[DTI5125 [EG] Data Science Applications
Group 10, Final Project

Sentiment Analysis

Names:

Rim Mahmoud Sayed Touny

Nada Mohamed Zakaria

Dina Ibrahim Mohammady

Sema Abdel Nasser Mosaad

Abstract

This sentiment analysis project focuses on analyzing customer sentiments across online platforms using NLP and a smart chatbot using Dialogflow Essentials.
It introduces innovative features like domain-specific customization, real-time feedback, and multilingual support. After data preparation and text preprocessing, the TF-BOW-LDA transformation is used. Various ML models are applied, and the evaluation shows their performance. The project empowers e-commerce businesses to understand customer perceptions, make data-driven decisions, and enhance customer satisfaction for improved performance.

Problem Formulation/Explanation

Understanding customer sentiments and perceptions about your product or brand is vital for business success. With feedback coming from product reviews, emails, and social media, sentiment analysis plays a crucial role in gauging public opinion. This project aims to leverage advanced technology and Natural Language Processing to analyze customer sentiments across online platforms. By introducing a smart chatbot for real-time feedback and domain-specific customization, businesses can enhance customer satisfaction and improve performance in the competitive e-commerce environment.

Legacy companies like Nokia, Kodak, and BlackBerry vanished from today's world, while companies like Apple, Google Photos, and Samsung thrived. A common pattern among these successful companies is their ability to understand customers' feelings and respond to their feedback, ensuring continued success and excellence in the market. Being aware of the tone and perception the public has of your brand is one of the most important factors for success in business, and this is where sentiment analysis comes into play. By implementing sentiment analysis in this project, we can gain valuable insights into customer opinions, sentiments, and feedback. This enables businesses to make data-driven decisions, identify areas of improvement, and cater to customer needs effectively. The innovative use of domain-specific customization and real-time feedback through the smart chatbot makes this sentiment analysis tool an essential asset for enhancing customer satisfaction and achieving business success in the dynamic e-commerce landscape.

Data Preparation

❖ Data Cleaning:

- **Handling Missing Data**: The dataset has a very low percentage of missing cells (less than 0.1%) (10 values in `reviews.title`), so we can safely drop or impute those missing values based on the specific context.
- **Handling Duplicate Rows:** The dataset has 1.5% duplicate rows, which can be removed to ensure data integrity.

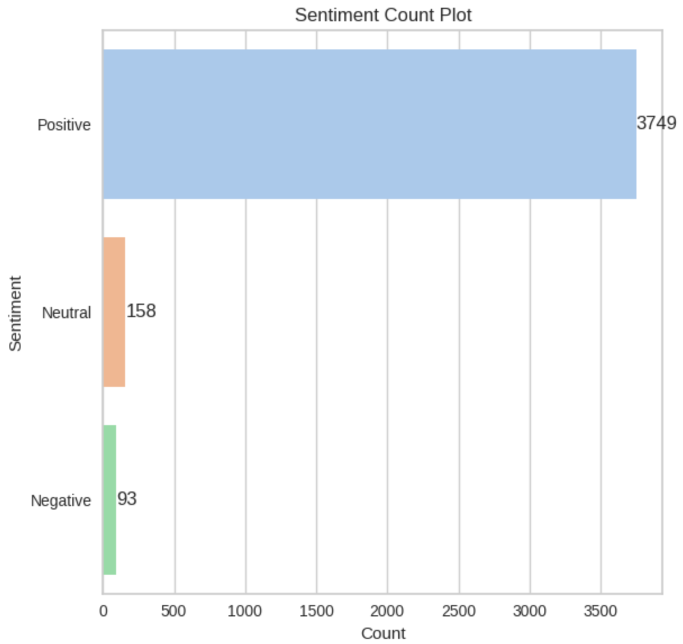
❖ **Renaming and Dropping Columns**: We have renamed the columns `'reviews.text,' 'reviews.title,' and 'reviews.date' to 'reviews_text,' 'reviews_title,' and 'reviews_date,'` respectively. Additionally, we dropped the columns 'name,' 'brand,' 'categories,' '`primaryCategories`,' and '`reviews.date`' from the dataset.

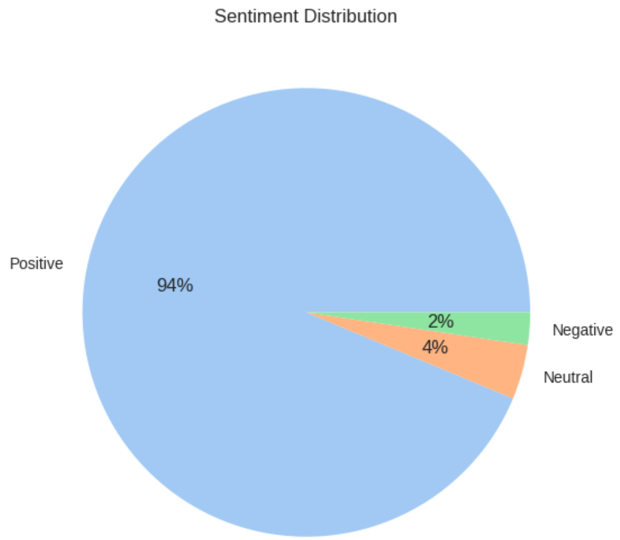
❖ **Sentiment Label Encoding:** We created a mapping dictionary for sentiment labels and encoded the 'sentiment' column into numerical form (1 for 'Positive,' -1 for 'Negative,' and 0 for 'Neutral').

❖ **Create new Column 'Polarity Scores':** We apply the `SentimentIntensityAnalyzer` to the `'reviews_text'` column to calculate polarity scores for each review. Polarity scores represent the sentiment of the text as a continuous value between -1 (negative) and 1 (positive).

	reviews_text	reviews_title	sentiment	polarity_score
0	Purchased on Black FridayPros - Great Price (e...	Powerful tablet	Positive	0.97
1	I purchased two Amazon in Echo Plus and two do...	Amazon Echo Plus AWESOME	Positive	0.97
2	Just an average Alexa option. Does show a few ...	Average	Neutral	-0.33
3	very good product. Exactly what I wanted, and ...	Greatttttttt	Positive	0.75
4	This is the 3rd one I've purchased. I've bough...	Very durable!	Positive	0.18

- ❖ **Balancing Data** : The classes are imbalanced, you may consider applying techniques like SMOTE to balance the data.





