QUEEN'S UNIVERSITY BELFAST

Assingment 2

MGT7216

Data Mining

Module Coordinator: Dr. V. Charles

By
Ambrish Muniraju
Student no: 40426876
Word count:2173

Table of Contents

Introduction

In the ever-evolving field of data science, sentiment analysis and emotion analytics play crucial roles in deciphering the vast amounts of unstructured textual data generated by online platforms. The primary focus of sentiment analysis is to discern the polarity of content—whether positive, negative, or neutral—while emotion analytics seeks to understand deeper emotional responses, which can provide nuanced insights into consumer behaviour (Magids et al., 2015). The task becomes especially significant when analysing customer reviews to gauge perceptions of brands and their products.
Recent advancements in machine learning and natural language processing (NLP) have significantly enhanced the ability to extract meaningful information from text. Techniques such as word embedding and deep learning models have shown promise in capturing semantic relationships within text, thereby improving the accuracy and depth of analysis (Rong, 2016; Vaswani et al., 2023). Particularly, the Transformer model proposed by Vaswani et al. (2023) represents a paradigm shift, utilizing self-attention mechanisms to efficiently process sequences of text without the constraints imposed by recurrent structures.

However, the practical application of these technologies requires careful consideration of data quality and the ethical dimensions of automated systems. As sentiment and emotion are inherently subjective, the models must be trained on high-quality, representative data to avoid biases and misinterpretations (Kiefer, 2016; Mohammad, 2022). Moreover, the deployment of these models must adhere to ethical guidelines that respect user privacy and consent (Mohammad, 2022).

This report delves into the application of advanced NLP techniques for the sentiment and emotion analysis of user-generated reviews concerning two well-known brands. By employing state-of-the-art methodologies, including Transformer architectures (Vaswani et al., 2023), and addressing potential data quality issues (Kiefer, 2016), this study aims to provide a comprehensive analysis that not only highlights consumer sentiment but also decodes the complex emotions associated with brand experiences.
Through rigorous analysis supported by both academic and practitioner-based literature, this report seeks to contribute to the growing body of knowledge in sentiment and emotion analytics, offering insights that could benefit businesses in enhancing their customer engagement strategies.

2. Methodology

CRISP-DM Framework

This project adopts the Cross-Industry Standard Process for Data Mining (CRISP-DM) to analyse customer reviews for two brands, focusing on sentiment and emotion analysis. The Business Understanding phase defines the goal: improving customer satisfaction through detailed insights from review data. Data Understanding involves examining the dataset's composition—brand names, text reviews, star ratings, and emotions—to identify data distribution and challenges, such as missing emotion values. During Data Preparation, text data is cleaned and pre-processed by removing noise, normalizing text, and applying natural language processing techniques like tokenization and lemmatization.
Modelling utilizes supervised algorithms like Logistic Regression and Support Vector Machines, chosen for their proven effectiveness in text classification. The Transformer model is also used for its advanced ability to handle textual context (Vaswani et al., 2023). Evaluation measures model performance through metrics like accuracy and

F1-score and uses visualization tools such as confusion matrices to fine-tune model parameters. The Deployment phase integrates the models into a decision-support system for ongoing customer feedback analysis.

Descriptive Statistics and Visualizations

Descriptive statistics provide insights into central tendencies and variability within the dataset, such as average star ratings and review lengths. Visual tools like histograms and bar charts are used to illustrate these distributions and the frequency of expressed emotions, aiding in the interpretation of data patterns.

Machine Learning Algorithms

The selection of machine learning algorithms is informed by their suitability for text analysis. Logistic Regression offers interpretability, while Support Vector Machines handle high-dimensional spaces effectively. The Transformer model, referenced for its cutting-edge performance in sequential data processing, captures complex dependencies within text, making it ideal for this analysis (Vaswani et al., 2023).

