DATA VISUALISATION

July 14, 2023

[1]:
```
import pandas as pd
import matplotlib.pyplot as plt
```

[2]:
```
df=pd.read_csv("depression_dataset_reddit_cleaned.csv")
```

[3]:
```
df.head
```

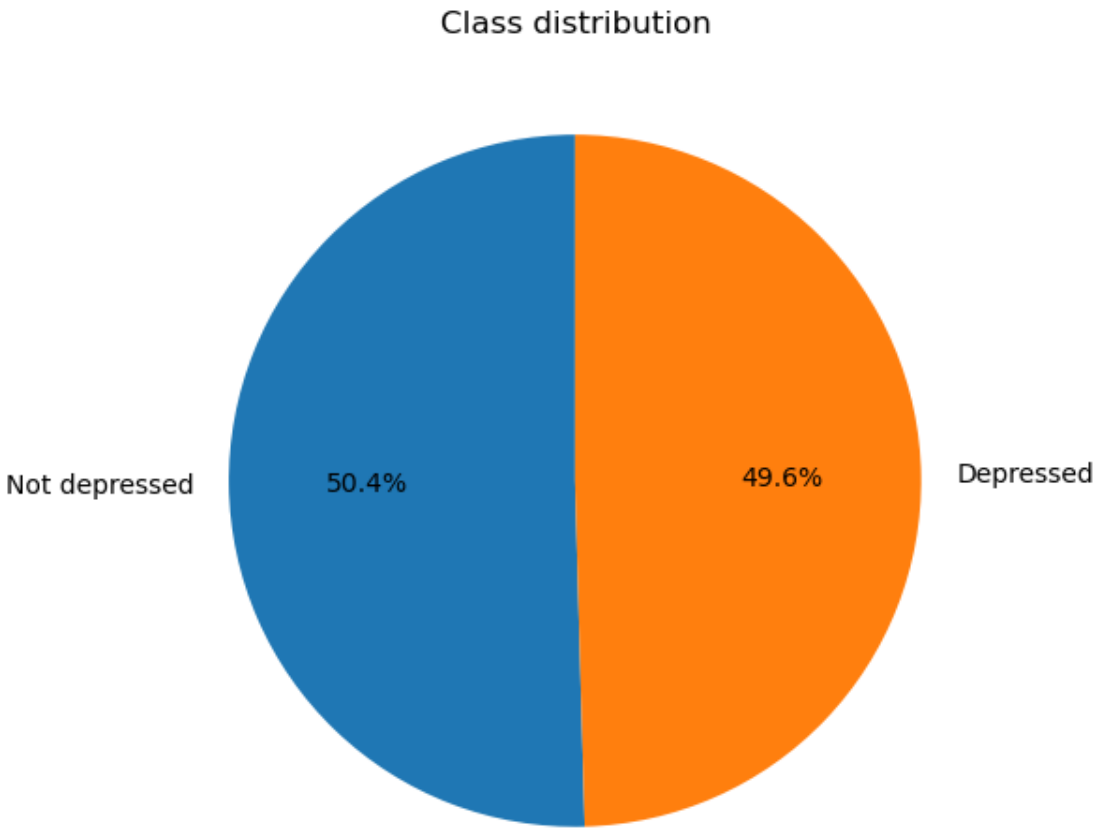
[3]:
```
<bound method NDFrame.head of
clean_text  is_depression
0     we understand that most people who reply immed…              1
1     welcome to r depression s check in post a plac…              1
2     anyone else instead of sleeping more when depr…              1
3     i ve kind of stuffed around a lot in my life d…              1
4     sleep is my greatest and most comforting escap…              1
…                                                   …            …
7726                                        is that snow              0
7727                moulin rouge mad me cry once again              0
7728  trying to shout but can t find people on the list              0
7729  ughh can t find my red sox hat got ta wear thi…              0
7730  slept wonderfully finally tried swatching for …              0

[7731 rows x 2 columns]>
```

