have used requests module to create HTML sessions, which are used to connect to the required Amazon web pages. We have used the *get* function, which takes the target URL and headers information.

- **BeautifulSoup Module**

We have used the BeautifulSoup module in our project to extract the HTML and XML scripts of the targeted Amazon review pages. We parse the HTML code of the required web pages, and use the *find* & *findAll* function of the module to find the required tags. We use appropriate attributes to pinpoint the required attributes and extract the data from the same.

- **Transformers Module**

Transformers provides APIs and tools for conveniently downloading and training cutting-edge pretrained models. Pretrained models can help you save money on computing, lower your carbon footprint, and save the time and resources required to train a model from scratch.

Transformers support framework interoperability between PyTorch, TensorFlow, and JAX. This provides the flexibility to use a different framework at each stage of a model's life; train a model in three lines of code in one framework, and load it for inference in another. Models can also be exported to a format like ONNX and TorchScript for deployment in production environments. [11]

In this project we have imported the following sub modules from transformers: AutoTokenizer, AutoModelSequenceClassification & pipeline.

AutoTokenizer[12]

This is a generic tokenizer class that will be instantiated as one of the tokenizer classes of the library when created with the AutoTokenizer.from_pretrained() class method. AutoTokenizer is used to tokenize a text given to it. It breaks down the sentence into individual words and stores it into a list.

AutoModelForSequenceClassification

This is a generic model class that will be instantiated as one of the model classes of the library (with a sequence classification head) when created with the from_pretrained() class method or the from_config() class method.

- **Hugging Face Pipelines**

Fig: Hugging Face Icon

The pipelines are a great and easy way to use models for inference. These pipelines are objects that abstract most of the complex code from the library, offering a simple API dedicated to several tasks, including Named Entity Recognition, Masked Language Modeling, Sentiment Analysis, Feature Extraction and Question Answering.

- **Matplotlib & Seaborn**

Matplotlib is a Python charting toolkit, as well as its numerical mathematics extension NumPy. It provides an object-oriented API for incorporating plots into programmes that use general-purpose GUI toolkits such as Tkinter, wxPython, Qt, or GTK. There is also a procedural "pylab" interface based on

a state machine (similar to OpenGL) that is intended to be similar to MATLAB, albeit its use is discouraged. Matplotlib is used by SciPy.

Seaborn is a library that uses Matplotlib underneath to plot graphs. It will be used to visualize random distributions.

CHAPTER 5

CONCLUSION & FUTURE WORK

5.1 Conclusion

The snapshots of the Jupyter notebook are attached below:

Amazon Review Sentiment Analysis

In this project, we will extract amazon reviews of a particular product and do sentiment analysis on it. We will also try to visualize the some important aspects of it.

This project has 2 parts to it:

1. Scraping Reviews
2. Sentiment Analysis

1. Scraping Reviews

We will use an inbuilt scaper to scrape the amazon reviews of the required product. For this we will use the following libraries:

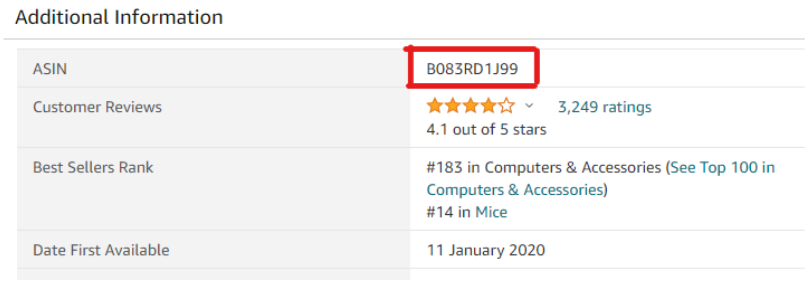
- BeautifulSoup
- Requests
- Pandas

How it works:

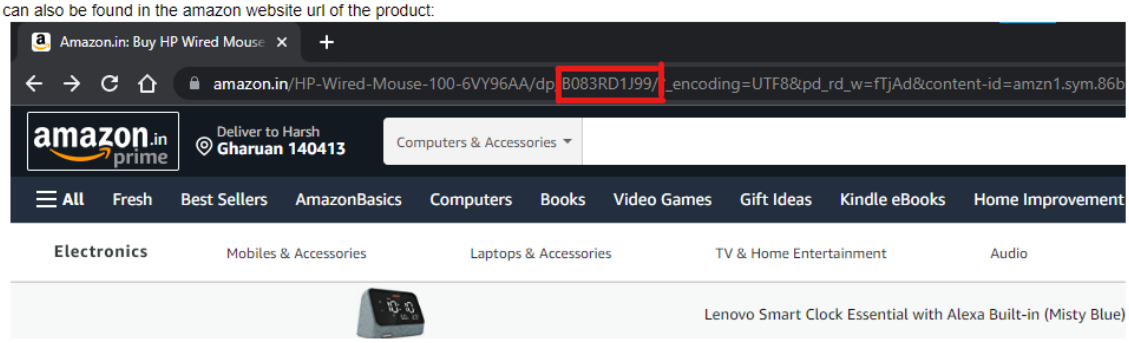
Amazon has assigned a unique **ASIN** number to each of it's product on the website(except the books). We will take the ASIN number from the user, and using this ASIN number we will extract the reviews of the required product.

HOW TO FIND ASIN NUMBER:

ASIN number is usually found in the "Additional Information" section of a product as shown below:



It can also be found in the amazon website url of the product:



Using BeautifulSoup

With the help of ASIN Number, we will get the desired review pages of the product.

> Beautiful Soup is a Python library for pulling data out of HTML and XML files. It works with your favorite parser to provide idiomatic ways of navigating, searching, and modifying the parse tree. It commonly saves programmers hours or days of work.

We will use beautiful soup to get the HTML data of the required review pages. From these pages, we will pull the required review data by using the specific tags and attributes which contain these data items.

Using Requests module

Requests allows you to send HTTP/1.1 requests extremely easily. There's no need to manually add query strings to your URLs, or to form-encode your PUT & POST data — but nowadays, just use the json method!

Using Pandas

For the scraper part, we will use pandas to convert dictionary lists into dataframes. We further export these dataframes to .csv or .xlsx files. Excel files are stored by the naming conventions : *asin_number*-reviews.xlsx

2. Sentiment Analysis

After scraping all the reviews for a particular product and storing it in a .csv or .xlsx format. We read this excel file into a dataframe, and further perform sentiment analysis on this review data that we extracted.

Libraries used:

- Pandas
- Transformers
- tqdm
- seaborn
- matplotlib

Using Pandas

Pandas is used to read the .csv data. We perform different operations on the dataframe that we create. Dataframes are a core part in data analysis.

Using Transformers

Transformers provides APIs and tools to easily download and train state-of-the-art pretrained models. Using pretrained models can reduce your compute costs, carbon footprint, and save you the time and resources required to train a model from scratch. These models support common tasks in different modalities, such as:

Transformers support framework interoperability between PyTorch, TensorFlow, and JAX. This provides the flexibility to use a different framework at each stage of a model's life; train a model in three lines of code in one framework, and load it for inference in another. Models can also be exported to a format like ONNX and TorchScript for deployment in production environments.

AutoTokenizer

This is a generic tokenizer class that will be instantiated as one of the tokenizer classes of the library when created with the AutoTokenizer.from_pretrained() class method. AutoTokenizer is used to tokenize a text given to it. It break down the sentence into individual words and stores it into a list.

AutoModelForSequenceClassification

This is a generic model class that will be instantiated as one of the model classes of the library (with a sequence classification head) when created with the from_pretrained() class method or the from_config() class method.