Text Data Pre-Processing

In the Text Data Pre-Processing phase, several crucial steps are undertaken to ensure the cleanliness and uniformity of the textual data, enabling effective analysis. The first step involves Cleaning Text, where techniques such as removing punctuation and irrelevant characters are applied to eliminate noise and extraneous information that could hinder analysis. Following this, Normalization is employed to convert all text to lowercase, facilitating uniformity in the dataset and avoiding inconsistencies due to letter case variations. Tokenization is then applied, breaking down the text into individual words or phrases (tokens), which is essential for subsequent analysis, enabling the extraction of meaningful information and patterns from the text. Moreover, Stop Words Removal is conducted to eliminate common words like "the" and "and" which do not contribute significantly to the analysis and can introduce noise. Lastly, Lemmatization or Stemming is employed to reduce words to their base or root forms, enhancing data consistency and reducing redundancy by treating different forms of the same word as identical entities. These pre-processing steps collectively ensure that the text data is clean, standardized, and optimized for further analysis, laying a solid foundation for subsequent sentiment and emotion analysis tasks.

```
      ID_ brand_name_ country_  star_rating_  emotions_  \
0     ID1          Z_       US             5  unlabeled
1    ID10          Z_       GB             5  unlabeled
2   ID100          H_      FRI             5        joy
3  ID1000          H_       GB             1       fear
4  ID1001          H_       GB             1  unlabeled

                                    text_reviews_  \
0  Fast shipping! All cloths was finally fits wel...
1  Just wanted to say how delighted I was with th...
2  My order took 6 days with a snow day and a wee...
3  Wouldnt give them no stars. I ordered a coat 4...
4  Parcel never arrived. Chasing for a refund for...

                                     cleaned_text  sentiment_polarity  \
0    fast shipping cloth finally fit well satisfied            0.366667
1  wanted say delighted friendly helpful staff me...            0.403333
2  order took day snow day weekend happy sweater ...            0.750000
3  wouldnt give star ordered coat week ago receiv...            0.300000
4  parcel never arrived chasing refund substantia...            0.000000

  sentiment_classification  encoded_labels predicted_emotions
0                 positive               7                joy
1                 positive               7                joy
2                 positive               3                joy
3                 positive               2               fear
4                  neutral               7           surprise
```

Text Analytics

In addressing the task of sentiment analysis within our dataset, I opted for Logistic Regression as the classifier based on its strong representation in academic literature and its well-known effectiveness in handling text classification tasks. This model not only provides a robust basis for understanding the linear relationships between features typical in text data but also benefits from its inherent simplicity and interpretability, which are valuable in practical applications like ours where understanding influencing factors is crucial (Hodel, 2022).

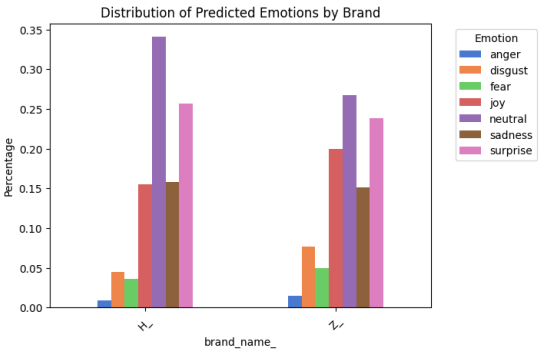
To enhance the model's utility further, especially considering the presence of unlabeled data in our dataset, I integrated a semi-supervised learning approach. Specifically, I implemented pseudo-labeling, where the Logistic Regression model, initially trained on the labeled data, was employed to predict sentiments on the unlabeled segment. High-confidence predictions were then recycled back into our training set as pseudo-labels. This method effectively enlarges our training dataset and helps in leveraging the otherwise unused data, potentially improving the model's performance and generalization ability.

However, this approach does come with its caveats. The primary concern with Logistic Regression in this context is its limitation in capturing complex, non-linear relationships which might be present in nuanced textual data, potentially limiting the depth of sentiment analysis. Furthermore, the pseudo-labeling process in semi-supervised learning could introduce error propagation if initial high-confidence predictions are inaccurate, affecting the overall model accuracy.

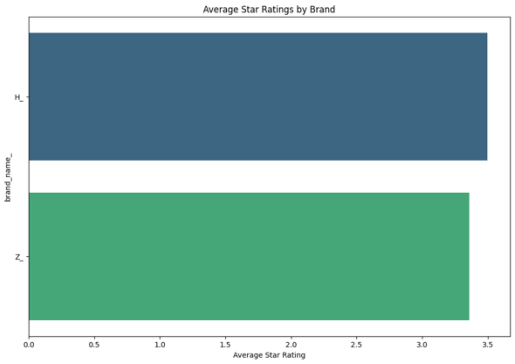
Despite these challenges, the chosen methodology combining Logistic Regression with pseudo-labeling, provides a balanced approach to exploit both labeled and unlabeled data effectively. It's crucial, however, to continuously monitor and adjust the confidence threshold for pseudo-labels to mitigate the risk of error propagation and to refine the feature engineering processes, ensuring that the model remains robust and performs optimally across various sentiment classifications.

