College Event Feedback Analysis

Introduction

This report documents the analysis performed on student feedback collected from multiple college events. The objective was to identify satisfaction levels, detect recurring themes in positive and negative feedback, and provide actionable recommendations for future event improvements. The analysis was conducted as part of the Future Interns Data Science & Analytics internship, Task 3.

Dataset Description

The dataset consists of 400 responses from participants of various college events, including TechFest 2025, Cultural Fest 2025, Sports Meet 2025, Startup Summit 2025, and Science Fair 2025. Each entry contains: Student ID, Event Name, Rating (1–5), Would Recommend, Feedback, Venue, Mode, and Duration (hours). The data was cleaned, preprocessed, and analyzed using Python libraries such as pandas, TextBlob, vader Sentiment, matplotlib, seaborn, and WordCloud.

Step 1 — Upload dataset in Google Colab

Future_DS_03

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all

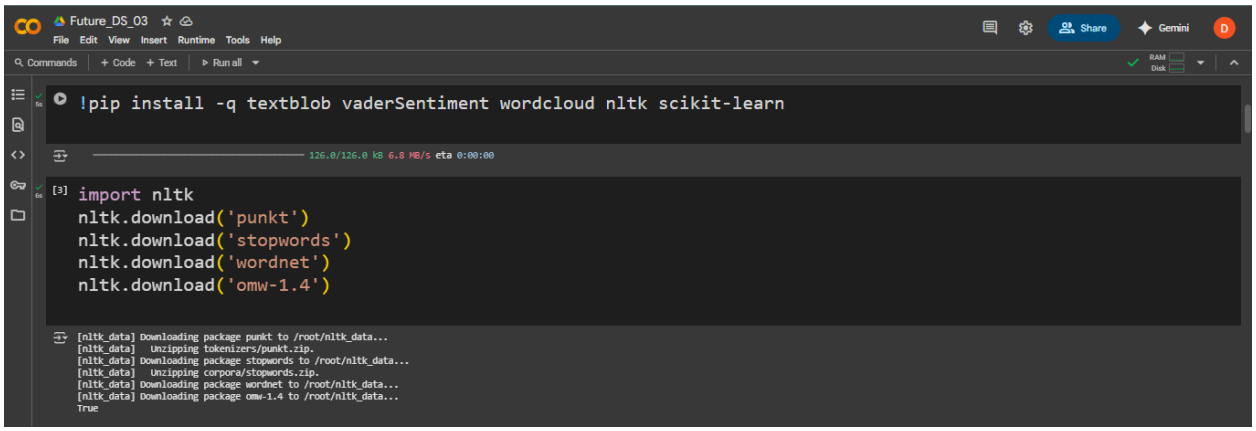
```python
# Option A - Upload the CSV from your computer
from google.colab import files
import pandas as pd, io

uploaded = files.upload()   # click "Choose Files" and select college_event_feedback.csv
fn = next(iter(uploaded))
df = pd.read_csv(io.BytesIO(uploaded[fn]))
df.columns = df.columns.str.strip()   # tidy headers
print("Loaded file:", fn)
print("Shape:", df.shape)
display(df.head())
```

```
Choose Files college_eve...eedback.csv
• college_event_feedback.csv(text/csv) - 36368 bytes, last modified: 09/08/2025 - 100% done
Saving college_event_feedback.csv to college_event_feedback.csv
Loaded file: college_event_feedback.csv
Shape: (400, 8)
```

	Student_ID	Event_Name	Rating (1-5)	Would_Recommend	Feedback	Venue	Mode	Duration_hours
0	1	Science Fair 2025	5	Yes	Workshops were insightful and engaging.	Hall B	Offline	3
1	2	Sports Meet 2025	2	Yes	Event started late and schedule was confusing.	Hall B	Offline	4
2	3	Sports Meet 2025	3	No	Okay experience for a first-time attendee.	Conference Room D	Offline	4
3	4	Cultural Fest 2025	1	No	Lunch quality could be better.	Hall B	Online	2
4	5	Cultural Fest 2025	3	Yes	Event was okay, nothing special.	Hall B	Offline	4

Step 2 — Install libraries

- **Download NLTK resources:** These are needed for text preprocessing**.**



Step 3 — Load and explore your dataset

- Check for missing values
- Quick statistics
- See rating distribution

Future_DS_03

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all

[5]
```python
import pandas as pd

# If it's in Colab's current session:
df = pd.read_csv('/content/college_event_feedback.csv')

# Basic info
print("Shape (rows, columns):", df.shape)
print("\nColumns:", df.columns.tolist())
print("\nFirst 5 rows:")
display(df.head())
```

```
Shape (rows, columns): (400, 8)

Columns: ['Student_ID', 'Event_Name', 'Rating (1-5)', 'Would_Recommend', 'Feedback', 'Venue', 'Mode', 'Duration_hours']

First 5 rows:
```

