SENTIMENT ANALYSIS FOR MARKETING

PHASE :3

DEVELOPMENT PART 01

SUBMITTED BY: Jeevitha C E

MAIL ID: cdekambaram@gmail.com

INTRODUCTION:

- To build a dataset for predicting IMDb scores, you'll need to collect data on numerous features of films. IMDb itself, movie databases like The Movie Database (TMDb) or IMDb API, online scraping, and potentially crowdsourcing are some common sources for this data.

DATA PREPROCESSING:

Once you have collected the data, you'll need to preprocess it to ensure it's clean and ready for analysis. This involves:

Handling missing data: You may need to deal with missing values in any of the fields,possibly by imputing them or removing the corresponding entries.

Data encoding: Categorical data like genre and languages need to be encoded into anumerical format. This can be done

using techniques like one-hot encoding or label encoding

Feature scaling: You might need to scale numeric features like runtime to have similarranges, especially if you plan to use algorithms that are sensitive to feature scales, such asgradient descent-based methods.

Outlier detection: Identify and handle outliers in IMDb scores or runtime that could skewthe predictions.

THE ESSENTIAL LIBRARIES:

We start by importing the main libraires that we will use:

(1) the **re** module (for regular expression matching operations)

(2) the **nltk** toolkit (for natural language operations)

(3) the **random** module (for random number generation)

(4) the **numpy** library (for arrays operations)

(5) the **pandas** library (for data analysis)

(6) the **scipy.stats** module (for statistics)

(7) the **seaborn** library (for statistical data visualization)

(8) the **matplotlib.pyplot** interface (for MATLAB-like plots)

We also download the stopwords and punkt data packages from the nltk toolkit.

In[1]:

```
import re
import nltk
import random
import numpy as np
import pandas as pd
import seaborn as sns
from scipy import stats
import matplotlib.pyplot as plt
#nltk.download('punkt')
#nltk.download('stopwords')
# Setting as large the xtick and ytick font sizes in graphs
plt.rcParams['xtick.labelsize'] = 'large'
plt.rcParams['ytick.labelsize'] = 'large'
```

IMDB DATASET:

We obtain the csv dataset "IMDB Dataset.csv" from Kaggle, which contains 50'000 IMDB movie and TV show reviews with their positive or negative sentiment classification.

In [2]:

```
# Storing the csv file into a DataFrame "df"
df = pd.read_csv('../input/imdb-dataset-of-50k-movie-reviews/IMDB Dataset.csv')
df
```

out[2]:

	review	sentiment
0	One of the other reviewers has mentioned that ...	positive
1	A wonderful little production. The...	positive
2	I thought this was a wonderful way to spend ti...	positive
3	Basically there's a family where a little boy ...	negative

	review	sentiment
4	Petter Mattei's "Love in the Time of Money" is...	positive
...	...	...
49995	I thought this movie did a down right good job...	positive
49996	Bad plot, bad dialogue, bad acting, idiotic di...	negative
49997	I am a Catholic taught in parochial elementary...	negative
49998	I'm going to have to disagree with the previou...	negative
49999	No one expects the Star Trek movie to high	negative

50000rows *2 columns

We print the DataFrame's fundamental attributes. We especially notice that there are no null values in the DataFrame.

In [3]:

```
print('\033[1m' + 'df.shape:' + '\033[0m', df.shape)

print('\033[1m' + 'df.columns:' + '\033[0m', df.columns, '\n')

print('\033[1m' + 'df.sentiment.value_counts():' + '\033[0m')

print(df.sentiment.value_counts(), '\n')

with sns.axes_style("darkgrid"):

    df['sentiment'].value_counts().plot.bar(color=['darkblue', 'r'], rot=0, fontsize='large')

    plt.show()

print('\033[1m' + 'df.info:' + '\033[0m')

df.info()
```

df.shape: (50000, 2)

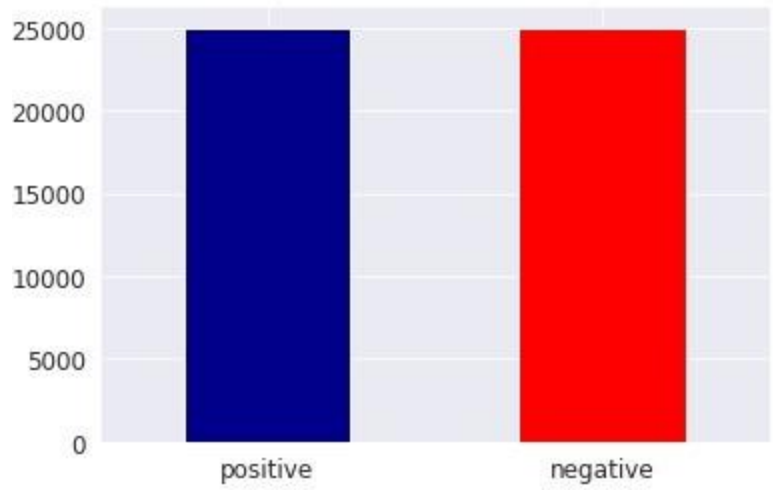
df.columns: Index(['review', 'sentiment'], dtype='object')

df.sentiment.value_counts():

positive : 25000

negative : 25000

Name: sentiment, dtype: int64



df.info:

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 50000 entries, 0 to 49999

Data columns (total 2 columns):

#	Column	Non-Null Count	Dtype
0	review	50000 non-null	object
1	sentiment	50000 non-null	object

dtypes: object(2)

memory usage: 781.4+ KB

To prepare the DataFrame for analysis, transform its sentiment values to integers:

positive → 1

negative → 0

In [4]:

```
df.sentiment = [1 if s == 'positive' else 0 for s in df.sentiment]
df
```

out[4]:

	review	sentiment
0	One of the other reviewers has mentioned that ...	1
1	A wonderful little production. The...	1
2	I thought this was a wonderful way to spend ti...	1
3	Basically there's a family where a little boy ...	0
4	Petter Mattei's "Love in the Time of Money" is...	1
...	...	...
49995	I thought this movie did a down right good job...	1

	review	sentiment
49996	Bad plot, bad dialogue, bad acting, idiotic di...	0
49997	I am a Catholic taught in parochial elementary...	0
49998	I'm going to have to disagree with the previou...	0
49999	No one expects the Star Trek movies to be high...	0

50000 rows × 2 columns

DATA PREPROCESSING:

In this we are going to preprocess the above data ,First, we utilize regular expressions to modify the reviews as following transformations to the reviews:

(1) remove punctuation marks

(2) remove HTML tags

(3) remove URL's

(4) remove characters which are not letters or digits

(5) remove successive whitespaces

(6) convert the text to lower case

(7) strip whitespaces from the beginning and the end of the reviews

In [5]:

Storing in "before_process" a random example of review before preprocessing

Defining and applying the function "process" performing the transformations of the reviews

Storing in "after_process" the example of review after preprocessing

```
idx = random.randint(0, len(df)-1)

before_process = df.iloc[idx][0]


def process(x):
    x = re.sub('[,\.!?:()"]', '', x)
    x = re.sub('<.*?>', ' ', x)
    x = re.sub('http\S+', ' ', x)
    x = re.sub('[^a-zA-Z0-9]', ' ', x)
    x = re.sub('\s+', ' ', x)
    return x.lower().strip()


df['review'] = df['review'].apply(lambda x: process(x))
```

after_process = df.iloc[idx][0]

Then, we use the word_tokenize() method from the nltk.tokenize package to remove stopwords from the reviews.

In [6]:

Storing in "sw_set" the set of English stopwords provided by nltk

Defining and applying the function "sw_remove" which remove stopwords from reviews

Storing in "after_removal" the example of review after removal of the stopwords

```
sw_set = set(nltk.corpus.stopwords.words('english'))


def sw_remove(x):
    words = nltk.tokenize.word_tokenize(x)
    filtered_list = [word for word in words if word not in sw_set]
    return ' '.join(filtered_list)


df['review'] = df['review'].apply(lambda x: sw_remove(x))
after_removal = sw_remove(after_process)
```

[9:47 pm, 17/10/2023] Anitha BME: As an example, we print the review before preprocessing, after preprocessing, and after stopwords removal.

In [7]:

```
print('\033[1m' + 'Review #%d before preprocessing:' % idx + '\033[0m' + '\n', before_process, '\n')

print('\033[1m' + 'Review #%d after preprocessing:' % idx + '\033[0m' + '\n', after_process, '\n')

print('\033[1m' + 'Review #%d after preprocessing and stopwords removal:' % idx + '\033[0m' + '\n', after_removal)
```

Review #49495 before preprocessing:

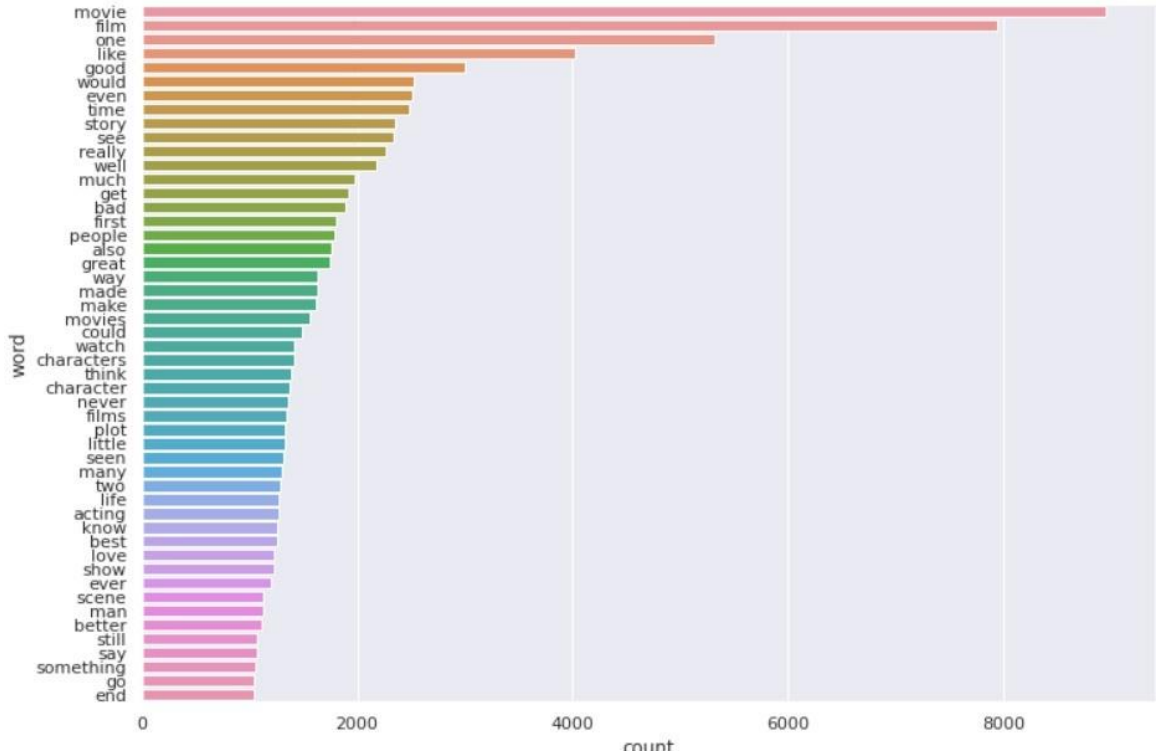
Personally, I think that the film was done very professionally, I loved the choreography and the acting. The plot is also gripping and mysterious. The film itself is very emotional, and what I liked about it most is that it makes you think afterwards. Antonio Gades has absolutely lived his role to the end, and I must say that it's one of my favourite pictures and Saura is a wonderful director.

Review #49495 after preprocessing:

personally i think that the film was done very professionally i loved the choreography and the acting the plot is also gripping and mysterious the film itself is very emotional and what i liked about it most is that it makes you think afterwards antonio gades has absolutely lived his role to the…

Review #49495 after preprocessing and stopwords removal:

personally think film done professionally loved choreography acting plot also gripping mysterious film emotional liked makes think afterwards antonio gades absolutely lived role end must say one favourite pictures saura wonderful director.



CONCLUSION:

Effective data preprocessing in the context of IMDB movie reviews not only refines the raw data but also serves as the foundation for smart analysis and forecasts. Moving forward, we must understand the crucial role of data preprocessing in determining the future of sentiment analysis, allowing us to delve deeper into the complexities of human expression and sentiment in the ever-expanding digital realm.

This preprocessing step not only enhances the efficiency of the sentiment analysis model but also significantly improves its predictive capabilities.