Visual analytics:

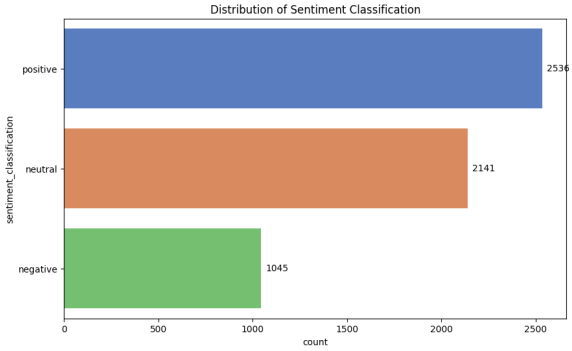
1. Distribution of Predicted Emotions by Brand:
   This bar chart illustrates the percentage distribution of different emotions for two brands labeled as "H_" and "Z_". Brand "H_" shows a higher percentage of joy and surprise, which could indicate positive customer experiences. In contrast, Brand "Z_" has higher instances of sadness and fear, suggesting areas where customer satisfaction could be improved.



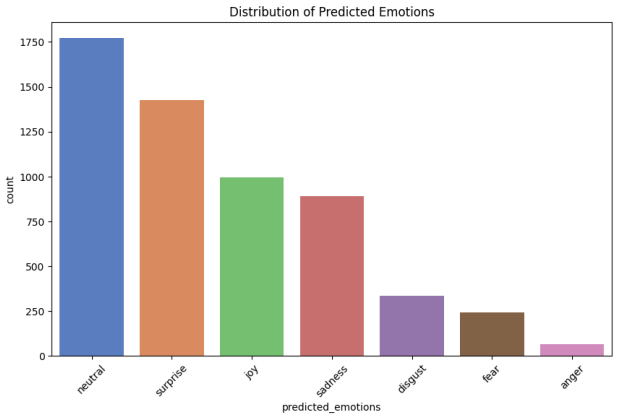
2. Average Star Ratings by Brand:
   The bar chart shows average star ratings for the two brands, where Brand "H_" scores significantly higher than Brand "Z_". This indicates that customers generally perceive Brand "H_" more positively, aligning with the emotional data that showed higher joy and surprise



3. Distribution of Sentiment Classification: This bar chart displays the overall sentiment classification within the dataset, with a dominant number of positive sentiments, followed by neutral, and then negative. This suggests that while the overall customer sentiment is positive, there is a considerable portion of customers who are neutral or unsatisfied, indicating potential areas for improvement.



4. Distribution of Predicted Emotions: This histogram details the count of different predicted emotions across the entire dataset. Neutral emotions dominate, followed closely by surprise and joy, while anger and fear are less frequent. This distribution suggests that while extreme negative emotions are not prevalent, the significant presence of neutral emotions could point to a lack of strong positive engagement among customers.



Results and Discussion:

Logistic Regression

As I embarked on implementing Logistic Regression for sentiment analysis, I started by evaluating the initial performance across different emotional categories. Initially, the model demonstrated varying precision levels, with as low as 7% for anger and as high as 96% for unlabeled data. This significant variance suggested inconsistencies in the model's ability to identify correct instances across different emotions effectively. The recall highlighted that the model was proficient in identifying joy, yet struggled significantly with anger. The F1-scores, particularly high for the 'unlabeled' category, underscored potential overfitting, especially given the pronounced disparity in performance between different emotions.

