#Exploratory Data Analysis on Book Summary

In [48]:
```
import pandas as pd
import numpy as np
from bs4 import BeautifulSoup as bs
import requests

import string
from collections import Counter
import seaborn as sns
import matplotlib.pyplot as plt
from nltk.corpus import stopwords
from nltk.sentiment.vader import SentimentIntensityAnalyzer
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize
import nltk
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')
nltk.download('averaged_perceptron_tagger')
```

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

/usr/local/lib/python3.7/dist-packages/nltk/twitter/__init__.py:20: UserWarning: The twython library has not b
een installed. Some functionality from the twitter package will not be available.
  warnings.warn("The twython library has not been installed. "
```

Out[48]: True

In [ ]:
```
books_data = pd.read_csv('/content/book_summaries.csv', encoding = 'latin1', error_bad_lines=False)
books_data.head()
```

```
/usr/local/lib/python3.7/dist-packages/IPython/core/interactiveshell.py:2882: FutureWarning: The error_bad_lin
es argument has been deprecated and will be removed in a future version.


  exec(code_obj, self.user_global_ns, self.user_ns)
```

Out[4]:

	title	category	summary
0	Steve Jobs	biography	Walter Isaacson's "enthralling" (The New Yorke...
1	John Adams	biography	The enthralling, often surprising story of Joh...
2	Becoming	biography	In a life filled with meaning and accomplishme...
3	The Shining	horror	Jack Torrance's new job at the Overlook Hotel ...
4	Carrie	horror	The story of misunderstood high school girl C...

In [ ]:
```
summary = [x.strip() for x in books_data.summary]
summary[1]
```

Out[5]:
```
'The enthralling, often surprising story of John Adams, one of the most important and fascinating Americans wh
o ever lived.In this powerful, epic biography, David McCullough unfolds the adventurous life-journey of John A
dams, the brilliant, fiercely independent, often irascible, always honest Yankee patriot -- "the colossus of i
ndependence," as Thomas Jefferson called him - The enthralling, often surprising story of John Adams, one of t
he most important and fascinating Americans who ever lived.In this powerful, epic biography, David McCullough
unfolds the adventurous life-journey of John Adams, the brilliant, fiercely independent, often irascible, alwa
ys honest Yankee patriot -- "the colossus of independence," as Thomas Jefferson called him -- who spared nothi
ng in his zeal for the American Revolution; who rose to become the second President of the United States and s
aved the country from blundering into an unnecessary war; who was learned beyond all but a few and regarded by
some as "out of his senses"; and whose marriage to the wise and valiant Abigail Adams is one of the moving lov
e stories in American history. Like his masterly, Pulitzer Prize-winning biography Truman, David McCullough\'s
John Adams has the sweep and vitality of a great novel. It is both a riveting portrait of an abundantly human
man and a vivid evocation of his time, much of it drawn from an outstanding collection of Adams family letters
and diaries. In particular, the more than one thousand surviving letters between John and Abigail Adams, nearl
y half of which have never been published, provide extraordinary access to their private lives and make it pos
sible to know John Adams as no other major American of his founding era. As he has with stunning effect in his
previous books, McCullough tells the story from within -- from the point of view of the amazing eighteenth cen
tury and of those who, caught up in events, had no sure way of knowing how things would turn out. George Washi
ngton, Benjamin Franklin, John Jay, the British spy Edward Bancroft, Madame Lafayette and Jefferson\'s Paris
"interest" Maria Cosway, Alexander Hamilton, James Madison, the scandalmonger James Callender, Sally Hemings,
John Marshall, Talleyrand, and Aaron Burr all figure in this panoramic chronicle, as does, importantly, John Q
uincy Adams, the adored son whom Adams would live to see become President. Crucial to the story, as it was to
history, is the relationship between Adams and Jefferson, born opposites -- one a Massachusetts farmer\'s son,
the other a Virginia aristocrat and slaveholder, one short and stout, the other tall and spare. Adams embraced
conflict; Jefferson avoided it. Adams had great humor; Jefferson, very little. But they were alike in their de
votion to their country. At first they were ardent co-revolutionaries, then fellow diplomats and close friend
s. With the advent of the two political parties, they became archrivals, even enemies, in the intense struggle
for the presidency in 1800, perhaps the most vicious election in history. Then, amazingly, they became friends
again, and ultimately, incredibly, they died on the same day -- their day of days -- July 4, in the year 1826.
Much about John Adams\'s life will come as a surprise to many readers. His courageous voyage on the frigate Bo
ston in the winter of 1778 and his later trek over the Pyrenees are exploits that few would have dared and tha
t few readers will ever forget. It is a life encompassing a huge arc -- Adams lived longer than any president.
The story ranges from the Boston Massacre to Philadelphia in 1776 to the Versailles of Louis XVI, from Spain t
o Amsterdam, from the Court of St. James\'s, where Adams was the first American to stand before King George II
I as a representative of the new nation, to the raw, half-finished Capital by the Potomac, where Adams was the
first President to occupy the White House. This is history on a grand scale -- a book about politics and war a
nd social issues, but also about human nature, love, religious faith, virtue, ambition, friendship and betraya
l, and the far-reaching consequences of noble ideas. Above all, John Adams is an enthralling, often surprising
story of one of the most important and fascinating Americans who ever lived.'
```

Data pre-processing

In [ ]:
```
from nltk.stem import WordNetLemmatizer
from nltk.corpus import stopwords
wordnet = WordNetLemmatizer()
import re

filtered_sum=[]


filtered_sent=[]
for i in range(len(summary)):
    summary_ = re.sub("[^A-Za-z" "]+"," ",summary[i])
    summary_ = re.sub("[0-9" "]+"," ",summary[i])

    summary_ = summary_.lower()
    summary_ =summary_.split()
    summary_ = [wordnet.lemmatize(word) for word in summary_ if not word in set(stopwords.words('english'))]
    summary_ = ' '.join(summary_)
    filtered_sum.append(summary_)
```

In [ ]:
```


with open("negative-words.txt","r", encoding='latin-1') as neg:
    negwords = neg.read().split("\n")
```

In [ ]:
```
with open("positive-words.txt","r") as pos:
    poswords = pos.read().split("\n")
```

Vectorization

In [ ]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
tf = TfidfVectorizer()
text_tf = tf.fit_transform(filtered_sum)
feature_names = tf.get_feature_names()
dense = text_tf.todense()
denselist = dense.tolist()
summary_df =pd.DataFrame(denselist, columns=feature_names)
summary_df.head()
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function get_feature_na
mes is deprecated; get_feature_names is deprecated in 1.0 and will be removed in 1.2. Please use get_feature_n
ames_out instead.
  warnings.warn(msg, category=FutureWarning)
```

Out[9]:

	aaron	abigail	ability	able	abundantly	academy	access	accomplishment	ache	active	...	written	xvi	yankee
0	0.047486	0.000000	0.0	0.0	0.000000	0.0	0.000000	0.000000	0.0	0.000000	...	0.142458	0.000000	0.000000
1	0.028399	0.066136	0.0	0.0	0.033068	0.0	0.033068	0.000000	0.0	0.000000	...	0.000000	0.033068	0.066136
2	0.000000	0.000000	0.0	0.0	0.000000	0.0	0.000000	0.135475	0.0	0.067737	...	0.000000	0.000000	0.000000
3	0.000000	0.000000	0.0	0.0	0.000000	0.0	0.000000	0.000000	0.0	0.000000	...	0.000000	0.000000	0.000000
4	0.000000	0.000000	0.0	0.0	0.000000	0.0	0.000000	0.000000	0.0	0.000000	...	0.000000	0.000000	0.000000

