الجامعة العربية المفتوحة
Arab Open University

THE SENTIMENT BEHIND THE REVIEWS

[TM417A & B REPORT]GRADATION PROJECT

Author

Abdelrahman Hamdy Rezk

Student ID

1551310668

Supervised By

DR. Hala Abbas

Abstract

Sentiment analysis also called opinion mining is one of the major tasks of NLU (Natural Language Understanding), and it has special interest of many researchers in recent years, since subjective and analysis texts are useful for many applications. In particular, sentiment analysis on online reviews has become a major research field. Studies on sentiment analysis mainly focus on lexicon construction, features extraction, topics extraction and classification. In this project we aim to work on the problem of sentiment polarity and categorization of text reviews, which is one of the fundamental problems of sentiment analysis along with web-application. Data will be used in this project are online product reviews collected from different resources like Souq and Jumia. Finally this project will be for review-level classification for online buyers and companies, and should also give insight into my future work on sentiment analysis.

Acknowledgements

Allah is the first one that I can thank because all these years are just passed because of Allah.

Behind all of these years of different things with different experiments, stages and work. I would like to express my special thanks of gratitude to my project's supervisor Dr. Hala Abbas as well as my thanksgiving to Dr. Eid Emary, Dr. Mustafa Abdul-Salam, Dr. Maged Wafy and Prof. Nabil Kamel, who are giving me the golden opportunity to do this wonderful project on the topic "The Sentiment Behind The Reviews", which also helped me in doing a lot of research and I came to know about so many new things I am really thankful to them.

Especially I am extremely respectful and thankful to my supervisor Dr. Hala Abbas for the patient guidance, encouragement and advice she has provided throughout my time as his student. I have been extremely lucky to have a supervisor who cared so much about my work, and who responded to my questions and queries.

Also, I would like to thank those of being with me during this trip and help me or give me a hand for fighting to be alive as a man.

So Thanks Ezz, Atia, Araby, Halawa, Rima Samir, Abdularman Kamar, Atwa, Abdallah, Romany, Amr Eldib, Ali Mettwely, Bakry, Mohamed Magdy Eng:Mohamed Fouad, Dir:Moahmed Elsayed, Eng:Mahmoud Elsiad, Dr:Mostafa Abd Elsabour, Eng:Ibrahim Sharaf and all of the others who helped me also days ago.

Finally, I would like to extend my thanks and gratitude to my family for their undeniable continuous observation, uninterrupted enthusing and encouragement, since they have provided me with all necessary requirements to implement this project.

IT WAS NOT EASY SIR.

Contents

Chapter 1: Introduction

Chapter 2: Literature Review

Chapter 3: Requirements and analysis

Chapter 4: Design

Chapter 5: Implementation

Chapter 6: Result, Testing & Evaluation

Chapter 7: conclusion and future work

List of Figures

Chapter 1:

Introduction

Introduction

For years ago and we still deal with structured data that you can handle it via SQL and other database language, and your data was structured in the way of columns and rows, and you can handle it in easily different ways. You can get what you need by writing some of the queries that return to you what you need. These data that resides in a fixed field within a record or file, including data contained in relational databases, it actually represents about 20% of the data, because now we are dealing with a huge data from different people on different applications that become 2.5 quintillion bytes of data created each day. All of these data are collected from different resources from different people, different cultures and different languages. All of these data have a lot of values behind, but in fact, we cannot as a human deal and get what is behind these unstructured data, like what is behind your customer complaints you cannot read and know what they are said by yourself for each client on every day. So imagine you need to check out what thoughts, positive or negative of Twitter tweets, Facebook posts and reviews of products. Also on the other hand there are a different reviews from different clients related to your products, and you must check out what is behind, how your customers are affected by your products, what they are saying, what's their intuition about usage of new products, is it a positive or negative? Beside that most of the applications working in this area work on English reviews. So this project is aimed at developing an online Arabic sentiment analysis tool that helps all of those who need to improve their business via knowing what is behind their customer reviews. It's also a helpful tool for those who need to buy things from online stores to look at what others said about things they need to buy by a simple graph that represents a quick overview of the positive and negative of these reviews.

1.1 Problem

Day by day we change and now most of us use online stores and try to buy things online, instead of offline marketing and wasting time, and these online stores help us get in touch with reviews of others who use these things or bought it, but you cannot check all of these reviews and decide to buy the product you look at, also on the other hand, a lot of corporations need to know what is behind customer reviews, what customer's opinion about their products, how they are affected by the product, what are negative or positive opinions and reviews they said, which help you at the end to buy this product or leave it and search for another thing similar to, or looking for the same product at another competitive company, so the corporation should take care of changes in the market, care about their product and customer's reviews, keeping up with technology changes. Roger Jones said in his book Strategy and Projects at Work: "From the 1970s onwards it is generally true to say that periods of stability have become shorter and the necessity to make changes more frequent". On the other hand, there are a different reviews from different clients related to your products, and you must check out what is behind, how customers are affected by your products, what they are saying, what their intuition about usage of new products you made, is it a positive or negative review, all of these help you understand your business, sense it will not be possible to employ people to check out all of the products reviews, which are growing up day by day.

Also most of the applications working in this area are working on English reviews and we also face another problem of Arabic reviews, so I will talk to some of the companies that work in this field searching for Arabic reviews and then I will target all of those interested in finding something useful about their products reviews to keep going in a good way and understand their customers attitude and opinions about the company,

what they said about their products. Also this helps them improve their products, change their processes to meet what others wait for them, and help those trying to buy from online stores to get a quick understanding of what they intend to buy.

1.2 Solution

Our solution will be web application that will be based on those trying to buying from online stores and companies that need to know what is behind their customer's reviews, and this is will be by collecting data from different resources like online store, scarping reviews or searching for Arabic dataset of product's reviews, which will help us to train our model on these data, but before that we will cleaning these data and removing stop words and punctuation some of these scarping reviews or collecting data will contain HTML tags and others things that need to handling before train our model, then after cleaning these data will classify it as positive or negative for training different ML(Machine Learning) models to learn from these data by looking at these labelled data and produce best weights along with evaluating these different models help us choosing best model with best performance which can applied on unseen or unclassified data like those uploaded or provided by user needed to classify as positive and negative generate a quick statically output of percentage of positive and negative reviews, which will help user to take an action of buying this product or not, also help companies to knowing what is behind their customer's reviews, what customers opinions about usage of their products, on the other hand NLU(Natural Language Understanding) will help us knowing the sentiment behind positive and negative words, extract these words applying some of methods like Bag Of Words which will help us classifying these data correctly.

Steps of these solution will be as following

Collecting and cleaning data –

Remove HTML tags

Tokenization and Remove punctuation:

This breaks up the strings into a list of words or pieces based on a specified pattern using regular expressions

Lemmatization or Stemming:

Both tools back words to their root form like believable, believably, believes,

All of these words should be reduced to be believe.

Remove Stop words:

Most frequently used words like التى - لو - اذا

Normalization text.

Segmenting sentences in running text.

Handling Word type and word Token.

Regular Expressions which can be very useful in capturing generalizations.

Handling spell error using Minimum Edit Distance Algorithm or by n-grams with its probability.

Classifying Data –

Also all of these collected data after be in formal way, need to be classified as positive or negative.

Machine Learning work behind -

At this stage of all labeled reviews we will divide it to training and testing dataset. Maybe 80% of data collecting will be for training, and the remaining data will be for testing set, then we will get at the end of training this pre-trained model now we can use to check

upcoming and unseeing reviews which are unlabeled to label them as positive or negative.

Sentiment Behind -

Here is we will define each of these reviews in file uploaded to be positive or negative.

Web Application –

All of the previous steps will be getting to work on the web page, that you will just upload a file or provide a APIs containing your review data, then our system will generate for you a file that holds on what you upload with negative or positive for each review, and also I will display some of the charts that help you get a quick understanding of what these reviews are.

1.3 Aims and objectives of the project

Our purpose of this Web-Application is to help Arabic Egyptian people to get a quick understanding and overview of what they intend to buy from online stores, like what others who used this product or bought it said about. Also help companies to know what is behind their customer's reviews which help them improve their products and keep on with customer's needs. This will be represented by a statistical graph which will be easily checkable and understandable.

A- Support Arabic Egyptian corporation dialect in the first run:

- Extract people's opinions, sentiments, and subjectivity from the texts
- Allow corporation to looks at their customer intuition, and this will help those keeping changes.
- Improve marketing campaigns and product messaging

- Improve customer experience
- Determine brand reputation
- Based on product reviews showing the result of reviews by graphs.
- Provide an accurate sentiment analysis results, a trust system give clients to use it.
- Gives insight into the emotion behind the words, how these word affected your business on the way.
- What customers like about the corporation.
- Represent weakness and strength of their products help them improving their products.

B- Support Arabic Egyptian People

- APIs help customer on what he will buy, this can help us getting an intuition before trying to get something that maybe not likeable by a lot of people who are buy it.
- Allow user check out what is behind his comment or review he need to write.
- How his review impact on the company.
- Statistics of how corporation affected by customers reviews.

C- Achieve a wide range of users in Egypt and the MENA (Middle East and North Africa) region in future.

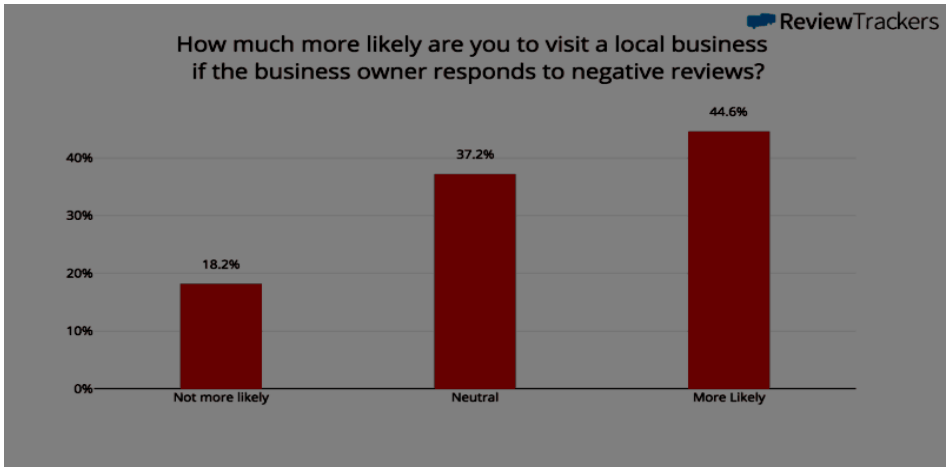

FIGURE 1 REVIEW TRAKER

1.4 Project Scope

Sentiment analysis of

- File level sentiment analysis obtains the sentiment of a complete file reviews.
- API product level sentiment analysis obtains the sentiment of a all product reviews.
- Paragraph level sentiment analysis obtains the sentiment of a single paragraph review.
- Sub-paragraph level sentiment analysis obtains the sentiment of a single sentence reviews.
- Sub-sentence level sentiment analysis obtains the sentiment of sub-expressions within a sentence reviews.

Machine Learning Models of

- Machine Learning Model that trained before with classified data.
- Run models on different unseen data for testing

- Evaluating different Models and choose best performance and most accurate

Connecting model with web-application that help others to:

- Provide data via uploaded files or links
- Statically graphs to obtain a quick understand and overview of product reviews
- Generate file to obtain classified data of you uploaded file or provided link

Target Customers

- Corporations who need to analyse their customer reviews.
- Clients who need to take a quick view before buy.
- Arabic Egyptian People

1.5 Constraints and limitations

- Internet connection to access our services.
- User should upload data in CSV format.
- Large data require more computing
- Receiving a lot of requests requires deployment handling.
- The interface is provided only in English so the user should know English

1.6 Planning the project

TASK	Week	Start	End
Planning and Organization **At this stage I will collect and analysis all of the requirements related to the project to start in specification of the requirements.**	**6 weeks**	**14/10/2019**	**27/11/2019**

Requirement Specification **At this stage after analyzing all of the requirements I'll define and specify some of them that are more likely and related to the project.**	**2 weeks**	**19/11/2019**	**4/12/2019**
Designing **At this stage and after specifying the requirements of my project I'll start to build the architecture of the project**	**3 weeks**	**5/12/2019**	**24/12/2019**
Execution and implementation **This stage will be divide to some of levels** **Collecting Data** **Web Application building**	**12 weeks**	**7/1/2020**	**1/4/2020**
Cleaning data **Remove HTML tags** **Tokenization and Remove punctuation** **regular expressions** **Lemmatization or Stemming** **Remove stop words** **Normalization text.** **Segmenting sentences in the text** **At this stage all of the data from different resources need to be cleaned which help our machine learning model to learn from and some of these steps to clean these data is mentioned above.**	**4 weeks**	**5/1/2020**	**3/2/2020** **Milestones** **Once our data is collected and cleaned, our model and web application can be tested with these data as training or uploaded to be cleaned on the web application.**
ML Algorithms, **Neural Network** **Logistic Regression** **Support Vector Machine** **At this stage I'll train clean data in some of the different machine**	**5 weeks**	**7/2/2020**	**12/3/2020**

learning algorithms and choose the one working well.			
Evaluating Model	3 weeks	13/3/2020	30/3/2020
Model Testing Milestone This is another millstone because after we test our model and its working well we can deal with any unseen data.	1 day	2/4/2020	2/4/2020
Web Application building Uploading and Downloading files, User Registration and Login Database Models, Home, About, How it works pages APIs(Application Programming Interface) At this stage I'm working with connecting machine learning pre-trained models with a web page that helps users connect to our services and uploading files or APIs that need to be classified and other helpful pages that help users to work on our services.	4 weeks	2/3/2020	30/3/2020 It's also a milestone of our project once that all of these functions should be working on web-application at the end.
Testing and Maintenance At the end all of these different functions should be worked together and tested against different stages with different stakeholders.	2 weeks	2/4/2020	15/4/2020
Closure and Evaluation Finally the project must be moved to be reviewed.	1 week	13/4/2020	23/4/2020

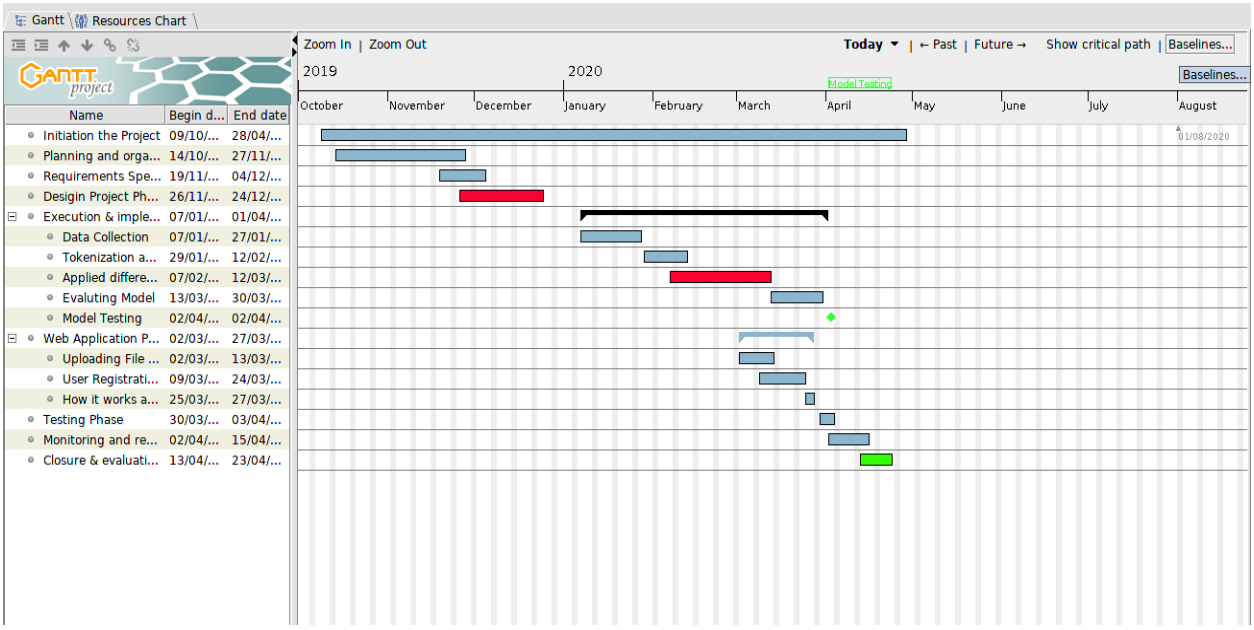

FIGURE 2 SOFTWARE DEVELOPMENT

1.7 Report Structure

Chapter 1: Background information, problem definition, suggested solution, aims and objectives, constrains and project plan
Chapter 2: include Literature Review and overview of similar system.
Chapter 3: Requirements and analysis of the system.
Chapter4: Design and Implementation of the system.
Chapter 5: Results and discussion that the results will discuss the expected hypotheses in detail and analyze their impact on the project and its thinking.
Chapter 6 Conclusions will discuss the, reasoning or inference, derivation and final theories that we arrived after all.

Chapter 2:

Literature Review

2.1 Background

One of the problems related to sentiment analysis is to categorize of what this text reflects what is behind this piece of text, review and opinion which is called also sentiment polarity, so you have a piece of written text, and you need to categorize the text into one specific sentiment polarity positive or negative or even neutral. Besides that there is a different level of these classification like document level which expresses negative or positive sentiment for the whole document, also there are other levels like sentence level, the entity and aspect level.

Also sentiment analysis is a term that refers to the use of Natural Language Understanding (NLU), besides Machine Learning (ML) in order to know the attitude of a speaker or writer toward a specific topic, brand, page or product or other things. Actually, sentiment analysis helps to determine whether a piece of text is expressing sentiments that are positive, negative, or neutral. So it is a good way to discover how people, particularly consumers, feel about a particular topic, product, or idea. However, sentiment analysis is widely used to extract meaning and subjective information from text on the Internet, including tweets, blogs, social media, news articles, reviews, and comments. This is done using a set of different techniques, including NLP, statistics, along with Machine Learning (ML) which is an important part of modern business and research. It uses Algorithms, Neural Network and Deep Learning models to give the computer the ability to learn as humans learn from experience. Machine learning algorithms automatically build a mathematical model using sample data, also known as training data, to make decisions without being specifically programmed to make those decisions.

2.2 History TimeLine of ML & NLP

1957- Frank Rosenblatt at the Cornell Aeronautical Laboratory combined Donald Hebb's model of brain cell interaction with Arthur Samuel Machine Learning efforts and created the perceptron, the perceptron is an algorithm of supervised learning that work with binary classification.

1960 - the discovery and use of multilayers opened a new path in neural network research. It was discovered that providing and using two or more layers in the perceptron offered significantly more processing power than a perceptron using one layer and The use of multiple layers led to Feedforward Neural Network and backprobgation.

1967- Marcello Pelillo, has been given credit for inventing the Nearest Neighbor Rule, it is one of the essential classification algorithms in Machine Learning. And it's another algorithm in the area of supervised learning.

1970- Seppo Linnainmaa publishes the general method for automatic differentiation (AD) of discrete connected networks of nested differentiable functions. This corresponds to the modern version of backpropagation.

1995-Corinna Cortes and Vladimir Vapnik, publish their work on support vector machines.

1989-Axcelis, Inc. releases Evolver, the first software package to commercialize the use of genetic algorithms on personal computers.

2001 – Bengio et al, The First Neural Language Model, a Feed-Forward Neural Network was proposed that help to predicting the next word in a text given the previous words.

2002-Torch, a software library for machine learning, is first released.

2006-The Netflix Prize, The Netflix Prize competition is launched by Netflix. The aim of the competition was to use machine learning to beat Netflix's own recommendation software's accuracy in predicting a user's rating for a film given their ratings for previous films by at least 10%.

2008- Bonilla et al, proposed a multi-task GP model which uses a task covariance matrix to model the relationships between tasks.

2013- Tomas Mikolov at Google, Word2Vec is a popular technique that learn word embedding using shallow neural network.

2013 and 2014 marked the time when neural network models started to get adopted in NLP.

2014-Facebook researchers publish their work on DeepFace, a system that uses neural networks that identifies faces with 97.35% accuracy.

2016-Google's AlphaGo program becomes the first Computer Go program to beat an unhandicapped professional human player using a combination of machine learning.

2018 – Pre-trained language models, Pre-trained word embedding are context-agnostic and only used to initialize the first layer in our models. In recent months, a range of supervised tasks has been used to pre trained neural networks.

2.3 System and comparisons

A- Scale

https://scale.com

Scale is a web-based application that works in the area of computer vision and natural language, including a lot of things like managed labeling services such as Sensor Fusion Cuboids, Video Annotation, 2D Box Annotation, 3D Cuboid Annotation, Semantic Segmentation, and Categorization. Combine manual labeling with best in class tools and machine driven checks to yield stunningly accurate training data. The scale provides two ways for customers to work with them on-demand and enterprise engagements. The On-Demand model means no commitment requirements, no platform fees, and you pay-as-you-go. Furthermore, with On-Demand, you can just sign up and send in tasks via our developer-friendly API. After a combination of human work and review, smart tools and statistical confidence checks and machine learning checks.

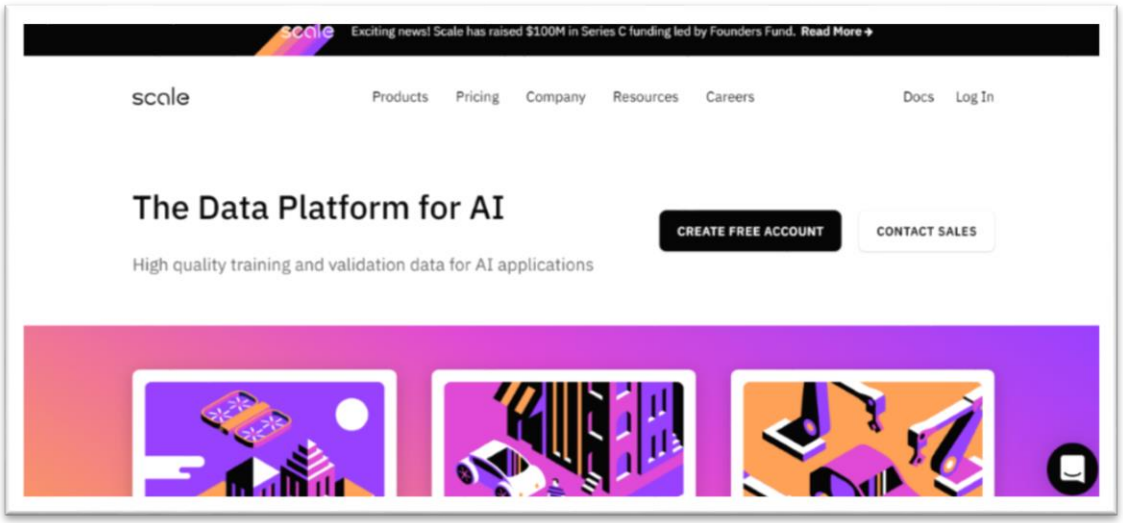

FIGURE 3 SCALE SITE

B- Lexalytics

https://www.lexalytics.com

In 2003 Lexalytics was founded by Jeff Catlin and Mike Marshall, which ships the world's first commercial sentiment analysis engine. Lexalytics provides solutions for multi-layered text analytics and natural language processing across a wide range of industries and applications. Lexalytics is a web-based application that helps corporations via sentiment analysis and opinion extraction of their customers' thoughts and translate that into actionable insights, and it's a platform that is implemented in a variety of industries for social media monitoring, reputation management and voice of the customer programs, and now days lexalytics support sentiment analysis for different languages and Unicode Emoji.

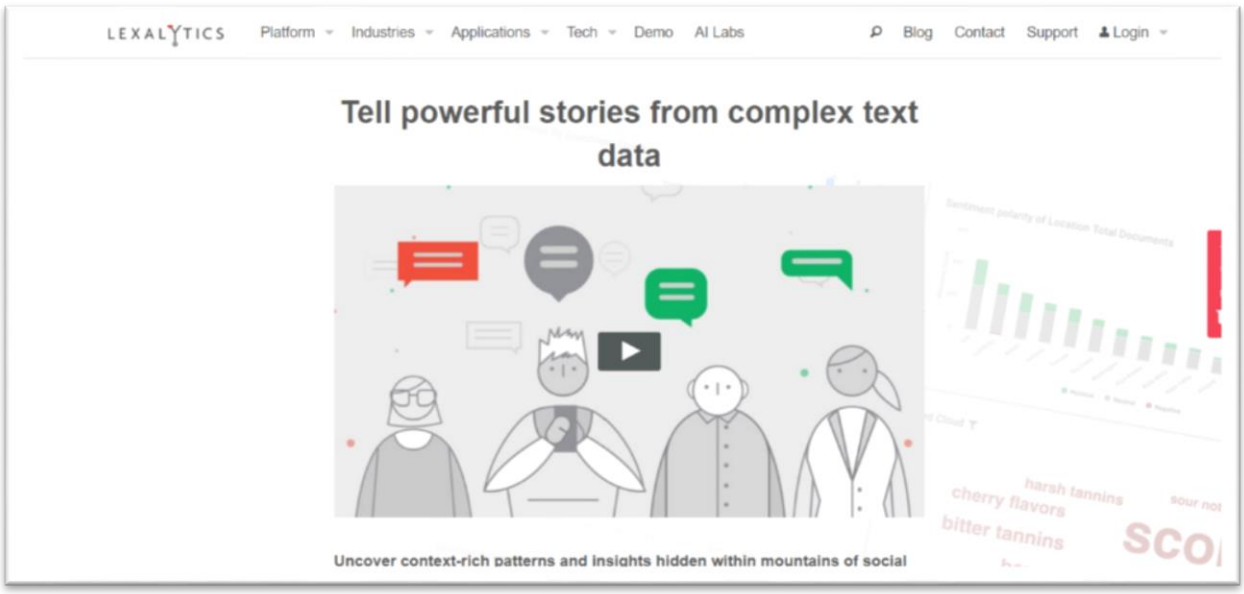

FIGURE 4 LEXALYTICS SITE

C- monkeylearn

https://monkeylearn.com

MonkeyLearn is a text mining cloud platform that allows companies to get data from text using machine learning technologies and NLU (Natural Language Understanding). It also can be employed to target appropriate ads or content, automatically sort emails to the appropriate department, or organize a news publication -- like this one -- by what kinds of people you follow, besides of this MonkeyLearn envisions adding such capabilities as automated summarization of content so that, for instance, all posted comments on a movie could be characterized in one paragraph.

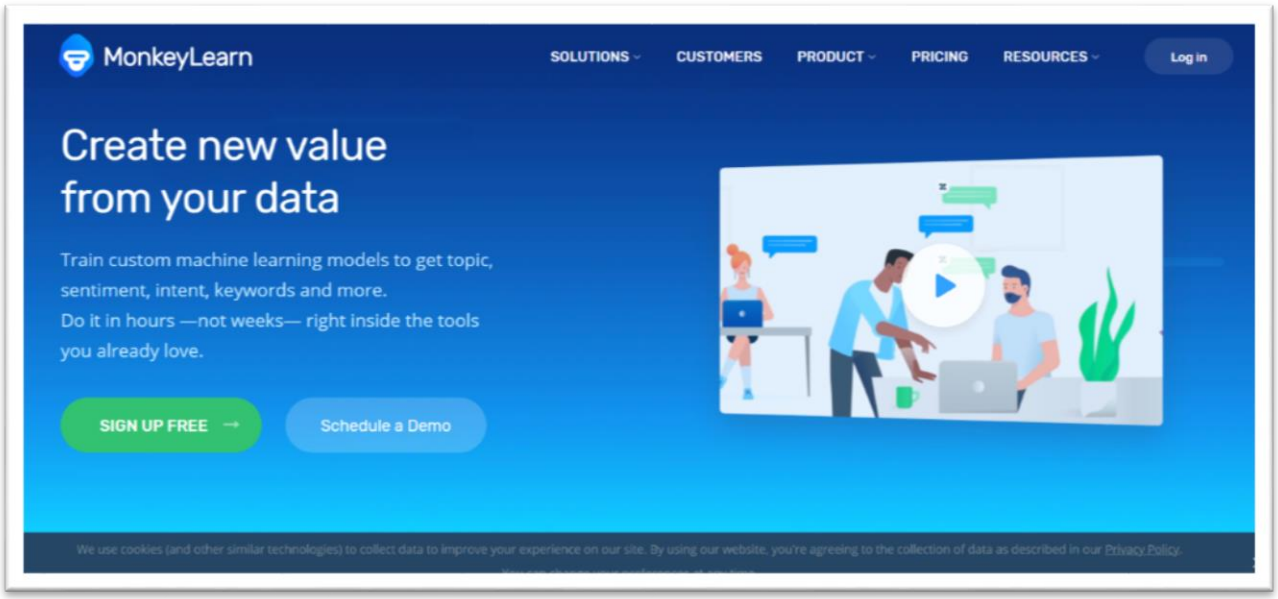

FIGURE 5 MONKEYLEARN SITE

D-brand24

https://brand24.com

Brand24 is a helpful online tool that helps brands and businesses to take care of positive comments from their brands and respond to an unsatisfied customer before the comments get ahead of your brand. Also they use advanced sentiment analysis tools to segment positive, negative and neutral opinion, track online reviews, social media mentions, blogs, news sites and a lot of. It provides the brand with instant alerts for negative mentions.

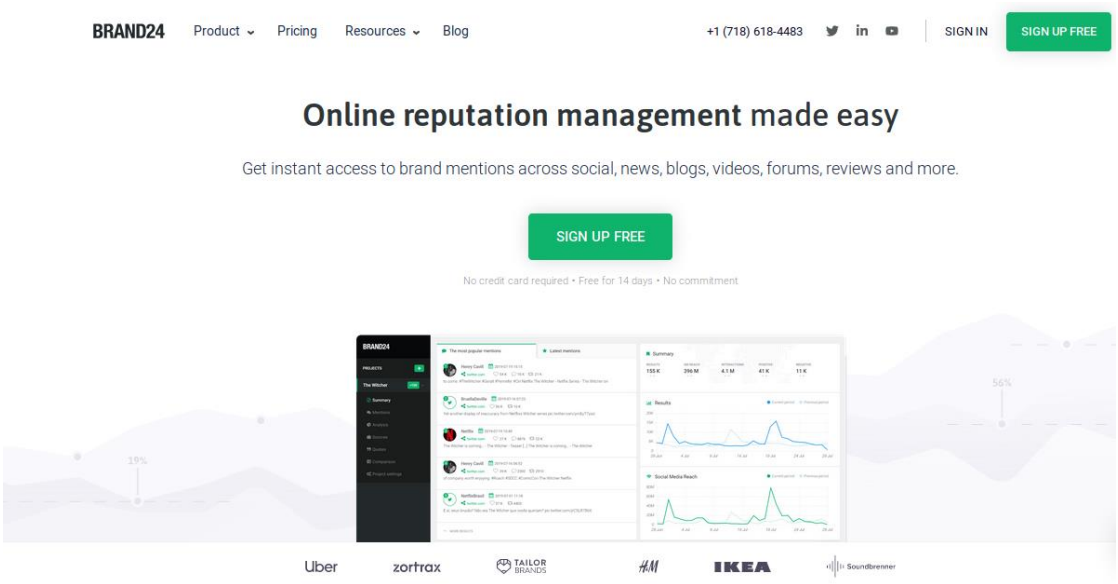

FIGURE 6 BRAND24 SITE

2.4 Drawback

- Most of the mentioned site comes with complicated features and user interface.
- Three of them work only with non-Arabic languages.
- Provides a lot of different features and requires a lot of steps to do what you need, which makes users to close the site from first check.
- Classified positive word as a negative word, but within the phrase "I wasn't disappointed", it should be classified as positive.
- It does not support normal users who need to check what is behind the product reviews they need to buy, and you should have a business account.
- Some of the sentences and pieces of text so be short to be classified as positive or negative, for example like those you find on Twitter especially, and sometimes on Facebook, but there might not be enough context for a reliable sentiment analysis.
- Most of these applications require a lot of money to check what is behind your business, until if you are a normal user you require to have a business account.
- Most of these applications give a different result every time of the same data.

Chapter 3: Requirements and Analysis

3.1 Use Case

- Customer Model
  - Sign Up: new user of our application require to sign up to use it.
  - Validate User: once user sign up he will require to verify email address.
  - Log In: once user is verified our system will login automatically and require login for him with new visits and verify it again with his password.
  - Upload file: this step can be directly open and showing for registered users, that require users to write the name of product reviews of uploaded files and with extension of the file and company or online store name.
  - Download file: once user uploads the file and its file classified he can download another file with classification of positive and negative for each review.
- Super User Model
  - Login: this model is related to Admin that can use a dashboard with only login as a super user and no required signup because we create this super user for those people we need them to login our dashboard of the system.
  - Archive User: actually some of user need to be archived instead of delete them because usually there is important historical data associated with that user, but they will be removed from our subscription and will not have access to the application.
  - Delete User.
  - Add Information: The system admin can access to write some of information details to be accessible for the user.
  - Reply contact Us E-mails.
  - Edit Information: The system admin can access to change the information data.

- System Class
  - Validate User data.
  - Validate file extension.
  - Validate APIs provided.
  - Cleaning data.
  - Run Machine learning to classify data as positive and negative.
  - Provide users with a simple graph to get a quick understanding.

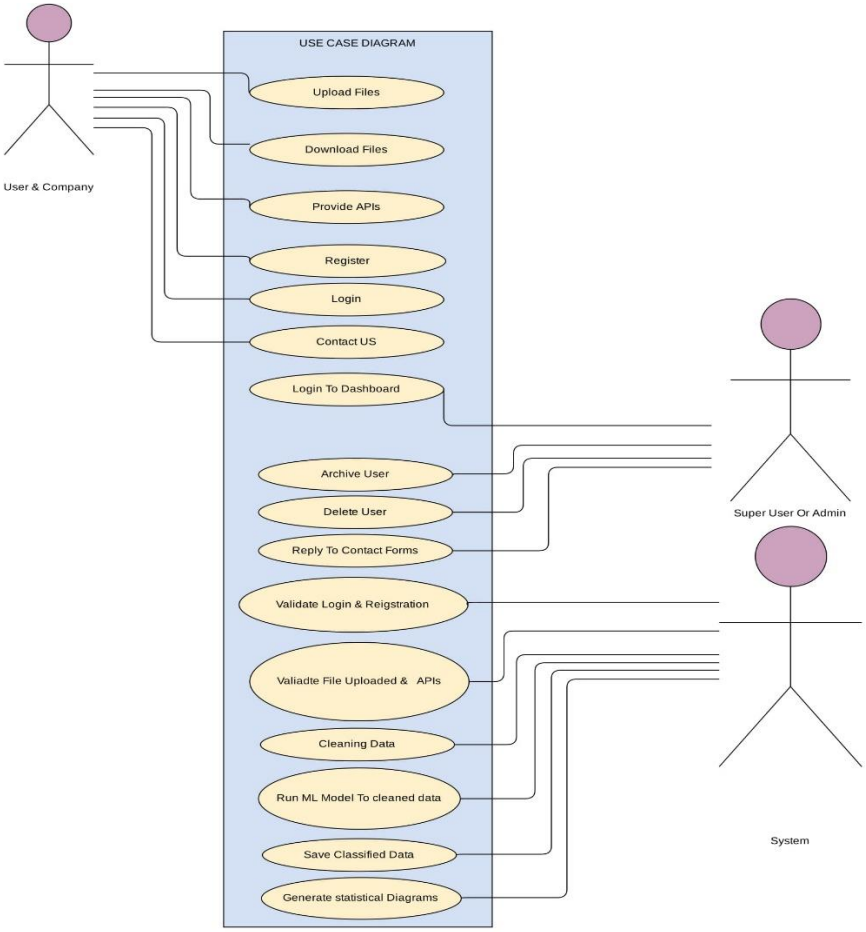

FIGURE 7 USE CASE

3.2 Functional Requirements

- User upload file.
- Analysis uploaded file
- Text Preparation
- Data of Non-textual filtered before analysis.
- Filter Stop words and steam Words and others tasks.
- Sentiment Extraction.

- Each sentence, phrase and opinion is examined to be positive or negative.
- Delivery (Presentation of Output)
- The result of converted unstructured text into meaningful information and classified reviews with file generated.
- User download file
- User input Product link
- APIs Handling to extract reviews.
- Classified reviews in statics way of positive and negatives for quick understand.
- User Registration and Login.
- Validate User Registration and Login.
- Send emails for system with new updates.
- Validate received of uploaded file in messy of software or hardware
- Receive User Opinion towards improve the System.

3.3 Non-Functional Requirements

- The Usage of the Web interface of your system should be easy and comfortable.
- The Product should visualize the results in a way of convenience colors as well as the user interface of it.
- The product should report user analysis as most as speeds.
- The Product should accurate of the result it's visualize.
- The product should be aware of user data.
- The Product should be upgraded over time with user needs, expected changes, time allowed making them.
- The Product should be fixed over time and once it fail any time.
- The Product should be responsive with all media from computer to mobile phones.
- The Product should be work on all of operating systems.

3.4 Software and Hardware Requirements

Backend

Python Language and its library.

- Pandas
- Numby
- Matplotlib
- Natural Language Toolkit (**NLTK**)
- Django Frame Work

SQL Database.

AWS (Amazon Web Services) for deployment.

Personal Computer or Labtop

Linux Operating System

Internet Connection

Front-End

- HTML
- CSS
- Java Script
- Bootstrap Responsive Design.