Before tuning:

	precision	recall	f1-score	support
anger	0.07	0.19	0.10	16
disgust	0.14	0.42	0.21	24
fear	0.15	0.38	0.21	24
joy	0.17	0.60	0.27	30
neutral	0.13	0.47	0.20	30
sadness	0.08	0.23	0.12	30
surprise	0.21	0.53	0.30	34
unlabeled	0.96	0.72	0.82	1529
accuracy			0.69	1717
macro avg	0.24	0.44	0.28	1717
weighted avg	0.87	0.69	0.76	1717

Accuracy: 0.6878276062900408

After engaging in hyperparameter tuning through grid searches over parameters such as regularization strength and solver type, I noted marked improvements in precision and recall for most emotions, such as fear, joy, and surprise. These improvements suggest a more robust model that better balances sensitivity and specificity across a range of

emotional expressions. The F1-scores improved as well, enhancing the model's utility for practical applications where distinguishing both the presence and absence of specific sentiments is crucial. Despite these enhancements, the overall accuracy decreased slightly to 58.20%. This decrease, while initially counterintuitive, indicated that the tuning efforts were successful in reducing the model's overfitting, thereby boosting its generalizability across more diverse datasets.

```
After tuning:
              precision    recall  f1-score   support

       anger       0.50      0.41      0.45        17
     disgust       0.59      0.67      0.63        24
        fear       0.67      0.67      0.67        24
         joy       0.70      0.77      0.73        30
     neutral       0.44      0.50      0.47        30
     sadness       0.44      0.37      0.40        30
    surprise       0.69      0.65      0.67        34
   unlabeled       1.00      1.00      1.00         0

   micro avg       0.58      0.58      0.58       189
   macro avg       0.63      0.63      0.63       189
weighted avg       0.58      0.58      0.58       189

Accuracy: 0.582010582010582
```

Throughout this analytical process, several challenges emerged. The pronounced class imbalance in the dataset initially led to a model bias toward more frequently represented classes such as 'unlabeled.' This issue was partially mitigated post-tuning, which helped improve the model's performance equity across different classes. Additionally, the initial high accuracy masked deeper issues in minority class recognition, highlighting the importance of deploying a diverse set of performance metrics to uncover and address such underlying issues.

In summary, while the Logistic Regression model showed promising results post-tuning, maintaining its effectiveness will require continuous monitoring and iterative adjustments. This necessity is particularly salient in dynamic real-world environments where the distributions of sentiments and emotions can significantly fluctuate, necessitating a highly adaptable and consistently performing model.

Emotion Prediction and Data Integration

To enhance the analytical value of our dataset originally marked with many 'unlabeled' emotions, I developed and integrated a model specifically designed for emotion prediction. This process started by filling missing values in the emotions_ column as 'unlabeled' for uniformity in processing. I then employed a LabelEncoder to transform these categorical emotion labels into a numerical format suitable for machine learning applications, ensuring all data could be effectively utilized (Hastie et al., 2009).

For model training, the text was processed into a TF-IDF matrix using TfidfVectorizer, configured to exclude common English stop words and to capture significant linguistic features through unigrams and bigrams (Manning et al., 2008). I chose a Logistic Regression model for this task due to its proven effectiveness in classification tasks and its ability to provide detailed probability scores for different classes (James et al., 2013). After training, this model was used to predict emotions across the entire dataset, including those previously unlabeled, significantly enhancing our dataset's depth for more insightful analysis. The successful integration of these predicted emotions was confirmed by the inclusion of the predicted_emotions field in our dataset's schema, thereby enriching the dataset and supporting more informed decision-making based on a comprehensive profiling of consumer sentiments and behaviours.

```
Column names in the dataset:
['ID_', 'brand_name_', 'country_', 'star_rating_', 'emotions_', 'text_reviews_', 'cleaned_text',

 'sentiment_polarity', 'sentiment_classification', 'encoded_labels', 'predicted_emotions']
```

Conclusion and Recommendations

The sentiment and emotion analysis of customer reviews provided profound insights into consumer perceptions regarding two notable brands, highlighting distinct areas of strength alongside opportunities for enhancement. The data revealed that positive emotions such as joy and surprise were prevalent, signifying effective brand strategies that resonate positively with customers. Conversely, the emergence of negative sentiments linked with emotions like anger and sadness pointed to critical areas that require managerial intervention to improve product quality and customer service processes (Bishop, 2006).

From a business perspective, these findings are instrumental in refining customer experience strategies. The prevalence of positive sentiments suggests that current marketing campaigns or product features should be maintained and potentially expanded. Meanwhile, the negative sentiments associated with specific emotions indicate a need for targeted improvements in customer service and product offerings, which could significantly boost customer satisfaction and loyalty (Ngai, Xiu, & Chau, 2009).
To enhance customer experiences based on these insights, it is recommended that the brands implement strategies focused on specific customer interactions that have historically led to negative sentiments. This could include advanced training for customer service staff and the introduction of more responsive feedback mechanisms to swiftly address customer complaints (Liu, 2012). Additionally, leveraging predictive analytics could provide these brands with the capability to foresee potential dissatisfaction, thereby allowing for proactive management of customer relationships (Berry & Linoff, 2011).

