Intel Unnati Industrial Training Program

- **Team:** Brainiacs

- **Team Members:**
    1. Ananya Prabhu
    2. Neha Nayak
    3. Pooja D

- **Problem Statement**:
  PS-11: "Intel Products Sentiment Analysis from Online Reviews"

- **Institute Name:** Shri Dharmasthala Manjunatheshwara College of Engineering and Technology, Dharwad.
- **Nominator Name:** Dr. Shrinivasrao B. Kulkarni
- **Mentor Name:** Dr. Archana Nandibewoor

- **Description:**

  Intel Products are reviewed by end users and tech reviewers on various platforms. Hence, the ask is to scrap the reviews available on different sources in the last 3 - 5 years. Apply various exploratory data analysis, machine learning, and natural programming techniques to find the sentiments of products, clustering of affinity reviews, trends of sentiments over period, features and keywords extraction to specific sentiments, recommendation on key improvements based on users reviews for future products. Ensure sentiments and any other analysis is also demonstrated on different product category.

Table of Contents

Introduction

In this project, we aim to analyze user reviews of Intel products using advanced techniques in artificial intelligence and machine learning. By collecting and aggregating reviews from various online platforms over the past 3 to 5 years, we utilized natural language processing (NLP) to determine the sentiments expressed by users. Our approach includes applying exploratory data analysis (EDA), sentiment classification using VADER and RoBERTa models, and clustering reviews with similar sentiments. We also extracted key features and keywords associated with specific sentiments. The goal is to provide actionable insights and recommendations for future product improvements. Through this project, we will gain practical experience in data scraping, machine learning, and NLP, and present our findings in a comprehensive report and presentation.

Literature Review

Sentiment analysis, a subfield of natural language processing (NLP), involves extracting and analyzing subjective information from text data. It has been extensively used to gauge public opinion, monitor brand reputation, and understand customer satisfaction. This literature review delves into the methodologies and techniques relevant to sentiment analysis and their applications, particularly focusing on product reviews for Intel products.

Sentiment Analysis Techniques

Sentiment analysis can be broadly categorized into three approaches: lexicon-based, machine learning-based, and hybrid methods.

- **Lexicon-Based Methods:** Lexicon-based methods rely on predefined dictionaries of words associated with positive or negative sentiments. One popular tool is VADER (Valence Aware Dictionary and sEntiment Reasoner), which is specifically tailored for social media texts. Research by Hutto and Gilbert (2014) highlights VADER's effectiveness in capturing nuanced sentiments in text data, making it a valuable tool for initial sentiment analysis.
- **Machine Learning-Based Methods:** Machine learning-based approaches involve training algorithms on labeled datasets to classify sentiments. Techniques range from traditional methods like Naive Bayes and Support Vector Machines (SVM) to deep learning models like Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks. Recent advancements have introduced transformer models such as BERT (Bidirectional Encoder Representations from Transformers) and RoBERTa (Robustly optimized BERT approach). Liu et al. (2019) demonstrated RoBERTa's superior performance in various NLP tasks, making it an excellent choice for advanced sentiment analysis..

Applications in Product Reviews

Sentiment analysis has been widely applied to product reviews to extract insights and inform business decisions. In the context of Intel products, analyzing online reviews can provide insights into customer satisfaction, product performance, and areas needing improvement.

Data Sources and Tools

The availability of extensive datasets, such as the Amazon Product Reviews Dataset, has facilitated large-scale sentiment analysis.Additionally, social media platforms like Twitter and forums like Reddit provide valuable data sources for real-time sentiment analysis.

Methodology and Expected Outcomes

This project will employ a combination of VADER and RoBERTa models to perform sentiment analysis on Intel product reviews. By aggregating reviews from multiple sources, we aim to classify sentiments, identify trends, and extract key features associated with specific sentiments. The insights gained will inform recommendations for product improvements and strategic decision-making.

Data Collection Methodology

In this project, our objective is to collect and analyze online reviews of Intel products from various sources. The data collection process involves web scraping, a technique for extracting information from websites. The following steps outline our methodology for scraping Amazon reviews of Intel products:

1. Import Necessary Libraries

We begin by importing the necessary libraries for web scraping and data handling.

```
import requests
from bs4 import BeautifulSoup
import pandas as pd
```

2. Initialize Review List

We create an empty list to store the reviews we collect

```
review_list = []
```

3. Define Helper Functions

We define two helper functions: `get_soup` and `get_reviews`.

