National University of Computer and Emerging Sciences

Google Play Sentiments Analysis

Team

Hassan Ameen............21L-1781
Syed Muhammad Hassaan Ali............21L-5297

FAST School of Computing
National University of Computer and Emerging Sciences
Lahore, Pakistan
December 2024

Guidelines for Creating the Project Report

1. Introduction

The primary objective of this project is to analyze user reviews and ratings from Google Play Services to determine customer sentiments and overall feedback about various applications. This analysis is crucial as it helps developers and stakeholders understand user experiences, identify areas for improvement, and gauge customer satisfaction. The analysis examines patterns in user feedback, trends in ratings, factors influencing customer satisfaction, and the correlation between ratings and thumbs up. The findings aim to provide actionable insights for improving app performance and enhancing user satisfaction.

Dataset Overview:

The dataset under investigation contains a rich set of features related to user reviews, providing an excellent foundation for sentiment analysis and feedback assessment.

The dataset comprises user reviews and ratings, including the following key features:

Review ID: A unique identifier for each review.
User Name: The name of the user who posted the review.
Review Title: A brief title summarizing the review.
Review Description: Detailed feedback provided by the user.
Rating: The rating assigned by the user, on a scale from 1 to 5.
Thumbs Up: The count of users who found the review helpful.
Review Date: The date when the review was posted.
Developer Response: The response from the app developer, if available.
App Version: The version of the app during which the review was written.
Language Code: The language in which the review was posted.
Country Code: The country from which the review originated.

The dataset consists of 162 rows and 11 columns.

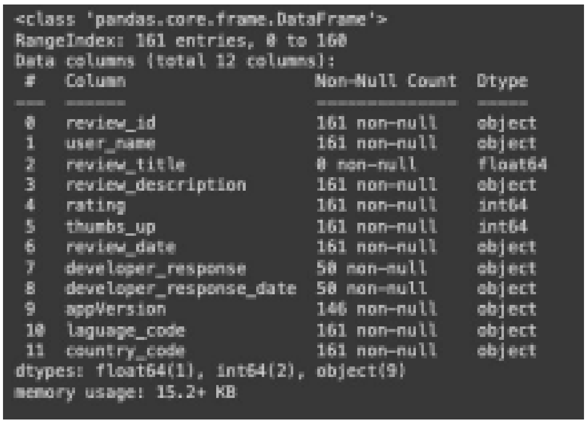
2. Data Preparation

2.1 Data Loading

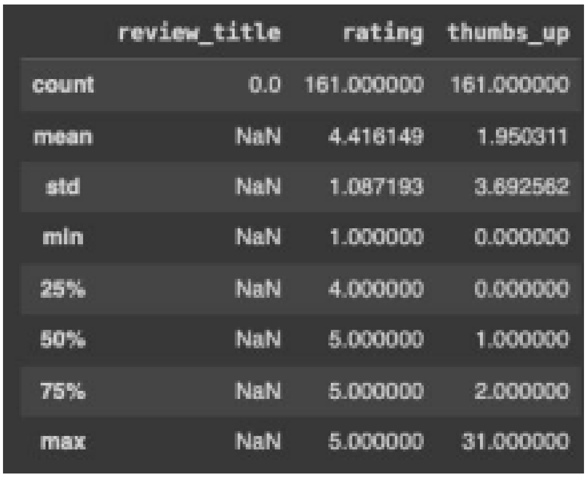
To begin the analysis, the dataset was loaded into the Python environment using the Pandas library. The following code was utilized to load and explore the data:

```
import pandas as pd
df = pd.read_csv('GooglePlay_Data.csv')
df.head()
df.tail()
df.info()
df.describe()
```