Text Feature Engineering

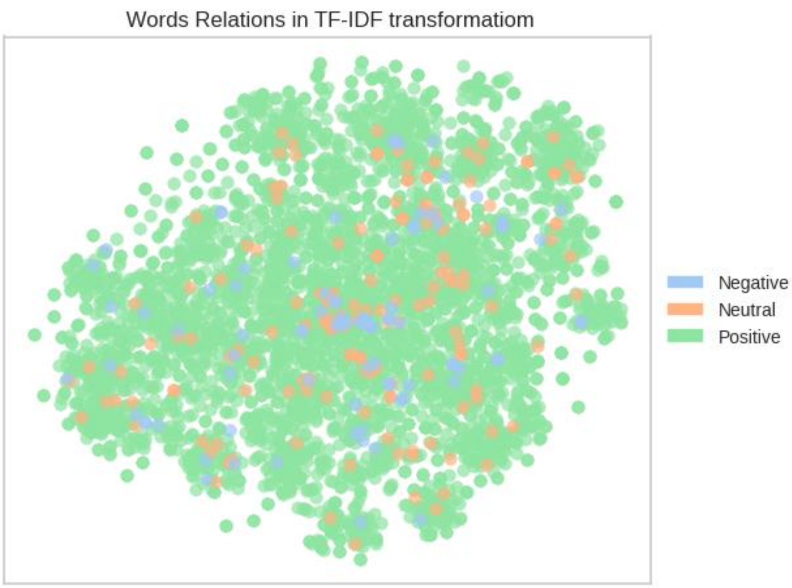
```python
# First step - tokenizing phrases
def text_preprocessing(dataframe, dependent_var):
  # Normalizing Case Folding - Uppercase to Lowercase
  dataframe[dependent_var] = dataframe[dependent_var].apply(lambda x: " ".join(x.lower() for x in str(x).split()))

  # Removing Punctuation
  dataframe[dependent_var] = dataframe[dependent_var].str.replace('[^\w\s]','')

  # Removing Numbers
  dataframe[dependent_var] = dataframe[dependent_var].str.replace('\d','')

  # StopWords
  sw = stopwords.words('english')
  dataframe[dependent_var] = dataframe[dependent_var].apply(lambda x: " ".join(x for x in x.split() if x not in sw))

  # Remove Rare Words
  temp_df = pd.Series(' '.join(dataframe[dependent_var]).split()).value_counts()
  drops = temp_df[temp_df <= 1]
  dataframe[dependent_var] = dataframe[dependent_var].apply(lambda x: " ".join(x for x in str(x).split() if x not in drops))

  # Lemmatize
  lemmatizer = WordNetLemmatizer()  # Create a lemmatizer object
  dataframe[dependent_var] = dataframe[dependent_var].apply(lambda x: " ".join([lemmatizer.lemmatize(word) for word in x.split()]))

  return dataframe
```

The function applies several text cleaning and normalization techniques to prepare the text data for further analysis and modeling. The steps involved in the text preprocessing function are as follows:

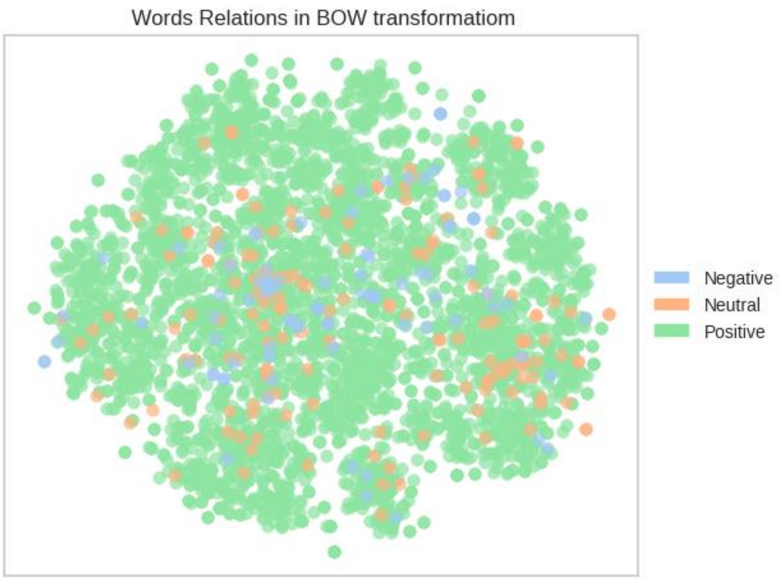
1. **Normalizing Case Folding**: Convert all text to lowercase to ensure consistent comparisons between words.
2. **Removing Punctuation**: Eliminate special characters and punctuation marks from the text to avoid any interference in analysis.
3. **Removing Numbers:** Exclude numerical digits from the text as they may not be relevant for certain tasks like sentiment analysis.
4. **Removing Stopwords:** Remove common words that do not carry much meaning (e.g., "the," "and," "is") using stopwords from the English language.
5. **Remove Rare Words:** Eliminate words that appear infrequently in the dataset, as they may not contribute significantly to the analysis.
6. **Lemmatization:** Convert words to their base or root form (lemmas) to reduce inflected words to a common base form. For example, "running," "runs," and "ran" will all be transformed to "run."

After text preprocessing, the data can be transformed into three different representations:

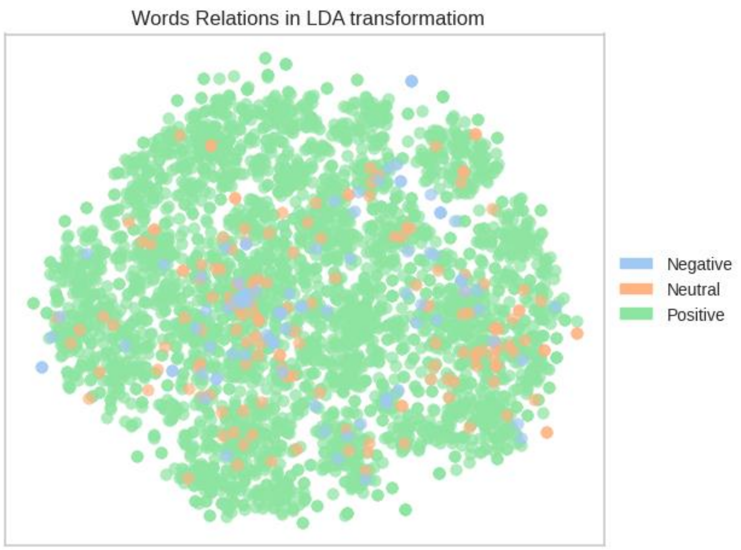
- **TF (Term Frequency):** Convert the text data into a bag-of-words representation, where each document is represented as a vector of word frequencies in the corpus.



- **Bag-of-Words (BOW):** Similar to TF, but it also ignores the frequency and considers only whether a word appears or not (binary representation).



- **LDA (Latent Dirichlet Allocation):** Perform topic modeling to extract latent topics from the text data. Each document is represented as a mixture of topics.



These transformations allow the text data to be represented in a numerical format, which can be used as input to various machine learning models for sentiment analysis, topic modeling, or other natural language processing tasks.

