In [1]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt


from gensim.models import Word2Vec
from sklearn.model_selection import train_test_split
from random import sample
import re

import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords

import warnings
warnings.filterwarnings("ignore")
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

In [2]:

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

In [3]:

```
# Import dataset
dataframe = pd.read_csv("/content/drive/MyDrive/ML Project/Reviews.csv", engine='python')
```

In [4]:

```
dataframe
```

Out[4]:

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400	Good Quality Dog Food
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1346976000	Not as Advertised
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	4	1219017600	"Delight" says it all
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	1307923200	Cough Medicine
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	5	1350777600	Great taffy
...	...	...	...	...	...	...	...	...	...
568449	568450	B001EO7N10	A28KG5XORO54AY	Lettie D. Carter	0	0	5	1299628800	Will not do without
568450	568451	B003S1WTCU	A3I8AFVPEE8KI5	R. Sawyer	0	0	2	1331251200	disappointed
568451	568452	B004I613EE	A121AA1GQV751Z	pksd "pk_007"	2	2	5	1329782400	Perfect for our maltipoo
568452	568453	B004I613EE	A3IBEVCTXKNOH	Kathy A. Welch "katwel"	1	1	5	1331596800	Favorite Training and reward treat
568453	568454	B001LR2CU2	A3LGQPJCZVL9UC	srfell17	0	0	5	1338422400	Great Honey

568454 rows × 10 columns

In [5]:

```
dataframe.shape
```

Out[5]:

```
(568454, 10)
```

In [6]:

```
# drop duplicate rows
data1 = dataframe.drop_duplicates(subset={"UserId","ProfileName","Time","Text"},keep='first')

# drop rows that do not meet the condition
data1 = data1[data1['HelpfulnessNumerator'] <= data1['HelpfulnessDenominator']]

data1.shape
```

Out[6]:

```
(393931, 10)
```

In [7]:

```
# count score values
data1['Score'].value_counts()
```

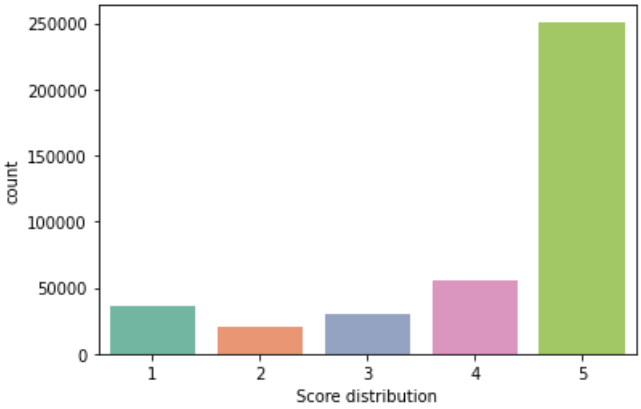
Out[7]:

```
5    250961
4     56093
1     36306
3     29769
2     20802
Name: Score, dtype: int64
```

In [8]:

```
#sns.countplot('Score',data = data1)
#plt.title("Score distribution")

import matplotlib.pyplot as plt
import seaborn as sns
plt.figure()
sns.countplot(x='Score', data=data1, palette='Set2')
plt.xlabel('Score distribution')
plt.show()
```



In [9]:

```
# Randomely select 20000 samples from each'Score' 1,2,4,5
S1 = data1[data1['Score'] ==1].sample(n=2500,random_state=0)
S2 = data1[data1['Score'] ==2].sample(n=2500,random_state=0)
S4 = data1[data1['Score'] ==4].sample(n=2500,random_state=0)
S5 = data1[data1['Score'] ==5].sample(n=2500,random_state=0)
data2 = pd.concat([S1,S2,S4,S5])
data2.shape
```

Out[9]:

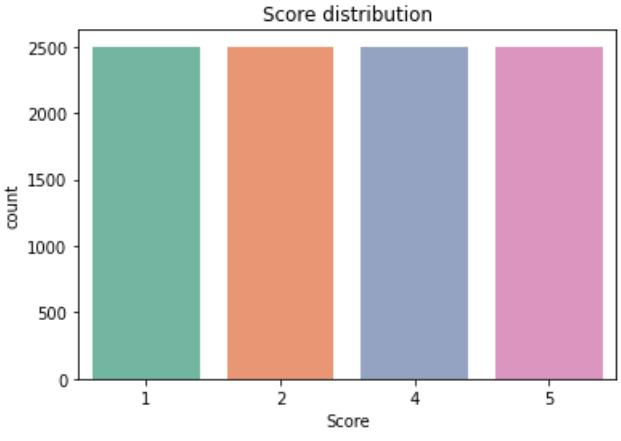
```
(10000, 10)
```

In [10]:

```
sns.countplot('Score',data = data2,palette='Set2')
plt.title("Score distribution")
```

