ARTIFICIAL INTELLIGENCE

LAB PROJECT SUBMISSION

PROJECT NAME : **REVIEWVIZ**
SUBMITTED TO : **DR. VARUN SRIVASTAVA**

SUBMITTED BY :
TEAM - BITS, Please!
SHOURYA DE - 102116106
LEENA GUPTA - 102116115
ADITI BINJOLA - 102116099
SAMARTH KAPOOR - 102116103

INTRODUCTION

AIM OF THE PROJECT

The aim of this project is to develop an AI system that can extract Amazon reviews from a given product link, perform sentiment analysis using Natural Language Processing (NLP), and categorize the words used in the reviews. The output of this analysis will be used to generate a word cloud, which will provide an easily digestible visualization of the features described in the reviews.

The proposed system will use NLP algorithms to analyse the reviews and determine their sentiment. It will categorize the words used in the reviews into positive, negative, and neutral sentiments. The system will then use these categorized words to generate a word cloud that will display the most positive and negative features in a visually appealing way.

The system has the potential to be used by businesses to gain insights into customer sentiment and identify areas for improvement in their products or services. The word cloud generated by the system will help businesses to quickly identify the most common themes in customer feedback and take appropriate actions to address them.

Overall, this project aims to demonstrate the potential of AI in analysing customer feedback and providing valuable insights to businesses.

LITERATURE SURVEY

Sentiment analysis is an important field of research in Natural Language Processing (NLP) that deals with the automatic identification and classification of subjective information present in text data. With the growth of online shopping, reviews and ratings have become an important source of information for customers to make informed purchasing decisions. Analysing these reviews can help businesses gain insights into customer preferences and identify areas for improvement in their products or services. The following is a brief survey of the relevant literature on sentiment analysis and its applications in analysing online reviews.

In their study, Hu and Liu (2004) proposed a lexicon-based approach to sentiment analysis, where they used a sentiment lexicon to determine the polarity of the words in the text data. They demonstrated the effectiveness of their approach in analysing film reviews and predicting their sentiment.

In a more recent study, Wang et al. (2021) proposed a hybrid approach to sentiment analysis that combines deep learning and lexicon-based methods. They used a neural network to learn the sentiment of words and phrases and combined it with a sentiment lexicon to classify the sentiment of the text data.

Word cloud generation is a popular visualization technique used to represent text data. In their study, Mehta and Sharma (2017) proposed a method for generating word clouds from Twitter data using sentiment analysis. They used sentiment analysis to categorize the words into positive, negative, and neutral sentiments and used the frequency of these words to generate the word cloud.

In summary, sentiment analysis and word cloud generation have been widely used in analysing online reviews and providing insights to businesses. The literature survey highlights the different approaches used in sentiment analysis and the potential of combining it with word cloud generation for effective visualization of text data.

METHODOLOGY

DATASET

The VADER lexicon is a pre-built rule-based approach to sentiment analysis that uses a combination of sentiment lexicons and grammatical rules to determine the sentiment of a given text. Since the VADER lexicon is pre-built and does not require any additional training on a specific dataset, there is no need to specify a dataset used for training.

ALGORITHM

NLP Model-NLP is an AI technique that enables machines and devices to comprehend and analyse human languages. The system uses NLP techniques to extract positive and negative reviews for word cloud generation.

Data Preprocessing- It is a crucial step in natural language processing (NLP) that involves cleaning and transforming raw text data into a format that can be easily analysed by machine learning algorithms. It includes:
1. Text cleaning: Remove any irrelevant information such as special characters, punctuation marks, and numbers from the reviews.
2. Tokenization: Break down the text into individual words or tokens using a tokenizer such as NLTK (Natural Language Toolkit).
3. Stop words removal: Remove common words that do not carry much meaning such as "the", "a", "and", etc. These words are known as stop words and can be removed using a list of stop words from a library such as NLTK.

RAKE Model-The RAKE algorithm is a keyword extraction algorithm that ranks candidate keywords by their co-occurrence with other words in the text. The algorithm first splits the input text into individual words, and then generates candidate keywords by combining adjacent words in the text. It then scores each candidate keyword based on its frequency and degree of co-occurrence with other words in the text.

POS Tagging- POS (part-of-speech) tagging is the process of marking up the words in a text corpus with their corresponding parts of speech. It first checks if a word is a stop word and whether its POS tag is an adjective. If the word is not a stop word and its POS tag is an adjective, the function checks if the next word in the text is a noun. If so, the function appends the two words as an adjective-noun pair to a list.

VADER Sentiment Analysis –(Valence Aware Dictionary and Sentiment Reasoner) Sentiment Analysis is the process of classifying the emotional intent of the text. The VADER model is a rule-based sentiment analysis tool that uses a lexicon of words and their valence scores to determine the sentiment of a given text.