3.5 Diagrams

- Flow Chart

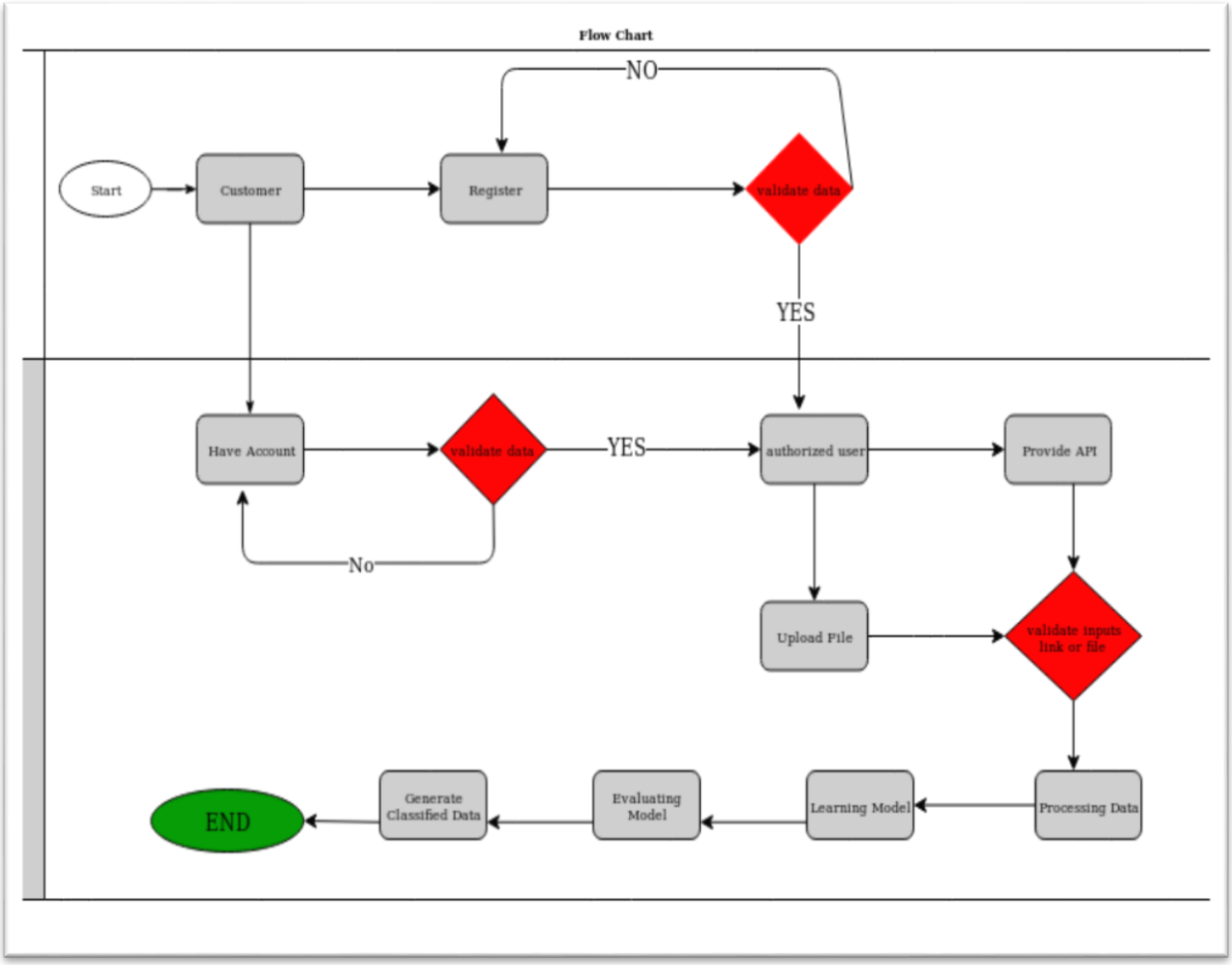

FIGURE 8 FLOWSHART 1

- Another helpful chart

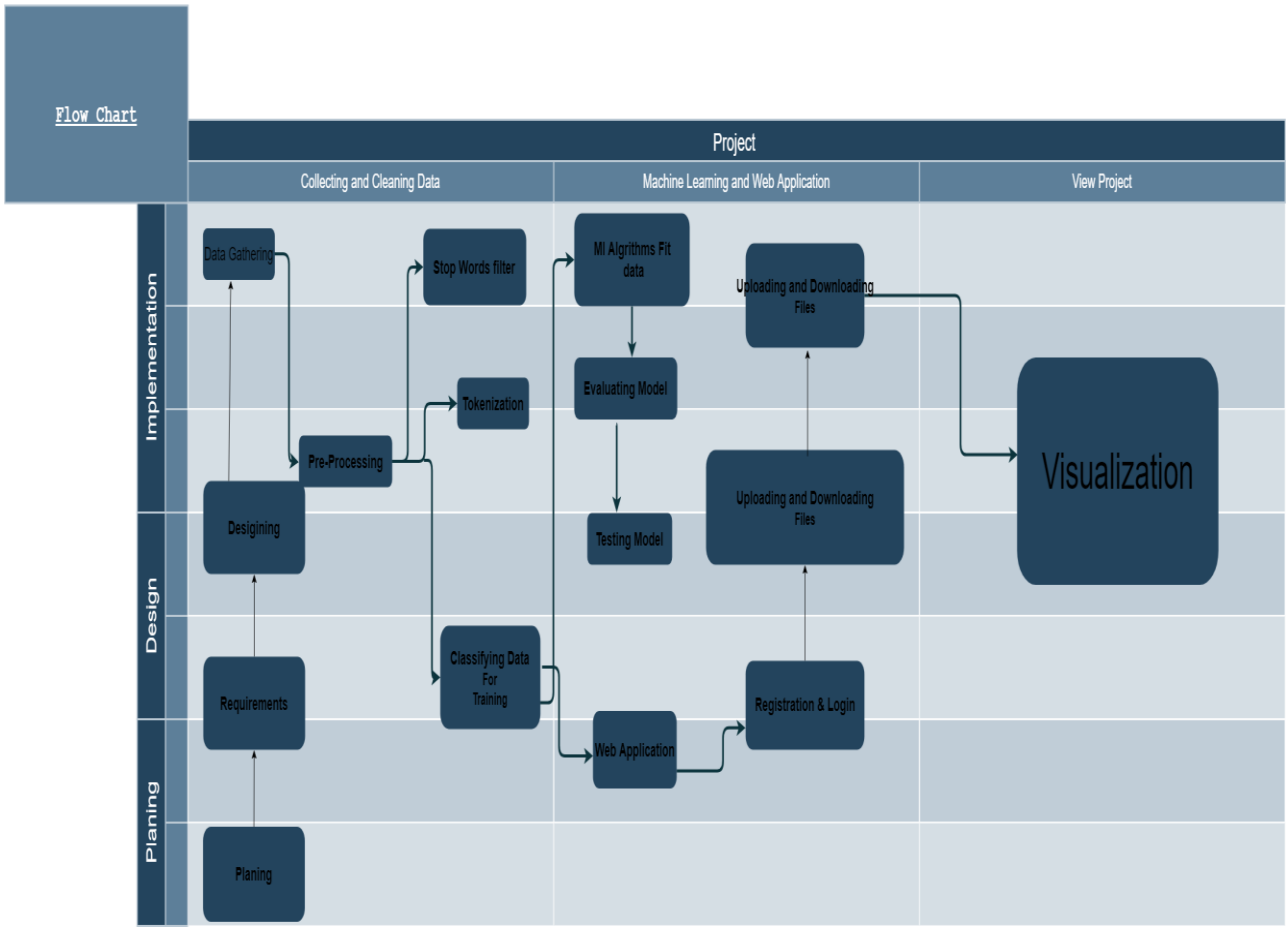

FIGURE 9 FLOWSHART 2

- ERD

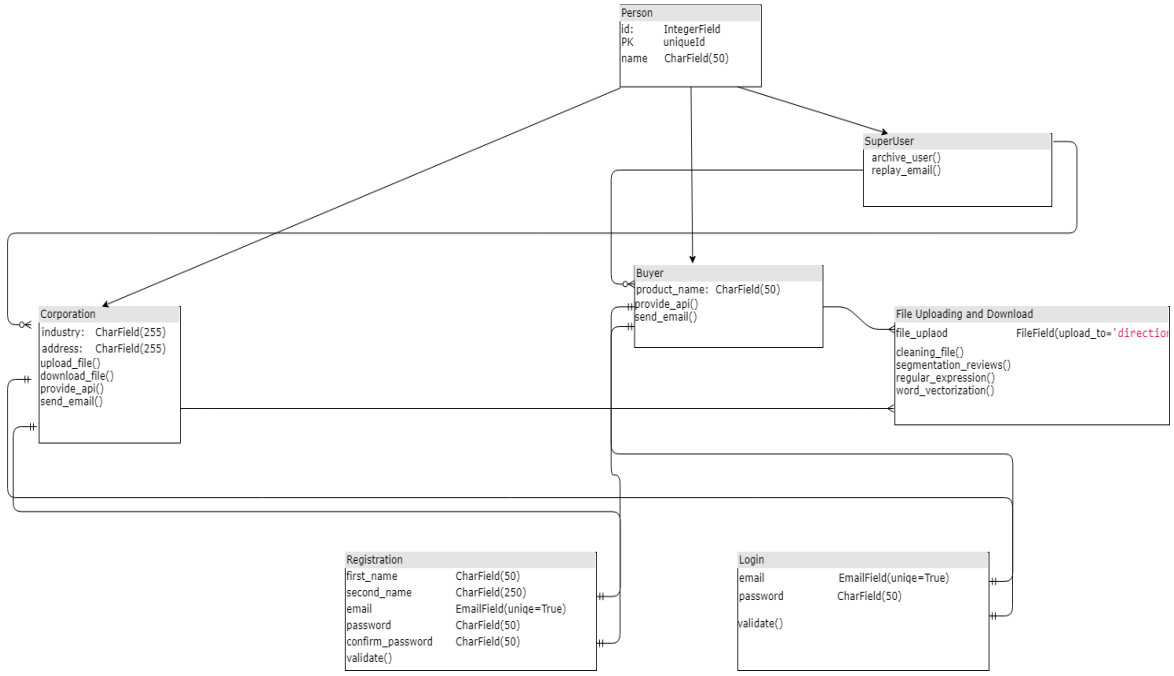

FIGURE 10 ERD

- Explanation

At this stage we will create our models of database that fits the data and user information provided to us with names and emails and user information and all of these data will be required because uncertain or invalid users will be refused and unregistered in our database. Also there is another separate phase of our system Admin dashboards that handle and deal with required action like archive and delete user, and also we will separate each part of the system like cleaning and handling data to be classified. Then all of these uploaded data will be classified and required to download by user who uploads it. Then we will provide the user with generated classified data and charts to get a quick understanding and take action if he is a buyer

to get a product or not based on our sentiment analysis of his provided product link. On the other hand, the corporation file will be classified which will help the company to improve their products and get what is behind their clients reviews.

- Sequence Diagram

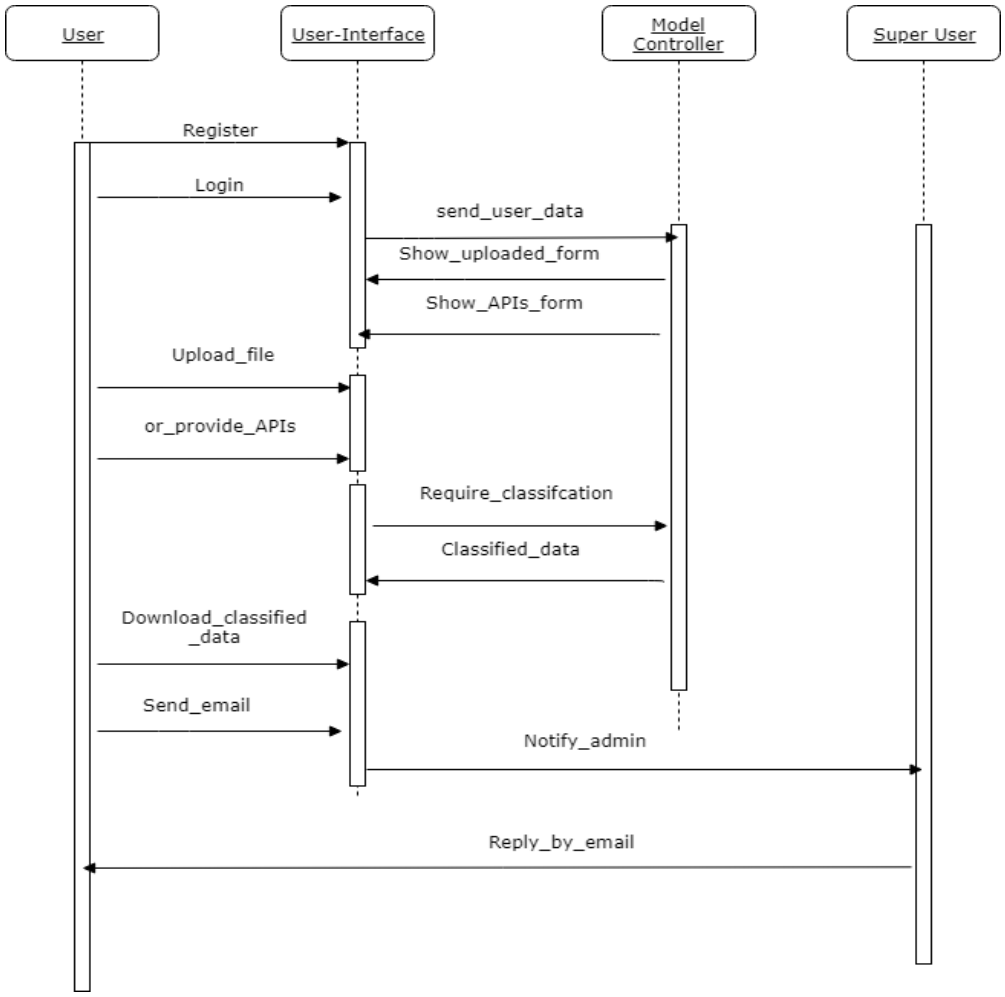

FIGURE 11 SEQUENTIAL DIAGRAM

Chapter 4:

Design

4.1 Design

First, the change in the project from part A to B is a bit different from what we discussed, and by the time issues and more features require to be added to our platform from the new discussion and changes are added and handled with a lot of debugging and different techniques. We start our process of the application with the issue of searching for data and it actually changes a lot of processes of our work and helps us know new techniques and deal with different platforms and technology. And of that I have followed the “Agile Approach”, and I have found it useful for different phases of the work I have done because of the flexibility that helps me update and back to different models of the project that I need to handle during the new issues and updates from different discussions during the process of the application with the DR.Hala Abbas.

AGILE methodology is a practice that promotes continuous iteration of development and testing throughout the software development lifecycle of the project. And both development and testing activities are concurrent unlike the Waterfall model. Agile Model & Methodology is a guide for Developers and Testers.

4.2 Advantage of the approach

4.2.1 Agile in Project management

- Agile method proposes an incremental and iterative approach to software design that involves dividing each project into specific requirements by priority, and presenting each project individually during an iterative cycle. Repetition is a routine that develops small sections of a project at a time. Each repeat is reviewed and evaluated by the development team and the customer.

- Errors can be fixed in the middle of the project.

- Every iterator has its own testing phase. It allows implementing regression testing every time new functions or logic are released.

- The agile process is broken into individual models that you can work on without the corruption of either work of others or dependencies that you require waiting for some phases of the project that should be done before you start as what we have in a waterfall approach.

- The customer has early and frequent opportunities to look at the different phases of the project and make decisions and changes to the project, which help that other dependencies that require these changes to be based on pure and complete requirements.

4.2.2 Agile In Businesses

The other side of this approach is that Businesses have demonstrated this model of project management by increasing the rate of customer satisfaction and value for companies using this model includes:

- Lower Cost

- Enables clients to be happier with the end product by making improvements and involving clients with development decisions throughout the process.

- Providing teams with a competitive advantage by catching defects and making changes throughout the development process, instead of at the end.

- At the end of every sprint, user acceptance is performed.

- Speeds up time spent on evaluations since each evaluation is only a small part of the whole project, and this helps a lot ensures changes can be made quicker and throughout the development process by having consistent evaluations to assess the product with the expected outcomes requested.

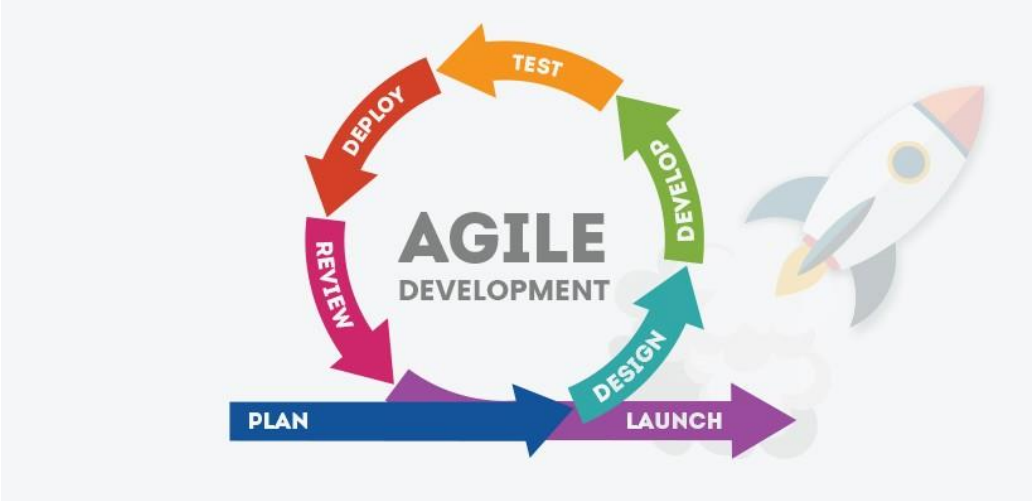

FIGURE 12 AGILE SOFTWARE APPROACH

4.3 Potential Alternative Approaches

There are different approaches to Agile. One of these approaches is the “Waterfall Approach”. The waterfall model is looking as a linear or sequential model in which the process ends and one follows each other and we cannot go back to the previous process that has ended. Waterfall Model followed in the sequential order, and so the project development team only moves to the next phase of development or testing if the previous step is completed successfully.

4.3.1 But what make agile is better than Waterfall Approach:

- **More productivity:**

Agile is incredibly effective for productivity because of its meaning to divide the problem, which helps keep everyone focused on one task at a time. That’s

precisely what teams need to do in order to complete large scope projects, and actually if the team tries to do too many things at once, they will ultimately fail from being overwhelmed and disorganized.

- **Decreased risk of missed goals:**

Any software project carries some risk, but many of these risks can be handled and reduced as possible and the agile approach reduces some risks, such as the possibility of developing products that the market does not need because we should think as we can of requirements related to market needs and agile processes help us do these processes that we need before starting to take a step in developing our software.

- **Faster implementation of solutions:**

Agile organizations can quickly respond to changing customer demands, which is essential for today's retailers. An agile approach allows organizations to easily revisit their requirements during the entire project implementation in order to keep up with a dynamic set of business drivers. In addition, by employing an agile methodology, retailers are able to greatly reduce the disconnect between business unit expectations and project delivery, as there are constant iterations of incremental delivery which can be reviewed and accepted by project sponsors. As retailers are pressured to do more with less and quickly adapt to new retail realities, an agile approach makes this possible.

- **More flexibility:**

Agile does not follow the sequential execution mode of SDLC (Software Development Life Cycle), neither has it delivered the final product at the end of the project. Agile works on small deliverables, which help fetch updates from clients as the project continues to deliver, and we can back from an unexpected fault or add new changes in an easy way. And this is the real flavor of agile of flexible working.

- **There are no ambiguous requirements:**

  Agile helps in solving the absence of ambiguity in information, inexactness, imprecision, being open to more than one interpretation, indistinct, or "fuzzy." Agile Project management uses an analytical approach to define the project and its requirements and ensure that it's understood by the client and the software company so that ambiguity is minimized.

- **Transparency:**

  Agile methodology facilitates team members to view progress directly from start to finish. This level of transparency plays an important role in creating a healthy working environment.

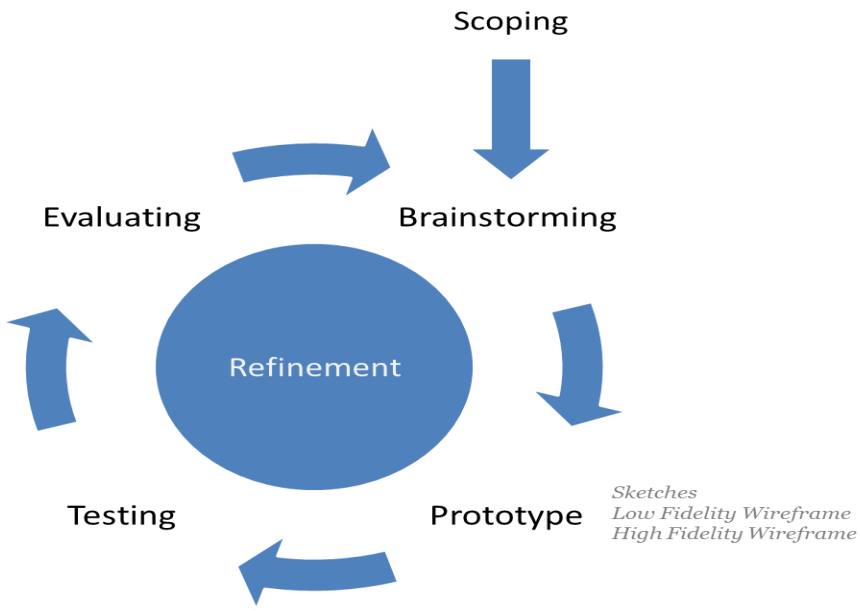

FIGURE 13 AGILE PROTOTPE

4.4 Agile In our Project

4.4.1 Agile with our project

After we have discussed some of the strongest points that make agile is a methodology that is a practice and promotes continuous iteration of development and testing throughout the software development lifecycle of the project, and also we give different differences between agile and other approaches in this section I hope to deliver how it's good this method to follow based on my project.

The Sentiment Behind Reviews project is a web-based application to help others to get an intuition about the product they need to buy from an online store. On the other hand, we have not yet had the data so we start with another process that is scraping these products from the online store and making it available to our site visitors. The data we have to get is not cleaned and requires much more process than we start with another process related to cleaning the data.

From this point how we can help clients to get an intuition about the product is via the reviews that other clients leaf on this product, and that leads us to another process which relates to machine learning algorithms that learn to predict which of these reviews are positive and which are negative, but as we know that the computer is based on (0,1) and machine learning models require numbers to deal with which of the data we scraped is text reviews so the point we go on features engineering from what we have to review.

4.4.2 Project Structure & requirements

- Front-End Design
- Back-End Design
- Scraping Data
- Cleaning Data
- Features Engineering Design
- Machine Leaning Algorithms Design

The problem here is that what is required to use the Agile Approach is that we tend to make a web application that should be responsive for all media and at the same time we have to scrap products and from these products we extract reviews and from these reviews extract features that machine learning models can work with, and based on the problem of machine learning algorithms that work as in the iterative way of idea think more tries to predict another idea tries again and so on.

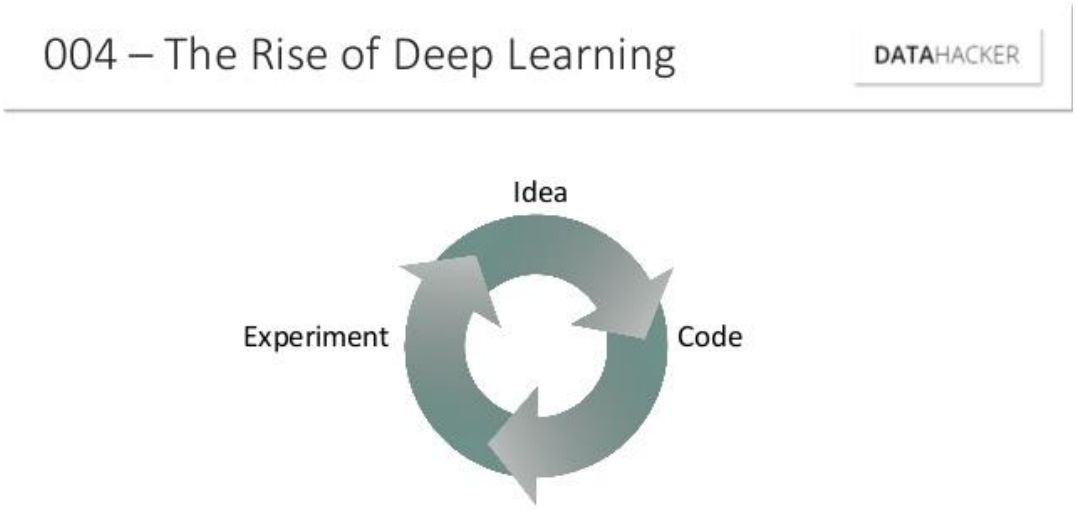

FIGURE 14 MACHINE LEARNING PROCESS

At this point, we should study the results of our model. Maybe it's a problem of overfitting and we need more training examples as we face the first phase of prediction which has a big gap between training and testing data which let us think of more reviews we need, enhance the cleaning of our reviews, think of other techniques of feature extraction and so on. Another thing that we face is under fitting, which requires more complex machine learning algorithms from naive Bayes to logistic regression moves to support vector machines.