5 rows × 923 columns

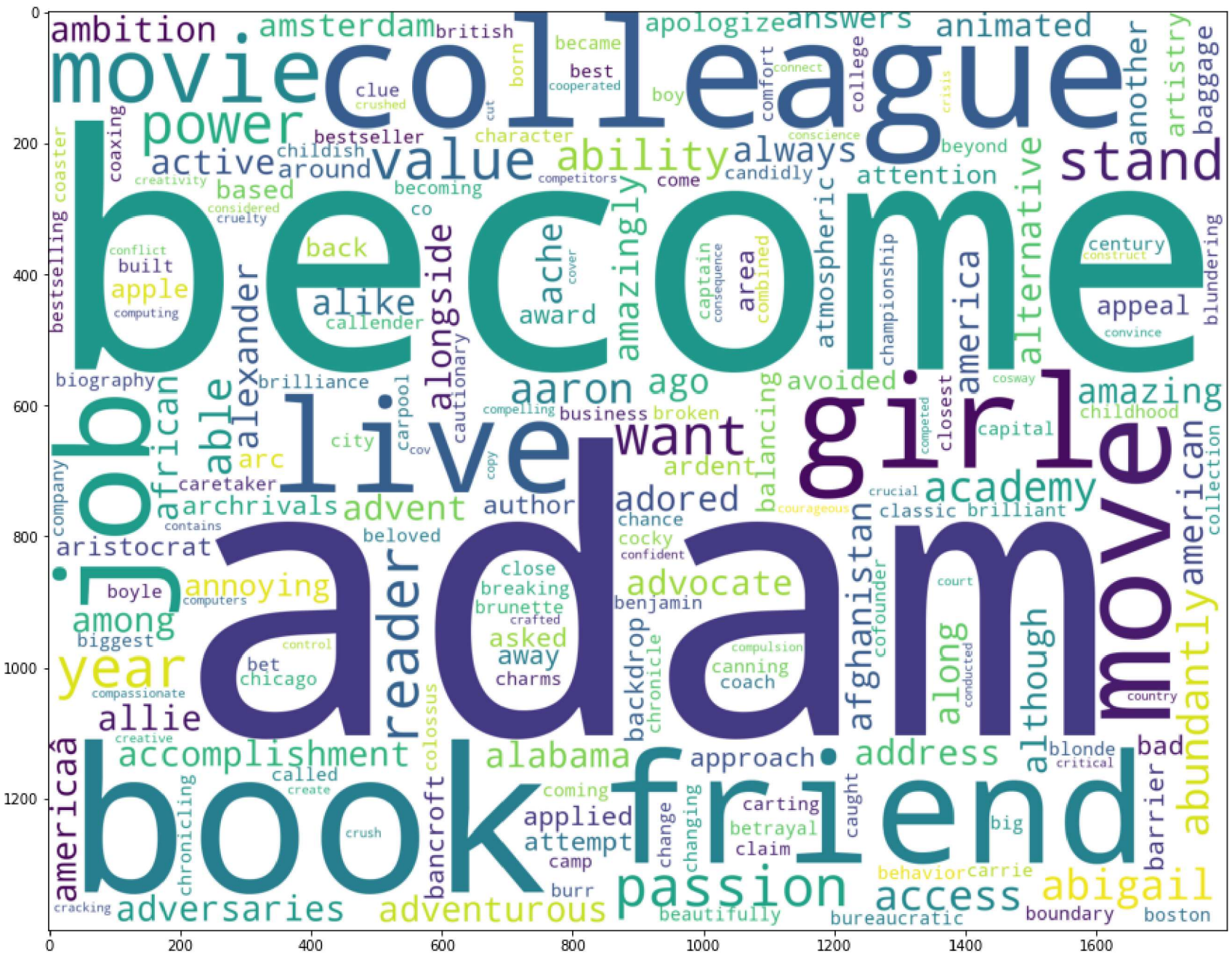
Exploratory Data Analysis

Wordcloud

In [ ]:
```
#plotting wordcloud on TFIDF
from wordcloud import WordCloud
import matplotlib.pyplot as plt
cloud = ' '.join(summary_df)
```

In [ ]:
```
f, axes = plt.subplots(figsize=(20,12))
wordcloud= WordCloud(
        background_color = 'white',
        width = 1800,
        height =1400).generate(cloud)
plt.imshow(wordcloud)
```

Out[11]: <matplotlib.figure.AxesImage at 0x7fe450928d50>



Positive words

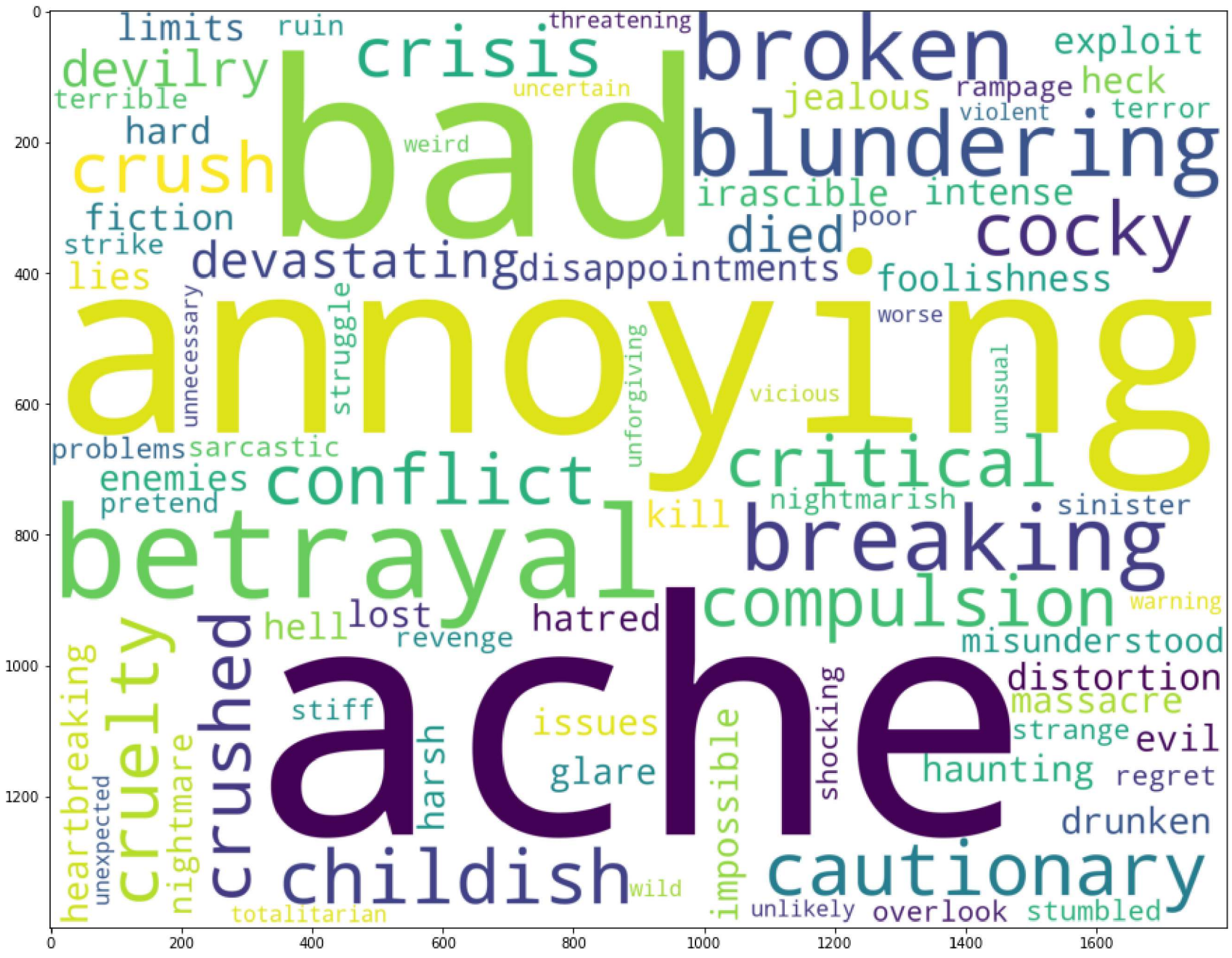
In [ ]:
```
f, axes = plt.subplots(figsize=(20,12))
pos_words = ' '.join([w for w in summary_df if w in poswords])

cloud_pos = WordCloud(
        background_color = 'white',
        width =1800,
        height=1400).generate(pos_words)
plt.imshow(cloud_pos)
```

Out[12]: <matplotlib.image.AxesImage at 0x7fe44eb813d0>

negative Words

In [ ]:
```
f, axes = plt.subplots(figsize=(20,12))
neg_words = ' '.join([w for w in summary_df if w in negwords])

cloud_neg = WordCloud(
        background_color='white',
        width =1800,
        height =1400).generate(neg_words)
plt.imshow(cloud_neg)
```

Out[13]: <matplotlib.figure.AxesImage at 0x7fe44eb10c50>



word cloud Category wise

In [ ]:
```
summary_df
```

Out[14]:

	aaron	abigail	ability	able	abundantly	academy	access	accomplishment	ache	active	...	written	xvi
0	0.047486	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.142458	0.000000
1	0.028399	0.066136	0.000000	0.000000	0.033068	0.000000	0.033068	0.000000	0.000000	0.000000	...	0.000000	0.033068
2	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.135475	0.000000	0.067737	...	0.000000	0.000000
3	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000
4	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000
5	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000
6	0.000000	0.000000	0.000000	0.000000	0.000000	0.157236	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000
7	0.000000	0.000000	0.077526	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.066580	0.000000
8	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000
9	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000
10	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000
11	0.000000	0.000000	0.000000	0.127459	0.000000	0.000000	0.000000	0.000000	0.063729	0.000000	...	0.000000	0.000000

12 rows × 923 columns

In [ ]:
```
#biography
#plotting wordcloud on TFIDF
from wordcloud import WordCloud
import matplotlib.pyplot as plt
cloud = ' '.join(summary_df[0:3])

f, axes = plt.subplots(figsize=(20,12))
wordcloud= WordCloud(
        background_color = 'white',
        width = 1800,
        height =1400).generate(cloud)
plt.imshow(wordcloud)
```