STEPS OF ALGORITHM

Input: User reviews for a product.

Output: An array of 60 words describing the product's characteristics according to the reviews.

1. Import the necessary libraries and modules, including nltk, pandas, VADER, RAKE etc

```
# importing the required libraries

import pandas as pd
import re
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.corpus import wordnet
from nltk.sentiment.vader import SentimentIntensityAnalyzer
from rake_nltk import Rake
from collections import defaultdict
```

2. Load the dataset from the backend.

```
# reading dataset (in 'array')
# communicates with the backend
```

```
# ratings in the form of: 4.0 out of 5 stars, hence extracting only required rating rather than the complete sentence

for i in range(len(array)):
    array[i][0] = array[i][0][0]
```

```
# converting the dataset into pandas dataframe

data=pd.DataFrame(array, columns=['rating','text'])
```

3. Calculate the rating's count.

```
# calculating the ratings' count

rating={'1':0,'2':0,'3':0,'4':0,'5':0}
rating_order=data['rating'].value_counts().to_dict()
rating.update(rating_order
```

4. Check if the data contains some null values and fill them accordingly.

```
# checking if there are any text rows with null value

print(data[data['text'].isnull()])  # rows with null values
print(data[data['text'].apply(lambda x: not isinstance(x, str))])  # rows with non-string values
```

```
#filling the null values

data['text'] = data['text'].fillna('')  # replace null values with empty string
data['text'] = data['text'].apply(lambda x: '' if not isinstance(x, str) else x)  # replace non-string values with empty string
```

4. Apply Rake model for keyword extraction

```
# extracting keywords using RAKE

r = Rake(min_length=3, max_length=4,stopwords=stopwords.words('english'),
    punctuations = [')','(',',',':','),',').','.'])
def extract_keywords(text):
    r.extract_keywords_from_text(text)
    return r.get_ranked_phrases()

data['keywords'] = data['text'].apply(extract_keywords)
```

5. Preprocess the data

```
# cleaning the data

def clean(text):
    text = re.sub('[^a-zA-Z0-9]', ' ', text) # Removes special characters
    text = text.lower() # Converts to lowercase
    text = re.sub('\s+', ' ', text) # Remove extra whitespace
    return text

data['Cleaned Reviews'] = data['text'].apply(clean)
```

```
# tokenization

def tokenize_text(text):
    tokens = nltk.word_tokenize(text)
    return tokens
```

```
# POS tagging

def pos_tag(tokens):
    tagged_tokens = nltk.pos_tag(tokens)
    return tagged_tokens
```

6. Apply POS tagging to the cleaned reviews to extract keywords consisting of adjectives which are followed by a noun.

```
# extracting keywords from the Cleaned Reviews such that the adjective is followed by a noun

pos_dict = {'J': 'A'}

def token_stop_adjectives(text):
    tags = pos_tag(tokenize_text(text))
    newlist = []
    for i in range(len(tags)-1):
        word, tag = tags[i]
        if word not in set(stopwords.words('english')) and tag[0] in pos_dict:
            if tags[i+1][1][0] == 'N':
                newlist.append((word + ' ' + tags[i+1][0]))
        elif word not in set(stopwords.words('english')) and tag[0] not in pos_dict:
            continue
    return newlist

data['POS tagged'] = data['Cleaned Reviews'].apply(token_stop_adjectives)
data.head()
```

7. Score each word in the text data using the VADER sentiment analysis model. The VADER model assigns positive, negative, and neutral scores to each word. Sort the words on the basis of their compound score to receive lists consisting of positive and negative features.
Words with positive sentiment have compound score>0.5
Words with negative sentiment have compound score<-0.5
Words with neutral sentiment have compound score ranging between [-0.5,0.5]

```
sia=SentimentIntensityAnalyzer()
```

```
# performing Sentimental Analysis

word_scores = defaultdict(list)

# calculating the sentiment scores for each word in each sentence
for sentence in data['POS tagged']:
    for word in sentence:
        scores = sia.polarity_scores(word)
        word_scores[word].append(scores['compound'])

for sentence in data['keywords']:
    for word in sentence:
        scores = sia.polarity_scores(word)
        word_scores[word].append(scores['compound'])

# sorting the words by compound score
positive_words = sorted(word_scores, key=lambda w: max(word_scores[w]), reverse=True)[:50]
negative_words = sorted(word_scores, key=lambda w: min(word_scores[w]))[:50]
```