Modeling

Classification

TF-IDF Technique

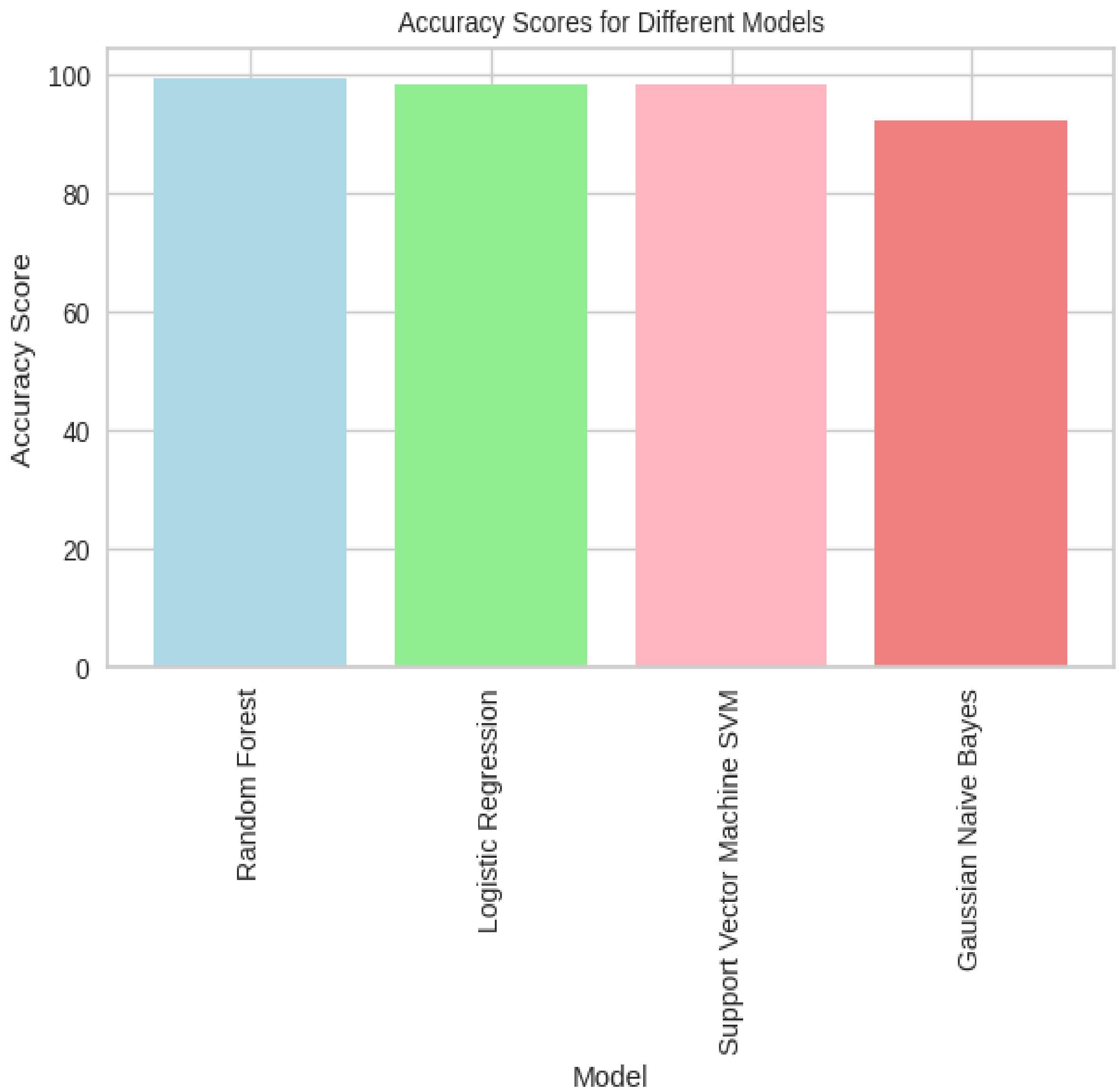
	Model	Accuracy_Score
0	Random Forest	99.366802
1	Logistic Regression	98.190864
2	Support Vector Machine SVM	98.145635
3	Gaussian Naive Bayes	92.130258

1. **Random Forest Accuracy**: The Random Forest model achieved the highest accuracy among the evaluated models when using the TF-IDF (Term Frequency-Inverse Document Frequency) transformation. TF-IDF takes into account not only the occurrence of words but also their importance in the entire dataset. Random Forest's ensemble of decision trees effectively leverages these weighted word features, resulting in a remarkably high accuracy.

2. **Logistic Regression Accuracy**: Logistic Regression achieved the second-highest accuracy. In the context of TF-IDF, Logistic Regression benefits from the weighted word features that help capture more nuanced relationships between words and their impact on sentiment. This contributes to its strong performance in accurately classifying sentiments.

3. **Support Vector Machine (SVM)** Accuracy: SVM achieved a slightly lower accuracy compared to Logistic Regression. Although SVM aims to find a hyperplane that maximizes the margin between classes, the nuanced word importance captured by TF-IDF might not align perfectly with SVM's linear separation assumption. This could lead to a minor drop in accuracy.

4. **Gaussian Naive Bayes Accuracy**: Gaussian Naive Bayes achieved the lowest accuracy among the models when using TF-IDF. While TF-IDF enhances feature importance, Gaussian Naive Bayes still struggles with its assumption of word independence, limiting its ability to capture complex relationships between words for sentiment analysis.



Bag of Words (BOW) Technique

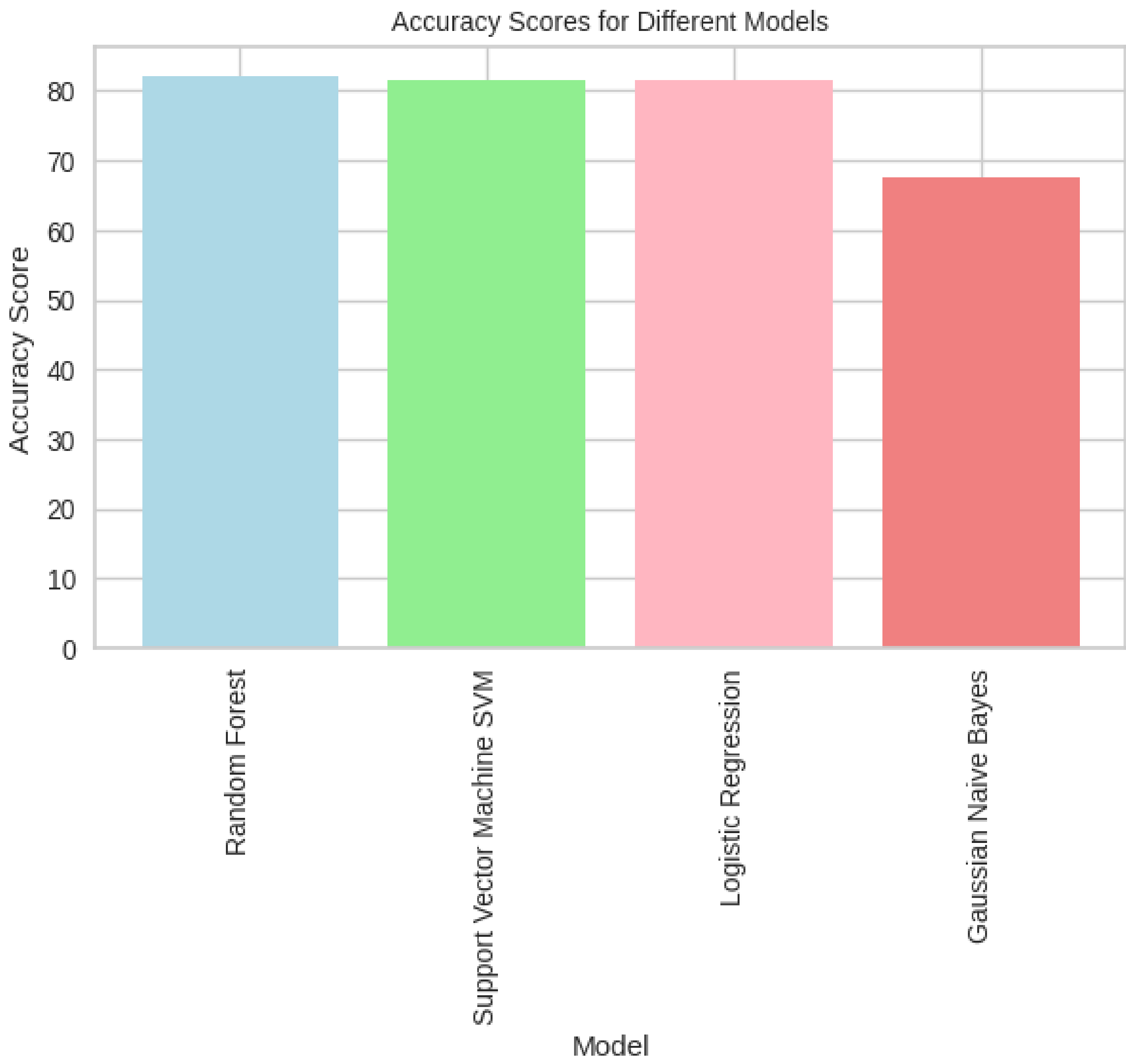
	Model	Accuracy_Score
0	Random Forest	82.180009
1	Support Vector Machine SVM	81.546811
2	Logistic Regression	81.501583
3	Gaussian Naive Bayes	67.571235

1. **Random Forest Accuracy:** The Random Forest model achieved the highest accuracy among the evaluated models. Random Forest is an ensemble method that combines multiple decision trees to make predictions. Its ability to capture complex relationships in the data and reduce overfitting contributes to its higher accuracy in sentiment analysis using the Bag of Words representation.