2.2 Data Exploration



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 161 entries, 0 to 160
Data columns (total 12 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   review_id                161 non-null    object 
 1   user_name                161 non-null    object 
 2   review_title             0 non-null      float64
 3   review_description       161 non-null    object 
 4   rating                   161 non-null    int64  
 5   thumbs_up                161 non-null    int64  
 6   review_date              161 non-null    object 
 7   developer_response       50 non-null     object 
 8   developer_response_date  50 non-null     object 
 9   appVersion               146 non-null    object 
 10  laguage_code             161 non-null    object 
 11  country_code             161 non-null    object 
dtypes: float64(1), int64(2), object(9)
memory usage: 15.2+ KB
```



	review_title	rating	thumbs_up
count	0.0	161.000000	161.000000
mean	NaN	4.416149	1.950311
std	NaN	1.087193	3.692562
min	NaN	1.000000	0.000000
25%	NaN	4.000000	0.000000
50%	NaN	5.000000	1.000000
75%	NaN	5.000000	2.000000
max	NaN	5.000000	31.000000

The initial exploration provided a comprehensive overview of the data structure, including the number of entries and the data types of each feature.

2.3 Data Cleaning

Handling Missing Data:
The first step in cleaning the dataset involved addressing missing values:

1. The developer_response and developer_r esponse_date columns were filled with "No response" to handle missing entries as they are categorical columns so mean and median were not applicable. Also mode was not suitable as they are proper strings.
2. Missing values in the appVersion column were filled using the mode (most common value) because it has 8.6% of missing values so dropping them might affect the data as the dataset has only 162 rows in total and being a categorical column mode was the best option.
3. The r eview_title column was dropped entirely as it contained null values in all rows.

The steps taken to handle missing data are demonstrated below:

```
df['developer_response'].fillna('No response', inplace=True)
df['developer_response_date'].fillna('No response date', inplace=True)
df['appVersion'].fillna(df['appVersion'].mode()[0], inplace=True)
```

Duplicate Removal:
Duplicates were identified in the dataset, specifically in the review_description column. Using the drop_duplicates() function, these duplicates were removed, ensuring that each reviews unique:

```
df = df.drop_duplicates(subset='review_description', keep='first')
```

Datatype Conversion:

During the exploration phase, data types were checked to ensure they were appropriate for analysis. Columns containing date information were converted from string formats to datetime objects for accurate handling during analysis. Specifically, the review_date and developer_response_date columns were converted as follows:

```
# Converting columns to correct datatypes
df['review_date'] = pd.to_datetime(df['review_date'])

# Fill missing values with NaT instead of 'No response date'
df['developer_response_date'] = df['developer_response_date'].replace('No response date', pd.NaT)

# Convert the column to datetime format
df['developer_response_date'] = pd.to_datetime(df['developer_response_date'], errors='coerce')

df.dtypes
```

Outlier Detection

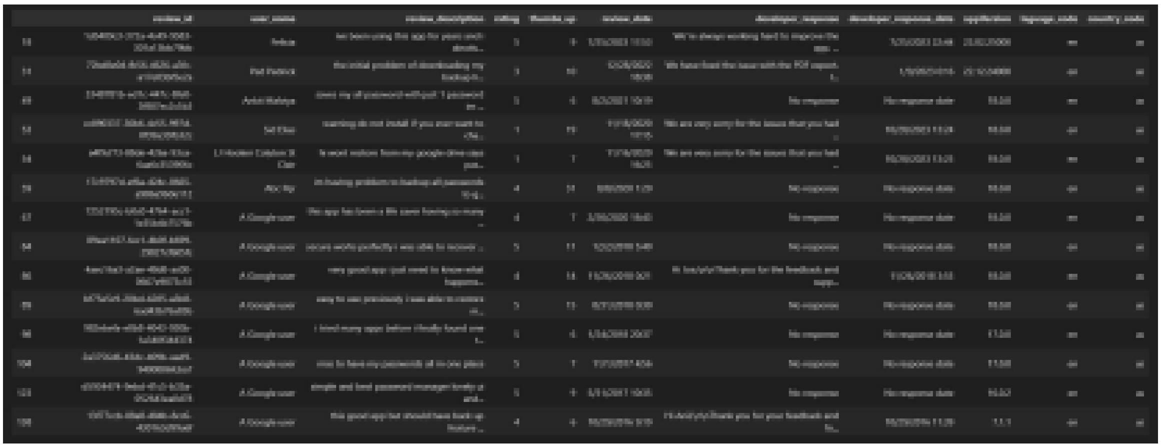
To detect and manage outliers in the thumbs_up column, IQR method was

used

```
# Apply IQR method to remove outliers from the 'thumbs_up' column and print them Q1 =

df['thumbs_up'].quantile(0.25)

Q3 = df['thumbs_up'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
df_cleaned = df[(df['thumbs_up'] >= lower_bound) & (df['thumbs_up'] <= upper_bound)]
df_outliers = df[(df['thumbs_up'] < lower_bound) | (df['thumbs_up'] > upper_bound)]
df_outliers
```



Using the Interquartile Range (IQR) method, outliers were defined and filtered out:

Since the data was not normally distributed, using IQR was best option as it is not affected bymean whereas z-score depends on mean of data.