8. Calculate the number of positive and negative features to be returned on the basis of overall ratings of the product.

```
# algorithm to calculate the number of positive and negative words to be returned on the basis of the Ratings

pos={'1': 10, '2': 8, '3': 6, '4': 2,'5': 1}
neg={'1': 2, '2': 4, '3': 6, '4': 8,'5': 10}

count_pos=0
count_neg=0

for key in rating:
    count_pos += round(rating[key]/pos[key])
    count_neg += round(rating[key]/neg[key])
```

9. Generate an array consisting of both the positive and negative features.

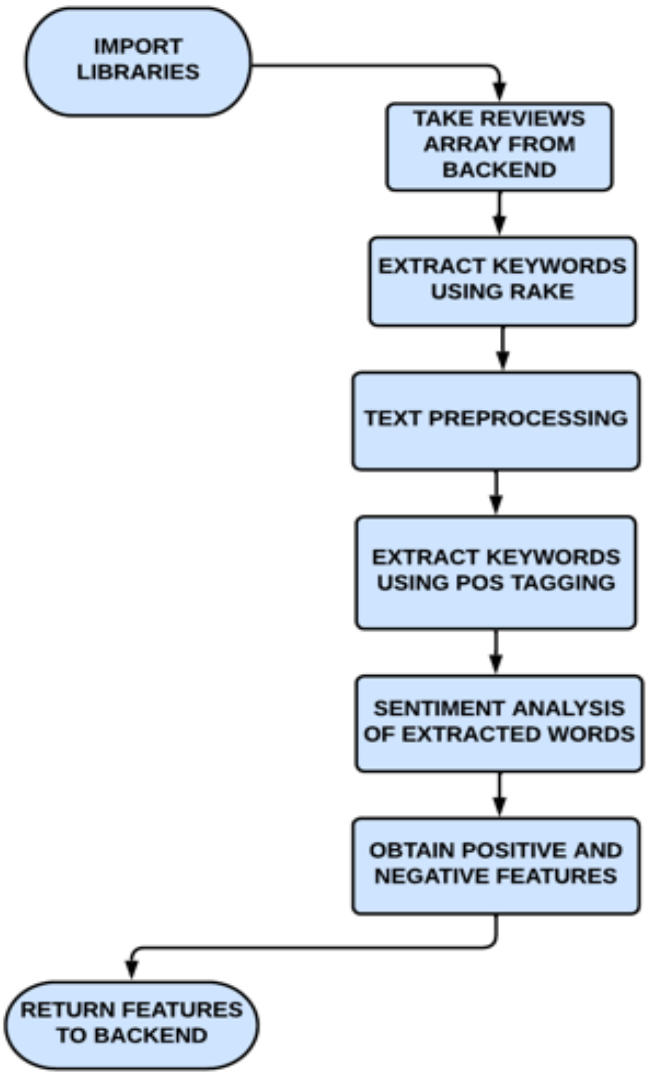
```
# generating an array of product's features

p=0
n=0
returnwords=[]
for word in positive_words:
    if(p<count_pos and max(word_scores[word])>0.5 ):
        returnwords.append({'text':clean(word), 'value':round(max(word_scores[word])*100,2)})
        p=p+1
    else:
        break

for word in negative_words:
    if(n<count_neg):
        returnwords.append({'text':clean(word), 'value':round(min(word_scores[word])*100,2)})
        n=n+1
    else:
        break

returnwords
```

FLOWCHART



RESULTS

The proposed system was developed using Python and the following libraries: Beautiful Soup for web scraping, NLTK for natural language processing, Vader for Sentiment Analysis. The system was tested on Amazon reviews for different products, and the results were analysed to determine the effectiveness of the sentiment analysis and word cloud generation.

The sentiment analysis algorithm used in the system was able to accurately categorize the reviews into positive, negative, and neutral sentiments. The results showed that the system was able to effectively analyse customer feedback and provide insights into the most common themes in the reviews.

FUTURE SCOPE

The proposed system has the potential to be used by businesses to gain insights into customer sentiment and identify areas for improvement in their products or services. The following are some potential future directions for this project:

- Integrating the system with a recommendation engine to provide personalized product recommendations based on customer feedback.
- Developing an interactive dashboard that allows businesses to visualize and analyse customer feedback in real-time.
- Expanding the system to analyse reviews from multiple sources, including social media platforms and other e-commerce websites.
- Incorporating machine learning algorithms to improve the accuracy of the sentiment analysis and categorization.
- Providing automated responses to customer feedback, based on the sentiment analysis, to improve customer engagement and satisfaction.

In conclusion, the proposed system has demonstrated the potential of AI in analysing customer feedback and providing valuable insights to businesses. There is a wide scope for further development and refinement of the system to improve its accuracy and effectiveness in analysing customer feedback.