(CSC-441) Natural Language Processing

Complex Computing Problem

Assignment 03

COMPUTER SCIENCE DEPARTMENT

(ARTIFICIAL INTELLIGENCE)

BAHRIA UNIVERSITY KARACHI CAMPUS

NAME	**ENROLMENT**
Muneeza Iftikhar	02-136212-012
Hafsa Hafeez Siddiqui	02-136212-026
Aqsa Khan	02-136212-039

SUBMITTED TO: Salas Akbar

Spring 2024

Contents

Introduction 4
Literature Review 4
Aspect Level Sentiment Analysis Approaches 4
Aspect Based Sentiment Analysis and Opinion Mining on Twitter Data Set Using Linguistic Rules (Sanjay Tanwani) 5
Aspect-based Sentiment Analysis to Review Products Using Naïve Bayes (Muhammad Dwi Aldhi) 5
Loading Dataset 6
Pre-Processing 6
Aspect Extraction 8
POS-Tagging 8
Dependency Parsing 9
Extraction Of Aspects (according to the above approaches) 10
Sentiment Analysis 11
Building the Model 12
ABSA on Chapters as a Whole 14
Creating new csv File 14
Calculating Sentiment Polarity 15
Building the Model 16
Evaluation and Results 16
Conclusion 17
References 18

List of Figures

i: Mind Platter DataFrame 6
ii: Preprocessed Text 7
iii: POS- Tagging on Preprocessed Text 8
iv: Dependency Parsing on tagged text 9
v: Aspects Extracted I 13
vi: Aspect Extracted II 13
vii: Aspect Extracted III 13
viii: Aspects Extracted IV 14
ix: CSV file for chapters as whole ABSA 15
x: Overall Sentiment Polarity 16
xi: Accuracy I: Performance of ABSA on chapters 17
xii: Accuracy II: Performance of ABSA on chapters as a whole 17

Introduction

Natural Language Processing (NLP) has revolutionized the way we analyze and understand human language. Aspect-Based Sentiment Analysis (ABSA), a subset of opinion mining, takes this a step further by identifying specific aspects and their associated emotions within a given text. This report aims to design a solution that extracts primary aspects and author sentiments from a selected text, using NLP and ABSA techniques.

For this assignment, the Mind Platter book as our sample text, applying our solution to 15 chapters to uncover the author's sentiments and emotions related to various aspects. Najwa Zebian writes "Mind Platter" reflecting on life experiences of feeling unheard, mistreated, or unseen. Zebian, an author, speaker, and educator, passionately explores human complexities through language. This report aims to create an NLP-based ML model to extract emotions from text and test its accuracy on new data. Analyzing Zebian's work using NLP and ABSA to reveal her emotions and test the model's understanding of human language.

Literature Review

Aspect-Based Sentiment Analysis (ABSA), a subset of Natural Language Processing (NLP), is a valuable tool for businesses to understand customer opinions by identifying aspects and sentiments in text data, with recent advancements in NLP and machine learning enhancing its accuracy and efficiency for applications like customer feedback analysis and social media monitoring.

However, despite these advancements, ABSA remains a challenging task, particularly when dealing with complex texts, ambiguous language, and domain-specific terminology. This literature review aims to provide an overview of the current state of ABSA, discussing its techniques, applications, and challenges, as well as exploring future directions for research in this field.

Aspect Level Sentiment Analysis Approaches (Bhavana R. Bhamare)

This paper surveys various methodologies used in aspect-based sentiment analysis (ABSA), focusing on both machine learning and lexicon-based approaches. It discusses several supervised classification methods, including Support Vector Machine (SVM) and Boolean Multinomial Naïve Bayes (BMNB), which involve feature extraction, feature selection, classifier training, and performance testing. Furthermore, unsupervised techniques like Latent Dirichlet Allocation (LDA) and clustering are explored for aspect extraction, while label propagation is discussed for sentiment classification.
The survey highlights that hybrid approaches, such as combining CRF for aspect extraction with Maximum Entropy classifiers for sentiment classification, generally yield better accuracy compared to pure machine learning or lexicon-based methods. Moreover, the paper demonstrates the effectiveness of deep learning models like Convolutional Neural Networks (CNN) for both aspect extraction and sentiment analysis tasks. Overall, the paper provides a comprehensive overview of ABSA methodologies, emphasizing the importance of hybrid and deep learning approaches in achieving improved performance metrics.

Through this research paper, we explored the various options to perform sentiment analysis on the book “Mind Platter”, our main focus while studying this research paper was to look into semi-supervised techniques as our preferred approach for sentiment analysis on the book "Mind Platter". We chose Multinomial Naive Bayes (MB) over other approaches mentioned in the research paper for several reasons.