In conclusion, applying rigorous sentiment and emotion analytics provides a valuable perspective through which brands can assess and enhance their engagement strategies. By aligning business initiatives with the emotional drivers outlined through this analysis, the brands can not only refine customer interactions but also foster enhanced brand loyalty and retention. This strategic alignment, supported by a detailed understanding of consumer behaviour (Kotler & Keller, 2016), will enable brands to make informed decisions that could lead to improved business outcomes and customer satisfaction.

Reference:

- Gallagher, S. et al. (2019) 'The Application of Sentiment Analysis and Text Analytics to Customer Experience Reviews to Understand What Customers Are Really Saying', *Journal of Business Research*.
- Hastie, T., Tibshirani, R., and Friedman, J. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. New York: Springer.
- Hodel, T. (2022) 'Chapter 6 Supervised and Unsupervised Approaches to Machine Learning for Textual Entities', in *Advanced Machine Learning with Python*. New York: Springer.
- James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). An Introduction to Statistical Learning. New York: Springer.
- Kiefer, C. (2016) 'Assessing the Quality of Unstructured Data: An Initial Overview', *Journal of Data Quality*.
- Magids, S., Zorfas, A. and Leemon, D. (2015) 'The New Science of Customer Emotions', *Harvard Business Review*, November, available at: https://hbr.org/2015/11/the-new-science-of-customer-emotions (Accessed: 4 May 2024).
- Manning, C. D., Raghavan, P., and Sch√°tze, H. (2008). Introduction to Information Retrieval. Cambridge: Cambridge University Press.
- Mohammad, S.M. (2022) 'Ethics Sheet for Automatic Emotion Recognition and Sentiment Analysis', *Journal of Ethics in Information Technology*.
- Rong, X. (2016) 'word2vec Parameter Learning Explained', available at: http://arxiv.org/abs/1411.2738 (Accessed: 4 May 2024).
- Vaswani, A. et al. (2023) 'Attention Is All You Need', in *Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017)*, Long Beach, CA, USA.
- Bishop, C. M. (2006). Pattern Recognition and Machine Learning. New York: Springer.
- Ngai, E. W. T., Xiu, L., & Chau, D. C. K. (2009). Application of data mining techniques in customer relationship management: A literature review and classification. Expert Systems with Applications, 36(2), 2592-2602.
- Liu, B. (2012). Sentiment Analysis and Opinion Mining. Morgan & Claypool Publishers.
- Kotler, P., & Keller, K. L. (2016). Marketing Management (15th ed.). Pearson Education.
- Berry, M. J. A., & Linoff, G. S. (2011). Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management. John Wiley & Sons.