Pipeline

The pipelines are a great and easy way to use models for inference. These pipelines are objects that abstract most of the complex code from the library, offering a simple API dedicated to several tasks, including Named Entity Recognition, Masked Language Modeling, Sentiment Analysis, Feature Extraction and Question Answering. This text classification pipeline can currently be loaded from pipeline() using the following task identifier: "sentiment-analysis" (for classifying sequences according to positive or negative sentiments).

If multiple classification labels are available (model.config.num_labels >= 2), the pipeline will run a softmax over the results. If there is a single label, the pipeline will run a sigmoid over the result.

Softmax

The softmax function transforms each element of a collection by computing the exponential of each element divided by the sum of the exponentials of all the elements.

tqdm

The *tqdm* library provides us with simple means of using and visualizing a progress bar in our program.Using tqdm is very simple, you just need to add your code between tqdm() after importing the library in your code. You need to make sure that the code you put in between the tqdm() function must be iterable or it would not work at all.

seaborn

Seaborn is an amazing visualization library for statistical graphics plotting in Python. It provides beautiful default styles and color palettes to make statistical plots more attractive. It is built on the top of matplotlib library and also closely integrated to the data structures from pandas.

matplotlib

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible.

How does it work?

The working of this model is straightforward.

- import the pretrained *Roberta* model to be used.
- model should be given the pretrained twitter sentiment data to learn from.
- Read the excel data that was extracted with teh helo if the *scraper*
- For each review, run it through a scoring function which will perform sentiment analysis on it and give it the respective positive, neutral and negative points along with a 'POSITIVE' or 'NEGATIVE' remark
- Store this review score data in a dataframe, and merge it with the review dataframe with ID as common column
- Remove the inaccurate results, such as 1 star positive review and 5 star negative reviews
- Visualise the remaining data, to show meaningful data

REVIEW SCRAPER:

In [2]:
```
# importing libraries
from bs4 import BeautifulSoup
import requests
import pandas as pd

# a list which will list all the review values that we extract later
reviewList = []
prodId = 1

# this function parses the html source code of the given url and returns it
def getSoup(url):
    headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.0.0 S
    page = requests.get(url, headers=headers)
    soup = BeautifulSoup(page.content, 'html.parser')
    return soup

# this function extracts the review data from a given soup data
def getReviews(soup):
    global prodId
    reviews = soup.find_all('div', {'data-hook':'review'})
    try:
        for item in reviews:
            review = {
                'Id': prodId,
                'reviewTitle': item.find('a', {'data-hook':'review-title'}).text.strip(),
                'rating': item.find('i', {'data-hook':'review-star-rating'}).text.replace('out of 5 stars','').strip(),
                'reviewText': item.find('span', {'data-hook':'review-body'}).text.replace('\n',' ').strip()
            }
            prodId += 1
            reviewList.append(review)
    except:
        pass
```

```
# -- MAIN PROGRAM --

# asin is a unique id given to every Amazon product
asin = input('Enter ASIN number of product: ')

x = 1
url = f'https://www.amazon.in/product-reviews/{asin}/ref=cm_cr_arp_d_viewopt_srt?ie=UTF8&reviewerType=all_reviews&sortBy=recent&p
soup = getSoup(url)
productName = soup.find('a', {'data-hook':'product-link'}).text.strip()
print(f'Product: {productName}')

# getting reviews from all the review pages
for x in range(1,999):
    soup = getSoup(f'https://www.amazon.in/product-reviews/{asin}/ref=cm_cr_arp_d_viewopt_srt?ie=UTF8&reviewerType=all_reviews&sc
    print(f'Getting review page {x}...')
    getReviews(soup)
    if soup.find('li', {'class':'a-disabled a-last'}):
        break
    else:
        pass


# converting list into a dataframe
df = pd.DataFrame(reviewList)

# converting dataframe to an excel sheet
print('Converting to .xlsx ...')
df.to_excel(asin+'-reviews.xlsx', index=False)
print('Review extraction finished! extracted to file ' + asin +'-review.xlsx :)')

# -- END OF SCRAPER PROGRAM --
```

```
Enter ASIN number of product: B08ZJT55V9
Product: Oppo A54 (Crystal Black, 4GB RAM, 128GB Storage) with No Cost EMI & Additional Exchange Offers
Getting review page 1...
Getting review page 2...
Getting review page 3...
Getting review page 4...
Getting review page 5...
Getting review page 6...
Getting review page 7...
Getting review page 8...
Getting review page 9...
Getting review page 10...
Getting review page 11...
Getting review page 12...
Getting review page 13...
Getting review page 14...
Getting review page 15...
Getting review page 16...
Getting review page 17...
Getting review page 18...
Getting review page 19...
Getting review page 20...
Getting review page 21...
Getting review page 22...
Getting review page 23...
Getting review page 24...
Getting review page 25...
Getting review page 26...
Getting review page 27...
Getting review page 28...
Getting review page 29...
Getting review page 30...
Getting review page 31...
Getting review page 67...
Getting review page 68...
Getting review page 69...
Getting review page 70...
Getting review page 71...
Getting review page 72...
Getting review page 73...
Getting review page 74...
Converting to .xlsx ...
Review extraction finished! extracted to file B08ZJT55V9-review.xlsx :)
```