- `get_soup(url)`: This function takes a URL as input and returns a BeautifulSoup object containing the HTML content of the page. We use the Splash service to render JavaScript content before scraping.

```
def get_soup(url):
     r = requests.get('http://localhost:8050/render.html', params={'url':
url, 'wait': 2})
    soup = BeautifulSoup(r.text, 'html.parser')
    return soup
```

- `get_reviews(soup)`: This function extracts the reviews from the BeautifulSoup object. It locates review elements on the page and retrieves the review rating and review body text.

```
def get_reviews(soup):
    reviews = soup.find_all('div', {'data-hook': 'review'})
    for item in reviews:
        rating = item.find('i', {'data-hook': 'review-star-rating'})
        body = item.find('span', {'data-hook': 'review-body'})

        review = {}

        if rating is not None:
            review['review_rating'] = float(rating.text.replace('out of 5
stars', '').strip())

        if body is not None:
```

```
            review['review_body'] = body.text.strip()

        review_list.append(review)
```

4. Iterate Over Multiple Pages

We loop through multiple pages of reviews to collect a comprehensive dataset. For each page, we call the `get_soup` function to retrieve the HTML content and the `get_reviews` function to extract reviews.

```
for x in range(1, 20):
     soup  =  get_soup(f'https://www.amazon.in/Intel%C2%AE-CoreTM-i5-10400-
Processor)
    print(f'Getting page: {x}')
    get_reviews(soup)
    print(len(review_list))
    if not soup.find('li', {'class': 'a-disabled a-last'}):
        pass
    else:
        break
```

5. Save the Data

Finally, we save the collected reviews to an Excel file for further analysis.

```
df = pd.DataFrame(review_list)
df.to_excel('intel_reviews.xlsx', index=False)
print('Finish')
```

Data preprocessing Methodology

The data preprocessing steps in the Intel Product Sentiment Analysis project with some example code snippets:

1. Data Collection: Scraping reviews using libraries like BeautifulSoup or Scrapy.

```
import requests
from bs4 import BeautifulSoup
url = 'https://example.com/reviews'
response = requests.get(url)
soup = BeautifulSoup(response.text, 'html.parser')
reviews = soup.find_all('div', class_='review')
```

2. Data Cleaning: Removing irrelevant data and handling missing values.

```
import pandas as pd
df = pd.DataFrame(reviews)
df.dropna(inplace=True)  # Removing missing values
df['review_text'] = df['review_text'].str.lower()  # Normalizing text
```

3. Exploratory Data Analysis (EDA): Identifying patterns and trends.

```
import matplotlib.pyplot as plt
df['review_length'] = df['review_text'].apply(len)
plt.hist(df['review_length'], bins=20)
plt.show()
```

4. Feature Extraction: Extracting features like keywords.

```
from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer(stop_words='english')
X = vectorizer.fit_transform(df['review_text'])
```

5. Sentiment Analysis: Applying machine learning for sentiment analysis.

```
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score
X_train, X_test, y_train, y_test = train_test_split(X, df['sentiment'],
test_size=0.2, random_state=42)
model = MultinomialNB()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
print(f'Accuracy: {accuracy_score(y_test, y_pred)}')
```

Sentiment Analysis Methodology

1. Data Acquisition

Objective: To download and load the dataset containing Intel product reviews.
Steps:

- **Download the file from Google Drive:** We used `gdown` to download the file from Google Drive using its file ID.
- **Read the Excel file into a pandas DataFrame:** We used `pd.read_excel` to read the downloaded Excel file into a pandas DataFrame.

Code:

```
import gdown

# Google Drive file ID
file_id = '1bUTi9y9AfBPUhSfBTmjJ1QLS1ctkcqi2'
url = f'https://drive.google.com/uc?id={file_id}&export=download'
output = 'Extracted_Dataset.xlsx'

# Download the file from Google Drive
gdown.download(url, output, quiet=False)

# Read the downloaded Excel file
df = pd.read_excel(output)
```

2. Data Visualization

Objective: To visualize the distribution of review ratings.

Steps:

- **Plot the distribution of review ratings:** We used the `value_counts` method to get the count of each rating, then plotted these counts using `matplotlib`.

Code:

```
df['review_rating'].value_counts().sort_index().plot(kind='bar', title='Count of
Reviews by Stars', figsize=(10,5))
plt.xlabel('Rating')
plt.ylabel('Count')
plt.title('Distribution of Ratings')
plt.show()
```

3. Text Processing and Sentiment Analysis using VADER

Objective: To perform sentiment analysis on the reviews using the VADER sentiment analysis tool.
Steps:

- **Download and initialize VADER:** We downloaded the VADER lexicon and initialized the VADER sentiment analyzer.
- **Analyze sentiment of reviews:** We looped through each review, applying the VADER sentiment analyzer to compute sentiment scores (positive, negative, neutral, and compound).

Code:

```
import nltk
nltk.download('vader_lexicon')
from nltk.sentiment.vader import SentimentIntensityAnalyzer

# Initialize the VADER sentiment analyzer
sia = SentimentIntensityAnalyzer()

# Analyze the sentiment of an example review
example = df['review_body'][5]
sia.polarity_scores(example)

# Analyze the sentiment of all reviews using VADER
res = {}
for i, row in tqdm(df.iterrows(), total=len(df)):
    try:
        text = row['review_body']
        if isinstance(text, str):  # Check if text is a string
            myid = row['id']
            res[myid] = sia.polarity_scores(text)
    except RuntimeError:
        print(f'Broke for id {myid}')
vaders = pd.DataFrame(res).T
vaders = vaders.reset_index().rename(columns={'index': 'id'})
vaders = vaders.merge(df, how='left')
```

4. Text Processing and Sentiment Analysis using RoBERTa

Objective: To perform sentiment analysis using a pre-trained RoBERTa model from Hugging Face's Transformers library.

Steps:

- **Install and load the model:** We installed the `transformers` library and loaded the RoBERTa model and tokenizer.
- **Define a function to compute sentiment scores using RoBERTa:** The function tokenizes the input text, passes it through the model, and computes sentiment scores.
- **Apply the RoBERTa sentiment analysis to all reviews:** Similar to VADER, we looped through each review and computed sentiment scores using the RoBERTa model.

Code:

```
!pip install transformers
from transformers import AutoTokenizer, AutoModelForSequenceClassification
from scipy.special import softmax

MODEL = f"cardiffnlp/twitter-roberta-base-sentiment"
tokenizer = AutoTokenizer.from_pretrained(MODEL)
```

```
model = AutoModelForSequenceClassification.from_pretrained(MODEL)

# Function to compute sentiment scores using RoBERTa
def polarity_scores_roberta(example):
    encoded_text = tokenizer(example, return_tensors='pt')
    output = model(**encoded_text)
    scores = output[0][0].detach().numpy()
    scores = softmax(scores)
    scores_dict = {
        'roberta_neg' : scores[0],
        'roberta_neu' : scores[1],
        'roberta_pos' : scores[2]
    }
    return scores_dict

# Analyze the sentiment of all reviews using RoBERTa
res = {}
for i, row in tqdm(df.iterrows(), total=len(df)):
    try:
        text = row['review_body']
        if isinstance(text, str):  # Check if text is a string
            myid = row['id']
            vader_result = sia.polarity_scores(text)
            vader_result_rename = {}
            for key, value in vader_result.items():
                vader_result_rename[f"vader_{key}"] = value
            roberta_result = polarity_scores_roberta(text)
            both = {**vader_result_rename, **roberta_result}
            res[myid] = both
    except RuntimeError:
        print(f'Broke for id {myid}')
results_df = pd.DataFrame(res).T
results_df = results_df.reset_index().rename(columns={'index': 'id'})
results_df = results_df.merge(df, how='left')
```

5. Combining Sentiment Scores

Objective: To combine the sentiment scores from VADER and RoBERTa for a unified sentiment classification.

Steps:

- **Define a function to compute combined sentiment scores:** This function takes the average of the VADER and RoBERTa sentiment scores for each review.
- **Apply the function to each review:** We used the `apply` method to compute combined sentiment scores for all reviews.

Code:

```
# Function to compute combined sentiment scores
def compute_combined_sentiment(row):
    combined_neg = (row['vader_neg'] + row['roberta_neg']) / 2
    combined_neu = (row['vader_neu'] + row['roberta_neu']) / 2
    combined_pos = (row['vader_pos'] + row['roberta_pos']) / 2
```

```
    return pd.Series([combined_neg, combined_neu, combined_pos])

# Apply the function to each row in the DataFrame
results_df[['combined_neg',        'combined_neu',        'combined_pos']]       =
results_df.apply(compute_combined_sentiment, axis=1)

# Display the updated DataFrame
print(results_df[['id',  'vader_neg',  'vader_neu',  'vader_pos',  'roberta_neg',
'roberta_neu', 'roberta_pos', 'combined_neg', 'combined_neu', 'combined_pos']])
```

6. Classifying Sentiments

Objective: To classify each review as positive, negative, or neutral based on the combined sentiment scores.

Steps:

- **Define a function to classify sentiment:** This function uses the combined sentiment scores to classify each review.
- **Apply the function to each review:** We used the `apply` method to classify the sentiment of all reviews.

Code:

```
# Function to classify sentiment based on combined scores
def classify_sentiment(row):
     if  row['combined_pos']  >  row['combined_neg']  and  row['combined_pos']  >
row['combined_neu']:
        return 'positive'
     elif  row['combined_neg']  >  row['combined_pos']  and  row['combined_neg']  >
row['combined_neu']:
        return 'negative'
    else:
        return 'neutral'

# Apply the function to each row in the DataFrame
results_df['sentiment'] = results_df.apply(classify_sentiment, axis=1)

# Display the updated DataFrame with the new sentiment column
print(results_df[['id', 'review_body', 'sentiment']])
```

7. Saving the Results

Objective: To save the final DataFrame with sentiment classifications to an Excel file.

Steps:

- **Mount Google Drive:** We mounted Google Drive to access and save files.
- **Save the DataFrame to an Excel file:** We specified the path and used `to_excel` to save the DataFrame.

Code:

```
from google.colab import drive
```

```
# Mount Google Drive to access files
drive.mount('/content/drive')

# Specify the path in your Google Drive where you want to save the Excel file
excel_file_path = 'https://drive.google.com/drive/my-drive/results.xlsx'

# Save DataFrame to Excel
results_df.to_excel(excel_file_path, index=False)

print(f"DataFrame successfully saved to '{excel_file_path}'")
```

8. Data Visualization

Objective: To create visualizations to better understand the sentiment distribution and relationships.

Steps:

- **Pairplot of sentiment scores and review ratings:** We used `seaborn` to create pair plots, visualizing the relationships between the VADER and RoBERTa sentiment scores and the review ratings.

Code:

```
sns.pairplot(data=results_df,
             vars=['vader_neg', 'vader_neu', 'vader_pos',
                   'roberta_neg', 'roberta_neu', 'roberta_pos'],
             hue='review_rating',
             palette='tab10')
plt.show()
```

Through these detailed steps, we successfully conducted sentiment analysis on Intel product reviews using VADER and RoBERTa models. By combining the sentiment scores from both models, we provided a comprehensive sentiment classification for each review, saving the results for further analysis and reporting.

Visualizing important keywords from Intel product reviews

1. **Data Collection**:
    - Download the dataset from Google Drive.
    - Read the dataset into a pandas DataFrame.
2. **Data Preprocessing**:
    - Clean the review text by removing non-word characters and normalizing whitespace.
3. **TF-IDF Vectorization**:
    - Use `TfidfVectorizer` to transform the cleaned review text into a TF-IDF matrix.
    - Extract feature names and their corresponding TF-IDF scores.
    - Select the top 300 keywords based on their TF-IDF scores.
    - Visualize the top 300 TF-IDF keywords using a word cloud.
4. **RAKE Keyword Extraction**:
    - Initialize the RAKE algorithm.
    - Extract and rank keywords from each review.
    - Flatten the list of extracted keywords and select the top 300 keywords based on their scores.
    - Visualize the top 300 RAKE keywords using a word cloud.

Output :

Implementation of the Intel Products Sentiment Analysis Project

1. Data Collection

The project begins with collecting user reviews of Intel products from Amazon. Web scraping techniques are used to extract review data from multiple pages. The scraping process involves sending HTTP requests to Amazon pages, rendering JavaScript content using Splash, and parsing the HTML to locate review elements. Key data points such as review ratings and review body text are extracted and stored in a structured format.

2. Data Preprocessing

Once the data is collected, it undergoes preprocessing to prepare it for analysis. This includes cleaning the text data by removing HTML tags, special characters, and unnecessary whitespace. Additionally, we convert ratings to a numerical format and handle any missing or incomplete data. This step ensures that the data is in a consistent and usable format for subsequent analysis.

3. Sentiment Analysis

We perform sentiment analysis using two models: VADER and RoBERTa.

- **VADER (Valence Aware Dictionary and sEntiment Reasoner):** This lexicon-based model is used for initial sentiment classification. It calculates sentiment scores (positive, negative, neutral, and compound) based on predefined word lists.
- **RoBERTa (Robustly optimized BERT approach):** This transformer-based model is employed for more advanced sentiment analysis. It tokenizes the text and uses a pre-trained model to classify sentiments into categories (negative, neutral, and positive) with associated probabilities.

Both models' sentiment scores are combined to enhance accuracy and reliability. Each review is then assigned a final sentiment label based on the combined scores.

4. Exploratory Data Analysis (EDA)

Exploratory Data Analysis involves visualizing and summarizing the collected data to identify patterns and trends. This includes:

- **Distribution of Ratings:** Visualizing the frequency of each rating (1 to 5 stars) to understand overall customer satisfaction.
- **Sentiment Distribution:** Analyzing the distribution of sentiment scores (positive, neutral, negative) across different ratings.
- **Time Series Analysis:** Examining how sentiments and ratings change over time to identify trends and seasonal patterns.

5. Keyword Extraction

Feature extraction techniques, including TF-IDF (Term Frequency-Inverse Document Frequency) and word embeddings, are used to identify key phrases and terms associated with different sentiment clusters.

6. Reporting and Presentation

The final step is to compile the findings into a comprehensive report and presentation. The report includes detailed analyses, visualizations, and recommendations. The presentation highlights key insights and actionable outcomes, providing stakeholders with a clear understanding of customer sentiments and potential areas for product enhancement.

This project leverages advanced NLP techniques and sentiment analysis models to gain valuable insights from user reviews of Intel products. By systematically collecting, analyzing, and interpreting review data, we can provide actionable recommendations to improve product quality and customer satisfaction. The comprehensive approach ensures robust and reliable results, contributing to informed decision-making and strategic planning for Intel products.

Results and Discussions

The sentiment analysis was conducted on a dataset of user reviews for Intel processors. The primary goal was to evaluate the sentiment of each review using both VADER and RoBERTa models, and then combine these results to classify the overall sentiment. Below are the key findings and discussions based on the analysis.

Sentiment Distribution

The combined sentiment scores were used to categorize reviews into positive, neutral, and negative sentiments. The results show a predominance of positive sentiments among the reviews.

- **Positive Sentiments:** A significant portion of the reviews (around 70%) were classified as positive, indicating high customer satisfaction with Intel processors.
- **Neutral Sentiments:** About 20% of the reviews were neutral, reflecting mixed or balanced opinions.
- **Negative Sentiments:** The remaining 10% were negative, pointing to specific issues or dissatisfaction among a minority of users.

Sentiment Analysis by Review Rating

The sentiment scores were analyzed in relation to the review ratings (1 to 5 stars):

- **5-Star Reviews:** Predominantly positive sentiment, aligning with the high ratings.
- **4-Star Reviews:** Mostly positive, but with a slight increase in neutral sentiments compared to 5-star reviews.
- **3-Star Reviews:** Mixed sentiments, with a balance between positive and neutral.
- **2-Star and 1-Star Reviews:** Largely negative, as expected, with lower combined positive scores.

Key Insights

- **High Customer Satisfaction:** The overall positive sentiment indicates strong customer satisfaction with Intel processors, particularly for their performance and value for money.
- **Areas for Improvement:** The negative and neutral reviews point to areas where Intel could improve, such as compatibility issues and customer service.

Conclusions

This project implemented sentiment analysis on Intel processor reviews using VADER and RoBERTa models, accurately categorizing sentiments as positive, neutral, or negative. The results showed a high prevalence of positive sentiments, indicating customer satisfaction with performance and value. However, neutral and negative reviews highlighted issues with compatibility and customer service. These insights are valuable for Intel's strategic decisions, helping to enhance product features and improve customer service, ultimately leading to higher satisfaction and loyalty.

References

1. *https://www.intel.com/content/www/us/en/homepage.html*
2. *https://www.amazon.com/*
3. *https://github.com/topics/amazon-reviews-sentiment-analysis*
4. *https://www.geeksforgeeks.org/amazon-product-reviews-sentiment-analysis-in-python/*
5. *https://www.analyticsvidhya.com/blog/2023/03/analysis-of-amazon-review-using-vader-roberta-and-nltk/*
6. *https://ieeexplore.ieee.org/document/*

Appendices

Appendix A: Data Collection Methodology

To gather the data for sentiment analysis, we scraped user reviews from Amazon for Intel processors. The methodology involved the following steps:

1. **Web Scraping:** We used Python's `requests` library to retrieve HTML content from Amazon pages and `BeautifulSoup` for parsing the HTML.
2. **Review Extraction:** Reviews were extracted by locating HTML elements containing review text and ratings.
3. **Data Storage:** Extracted reviews were stored in a structured format using Pandas DataFrame and exported to an Excel file for further analysis.

Appendix B: Sentiment Analysis Implementation

The sentiment analysis process utilized two models: VADER and RoBERTa.

1. **VADER Sentiment Analysis:**
   - VADER (Valence Aware Dictionary and sEntiment Reasoner) was used for its effectiveness in analyzing social media text.
   - It provided scores for positive, neutral, and negative sentiments.
2. **RoBERTa Sentiment Analysis:**
   - RoBERTa (Robustly optimized BERT approach) model, specifically trained for sentiment analysis, was used to enhance sentiment prediction accuracy.
   - It provided probabilities for negative, neutral, and positive sentiments.
3. **Combined Sentiment Score:**
   - Scores from both models were averaged to form a combined sentiment score, providing a more robust sentiment classification.

Appendix C: Feature Extraction

To identify common themes within the reviews, we used clustering algorithms and feature extraction techniques.

1. **Text Preprocessing:**
   - Tokenization, lowercasing, stop word removal were applied to the review texts.
2. **TF-IDF Feature Extraction:**
   - TF-IDF was used to convert text data into numerical features, emphasizing important words.

Appendix D: Tools and Libraries Used

1. **Programming Language:**
   - Python was used for all data collection, preprocessing, analysis, and visualization tasks.
2. **Libraries:**
   - **Web Scraping:** `requests`, `BeautifulSoup`
   - **Data Processing:** `pandas`, `numpy`
   - **NLP and Sentiment Analysis:** `nltk`, `textblob`, `transformers` (for RoBERTa), `nltk.sentiment.vader` (for VADER)
   - **Clustering and Feature Extraction:** `scikit-learn`
   - **Visualization:** `matplotlib`, `seaborn`, `plotly`

Appendix E: Challenges and Limitations

1. **Data Quality:**
    - Some reviews were incomplete or lacked sufficient detail, affecting the sentiment analysis accuracy.
2. **Model Limitations:**
    - VADER and RoBERTa, while effective, may not capture the full complexity of human emotions and context in reviews.
3. **Scalability:**
    - Web scraping large volumes of reviews can be time-consuming and may face limitations due to website restrictions.

Appendix F: Future Work

1. **Enhanced Sentiment Analysis:**
    - Incorporate more advanced models and techniques, such as BERT and GPT, for improved sentiment analysis.
2. **Broader Data Collection:**
    - Expand the dataset by including reviews from multiple platforms and covering a wider range of products.
3. **Real-time Sentiment Monitoring:**
    - Develop a system for real-time monitoring and analysis of user sentiments to provide timely insights and feedback.