So as we mentioned we cannot process over one thing and move to another and should not return to the previous process. It's all about the idea, code, and trying again.

Another thing is that the web application we need to represent the result. During this process, we start to implement some of the function that helps us represent the result on the site. At the same time, we work just on a simple web application and day by day start to improve it and add responsive dealing to our application to work on all media, also the process of getting more data to cost the machine a lot to work like robots and get more data so we get data on different days and test the model. It doesn't work as we need to start to get more scraping products and reviews till we have about 6000 products with more than 100,000 reviews overall on these products, so what we have mentioned is making Agile the first thing that we can think of in the phase of designing our application.

4.5 Software Design Tools

- **Jupyter Lab:**

Jupyter Lab is a web-based interactive development environment for Jupyter notebooks, code, and data. Jupyter Lab is flexible: configure and arrange the user interface to support a wide range of workflows in data science, scientific computing, and machine learning. Jupyter Lab is extensible and modular: write plugins that add new components and integrate with existing ones.

FIGURE 15 JUPYTER NOTEBOOK

- **Django:**

  Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It's free and open source.

- **Sublime**

  Sublime Text is a shareware cross-platform source code editor with a Python application programming interface (API). It natively supports many programming languages and markup languages, and functions can be added by users with plugins, typically community-built and maintained under free-software licenses.

FIGURE 16 SUBLIME CODE EDITOR

- **SQLite:**

SQLite is in-process library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine. The code for SQLite is in the public domain and is thus free for use for any purpose, commercial or private. SQLite is the most widely deployed database in the world with more applications than we can count, including several high-profile projects.

- **MongoDB :**

MongoDB is an object-oriented, simple, dynamic, and scalable NoSQL database. It is based on the NoSQL document store model. The data objects are stored as separate documents inside a collection — instead of storing the data into the columns and rows of a traditional relational database.

FIGURE 17 MONGO NON-SQL DATA BASE

- SELENIUM

SELENIUM is a free (open-source) automated testing framework used to validate web applications across different browsers and platforms. You can use multiple programming languages like Java, C#, Python etc to create Selenium

Test Scripts. Testing done using the Selenium tool is usually referred to as Selenium Testing.

- Natural Language Toolkit

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.

- Scikit-learn :

Scikit-learn is a free machine learning library for Python. It features various algorithms like support vector machine, random forests, and k-neighbours, and it also supports Python numerical and scientific libraries like NumPy and SciPy .

FIGURE 18 SCIKIT LEARN

Chapter 5:
Implementation

5.1 Implementation

Moving from what we have discussed in our design phase, we start to implement the application and besides that try different machine learning algorithms and more than two techniques for feature engineering and we will discuss that in detail below beside the implemented function.

The implementation process requires us to dealing with different technology and learn what is best for us also its start with collection data which our project is based on.

Data actually is not something that you can get for free, but we have learned how to get data for free via scraping. I have scraped over 5000 products with more than 100,000 reviews for both English and Arabic. Then we start to clean all of these data and take the Arabic reviews which our project is based on, So now we can start by discussing the scraping we used and then moving to other phases of the project.

5.2.1 Scraping Products:

The scrapping process requires some default functions that are different based on the browser we use. We have used Firefox for scraping. We start the scraping process on the first page, then for each page we enter to each product in this page and for each product, we enter to we start to get all reviews even with the difficulty that requires us to click buttons because the sites like Souq require to click more button for more reviews. The scraping works as you employ your machine as a robot and you just watch what your device can do.

Also we send all errors during the process of scraping to one big folder and for each function we create a file with errors related to the function itself.

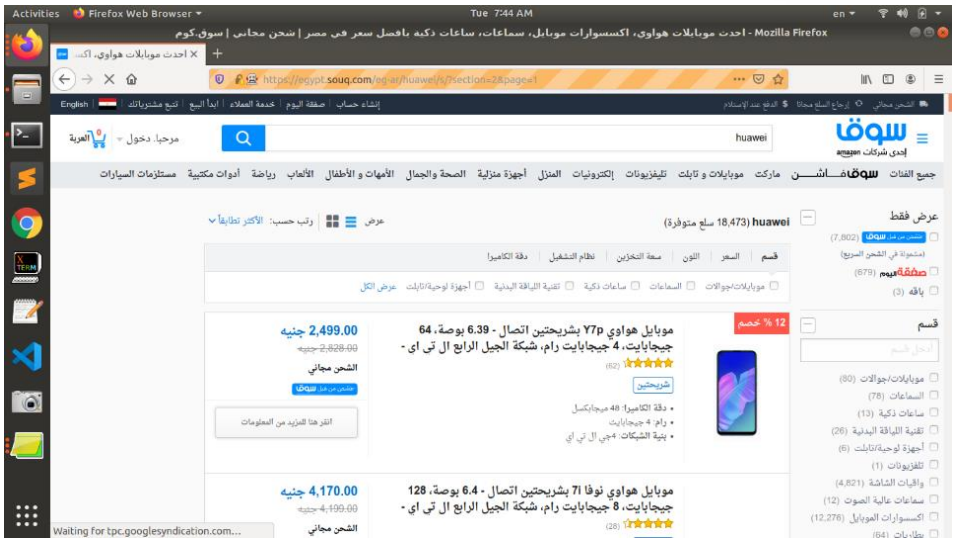

FIGURE 19 SCRAP PAGES

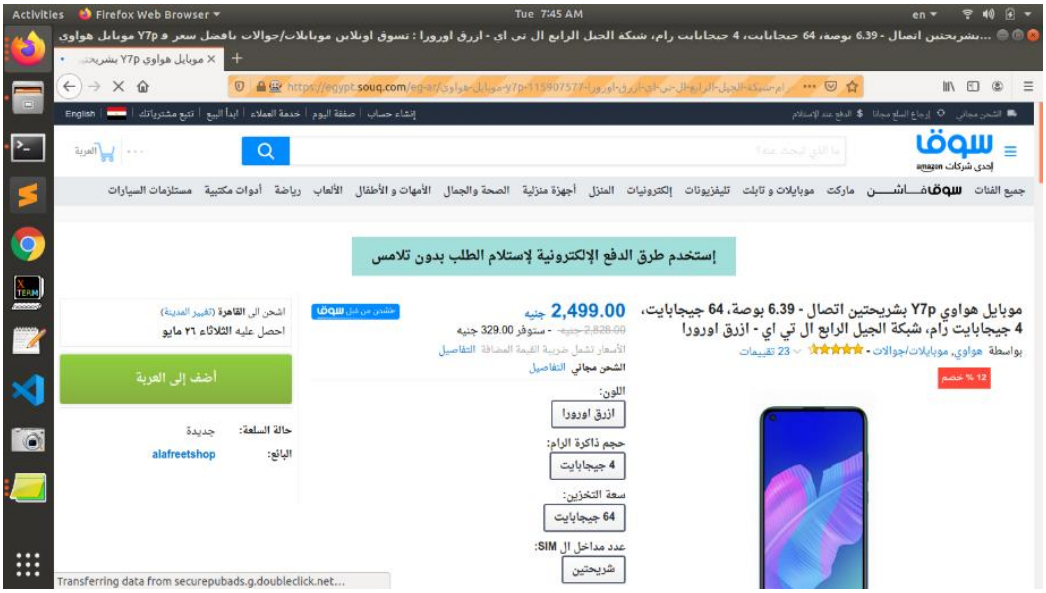

FIGURE 20 SCRAP PRODUCT

5.2.2 Scrap default requirements code

The initialize of the driver is to handle some of the issues related to firefox and from this function we can go to implement what we need.

Then we start with getting the URL as a function that takes the URL and the driver we have initialized. Then get the URL moving to what you send as URL and start to open this URL and make it available for us to get the data we need from this URL. As we can see above we send the product URL to start to get the related data of the product and for pages also.

- Initialize Driver Code:

```
- def init_driver(gecko_driver='', user_agent='', load_images=True,
  is_headless=False):
-     '''
-         This function is just to set up some of default for browser
-     '''
-     firefox_profile = webdriver.FirefoxProfile()
-
-
  firefox_profile.set_preference('dom.ipc.plugins.enabled.libflashplay
  er.so', False)
-     firefox_profile.set_preference("media.volume_scale", "0.0")
-     firefox_profile.set_preference("dom.webnotifications.enabled",
  False)
-     if user_agent != '':
-         firefox_profile.set_preference("general.useragent.override",
  user_agent)
-     if not load_images:
-         firefox_profile.set_preference('permissions.default.image',
  2)
-
-     options = Options()
-     options.add_argument('headless')
- #     options.headless = is_headless
-
-     driver = webdriver.Firefox(options=options,
```

```
-
    executable_path=f'{current_path}/{gecko_driver}',
-                                   firefox_profile=firefox_profile)
-
-       return driver
```

- Get URL Code:

```
-   def get_url(url, driver):
-       '''
-       Argument:
-           url of any page to get
-           driver that was inilized
-       return:
-           True
-       '''
-       driver.get(url)
-       driver.refresh()
-       sleep(2)
-       return True
```

5.2.3 Main features of product

The project we have built is all about reviews related to the product for future work. Maybe it will require to know some features of the product to make another Natural Language Processing analysis or Machine Learning like predicting the price of the product based on some analysis and the features of this product we have to get most of the features of each product we scrap.

- We send the driver we have initialized and start to extract the main features of the product we open is URL, also because some products may have not any features and for these products, they do not have any CSS class “product-details” and for this problem and other problem of scraping like internet connection and the process of scraping which consume a lot of time and it may fill some time because of electricity or even of the internet problems so we send all of the scraping data to online Mongo Database and we check for other time we run the scraping model as the product in our database or not and also we check for the price or the data of the product we have updated we should update our database with new changes on the product.

- Main features Code

```
-   def main_feature(driver2):
```

```
-       '''
-           Argument:
-               driver to find elements in the page
-           return:
-               most of feature related to one product
-       '''
-   # expand to get all information about product
-       try:
-           show_more = driver2.find_element_by_css_selector(".product-
    details #specs .expand")
-           if len(show_more.text):
-               show_more.click()
-       except Exception as e:
-   # send exception to log folder
-           file = open("logs_files/main_feature_product.log","+a")
-           file.write("This error related to function main_feature of
    Souq_scrapping_multithreading file\n"
-                      + str(e) + "\n" + "#" *99 + "\n") # "#" *99 as
    separated lines
-       '''
-           we have got most information about product in above try
    except
-           then for each of these features append to list
-
-       '''
-       genral_info = []
-       genral_info_dd =
    driver2.find_elements_by_css_selector(".product-details #specs-full
    dl.stats dd")
-       for dd in genral_info_dd:
-           if len(dd.text):
-               genral_info.append(dd.text)
-       return genral_info
```

5.2.4 Product Reviews

The product reviews are not easy to process which require us to get all of the reviews related to the product and some of these reviews are more than 300 reviews and it consumes a lot of time that require

us to inspect and click the button that viewers click on to see more reviews but actually what we click is our Scraping robot.

We start first of all by clicking “show-more-result” button before scraping any reviews and it helps us to make all reviews available for us first. Then in one process, we get all reviews, and actually it saves a lot of time and machine work instead of clicking, then get the reviews then check again for more reviews and get the reviews which will need a lot of handling because it will start for each time also for the new reviews we display, but this process we mentioned of click to display all reviews then get it for just one process is save us from multiple debugging and issues.

Also we wait for just 1 millisecond for each time we click the button because some sites can be knowing that is not a client and can block us from scraping. After that we save all of reviews we scraped to list and return this list to the all products function.

- ## Product Reviews Code

```
- def one_product_reviews(driver2):
-     '''
-         This functions used to get one product reviews for any
  product in souq site
-         just pass driver for this product then you will get all
  reviews
-         some of these products have more than 100 reviews but souq
  display just first 5 reviews
-         so we use the button show-more-result to display all reviews
  then we get all of product reviews
-     Argument:
-         driver of product page
-     return:
-         All reviews of this products as list of lists each of them
  display one use review.
-         some of these reviews are arabic and english,
-         this handling at second stage of cleaning data we separate
  them.
-     '''
-
-     all_reviews_for_one_pro = []
-     try:
```

```
-          show_more = driver2.find_element_by_css_selector("a.show-
   more-result")
-          while True:
-   # click until the button showmore disappear
-              if(len(show_more.text) > 1):
-                  show_more.click()
-   # wait one second after each click
-                  sleep(1)
-              else:
-                  break # break once there no other reviews you can
   display
-
-   # after you display all reviews of this product then extract all of
   them
-          reviews = driver2.find_elements_by_css_selector('ul.reviews-
   list .level-1 p')
-      except Exception as e:
-          try:
-              '''
-                  some products has a small few reviews so there is no
   showmore button,
-                  but maybe has 0 or few reviews so we get also
-              '''
-              reviews =
   driver2.find_elements_by_css_selector('ul.reviews-list .level-1 p')
-          except:
-              pass
-   # send exception to log folder
-          file = open("logs_files/one_product_reviews.log","+a")
-          file.write("This error related to function
   one_product_reviews of Souq_scrapping_multithreading file\n"
-                 + str(e) + "\n" + "#" *99 + "\n") # "#" *99 as
   separated lines
-
-      for review in reviews:
-          all_reviews_for_one_pro.append(review.text)
-          all_pro_reviews.append(review.text)
-      return all_reviews_for_one_pro
```

5.2.5 All Products Data

Moving from just one product to all of the product in the page and then we loop over all of these products and start to get each product data as we mentioned in functions above.

The function starts to get all the product in the page we send to the driver as a parameter that we use to open the page, and for each product, we start to get most of the product information and its different from the product main features we discussed above because we mean here product information is like the name of the product or the image of the product. All of these information are required which will be used to display on our application, while the main features we mean are the mobile phone type like its iPhone of Samsung, the number of cameras that are in the mobile, and others.

Once we get these information we call the function we mentioned above to get the reviews and the main features of the product and then start to check if the product we are in has been inserted to our online mongo database which we connected to during the process of scraping. Besides all of these processes we think of the logic that scraping works with and for the data like main features of the product. We do not add more load on scraping model since the product main features do not change while the price can be updated and add some discount to the product so we check for these things also product reviews if the product is inserted in the database. We need to check if there are more reviews that other client leaves for this product

- All products Data Code

```
-
-   def products_info(driver):
-       '''
-       Argumetn:
-           Driver of page with products
-       return:
-           all info related to these prodcuts for each prodcut
-       '''
-
```


```
-       products = driver.find_elements_by_css_selector("div.tpl-results
    div.list-view div.single-item")
-       page_products_info = []
-
-       for pro in products:
-           pro_url        = ''
-           pro_title      = ''
-           old_price      = ''
-           new_price      = ''
-           pro_disc_prc   = 0.0
-           pro_disc_val   = 0.0
-           image_src = ''
-           selector = pro.find_elements_by_css_selector
-
-   # first try to get main info about the product like title and url
-           try:
-               pro_url = selector('div.item-content
    a.itemLink')[0].get_attribute('href')
-               pro_title = selector('div.item-content a.itemLink
    h1.itemTitle')[0].text
-               new_price = selector('div.col-buy ul.list-blocks li
    .price-inline .sk-clr1 h3.itemPrice')[0].text
-               new_price = clean_number(new_price)
-               image_src = selector('a.img-bucket img')
-               image_src = image_src[0].get_attribute('data-src')
-
-   # check if there is oldprice of this product to get discount
-               try:
-                   len(selector('div.col-buy ul.list-blocks li .price-
    inline span.itemOldPrice')[0].text)
-                   old_price = selector('div.col-buy ul.list-blocks li
    .price-inline span.itemOldPrice')[0].text
-                   old_price = clean_number(old_price)
-                   pro_disc_prc = round(100 - ((new_price / old_price)
    * 100))
-                   pro_disc_val = old_price - new_price
-               except:
-                   old_price = 0.0
-
```

```
-   # Check of this product on our mongo cloud database
-               if db.products.count_documents({'$or': [{"product_url":
    pro_url}, {"product_title":pro_title}]}) == 0:
-
-   #get the features and reviews of the prodcut
-                   driver2 =
    init_driver(gecko_driver,user_agent=user_agent)
-                   _ = get_url(pro_url, driver2)
-                   product_reviews = one_product_reviews(driver2)
-                   main_feature_of_product = main_feature(driver2)
-                   driver2.close()
-                   one_product_info = {
-                       'product_title'               :pro_title,
-                       'product_url'                 : pro_url,
-                       'image_src'                   : image_src,
-                       'product_new_price'           : new_price,
-                       'product_old_price'           : old_price,
-                       'product_discount_percentage' : pro_disc_prc,
-                       'product_discount_value'      : pro_disc_val,
-                       'product_reviews'             : product_reviews,
-                       'main_feature_of_product'     :
    main_feature_of_product,
-                       'Uploaded_product'                   : False
-                   }
-                   _ = db.products.insert_one(one_product_info)
-                   page_products_info.append(one_product_info)
-
-               else:
-   # once product is exist get it and update it
-                   pd = db.products.find_one({'$or': [{"product_url":
    pro_url}, {"product_title":pro_title}]})
-                   driver2 =
    init_driver(gecko_driver,user_agent=user_agent)
-                   _ = get_url(pro_url, driver2)
-                   reviews_number =
    driver2.find_element_by_css_selector(".reviewInfo
    .show_reviews_number")
-                   reviews_number =
    int(clean_number(reviews_number.text))
```

```
-
-   # no need to call one_product_reviews function and hit the show_more
    button for just few added reviews
-   # so comapre different between last count of this product reviews
    with new added reviews
-                   if abs(len(pd['product_reviews']) - reviews_number)
    > 10:
-                       product_reviews = one_product_reviews(driver2)
-
-                   if pd['product_new_price'] != new_price or
    pd['product_old_price'] != old_price:
-                       db.products.update_one({'_id': pd['_id']}, {
    '$set':{
-                           'product_title'               :pro_title,
-                           'product_url'                 : pro_url,
-                           'product_new_price'           : new_price,
-                           'product_old_price'           : old_price,
-                           'product_discount_percentage' :
    pro_disc_prc,
-                           'product_discount_value'      :
    pro_disc_val,
-                           'product_reviews'             :
    product_reviews,
-                           }
-
-                   }) # end of update_one
-
-                   driver2.close()
-           except Exception as e:
-               file = open("logs_files/products_info.log","+a")
-               file.write("This error related to function products_info
    of Souq_scrapping_multithreading file\n"
-                   + str(e) + "\n" + "#" *99 + "\n") # "#" *99 as
    separated lines
-       return page_products_info
```

5.2.6 All page scraping and multithread

The scarping we used is dived to section each of them is multiple pages which differ from each other like maybe Apple products on Souq is different from Samsung products so for each of them we run its thread as different process and it all works together and it helps us a lot to work on parallel for a different category not waiting for one process to done then move to next and for this we think of something can run in parallel process and we find that multithreading process can help us doing like these process because running several threads is similar to running several different programs concurrently, but with the following benefits

Multiple threads within a process share the same data space with the main thread and can, therefore, share information or communicate with each other more easily than if they were separate processes.

Threads are sometimes called light-weight processes and they do not require much memory overhead; they are cheaper than processes.

The function starts to call the product information function which calls other functions like getting reviews and main features and this process is dependent upon each other and here we start with page one based on the URL check then move to next page until the last page then we set the next page to be the first page and all of this should be used on the server to run all the time because of updated information and because of limited resources I used my laptop to do like this process and for future, I will use online hosting to provide me with all of the new updates.