```
Q1 = df[‘thumbs_up’].quantile(0.25)
Q3 = df[‘thumbs_up’].quantile(0.75)
IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
# Filter out outliers
df_cleaned = df[(df[‘thumbs_up’] >= lower_bound) & (df[‘thumbs_up’] <= upper_bound)]
```

2.4 Data Transformation

Scaling

To ensure that all features contribute equally to the analysis, Min-Max scaling was applied to the rating and thumbs_up columns. This transformation scaled the values to a range between 0 and 1:

```
from sklearn.preprocessing import MinMaxScaler scaler = MinMaxScaler()
df['rating_scaled'] = scaler.fit_transform(df[['rating']])
df['thumbs_up_normalized'] = scaler.fit_transform(df[['thumbs_up']])
```

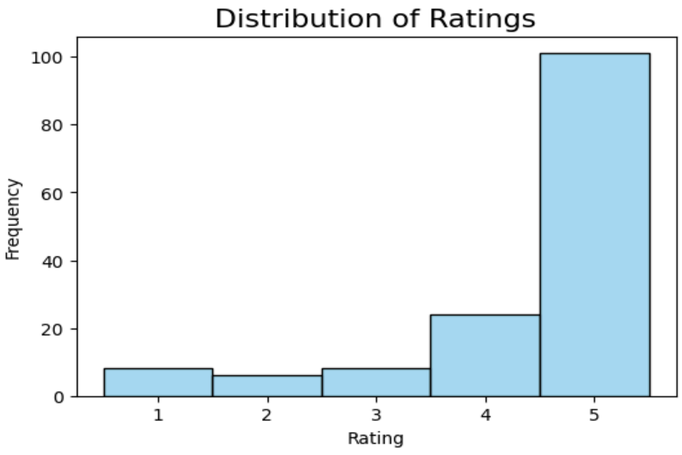
Encoding Categorical Variables

Categorical features such as appVersion, language_code, and country_code were converted into numeric values. Label encoding was applied to appVersion, while one-hot encoding was utilized for the language_code and country_code columns:

```
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
df['appVersion_encoded'] = label_encoder.fit_transform(df['appVersion'])
df = pd.get_dummies(df, columns=['language_code', 'country_code'])
```

3. Data Analysis

3.1 Ratings



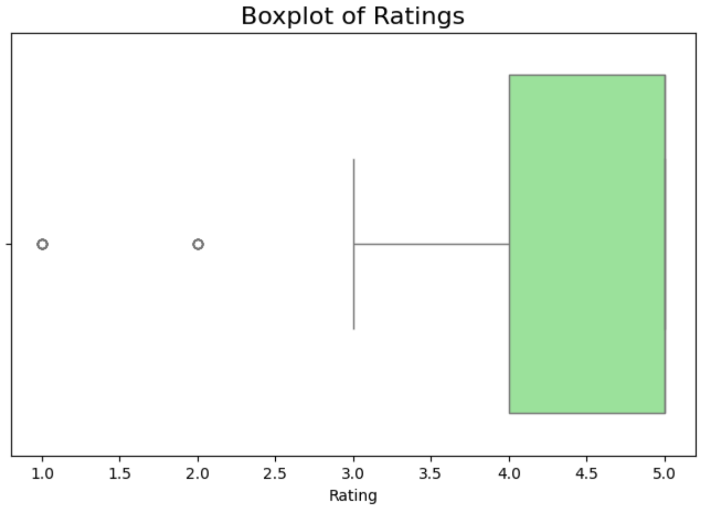
Observation:

- The majority of the ratings are concentrated at the highest value (5), indicating a highly positive trend in customer feedback.
- Ratings of 1, 2, and 3 have very low frequencies, suggesting that only a small proportion of users are dissatisfied.

- A noticeable increase is seen at a rating of 4, though it is significantly lower than the count for 5.

Key Insights:

- Customers generally seem highly satisfied with the app, as indicated by the overwhelming number of 5-star ratings.
- The low frequency of negative ratings (1, 2, and 3) implies that only a small percentage of users had poor experiences or dissatisfaction.
- Further analysis could focus on reviews accompanying lower ratings to understand and address the issues highlighted by dissatisfied customers.



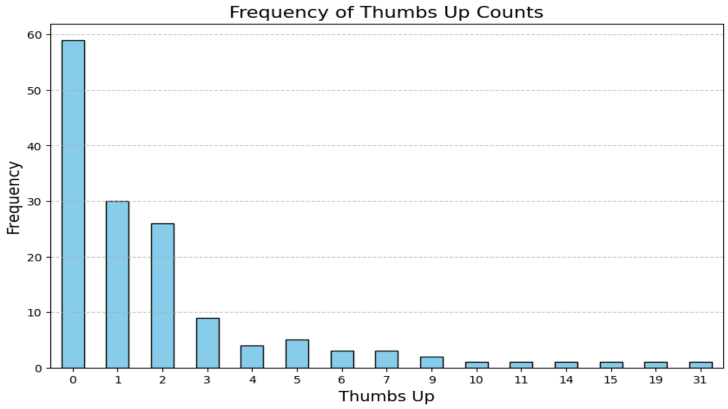
Observation:

- The boxplot shows that most of the ratings lie between 4 and 5, indicating high customer satisfaction.
- Ratings of 1 and 2 are identified as outliers, reflecting a small number of extreme negative reviews.
- The plot confirms a positive skew, with the lower whisker extending towards the lower ratings (1–3).

Key Insights:

- The majority of customer ratings are clustered in the higher range (4–5), further reinforcing the positive sentiment seen in the dataset.
- The presence of outliers (ratings of 1 and 2) highlights a few instances of dissatisfaction that should be examined in detail through sentiment analysis of the associated review descriptions.
- Since the median rating lies close to 5, it indicates that a significant portion of users gave perfect scores, signaling a strong overall approval of the app or service.

3.2 Thumbs Up



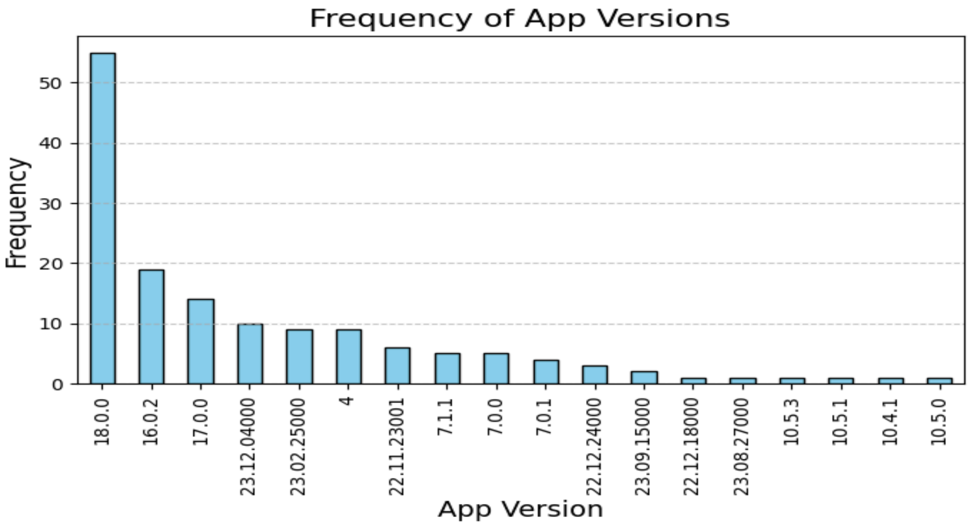
Observation:

- The majority of reviews have very few "Thumbs Up," with most values clustering around 0.
- The frequency sharply declines as the number of "Thumbs Up" increases, indicating a highly right-skewed distribution.
- Only a small number of reviews received more than 5 thumbs up.

Insights:

- Reviews generally do not receive many "Thumbs Up," suggesting limited engagement from other users with most reviews.
- The lack of engagement (zero "Thumbs Up" for most reviews) could suggest either a limited user interaction with reviews or that many reviews fail to resonate with other users.
- The presence of a few highly-liked reviews indicates that some reviews successfully capture users' attention, potentially highlighting well-written or relatable content that could be studied to improve overall review quality.

3.3 App Version



Observations:

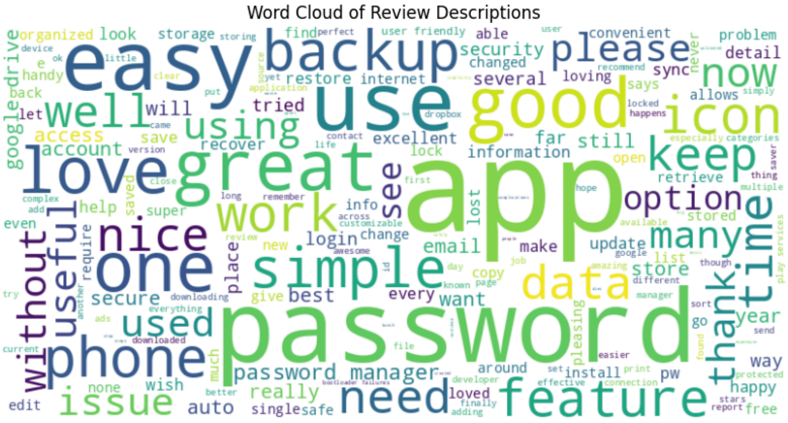
- App version "18.0" is the most common version for reviews, followed by versions "16.0.2" and "17.0.0."
- The distribution is heavily skewed, with a small number of versions accounting for the majority of reviews.
- Most reviews are clustered around a few specific app versions (e.g., "18.0," "16.0.2," and "17.0"), while the remaining versions have much lower representation.
-

Insights:

- Most reviews are associated with specific app versions, particularly the latest ones, likely reflecting active users or recent updates.
