Intenship project

Title: Sentiment Analysis

Subtitle: Decoding Emotions: Sentiment Analysis of Text Data Using NLP and Machine Learning

Presented by: Faleye Doyin Opeyemi

Date: 15-8-2025

1. Data Preparation & Feature Engineering

In [316]:
```
# Install required libraries (run once)

!pip install nltk textblob scikit-learn matplotlib seaborn wordcloud

# Import libraries

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import nltk
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
import re
from wordcloud import WordCloud
import string
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics import accuracy_score
from sklearn.tree import DecisionTreeClassifier
```

```
Requirement already satisfied: nltk in c:\users\faleye doyinsola\anaconda3\lib
\site-packages (3.9.1)
Requirement already satisfied: textblob in c:\users\faleye doyinsola\anaconda3
\lib\site-packages (0.19.0)
Requirement already satisfied: scikit-learn in c:\users\faleye doyinsola\anacon
da3\lib\site-packages (1.6.1)
Requirement already satisfied: matplotlib in c:\users\faleye doyinsola\anaconda
3\lib\site-packages (3.7.0)
Requirement already satisfied: seaborn in c:\users\faleye doyinsola\anaconda3\l
ib\site-packages (0.12.2)
Requirement already satisfied: wordcloud in c:\users\faleye doyinsola\anaconda3
\lib\site-packages (1.9.4)
Requirement already satisfied: regex>=2021.8.3 in c:\users\faleye doyinsola\ana
conda3\lib\site-packages (from nltk) (2022.7.9)
Requirement already satisfied: joblib in c:\users\faleye doyinsola\anaconda3\li
b\site-packages (from nltk) (1.4.2)
Requirement already satisfied: click in c:\users\faleye doyinsola\anaconda3\lib
\site-packages (from nltk) (8.0.4)
Requirement already satisfied: tqdm in c:\users\faleye doyinsola\anaconda3\lib
\site-packages (from nltk) (4.64.1)
Requirement already satisfied: scipy>=1.6.0 in c:\users\faleye doyinsola\anacon
da3\lib\site-packages (from scikit-learn) (1.10.0)
Requirement already satisfied: threadpoolctl>=3.1.0 in c:\users\faleye doyinsol
a\anaconda3\lib\site-packages (from scikit-learn) (3.5.0)
Requirement already satisfied: numpy>=1.19.5 in c:\users\faleye doyinsola\anaco
nda3\lib\site-packages (from scikit-learn) (1.23.5)
Requirement already satisfied: cycler>=0.10 in c:\users\faleye doyinsola\anacon
da3\lib\site-packages (from matplotlib) (0.11.0)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\faleye doyinsola\an
aconda3\lib\site-packages (from matplotlib) (1.0.5)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\faleye doyinsol
a\anaconda3\lib\site-packages (from matplotlib) (2.8.2)
Requirement already satisfied: packaging>=20.0 in c:\users\faleye doyinsola\ana
conda3\lib\site-packages (from matplotlib) (22.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\faleye doyinsola\an
aconda3\lib\site-packages (from matplotlib) (3.0.9)
Requirement already satisfied: pillow>=6.2.0 in c:\users\faleye doyinsola\anaco
nda3\lib\site-packages (from matplotlib) (9.4.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\faleye doyinsola\a
naconda3\lib\site-packages (from matplotlib) (4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\faleye doyinsola\a
naconda3\lib\site-packages (from matplotlib) (1.4.4)
Requirement already satisfied: pandas>=0.25 in c:\users\faleye doyinsola\anacon
da3\lib\site-packages (from seaborn) (1.5.3)
Requirement already satisfied: pytz>=2020.1 in c:\users\faleye doyinsola\anacon
da3\lib\site-packages (from pandas>=0.25->seaborn) (2022.7)
Requirement already satisfied: six>=1.5 in c:\users\faleye doyinsola\anaconda3
\lib\site-packages (from python-dateutil>=2.7->matplotlib) (1.16.0)
Requirement already satisfied: colorama in c:\users\faleye doyinsola\anaconda3
\lib\site-packages (from click->nltk) (0.4.6)
```

In [318]:
```
df = pd.read_csv("C:\\Users\\FALEYE DOYINSOLA\\user_reviews project sentiment ana
```