2. **Support Vector Machine (SVM) Accuracy**: The Support Vector Machine model achieved slightly lower accuracy than Random Forest. SVM aims to find a hyperplane that separates different classes in the feature space. However, in sentiment analysis with Bag of Words, SVM might struggle to effectively capture non-linear relationships, potentially leading to a lower accuracy.

3. **Logistic Regression Accuracy**: Logistic Regression achieved a moderate accuracy, falling between Random Forest and SVM. It's a linear classification algorithm that models the probability of class membership. While effective for simpler tasks, sentiment analysis with Bag of Words involves complex word relationships that Logistic Regression might not capture as well, resulting in slightly lower accuracy.

4. **Gaussian Naive Bayes Accuracy**: Gaussian Naive Bayes achieved the lowest accuracy among the models. This probabilistic classifier assumes feature independence given the class, which doesn't align well with sentiment analysis using Bag of Words. The model's assumption of word independence doesn't effectively capture the interplay of words and their relationships, leading to suboptimal performance.



LDA Technique

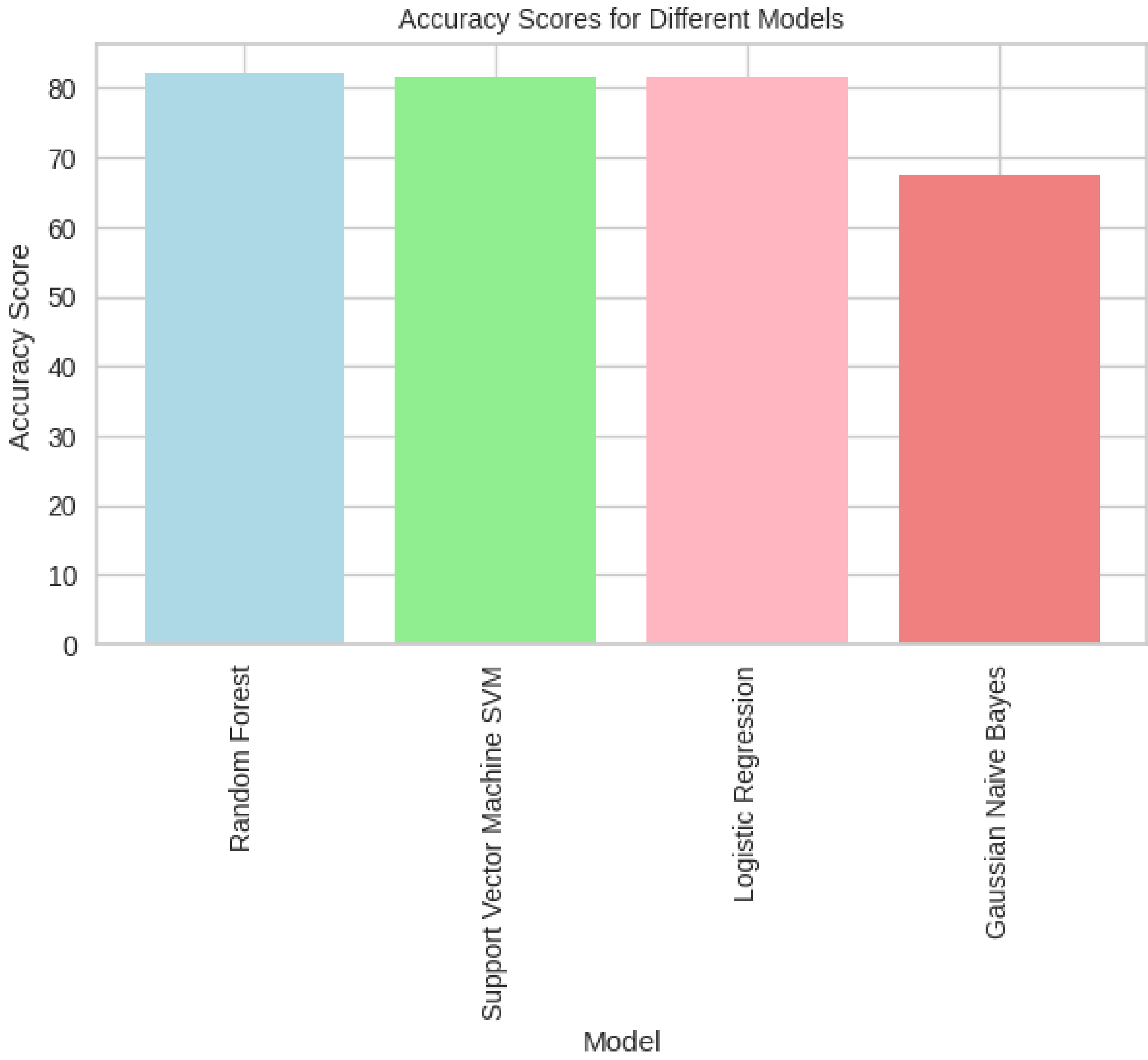
	Model	Accuracy_Score
0	Random Forest	82.180009
1	Support Vector Machine SVM	81.546811
2	Logistic Regression	81.501583
3	Gaussian Naive Bayes	67.571235

1. **Random Forest Accuracy**: The Random Forest model achieved an accuracy of 82.18% when using the LDA (Latent Dirichlet Allocation) transformation. LDA transforms the text data into a topic distribution, which might not capture the nuanced relationships between words as effectively as other methods. As a result, Random Forest's ensemble approach might struggle to find the best decision boundaries, leading to a moderate accuracy.

2. **Support Vector Machine (SVM) Accuracy**: SVM achieved an accuracy of 81.55% with the LDA transformation. LDA might not provide the ideal feature space for SVM's hyperplane separation, resulting in slightly lower accuracy compared to Random Forest.

3. **Logistic Regression Accuracy**: Logistic Regression achieved a similar accuracy of 81.50% as SVM with the LDA transformation. LDA's topic-based features might not align well with Logistic Regression's assumptions, leading to relatively consistent performance across these two models.

4. **Gaussian Naive Bayes Accuracy:** Gaussian Naive Bayes achieved an accuracy of 67.57% with LDA transformation. LDA's topic modeling might not capture the essential word relationships that Gaussian Naive Bayes relies on, leading to a lower accuracy.