Out[15]: <matplotlib.image.AxesImage at 0x7fe44ea9df50>

In [ ]:
```
books_data["filtered_summary"] = filtered_sum
books_data
```

Out[16]:

	title	category	summary	filtered_summary
0	Steve Jobs	biography	Walter Isaacson's "enthralling" (The New Yorke...	walter isaacson's "enthralling" (the new yorke...
1	John Adams	biography	The enthralling, often surprising story of Joh...	enthralling, often surprising story john adams...
2	Becoming	biography	In a life filled with meaning and accomplishme...	life filled meaning accomplishment, michelle o...
3	The Shining	horror	Jack Torrance's new job at the Overlook Hotel ...	jack torrance's new job overlook hotel perfect...
4	Carrie	horror	The story of misunderstood high school girl C...	story misunderstood high school girl carrie wh...
5	It	horror	Welcome to Derry, Maine ...Itâ s a small city...	welcome derry, maine ...itâ s small city, pla...
6	To kill a mockingbird	fiction	The unforgettable novel of a childhood in a sl...	unforgettable novel childhood sleepy southern ...
7	1984	fiction	Among the seminal texts of the 20th century, N...	among seminal text th century, nineteen eighty...
8	The Kite Runner	fiction	The unforgettable, heartbreaking story of the ...	unforgettable, heartbreaking story unlikely fr...
9	The deal	sports	An alternative cover edition can be found here...	alternative cover edition found here.she's mak...
10	The Score	sports	He knows how to score, on and off the iceAllie...	know score, iceallie hayes crisis mode. gradua...
11	Him	sports	They donâ t play for the same team. Or do the...	donâ t play team. they?jamie canning never ab...

Parts of Speech distribution analysis : Categorizing and POS tagging the words

In [ ]:
```
nltk.download('universal_tagset')
def get_pos_tags(sentences, tagset='universal'):
    ''' Extract the part-of-speech taggings of the sentence
        Input:
        - sentence: string, sentence to tag
        - tagset: string, tagset or the set of tags to search for
    '''
    #Create the Dataframe to store the count of tags
    df = pd.DataFrame(columns=['ADJ','ADP','ADV','CONJ','DET','NOUN','NUM','PRT','PRON','VERB','.','X'])
    for sent in sentences:
        # Extract the part of Speech tags in the sentence
        pos_tags = Counter([j for i,j in nltk.pos_tag(word_tokenize(sent), tagset='universal')])
        #Appends the pos tags to the dataframe, fill NaN values with 0
        df = df.append(pos_tags, ignore_index=True).fillna(0)

    return df.astype(int)


sum_text = books_data.filtered_summary.values
df_text = get_pos_tags(sum_text)
```

```
[nltk_data] Downloading package universal_tagset to /root/nltk_data...
[nltk_data]   Unzipping taggers/universal_tagset.zip.
```

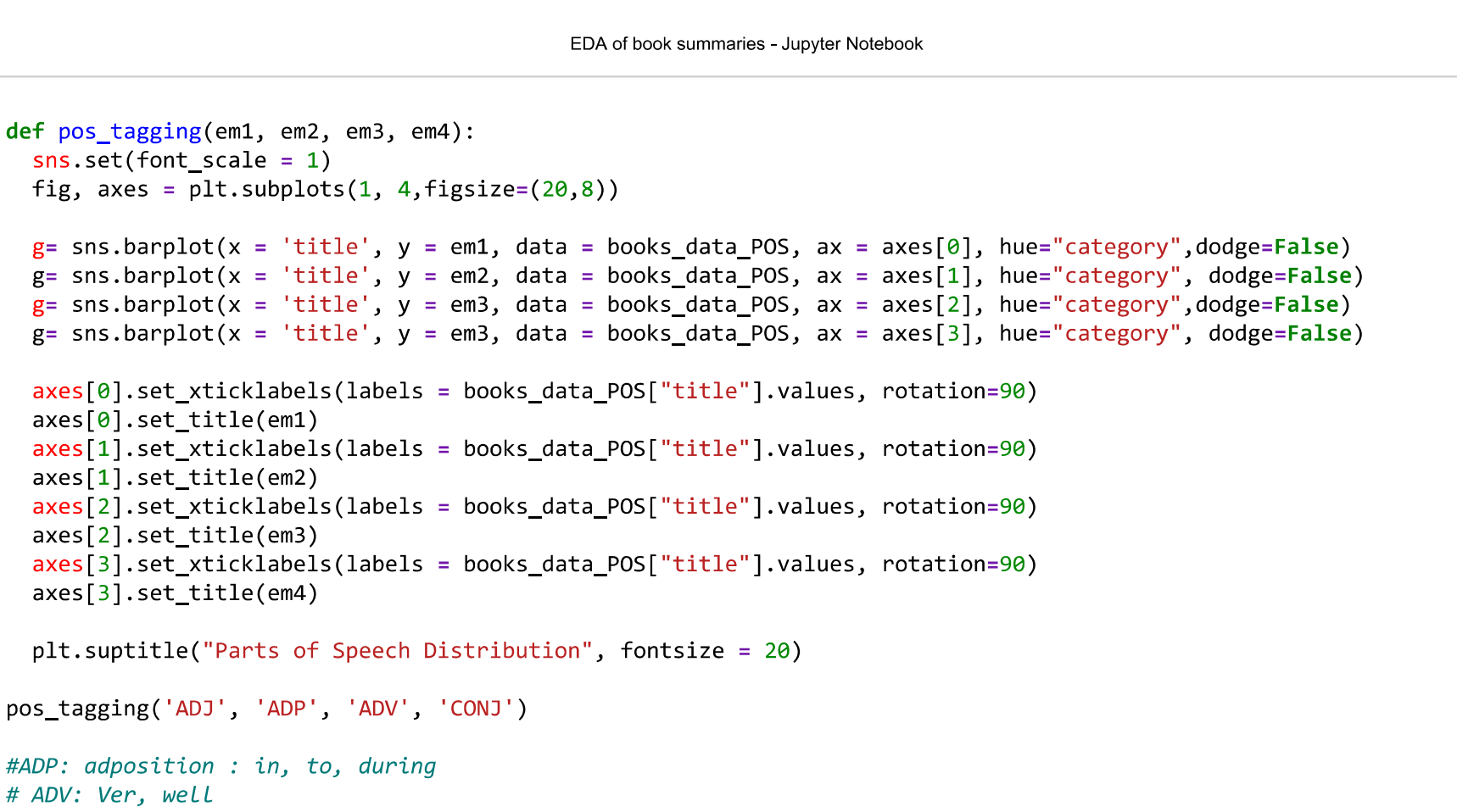
In [ ]:
```
books_data_POS = pd.concat([books_data, df_text], axis=1)
books_data_POS.head(2)
```

Out[18]:

	title	category	summary	filtered_summary	ADJ	ADP	ADV	CONJ	DET	NOUN	NUM	PRT	PRON	VERB	.	X
0	Steve Jobs	biography	Walter Isaacson's "enthralling" (The New Yorke...	walter isaacson's "enthralling" (the new yorke...	31	5	7	0	2	116	4	3	1	37	57	0
1	John Adams	biography	The enthralling, often surprising story of Joh...	enthralling, often surprising story john adams...	89	6	39	0	4	195	8	7	3	52	117	2