Out[10]:

```
Text(0.5, 1.0, 'Score distribution')
```



In [11]:

```
# Identifing missing Values
miss_val = data2.isna().sum()
```

```
miss_val
```

Out[11]:

```
Id                         0
ProductId                  0
UserId                     0
ProfileName                1
HelpfulnessNumerator       0
HelpfulnessDenominator     0
Score                      0
Time                       0
Summary                    0
Text                       0
dtype: int64
```

In [12]:

```
# Converting Score values into class label either Posituve or Negative.
def partition(x):
    if x < 3:
        return 0
    else:
        return 1

data3 = data2.reset_index(drop=True)
actual_score = data3['Score']
label = actual_score.map(partition)
data3['Label']=label
```

In [15]:

```
# Modify score range from (1,5) to (1,4)
data4 = data3.iloc[:,:]
Score = [1]*2500 + [2]*2500 + [3]*2500 +[4]*2500
data4['Score'] = Score
data4
```

Out[15]:

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary
0	318929	B006OSAH28	A372R1Z11FUDZA	Michelle Harris	1	2	1	1337817600	Ripping people off
1	133502	B002YP8556	AG3KUNENZK0LB	fernwood "fern"	2	2	1	1297728000	Uselss product. May as well spray water.
2	99571	B000ILIHA6	A14UIV47XOOZU4	ElizVail	4	7	1	1245888000	Blue Dog Bakery Premium Peanut Butter and Mola...
3	519816	B000FZ0TCE	A1L5TJXLY1VIE9	Phyco126	1	2	1	1317772800	A rubber eraser tastes better...
4	384447	B000YPH8CE	AR3CFN3EBZU17	M. CHAN	0	0	1	1225756800	Too Hard
...	...	...	...	...	...	...	...	...	...
9995	150415	B001EQ596O	A1ADXPIDP2K1AD	auroramonroe "auroramonroe"	2	2	4	1224288000	Yummy!!!
9996	45474	B000LKVHOW	A3NT4W3WMTKXX6	Avid Reader	0	0	4	1326758400	Very convenient and tastes great
9997	13830	B004WJUXCK	A1OIWJVBR7INDX	Shari L. Swedlund	0	0	4	1331596800	canada wintergreen mints
9998	221333	B000B7PNI6	A21VGNU5959O85	Laura Terese Henri	4	4	4	1141344000	All Time Favorite Flavor.
9999	450704	B005TY2F3W	A29JAG9ML7C9DR	Karmala	1	1	4	1345161600	Yum!

10000 rows × 11 columns

In [16]:

```
import nltk
from nltk.corpus import stopwords
from wordcloud import WordCloud
import string
import matplotlib.pyplot as plt
```

In [17]:

```
def create_Word_Corpus(temp):
    words_corpus = ''
    for val in temp["Summary"]:
        text = str(val).lower()
        #text = text.translate(trantab)
        tokens = nltk.word_tokenize(text)
        tokens = [word for word in tokens if word not in stopwords.words('english')]
```

```
        for words in tokens:
            words_corpus = words_corpus + words + ' '
    return words_corpus
```

In [20]:

```
import nltk
nltk.download('punkt')

pos_wordcloud = WordCloud(width=900, height=500, background_color="white").generate(create_Word_Corpus(da
neg_wordcloud = WordCloud(width=900, height=500, background_color="white").generate(create_Word_Corpus(da
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

In [21]:

```
wordcloud_1 = WordCloud(width=900, height=500, background_color="white").generate(create_Word_Corpus(data
wordcloud_2 = WordCloud(width=900, height=500, background_color="white").generate(create_Word_Corpus(data
wordcloud_4 = WordCloud(width=900, height=500, background_color="white").generate(create_Word_Corpus(data
wordcloud_5 = WordCloud(width=900, height=500, background_color="white").generate(create_Word_Corpus(data
```

In [22]:

```
# Plot cloud
def plot_Cloud(wordCloud):
    plt.figure( figsize=(20,10), facecolor='white')
    plt.imshow(wordCloud)
    plt.axis("off")
    plt.tight_layout(pad=0)
    plt.show()
```

In [23]:

```
plot_Cloud(pos_wordcloud)
```

In [24]:

```
plot_Cloud(neg_wordcloud)
```

In [25]:

```
plot_Cloud(wordcloud_1) ####
```

In [26]:

```
plot_Cloud(wordcloud_2)
```

In [27]:

```
plot_Cloud(wordcloud_4)
```

In [28]:

```
plot_Cloud(wordcloud_5) ####
```

In [29]:

```
data5 = data4.iloc[:,:]

data5["Usefulness"] = (data5["HelpfulnessNumerator"]/data5["HelpfulnessDenominator"]).apply(lambda n: ">7
                                                                                                      else ("<25%" if
                                                                                                            else ("
                                                                                                                  e


data5.loc[data5.HelpfulnessDenominator == 0, 'Usefulness'] = "useless"
data5
```

Out[29]:

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary
0	318929	B006OSAH28	A372R1Z11FUDZA	Michelle Harris	1	2	1	1337817600	Ripping people off
1	133502	B002YP8556	AG3KUNENZK0LB	fernwood "fern"	2	2	1	1297728000	Uselss product. May as well spray water.
2	99571	B000ILIHA6	A14UIV47XOOZU4	ElizVail	4	7	1	1245888000	Blue Dog Bakery Premium Peanut Butter and Mola...
3	519816	B000FZ0TCE	A1L5TJXLY1VIE9	Phyco126	1	2	1	1317772800	A rubber eraser tastes better...
4	384447	B000YPH8CE	AR3CFN3EBZU17	M. CHAN	0	0	1	1225756800	Too Hard
...	...	...	...	...	...	...	...	...	...
9995	150415	B001EQ596O	A1ADXPIDP2K1AD	auroramonroe "auroramonroe"	2	2	4	1224288000	Yummy!!!
9996	45474	B000LKVHOW	A3NT4W3WMTKXX6	Avid Reader	0	0	4	1326758400	Very convenient and tastes great
9997	13830	B004WJUXCK	A1OIWJVBR7INDX	Shari L. Swedlund	0	0	4	1331596800	canada wintergreen mints
9998	221333	B000B7PNI6	A21VGNU5959O85	Laura Terese Henri	4	4	4	1141344000	All Time Favorite Flavor.
9999	450704	B005TY2F3W	A29JAG9ML7C9DR	Karmala	1	1	4	1345161600	Yum!

10000 rows × 12 columns

In [30]:

```
# save the dataframe as a csv file
data5.to_csv("data5.csv")
```

In [31]:

```
data5.Usefulness.value_counts()
```

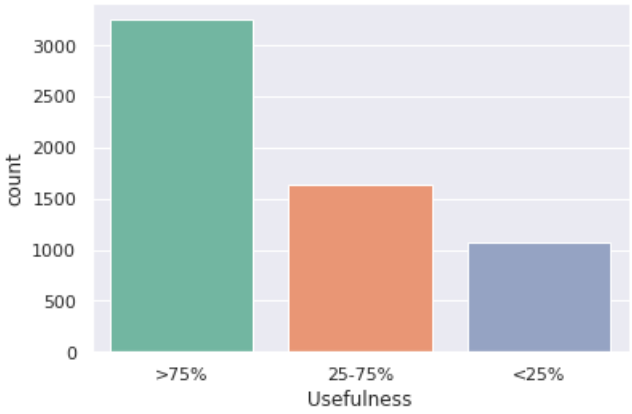
Out[31]:

```
useless    4046
>75%       3251
25-75%     1636
<25%       1067
Name: Usefulness, dtype: int64
```

In [32]:

```
# useful reviews are commom
```

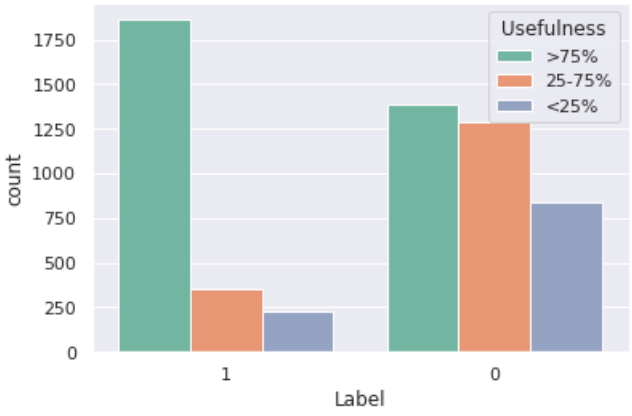
```
sns.set()
sns.countplot(x='Usefulness', order=[ '>75%', '25-75%', '<25%'], data=data5, palette='Set2')
plt.xlabel('Usefulness')
plt.show()
```



In [33]:

```
# nagetive reviews are more helpful

sns.set()
sns.countplot(x='Label', hue='Usefulness', order=[1, 0], \
              hue_order=['>75%', '25-75%', '<25%'], data=data5, palette="Set2")
plt.xlabel('Label')
plt.show()
```



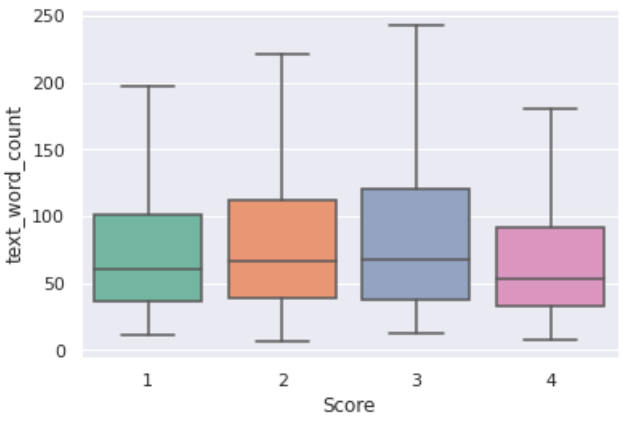
In [34]:

```
# Word Count
data5["text_word_count"] = data5["Text"].apply(lambda text: len(text.split()))
```

In [35]:

```
# Positive reviews (Score 4) are shorter.

sns.boxplot(x='Score',y='text_word_count', data=data5
, palette='Set2', showfliers=False)
plt.show()
```



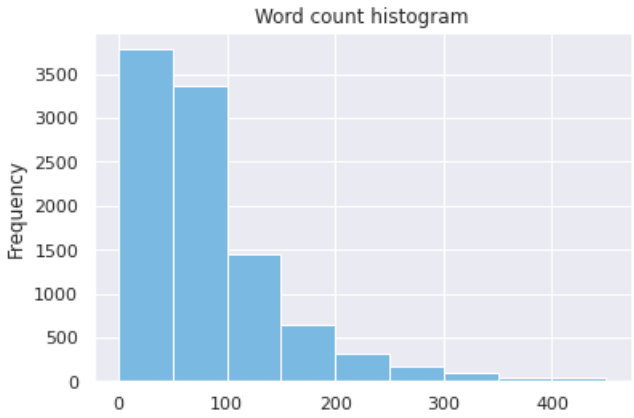
In [36]:

```
#Word count histogram
bins_list = [50*x for x in range(0,10)]
```

```
data5.Text.apply(lambda x:len(x.split(' '))).plot(kind='hist',bins = bins_list,color='#7AB9E1')
plt.title('Word count histogram')
```

Out[36]:

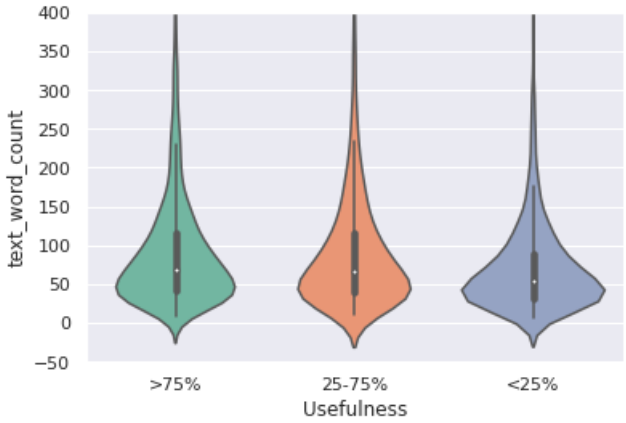
```
Text(0.5, 1.0, 'Word count histogram')
```



In [37]:

```
# How does word count relate to helpfulness?
sns.violinplot(x='Usefulness', y='text_word_count', order=['>75%', '25-75%', '<25%'], \
               data=data5, palette='Set2')
plt.ylim(-50, 400)
plt.show()

#
```



In [37]:

In [ ]: