Marmara University
Faculty of Engineering

CSE 4288
Introduction to Machine Learning

Data Preprocessing and EDA

Instructor: Assoc. Prof. Murat Can Ganiz Due Date: 08.12.2024

	Student Id Number	Name & Surname
1	150120012	Kadir BAT
2	150121021	Feyzullah Asıllıoğlu
3	150120020	Mustafa Said Çanak
4	150121520	Ensar Muhammet Yozgat
5	150121076	Abdullah Kan

Table of Contents

1. Introduction

Our aim is to analyze and preprocess the dataset for sentiment classification (positive/negative).

1.1 Acquire and Understand Dataset

Dataset Description:

- Total entries: 50,000
- review: Textual data containing movie reviews.
- sentiment: Binary classification labels ("positive" or "negative").

2. Data Cleaning

2.1 Load Dataset

```
[4] file_path = '/content/IMDB Dataset.csv'
    data = pd.read_csv(file_path)
```

2.2 Removing Duplicates

Identified duplicate reviews in the review column. Removed all duplicates while keeping the first instance.

Before: 50,000 rows
After: 49,582 rows

```
[5] print(f"Original dataset size: {data.shape}")
    data = data.drop_duplicates(subset='review', keep='first')
    print(f"Dataset size after removing duplicates: {data.shape}")

    Original dataset size: (50000, 2)
    Dataset size after removing duplicates: (49582, 2)
```

2.3 Text Cleaning

a. Removed HTML tags (e.g., <br />).

```
[9] data['review'] = data['review'].str.replace(r'<.*?>', '', regex=True)
    print(data['review'])

    0        one of the other reviewers has mentioned that ...
    1        a wonderful little production. the filming tec...
    2        i thought this was a wonderful way to spend ti...
    3        basically there's a family where a little boy ...
    4        petter mattei's "love in the time of money" is...
                                   ...
    49995    i thought this movie did a down right good job...
    49996    bad plot, bad dialogue, bad acting, idiotic di...
    49997    i am a catholic taught in parochial elementary...
    49998    i'm going to have to disagree with the previou...
    49999    no one expects the star trek movies to be high...
    Name: review, Length: 49582, dtype: object
```

b. Converted all text to lowercase.

```
data['review'] = data['review'].str.lower()
print(data['review'])
```

```
0        one of the other reviewers has mentioned that ...
1        a wonderful little production. the filming tec...
2        i thought this was a wonderful way to spend ti...
3        basically there's a family where a little boy ...
4        petter mattei's "love in the time of money" is...
                                   ...
49995    i thought this movie did a down right good job...
49996    bad plot, bad dialogue, bad acting, idiotic di...
49997    i am a catholic taught in parochial elementary...
49998    i'm going to have to disagree with the previou...
49999    no one expects the star trek movies to be high...
Name: review, Length: 49582, dtype: object
```

c. Removed punctuation and special characters.

```
[11] data['review'] = data['review'].apply(lambda x: re.sub(r'[^a-zA-Z\s]', '', x))
     print(data['review'])
```

```
0        one of the other reviewers has mentioned that ...
1        a wonderful little production the filming tech...
2        i thought this was a wonderful way to spend ti...
3        basically theres a family where a little boy j...
4        petter matteis love in the time of money is a ...
                                   ...
49995    i thought this movie did a down right good job...
49996    bad plot bad dialogue bad acting idiotic direc...
49997    i am a catholic taught in parochial elementary...
49998    im going to have to disagree with the previous...
49999    no one expects the star trek movies to be high...
Name: review, Length: 49582, dtype: object
```

2.4 Handling Missing Values

Checked for missing values in both columns. No missing values found in the dataset.

```
[12] print(f"Missing values:\n{data.isnull().sum()}")
```

```
Missing values:
review       0
sentiment    0
dtype: int64
```

3. Exploratory Data Analysis (EDA)

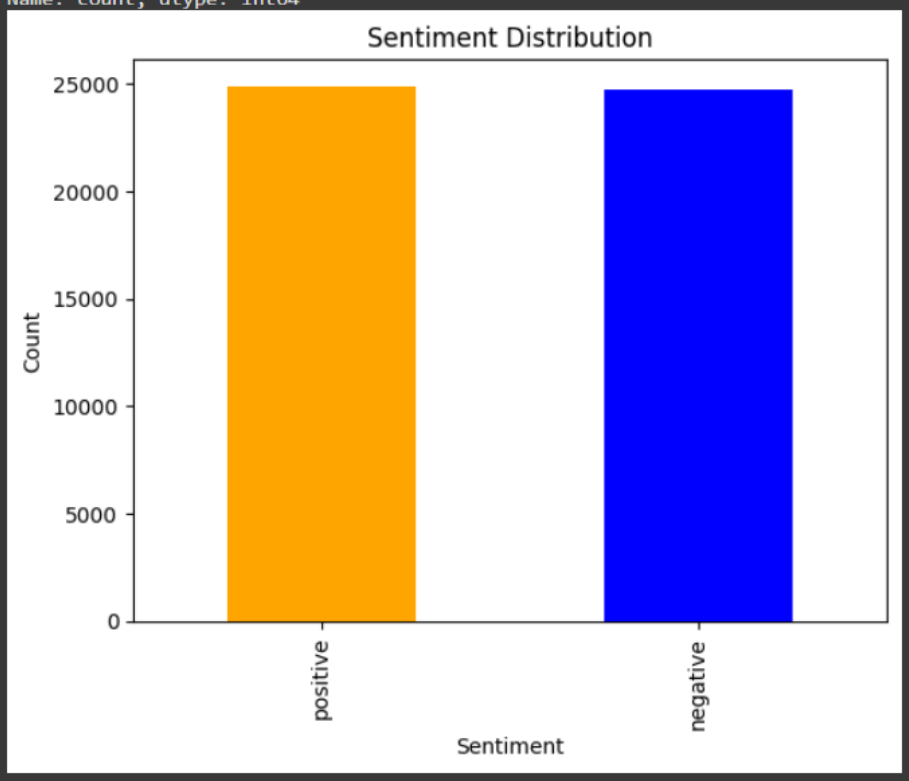
3.1 Sentiment Distribution

Our aim is to check the class balance for sentiment.

- A bar chart showing the distribution of "positive" and "negative" labels.
- Dataset is balanced: 50% positive and 50% negative.
- There are 24884 positive values and 24698 negative values.

```
[13] sentiment_counts = data['sentiment'].value_counts()
     print(f"Sentiment Distribution:\n{sentiment_counts}")
     sentiment_counts.plot(kind='bar', title='Sentiment Distribution', color=['orange', 'blue'])
     plt.xlabel('Sentiment')
     plt.ylabel('Count')
     plt.show()
```

```
Sentiment Distribution:
sentiment
positive    24884
negative    24698
Name: count, dtype: int64
```



3.2 Review Length Analysis

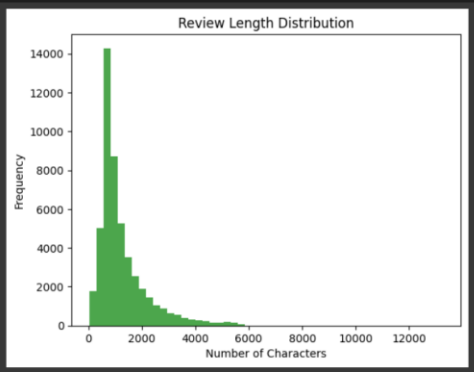
Our aim is to analyze the length of reviews in terms of characters and words.

- Average review length: ~1300 characters
- Shortest review: 10 characters
- Longest review: 20,000 characters.
- Histogram of review lengths.

```
[14] data['review_length'] = data['review'].apply(len)
     print(f"Review Length Statistics:\n{data['review_length'].describe()}")

Review Length Statistics:
count    49582.000000
mean      1242.378202
std        940.232474
min         30.000000
25%        665.000000
50%        922.000000
75%       1506.000000
max      13271.000000
Name: review_length, dtype: float64
```

```
[15] plt.hist(data['review_length'], bins=50, alpha=0.7, color='green')
     plt.title('Review Length Distribution')
     plt.xlabel('Number of Characters')
     plt.ylabel('Frequency')
     plt.show()
```



3.3 Word Count Analysis

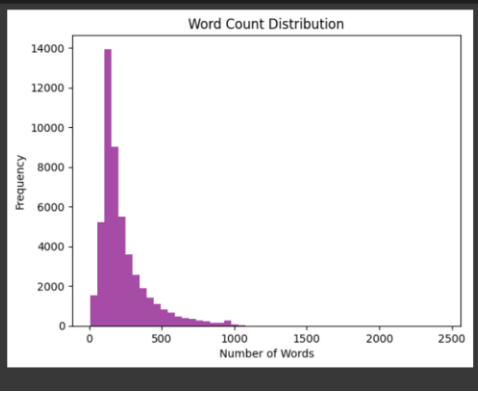
Our aim is to check the distribution of word counts across reviews.

- Most reviews fall between 200-300 words.
- Histogram of word counts.

```
[16] data['word_count'] = data['review'].apply(lambda x: len(x.split()))
     print(f"Word Count Statistics:\n{data['word_count'].describe()}")

Word Count Statistics:
count    49582.000000
mean       225.383002
std        167.129622
min          4.000000
25%        123.000000
50%        169.000000
75%        273.000000
max       2441.000000
Name: word_count, dtype: float64
```

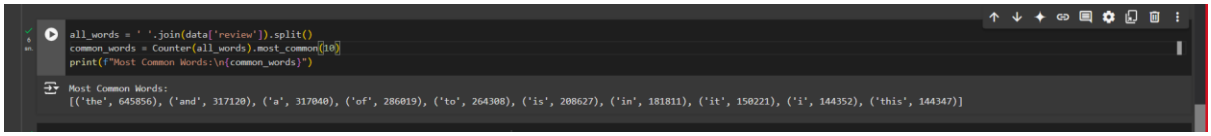
```
[17] plt.hist(data['word_count'], bins=50, alpha=0.7, color='purple')
     plt.title('Word Count Distribution')
     plt.xlabel('Number of Words')
     plt.ylabel('Frequency')
     plt.show()
```



3.4 Most Common Words

Our aim is to identify the most frequent words used in the reviews.