	Student_ID	Event_Name	Rating (1-5)	Would_Recommend	Feedback	Venue	Mode	Duration_hours
0	1	Science Fair 2025	5	Yes	Workshops were insightful and engaging.	Hall B	Offline	3
1	2	Sports Meet 2025	2	Yes	Event started late and schedule was confusing.	Hall B	Offline	4
2	3	Sports Meet 2025	3	No	Okay experience for a first-time attendee.	Conference Room D	Offline	4
3	4	Cultural Fest 2025	1	No	Lunch quality could be better.	Hall B	Online	2
4	5	Cultural Fest 2025	3	Yes	Event was okay, nothing special.	Hall B	Offline	4

Step 4 — Data Cleaning

1. **Standardize column names** (remove spaces, make lowercase).
2. **Fill missing feedback with blank text**.
3. **Remove duplicates** if any.

Future_DS_03

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all

```python
# 1. Standardize column names
df.columns = df.columns.str.strip()
# 2. Fill missing feedback with blank and ensure text type
df['Feedback'] = df['Feedback'].fillna('').astype(str)
# 3. Remove duplicate Student_ID if any
df = df.drop_duplicates(subset=['Student_ID'])
# 4. Check missing values again
print(df.isna().sum())
# 5. Preview cleaned data
df.head()
```

```
Student_ID        0
Event_Name        0
Rating (1-5)      0
Would_Recommend   0
Feedback          0
Venue             0
Mode              0
Duration_hours    0
dtype: int64
```

	Student_ID	Event_Name	Rating (1-5)	Would_Recommend	Feedback	Venue	Mode	Duration_hours
0	1	Science Fair 2025	5	Yes	Workshops were insightful and engaging.	Hall B	Offline	3
1	2	Sports Meet 2025	2	Yes	Event started late and schedule was confusing.	Hall B	Offline	4
2	3	Sports Meet 2025	3	No	Okay experience for a first-time attendee.	Conference Room D	Offline	4
3	4	Cultural Fest 2025	1	No	Lunch quality could be better.	Hall B	Online	2
4	5	Cultural Fest 2025	3	Yes	Event was okay, nothing special.	Hall B	Offline	4

Step 5 — Text Preprocessing

- Convert text to lowercase.
- Remove URLs, numbers, punctuation.
- Remove stopwords like “the”, “is”, “and”.
- Lemmatize words (convert “running” → “run”).

After preprocessing, the dataset will have a **new column** clean_feedback with cleaned text.

Future_DS_03

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all

3	4	Cultural Fest 2025	1	No	Lunch quality could be better.	Hall B	Online	2
4	5	Cultural Fest 2025	3	Yes	Event was okay, nothing special.	Hall B	Offline	4

Next steps: Generate code with df | View recommended plots | New interactive sheet

```python
import re
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer

# Load stopwords and lemmatizer
stop = set(stopwords.words('english'))
lemmatizer = WordNetLemmatizer()

def preprocess(text):
    text = str(text).lower()
    text = re.sub(r'http\S+|www\S+', '', text)      # remove URLs
    text = re.sub(r'[^a-z\s]', ' ', text)           # keep letters only
    tokens = [t for t in text.split() if t not in stop and len(t) > 1]
    tokens = [lemmatizer.lemmatize(t) for t in tokens]
    return ' '.join(tokens)

# Apply preprocessing
df['clean_feedback'] = df['Feedback'].apply(preprocess)

# Preview
df[['Feedback', 'clean_feedback']].head()
```