- All page scraping and multithread code

```
- def scrap_pages(page_url,next_page = 1):
-     '''
-     Argument:
-         next page = 1 as default value
-         page_url to as start page
-     return:
-         dictionary for all pages contain:
-         for each page get all prdocuts info contain:
-         for each prodcut get all reviews and main features
-     '''
```

```
-       all_page_products = {}
-       while next_page:
-   # get the driver first
-           url = page_url + str(next_page)
-           driver = init_driver(gecko_driver,user_agent=user_agent)
-           _ = get_url(url, driver)
-   # get page products info and for each product get all features and
    reviews
-           products_infos = products_info(driver)
-           all_page_products[str(next_page)] = products_info
-   # check for new pages
-           showMore = driver.find_element_by_css_selector('.pagination-
    next a')
-           next_page_url = showMore.get_attribute('href')
-           _ = get_url(next_page_url, driver)
-           next_page +=1
-           current_url = driver.current_url
-           driver.close()
-   # get the page current page number
-           current_url = re.findall('page=[0-9]+', current_url)
-           current_url = re.findall('[0-9]+', str(current_url))
-           current_url = "".join(current_url)
-           if current_url != str(next_page):
-               next_page = 1
-           driver.quit()
-       return all_page_products
```

```
if __name__ == '__main__':
    p1 = Process(target=scrap_pages, args=(souq__section_url_apple,1))
    p1.start()
    p2 = Process(target=scrap_pages, args=(souq__section_url_samsung,1))
    p2.start()
    p3 = Process(target=scrap_pages, args=(souq__section_url_huawei,1))
    p3.start()
    p1.join()
```

```
p2.join()
p3.join()
```

5.3.1 Cleaning Data:

The whole process that the web application is based on is to analyze the reviews of these products we scraped and at this point we start to extract the reviews from Mongo database, and for these reviews, we extract the Arabic reviews and start our process of cleaning these Arabic reviews from a thing like Punctuation or stop words and another thing that needs to handle before we training the model for classifying these reviews as positive or negative.

5.3.2 Remove Diacritics

The Arabic language has some of its own features that make it amazing language but with the problem of Sentiment Analysis, some of these things need to be handled because it consumes other process and memory and actually it has no effect and maybe lead to missing actual information because each char is required memory and then each of them will require machine learning algorithms to learn about as it will transform to number and take place in our learning process so chars are called diacritics which we explain in function comment. We need to delete it from our text so the two functions work as a base level string which can provide just one string and the other function which works for all of the reviews we scraped.

- Remove Diacritics Code

```
- def one_string_remove_diacritics(sentence):
```

```
-       noise = re.compile("""  ّ    | # Tashdid
-                               َ    | # Fatha
-                               ً    | # Tanwin Fath
-                               ُ    | # Damma
-                               ٌ    | # Tanwin Damm
-                               ِ    | # Kasra
-                               ٍ    | # Tanwin Kasr
-                               ْ    | # Sukun
-                               ـ     # Tatwil/Kashida
-                           """, re.VERBOSE)
-       sentence = re.sub(noise, '', sentence)
-       return sentence
-
-   def all_string_remove_diacritics(text_list):
-       '''
-       Argument:
-           list of strings
-       return:
-           list of string without special chars from Arabic language
-       '''
-       text_list = [one_string_remove_diacritics(sentence) for sentence
    in text_list]
-       return text_list
```

5.3.3 Remove Stop words

The stop words are words like [and, or, not and others], words that are mentioned in the text much more than other words and actually it does not add information to our model when we start to classify the reviews but for Sentiment Analysis process there are different issues of these stop words because libraries are deleted all of the stop words from these words [not] and this is just one word that can be deleted if we use other libraries code and for that, we have initialized our stop word file that can be used for like this problem which contains about 500 words that can be deleted without negative effect like word [not] that if deleted will change the whole meaning of the review from Negative review to Positive review and for this, we have created our stop words file.

- Remove Stope Words Code

```
- def one_string_stop_words(sentence, language):
-     '''
-     Argument:
-         string of words
-     return:
-         remove stop words from this string like this, did
-         but other words like not, no dont remove
-     '''
-     if language == 'English' or language == 'english':
-         stop_words = NLP().stopword_list # retrive stopwords list
-         sentence = sentence.split(' ')
-         updated_sentence = ''
-         for word in sentence:
-             if word not in stop_words:
-                 updated_sentence += word + ' '
-
-     elif language == 'Arabic' or language == 'arabic':
-
-         file_dir1 =
  'sentiment_behind_reviews/ml_work/stop_words/nltk_stop_words_handle.
  txt'
-         file_di2 =
  'sentiment_behind_reviews/ml_work/stop_words/stop_list1.txt'
-         file_di3
  ='sentiment_behind_reviews/ml_work/stop_words/updated_stop_words.txt
  '
-
-         stop_words_designed = []
-
  stop_words_designed.extend(convert_file_of_stop_words_to_list(file_d
  i2))
-
-         stop_words_designed = set(stop_words_designed)
-         stop_words_designed = list(stop_words_designed)
-         arabic_stop_words_designed =
  convert_file_of_stop_words_to_list(file_di3)
-
```

```
-           stop_words = arabic_stop_words_designed
-           sentence = sentence.split(' ')
-           updated_sentence = ''
-           for word in sentence:
-               if word not in stop_words:
-                   updated_sentence += word + ' '
-       return updated_sentence
```

```
-   def all_string_stop_words(text_list, language):
-       '''
-       Argument:
-           list of string
-       return:
-           list of string without stop words
-       '''
-       text_list = [one_string_stop_words(sentence, language) for
    sentence in text_list]
-       return text_list
```

5.3.4 Normalize Reviews

Normalization is actually the first step when we deal with numbers which we need to make all of the numbers in just one range because some of the features of the data take a range from 0-1 to 100-1000 and maybe in the range of millions and all of these ranges can lead to missing important information so we start with making all of these ranges from -1 to 1 or 0 to 1 based on the standard deviation and mean of mathematics, but here normalization is a bit different and actually because it's in the Arabic language that some words have a different alphabet char which at the end is similar to each other and because of sentiment analysis we should normalize these chars that represent the same thing to just one char like what we mentioned in the function below.

- Normalization Code

```
-  def one_string_normalize_arabic(sentence):
-      '''
-      Argument:
-          string of words
-      return:
-          string of words but standardize the words
-      '''
-      sentence = re.sub("ا" ,"[إأآا]", sentence)
-      sentence = re.sub("ي" ,"ى", sentence)
-      sentence = re.sub("ء" ,"ؤ", sentence)
-      sentence = re.sub("ء" ,"ئ", sentence)
-      sentence = re.sub("ه" ,"ة", sentence)
-      sentence = re.sub("ك" ,"گ", sentence)
-      return sentence



-
-  def all_string_normalize_arabic(text_list):
-      '''
-      Argument:
-          list of strings
-      return:
-          list of strings but replace some of chars  like ة to ه
   Arabic words
-      '''
-      text_list = [one_string_normalize_arabic(sentence) for sentence
   in text_list]
-      return text_list
```

5.3.5 Remove Punctuations

Punctuations are chars that can be found in any text like [?,*!] and others of these chars which also may lead to miss the learning of our model so we remove most of these punctuations and keep our text free of them.

- Remove Punctuations Code

```
- def one_string_remove_punctuation(sentence):
-     '''
-     Argument:
-         string of words
-     reutrn:
-         string without punctuation like [.!?] and others
-     '''
-     sentence = sentence.split(' ')
-     strs = ''
-     punctuations = string.punctuation
-     for word in sentence:
-         word = re.sub('[^\w\s+]',' ',word)
-         if len(word) > 1 and not (word[0] >= 'a' and word[0] < 'z'
  or word[0] >= 'A' and word[0] < 'Z'):
-             strs += word + ' '
-     translator = str.maketrans('', '', punctuations)
-     strs.translate(translator)
-     return strs

- def all_strings_remove_punctuation(text_list):
-     '''
-     Argument:
-         list of strings
-     reutrn:
-         list of strings without punctuation like [.!?] and others
-     '''
-     text_list = [one_string_remove_punctuation(sentence) for
  sentence in text_list]
-     return text_list
```

5.3.6 PIP Line

After we have done all of the required cleaning processes we pass the reviews to the pipeline of these created functions, and we can see returned cleaner reviews.

```
- def arabic_pip_line(text_list):
-     text_list = all_string_remove_diacritics(text_list)
-     text_list = all_strings_remove_punctuation(text_list)
-     text_list = all_string_normalize_arabic(text_list)
-     text_list = all_string_stop_words(text_list, 'Arabic')
-     return text_list
```

5.4.1 Features Extraction & Model Prediction

The machine learning model can work with numbers not text data and for this point we need to provide the Machine Algorithms the data in numbers and this process can be done by what is called features extraction and there are different Algorithms of these process one of them and the basic Algorithm is called One-Hot Vectorization and its work as for all of the words in the corps we used ordered the words using the alphabetic order and give each of the indexes from 0 to the last word in our available data and for this ordered the algorithm used to predict which words are similar to each other and thus lead to bad result because some words like [Orange & Apple] have the same meaning but the model will keep these words away of each other and make words like [Apple and again] to have similar meaning based on alphabetic order and for this we used another technique

that helps us get a better result in training the machine learning algorithms like TF-IDF (term frequency & inverse term frequency).

For each of these feature techniques we will discuss the different Machine Learning Models we used to predict the negative and positive reviews.

Also based on the reporting requirement I will discuss the algorithms and its implementation for this chapter and move to the next chapter of the result. I will discuss the result of each model based on the features extractions we used.

5.4.2 TF-IDF

The first part is TF that the term frequency is the number of times a word appears in a document divided by the total number of words in the document. Every document has its own term frequency.

The second part is IDF inverse term frequency is the log of the number of documents divided by the number of documents that contain the word w. Inverse data frequency determines the weight of rare words across all documents in the corpus. Also, the TF-IDF makes these rare words have more weights because its repeat is not a big as other words like [they are and others] that may have a lot of time in our text so it takes the inverse after counting the number of each term over its document and overall corpus.

The corps we have used here is our collected reviews which are about 26,000 Arabic reviews and about 10,000 of them are Negative and the others are Positive. We have divide these reviews to training and testing reviews in the ratio of 80% of them for training and the rest are negative, and we have randomly ordered these reviews for Machine Learning Algorithms.

```
print( Our testing data now are:  + str(len(y_test)) +  labels )
```

```
Our training data now are: 21022 Reviews
Our testing data now are: 5256 Reviews
Our training data now are: 21022 labels
Our testing data now are: 5256 labels
```

We provide the whole data for tf-idf and then we start to transform the training and testing data.

- TF-IDF Code

```
- def tfidf_vectorizer(df):
-     '''
-     Argumen:
-         df dataframe of multiple reviews
-     return:
-         Train & test arrays that can fir to the model
-     '''
- # I fit the vector to all of the data
-     tfidf_vectorizer = TfidfVectorizer()
-     tfidf_vectorizer = tfidf_vectorizer.fit(df)
-     word_idf_weights = tfidf_vectorizer.idf_
-     print("Our 10 words weights\n\n",word_idf_weights[:10])
-     return tfidf_vectorizer
- X = df_file['Arabic Reviews']
- y = df_file['polarity']
- X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=
  .2)
- tfidf_vectorizer = tfidf_vectorizer(X)
-
-
- training_data = tfidf_vectorizer.transform(X_train)
```

```
- training_data  = training_data.toarray()
- testing_data = tfidf_vectorizer.transform(X_test)
- testing_data  = testing_data.toarray()
```

5.4.3 Multinomial Model

After we have used the tf-idf features extraction I provide the model with these converted text to number to make prediction on.

Multinomial Naive Bayes is a specialized version of Naive Bayes that is designed more for text documents. Whereas simple naive Bayes would model a document as the presence and absence of particular words, **multinomial** naive bayes explicitly models the word counts and adjusts the underlying calculations to deal with in.

- Multinomial Model Code

```
- clf_MultinomialNB = MultinomialNB()
- model = clf_MultinomialNB.fit(training_data, y_train)
- predict = model.predict(training_data)
- print("F1 score of our training data is: ", f1_score(y_train, predic
  t, average='micro'))
- print("Evalution Matrix of training data is \n", confusion_matrix(y_
  train, predict))
- predict = model.predict(testing_data)
- print("F1 score of our testing data is: ", f1_score(y_test, predict,
   average='micro'))
- print("Evalution Matrix of training data is \n", confusion_matrix(y_
  test, predict))
```

5.4.4 Logistic Regression Model

Logistic regression is a type of statistical analysis that is often used for predictive analytics and modeling, and extends to applications in machine learning. In this analytical approach, the dependent variable is finite or categorical: either A or B (binary regression) or a range of finite options A, B, C or D (multinomial regression). It is used in statistical software to understand the relationship between the dependent variable and one or more independent variables by estimating probabilities using a logistic regression equation.

And because of our problem of Sentiment Analysis we use these models which helpful for categorical data either the result is positive or negative or even more like in images problem of predict handwriting number from [0-9].

- Logistic Regression Model Code

```
- clf_LogisticRegression = LogisticRegression(penalty='l2', solver='li
  blinear',max_iter=1000)
- logistic_model = clf_LogisticRegression.fit(training_data, y_train)
- predict = logistic_model.predict(training_data)
- print("F1 score of our testing data is: ", f1_score(y_train, predict
  , average='micro'))
- print("Evalution Matrix of training data is \n", confusion_matrix(y_
  train, predict))
- predict = logistic_model.predict(testing_data)
- print("F1 score of our testing data is: ", f1_score(y_test, predict,
   average='micro'))
- print("Evalution Matrix of training data is \n", confusion_matrix(y_
  test, predict))
```

5.4.5 Count Vectorizer

Count Vectorizer is actually like the TF-IDF, but it's just for the first part, which relates to only counting the number of times a word appears in the document, which results in bias in favour of most frequent words. This ends up ignoring rare words which could have helped us in processing our data more efficiently.

- Count Vectorizer Code

```
- def count_vectorize(df):
-     '''
-     Argumen:
-         df dataframe of multiple reviews
-     return:
-         Train & test arrays that can fir to the model
-     '''
- # I fit the vector to all of the data
-     vectorizer = CountVectorizer()
-     vectorize = vectorizer.fit(df)
-     return vectorizer
-
- count_vectorizer = count_vectorize(X)
-
- training_data = count_vectorizer.transform(X_train)
- training_data  = training_data.toarray()
- testing_data = count_vectorizer.transform(X_test)
- testing_data  = testing_data.toarray()
```

5.4.6 Multinomial Model

Also we have predict with the same models for Count Vectorizer features extraction.

- Multinomial Model With Count Vectorizer Code

```
- clf_MultinomialNB = MultinomialNB()
- model = clf_MultinomialNB.fit(training_data, y_train)
- predict = model.predict(training_data)
- print("F1 score of our training data is: ", f1_score(y_train, predic
  t, average='micro'))
- print("Evalution Matrix of training data is \n", confusion_matrix(y_
  train, predict))
```

```
predict = model.predict(testing_data)
print("F1 score of our testing data is: ", f1_score(y_test, predict,
 average='micro'))
print("Evalution Matrix of training data is \n", confusion_matrix(y_
test, predict))
```

5.4.7 Logistic Regression Model

- Logistic Regression Model With Count Vectorizer Code

```
clf_LogisticRegression = LogisticRegression(penalty='l2', solver='li
blinear',max_iter=1000)
logistic_model = clf_LogisticRegression.fit(training_data, y_train)
predict = logistic_model.predict(training_data)
print("F1 score of our testing data is: ", f1_score(y_train, predict
, average='micro'))
print("Evalution Matrix of training data is \n", confusion_matrix(y_
train, predict))
predict = logistic_model.predict(testing_data)
print("F1 score of our testing data is: ", f1_score(y_test, predict,
 average='micro'))
print("Evalution Matrix of training data is \n", confusion_matrix(y_
test, predict))
```

5.4.8 Word2Vec

The most important feature extraction I have used is word2vec because of its features which depend on Neural Network and it does not like other tf-idf based on count words. It's understanding the words itself from which it is like other words like [Orange and Apple] words related to each other and can happen with each other and they are very similar to each other. Word2vec is a two-layer neural net that processes text by “vectorizing” words. Its input is a text corpus and its output is a set of vectors: feature vectors that represent words in that corpus. While Word2vec is not a deep neural network, it turns text into a numerical form that deep neural networks can understand.

Word2vec's applications extend beyond parsing sentences in the wild. It can be applied just as well to genes, code, likes, playlists, social media graphs and other verbal or symbolic series in which patterns may be discerned.

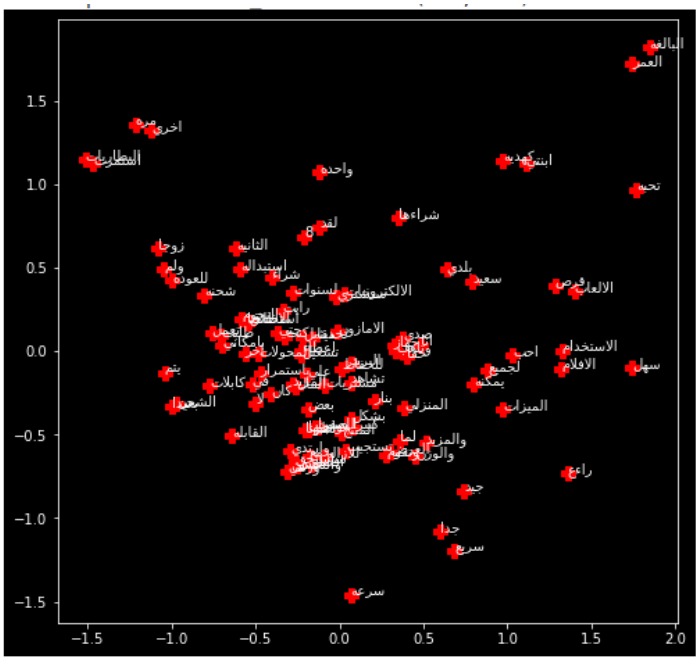

FIGURE 21 WORD2VEC GRAPH

Because of Word2Vec based on neural networks that require to know the maximum words of the bigger reviews in the dataset, then we need each review as a list of words. We define that number of features we need for each word and the window size is that for each word we need to extract its features from next and previous five words and we consider even words that are repeated for just one time. Then we start to train our Word2Vec model and make the matrix that we will use to train the model later.

- Word2Vec Code

```
- X = df_file['Arabic Reviews']
- y = df_file['polarity']
- text_list = list(X)
- text_list = text_list[:12000]
```

```
- max_len_str = max([len(i.split()) for i in text_list])
- text_list = [i.split() for i in text_list]
-
- number_of_features = 100
- window_size = 5
- min_words_count = 1
-
- def word_to_vec(text_list, number_of_features, window_size, min_word
  s_count):
-     '''
-     Argument:
-         list of strings and each string is list of words
-         size = extract 50 features for each word
-         window = 7 take the context of neighbour words
-         min_count = 1 consider each word that even repeated 1 time
-     return:
-         the word2vec model
-     '''
-     word_to_vec_model = Word2Vec(text_list, size =number_of_features
  , window = window_size, min_count=min_words_count, sg = 1)
-     print("Our word2vec model: ", word_to_vec_model)
-     print("The number of frequent words of our data: ", len(word_to_
  vec_model.wv.vocab)) # the frequent words
-     return word_to_vec_model
- word_to_vec_model = word_to_vec(text_list, number_of_features, windo
  w_size, min_words_count)
-
- def word_2_vec_matrix(text_list,word_to_vec_model,number_of_features
  , max_len_str):
-     '''
-     Argument:
-             List of string each of them is list of words
-             the word_to_vec_model model
-             number of features you apply to word2vec model
-             number of words of greatest string in your reviews
-     return:
-         embedding matrix that can apply to machine learning algorith
  ms
-     '''
```

```
-       print(len(text_list))
-       embedding_matrix = np.zeros((len(text_list), number_of_features*
    max_len_str)) # largest sentence and 5 fetures
-       print("The shape of matrix", embedding_matrix.shape)
-   #loop over each review
-       for index,review in enumerate(text_list):
-   # list of each reviw which will be appended to embedding matrix
-           one_sentence_list = []
-           for word in review:
-               word = word_to_vec_model[word]
-               one_sentence_list.extend(word)
-
-   # make padding for small strings
-           zero_pad = np.zeros(number_of_features*max_len_str-
    len(one_sentence_list))
-           zero_pad = list(zero_pad)
-
-   # apply the padding
-           one_sentence_list.extend(zero_pad)
-           embedding_matrix[index] = one_sentence_list
-       return embedding_matrix
-   embedding_matrix = word_2_vec_matrix(text_list,word_to_vec_model, nu
    mber_of_features, max_len_str)
```

5.4.9 Logistic Regression Model

In Word2Vec we used only Logistic regression.