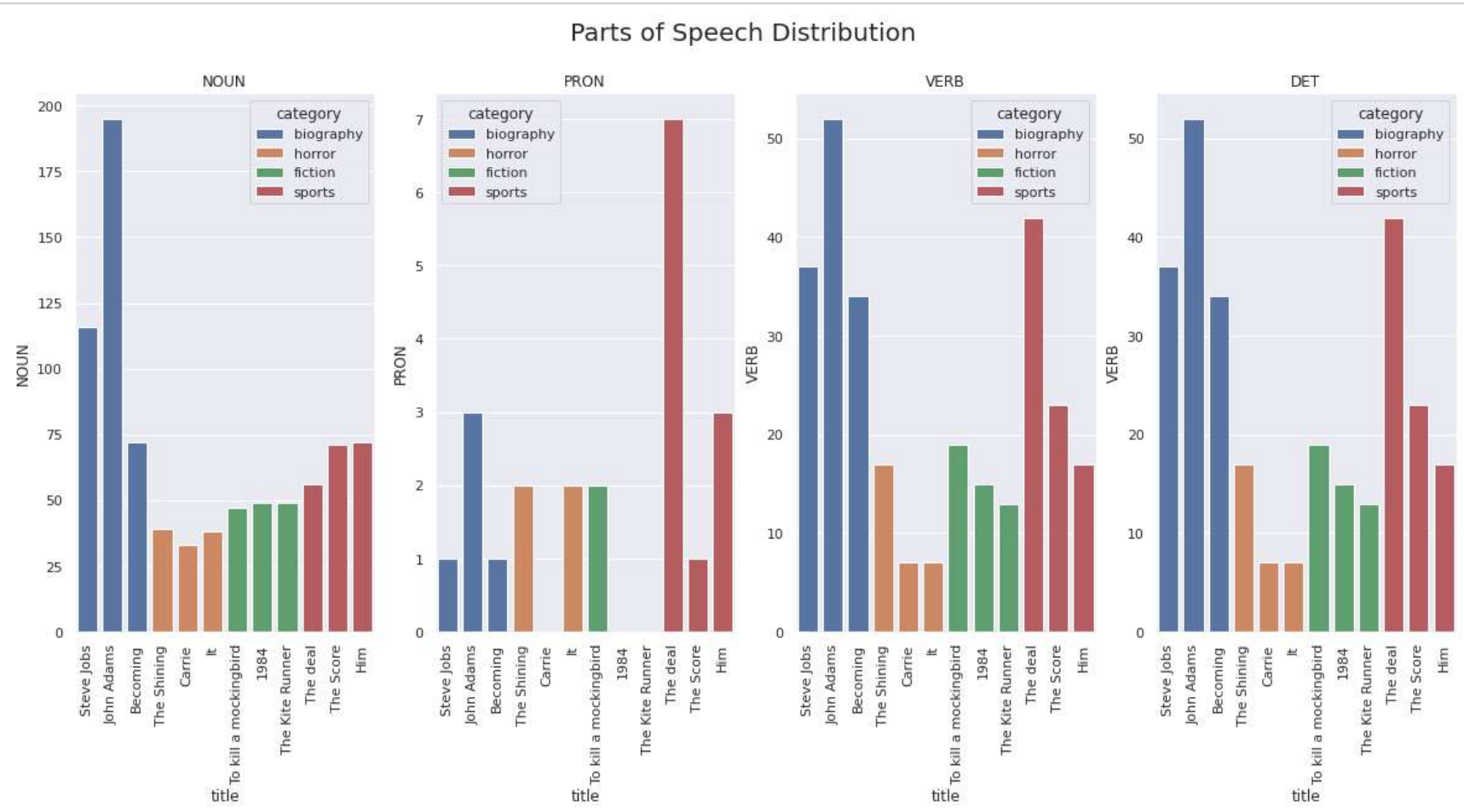
In [ ]:

```
def pos_tagging(em1, em2, em3, em4):
  sns.set(font_scale = 1)
  fig, axes = plt.subplots(1, 4,figsize=(20,8))

  g= sns.barplot(x = 'title', y = em1, data = books_data_POS, ax = axes[0], hue="category",dodge=False)
  g= sns.barplot(x = 'title', y = em2, data = books_data_POS, ax = axes[1], hue="category", dodge=False)
  g= sns.barplot(x = 'title', y = em3, data = books_data_POS, ax = axes[2], hue="category",dodge=False)
  g= sns.barplot(x = 'title', y = em3, data = books_data_POS, ax = axes[3], hue="category", dodge=False)

  axes[0].set_xticklabels(labels = books_data_POS["title"].values, rotation=90)
  axes[0].set_title(em1)
  axes[1].set_xticklabels(labels = books_data_POS["title"].values, rotation=90)
  axes[1].set_title(em2)
  axes[2].set_xticklabels(labels = books_data_POS["title"].values, rotation=90)
  axes[2].set_title(em3)
  axes[3].set_xticklabels(labels = books_data_POS["title"].values, rotation=90)
  axes[3].set_title(em4)

  plt.suptitle("Parts of Speech Distribution", fontsize = 20)

pos_tagging('ADJ', 'ADP', 'ADV', 'CONJ')

#ADP: adposition : in, to, during
# ADV: Ver, well
# CONJ: And, or, but
```



In [ ]:

```
pos_tagging('NOUN', 'PRON', 'VERB', 'DET')
```



Determiner (DET)

Determiners are words that modify nouns or noun phrases and express the reference of the noun phrase in context.

possessive determiners (which modify a nominal): [cs] můj, tvůj, jeho, její, náš, váš, jejich; [en] my, your
demonstrative determiners: this as in I saw this car yesterday.
interrogative determiners: which as in “Which car do you like?”
relative determiners: which as in “I wonder which car you like.”
quantity determiners (quantifiers): indefinite any, universal: all, and negative no as in “We have no cars available.”

#Unique words

In [ ]:
```
from textblob import TextBlob
books_data['textblob_sentiment'] = books_data['filtered_summary'].apply(lambda x: TextBlob(x).sentiment[0])
books_data['Unique Terms'] = books_data['filtered_summary'].str.split().explode().drop_duplicates().groupby(leve
```

In [ ]:
```
books_data[["title","category","Unique Terms"]]
```

Out[22]:

	title	category	Unique Terms
0	Steve Jobs	biography	[walter, isaacson's, "enthralling", (the, new,...
1	John Adams	biography	[enthralling,, often, surprising, john, adams,...
2	Becoming	biography	[meaning, accomplishment,, michelle, obama, em...
3	The Shining	horror	[jack, torrance's, overlook, hotel, perfect, c...
4	Carrie	horror	[misunderstood, high, school, carrie, white,, ...
5	It	horror	[welcome, derry,, maine, ...itâ s, small, cit...
6	To kill a mockingbird	fiction	[unforgettable, sleepy, southern, town, crisis...
7	1984	fiction	[among, seminal, text, th, century,, nineteen,...
8	The Kite Runner	fiction	[unforgettable,, heartbreaking, unlikely, weal...
9	The deal	sports	[alternative, cover, edition, found, here.she'...
10	The Score	sports	[score,, iceallie, hayes, mode., graduation, l...
11	Him	sports	[donâ t, team., they?jamie, canning, able, lo...

N- Grams

In [ ]:
```
import collections
import re
import sys
import time


def listandtokenize(data):
  yourlist = data.tolist()
  string = ' '.join(map(str, yourlist))
  return re.findall(r'w+', string.lower())


# function to prepare n-grams
def count_ngrams(lines, min_length=2, max_length=4):
    lengths = range(min_length, max_length+1)
    ngrams = {length: collections.Counter() for length in lengths}
    queue = collections.deque(maxlen = max_length)
    def add_queue():
        current = tuple(queue)
        for length in lengths:
            if len(current)>= length:
                ngrams[length][current[:length]] +=1
    for line in lines:
        for word in nltk.word_tokenize(line):
            queue.append(word)
            if len(queue) >= max_length:
                add_queue()
    while len(queue) > min_length:
        queue.popleft()
        add_queue()
    return ngrams


# function to print 15 most frequent n-grams
# change the print number as applicable
def print_most_freq_ng(ngrams, num=15):
    for n in sorted(ngrams):
        print('----{} most frequent {}-grams ----'.format(num, n))
        for gram, count in ngrams[n].most_common(num):
            print('{0}: {1}'.format(' '.join(gram), count))
        print('')
```

In [ ]:
```
print_most_freq_ng(count_ngrams(summary_df))
```

```
----15 most frequent 2-grams ----
aaron abigail: 1
abigail ability: 1
ability able: 1
able abundantly: 1
abundantly academy: 1
academy access: 1
access accomplishment: 1
accomplishment ache: 1
ache active: 1
active adam: 1
adam adams: 1
adams address: 1
address adored: 1
adored advent: 1
advent adventurous: 1

----15 most frequent 3-grams ----
aaron abigail ability: 1
abigail ability able: 1
ability able abundantly: 1
able abundantly academy: 1
abundantly academy access: 1
academy access accomplishment: 1
access accomplishment ache: 1
accomplishment ache active: 1
ache active adam: 1
active adam adams: 1
adam adams address: 1
adams address adored: 1
address adored advent: 1
adored advent adventurous: 1
advent adventurous adversaries: 1

----15 most frequent 4-grams ----
aaron abigail ability able: 1
abigail ability able abundantly: 1
ability able abundantly academy: 1
able abundantly academy access: 1
abundantly academy access accomplishment: 1
academy access accomplishment ache: 1
access accomplishment ache active: 1
accomplishment ache active adam: 1
ache active adam adams: 1
active adam adams address: 1
adam adams address adored: 1
adams address adored advent: 1
address adored advent adventurous: 1
adored advent adventurous adversaries: 1
advent adventurous adversaries advocate: 1
```

Calculating Subjectivity and polarity

Subjectivity is nothing but a sentence that expresses some personal feelings, views, or beliefs. Its values range from 0 to 1 where 0 is very objective and 1 is very subjective,

while **polarity** simply means emotions expressed in a sentence. Its value ranges from -1 to 1, where -1 represents the most negative comment and 1 represent the most positive comment

In [ ]:
```
from textblob import TextBlob

def calc_subj(sum_):
    return TextBlob(sum_).sentiment.subjectivity

# function for Polarity
def calc_pola(sum_):
    return TextBlob(sum_).sentiment.polarity

books_data['Subjectivity'] = books_data.filtered_summary.apply(calc_subj)
books_data['Polarity'] = books_data.filtered_summary.apply(calc_pola)
books_data
```