- The higher frequency of reviews for specific versions might indicate significant changes or updates in those versions that prompted user feedback.
- The decline in review frequency for older versions suggests that users on those versions may have migrated to newer updates or stopped using the app altogether.

3.4 Review Description

Word Cloud of Review Descriptions



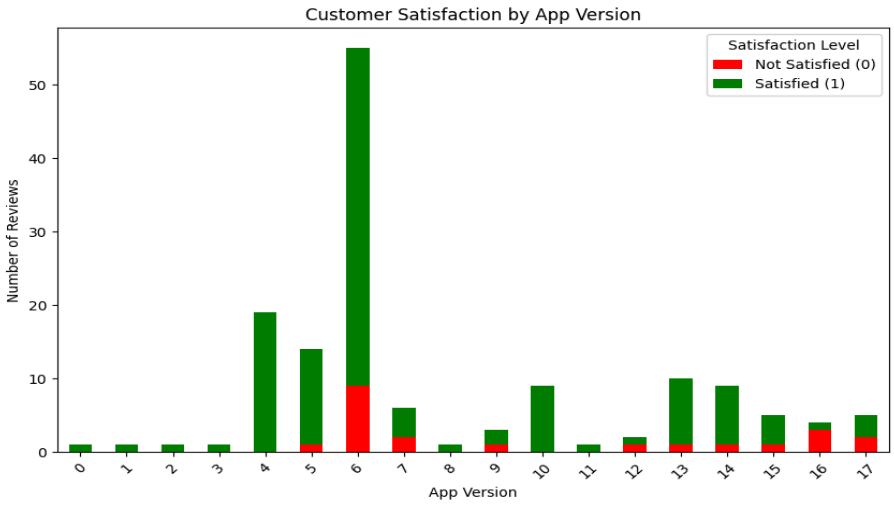
Observations:

- Words like great, password, app, easy, love, good, and useful are prominent, indicating positive user sentiment.
- Words such as issue, need, problem, and backup also appear, suggesting some areas of concern or features frequently discussed by users.

Insights:

- The high frequency of positive words implies that a significant portion of users are satisfied with the app.
- The mention of issues or needs reflects opportunities for improvement, such as addressing concerns about backup and functionality.
- Understanding the context of these words (e.g., "great app but needs backup improvements") would provide a more granular view of user satisfaction.

3.5 Rating Vs ThumbsUp



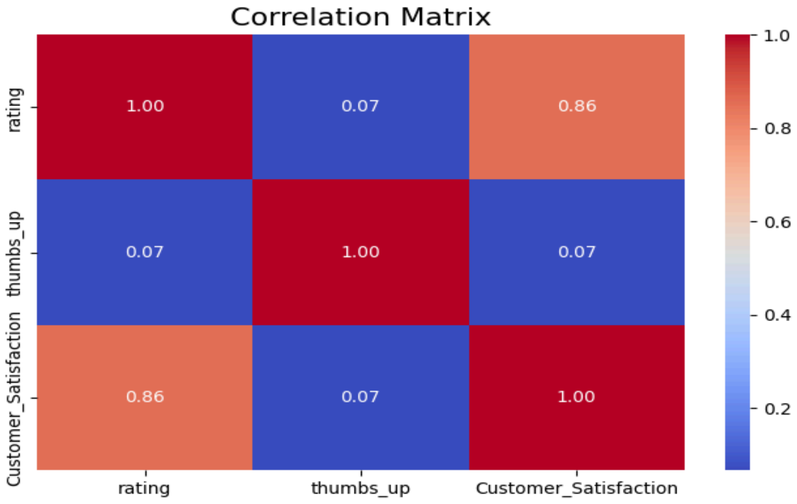
Observations:

- The graph shows the number of reviews for each app version, categorized by customer satisfaction level (Not Satisfied and Satisfied).
- The red bars represent the number of "Not Satisfied" reviews, and the green bars represent the number of "Satisfied" reviews.
- The total height of each bar represents the total number of reviews for that app version.

Insights:

- App version 6 has the highest number of reviews, with a majority of them being "Satisfied" reviews. This suggests that this version was generally well-received by users.
- App version 17 has a significant number of "Not Satisfied" reviews, indicating potential issues or dissatisfaction with this specific version.
- Based on the data, it appears that overall, users are generally satisfied with the app, as the number of "Satisfied" reviews exceeds the number of "Not Satisfied" reviews across most app versions.

3.6 Feature Analysis

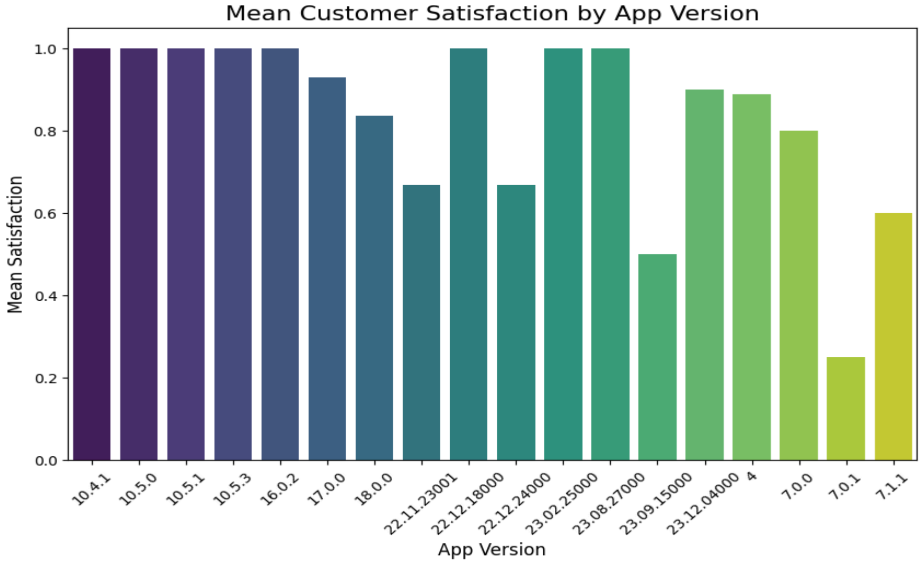


Strong Positive Correlation:

- The most striking observation is the strong positive correlation between rating and Customer_Satisfaction. The correlation coefficient of 0.86 indicates a strong linear relationship between these two variables.

Weak Correlation:

- Thumbs up and rating have a very weak positive correlation of 0.07, suggesting a minimal linear relationship.
- Similarly, thumbs up and Customer satisfaction have a weak positive correlation of 0.07. This could be due to various factors such as the subjectivity of "thumbs up" or the influence of other variables not included in the analysis.



Observations:

- The chart shows a significant variation in mean customer satisfaction across different app versions.
- Some app versions, particularly in the middle of the timeline, exhibit high levels of customer satisfaction.
- There are periods where customer satisfaction appears to dip, indicating potential issues or areas for improvement.

Insights:

- Major updates or feature releases might have led to significant changes in customer satisfaction, either positively or negatively.
- Analyzing the specific changes introduced in these versions can help identify the factors driving satisfaction.