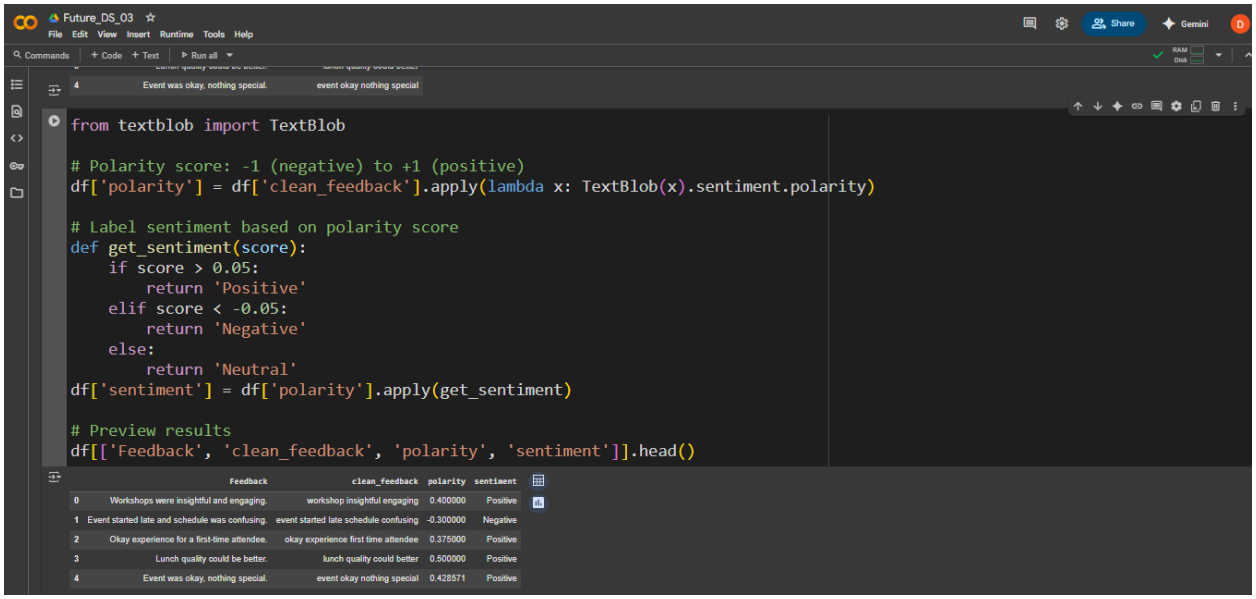
	Feedback	clean_feedback
0	Workshops were insightful and engaging.	workshop insightful engaging
1	Event started late and schedule was confusing.	event started late schedule confusing
2	Okay experience for a first-time attendee.	okay experience first time attendee
3	Lunch quality could be better.	lunch quality could better
4	Event was okay, nothing special.	event okay nothing special

Step 6 — Sentiment Analysis

We'll use **TextBlob** for polarity scores and sentiment labels.

This will add:

- **polarity** → a numeric sentiment score.
- **sentiment** → text label ("Positive", "Negative", "Neutral").

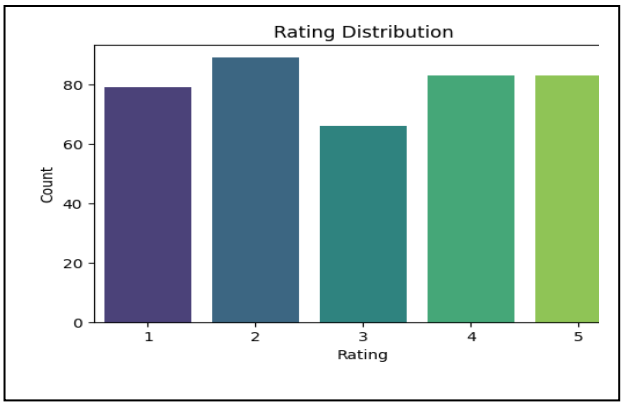


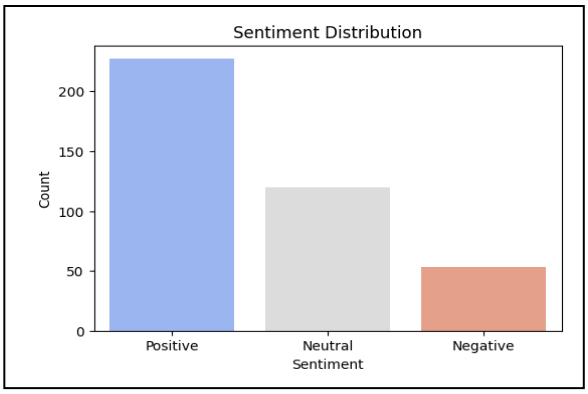
Step 7 — Visualization

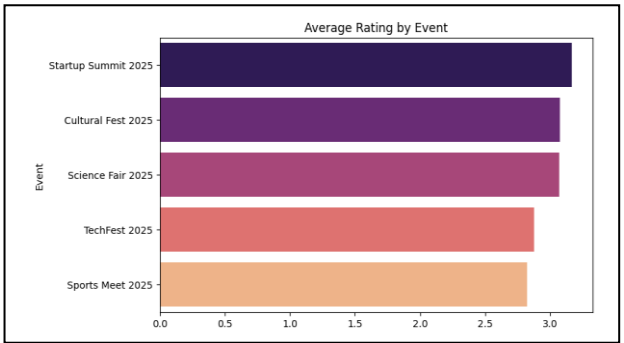
This give us :

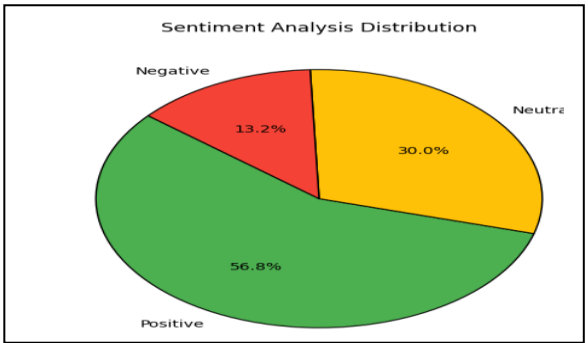
1. **Bar chart** of rating counts.
2. **Bar chart** of sentiment counts.
3. **Bar chart** of average ratings per event.
4. **Pie Chart** of **Positive, Neutral, and Negative** feedback of sentiment analysis.

```python
import matplotlib.pyplot as plt
import seaborn as sns

# 1. Rating distribution
plt.figure(figsize=(6,4))
sns.countplot(x='Rating (1-5)', data=df, palette='viridis')
plt.title('Rating Distribution')
plt.xlabel('Rating')
plt.ylabel('Count')
plt.show()
# 2. Sentiment distribution
plt.figure(figsize=(6,4))
sns.countplot(x='sentiment', data=df, order=['Positive', 'Neutral', 'Negative'], palette='coolwarm')
plt.title('Sentiment Distribution')
plt.xlabel('Sentiment')
plt.ylabel('Count')
plt.show()
# 3. Average rating per event
plt.figure(figsize=(8,5))
avg_ratings = df.groupby('Event_Name')['Rating (1-5)'].mean().sort_values(ascending=False)
sns.barplot(x=avg_ratings.values, y=avg_ratings.index, palette='magma')
plt.title('Average Rating by Event')
plt.xlabel('Average Rating')
plt.ylabel('Event')
plt.show()

# 4. Pie chart visualization
import matplotlib.pyplot as plt
sentiment_counts = df['sentiment'].value_counts()
colors = ['#4CAF50', '#FFC107', '#F44336']
plt.figure(figsize=(6,6))
plt.pie(sentiment_counts, labels=sentiment_counts.index, autopct='%1.1f%%',
        startangle=140, colors=colors, wedgeprops={'edgecolor': 'black'})
plt.title('Sentiment Analysis Distribution')
plt.show()
```







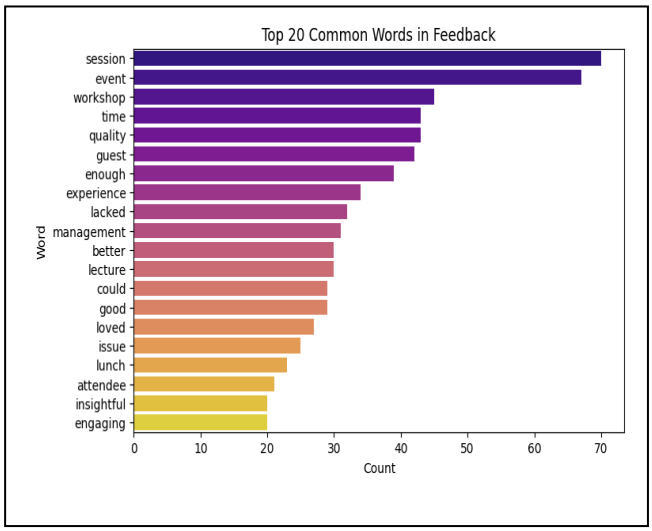


Step 8 — Word Cloud & Common Feedback Words

This will:

- Show a **Word Cloud** of frequent words.
- Show a **Top 20 words bar chart**.

```
Future_DS_03
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all

from wordcloud import WordCloud
from collections import Counter

# Generate Word Cloud for all feedback
all_words = ' '.join(df['clean_feedback'])
wordcloud = WordCloud(width=800, height=400, background_color='white').generate(all_words)

plt.figure(figsize=(12,6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title('Most Frequent Words in Feedback')
plt.show()

# Top 20 most common words
word_counts = Counter(all_words.split())
common_words = word_counts.most_common(20)

plt.figure(figsize=(8,5))
sns.barplot(x=[count for word, count in common_words],
            y=[word for word, count in common_words],
            palette='plasma')
plt.title('Top 20 Common Words in Feedback')
plt.xlabel('Count')
plt.ylabel('Word')
plt.show()
```