Out[25]:

	title	category	summary	filtered_summary	textblob_sentiment	Unique Terms	Subjectivity	Polarity
0	Steve Jobs	biography	Walter Isaacson's "enthralling" (The New Yorke...	walter isaacson's "enthralling" (the new yorke...	0.395170	[walter, isaacson's, "enthralling", (the, new,...	0.741193	0.395170
1	John Adams	biography	The enthralling, often surprising story of Joh...	enthralling, often surprising story john adams...	0.266564	[enthralling,, often, surprising, john, adams,...	0.556901	0.266564
2	Becoming	biography	In a life filled with meaning and accomplishme...	life filled meaning accomplishment, michelle o...	0.202050	[meaning, accomplishment,, michelle, obama, em...	0.441176	0.202050
3	The Shining	horror	Jack Torrance's new job at the Overlook Hotel ...	jack torrance's new job overlook hotel perfect...	0.066043	[jack, torrance's, overlook, hotel, perfect, c...	0.579947	0.066043
4	Carrie	horror	The story of misunderstood high school girl C...	story misunderstood high school girl carrie wh...	-0.114848	[misunderstood, high, school, carrie, white,, ...	0.562576	-0.114848
5	It	horror	Welcome to Derry, Maine ...Itâ s a small city...	welcome derry, maine ...itâ s small city, pla...	0.197368	[welcome, derry,, maine, ...itâ s, small, cit...	0.340351	0.197368
6	To kill a mockingbird	fiction	The unforgettable novel of a childhood in a sl...	unforgettable novel childhood sleepy southern ...	0.174359	[unforgettable, sleepy, southern, town, crisis...	0.386813	0.174359
7	1984	fiction	Among the seminal texts of the 20th century, N...	among seminal text th century, nineteen eighty...	-0.140659	[among, seminal, text, th, century,, nineteen,...	0.538462	-0.140659
8	The Kite Runner	fiction	The unforgettable, heartbreaking story of the ...	unforgettable, heartbreaking story unlikely fr...	0.291667	[unforgettable,, heartbreaking, unlikely, weal...	0.777083	0.291667
9	The deal	sports	An alternative cover edition can be found here...	alternative cover edition found here.she's mak...	-0.047454	[alternative, cover, edition, found, here.she'...	0.712731	-0.047454
10	The Score	sports	He knows how to score, on and off the iceAllie...	know score, iceallie hayes crisis mode. gradua...	0.084127	[score,, iceallie, hayes, mode., graduation, l...	0.476720	0.084127
11	Him	sports	They donâ t play for the same team. Or do the...	donâ t play team. they?jamie canning never ab...	-0.067593	[donâ t, team., they?jamie, canning, able, lo...	0.512963	-0.067593

Affinity score finds out the semantic relations between the words

In [ ]:
```
with open("afinn2.txt","r") as affin:
    affinity = affin.read().split("\n")
```

In [ ]:
```
affinity_data = pd.read_csv('afinn2.txt', sep="\t", header=None, names=["word", "value"])
affinity_data.head()
```

Out[27]:

	word	value
0	abandon	-2
1	abandoned	-2
2	abandons	-2
3	abducted	-2
4	abduction	-2

In [ ]:
```
affinity_scores = affinity_data.set_index('word')['value'].to_dict()
#affinity_scores
```

In [ ]:
```
#pip install spacy
import spacy
nlp = spacy.load("en_core_web_sm")
sentiment_lexicon = affinity_scores
```

In [ ]:
```
def calculate_sentiment(text: str = None):
    sent_score = 0
    if text:
        sentence = nlp(text)
        for word in sentence:
            sent_score += sentiment_lexicon.get(word.lemma_, 0)
    return sent_score
```

In [ ]:
```
books_data['sentiment_value'] = books_data['filtered_summary'].apply(calculate_sentiment)
books_data
```

Out[31]:

	title	category	summary	filtered_summary	textblob_sentiment	Unique Terms	Subjectivity	Polarity	sentiment_value
0	Steve Jobs	biography	Walter Isaacson's "enthralling" (The New Yorke...	walter isaacson's "enthralling" (the new yorke...	0.395170	[walter, isaacson's, "enthralling", (the, new,...	0.741193	0.395170	23
1	John Adams	biography	The enthralling, often surprising story of Joh...	enthralling, often surprising story john adams...	0.266564	[enthralling,, often, surprising, john, adams,...	0.556901	0.266564	78
2	Becoming	biography	In a life filled with meaning and accomplishme...	life filled meaning accomplishment, michelle o...	0.202050	[meaning, accomplishment,, michelle, obama, em...	0.441176	0.202050	23
3	The Shining	horror	Jack Torrance's new job at the Overlook Hotel ...	jack torrance's new job overlook hotel perfect...	0.066043	[jack, torrance's, overlook, hotel, perfect, c...	0.579947	0.066043	6
4	Carrie	horror	The story of misunderstood high school girl C...	story misunderstood high school girl carrie wh...	-0.114848	[misunderstood, high, school, carrie, white,, ...	0.562576	-0.114848	-24
5	It	horror	Welcome to Derry, Maine ...Itâ s a small city...	welcome derry, maine ...itâ s small city, pla...	0.197368	[welcome, derry,, maine, ...itâ s, small, cit...	0.340351	0.197368	19
6	To kill a mockingbird	fiction	The unforgettable novel of a childhood in a sl...	unforgettable novel childhood sleepy southern ...	0.174359	[unforgettable, sleepy, southern, town, crisis...	0.386813	0.174359	24
7	1984	fiction	Among the seminal texts of the 20th century, N...	among seminal text th century, nineteen eighty...	-0.140659	[among, seminal, text, th, century,, nineteen,...	0.538462	-0.140659	-4
8	The Kite Runner	fiction	The unforgettable, heartbreaking story of the ...	unforgettable, heartbreaking story unlikely fr...	0.291667	[unforgettable,, heartbreaking, unlikely, weal...	0.777083	0.291667	17
9	The deal	sports	An alternative cover edition can be found here...	alternative cover edition found here.she's mak...	-0.047454	[alternative, cover, edition, found, here.she'...	0.712731	-0.047454	-3
10	The Score	sports	He knows how to score, on and off the iceAllie...	know score, iceallie hayes crisis mode. gradua...	0.084127	[score,, iceallie, hayes, mode., graduation, l...	0.476720	0.084127	-1
11	Him	sports	They donâ t play for the same team. Or do the...	donâ t play team. they?jamie canning never ab...	-0.067593	[donâ t, team., they?jamie, canning, able, lo...	0.512963	-0.067593	-7

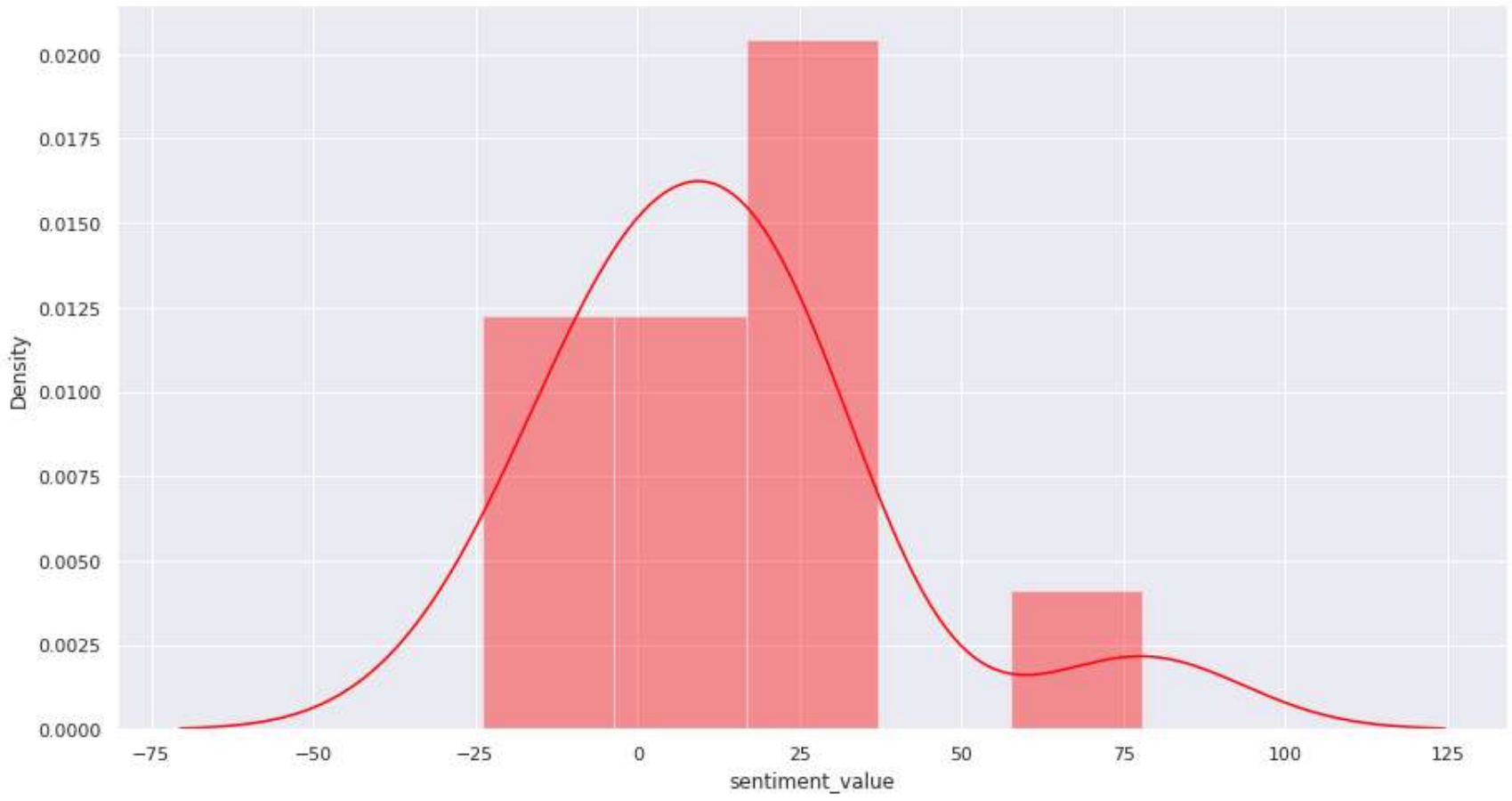
In [ ]:
```
# how many words are in the sentence?
books_data['word_count'] = books_data['filtered_summary'].str.split().apply(len)
books_data
```