[4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7731 entries, 0 to 7730
Data columns (total 2 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   clean_text     7731 non-null   object
 1   is_depression  7731 non-null   int64
dtypes: int64(1), object(1)
memory usage: 120.9+ KB
```

[5]:
```
class_counts = df['is_depression'].value_counts()
```

[6]:
```
plt.figure(figsize=(6, 6))
plt.pie(class_counts, labels=['Not depressed', 'Depressed'], autopct='%1.1f%%',␣
↪startangle=90)
plt.title('Class distribution')
plt.show()
```



[8]:
```
from wordcloud import WordCloud
from nltk.corpus import stopwords
import nltk
```

[9]:
```
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to C:\Users\MSI
[nltk_data]     PC\AppData\Roaming\nltk_data…
[nltk_data]   Package stopwords is already up-to-date!
```

[9]: True

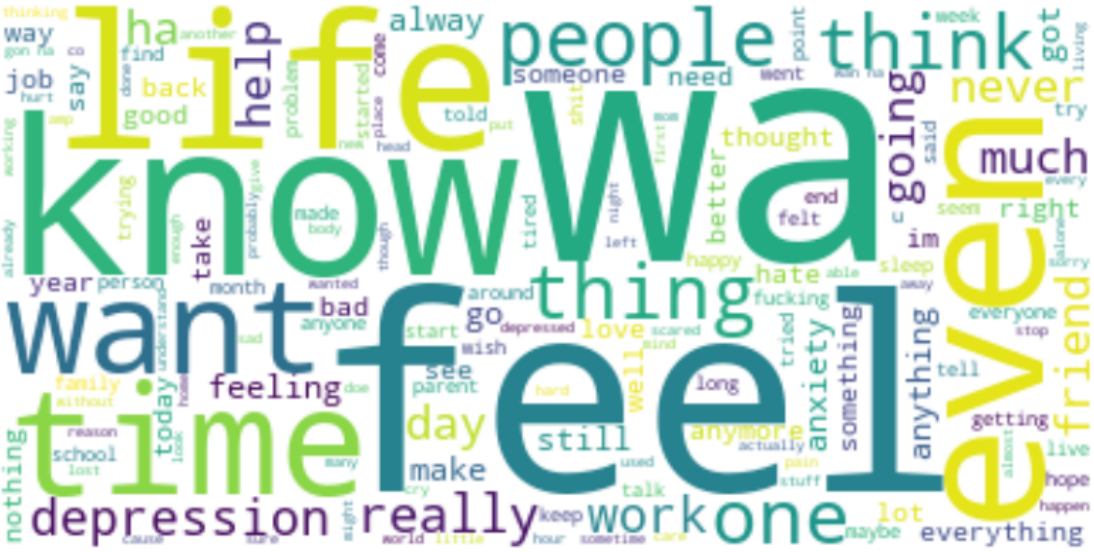
[10]:
```
stop_words = set(stopwords.words('english'))
```

[11]:
```
text = ' '.join(review for review in df['clean_text'])
```

[12]:
```
text = ' '.join([word for word in text.split() if word not in stop_words])
```

[13]:
```
wordcloud = WordCloud(background_color="white").generate(text)
```

[14]:
```
plt.figure(figsize=(10, 5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.show()
```



[15]:
```
from nltk.stem import PorterStemmer
from nltk.tokenize import word_tokenize
import nltk
```

[16]:
```
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to C:\Users\MSI
[nltk_data]     PC\AppData\Roaming\nltk_data…
[nltk_data]   Package punkt is already up-to-date!
```

[16]: True

[17]:
```
ps = PorterStemmer()
```

[18]:
```
def stem_sentences(sentence):
    tokens = word_tokenize(sentence)
    stemmed_tokens = [ps.stem(token) for token in tokens]
    return ' '.join(stemmed_tokens)
```

[19]:
```
df['stemmed_text'] = df['clean_text'].apply(stem_sentences)
```

[20]:
```
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize
```

[21]:
```
nltk.download('wordnet')
```