SENTIMENT ANALYSIS

Importing the required libraries

In [3]:
```
import pandas as pd
from transformers import AutoTokenizer
from transformers import AutoModelForSequenceClassification
from transformers import pipeline
from scipy.special import softmax
from tqdm.notebook import tqdm
import seaborn as sns
import matplotlib.pyplot as plt
```

Using Pretrained model

In [4]:
```
MODEL = f"cardiffnlp/twitter-roberta-base-sentiment"
tokenizer = AutoTokenizer.from_pretrained(MODEL)
model = AutoModelForSequenceClassification.from_pretrained(MODEL)
pipe = pipeline('sentiment-analysis')
```

```
No model was supplied, defaulted to distilbert-base-uncased-finetuned-sst-2-english and revision af0f99b (https://huggingface.co/distilbert-base-uncased-finetuned-sst-2-english).
Using a pipeline without specifying a model name and revision in production is not recommended.
```

Defining the scoring function

In [5]:
```
def scoring(text):
    # running for Roberta model
    tokenizedText = tokenizer(text, return_tensors='pt')
    output = model(**tokenizedText)
    scores = output[0][0].detach().numpy()
    scores = softmax(scores)
    remark = pipe(text)[0]['label']

    scores_dict = {
        'negative' : scores[0],
        'neutral' : scores[1],
        'positive' : scores[2],
        'remark' : remark
    }

    return scores_dict
```

Scoring each review and storing it in a results dataframe. Merge this dataframe with the original dataframe of reviews.

In [6]:
```
df = pd.read_excel(asin+'-reviews.xlsx')

res = {}

for i, row in tqdm(df.iterrows(), total = len(df)):
    try:
        text = row['reviewText'][0]
        myId = row['Id']
        res[myId] = scoring(text)

    except:
        pass


results_df = pd.DataFrame(res).T
results_df = results_df.reset_index().rename(columns={'index': 'Id'})
results_df = results_df.merge(df, how='left')
```

Removing incorrect data rows

In [7]:
```
for ind in results_df.index:
    if(results_df['rating'][ind] >= 4.0 and results_df['remark'][ind] == 'NEGATIVE'):
        results_df = results_df.drop(ind)
    elif(results_df['rating'][ind] < 3.0 and results_df['remark'][ind] == 'POSITIVE'):
        results_df = results_df.drop(ind)
    else:
        pass
```

In [8]:
```
results_df
```

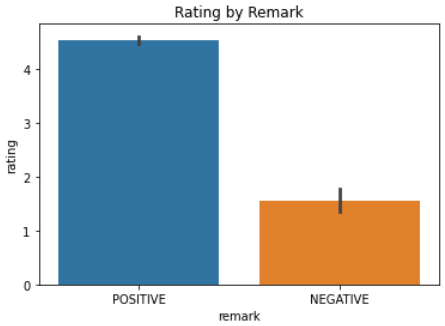
Out[8]:

	Id	negative	neutral	positive	remark	reviewTitle	rating	reviewText
0	1	0.245331	0.510024	0.244646	POSITIVE	Excellent	5.0	Good
1	2	0.245482	0.541937	0.212581	POSITIVE	Mobile Is Best For Average User, For Normal Use.	5.0	Some Important Features Are:-1.) Long Lasting ...
2	3	0.247214	0.504908	0.247878	POSITIVE	Excellent smartphone	5.0	Best smartphone in this segment by oppo brand
3	4	0.319863	0.478043	0.202093	POSITIVE	Good product	5.0	Level good
5	6	0.265475	0.49574	0.238785	POSITIVE	Useful	4.0	Useful. Camera quality is good. But seems litt...
...	...	...	...	...	...	...	...	...
686	725	0.236289	0.547696	0.216015	NEGATIVE	Wastage for money	1.0	Poor phone
687	726	0.212596	0.514271	0.273133	POSITIVE	Ok	5.0	V good
688	727	0.220812	0.562159	0.217029	POSITIVE	For my work	5.0	It was quit nice
691	730	0.212596	0.514271	0.273133	POSITIVE	Super	5.0	Very nice
694	733	0.242127	0.496443	0.26143	POSITIVE	primium looks	5.0	best phone in this price range & touch feeling...

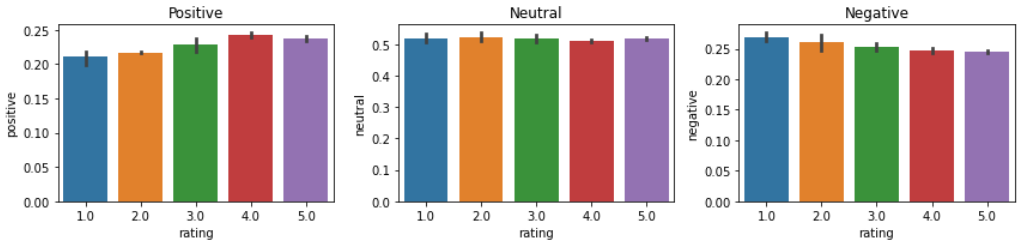
501 rows × 8 columns

Plotting the data

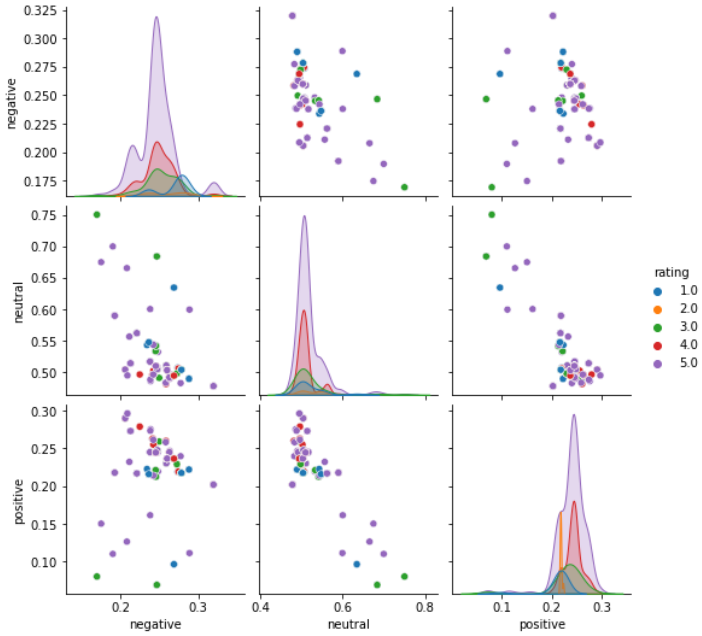
In [9]:
```
ax = sns.barplot(data=results_df, x='remark', y='rating')
ax.set_title('Rating by Remark')
plt.show()
```