- Logistic Regression Model With Word2Vec Code

```
-   clf_LogisticRegression = LogisticRegression(penalty='l2', tol=0.0000
    1, solver='liblinear',max_iter=1000)
-   logistic_model = clf_LogisticRegression.fit(X_train, y_train)
-   predict = logistic_model.predict(X_train)
-   print("F1 score of our testing data is: ", f1_score(y_train, predict
    , average='micro'))
```

```
- print("Evalution Matrix of training data is \n", confusion_matrix(y_
  train, predict))
- predict = logistic_model.predict(X_test)
- print("F1 score of our testing data is: ", f1_score(y_test, predict,
   average='micro'))
- print("Evalution Matrix of testing data is \n", confusion_matrix(y_t
  est, predict))
-
```

5.4.10 Comparison of the three features extraction

But because of the limited resource of our machine that can use the whole corpus to get more results I just use Google Colab with 12 GB of ram and even of this I cannot train except 12000 reviews from 26000 reviews. I think one of the things in future work is buying a host that can help me improve the result of prediction but as beta version application now I use TF-IDF as it learns more from 26000 reviews than Word2Vec and actually the thing that leads me to choose TF-IDF with Logistic Regression for the first Beta Version of the application is its result which was about 90% in training and 85% in testing and even Count-Vectorize gave me the more accurate result on testing which was 87% in training give me 95% and from this point of Overfitting I used TF-IDF even if it was also Overfitting but the ratio between training and testing is less as what we have in Count-Vectorizer.

5.5.1 Front-End & Back-End:

SBR Logo, Run Project & Welcome Page

After what we have discussed in all of these processes we end up with what the user can see now. The Sentiment behind reviews is designed to get products from online stores like Souq & Jumia to make analytical reviews of these products and predict the Sentiment of each review on the product (Positive Or Negative),

besides a pie chart that represents a quick overview of the ratio between positive and negative.
On the homepage, we introduce new visitors to our platform and give them a quick overview of what our platform is about testing their text as positive or negative.

FIGURE 22 APPLICATION LOGO

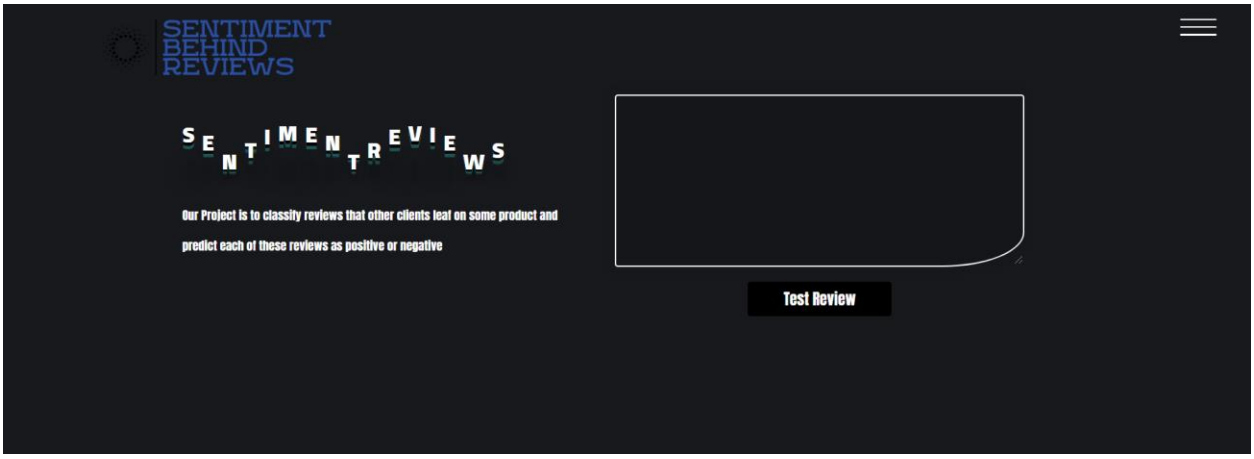

FIGURE 23 HOME PAGE VIEW

5.5.2 Signup , Login, Logout & contact Form

This Form allows users to make a new account if they aren't having old accounts he/she can make sign-up, If the user tries to enter invalid information or invalid match password, our signup validates all of the data that the user applies so we first start with a database that has no user except the super user which has access to the

dashboard of our project and can add another super user, product and control role of each one on the site.

Also we validate that the user does not leave any required inputs empty, also we ensure that once the form is wrong we still keep the user inputs data to complete other data not to start from the beginning.

Once the user has successfully signed up without any missing data that should be provided like e-mail and first name, we start to process and update the back end of our project and assign the new user to our User Model Class and redirect the user to the home page. Also we mentioned his first name in our Navbar and made processing a login for him.

FIGURE 24 SIGNUP FORM

The signup form is in a separate page that the user can access, but there are two different accesses of the form: the Get request which the user accesses in a general way via the URL but another access is via submitting the form with data which is converted to a POST request, and from this point, we analyze the data that the user enters and the first level of validation is via javascript, that process that the inputs data that are required are not empty. The second level is sending data via Ajax to Django Application of the views that process these data like if the user has no account with provided data or even if the passwords don't match then we return JSON request in the form of success or errors of some data and then we represent these errors to the user for each level of validation and once the form is filled successfully we redirect the user to our Home Page.

Signup Code

- Python Code

```
- def signup_form(request):
-     if request.method == 'POST':
-         form = SignUpForm(request.POST)
-         if form.is_valid():
-             user = form.save(commit=False)
-             user.email = form.cleaned_data.get('username')
-             form.save()
-             username = form.cleaned_data.get('username')
-             raw_password = form.cleaned_data.get('password1')
-             user = authenticate(username=username,
  password=raw_password)
-             login(request, user)
-             return redirect('home_page')
-         else:
-             return render(request,
  'sentiment_behind_reviews/signup.html', {'form': form})
-     else:
-         form = SignUpForm()
-     return render(request, 'sentiment_behind_reviews/signup.html',
  {'form': form})
-
```

- Ajax and JavaScript code

```
- $(".signup_page .signup_form .submit_form").click(function (e){
-       e.preventDefault();
-       var first_name    = $('.signup_page
  .signup_form').find('input[name="first_name"]').val(),
-       last_name = $('.signup_page
  .signup_form').find('input[name="last_name"]').val(),
-       username    = $('.signup_page
  .signup_form').find('input[name="username"]').val(),
```

```
-       password1= $('.signup_page
    .signup_form').find('input[name="password1"]').val(),
-       password2= $('.signup_page
    .signup_form').find('input[name="password2"]').val(),
-      csrfmiddlewaretoken=$(".signup_page
    .signup_form").find('input[name="csrfmiddlewaretoken"]').val();
-        $('.signup_page .signup_error').removeClass('alert alert-
    danger');
-         $.ajax({
-             url: '',
-             method: 'POST',
-             headers: {
-                 'X-CSRFToken': csrfmiddlewaretoken,
-             },
-             data: JSON.stringify({
-                  'first_name': first_name,
-                  'last_name': last_name,
-                  'username': username,
-                  'password1': password1,
-                  'password2': password2,
-        'csrfmiddlewaretoken': csrfmiddlewaretoken,
-
-                  }),
-             success: function (data) {
-                  if(data['errors']){
-                       if(!first_name || !username || !password1 ||
    !password2) {
-                            $('.signup_page
    .signup_error').append("Fill Your data please");
-                        $('.signup_page
    .signup_error').addClass('alert alert-danger');
-                       }else{
-                            $('.signup_page
    .signup_error').append("Invalid E-mail or password matching");
-                       }     $('.signup_page
    .signup_error').addClass('alert alert-danger');
-                  }else{
-                       window.location.href =
    "http://localhost:8000";
```

```
-                           }
-
-                    }
-              }) }
```

5.5.3 Log In Form

Now if the user has an account he/she can log in as we can see Ahmed has an account on our site So he can log in, but also it requires validation of unsigned users or empty inputs.

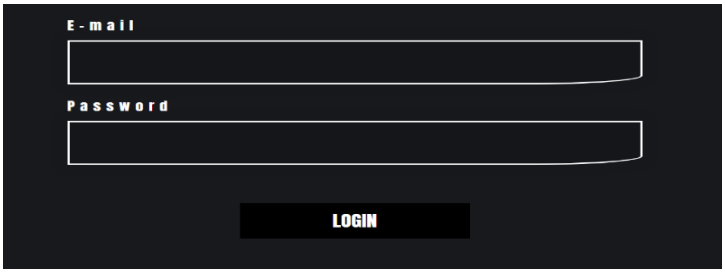

FIGURE 25 LOGIN FORM

Also, the login form is based on some validation that we have to handle as two separate steps of JavaScript: first that once some data are empty and the second is via Django which validate that return page if the user comes from GET request and validate the data if the user tries to make a POST Request and this should validate the user already having an account to log in with.

Log in Form Code

- Python Code

```
- def login_form(request):
-     if request.POST:
-         username = request.POST['username']
-         password = request.POST['password']
-         user = authenticate(username=username, password=password)
-         if user is not None:
-             if user.is_active:
-                 login(request, user)
-                 return redirect('home_page')
-         return render(request,
  'sentiment_behind_reviews/login.html')
-     return render(request, 'sentiment_behind_reviews/login.html')
```

- Ajax and JavaScript code

```
-
- $(".login_page .login_form .submit_login").click(function (e){
-             e.preventDefault();
-              var username    = $('.login_page
  .login_form').find('input[name="username"]').val(),
-              password    = $('.login_page
  .login_form').find('input[name="password"]').val(),
-           csrfmiddlewaretoken   = $(".login_page
  .login_form").find('input[name="csrfmiddlewaretoken"]').val();
-           $('.login_page .login_error').empty();
-           $('.login_page .login_error').removeClass('alert alert-
  danger');
-             $.ajax({
-             url: '',
-             method: 'POST',
-             headers: {
-                 'X-CSRFToken': csrfmiddlewaretoken,
-             },
-             data: JSON.stringify({
-                  'username': username,
```

```
-                   'password': password,
-
-                   }),
-               success: function (data) {
-
-                   if(data['username']){
-                        window.location.href =
    "http://localhost:8000";
-                   }else{
-                        if(!username || !password){
-                             $('.login_page
    .login_error').append("Please fill your data");
-                             $('.login_page
    .login_error').addClass('alert alert-danger');
-
-                        }else{
-                             $('.login_page
    .login_error').addClass('alert alert-danger');
-                             $('.login_page
    .login_error').append("You do not have an account with this e-
    mail");
-                        }
-                   }
-
-               }
-          })
```

5.5.4 Logout Form

As we can see if Ahmed is login so there is no signup or login form, so the logic is work as we need, and Ahmed has logout via logout button.

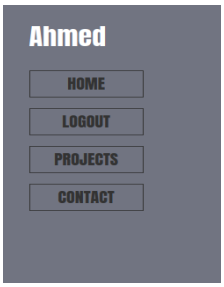

FIGURE 26 AFTER LOGIN

Logout Form code

- Python Code

```
def logout_form(request):
    logout(request)
    return redirect('home_page')
```

5.5.5 Contact US

Everyone has the freedom to say something about our platform or even ask about something or query that makes him keep going using our platform and inform about issues they face or even new suggestions. So we start to help them send us a form of their query and it works for all visitors even if those are not signed for our platform.

In Contact us form We distinguish between two types of users and visitors that have no account and for that we should get all of the information, but other users that have an account and already logged in to our platform. We then let him fill the phone and message of his query.

Our analysis also shows that the number of users entering is a valid number using regular expressions that help us validate that the user input is not empty or even text.

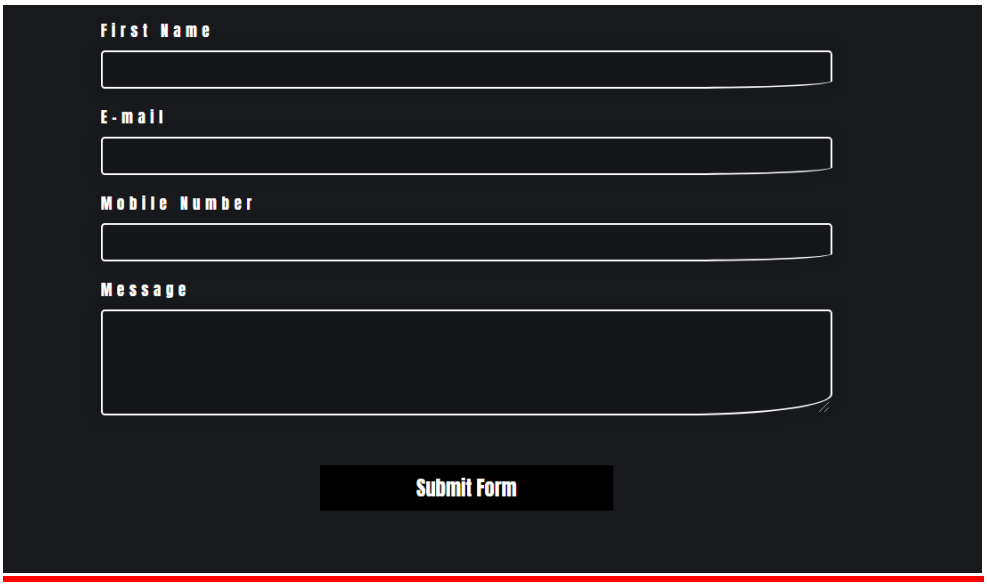

FIGURE 27 CONTACT US FPRM

Contact US Code

- Python Code Model

```
- class Contact_us(models.Model):
-     first_name = models.CharField(max_length=50)
-     mail = models.EmailField()
-     phonenumber = models.CharField(max_length=50)
-     message = models.CharField(max_length=2000)
-
-     class Meta:
-       verbose_name_plural = "contact_us"
-       db_table = "contact_us"


- def contact_form(request):
-     if request.method == "POST":
-         data = json.loads(request.body)
-         if request.user.is_authenticated:
```

```
-             data['mail'] = request.user.username
-             data['first_name'] = request.user.first_name
-         Contact_us.objects.create(
-             mail=data['mail'],
-             first_name=data['first_name'],
-             phonenumber=data['phonenumber'],
-             message=data['message']
-         )
-         return JsonResponse(data)
-     else:
-         return render(request,
  'sentiment_behind_reviews/contact_us.html')
```

- Ajax and JavaScript code

```
- $(".contact_us_page .contact_us_form
  .submit_contact").click(function (e){
-         e.preventDefault();
-          var message    = $('.contact_us_page
  .contact_us_form').find('textarea[name="message"]').val(),
-          phonenumber    = $('.contact_us_page
  .contact_us_form').find('input[name="phonenumber"]').val(),
-          mail    = $('.contact_us_page
  .contact_us_form').find('input[name="mail"]').val(),
-          first_name    = $('.contact_us_page
  .contact_us_form').find('input[name="first_name"]').val(),
-         csrfmiddlewaretoken   = $(".contact_us_page
  .contact_us_form").find('input[name="csrfmiddlewaretoken"]').val();
-         $(".contact_us_page .submit_contact_error").hide();
-         if(!message || !phonenumber){
-             $(this).before("<p class=\"submit_contact_error\">Please
  complete form data</p>");
```

```
-          }else{
-              var email_regex = /^[a-zA-Z0-9._-]+@[a-zA-Z0-9.-]+\.[a-
   zA-Z]{2,4}$/i,
-              phonenumber_regex = /[0-9]{11}/;
-              if(!email_regex.test(mail) && mail){
-              $(this).before("<p class=\"submit_contact_error\">this
   is not valid email</p>");
-              }else if(!phonenumber_regex.test(phonenumber) &&
   phonenumber){
-                   $(this).before("<p
   class=\"submit_contact_error\">This is not valid phone number valid
   number as 01116259370 with 11 number</p>");
-              }else{
-                   $.ajax({
-              url: '',
-              method: 'POST',
-              headers: {
-                  'X-CSRFToken': csrfmiddlewaretoken,
-              },
-              data: JSON.stringify({
-                  'message': message,
-                  'phonenumber': phonenumber,
-                  'mail': mail,
-                  'first_name': first_name,
-                  }),
-              success: function (data) {
-
-                 window.location.href = "http://localhost:8000";
-                  }
-              })
-
-              }
-          }
-
-          })
```

5.5.6 Navbar Animation

Navbar is what the user clicks to access another page and forward to what page he needs and for that, we make it with animation that helps the user for the great user interface.

- JavaScript code

```
- function scrolling_behavior(){
-       $("body").toggleClass("scrolling_behavior")
- }
- $('#nav-icon').click(function(){
-       scrolling_behavior();
-       var $this =  $(this);
- if(!clicked){
-            clicked = 1;
- $(".main-slider").toggleClass("go-back");
- $this.toggleClass('open');
- $this.next().toggleClass("animated");
- setTimeout(function(){
- clicked = 0; }, 900); }});
```

5.5.7 Product Models

After we have scraped the data from online stores we need to display these products in our web application but because of these data stored in Mongo Database and its in a form of each product as a document which is a dictionary with key and value but the process of connecting to this online Mongo database we assign the product to is too much process and this is another issue that ensures that we can go back a step and solve some issues and return to what we were before and that makes us use agile, so we start to implement our model for this product and for each product we make a foreign key with the related products. Then we extract all of these products as one big JSON file with all data and reviews for each product and make a function that loop over

this file and create the product in our database and for each product loop over its related reviews and assign to it all its reviews.
Also we ensure that the product we scraped and tried to assign to the model is all of the information required like URL that when the client is taking a look at the product reviews and get quick information about positive and negative it can move to the store directly and buy it.
Once we assign product successfully a new product is added to the database

Also we should mention what product we need to assign a new review to.

FIGURE 28 PRODUCT PAGE

- Product & Reviews Models Code

```
- class Products(models.Model):
- product_title=models.CharField(max_length=1000,verbose_name=u"Product_title",null=False, blank=False)
- product_url = models.URLField(null=False, blank=False)
```

```
- image_src = models.URLField(null=False, blank=False)
- product_new_price = models.DecimalField(decimal_places=2,
  max_digits=15)
- product_old_price = models.DecimalField(decimal_places=2,
  max_digits=15, default=0)
- product_discount_percentage = models.DecimalField(decimal_places=2,
  max_digits=15, default=0)
- product_discount_value=models.DecimalField(decimal_places=2,max_digi
  ts=15, default=0)
- class Meta:
- db_table = "products"
-
- class Review(models.Model):
- review=models.ForeignKey(Products,related_name='product_review',on_d
  elete=models.CASDE,)
-       product_review = models.CharField(max_length=1000)
```

Chapter 6:

Testing, Result & Evaluation

6.1 Result, Testing & Evaluation

The previous chapter we have mentioned above is all about the code and implementation process and different functions, but what about the result we get what about our application as we say it is responsive for all media and all users can access from Mobile phone, Tablet or Computer for all of that you can access our platform in simple design, also the Machine Learning Algorithms we use, the feature extraction techniques we use for text data, also what about the data itself is a raw data which needs more cleaning and discovery to make the process that can help us in the phase of extracting features and actually most of the time should be used for cleaning the data. All of that we will discuss in this chapter and end up with future work that we aim to achieve for the next time in the next chapter.

As we talked above from scraping to cleaning to feature extraction and machine learning models, then we end up with the web application. I would like to display results and evaluation of all of these processes as in the previous chapter.

6.2 Connect To Database & Scraping

We start with connecting to the mongo database and then run the scraping functions to get the product and for testing. If there are any issues that happen during the scraping we try to get the data and in the Exceptions we send the error to one folder with the function that the error happens in and also the file.

- Test after scraping one product

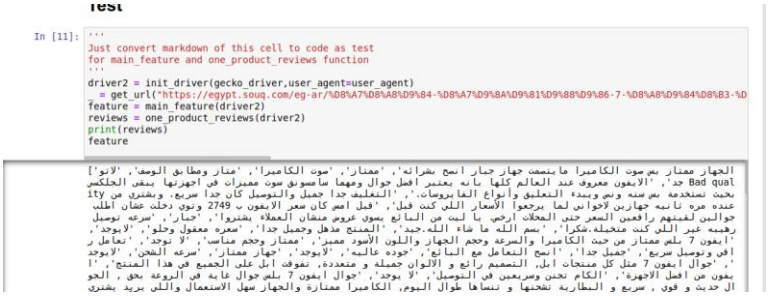

FIGURE 29 ONE PRODUCT SCRAP

- Main Features of the product

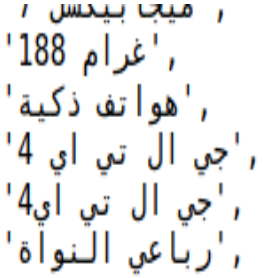

FIGURE 30 ONE PRODUCT MAIN FEATURES

- Log Error file when scraping cannot get some data

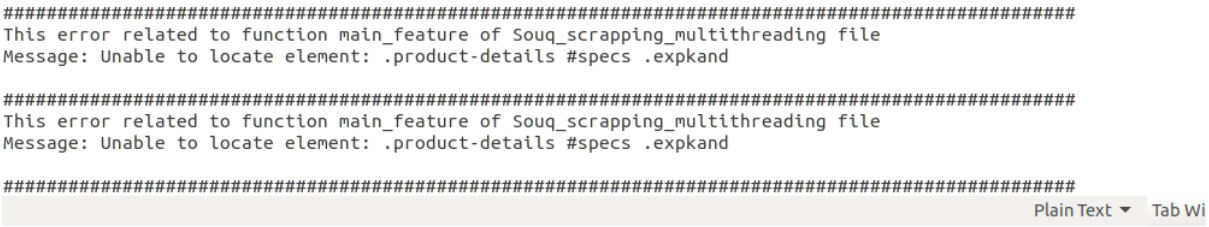

FIGURE 31 SCRAP ERROR1

6.3 Extract & Cleaning Reviews

Now after we scraped all of the data we need to connect again to database and get all of these products then we can extract the reviews of all products in one big list to make the process of cleaning and handling all of the requirements.