Out[32]:

	title	category	summary	filtered_summary	textblob_sentiment	Unique Terms	Subjectivity	Polarity	sentiment_value
0	Steve Jobs	biography	Walter Isaacson's "enthralling" (The New Yorke...	walter isaacson's "enthralling" (the new yorke...	0.395170	[walter, isaacson's, "enthralling", (the, new,...	0.741193	0.395170	23
1	John Adams	biography	The enthralling, often surprising story of Joh...	enthralling, often surprising story john adams...	0.266564	[enthralling,, often, surprising, john, adams,...	0.556901	0.266564	78
2	Becoming	biography	In a life filled with meaning and accomplishme...	life filled meaning accomplishment, michelle o...	0.202050	[meaning, accomplishment,, michelle, obama, em...	0.441176	0.202050	23
3	The Shining	horror	Jack Torrance's new job at the Overlook Hotel ...	jack torrance's new job overlook hotel perfect...	0.066043	[jack, torrance's, overlook, hotel, perfect, c...	0.579947	0.066043	6
4	Carrie	horror	The story of misunderstood high school girl C...	story misunderstood high school girl carrie wh...	-0.114848	[misunderstood, high, school, carrie, white,, ...	0.562576	-0.114848	-24
5	It	horror	Welcome to Derry, Maine ...Itâ s a small city...	welcome derry, maine ...itâ s small city, pla...	0.197368	[welcome, derry,, maine, ...itâ s, small, cit...	0.340351	0.197368	19
6	To kill a mockingbird	fiction	The unforgettable novel of a childhood in a sl...	unforgettable novel childhood sleepy southern ...	0.174359	[unforgettable, sleepy, southern, town, crisis...	0.386813	0.174359	24
7	1984	fiction	Among the seminal texts of the 20th century, N...	among seminal text th century, nineteen eighty...	-0.140659	[among, seminal, text, th, century,, nineteen,...	0.538462	-0.140659	-4
8	The Kite Runner	fiction	The unforgettable, heartbreaking story of the ...	unforgettable, heartbreaking story unlikely fr...	0.291667	[unforgettable,, heartbreaking, unlikely, weal...	0.777083	0.291667	17
9	The deal	sports	An alternative cover edition can be found here...	alternative cover edition found here.she's mak...	-0.047454	[alternative, cover, edition, found, here.she'...	0.712731	-0.047454	-3
10	The Score	sports	He knows how to score, on and off the iceAllie...	know score, iceallie hayes crisis mode. gradua...	0.084127	[score,, iceallie, hayes, mode., graduation, l...	0.476720	0.084127	-1
11	Him	sports	They donâ t play for the same team. Or do the...	donâ t play team. they?jamie canning never ab...	-0.067593	[donâ t, team., they?jamie, canning, able, lo...	0.512963	-0.067593	-7