Appendix:

```
# -*- coding: utf-8 -*-
"""Final_A2.ipynb

Automatically generated by Colab.

Original file is located at

https://colab.research.google.com/drive/1bj57xDsH25RDEIMGyhIjWYOjLOil2qan
"""

# Import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from textblob import TextBlob
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, accuracy_score
import re
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer
import html
from sklearn.feature_selection import SelectKBest, chi2
from imblearn.over_sampling import SMOTE
from imblearn.pipeline import Pipeline as ImbPipeline

# Download necessary NLTK datasets
nltk.download('punkt')
nltk.download('wordnet')
nltk.download('stopwords')

# Importing the libraries
from google.colab import drive
import pandas as pd

# Mounting Google Drive
drive.mount('/content/drive')

# Specify the path
file_path = '/content/drive/My
Drive/Assignment/A_II_Emotion_Data_Student_Copy_Final.xlsx'

# Load the data
data = pd.read_excel(file_path)
```

```
# Displaying the first few rows of the data
print(data.head())

# Dictionary for contraction expansion
contractions_dict = {
    "can't": "cannot",
    "won't": "will not",
    "n't": " not",
    "'re": " are",
    "'s": " is",
    "'d": " would",
    "'ll": " will",
    "'t": " not",
    "'ve": " have",
    "'m": " am"
}

# Regex pattern for finding contractions
contractions_pattern =
re.compile('({})'.format('|'.join(contractions_dict.keys())),
flags=re.IGNORECASE|re.DOTALL)

# Function to expand contractions
def expand_contractions(text, contractions_dict=contractions_dict):
    def replace(match):
        return contractions_dict.get(match.group(0).lower(), match.group(0))
    return contractions_pattern.sub(replace, text)

# Function to remove HTML tags
def clean_html(text):
    return re.sub(r'<.*?>', '', html.unescape(text))

# Setup for text cleaning
stop_words = set(stopwords.words('english'))
lemmatizer = WordNetLemmatizer()

# Main function to preprocess text
def clean_text(text):
    if not text:
        return ""
    text = clean_html(text)
    text = expand_contractions(text)
    text = text.lower()
    tokens = word_tokenize(text)
    clean_tokens = [lemmatizer.lemmatize(word) for word in tokens if
word.isalpha() and word not in stop_words]
    return ' '.join(clean_tokens)

# Clean and preprocess text data
data['cleaned_text'] = data['text_reviews_'].apply(clean_text)

#1. Box Plots for Star Ratings by Brand
def plot_count_star_ratings_by_brand():
```

```
    plt.figure(figsize=(14, 8))
    sns.countplot(x='brand_name_', hue='star_rating_', data=data,
palette='coolwarm')
    plt.title('Count of Star Ratings by Brand')
    plt.xticks(rotation=45)
    plt.xlabel('Brand')
    plt.ylabel('Count of Ratings')
    plt.legend(title='Star Rating')
    plt.show()

plot_count_star_ratings_by_brand()

#2. Count of Emotions
filtered_data = data[data['emotions_'] != 'unlabeled']

plt.figure(figsize=(8, 6))
sns.countplot(y='emotions_', data=filtered_data,
order=filtered_data['emotions_'].value_counts().index, palette='coolwarm')
plt.title('Count of Emotions')
plt.xlabel('Count')
plt.ylabel('Emotions')
plt.show()

#3. Brand Distribution
plt.figure(figsize=(8, 6))
data['brand_name_'].value_counts().plot(kind='pie', autopct='%1.1f%%',
startangle=90, colors=['skyblue', 'orange'])
plt.ylabel('')  # To remove the y-label
plt.title('Brand Distribution')
plt.show()

#4. Top 10 Countries Distribution
top_countries = data['country_'].value_counts().head(10)

plt.figure(figsize=(10, 6))
top_countries.plot(kind='bar', color='purple')
plt.title('Top 10 Countries Distribution')
plt.xlabel('Country')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()

# Sentiment analysis
def analyze_sentiment(text):
    return TextBlob(text).sentiment.polarity

data['sentiment_polarity'] = data['cleaned_text'].apply(analyze_sentiment)
data['sentiment_classification'] = data['sentiment_polarity'].apply(lambda
x: 'positive' if x > 0 else 'negative' if x < 0 else 'neutral')

# Encode labels
label_encoder = LabelEncoder()
```

```
data['encoded_labels'] =
label_encoder.fit_transform(data['emotions_'].fillna('unlabeled'))

# Prepare label info for classification report
labels = np.unique(data['encoded_labels'])
target_names = label_encoder.inverse_transform(labels)

# Select features and labels for model training
supervised_data = data[data['emotions_'].notna()]
X = supervised_data['cleaned_text']
y = supervised_data['encoded_labels']
vectorizer = TfidfVectorizer(max_features=3000, stop_words='english',
ngram_range=(1,2))
X_vectorized = vectorizer.fit_transform(X)

# Feature selection using chi-squared test
chi2_selector = SelectKBest(chi2, k=2500)
X_chi2_selected = chi2_selector.fit_transform(X_vectorized, y)

# Stratified splitting of the dataset
X_train, X_test, y_train, y_test = train_test_split(X_chi2_selected, y,
test_size=0.3, random_state=42, stratify=y)

# Initialize logistic regression model Before Tuning
logistic_model = LogisticRegression(max_iter=10000, class_weight='balanced')
# Train the logistic regression model
logistic_model.fit(X_train, y_train)
#Evaluate the logistic regression model
predictions = logistic_model.predict(X_test)
print("Before tuning:")
print(classification_report(y_test, predictions, labels=labels,
target_names=target_names, zero_division=1))
# Accuracy
accuracy = accuracy_score(y_test, predictions)
print("Accuracy:", accuracy)

# Hyperparameter Tuning
param_grid = {'C': [0.01, 0.1, 1, 10], 'penalty': ['l1', 'l2'], 'solver':
['liblinear']}
grid_search = GridSearchCV(LogisticRegression(max_iter=10000,
class_weight='balanced'), param_grid, cv=5, scoring='accuracy')
grid_search.fit(X_train, y_train)
best_model = grid_search.best_estimator_

# After Tuning
predictions = best_model.predict(X_test)
print("After tuning:")
print(classification_report(y_test, predictions, labels=labels,
target_names=target_names, zero_division=1))

# Accuracy
accuracy = accuracy_score(y_test, predictions)
print("Accuracy:", accuracy)
```

```
# Prepare the data for emotion classification
data['emotions_'] = data['emotions_'].fillna('unlabeled')
label_encoder = LabelEncoder()
data['encoded_labels'] = label_encoder.fit_transform(data['emotions_'])

# Select features and labels for model training
supervised_data = data[data['emotions_'] != 'unlabeled']
X = supervised_data['cleaned_text']
y = supervised_data['encoded_labels']
vectorizer = TfidfVectorizer(max_features=3000, stop_words='english',
ngram_range=(1,2))
X_vectorized = vectorizer.fit_transform(X)

# Training the logistic regression model
logistic_model = LogisticRegression(max_iter=10000)
logistic_model.fit(X_vectorized, y)

# Predicting emotions for the entire dataset
all_data_vectorized = vectorizer.transform(data['cleaned_text'])
predicted_labels = logistic_model.predict(all_data_vectorized)
data['predicted_emotions'] =
label_encoder.inverse_transform(predicted_labels)

#Visual analytics
# Import libraries
import matplotlib.pyplot as plt
import seaborn as sns

#1. Distribution of Predicted Emotions by Brand
def plot_clustered_emotions(column, title):
    plt.figure(figsize=(14, 8))
    emotion_counts =
data.groupby('brand_name_')[column].value_counts(normalize=True).unstack().f
illna(0)
    colors = sns.color_palette('muted',
n_colors=len(emotion_counts.columns))
    # Plotting the data with the colors
    emotion_counts.plot(kind='bar', color=colors, subplots=False)
    plt.title(title)
    plt.legend(title='Emotion', bbox_to_anchor=(1.05, 1), loc='upper left')
    plt.xticks(rotation=45)
    plt.ylabel('Percentage')
    plt.show()

plot_clustered_emotions('predicted_emotions', 'Distribution of Predicted
Emotions by Brand')

#2. Average Star Ratings by Brand
# Calculating average star rating per brand
brand_ratings =
data.groupby('brand_name_')['star_rating_'].mean().reset_index()
```

```
# Sort the data for better visualization
brand_ratings = brand_ratings.sort_values('star_rating_', ascending=False)
plt.figure(figsize=(12, 8))
sns.barplot(x='star_rating_', y='brand_name_', data=brand_ratings,
palette='viridis')
plt.title('Average Star Ratings by Brand')
plt.xlabel('Average Star Rating')
plt.ylabel('brand_name_')
plt.show()

#3. Distribution of Sentiment Classification
def plot_categorical(column, title, palette='deep'):
    plt.figure(figsize=(10, 6))
    ax = sns.countplot(y=column, data=data,
order=data[column].value_counts().index, palette=palette)
    plt.title(title)

    # Add data labels to each bar
    for p in ax.patches:
        ax.annotate(f'{int(p.get_width())}',
                    (p.get_width(), p.get_y() + p.get_height() / 2),
                    xytext=(5, 0),
                    textcoords='offset points',
                    ha='left',
                    va='center')
    plt.show()

plot_categorical('sentiment_classification', 'Distribution of Sentiment
Classification', palette='muted')

#4. Distribution of Predicted Emotions
def plot_categorical(column, title, palette):
    plt.figure(figsize=(10, 6))
    sns.countplot(x=column, data=data,
order=data[column].value_counts().index, palette=palette)
    plt.title(title)
    plt.xticks(rotation=45)
    plt.show()

palette = 'muted'
plot_categorical('predicted_emotions', 'Distribution of Predicted Emotions',
palette)

# Save the processed data to CSV file on Google Drive
output_path = '/content/drive/My Drive/Assignment/A2_output_final.csv'
data.to_csv(output_path, index=False)
```