In [10]:
```
fig, axs = plt.subplots(1, 3, figsize=(12, 3))
sns.barplot(data=results_df, x='rating', y='positive', ax=axs[0])
sns.barplot(data=results_df, x='rating', y='neutral', ax=axs[1])
sns.barplot(data=results_df, x='rating', y='negative', ax=axs[2])
axs[0].set_title('Positive')
axs[1].set_title('Neutral')
axs[2].set_title('Negative')
plt.tight_layout()
plt.show()
```



In [11]:
```
sns.pairplot(data=results_df,
             vars=['negative', 'neutral', 'positive'],
             hue='rating',
             palette='tab10')
plt.show()
```



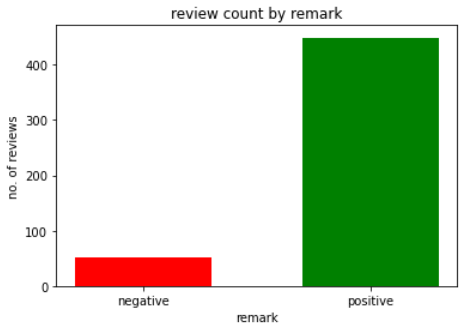
In [26]:
```
positiveCount = 0
negativeCount = 0
for ind in results_df.index:
    global positiveCount
    global negativeCount
    if results_df['remark'][ind] == 'POSITIVE':
        positiveCount += 1
    else:
        negativeCount += 1

remark_count=[negativeCount, positiveCount]
left = [1, 2]
label = ['negative', 'positive']

plt.bar(left, remark_count, tick_label = label,
        width = 0.6, color = ['red', 'green'])

# naming the x-axis
plt.xlabel('remark')
# naming the y-axis
plt.ylabel('no. of reviews')
# plot title
plt.title('review count by remark')

# function to show the plot
plt.show()
```



In [ ]:

END OF PROJECT

5.2 Future Scope

While the project at this time itself is capable of carrying out all the required functions that was originally intended for it to perform, it can be tweaked and made better for the future users. Some ways in which it can be done are listed below:

- In the scraper part, rotating proxies can be used in the program. This will enable the program to scrape data through websites without the caution of it ever being blocked on the notion of being a bot. With the addition of rotating proxies, large amounts of data can be scraped at once.
- The project is currently limited to only a Jupyter notebook and doesn't have any practical use. No third party can access it and actually make the use of it to perform sentiment analysis of their product. To deal with this, we can deploy the current pipeline of our sentiment analysis model on a website. We can create a website in which the user can interact and is able to perform the sentiment analysis on any of their desired product. The way this will work is the user can input the ASIN number on the website and then after processing, the website will output all the required data analysis and visual data required by the user/client.

5.3 References

[1] Haque, Tanjim & Saber, Nudrat & Shah, Faisal. (2018). Sentiment analysis on large scale Amazon product reviews. 10.1109/ICIRD.2018.8376299

[2] Najma Sultana &Pintu Kumar & Monika Rani Patra & Sourabh Chandra and S.K. Safikul Alam (2019): Sentimental Analysis of product reviews

[3] Dublin, Griffith & Joseph, Roshan. (2020). Amazon Reviews Sentiment Analysis: A Reinforcement Learning Approach. 10.13140/RG.2.2.31842.35523.

[4] https://www.geeksforgeeks.org/python-requests-tutorial/

[5] https://www.crummy.com/software/BeautifulSoup/bs4/doc/

[6] https://www.geeksforgeeks.org/overview-of-roberta-model/

[7] https://huggingface.co/

[8] https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment?text=I+hate+cold+hands+on+my+face

[9] https://en.wikipedia.org/wiki/Python_(programming_language)

[10] https://requests.readthedocs.io/en/latest/

[12] https://huggingface.co/docs/transformers/model_doc/auto