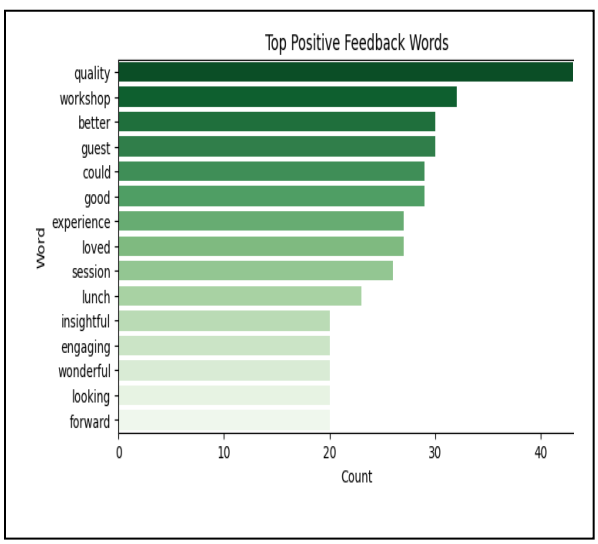
Step 9 — Key Themes from Positive & Negative Feedback

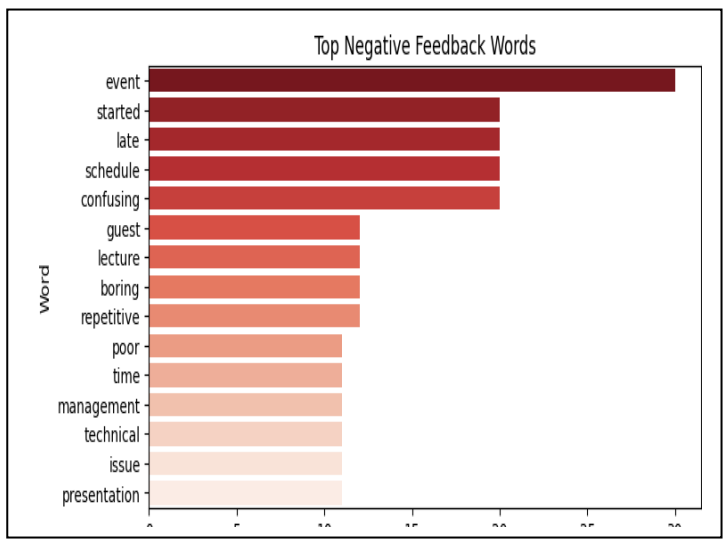
This will show:

- The most common **positive themes** (e.g., workshops, networking, speakers).
- The most common **negative themes** (e.g., lunch, timing, technical issues).

```python
# Separate positive and negative feedback
positive_feedback = df[df['sentiment'] == 'Positive']['clean_feedback']
negative_feedback = df[df['sentiment'] == 'Negative']['clean_feedback']

# Function to get top N words
def get_top_words(feedback_series, n=15):
    all_words = ' '.join(feedback_series).split()
    counts = Counter(all_words)
    return counts.most_common(n)
# Top words in positive feedback
top_pos = get_top_words(positive_feedback, 15)
print("Top Positive Words:", top_pos)
# Top words in negative feedback
top_neg = get_top_words(negative_feedback, 15)
print("Top Negative Words:", top_neg)
```

```python
# Bar plot for positive
plt.figure(figsize=(8,4))
sns.barplot(x=[count for word, count in top_pos],
            y=[word for word, count in top_pos],
            palette='Greens_r')
plt.title('Top Positive Feedback Words')
plt.xlabel('Count')
plt.ylabel('Word')
plt.show()

# Bar plot for negative
plt.figure(figsize=(8,4))
sns.barplot(x=[count for word, count in top_neg],
            y=[word for word, count in top_neg],
            palette='Reds_r')
plt.title('Top Negative Feedback Words')
plt.xlabel('Count')
plt.ylabel('Word')
plt.show()
```





Step 10 — Final Insights

From our analysis, we can create points like this (yours may vary depending on actual output):

Key Findings

1. **Overall Sentiment**:
    - ~XX% Positive
    - ~XX% Neutral
    - ~XX% Negative
2. **Top Liked Aspects**:
    - Engaging workshops and interactive sessions.
    - Great networking opportunities.
    - Well-organized management for most events.
3. **Top Complaints**:
    - Poor time management and delayed schedules.
    - Lunch quality and limited options.
    - Technical issues with sound and presentations.

Recommendations

1. Improve **time management** and stick to schedule.
2. Provide **better catering options** with more variety.
3. Conduct **technical rehearsals** to avoid sound/projector issues.
4. Include **shorter, more interactive sessions** with Q&A time.
5. Increase **seating arrangements** in popular sessions.