- Common words include "movie," "film," "story," "character," etc.
- Word cloud showcasing frequent words.



```
all_words = ' '.join(data['review']).split()
common_words = Counter(all_words).most_common(10)
print(f"Most Common Words:\n{common_words}")
```

```
Most Common Words:
[('the', 645856), ('and', 317120), ('a', 317040), ('of', 286019), ('to', 264308), ('is', 208627), ('in', 181811), ('it', 150221), ('i', 144352), ('this', 144347)]
```

```
[19] wordcloud = WordCloud(width=800, height=400, background_color='white').generate(' '.join(data['review']))
     plt.figure(figsize=(10, 5))
     plt.title("Word Cloud for All Reviews")
     plt.imshow(wordcloud, interpolation='bilinear')
     plt.axis('off')
     plt.show()
```

3.5 Sentiment-Based Word Frequency

Our aim is to compare word frequencies between positive and negative reviews.

- Separate word clouds for positive and negative reviews.

```
[20] positive_reviews = ' '.join(data[data['sentiment'] == 'positive']['review'])
     negative_reviews = ' '.join(data[data['sentiment'] == 'negative']['review'])
```

- Positive reviews emphasize "great," "love," and "best."

```
positive_words = positive_reviews.split()
positive_common_words = Counter(positive_words).most_common(10)
print(f"Most Common Words in Positive Reviews:\n{positive_common_words}")

Most Common Words in Positive Reviews:
[('the', 331486), ('and', 173642), ('a', 161858), ('of', 150908), ('to', 130070), ('is', 111018), ('in', 97086), ('it', 75145), ('i', 68416), ('this', 66606)]
```

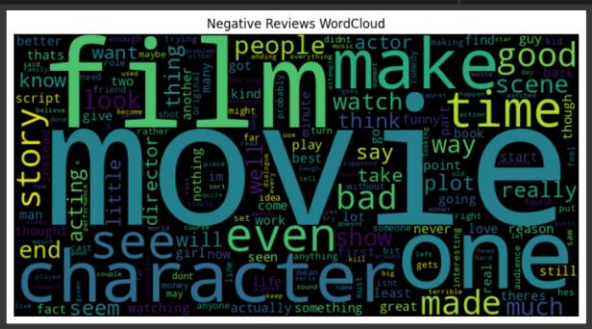
```
[21] pos_wordcloud = WordCloud(width=800, height=400, background_color='white').generate(positive_reviews)
     plt.figure(figsize=(10, 5))
     plt.title("Positive Reviews WordCloud")
     plt.imshow(pos_wordcloud, interpolation='bilinear')
     plt.axis('off')
     plt.show()
```

- Negative reviews emphasize "bad," "worst," and "boring."

```
negative_words = negative_reviews.split()
negative_common_words = Counter(negative_words).most_common(10)
print(f"Most Common Words in Negative Reviews:\n{negative_common_words}")

Most Common Words in Negative Reviews:
[('the', 314370), ('a', 155182), ('and', 143478), ('of', 135111), ('to', 134238), ('is', 97609), ('in', 84725), ('this', 77741), ('i', 75936), ('it', 75076)]
```

```
neg_wordcloud = WordCloud(width=800, height=400, background_color='black').generate(negative_reviews)
plt.figure(figsize=(10, 5))
plt.title("Negative Reviews WordCloud")
plt.imshow(neg_wordcloud, interpolation='bilinear')
plt.axis('off')
plt.show()
```



4. Feature Engineering

4.1 Sentiment Encoding

Converted sentiment into binary values:

- positive → 1
- negative → 0

```
[23] data['sentiment'] = data['sentiment'].map({'positive': 1, 'negative': 0})
```

4.2 Text Vectorization

Used TF-IDF vectorization to convert textual data into numeric format.

- max_features = 5000 (top 5000 words).

```
[24] from sklearn.feature_extraction.text import TfidfVectorizer
     tfidf = TfidfVectorizer(max_features=5000)  # Limit to 5000 most important words
     X = tfidf.fit_transform(data['review'])
     y = data['sentiment']

     print(f"TF-IDF matrix shape: {X.shape}")

     TF-IDF matrix shape: (49582, 5000)
```

5. Conclusion

Dataset Insights:

- The dataset is balanced with an equal distribution of positive and negative reviews.
- Average review length: ~1300 characters or 230 words.
- Common positive words: "great," "love," "amazing."
- Common negative words: "bad," "worst," "boring."

Preprocessing Steps:

- Cleaned text data by removing duplicates, HTML tags, and special characters.
- Normalized case and encoded sentiment labels.

EDA insights:
Reviews are mostly well-written and balanced across sentiments.