Firstly, MB is a semi-supervised technique that can effectively handle the large amount of unlabeled data in our text corpus, making it a suitable choice for our task. Additionally, MB is a robust and efficient algorithm that can handle high-dimensional data and is less prone to overfitting, which is a common issue

in deep learning models like CNN. While hybrid approaches like combining CRF with Maximum Entropy classifiers showed promising results, we opted for MB due to its simplicity and ease of implementation, allowing us to focus on feature engineering and hyperparameter tuning. Furthermore, MB's ability to handle multi-class classification problems made it a suitable choice for our sentiment analysis task, where we aim to classify text into multiple sentiment categories. Overall, we believed that MB offered a balance between model complexity and performance, making it an ideal choice for our sentiment analysis task

Aspect Based Sentiment Analysis and Opinion Mining on Twitter Data Set Using Linguistic Rules (Sanjay Tanwani)

This study proposes a model named SVMS (SVM using Spacy) for aspect-based sentiment analysis of Twitter data. The dataset comprised tweets collected using the Twitter API, resulting in an initial set of 33,036 tweets, which were unlabeled and required extensive preprocessing.
After data cleaning, the dataset was reduced to 7,570 tweets. The preprocessing techniques employed included tokenization, stop word removal, lemmatization, and normalization, all facilitated by Python's Spacy package. For feature extraction, dependency relations and opinion word extraction were utilized to identify product features and their associated sentiments. Dependency parsing, carried out using the StanfordNLP parser, decomposed sentences into their grammatical structures, identifying the relationships between words. Named Entity Recognition (NER) was applied to classify named entities into predefined categories such as persons, locations, and organizations. Part-of-speech (POS) tagging further aided in extracting product features and opinion words by classifying words into their respective parts of speech. Additionally, chunking was used to group individual pieces of information into larger chunks for entity detection.

Similarly, in our research, we also deal with unlabeled data, specifically the text from the book "Mind Platter". We were interested in exploring the application of NLP techniques, such as dependency parsing, POS tagging and Named Entity Recognition (NER), on our unlabeled data, just like the SVMS model. However, we opted for Multinomial Naive Bayes (MB) as our classification algorithm, rather than SVM. By leveraging dependency parsing and NER, we aimed to extract relevant features from our text data and improve the performance of our MB model. The SVMS model's ability to handle unlabeled Twitter data and achieve promising results encouraged us to adapt similar NLP techniques for our own research, tailoring them to suit our specific task and dataset.

Aspect-based Sentiment Analysis to Review Products Using Naïve Bayes (Muhammad Dwi Aldhi)

This research focuses on aspect-based sentiment analysis using a Naïve Bayes classifier. The methodology involves several steps: data preprocessing, including case folding, tokenization, stop word removal, and stemming, with POS tagging performed using Stanford CoreNLP tools; feature selection using the Chi Square method to select relevant features from the preprocessed data, with two bag-of-words created for aspects and sentiment polarity terms; and classification using the Naïve Bayes classifier to determine the sentiment polarity of aspects, with the model's performance evaluated using metrics such as accuracy, precision, recall, and F1 measure. We drew inspiration from this research paper to implement our own Multinomial Naive Bayes (MB) classifier for ABSA, recognizing the advantages of Naive Bayes in handling high-dimensional data and its robustness to noise and outliers.

By choosing MB, we leveraged its ability to handle multi-class classification problems and its efficiency in computation, making it an ideal choice for our sentiment analysis task. Additionally, MB's simplicity and ease of implementation allowed us to focus on feature engineering and hyperparameter tuning, further enhancing our model's performance.

Loading Dataset

Fifteen chapters from the book 'Mind Platter' were selected as the dataset, which was then loaded into a Pandas dataframe. The dataframe contains two columns: 'Title' with the chapter names and 'Text' with the corresponding chapter texts.

Code Snippet:

```
import pandas as pd
df = pd.read_csv('/content/drive/MyDrive/mindplatter.csv', header=None,
names=['chapter'])
# Split the 'chapter' column into 'title' and 'text' columns
df[['title', 'text']] = df['chapter'].str.split(':', expand=True)
df = df.drop('chapter', axis=1)
print(df)
```

	title	text
0	Power	Respect the freedom that you were born with b...
1	Rest Your Heart	Your heart is precious, so take care of it. I...
2	Let the Mask Fall Off	I appreciate a genuine effort over a fake att...
3	Stay True to Yourself	Don't worry about what people think of you or...
4	Be Sought	Whatever you do, do it with purpose. Being fo...
5	Lifelong Learning	If you think that education only takes place ...
6	Take the Lead	You choose how your life is going to be. Don'...
7	Be Honest with Me	Don't tell me what I want to hear. Tell me th...
8	The Way They Treat You	Have you ever been told to treat people the w...
9	The Power of Silence	Silence can hold more meaning than words. It ...
10	Take Responsibility	Just as you can't deny that you can feel love...
11	Through Their Eyes	See yourself through the eyes of those who lo...
12	Childhood Nostalgia	As a child, you usually make decisions withou...
13	Know Yourself	Have that wise instinct of knowing how you wo...
14	Thoughts Unexpressed	There are so many thoughts that would be much...

i: Mind Platter DataFrame

Pre-Processing

To preprocess the data and prepare it for Aspect-Based Sentiment Analysis (ABSA), we performed the following steps: tokenization of the text to split it into individual words, removal of abbreviations, punctuations, and stop words to eliminate noise, conversion of all text to lowercase to maintain consistency, and enclosure of each text sample in quotes to ensure proper formatting.

Code Snippet:

```
import string
import nltk
from nltk.corpus import stopwords

def preprocess_text(text):
    # Tokenization
```

```
    tokens = nltk.word_tokenize(text)

    # Remove abbreviations
    tokens = [token for token in tokens if "'" not in token]

    # Convert to lower case
    tokens = [token.lower() for token in tokens]

    # Remove punctuation marks
    punctuation = string.punctuation
    tokens = [token for token in tokens if token not in punctuation]

    # Remove stopwords
    stop_words = set(stopwords.words('english'))
    tokens = [token for token in tokens if token not in stop_words]

    # Enclose each token in quotes
    tokens = [f'"{token}"' for token in tokens]

    return tokens

# Apply the pre-processing function to the 'text' column
df['text'] = df['text'].apply(preprocess_text)
```

	title	text
0	Power	["respect", "freedom", "born", "denied", "brea...
1	Rest Your Heart	["heart", "precious", "take", "care", "may", "...
2	Let the Mask Fall Off	["appreciate", "genuine", "effort", "fake", "a...
3	Stay True to Yourself	["worry", "people", "think", "way", "try", "ma...
4	Be Sought	["whatever", "purpose", "focused", "something"...
5	Lifelong Learning	["think", "education", "takes", "place", "inst...
6	Take the Lead	["choose", "life", "going", "let", "anyone", "...
7	Be Honest with Me	["tell", "want", "hear", "tell", "truth", "may...
8	The Way They Treat You	["ever", "told", "treat", "people", "way", "tr...
9	The Power of Silence	["silence", "hold", "meaning", "words", "power...
10	Take Responsibility	["ca", "deny", "feel", "love", "hate", "happin...
11	Through Their Eyes	["see", "eyes", "love", "see", "goodness", "fa...
12	Childhood Nostalgia	["child", "usually", "make", "decisions", "wit...
13	Know Yourself	["wise", "instinct", "knowing", "would", "reac...
14	Thoughts Unexpressed	["many", "thoughts", "would", "much", "beautif...

ii: Preprocessed Text

Aspect Extraction

Extracting the aspects is the most essential part of this report because it enables the identification of the specific entities, features, and attributes that are being evaluated or discussed in the text. By pinpointing these aspects, we can accurately determine the sentiment and opinions expressed towards them, which is crucial for understanding the overall sentiment and meaning of the text.

The extract_aspects function leverages the NLP model and matcher to identify specific linguistic patterns, such as adjective-noun or verb-adverb combinations, within the text. These patterns are crucial for extracting relevant aspects, which are key phrases or terms that convey meaningful information. To achieve this, we employed a combination of Natural Language Processing (NLP) techniques, including Part-of-Speech (POS) tagging, dependency parsing, and aspect extraction.

POS-Tagging

POS tagging was applied to identify the grammatical categories of each word in the text, such as nouns, verbs, adjectives, and adverbs. This step helped to disambiguate word meanings and prepare the text for further analysis.

Code Snippet:

```
import nltk
def pos_tagging(tokens):
    # POS Tagging
    tagged_tokens = nltk.pos_tag(tokens)

    return tagged_tokens

# Apply the POS tagging function to the 'text' column
df['tagged_text'] = df['text'].apply(pos_tagging)
```

	tagged_text
0	[("respect", JJ), ("freedom", NNP), ("born", N...
1	[("heart", JJ), ("precious", NNP), ("take", NN...
2	[("appreciate", JJ), ("genuine", NNP), ("effor...
3	[("worry", JJ), ("people", NNP), ("think", NNP...
4	[("whatever", JJ), ("purpose", NNP), ("focused...
5	[("think", JJ), ("education", NNP), ("takes", ...
6	[("choose", JJ), ("life", NNP), ("going", NNP)...
7	[("tell", JJ), ("want", NNP), ("hear", NNP), (...
8	[("ever", JJ), ("told", NNP), ("treat", NNP), ...
9	[("silence", JJ), ("hold", NNP), ("meaning", N...
10	[("ca", JJ), ("deny", NNP), ("feel", NNP), ("l...
11	[("see", JJ), ("eyes", NNP), ("love", NNP), ("...
12	[("child", JJ), ("usually", NNP), ("make", NNP...
13	[("wise", JJ), ("instinct", NNP), ("knowing", ...
14	[("many", JJ), ("thoughts", NNP), ("would", NN...

iii: POS- Tagging on Preprocessed Text

Dependency Parsing

Dependency parsing was then used to analyze the grammatical structure of the text, including subject-verb relationships, object dependencies, and modifier attachments. This step revealed the syntactic dependencies between words and phrases, enabling the identification of aspects in context.

Code Snippet :

```
import spacy

# Load the spaCy model for English
nlp = spacy.load("en_core_web_lg")

def dependency_parsing(tagged_tokens):
    # Convert the list of tuples to a string
    sentence = " ".join([token[0] for token in tagged_tokens])

    # Parse the sentence using spaCy
    parsed_sentence = nlp(sentence)

    # Extract the dependencies
    dependencies = [(token.text, token.dep_, token.head.text) for token in
parsed_sentence]

    return dependencies

# Apply the dependency parsing function to the 'text' column
df['dependencies'] = df['tagged_text'].apply(dependency_parsing)
```

```
0     [(", punct, freedom), (respect, nmod, freedom)...
1     [(", punct, take), (heart, nsubj, take), (", p...
2     [(", punct, appreciate), (appreciate, nmod, cl...
3     [(", punct, people), (worry, nmod, people), ("...
4     [(", punct, focused), (whatever, nmod, purpose...
5     [(", punct, think), (think, nmod, area), (", p...
6     [(", punct, choose), (choose, nmod, point), ("...
7     [(", punct, tell), (tell, ROOT, tell), (", pun...
8     [(", punct, told), (ever, advmod, told), (", p...
9     [(", punct, hold), (silence, nsubj, hold), (",...
10    [(", punct, deny), (ca, aux, deny), (", punct,...
11    [(", punct, see), (see, advcl, see), (", punct...
12    [(", punct, child), (child, nsubj, make), (", ...
13    [(", punct, instinct), (wise, amod, instinct),...
14    [(", punct, thoughts), (many, amod, thoughts),...
Name: dependencies, dtype: object
```

iv: Dependency Parsing on tagged text

Extraction Of Aspects (according to the above approaches)

Finally, aspect extraction was performed using a matcher that identified the pre-defined patterns, such as adjective-noun or verb-adverb combinations, and extracted the corresponding text spans as aspects. These extracted aspects represent the key phrases or terms that convey the sentiment and meaning of the text

Code Snippet:

```
import spacy
from spacy.matcher import Matcher
from nltk.corpus import stopwords
from nltk.sentiment.vader import SentimentIntensityAnalyzer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, classification_report

# Load spaCy model
nlp = spacy.load("en_core_web_lg")

# Define patterns to match
pattern_adj_noun = [
    {"POS": "ADJ"},
    {"POS": "NOUN"}
]
pattern_vrb_adv = [
    {"POS": "VERB"},
    {"POS": "ADV"}
]

# Initialize Matcher
matcher = Matcher(nlp.vocab)

# Add patterns to the Matcher
matcher.add("ADJ_NOUN_PATTERN", [pattern_adj_noun])
matcher.add("VRB_ADV_PATTERN", [pattern_vrb_adv])

# Extract the chapter texts from the 'text' column
chapters = df['text'].tolist()

# Extract aspects using the Matcher
def extract_aspects(text):
    doc = nlp(text)
    matches = matcher(doc)
    aspects = []
```

```
    for match_id, start, end in matches:
        string_id = nlp.vocab.strings[match_id]
        if string_id == "ADJ_NOUN_PATTERN" or string_id == "VRB_ADV_PATTERN":
            aspects.append(doc[start:end].text)
    return aspects
```

Sentiment Analysis

The get_sentiment function uses the VADER sentiment analysis tool to assess the sentiment polarity of the text. It categorizes the sentiment as positive, negative, or neutral based on the compound score. The absa function integrates the previous two functions to perform sentiment analysis on each extracted aspect. It returns a dictionary with aspects as keys and their corresponding sentiment polarities as values. For each chapter in the provided dataset (df), the text is preprocessed, aspects are extracted, and sentiments are analyzed. The results are stored in a dictionary chapter_aspects with chapter titles as keys. The aspects and their sentiments are displayed chapter-wise, providing a detailed view of sentiment distribution across different sections of the text.

Code Snippet:

```
# Sentiment analysis using VADER
def get_sentiment(text):
    sid = SentimentIntensityAnalyzer()
    scores = sid.polarity_scores(text)
    if scores['compound'] >= 0.05:
        return 'positive'
    elif scores['compound'] <= -0.05:
        return 'negative'
    else:
        return 'neutral'

# Aspect-Based Sentiment Analysis (ABSA)
def absa(text):
    aspects = extract_aspects(text)
    aspect_sentiments = {aspect: get_sentiment(aspect) for aspect in aspects}
    return aspect_sentiments

# Perform ABSA on each chapter
chapter_aspects = {}
for chapter, preprocessed_text in zip(df['title'], preprocessed_texts):
    aspects = extract_aspects(preprocessed_text)
    aspect_sentiments = {aspect: get_sentiment(aspect) for aspect in aspects}
    chapter_aspects[chapter] = aspect_sentiments

# Display aspects and sentiments, segregated by chapter
for chapter, aspects in chapter_aspects.items():
    print(f"Chapter: {chapter}")
```

```
    for aspect, sentiment in aspects.items():
        print(f"Aspect: {aspect} (Sentiment Term: {aspect}, Sentiment
Polarity: {sentiment})")
    print()  # Empty line for separation
```

Building the Model

As previously mentioned in this report, we opted to implement the Multinomial Naïve Bayes (MNB) classifier due to its simplicity and promising performance, as discussed in the literature review (Muhammad Dwi Aldhi, 2017) where it achieved an accuracy of 78%. Before training the model, we preprocessed the data by flattening the nested dictionary of aspect-sentiment pairs into a single list of tuples and then separating the aspects and sentiments into two distinct lists. We then employed TfidfVectorizer to extract features from the data and fed them into the MNB model, using an 80%-20% data split for training and testing purposes.

Code Snippet:

```
# Prepare data for model training
aspect_sentiment_pairs = [(aspect, sentiment) for chapter_aspects in
chapter_aspects.values() for aspect, sentiment in chapter_aspects.items()]
aspects, sentiments = zip(*aspect_sentiment_pairs)


# Feature extraction using TF-IDF
vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(aspects)
y = sentiments

# Split data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=50)

# Train the model
model = MultinomialNB()
model.fit(X_train, y_train)

# Predict and evaluate
y_pred = model.predict(X_test)
print(f"Accuracy: {accuracy_score(y_test, y_pred)}")
print(classification_report(y_test, y_pred))
```

```
Chapter: Power
Aspect: free power (Sentiment Term: free power, Sentiment Polarity: positive)

Chapter: Rest Your Heart
Aspect: precious take (Sentiment Term: precious take, Sentiment Polarity: positive)
Aspect: forgive unconditionally (Sentiment Term: forgive unconditionally, Sentiment Polarity: positive)
Aspect: surround value (Sentiment Term: surround value, Sentiment Polarity: positive)

Chapter: Let the Mask Fall Off
Aspect: genuine effort (Sentiment Term: genuine effort, Sentiment Polarity: neutral)
Aspect: fake attempt (Sentiment Term: fake attempt, Sentiment Polarity: negative)
Aspect: little sun (Sentiment Term: little sun, Sentiment Polarity: neutral)

Chapter: Stay True to Yourself
Aspect: good person (Sentiment Term: good person, Sentiment Polarity: positive)
Aspect: bad person (Sentiment Term: bad person, Sentiment Polarity: negative)
Aspect: stop ironically (Sentiment Term: stop ironically, Sentiment Polarity: negative)
Aspect: good intention (Sentiment Term: good intention, Sentiment Polarity: positive)
Aspect: confident intention (Sentiment Term: confident intention, Sentiment Polarity: positive)
Aspect: drag back (Sentiment Term: drag back, Sentiment Polarity: negative)
Aspect: true change (Sentiment Term: true change, Sentiment Polarity: positive)
```

v: Aspects Extracted I

```
Chapter: Be Sought
Aspect: clear vision (Sentiment Term: clear vision, Sentiment Polarity: positive)

Chapter: Lifelong Learning
Aspect: think long (Sentiment Term: think long, Sentiment Polarity: neutral)
Aspect: breathe mind (Sentiment Term: breathe mind, Sentiment Polarity: neutral)
Aspect: dynamic knowledge (Sentiment Term: dynamic knowledge, Sentiment Polarity: positive)
Aspect: subject area (Sentiment Term: subject area, Sentiment Polarity: neutral)

Chapter: Take the Lead
Aspect: popular people (Sentiment Term: popular people, Sentiment Polarity: positive)
Aspect: cognizant decision (Sentiment Term: cognizant decision, Sentiment Polarity: neutral)

Chapter: Be Honest with Me
Aspect: hurt definitely (Sentiment Term: hurt definitely, Sentiment Polarity: negative)
Aspect: right person (Sentiment Term: right person, Sentiment Polarity: neutral)
Aspect: front word (Sentiment Term: front word, Sentiment Polarity: neutral)
Aspect: many time (Sentiment Term: many time, Sentiment Polarity: neutral)
```

vi: Aspect Extracted II

```
Chapter: The Way They Treat You
Aspect: treat well (Sentiment Term: treat well, Sentiment Polarity: positive)
Aspect: bad action (Sentiment Term: bad action, Sentiment Polarity: negative)
Aspect: good encourage (Sentiment Term: good encourage, Sentiment Polarity: positive)
Aspect: listen also (Sentiment Term: listen also, Sentiment Polarity: neutral)
Aspect: answer back (Sentiment Term: answer back, Sentiment Polarity: neutral)
Aspect: good compare (Sentiment Term: good compare, Sentiment Polarity: positive)
Aspect: bad enemy (Sentiment Term: bad enemy, Sentiment Polarity: negative)

Chapter: The Power of Silence
Aspect: heartless person (Sentiment Term: heartless person, Sentiment Polarity: negative)
Aspect: innocent victim (Sentiment Term: innocent victim, Sentiment Polarity: positive)
Aspect: powerful word (Sentiment Term: powerful word, Sentiment Polarity: positive)
Aspect: close mouth (Sentiment Term: close mouth, Sentiment Polarity: neutral)
Aspect: inner power (Sentiment Term: inner power, Sentiment Polarity: neutral)

Chapter: Take Responsibility
Aspect: wrong take (Sentiment Term: wrong take, Sentiment Polarity: negative)
Aspect: say instead (Sentiment Term: say instead, Sentiment Polarity: neutral)
Aspect: silent need (Sentiment Term: silent need, Sentiment Polarity: neutral)

Chapter: Through Their Eyes
Aspect: ready lift (Sentiment Term: ready lift, Sentiment Polarity: positive)
Aspect: make even (Sentiment Term: make even, Sentiment Polarity: neutral)
```

vii: Aspect Extracted III

```
Chapter: Childhood Nostalgia
Aspect: spontaneous decision (Sentiment Term: spontaneous decision, Sentiment Polarity: neutral)
Aspect: instant happiness (Sentiment Term: instant happiness, Sentiment Polarity: positive)
Aspect: move away (Sentiment Term: move away, Sentiment Polarity: neutral)
Aspect: kind stress (Sentiment Term: kind stress, Sentiment Polarity: positive)
Aspect: think long (Sentiment Term: think long, Sentiment Polarity: neutral)
Aspect: happy decision (Sentiment Term: happy decision, Sentiment Polarity: positive)
Aspect: dear self (Sentiment Term: dear self, Sentiment Polarity: positive)

Chapter: Know Yourself
Aspect: wise instinct (Sentiment Term: wise instinct, Sentiment Polarity: positive)
Aspect: certain situation (Sentiment Term: certain situation, Sentiment Polarity: positive)
Aspect: happen well (Sentiment Term: happen well, Sentiment Polarity: positive)
Aspect: react wisely (Sentiment Term: react wisely, Sentiment Polarity: positive)
Aspect: unexpected situation (Sentiment Term: unexpected situation, Sentiment Polarity: neutral)
Aspect: react well (Sentiment Term: react well, Sentiment Polarity: positive)
Aspect: next time (Sentiment Term: next time, Sentiment Polarity: neutral)
Aspect: long stand (Sentiment Term: long stand, Sentiment Polarity: neutral)

Chapter: Thoughts Unexpressed
Aspect: feeling well (Sentiment Term: feeling well, Sentiment Polarity: positive)
Aspect: keep inside (Sentiment Term: keep inside, Sentiment Polarity: neutral)
Aspect: right time (Sentiment Term: right time, Sentiment Polarity: neutral)
Aspect: express meanwhile (Sentiment Term: express meanwhile, Sentiment Polarity: neutral)
Aspect: strong want (Sentiment Term: strong want, Sentiment Polarity: positive)
```

viii: Aspects Extracted IV

ABSA on Chapters as a Whole

To provide a comprehensive evaluation, we also conducted Aspect-Based Sentiment Analysis (ABSA) on the chapters as a single entity, allowing us to assess the performance of the Multinomial Naïve Bayes (MNB) classifier. To facilitate this comparison, we extracted the pre-identified aspects and their corresponding sentiment polarities into a new CSV file named 'absa_output.csv'. This step enabled us to benchmark the MNB model's performance against the chapters' overall sentiment, providing a more holistic understanding of its effectiveness

Creating new csv File

```
import csv

with open('absa_output.csv', 'w', newline='') as csvfile:
    writer = csv.writer(csvfile)
    writer.writerow(["Chapter Name", "All Aspects", "All Sentiment
Polarity"])  # header row

    for chapter, aspects in chapter_aspects.items():
        all_aspects = ",".join(aspects.keys())  # Join all aspects with
commas
        all_sentiments = ",".join(aspects.values())  # Join all sentiments
with commas
        writer.writerow([chapter, all_aspects, all_sentiments])

df_2 = pd.DataFrame([
    (chapter, ",".join(aspects.keys()), ",".join(aspects.values()))
    for chapter, aspects in chapter_aspects.items()
], columns=["Chapter Name", "All Aspects", "All Sentiment Polarity"])

df_2.to_csv("absa_output.csv", index=False)
```

	Chapter Name	All Aspects	All Sentiment Polarity
0	Power	free power	positive
1	Rest Your Heart	precious take forgive unconditionally surround value	positive,positive,positive
2	Let the Mask Fall Off	genuine effort fake attempt little sun	neutral,negative,neutral
3	Stay True to Yourself	good person bad person stop ironically good intention confident intention drag back true change	positive,negative,negative,positive,positive,negative,positive
4	Be Sought	clear vision	positive
5	Lifelong Learning	think long breathe mind dynamic knowledge subject area	neutral,neutral,positive,neutral
6	Take the Lead	popular people cognizant decision	positive,neutral
7	Be Honest with Me	hurt definitely right person front word many time	negative,neutral,neutral,neutral
8	The Way They Treat You	treat bad action good encourage listen also answer back good compare bad enemy	positive,negative,positive,neutral,neutral,positive,negative
9	The Power of Silence	heartless person innocent victim powerful word close mouth inner power	negative,positive,positive,neutral,neutral
10	Take Responsibility	wrong take say instead silent need	negative,neutral,neutral
11	Through Their Eyes	ready lift make	positive,neutral
12	Childhood Nostalgia	spontaneous decision instant happiness move away kind stress think long happy decision dear self	neutral,positive,neutral,positive,neutral,positive,positive
13	Know Yourself	wise instinct certain situation happen react wisely unexpected situation react next time long stand	positive,positive,positive,positive,neutral,positive,neutral,neutral

ix: CSV file for chapters as whole ABSA

Calculating Sentiment Polarity

Next, since the sentiment polarity was aspect-based, with each word having its own polarity, we decided to determine the overall sentiment polarity of each chapter by considering the frequency of polarity words. Specifically, if a polarity word (e.g., positive, negative or neutral) appeared three or more times in a chapter, we assigned that polarity to the chapter as a whole. This approach led to most chapters being classified as either positive or neutral, with few chapters exhibiting negative sentiment.

```
df_2['Overall Sentiment Polarity'] = ''

# Iterate over each row in the DataFrame
for index, row in df_2.iterrows():
    # Split the sentiments into a list
    sentiments = row['All Sentiment Polarity'].split(',')

    # Count the occurrences of each sentiment
    sentiment_counts = {}
    for sentiment in sentiments:
        if sentiment in sentiment_counts:
            sentiment_counts[sentiment] += 1
        else:
            sentiment_counts[sentiment] = 1

    # Find the sentiment with the highest count
    overall_sentiment = max(sentiment_counts, key=sentiment_counts.get)
```

```
# Assign the overall sentiment to the new column
df_2.at[index, 'Overall Sentiment Polarity'] = overall_sentiment
```

	Chapter Name	All Aspects	All Sentiment Polarity	Overall Sentiment Polarity
0	Power	free power	positive	positive
1	Rest Your Heart	precious take,forgive unconditionally,surround value	positive,positive,positive	positive
2	Let the Mask Fall Off	genuine effort,fake attempt,little sun	neutral,negative,neutral	neutral
3	Stay True to Yourself	good person,bad person,stop ironically,good intention,confident intention,drag back,true change	positive,negative,negative,positive,positive,negative,positive	positive
4	Be Sought	clear vision	positive	positive
5	Lifelong Learning	think long,breathe mind,dynamic knowledge,subject area	neutral,neutral,positive,neutral	neutral
6	Take the Lead	popular people,cognizant decision	positive,neutral	positive
7	Be Honest with Me	hurt definitely,right person,front word,many time	negative,neutral,neutral,neutral	neutral
8	The Way They Treat You	treat well,bad action,good encourage,listen also,answer back,good compare,bad enemy	positive,negative,positive,neutral,neutral,positive,negative	positive