In [319]:
```
df.head()
```

Out[319]:

	App	Translated_Review	Sentiment	Sentiment_Polarity	Sentiment_Subjectivity
0	10 Best Foods for You	I like eat delicious food. That's I'm cooking ...	Positive	1.00	0.533333
1	10 Best Foods for You	This help eating healthy exercise regular basis	Positive	0.25	0.288462
2	10 Best Foods for You	NaN	NaN	NaN	NaN
3	10 Best Foods for You	Works great especially going grocery store	Positive	0.40	0.875000
4	10 Best Foods for You	Best idea us	Positive	1.00	0.300000

In [320]:
```
# checking the columns
df.columns
```

Out[320]:
```
Index(['App', 'Translated_Review', 'Sentiment', 'Sentiment_Polarity',
       'Sentiment_Subjectivity'],
      dtype='object')
```

In [321]:
```
#checking for missing values
df.isnull().sum()
```

Out[321]:
```
App                        0
Translated_Review      26868
Sentiment              26863
Sentiment_Polarity     26863
Sentiment_Subjectivity 26863
dtype: int64
```

In [322]:
```
# checking for duplicate
df.duplicated()
```

Out[322]:
```
0        False
1        False
2        False
3        False
4        False
         ...
64290     True
64291     True
64292     True
64293     True
64294     True
Length: 64295, dtype: bool
```

In [323]:
```
# droping all duplicated
df.drop_duplicates(inplace=True)
df.duplicated()
```

Out[323]:
```
0        False
1        False
2        False
3        False
4        False
         ...
64223    False
64226    False
64227    False
64230    False
64236    False
Length: 30679, dtype: bool
```

In [324]:
```
# remove the unwanted columns
df.drop(['App', 'Sentiment_Polarity', 'Sentiment_Subjectivity'],axis =1, inplace
```

In [325]:
```
#preveiw the data
df.head()
```

Out[325]:

	Translated_Review	Sentiment
0	I like eat delicious food. That's I'm cooking ...	Positive
1	This help eating healthy exercise regular basis	Positive
2	NaN	NaN
3	Works great especially going grocery store	Positive
4	Best idea us	Positive

In [326]:
```
# dropping all the Missing values
df.dropna(inplace=True)
df.isnull().sum()
```

Out[326]:
```
Translated_Review    0
Sentiment            0
dtype: int64
```

In [327]:
```
# previewing 10 list of the 'Translated_Review' column
df['Translated_Review']. iloc[:10]
```

Out[327]:
```
0     I like eat delicious food. That's I'm cooking ...
1       This help eating healthy exercise regular basis
3            Works great especially going grocery store
4                                          Best idea us
5                                              Best way
6                                               Amazing
8                                Looking forward app,
9                 It helpful site ! It help foods get !
10                                             good you.
11    Useful information The amount spelling errors ...
Name: Translated_Review, dtype: object
```

2. Text Preprocessing (NLP)

In [328]:
```
# Define preprocessing function,
# we are using this function for the User Review dataset coz the Translated_Revie
# it also consist of upper and lower case word in a sentence
import string
def clean_review(text):
    text = text.lower() # lowercase
    text= re.sub(r"http\S+|www\S+https\S+", '',text, flags=re.MULTILINE) # regre
    text= text.translate(str.maketrans('','',string.punctuation)) # removing all
    text = re.sub(r'\d+', '', text) # remove all numbers
    return text
```

In [329]:
```
## let run the clean review function we created
df['Clean_review']= df['Translated_Review'].astype(str).apply(clean_review)
```

In [330]:
```
# let preview the dataset again to see if the function worked
df.head()
```

Out[330]:

	Translated_Review	Sentiment	Clean_review
0	I like eat delicious food. That's I'm cooking ...	Positive	i like eat delicious food thats im cooking foo...
1	This help eating healthy exercise regular basis	Positive	this help eating healthy exercise regular basis
3	Works great especially going grocery store	Positive	works great especially going grocery store
4	Best idea us	Positive	best idea us
5	Best way	Positive	best way

In [331]:
```
# previewing 10 list of the 'clean_review' column
df['Clean_review']. iloc[:10]
```