```
[nltk_data] Downloading package wordnet to C:\Users\MSI
[nltk_data]     PC\AppData\Roaming\nltk_data…
[nltk_data]   Package wordnet is already up-to-date!
```

[21]:
```
True
```

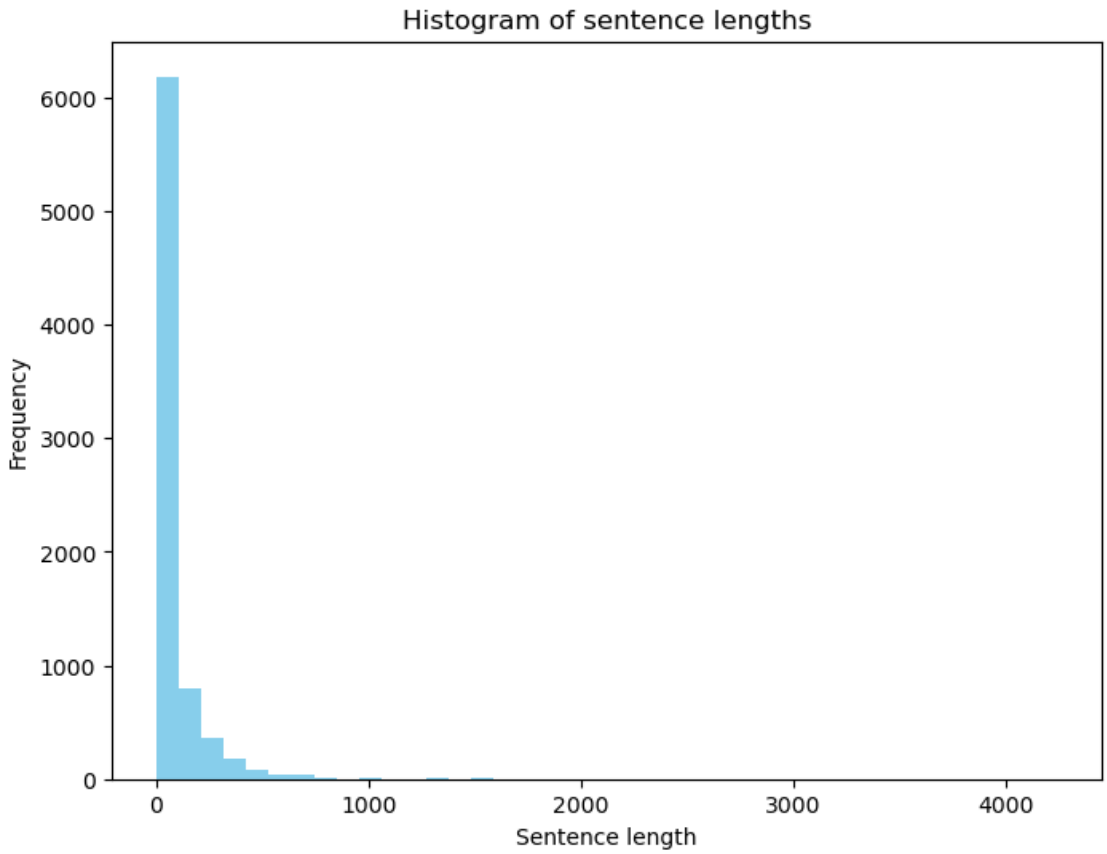
[22]:
```
lemmatizer = WordNetLemmatizer()
```

[23]:
```
def lemmatize_sentences(sentence):
    tokens = word_tokenize(sentence)
    lemmatized_tokens = [lemmatizer.lemmatize(token) for token in tokens]
    return ' '.join(lemmatized_tokens)
```

[24]:
```
df['lemmatized_text'] = df['clean_text'].apply(lemmatize_sentences)
```

[25]:
```
sentence_lengths = df['clean_text'].apply(lambda x: len(x.split()))
```

[29]:
```
plt.figure(figsize=(8, 6))
plt.hist(sentence_lengths, bins=40, color='skyblue')
plt.title('Histogram of sentence lengths')
plt.xlabel('Sentence length')
plt.ylabel('Frequency')
plt.show()
```



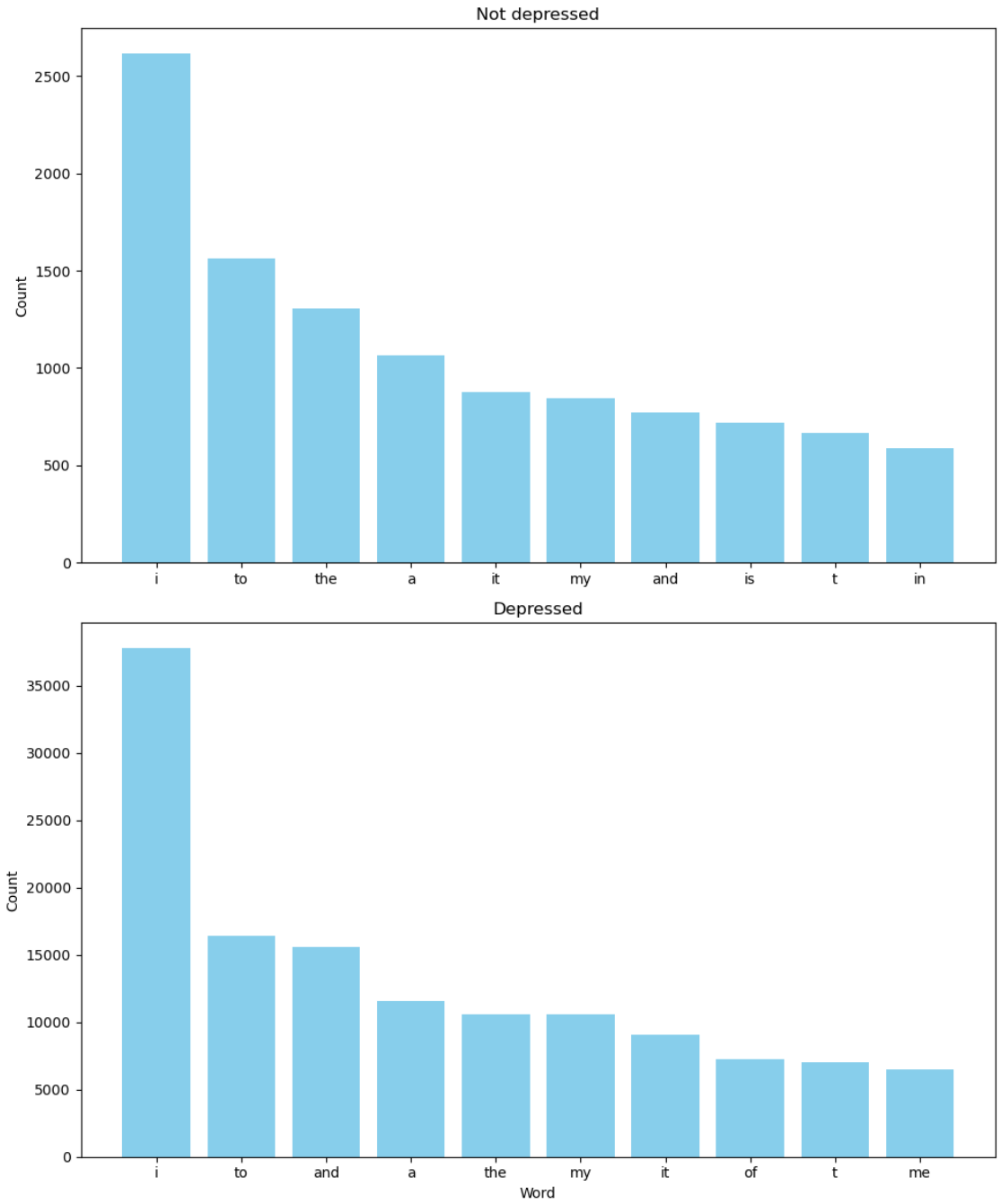
[30]:
```
from collections import Counter
import numpy as np
```

[31]:
```
not_depressed_text = ' '.join(df[df['is_depression'] == 0]['clean_text'])
depressed_text = ' '.join(df[df['is_depression'] == 1]['clean_text'])
not_depressed_words = Counter(not_depressed_text.split())
depressed_words = Counter(depressed_text.split())
```

[32]:
```
not_depressed_common = not_depressed_words.most_common(10)
depressed_common = depressed_words.most_common(10)
```

[33]:
```
fig, axs = plt.subplots(2, 1, figsize=(10, 12))
axs[0].bar(*zip(*not_depressed_common), color='skyblue')
axs[0].set_title('Not depressed')
axs[0].set_ylabel('Count')
axs[1].bar(*zip(*depressed_common), color='skyblue')
axs[1].set_title('Depressed')
axs[1].set_xlabel('Word')
axs[1].set_ylabel('Count')
plt.tight_layout()
```

```
plt.show()
```



[35]:
```
import seaborn as sns
```

[38]:
```
df['word_count'] = df['clean_text'].apply(lambda x: len(x.split()))
```

```
plt.figure(figsize=(10,6))
sns.boxplot(x='is_depression', y='word_count', data=df)
plt.title('Boxplot of Sentence Word Counts by Depression State')
plt.show()
```