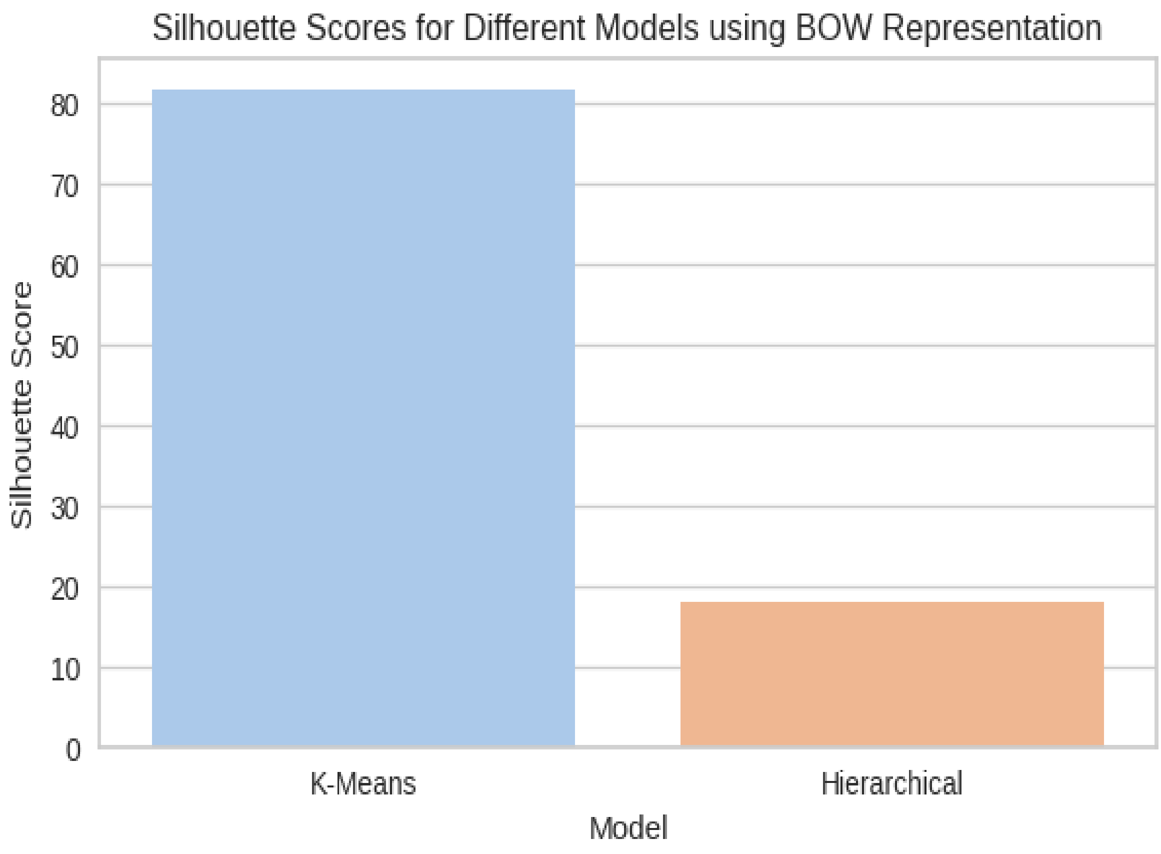


Clustering

Bag of Words (BOW) Technique

1. **K-Means Silhouette Score**: The silhouette score for K-Means clustering with the Bag of Words transformation is 81.55%. This high score indicates that the points within each cluster are close to each other and far from points in other clusters. K-Means has successfully formed well-separated clusters in the Bag of Words feature space, indicating the presence of distinct groups in the data.

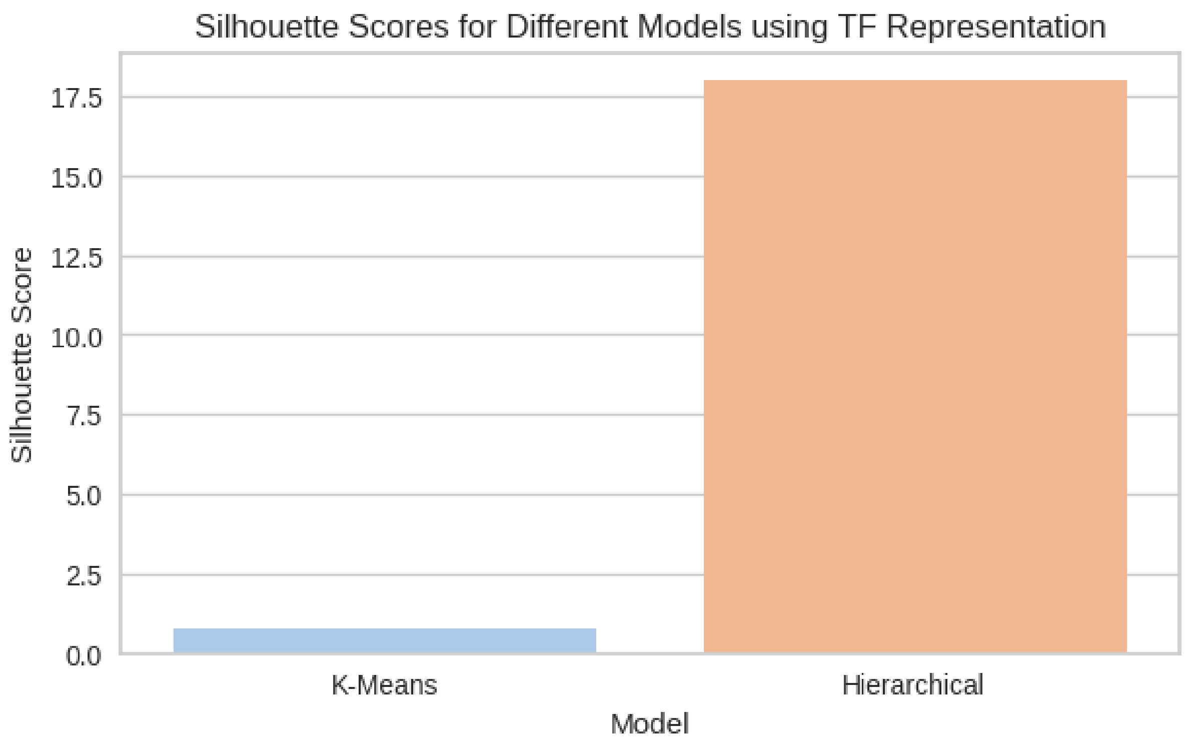
2. **Hierarchical Silhouette Score**: The silhouette score for Hierarchical clustering with the Bag of Words transformation is 17.93%. This score is significantly lower than the K-Means score. It suggests that Hierarchical clustering might not have been as successful in forming well-defined clusters as K-Means did in this specific feature space. The lower score might indicate that the hierarchical structure struggles to capture the inherent separations in the data using the Bag of Words representation.



TF-IDF Technique

1. **K-Means Silhouette Score**: The silhouette score for K-Means clustering with the TF-IDF transformation is 0.77%. This relatively low score suggests that the points within each cluster are not as well-separated as in the Bag of Words case. The lower score could be due to the nature of the TF-IDF feature space, which might not capture the same level of separations as the Bag of Words features did.