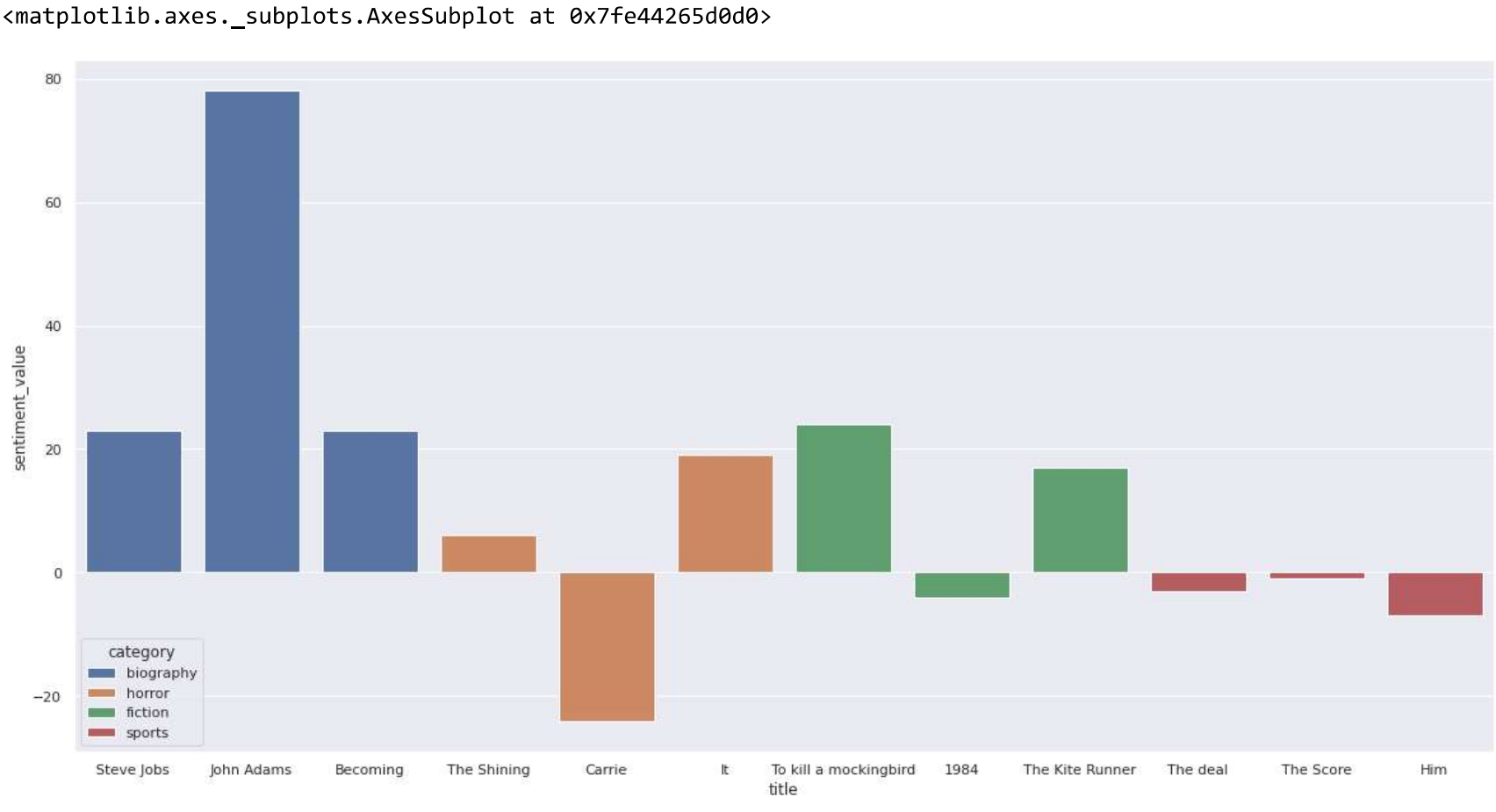
In [ ]:
```
f, axes = plt.subplots(figsize= (15,8))

sns.distplot(books_data['sentiment_value'],color = "red")
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecate
d function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-l
evel function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



In [ ]:
```
plt.figure(figsize=(20, 10))
sns.barplot(y='sentiment_value',x='title',data=books_data, hue="category", dodge=False)
```

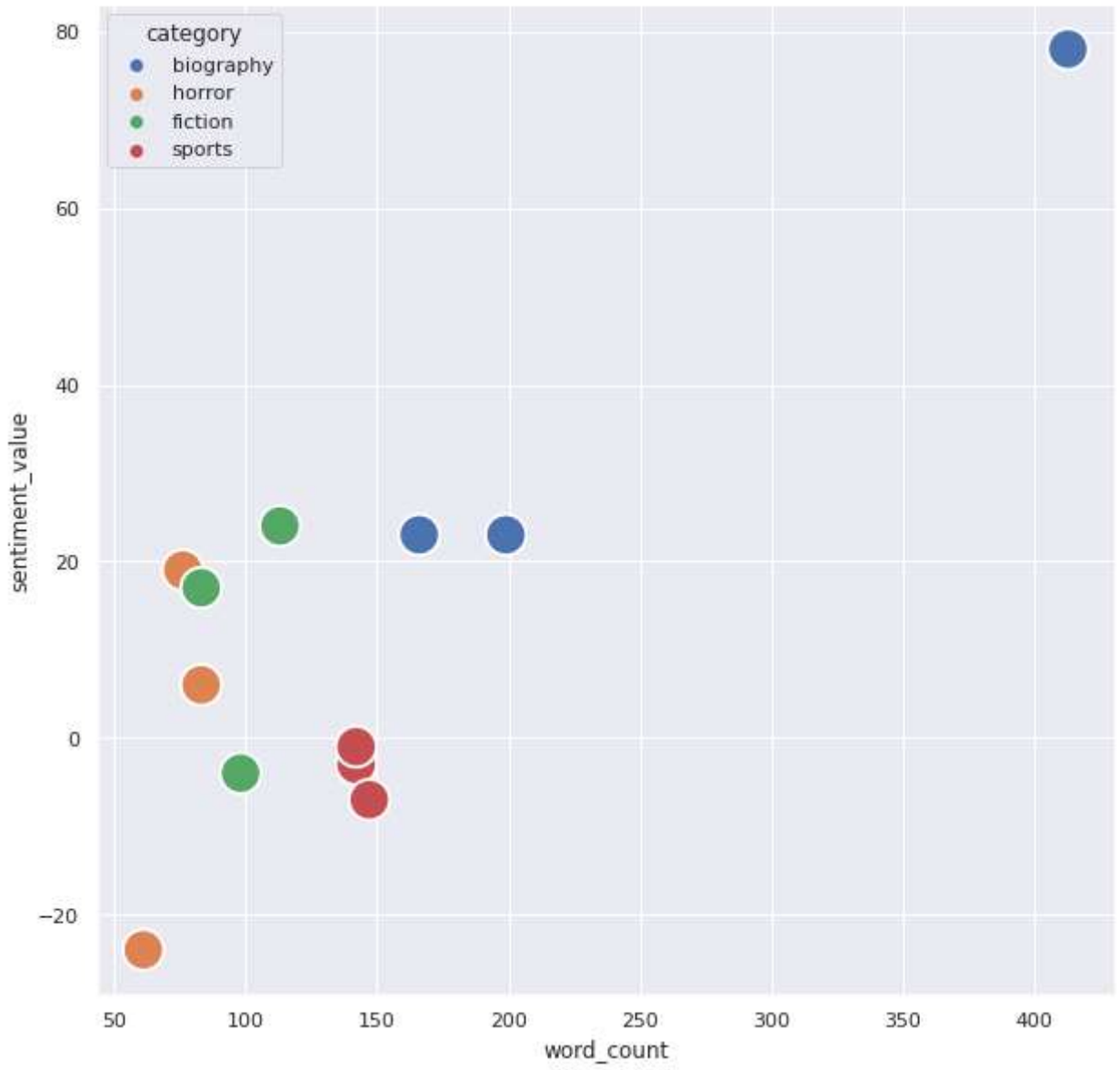
Out[34]: <matplotlib.axes._subplots.AxesSubplot at 0x7fe44265d0d0>



Sentiment value to Word count

In [ ]:
```
plt.figure(figsize=(10, 10))
sns.scatterplot(data=books_data, x="word_count", y="sentiment_value", hue="category", s=500)
```

Out[35]: <matplotlib.axes._subplots.AxesSubplot at 0x7fe44488ae10>

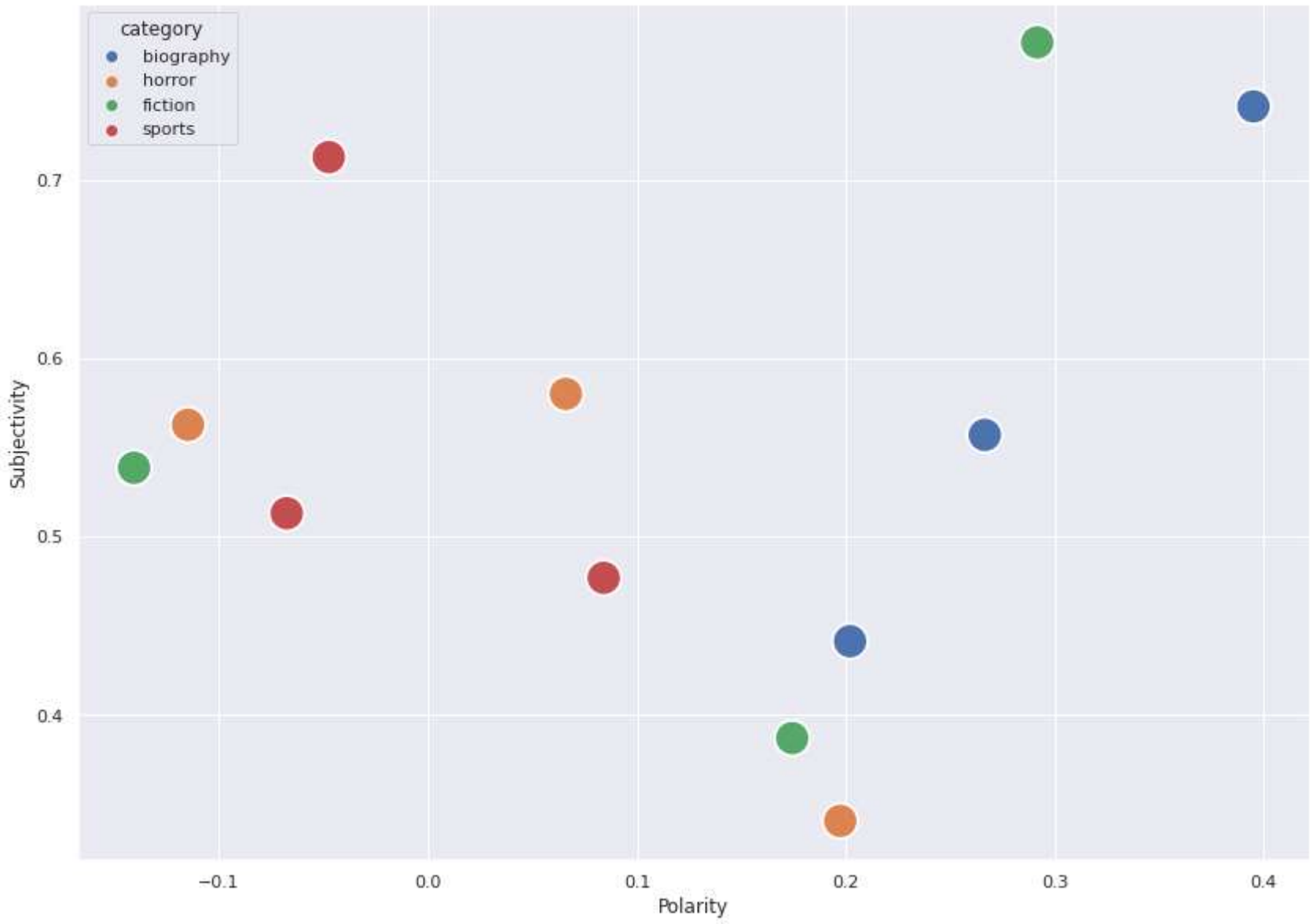


Subjectivity and Polarity Analysis

In [ ]:
```
from plotly.offline import iplot
import cufflinks as cf
cf.go_offline()
cf.set_config_file(offline=False, world_readable=True)
import plotly.io as pio
pio.renderers.default = "colab"
```

In [ ]:
```
plt.figure(figsize=(14, 10))
sns.scatterplot(data=books_data, x="Polarity", y="Subjectivity", hue="category", s=500)
```

Out[37]: <matplotlib.axes._subplots.AxesSubplot at 0x7fe44169dfd0>



Raw emotion scores

In [ ]:
```
!pip install NRCLex
import nltk
from nrclex import NRCLex
```