- Returned data from Mongo db

FIGURE 32 RETURN DATA FROM MONGO DB

- Export Arabic & English as separated two lists

```
In [7]: all_arabic_reviews, all_english_reviews = export_all_reviews(products)
        all_arabic_reviews[:10]

Out[7]: ['جهاز ممتاز و تسليم من سوق فوق الممتاز',
         'ممتاز شكرا لسوق كوم وللبائع وتوصيل سريع أيضا',
         'شيء خرافي',
         'ابداع جديد من سامسونج',
         'لايوجد',
         'افضل جهاز اندرويد بلا منازع',
         'الجهاز جيد جدا لكن تم شحنه بدون الهدية المجانية',
         'جوال أصلي ورائع وتوصيل سريع',
         'قمة الروعة وشكر سوق كوم علي سرعة التوصيل',
         'الهاتف اصلي']
```

FIGURE 33 EXPORT ARABIC & ENGLISH REVIEWS

- Cleaning data without lemmatization

FIGURE 34 CLEANING WITHOUT LEMMATIZATION

We can see that stop words like in third review are removed and we can compare results before cleaning and after cleaning.

- Cleaning data with lemmatization

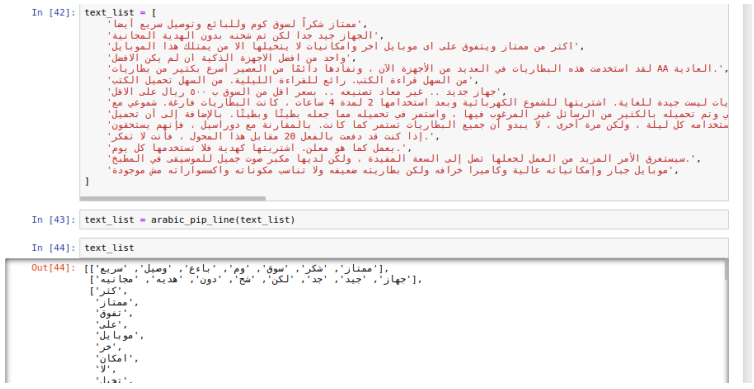

FIGURE 35 CLEANING WITH LEMMATIZATION

We can see the difference of adding lemmatization and removing this process from the pipeline. The lemmatization some time because of Arabic words leads to missing the word meaning because of its return the word to the root of it.
Also for errors that can be generated when we try to extract the Arabic and English reviews we send to the log folder the error related to which function and which file.

- Export reviews error

```
In [6]: def export_all_reviews(products):
            '''
            Argument:
                list of products each of them as object with key and value
            return:
                from this products we just need the reviews so
                Arabic Reviews
                English Reviews
            '''
            all_arabic_reviews = []
            all_english_reviews = []
        # Loop over products
            for indx,pro_val in enumerate(products):
        # Loop over reviews
                for review in pro_val['product_reviews']:
                    try:
                        char_check = review[0]
                        if char_check >= 'a' and char_check <= 'z' or char_check >= 'A' and char_check <= 'Z':
                            all_english_reviews.append(review)
                        else:
                            all_arabic_reviews.append(review)
                    except Exception as e:
        # send exception to log folder
                        logf = open("../logs_files/cleaning_reviews_error.log", "a+")
                        logf.write("This error related to function export_all_reviews of cleaning_data file\n"
                            + str(e) + "\n" + "#" *99 + "\n") # "#" *99 as separated lines
            return all_arabic_reviews, all_english_reviews
```

FIGURE 36 ERRORT REVIEWS

```
####################################################################################################
This error related to function export_all_reviews of cleaning_data file
string index out of range
####################################################################################################
This error related to function export_all_reviews of cleaning_data file
string index out of range
####################################################################################################
This error related to function export_all_reviews of cleaning_data file
string index out of range
####################################################################################################
```

FIGURE 37 CLEANING ERROR

6.4 Features Extraction & Model Evaluation

I have used F1 score to be the measurements analysis of the models and this because F1 Score is needed when we want to seek a balance between Precision and Recall.

Also because our problem is classification problem so tend to make balance between the two different class we have as Positive and negative.

The difference between measurements is that each of these algorithms tend to fit the data very well and the two models we used are close to each other, but I used at the end the TF-IDF with Logistic regression because even that Count Vectorizer gives a better result than it, but in real life, I think that TF-IDF will work better because of attention to rare Words that come in small time like other words that appear a lot in text, the other thing that I think of is the model result and the problem of overfitting and underfitting. The two models have a problem of overfitting which in training is doing well and get 90% and other 95% but with testing, we end up with overfitting that the model does well in training but not as well in testing and the last thing I deal with is the Word2Vec and this actually gives the best result is even 80% on testing but it works using neural networks to extract most of the features of the words and related and similar words to this words and this helps for machine learning Algorithms to know that words like [Orange and Apple] have the same weights as they used with the similar context, but because of its cost I end up using TF-IDF, the cost of that is in TF-IDF I can use over 26000 reviews and it can work without a problem while in Word2Vec the most important thing that I can use from the dataset to train and test is just 12000 which is not good because Machine learning model needs more data to learn more about the different cases of the data. Other things that Word2Vec is working better with other Algorithms like Neural Network actually for something that is called Recurrent Neural Network that I will use for future work.

- TF-IDF Result with Multinomial Naïve Bayes

```
clf_MultinomialNB = MultinomialNB()
model = clf_MultinomialNB.fit(training_data, y_train)
predict = model.predict(training_data)
print("F1 score of our training data is: ", f1_score(y_train, predict, average='micro'))
print("Evalution Matrix of training data is \n", confusion_matrix(y_train, predict))
predict = model.predict(testing_data)
print("F1 score of our testing data is: ", f1_score(y_test, predict, average='micro'))
print("Evalution Matrix of training data is \n", confusion_matrix(y_test, predict))
```

```
F1 score of our training data is:  0.8923033013033964
Evalution Matrix of training data is
 [[ 7836  1292]
 [  972 10922]]
F1 score of our testing data is:  0.8458904109589042
Evalution Matrix of training data is
 [[1814  433]
 [ 377 2632]]
```

FIGURE 38 TF-IDF RESULT WITH MULTINOMIAL

- TF-IDF Result with Logistic Regression

```
[14] clf_LogisticRegression = LogisticRegression(penalty='l2', solver='liblinear',max_iter=1000)
     logistic_model = clf_LogisticRegression.fit(training_data, y_train)
     predict = logistic_model.predict(training_data)
     print("F1 score of our testing data is: ", f1_score(y_train, predict, average='micro'))
     print("Evalution Matrix of training data is \n", confusion_matrix(y_train, predict))
     predict = logistic_model.predict(testing_data)
     print("F1 score of our testing data is: ", f1_score(y_test, predict, average='micro'))
     print("Evalution Matrix of training data is \n", confusion_matrix(y_test, predict))
```

```
F1 score of our testing data is:  0.9046237275235468
Evalution Matrix of training data is
 [[ 7900  1228]
 [  777 11117]]
F1 score of our testing data is:  0.8593987823439878
Evalution Matrix of training data is
 [[1829  418]
 [ 321 2688]]
```

FIGURE 39 TF-IDF RESULT WITH LOGISTIC REGRESSION

- Count Vectorizer with Multinomial Naïve Bayes

```
[16] clf_MultinomialNB = MultinomialNB()
     model = clf_MultinomialNB.fit(training_data, y_train)
     predict = model.predict(training_data)
     print("F1 score of our training data is: ", f1_score(y_train, predict, average='micro'))
     print("Evalution Matrix of training data is \n", confusion_matrix(y_train, predict))
     predict = model.predict(testing_data)
     print("F1 score of our testing data is: ", f1_score(y_test, predict, average='micro'))
     print("Evalution Matrix of training data is \n", confusion_matrix(y_test, predict))

     F1 score of our training data is:  0.8771287222909333
     Evalution Matrix of training data is
      [[ 7511  1551]
      [ 1032 10928]]
     F1 score of our testing data is:  0.835806697108067
     Evalution Matrix of training data is
      [[1812  501]
      [ 362 2581]]
```

FIGURE 40 COUNT VECTORIZER WITH MULTINOMIAL

- Count Vectorizer with Logistic Regression

```
[17] clf_LogisticRegression = LogisticRegression(penalty='l2', solver='liblinear',max_iter=1000)
     logistic_model = clf_LogisticRegression.fit(training_data, y_train)
     predict = logistic_model.predict(training_data)
     print("F1 score of our testing data is: ", f1_score(y_train, predict, average='micro'))
     print("Evalution Matrix of training data is \n", confusion_matrix(y_train, predict))
     predict = logistic_model.predict(testing_data)
     print("F1 score of our testing data is: ", f1_score(y_test, predict, average='micro'))
     print("Evalution Matrix of training data is \n", confusion_matrix(y_test, predict))

     F1 score of our testing data is:  0.9564266007040244
     Evalution Matrix of training data is
      [[ 8418   644]
      [  272 11688]]
     F1 score of our testing data is:  0.871765601217656
     Evalution Matrix of training data is
      [[1929  384]
      [ 290 2653]]
```

FIGURE 41 COUNT VECTORIZER WITH LOGISTIC REGRESSION

- Word2Vec with Logistic Regression

```
[19] clf_LogisticRegression = LogisticRegression(penalty='l2', tol=0.00001, solver='liblinear',max_iter=1000)
     logistic_model = clf_LogisticRegression.fit(X_train, y_train)
     predict = logistic_model.predict(X_train)
     print("F1 score of our testing data is: ", f1_score(y_train, predict, average='micro'))
     print("Evalution Matrix of training data is \n", confusion_matrix(y_train, predict))
     predict = logistic_model.predict(X_test)
     print("F1 score of our testing data is: ", f1_score(y_test, predict, average='micro'))
     print("Evalution Matrix of testing data is \n", confusion_matrix(y_test, predict))

F1 score of our testing data is:  0.883375
Evalution Matrix of training data is
 [[5741 1160]
 [ 706 8393]]
F1 score of our testing data is:  0.80975
Evalution Matrix of testing data is
 [[1325  412]
 [ 349 1914]]
```

FIGURE 42 WORD2VEC WITH LOGISTIC REGRESSION

Also we can see that Word2Vec gets the attention of similar words and we can see that after training our models and checking which words are similar to each other and also graph the result after making dimension reduction to 2D for some of the words of the dataset.

- Word2Vec with Similar words

```
[17] word_to_vec_model.most_similar('ممتاز')

/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:1: Deprecatio
  """Entry point for launching an IPython kernel.
/usr/local/lib/python3.6/dist-packages/gensim/matutils.py:737: FutureWarni
  if np.issubdtype(vec.dtype, np.int):
[('0.853896975517273', 'مميز'),
 ('0.8463045954704285', 'منافس'),
 ('0.8411762714385986', 'اصلي'),
 ('0.8382428884506226', 'وسرعه'),
 ('0.8316744565963745', 'جميل'),
 ('0.8225611448287964', 'حلو'),
 ('0.8182693719863892', 'واصلي'),
 ('0.8139151334762573', 'متميز'),
 ('0.813255250453949', 'والتوصيل'),
 ('0.8128101825714111', 'وسعره')]
```

FIGURE 43 SIMILAR WORD 1

```
[20] word_to_vec_model.most_similar('سيء')

/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:1: DeprecationWarning: Call to deprecated `most_similar` (Method will be remov
  """Entry point for launching an IPython kernel.
/usr/local/lib/python3.6/dist-packages/gensim/matutils.py:737: FutureWarning: Conversion of the second argument of issubdtype from `int` to
  if np.issubdtype(vec.dtype, np.int):
[('0.8173466324806213', 'قطيع'),
 ('0.8103188276290894', 'ممنوع'),
 ('0.8049452304840088', 'وصف'),
 ('0.802409291267395', 'بخس'),
 ('0.796673595905304', 'محبط'),
 ('0.7956256866455078', 'رديء'),
 ('0.7949675917625427', 'ضعيف'),
 ('0.7926079034805298', 'قمامة'),
 ('0.789840817451477', 'بئس'),
 ('0.7884291410446167', 'الرهيب')]
```

FIGURE 44 SIMILAR WORD 2

- Word2Vec dimension reduction

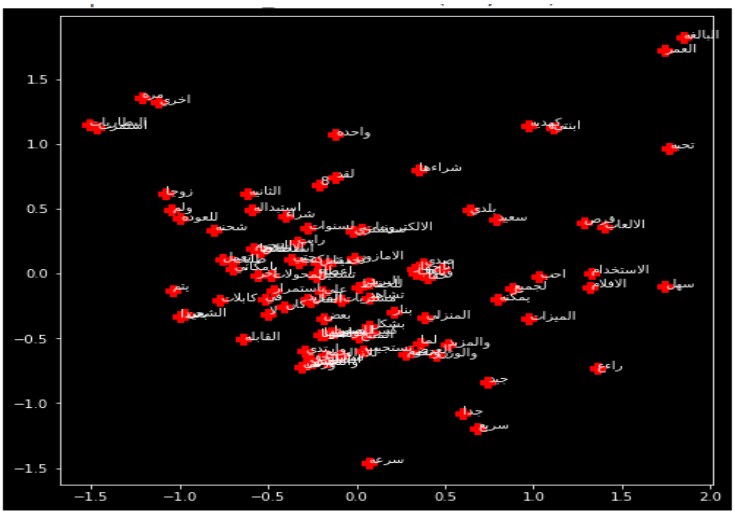

FIGURE 45 WORD2VEC GRAPH

After we have get like these result we have saved our model weight to use in the web application but we first test the models work with other unseen data to ensure that our model can work for real life application.

- Testing Unseen Data with TF-IDF

```
[42] test_radnom_reviews = tfidf_vectorizer.transform(test_radnom_reviews)
     test_radnom_reviews  = test_radnom_reviews.toarray()
     predict = logistic_model.predict(test_radnom_reviews)
     print(predict)
     poistive_nagative = []
     for i in predict:
       if i:
         poistive_nagative.append("Positive")
       else: poistive_nagative.append("Negative")
     poistive_nagative

[1 1 1 1 1 1 0 0 0 0 0 1 0 0]
['Positive',
 'Positive',
 'Positive',
 'Positive',
 'Positive',
 'Positive',
 'Negative',
 'Negative',
 'Negative',
 'Negative',
 'Negative',
 'Positive',
 'Negative',
 'Negative']
```

FIGURE 46 PREDICT RESULT 1

- Testing Unseen Data with Count Vecotrizer

```
[19] test_radnom_reviews = count_vectorizer.transform(test_radnom_reviews)
     test_radnom_reviews  = test_radnom_reviews.toarray()
     predict = logistic_model.predict(test_radnom_reviews)
     print(predict)
     poistive_nagative = []
     for i in predict:
       if i:
         poistive_nagative.append("Positive")
       else: poistive_nagative.append("Negative")
     poistive_nagative

[1 1 1 1 1 1 0 0 0 0 0 1 0 0]
['Positive',
 'Positive',
 'Positive',
 'Positive',
 'Positive',
 'Positive',
 'Negative',
 'Negative',
 'Negative',
 'Negative',
 'Negative',
 'Positive',
 'Negative',
 'Negative']
```

FIGURE 47 PREDICT RESULT 2

- All weights In one Table

Features Extraction	Model	F-1Training Result Score	F-1Testing Result Score
TF-IDF	Logistic Regression	90%	85%
TF-IDF	Multinomial Naïve Bayes	89%	84%
Count Vectorizer	Logistic Regression	95%	87%
Count Vectorizer	Multinomial Naïve Bayes	87%	83%
Word2Vec	Logistic Regression	88%	80%

6.5 Front & Back-End Testing

6.5.1 Signup Testing

The signup form that is available for the user to make a new account has different tests that prevent users from submitting the form before filling all of the required information, and also check password matching and email validation.

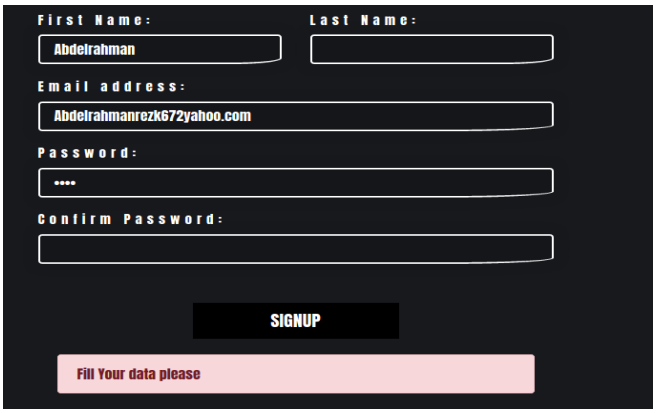

FIGURE 48 SIGNUP ERROR 1

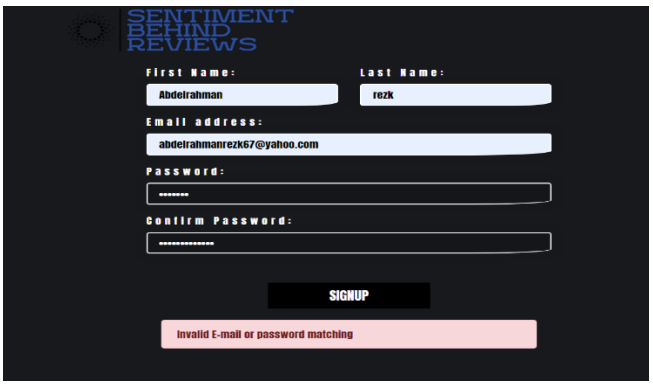

FIGURE 49 SIGNUP ERROR 2

We also ensure that our signed user has added to our backend database and once the user has signed up without any issues we make a login for him to see his/her name on our Navbar and we can see that new user added after successfully signing up, and we redirect the user to the home page.

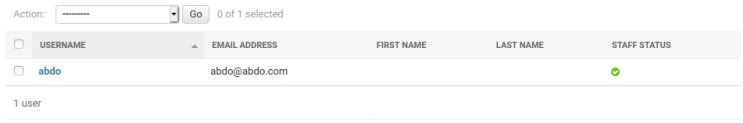

FIGURE 50 SIGNUP ONE USER

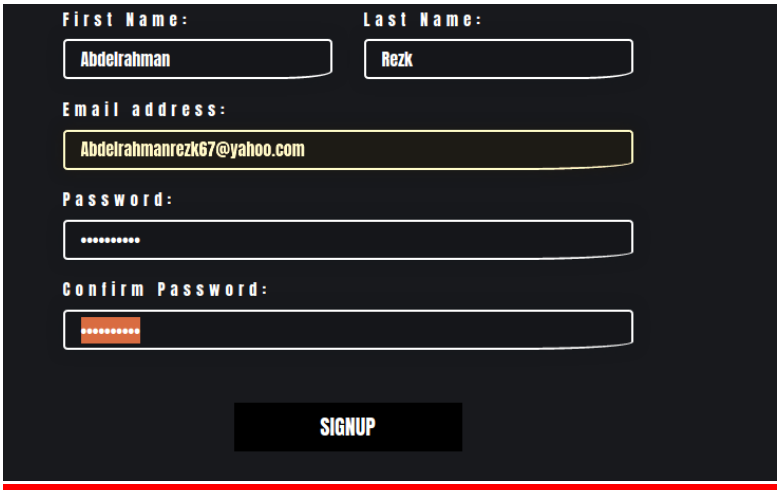

FIGURE 51 SIGNUP NEW USER

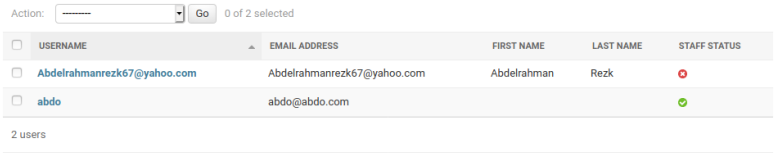

FIGURE 52 SIGNUP ADDED USER

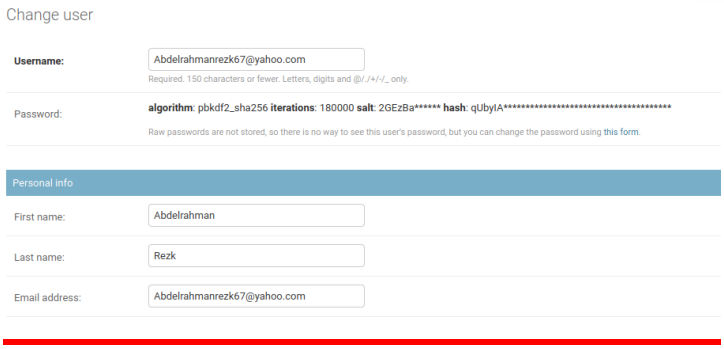

FIGURE 53 USER AFTER SIGNUP1

FIGURE 54 USER AFTER SIGNUP2

6.5.2 Login Testing

Once the user has an account he/she can log in, then we validate their email and password and ensure that they already have an account or not.