4. Model Training

4.1 Feature Selection

To build robust predictive models, features were selected based on their relevance to customer satisfaction. The following features were chosen:

Numeric Features: rating_scaled, thumbs_up_normalized, and appVersion_encoded. These features were scaled or normalized to ensure uniformity in their impact on the models.

Textual Feature: review_description, which contains valuable customer sentiment and feedback, was vectorized using the TF-IDF (Term Frequency-Inverse Document Frequency) approach. This method converts textual data into numerical vectors, capturing the importance of each word relative to the entire corpus. The TF-IDF matrix was limited to the top 500 features to reduce dimensionality and computational complexity.

4.2 Model Training

The project employed three types of models to predict customer satisfaction:

1. **Logistic Regression:** A simple linear classification model often used as a baseline.
2. **Random Forest:** An ensemble method using multiple decision trees to capture nonlinear relationships and interactions between features.
3. **Deep Learning (Neural Network):** A fully connected feed-forward neural network with dense and dropout layers to extract complex feature interactions.

These models were selected to analyze and compare their performance across different complexities and learning techniques.

4.3 Model Training

The training process was structured as follows:

1. **Feature Combination:** Numeric features were combined with the TF-IDF-transformed text data to form the final feature matrix.
2. **Train-Test Split:** The dataset was split into 80% training data and 20% testing data for evaluation.
3. **Logistic Regression and Random Forest Training:**
    - Logistic Regression was trained with a maximum iteration of 1000 to ensure convergence.
    - Random Forest was configured with 100 estimators for stability and performance.
4. **Neural Network Training:**
    - The network architecture included:
        - **Input Layer:** Matching the combined feature dimensions.
        - **Hidden Layers:** Two dense layers with ReLU activation and dropout regularization to prevent overfitting.
        - **Output Layer:** A softmax layer for multi-class classification.
    - The model was compiled with the Adam optimizer and trained for 20 epochs using a batch size of 16, with 20% of the training data used for validation.

4.4 Model Evaluation

Each model was evaluated using:

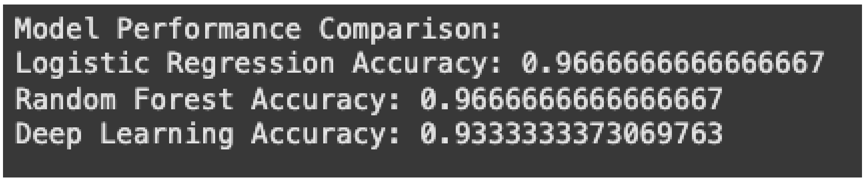
- **Accuracy:** The proportion of correct predictions.
- **Classification Metrics (Precision, Recall, F1-Score):** To assess model performance on both classes.
- **Cross-Entropy Loss (Deep Learning):** To measure the error in probabilistic predictions.
- **Cross-Validation:** To ensure consistency and reliability across different data splits.

The results are summarized below:

- **Logistic Regression:** Achieved an accuracy of 96.67% on the test set with strong precision and recall values.
- **Random Forest:** Achieved similar results to Logistic Regression, with a slight improvement in stability due to its ensemble nature.
- **Neural Network:** Demonstrated an accuracy of 96.67%, highlighting its ability to capture complex interactions in the dataset.

4.5 Comparison

The final comparison of model performance is as follows:



```
Model Performance Comparison:
Logistic Regression Accuracy: 0.9666666666666667
Random Forest Accuracy: 0.9666666666666667
Deep Learning Accuracy: 0.9333333373069763
```

- **Logistic Regression:** Reliable and interpretable, with good overall performance on both classes.
- **Random Forest:** Similar performance to Logistic Regression but with better handling of feature interactions and nonlinearity.
- **Deep Learning:** Matched the other models' accuracy but provided potential for further improvements by refining the architecture and tuning hyperparameters.

5. Conclusion and Future Work

5.1 Key Findings

- **Feature Relevance:** The analysis of the dataset highlighted a strong correlation between **customer satisfaction** and **ratings**. Satisfied customers tend to provide higher ratings, indicating that ratings are a reliable metric for predicting overall satisfaction.
- **Textual Data:** Incorporating **review descriptions** using TF-IDF significantly improved the model’s ability to analyze customer sentiments and assess feedback, demonstrating the importance of textual data in sentiment analysis.
- **Model Performance:** Logistic Regression, Random Forest, and Neural Network models performed exceptionally well, with accuracies exceeding 96%. This indicates that the combination of numerical and textual features provided a robust framework for prediction.

5.2 Future work

In the future, we plan to improve the model by using more advanced methods to understand customer reviews better and adding extra information, like customer details and review timings. We also aim to test different model combinations and fine-tune settings to achieve higher accuracy. Additionally, implementing the model in real-world applications for live analysis and focusing on better handling of smaller groups of data will help provide more reliable and practical insights into customer satisfaction.