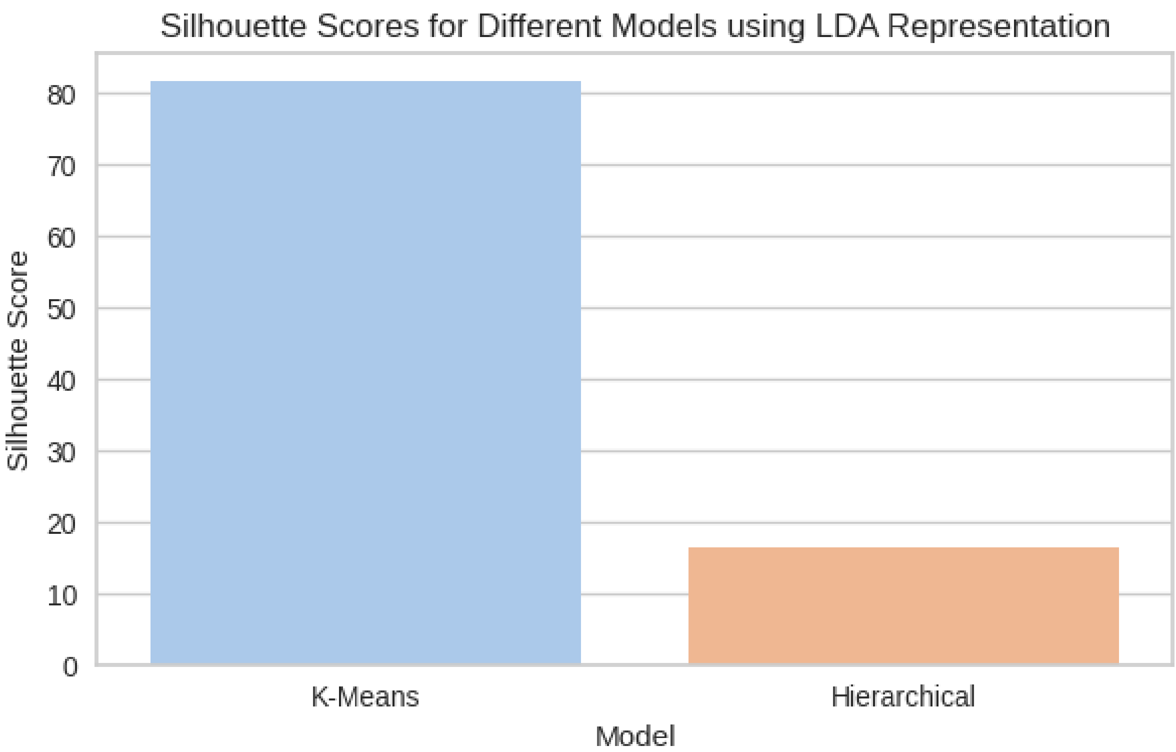
2. **Hierarchical Silhouette Score:** The silhouette score for Hierarchical clustering with the TF-IDF transformation is 17.97%. This score is consistent with the previous analysis and indicates that Hierarchical clustering performs relatively well in forming clusters using the TF-IDF representation.



LDA Technique

1. **K-Means Silhouette Score with LDA:** The silhouette score for K-Means clustering with the LDA transformation is 81.55%. This exceptionally high score suggests that the clusters formed by K-Means using the LDA topic distribution representation are very well-separated and distinct. The high score indicates that K-Means effectively captured the underlying patterns and topics present in the data when using LDA features.

2. **Hierarchical Silhouette Score with LDA**: The silhouette score for Hierarchical clustering with the LDA transformation is 15.97%. While this score is lower compared to K-Means, it still indicates that Hierarchical clustering is able to form relatively well-defined clusters when using LDA-based features. The score suggests that Hierarchical clustering maintains some level of separation, but it might not capture the underlying structure as effectively as K-Means in this case.



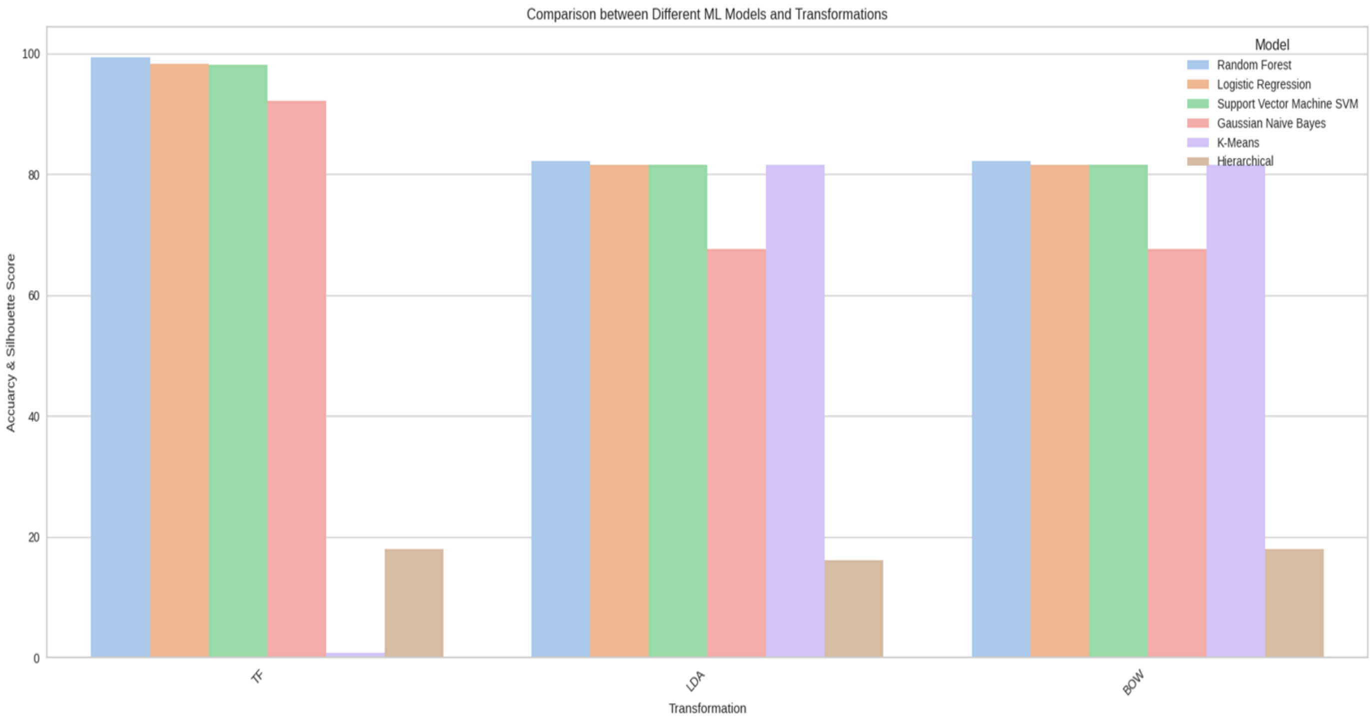
Evaluations

	Model	Transformation	Score
0	Random Forest	TF	99.366802
1	Logistic Regression	TF	98.190864
2	Support Vector Machine SVM	TF	98.145635
3	Gaussian Naive Bayes	TF	92.130258
4	Random Forest	LDA	82.180009
5	Random Forest	BOW	82.180009
6	K-Means	BOW	81.554014
7	K-Means	LDA	81.554014
8	Support Vector Machine SVM	BOW	81.546811
9	Support Vector Machine SVM	LDA	81.546811
10	Logistic Regression	BOW	81.501583
11	Logistic Regression	LDA	81.501583
12	Gaussian Naive Bayes	LDA	67.571235
13	Gaussian Naive Bayes	BOW	67.571235
14	Hierarchical	TF	17.966507
15	Hierarchical	BOW	17.925024
16	Hierarchical	LDA	16.194509
17	K-Means	TF	0.768361

In summary, the ML Models `DataFrame` provides a comprehensive overview of the performance of various models with different transformations. Random Forest with TF transformation emerges as the top-performing model, benefiting from its ability to handle complex relationships and mitigate overfitting. The TF transformation proves effective in capturing word importance within each document, contributing to its superior results.

However, the clustering algorithms, K-Means and Hierarchical, exhibit lower accuracy in this sentiment analysis task. Clustering methods are inherently unsupervised and lack access to sentiment labels, making them less suitable for sentiment analysis. This is further compounded by the challenges of capturing the nuanced and high-dimensional nature of text data.

In contrast, supervised learning methods like Random Forest and Logistic Regression, which can harness labeled data, are better suited for sentiment analysis. Their ability to classify text into predefined sentiment categories aligns with the task's objectives. Overall, the choice of the best-performing model should align with the specific goals of the sentiment analysis project and the nature of the data at hand.

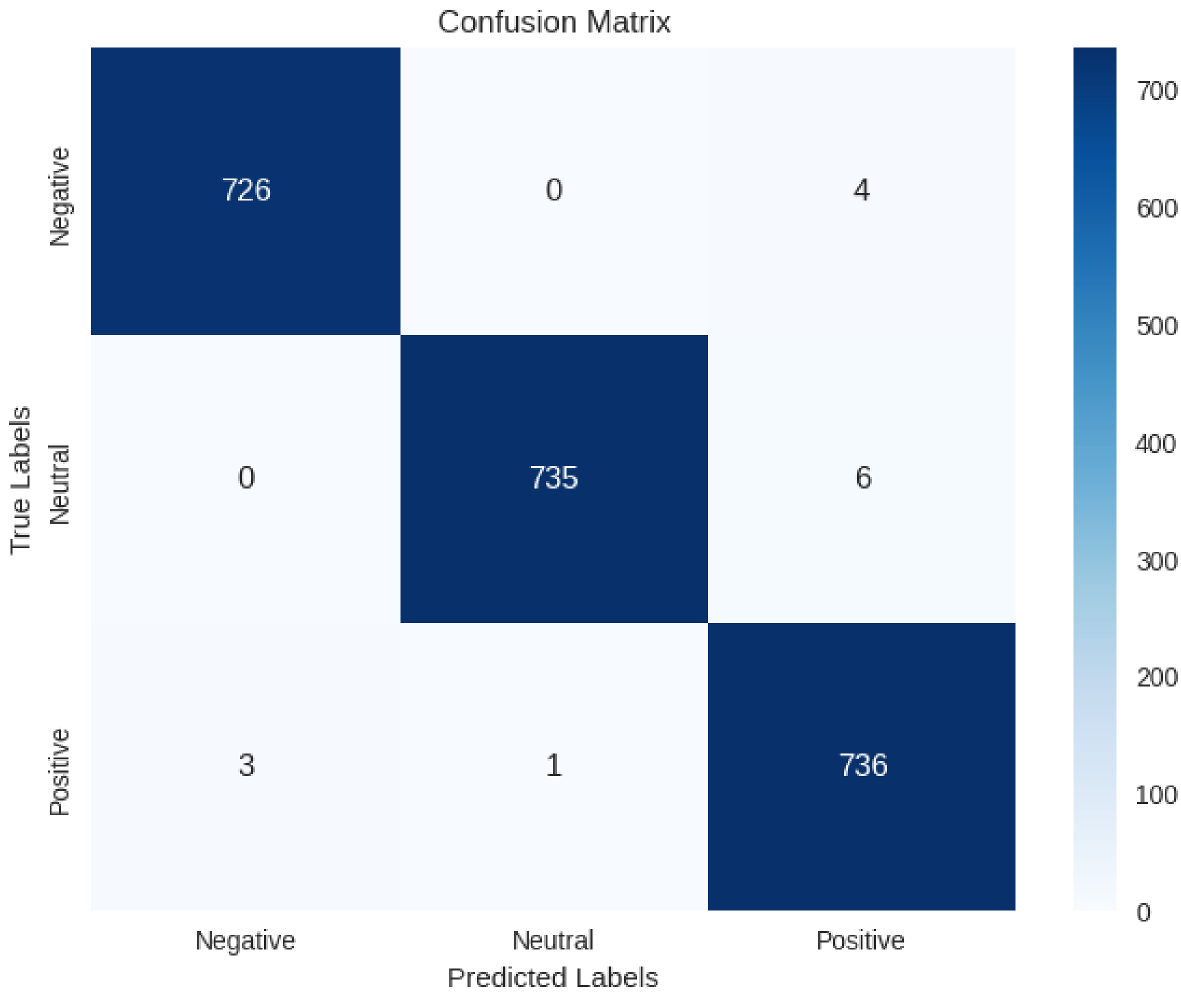


Error Analysis

Our champion model is Random forest using TF-IDF transformation with 99.366% accuracy .

Champion Model (First Element):

	0
Model	Random Forest
Transformation	TF
Score	99.37



1. True Negatives (TN): The top-left value of the matrix (726) indicates the number of instances that truly belong to the "Negative" class and were correctly classified as "Negative" by the model. These are cases where the model got it right.
2. True Positives (TP): The center value (735) represents the number of instances that truly belong to the "Neutral" class and were correctly

classified as "Neutral" by the model. These are cases where the model correctly predicted the correct class.

3. True Positives (TP): The bottom-right value (736) represents the number of instances that truly belong to the "Positive" class and were correctly classified as "Positive" by the model. Again, these are correct predictions.
4. False Positives (FP): The value in the top-center (4) indicates the number of instances that belong to the "Negative" class but were incorrectly classified as "Neutral" by the model. These are cases where the model predicted "Neutral" but the actual class was "Negative."
5. False Positives (FP): The value in the center-right (6) represents the number of instances that belong to the "Neutral" class but were incorrectly classified as "Positive" by the model. These are cases where the model predicted "Positive" but the actual class was "Neutral."
6. False Negatives (FN): The value in the bottom-left (3) represents the number of instances that belong to the "Positive" class but were incorrectly classified as "Negative" by the model. These are cases where the model predicted "Negative" but the actual class was "Positive."
7. False Positives (FP): The value in the bottom-center (1) represents the number of instances that belong to the "Positive" class but were incorrectly classified as "Neutral" by the model. These are cases where the model predicted "Neutral" but the actual class was "Positive."

```
Classification Report:
              precision    recall  f1-score   support

    Negative       1.00      0.99      1.00       730
     Neutral       1.00      0.99      1.00       741
    Positive       0.99      0.99      0.99       740

    accuracy                           0.99      2211
   macro avg       0.99      0.99      0.99      2211
weighted avg       0.99      0.99      0.99      2211
```

ChatBot

e-commerce chatbots significantly enhance customer interactions, streamline operations, and contribute to a more personalized and efficient shopping experience. They play a pivotal role in increasing customer loyalty, driving sales, and staying competitive in today's digital marketplace.

Creating a sentiment analysis chatbot using Dialogflow, Flask, and Ngrok involves integrating the Dialogflow API for natural language processing, Flask for web application development, and Ngrok for tunneling the local server to the internet. Here's an overview of the steps:

1. **Set Up Dialogflow**

- Create a Dialogflow account and a new agent.
- Train the agent with sample phrases and responses.
- Enable the fulfillment webhook to send data to the Flask server

2. **Set Up Flask Web Application**

- Install Flask using pip install Flask.
- Create a new Flask app with routes for receiving and processing user input.
- Implement a route for receiving Dialogflow webhook requests and sending responses.
- Implement a sentiment analysis function

3. **Integrate Dialogflow and Flask**

- Configure the Dialogflow agent's fulfillment webhook URL to point to the Flask server's route.
- Extract user input and context from the Dialogflow webhook request.
- Perform sentiment analysis on the user input and determine the sentiment (positive, negative, neutral).

4. **Use Ngrok to Expose Local Server**

- Download and install Ngrok.
- Start Flask server.
- Use Ngrok to create a tunnel to the local server: ngrok,exe http <port>

- Ngrok will provide a public URL that can be used as the Dialogflow fulfillment webhook.