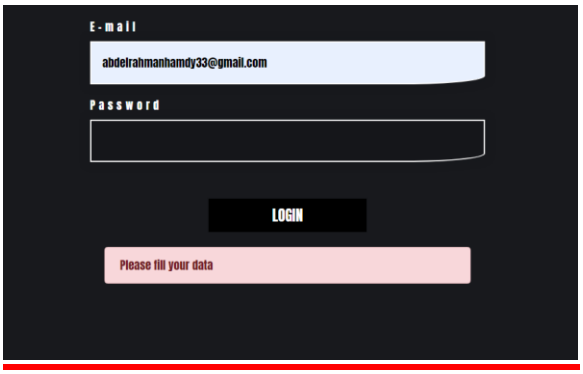

FIGURE 55 USER LGOIN ERROR1

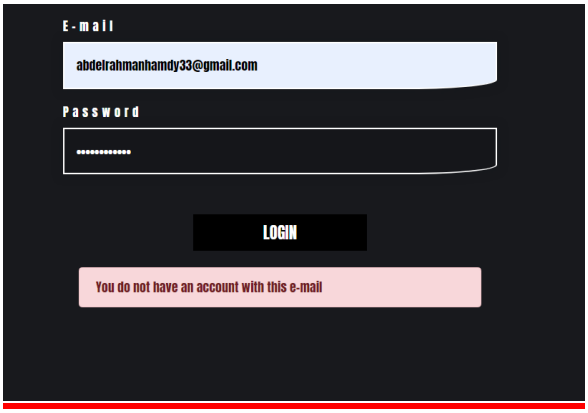

FIGURE 56 USER LOGIN ERROR2

If user login successfully then the result will be

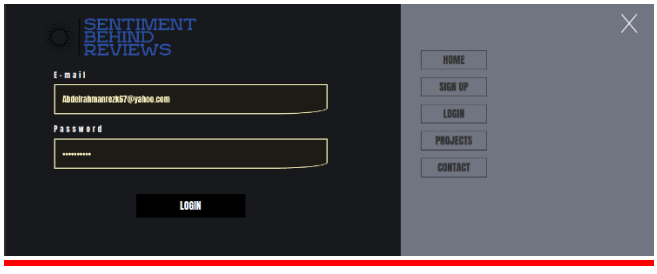

FIGURE 57 USER LOGIN SUCCESS

FIGURE 58 AFTER LOGIN

6.5.3 Contact Us Testing

Users can send us a query and if the user is already enrolled we just take the query and number else they require to enter their e-mail and names.

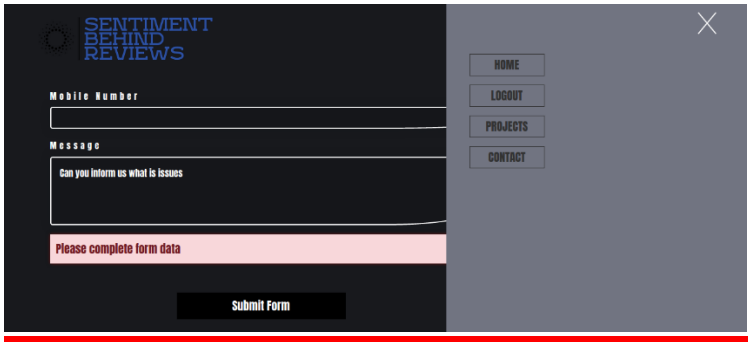

FIGURE 59 CONTACT US ERROR1

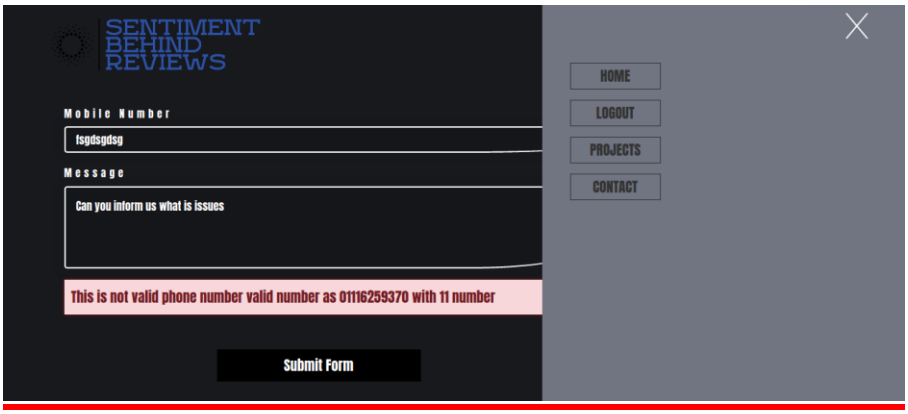

FIGURE 60 CONTACT US ERROR2

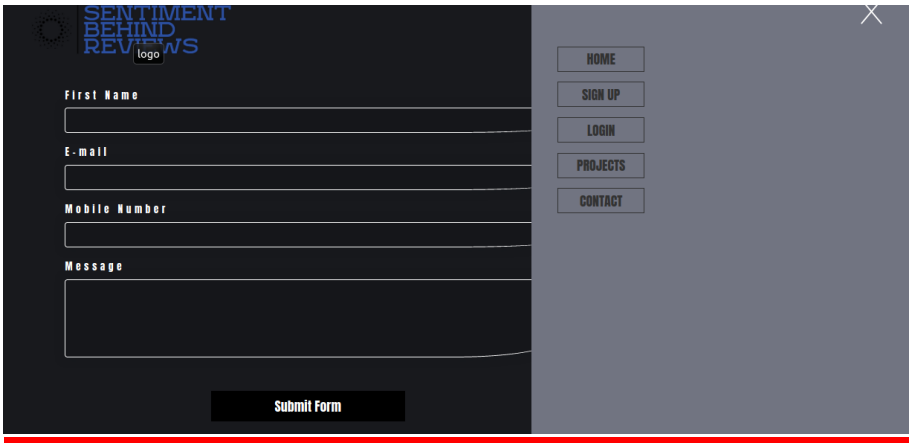

FIGURE 61 CONTACT US UNSINED USER

Also, we ensure that the form added to the database if they submitted without any errors,

And we can see that the user who has an account required to enter just their number and the query and in the backend we get their e-mail and names to complete the data and add a new form to our database.

FIGURE 62 CONTACT US SUCCESS

FIGURE 63 EMPTY CONTACT FORM

FIGURE 64 ADD CONTACT FORM

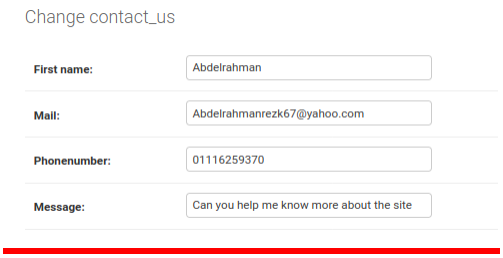

FIGURE 65 USER CONTACT FORM DATA

6.5.4 Product & Reviews Testing

The super user of the dashboard can add new products and each product can add reviews but this requires him to specify the product which needs to add the review.

Also we validate that all the required data are provided.

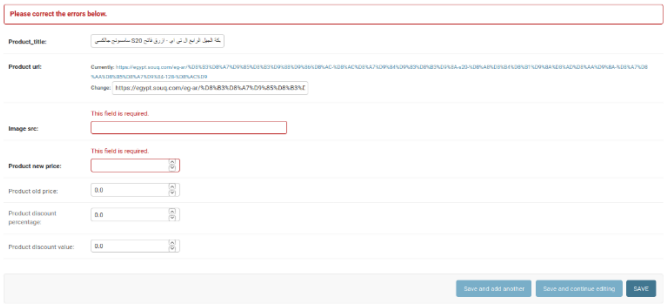

FIGURE 66 ADD PRODUCT ERROR1

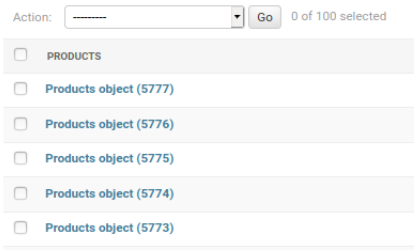

FIGURE 67 BEFORE ADD PRODUCT

FIGURE 68 AFTER ADD PRODUCT

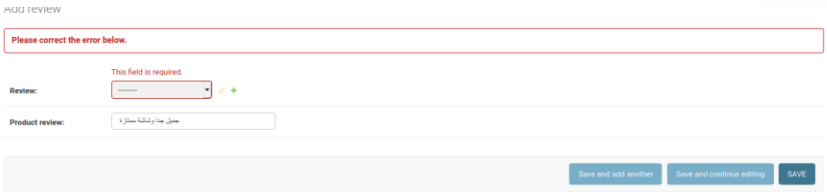

FIGURE 69 REVIEW ADD ERROR

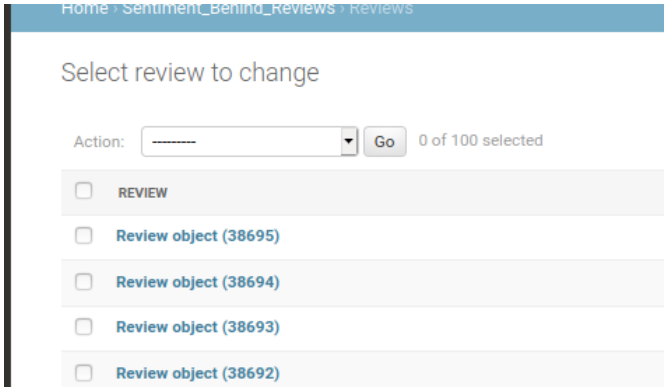

FIGURE 70 BEFORE ADD REVIEW

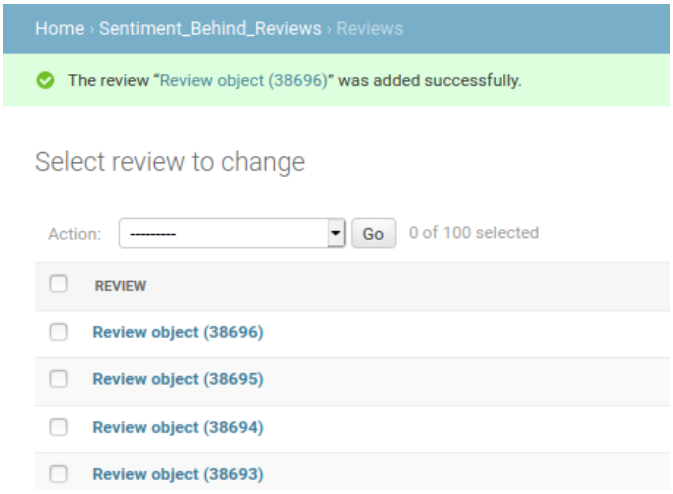

FIGURE 71 AFTER ADD REVIEW

FIGURE 72 PRODUCT PAGE

FIGURE 73 REVIEWS PAGE

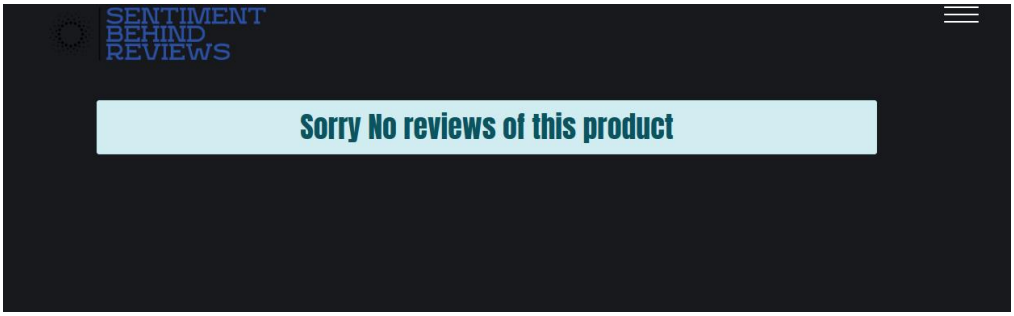

FIGURE 74 NO REVIEWS OF PRODUCT

6.6 The Web Application

In this section, I would like to present some different images for ensuring that our site works for different screens from computers to laptops, tablets and mobile phones so all users from any machine can connect and use our application.

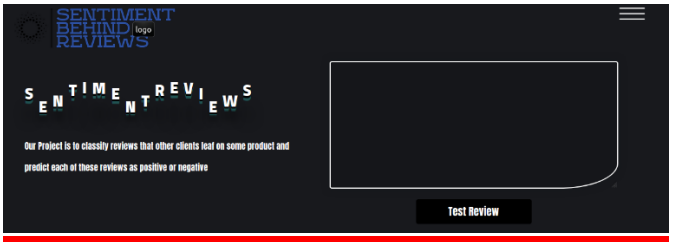

FIGURE 75 HOME PAGE LAPTOP

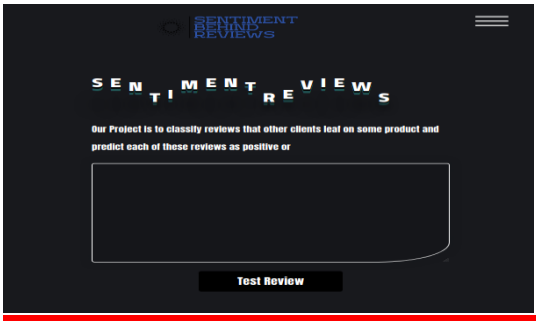

FIGURE 76 HOME PAGE TAP

FIGURE 77 HOME PAGE MOBILE

FIGURE 78 SIGNUP PAGE LAPTOP

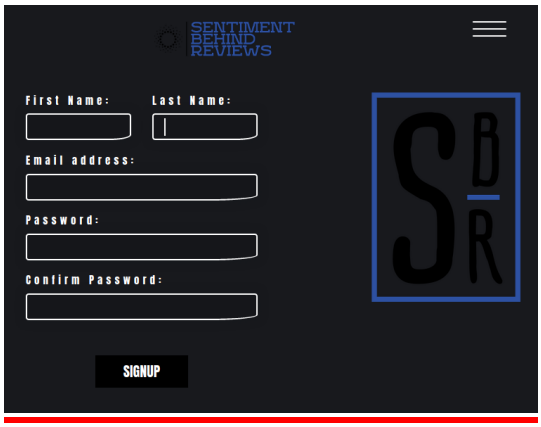

FIGURE 79 SIGNUP PAGE TAP

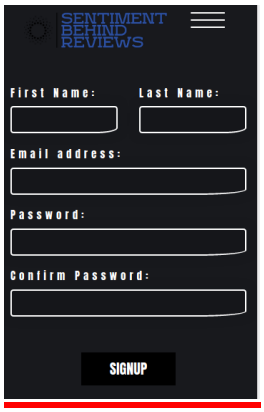

FIGURE 80 SIGNUP PAGE MOBILE

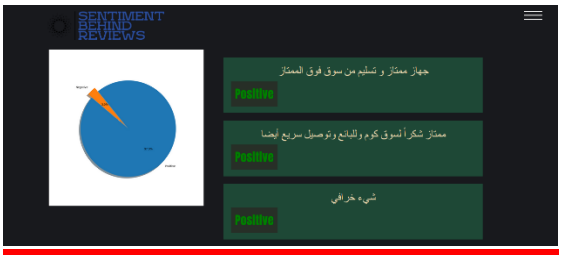

FIGURE 81 REVIEWS PAGE LAPTOP

FIGURE 82 REVIEWS PAGE TAP

FIGURE 83 REVIEWS PAGE MOBILE

Chapter 7:

Conclusion and future work

7.1 Summary

The Sentiment behind reviews aims to let the user get a quick overview of the product they tend to buy. This view is about how other clients say about the product. Besides that they show a Piegraph that displays how they review related to each other from positive to negative, also it helps the user to test their review before providing the product. Some clients tend to leave a positive review, but their words are mixed with some negative, so we test their reviews to be positive as they tend to leave the product. From our application client can take an action of buying the product or not, our client has the ability to assign and send us new features for our application to be added. The Sentiment behind reviews also is fully responsive to all media, and it's based on how the new technology of Machine Learning & Natural Language Processing can give us the new and different appearance of the technology, how new technology of Machine Learning & Natural Language Processing can give us the new added and different appearance of the technology when it's mixed with artificial intelligence, can increase the clients buying or even help others get a quick view of their product.

The aims can be showed as:

- Extract people's opinions, sentiments, and subjectivity from the reviews
- Allow corporation to looks at their customer intuition, and this will help those keeping changes.
- Improve marketing campaigns and product messaging
- Improve customer experience
- Determine brand reputation
- Based on product reviews showing the result of reviews by graphs.
- Provide an accurate sentiment analysis results, a trust system give clients to use it.

- Gives insight into the emotion behind the words, how these word affected your business on the way.
- What customers like about the corporation.
- Represent weakness and strength of their products help them improving their products.
- Ensure that client can contact us from our form for new features they need
- Client can test the reviews they provide on online stores
- Clients are able to sign up and use our platform for free

7.2 Future Work

The sentiment behind reviews are built actually to be online platform and actually it have been deployed on https://sentiment-behind-reviews.herokuapp.com/ and client can now get the intuition of the product but for some of user experience it will added new features and enhanced more for the Sentiment of the reviews using different Machine Learning & Natural Language Processing Models like it will improve using Deep Learning Neural Network by the model called Bi-dircation Recurrent Neural Network, also the features will be extracted using other features techniques that based on Neural network like BERT and Google Universe word Embedding which based not on the words count like what we have done, it's based on chars level and the context itself, Bi-direction model using these features extraction can learn the different between sentence by forward and backward because the meaning of sentence can be found not from previous words but also from next words so it works in the direction and these features extraction help in char level which can help for unknown words that maybe other user can write.

REFERENCES

Amira Thabet, A. H. M. E. S., 2011. *Requirements Specification Document* , Cairo: AUC.

Anon.,n.d.*explore-history-machine-learning.*[Online]
Availableat:https://cloud.withgoogle.com/build/data-analytics/explore-history-machine-learning/
[Accessed 11 2019].

Guinn,J.,2019.*software-implementation-plan.*[Online]
Availableat:https://www.softwareadvice.com/resources/software-implementation-plan/
[Accessed 5 2020].

Ibanez,L.,2017.*a-quick-overview-to-agile.*[Online]
Availableat:https://medium.com/theagilemanager/a-quick-overview-to-agile-5c87ffc9e0f2
[Accessed 5 2020].

Jones, R., 2018. *Change, Strategy and Projects at Work.* UK: s.n.

Jurafsky,D.,2013.*SoftwareImplementation.*[Online]
Availableat:https://web.stanford.edu/class/cs124/lec/sentiment.pdf
[Accessed 5 2020].

Justin B. Hollander, E. G. H. R. C. F.-K. A. W. D. D., 2016. *Urban Social Listening.* s.l.:s.n.

Rolland, M. F. a. E., 2016. *Fundamentals of Sentiment Analysis and Its Applications,* California: researchgate.

Ruder,S.,2018.*a-review-of-the-recent-history-of-natural-language-processing.* [Online]
Availableat:https://blog.aylien.com/a-review-of-the-recent-history-of-natural-language-processing/
[Accessed 11 2019].

Sacolick, I., 2020. *What is agile methodology? Modern software development explained.* [Online]
Availableat:https://www.infoworld.com/article/3237508/what-is-agile-methodology-modern-software-development-explained.html
[Accessed 5 2020].

ScottSims,B.,2015.*sentiment-analysis-101.*[Online]
Availableat:https://www.kdnuggets.com/2015/12/sentiment-analysis-101.html
[Accessed 11 2020].

Stemmler, K., 2019. *How to Learn Software Design and Architecture - aRoadmap.* [Online] Availableat:https://www.freecodecamp.org/news/software-design/ [Accessed 5 2020].

Stemmler,K.,2019.*software-design.*[Online] Availableat:https://www.freecodecamp.org/news/software-design/ [Accessed 5 2020].

Stemmler,K.,2019.*software-design.*[Online] Availableat:https://www.freecodecamp.org/news/software-design/ [Accessed 5 2020].

Trackers,R.,n.d.*online-reviews-survey.*[Online] Availableat:https://www.reviewtrackers.com/reports/online-reviews-survey/ [Accessed 11 2019].

Online tools

https://www.draw.io/

https://colab.research.google.com/