Out[331]:
```
0     i like eat delicious food thats im cooking foo...
1       this help eating healthy exercise regular basis
3            works great especially going grocery store
4                                          best idea us
5                                              best way
6                                               amazing
8                                    looking forward app
9                   it helpful site  it help foods get
10                                             good you
11    useful information the amount spelling errors ...
Name: Clean_review, dtype: object
```

In [332]:
```
# let drop Translated column because its not useful anymore
df.drop(['Translated_Review'],axis =1, inplace=True)
```

In [333]:
```
# let preview the dataset
df.head()
```

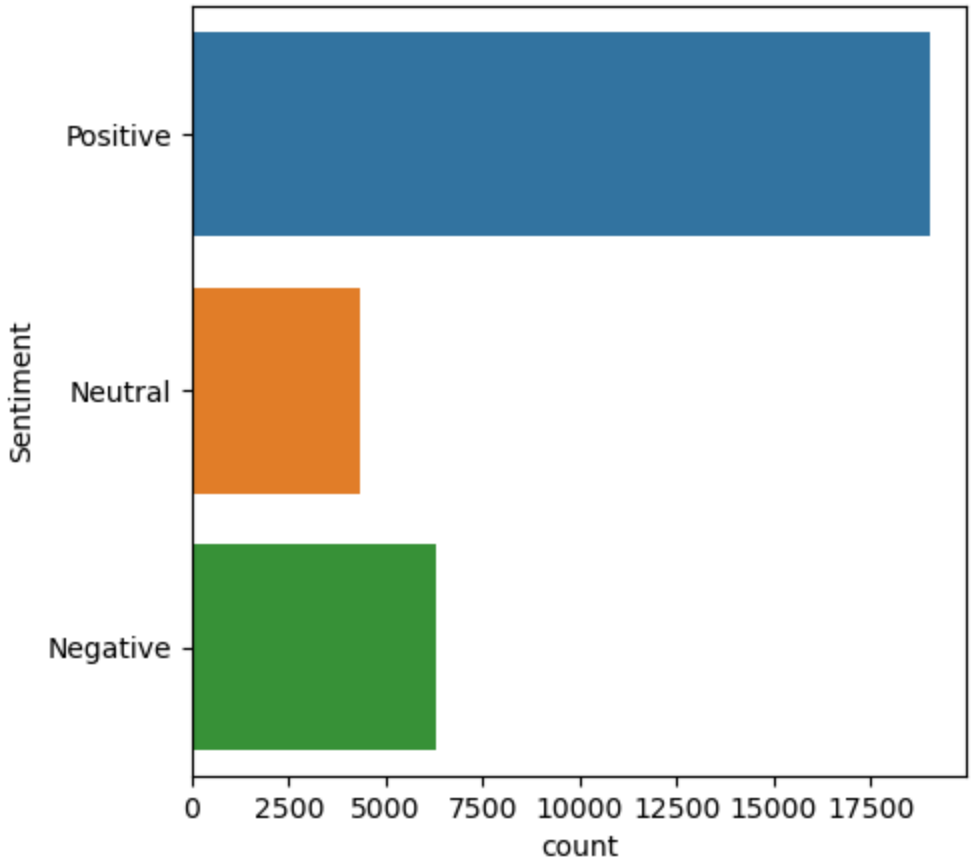
Out[333]:

	Sentiment	Clean_review
0	Positive	i like eat delicious food thats im cooking foo...
1	Positive	this help eating healthy exercise regular basis
3	Positive	works great especially going grocery store
4	Positive	best idea us
5	Positive	best way

3. Sentiment Labeling

In [ ]:
```
```

In [261]:
```
plt.figure(figsize = (5,5))
sns.countplot(df, y = 'Sentiment')
plt.show()
```



4. Machine Learning Models

In [302]:
```
from sklearn.preprocessing import LabelEncoder
# let change the text value into numeric values
```

In [267]:
```
Label = LabelEncoder()
```

In [269]:
```
df['Sentiment'] = Label.fit_transform(df['Sentiment'])
```

In [272]:
```
df.head()
```

Out[272]:

	Sentiment	Clean_review
0	2	i like eat delicious food thats im cooking foo...
1	2	this help eating healthy exercise regular basis
3	2	works great especially going grocery store
4	2	best idea us
5	2	best way