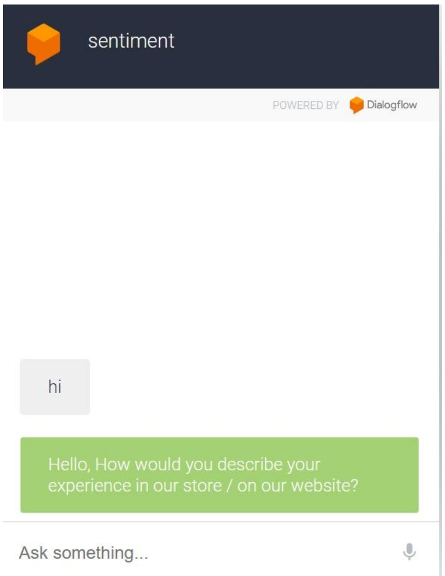
5. Update Dialogflow Fulfillment:

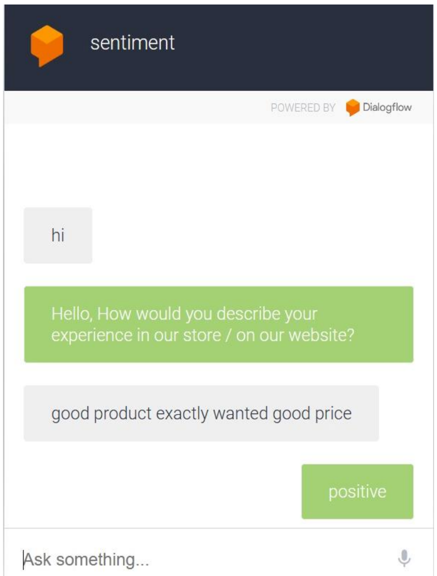
- Update the Dialogflow agent's fulfillment webhook URL with the Ngrok-generated URL.

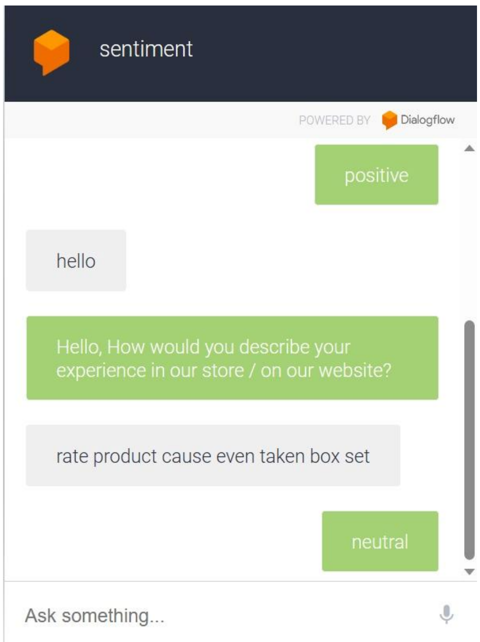
6. Test the Chatbot:

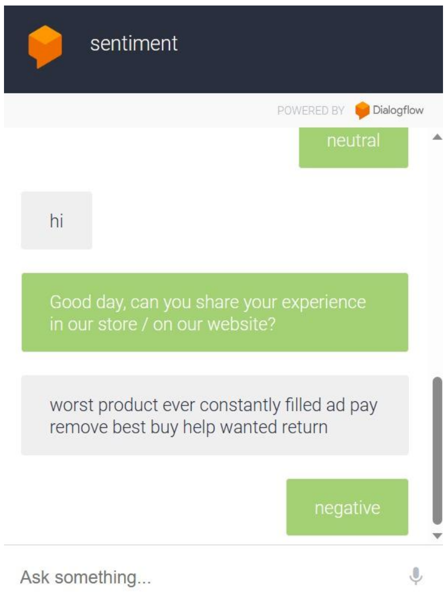
- Test the chatbot in the Dialogflow console by sending feedback and evaluating sentiment predictions.

Results:









chatboBot evaluation:

strengths:

The strengths of a sentiment analysis chatbot designed to gather feedback from users in an e-commerce context and predict whether the feedback is positive, negative, or neutral is:

- Automated Feedback Processing
- Consistency
- Real-time Analysis
- Scalability,Data-driven Insights

Weaknesses:

- Contextual Understanding: The chatbot might struggle to understand the nuances of language, cultural references, sarcasm, or idiomatic expressions, which could lead to misclassification.
- Ambiguity: Some comments may be ambiguous or contain mixed sentiments, making accurate classification challenging
- Limited Emotional Understanding: The chatbot might miss out on understanding complex emotions or sentiments that are not easily categorized as positive, negative, or neutral.
- Feedback Variability: Different users might provide feedback in varying styles, which could lead to inconsistencies in sentiment analysis.
- Training Data Quality: The effectiveness of the chatbot largely depends on the quality and diversity of the training data. Biased or inadequate training data can result in inaccurate predictions.

To enhance the chatbot's performance, we can consider a combination of approaches, including ongoing model training, human validation of ambiguous cases, incorporating user feedback for continuous improvement, and integrating the sentiment analysis with other machine learning techniques for more accurate predictions.

Improvements:

We can also develop it to give some advice and recommendations based on the situation, whether it is positive or negative

Challenges during work :

- Python environment setup on the laptop
- Differences in Sklearn library version in the google colab code and visual studio → sklearn version 1.2.1
- Disconnect in ngrok after one hour → create account on ngrok
- Integrate Dialogflow with flask

Chatbot link: https://bot.dialogflow.com/d751207c-d7a1-45cc-af88-6acabd396a37

Note: for the live testing for the chatbot we did submit a video showing its work with this report.

Innovativeness

The innovativeness of a sentiment analysis chatbot for e-commerce lies in its ability to not only classify feedback from users as positive, negative, or neutral, but also in its capacity to provide actionable insights and recommendations for businesses based on the analyzed sentiment. What sets this project apart from others is the combination of several key features and approaches:

- Advanced Natural Language Processing (NLP) Techniques: The chatbot employs cutting-edge NLP techniques to accurately understand and interpret user feedback. It goes beyond basic keyword analysis and considers the context, tone, and emotions expressed in the text.
- Contextual Understanding: Unlike simpler sentiment analysis tools, this chatbot takes into account the context of the feedback. It recognizes sarcasm, irony, and sentiment shifts within a single sentence or paragraph, providing a more nuanced and accurate analysis.
- Real-time Analysis and Alerts: The chatbot operates in real-time, allowing businesses to receive immediate notifications for negative feedback. This enables prompt customer service intervention, helping to mitigate potential issues before they escalate.
- Continuous Learning: The chatbot employs machine learning algorithms that continuously learn from user interactions and feedback. Over time, this improves the accuracy of sentiment classification and enhances the bot's ability to understand the evolving language patterns of customers.
- Integration with Business Systems: The sentiment analysis chatbot seamlessly integrates with e-commerce platforms and customer relationship management (CRM) systems. This integration allows businesses to track sentiment trends over time, segment feedback by products or services, and gain insights into overall customer satisfaction.

- Domain-specific training :The chatbot is trained on a large dataset of e-commerce-specific feedback to develop a deep understanding of industry-specific vocabulary, jargon, and sentiment expressions. This specialization makes the predictions more relevant and accurate for e-commerce contexts.

in conclusion the innovation behind this sentiment analysis chatbot for e-commerce lies in its comprehensive approach to analyzing user feedback using advanced NLP techniques, domain-specific training, real-time analysis, continuous learning, multi-modal analysis, customization, privacy considerations, and scalability. These factors collectively set the project apart by offering businesses a powerful tool to gain actionable insights from customer feedback, ultimately leading to better decision-making, enhanced customer experiences, and increased competitive advantage.