# Importing Necessary Libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.cluster import KMeans

from sklearn.decomposition import PCA

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.preprocessing import StandardScaler

from nltk.sentiment.vader import SentimentIntensityAnalyzer

import nltk

nltk.download('vader\_lexicon')

# Loading Datasets

metadata = pd.read\_csv('Data/Restaurant names and Metadata.csv')

reviews = pd.read\_csv('Data/ Restaurant reviews.csv')

# Data Preprocessing

# Cleaning and preparing metadata

metadata.dropna(subset=['Cost'], inplace=True)

metadata['Cost'] = metadata['Cost'].str.replace(',', '').astype(float)

# Cleaning and preparing reviews

reviews.dropna(subset=['Review'], inplace=True)

# Feature Engineering

# Extracting features from metadata

metadata['Cuisine\_Count'] = metadata['Cuisines'].apply(lambda x: len(x.split(',')))

metadata['Timings\_Count'] = metadata['Timings'].apply(lambda x: len(x.split(',')) if pd.notnull(x) else 0)

# Sentiment Analysis on Reviews

sid = SentimentIntensityAnalyzer()

reviews['Sentiment\_Score'] = reviews['Review'].apply(lambda x: sid.polarity\_scores(x)['compound'])

# Aggregating sentiment scores with metadata

avg\_sentiment = reviews.groupby('Restaurant')['Sentiment\_Score'].mean().reset\_index()

metadata = metadata.merge(avg\_sentiment, left\_on='Name', right\_on='Restaurant', how='left')

# Clustering Restaurants

features = ['Cost', 'Cuisine\_Count', 'Sentiment\_Score']

metadata.dropna(subset=features, inplace=True)

X = metadata[features]

# Standardizing features

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

# Applying KMeans Clustering

kmeans = KMeans(n\_clusters=5, random\_state=42)

metadata['Cluster'] = kmeans.fit\_predict(X\_scaled)

# Dimensionality Reduction for Visualization

pca = PCA(n\_components=2)

components = pca.fit\_transform(X\_scaled)

metadata['Component\_1'] = components[:, 0]

metadata['Component\_2'] = components[:, 1]

# Visualization

plt.figure(figsize=(10, 6))

sns.scatterplot(data=metadata, x='Component\_1', y='Component\_2', hue='Cluster', palette='viridis')

plt.title('Restaurant Clusters')

plt.show()