In [273]:
```
x = df['Clean_review']
y = df['Sentiment']
```

In [279]:
```
# Vectorize text
# apply tfidf

Vec = TfidfVectorizer(max_features= 5000, stop_words= 'english')
```

In [280]:
```
xtfid = Vec.fit_transform(x)
```

In [281]:
```
xtrain,xtest,ytrain,ytest = train_test_split(xtfid,y,test_size=0.2,random_state=
```

In [282]:
```
xtrain.shape
```

Out[282]:
```
(23753, 5000)
```

In [283]:
```
xtest.shape
```

Out[283]:
```
(5939, 5000)
```

In [285]:
```
dtc =DecisionTreeClassifier()
```

In [286]:
```
dtc.fit(xtrain,ytrain)
```

Out[286]:
```
DecisionTreeClassifier()
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [288]:
```
dtcpred = dtc.predict(xtest)
```

In [289]:
```
dtcpred[:10]
```

Out[289]:
```
array([1, 0, 2, 2, 0, 2, 1, 0, 2, 2])
```

In [290]:
```
dtc_accuracy = accuracy_score(dtcpred,ytest)*100
```

In [291]:
```
dtc_accuracy
```

Out[291]:
```
84.25660885670987
```

In [292]:
```
log = LogisticRegression()
```

In [293]:
```
ytrain.shape
```

Out[293]:
```
(23753,)
```

In [294]:
```
log.fit(xtrain,ytrain)
```

```
C:\Users\FALEYE DOYINSOLA\anaconda3\lib\site-packages\sklearn\linear_model\_log
istic.py:465: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-
learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regressi
on (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regressi
on)
  n_iter_i = _check_optimize_result(
```

Out[294]:
```
LogisticRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [295]:
```
# Prediction
logpred = log.predict(xtest)
```

In [296]:
```
logpred[:10]
```

Out[296]:
```
array([1, 0, 0, 2, 0, 2, 1, 0, 2, 2])
```

In [297]:
```
from sklearn.metrics import accuracy_score
```

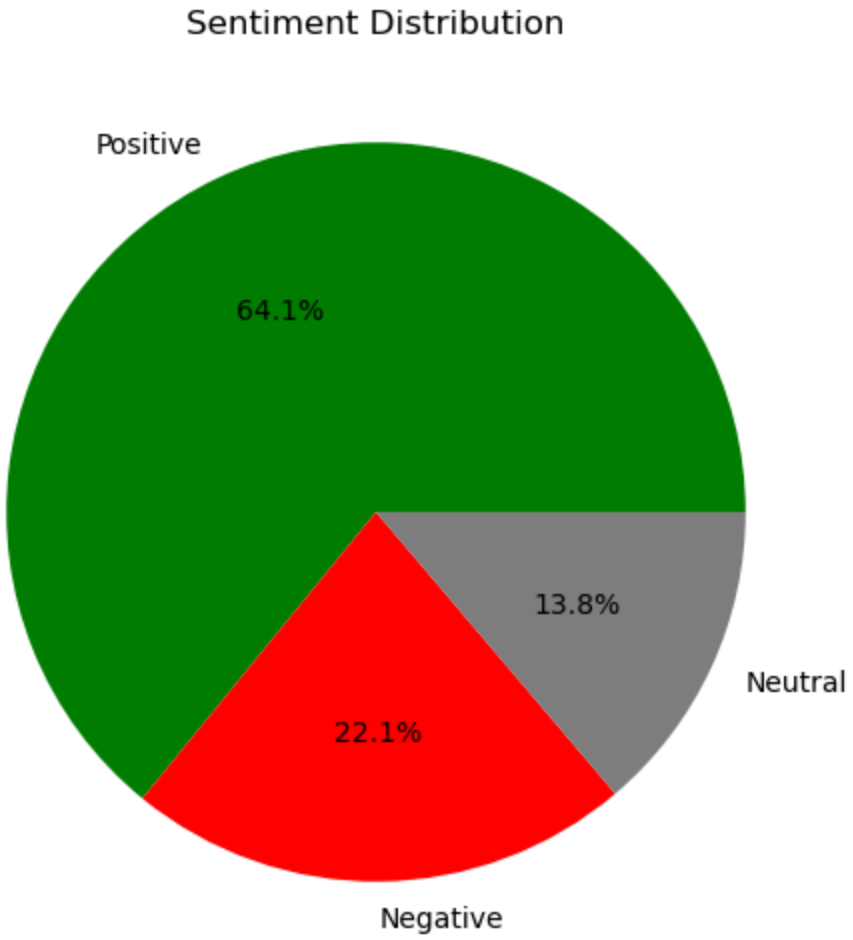
In [298]:
```
# checking for accuracy_score
logacc= accuracy_score(ytest,logpred)*100
```

In [299]:
```
logacc
```

Out[299]:
```
87.8430712241118
```

In [ ]:
```
# 5. Data Visualization
```

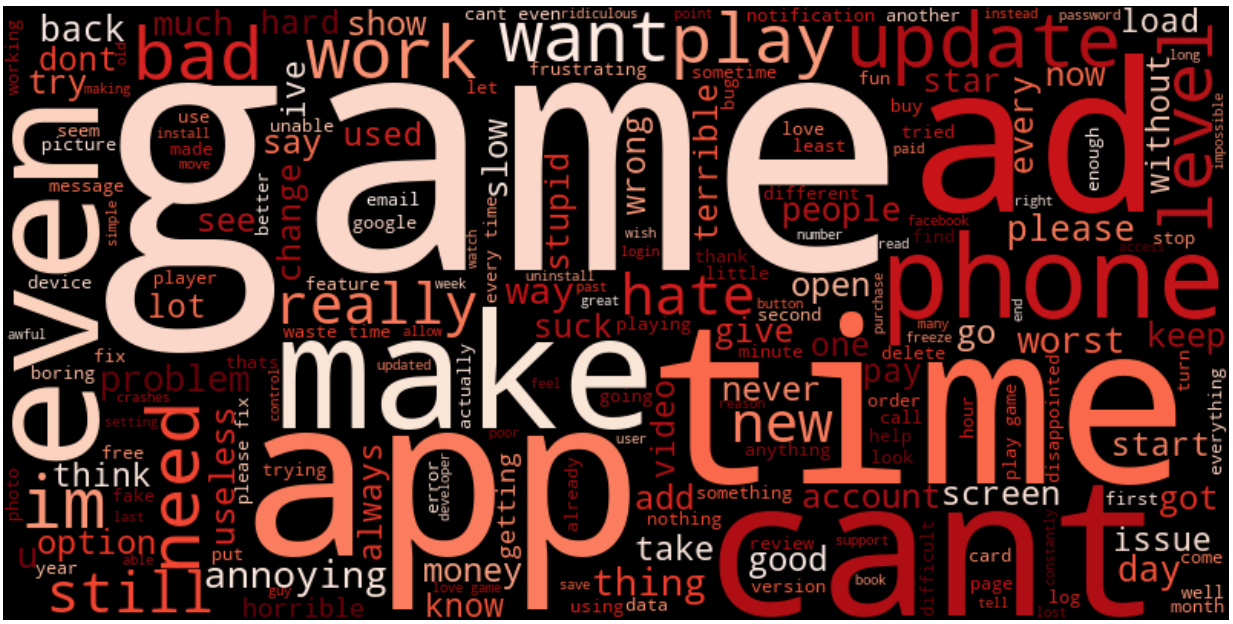
In [307]:
```
plt.figure(figsize=(6,6))
df['Sentiment'].value_counts().plot(kind='pie', autopct='%1.1f%%', colors=['greer
plt.title('Sentiment Distribution')
plt.ylabel('')
plt.show()
```



In [334]:
```
# Word cloud for positive sentiment
positive_text = ' '.join(df[df['Sentiment'] == 'Positive']['Clean_review'])
WordCloud(width=800, height=400, background_color='white').generate(positive_text

# Word cloud for negative sentiment
negative_text = ' '.join(df[df['Sentiment'] == 'Negative']['Clean_review'])
WordCloud(width=800, height=400, background_color='black', colormap='Reds').gener
```

Out[334]:



Project Summary

- Applied NLP techniques to clean and preprocess text data.
- Engineered features like text length and punctuation count.
- Trained a Naive Bayes classifier using TF-IDF vectors.
- Visualized sentiment distribution, word clouds, and model performance.

In [ ]:
```
```