x: Overall Sentiment Polarity

Building the Model

Moreover, since the aspects were stored in a comma-separated format, we needed to preprocess the data by removing punctuation marks to provide a clean input for the model. As we intended to utilize the same Multinomial Naïve Bayes (MNB) classifier for this task, we extracted features using TF-IDF and transformed the aspects into this format. This step enabled us to prepare a consistent and formatted dataset for the MNB model to learn from.

```
# Feature extraction using TF-IDF
aspects = df_2['All Aspects']
sentiments = df_2['Overall Sentiment Polarity']
vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(aspects)
y = sentiments

# Split data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=50)

# Train the model
model = MultinomialNB()
model.fit(X_train, y_train)

# Predict and evaluate
y_pred = model.predict(X_test)
print(f"Chapters as whole Accuracy: {accuracy_score(y_test, y_pred)}")
print(classification_report(y_test, y_pred))
```

Evaluation and Results

The Multinomial Naïve Bayes (MNB) classifier achieved an overall accuracy of 84.6% on the test dataset.

The model performed exceptionally well on the neutral class, with a precision, recall, and F1-score of 0.84, 1.00, and 0.91, respectively. However, it struggled with the negative class, failing to classify any samples correctly. The model showed moderate performance on the positive class, with a precision of 1.00 but a lower recall and F1-score of 0.25 and 0.40, respectively.

The same MNB classifier achieved an overall accuracy of 66.7% when performing ABSA on the chapters as a whole. The model performed moderately on the positive class, with a precision of 0.67, recall of 1.00, and F1-score of 0.80. However, it failed to classify the single neutral chapter correctly, resulting in zero precision, recall, and F1-score for that class. The macro average and weighted average metrics indicate that the model's performance is biased towards the positive class, highlighting the need for improvement in classifying neutral and potentially other classes.

Performing ABSA on every single chapter yielded better results than performing ABSA on the chapters as whole, with the first one achieving 84% accuracy overall and the chapters as a whole achieving 66% accuracy.

Accuracy: 0.8461538461538461

	precision	recall	f1-score	support
negative	0.00	0.00	0.00	1
neutral	0.84	1.00	0.91	21
positive	1.00	0.25	0.40	4
accuracy			0.85	26
macro avg	0.61	0.42	0.44	26
weighted avg	0.83	0.85	0.80	26

xi: Accuracy I: Performance of ABSA on chapters

Chapters as whole Accuracy: 0.6666666666666666

	precision	recall	f1-score	support
neutral	0.00	0.00	0.00	1
positive	0.67	1.00	0.80	2
accuracy			0.67	3
macro avg	0.33	0.50	0.40	3
weighted avg	0.44	0.67	0.53	3

xii: Accuracy II: Performance of ABSA on chapters as a whole

Conclusion

In conclusion, the extraction of aspects from the provided text was conducted using two approaches: the Matcher method for identifying adjective-noun patterns and verb-adverb patterns, and the VADER tool for

categorizing aspects as positive, negative, or neutral based on sentiment polarity scores. The Aspect-Based Sentiment Analysis (ABSA) approach was then applied to each chapter individually, achieving an impressive 84% accuracy with the Multinomial Naïve Bayes (MNB) classifier.

This process provided a comprehensive overview of aspect sentiments segregated by chapter, displaying aspects and their associated sentiments for each chapter and facilitating a detailed understanding of sentiment dynamics throughout the text. Furthermore, the data collected from aspect sentiment analysis was structured for model training purposes, enabling further analysis and potential predictive modeling. Overall, this study demonstrates the effectiveness of ABSA in aspect sentiment analysis and highlights its potential applications in various fields.

References

Girija V R, S. T. (2023, Febuary 2). *A Comparative Review on Approaches of Aspect Level Sentiment Analysis.* Retrieved from IEEE Xplore: https://ieeexplore.ieee.org/document/10073770

Muhammad Dwi Aldhi, K. A. (2017, August). *Aspect-based sentiment analysis to review products using Naïve Bayes.* Retrieved from ResearchGate:

https://www.researchgate.net/publication/318857333_Aspect-based_sentiment_analysis_to_review_products_using_Naive_Bayes

Sanjay Tanwani, R. A. (2023, March 11). *Aspect Based Sentiment Analysis and Opinion Mining on Twitter Data Set Using Linguistic Rules.* Retrieved from ResearchGate: https://www.researchgate.net/publication/369427699_Aspect_Based_Sentiment_Analysis_and_Opinion_Mining_on_Twitter_Data_Set_Using_Linguistic_Rules