```
Collecting NRCLex
  Downloading NRCLex-3.0.0.tar.gz (396 kB)
     |████████████████████████████████| 396 kB 24.2 MB/s eta 0:00:01
Requirement already satisfied: textblob in /usr/local/lib/python3.7/dist-packages (from NRCLex) (0.15.3)
Requirement already satisfied: nltk>=3.1 in /usr/local/lib/python3.7/dist-packages (from textblob->NRCLex) (3.
2.5)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from nltk>=3.1->textblob->NRCLe
x) (1.15.0)
Building wheels for collected packages: NRCLex
  Building wheel for NRCLex (setup.py) ... done
  Created wheel for NRCLex: filename=NRCLex-3.0.0-py3-none-any.whl size=43329 sha256=f05ed6c74cc90245d0e1aeb52
430365a6ae6351c40a506f5ee807620caa2a51f
  Stored in directory: /root/.cache/pip/wheels/af/2c/9c/dfa19d1b65326c520b32850a9311f6d4eda679ac04dba26081
Successfully built NRCLex
Installing collected packages: NRCLex
Successfully installed NRCLex-3.0.0
```

In [ ]:
```
emotion_ = NRCLex(books_data[4:5].filtered_summary[4])
```

In [ ]:
```
emotion_.raw_emotion_scores
```

Out[40]:
```
{'anger': 6,
 'anticipation': 4,
 'disgust': 4,
 'fear': 12,
 'joy': 2,
 'negative': 12,
 'positive': 8,
 'surprise': 4,
 'trust': 6}
```

In [ ]:
```
anger=[];disgust=[];fear=[];joy=[];surprise=[];trust=[];anticipation=[];sadness=[];positive=[];negative=[]
emotions= ["anger","disgust","fear","joy","surprise","trust","anticipation","sadness","positive","negative"]


for i,j in enumerate(books_data.filtered_summary.values,0):
  #print(i)
  emotion = NRCLex(j)

  if "positive" in emotion.raw_emotion_scores.keys():
    positive.append(emotion.raw_emotion_scores['positive'])
  else:
    positive.append(0)


  if "anger" in emotion.raw_emotion_scores.keys():
    anger.append(emotion.raw_emotion_scores['anger'])
  else:
    anger.append(0)

  if "disgust" in emotion.raw_emotion_scores.keys():
    disgust.append(emotion.raw_emotion_scores['disgust'])
  else:
    disgust.append(0)

  if "fear" in emotion.raw_emotion_scores.keys():
    fear.append(emotion.raw_emotion_scores['fear'])
  else:
    fear.append(0)

if "joy" in emotion.raw_emotion_scores.keys():
  joy.append(emotion.raw_emotion_scores['joy'])
else:
  joy.append(0)

if "surprise" in emotion.raw_emotion_scores.keys():
  surprise.append(emotion.raw_emotion_scores['surprise'])
else:
  surprise.append(0)

if "trust" in emotion.raw_emotion_scores.keys():
  trust.append(emotion.raw_emotion_scores['trust'])
else:
  trust.append(0)

if "anticipation" in emotion.raw_emotion_scores.keys():
  anticipation.append(emotion.raw_emotion_scores['anticipation'])
else:
  anticipation.append(0)

if "sadness" in emotion.raw_emotion_scores.keys():
  sadness.append(emotion.raw_emotion_scores['sadness'])
else:
  sadness.append(0)

if "negative" in emotion.raw_emotion_scores.keys():
  negative.append(emotion.raw_emotion_scores['negative'])
else:
  negative.append(0)
```

In [ ]:
```
emotions_df = pd.DataFrame(list(zip(anger, anticipation, disgust, fear, joy, negative,positive, sadness, surpris
                columns =['anger', 'anticipation', 'disgust', 'fear', 'joy', 'negative',
       'positive', 'sadness', 'surprise', 'trust'])
emotions_df
```

Out[42]:

	anger	anticipation	disgust	fear	joy	negative	positive	sadness	surprise	trust
0	3	7	2	3	9	5	23	2	4	7
1	13	20	20	15	25	23	59	7	11	33
2	6	8	3	5	12	8	30	2	1	16
3	4	6	3	4	2	4	9	1	2	4
4	6	4	4	12	2	12	8	0	4	6
5	5	8	3	8	6	8	12	3	2	4
6	2	4	2	5	8	8	15	5	1	2
7	1	4	3	3	0	6	11	1	0	2
8	6	3	10	5	6	13	11	6	0	9
9	5	15	8	5	17	13	26	2	7	22
10	5	12	1	9	11	13	15	5	7	6
11	1	5	5	1	7	11	10	6	2	13

In [ ]:
```
books_data_EDA = pd.concat([books_data, emotions_df], axis=1)
books_data_EDA.head()
```

Out[43]:

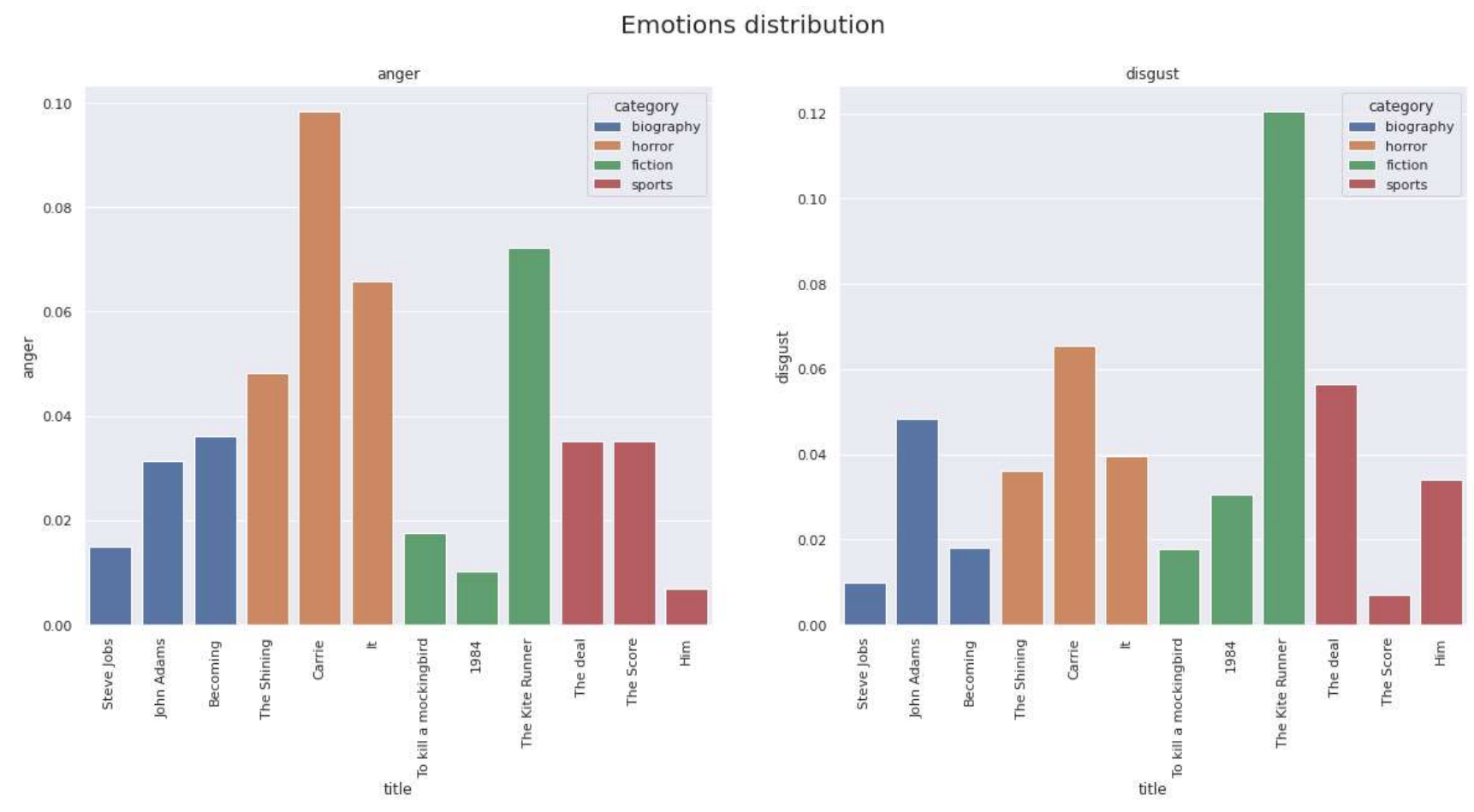
	title	category	summary	filtered_summary	textblob_sentiment	Unique Terms	Subjectivity	Polarity	sentiment_value	w
0	Steve Jobs	biography	Walter Isaacson's "enthralling" (The New Yorke...	walter isaacson's "enthralling" (the new yorke...	0.395170	[walter, isaacson's, "enthralling", (the, new,...	0.741193	0.395170	23	
1	John Adams	biography	The enthralling, often surprising story of Joh...	enthralling, often surprising story john adams...	0.266564	[enthralling,, often, surprising, john, adams,...	0.556901	0.266564	78	
2	Becoming	biography	In a life filled with meaning and accomplishme...	life filled meaning accomplishment, michelle o...	0.202050	[meaning, accomplishment,, michelle, obama, em...	0.441176	0.202050	23	
3	The Shining	horror	Jack Torrance's new job at the Overlook Hotel ...	jack torrance's new job overlook hotel perfect...	0.066043	[jack, torrance's, overlook, hotel, perfect, c...	0.579947	0.066043	6	
4	Carrie	horror	The story of misunderstood high school girl C...	story misunderstood high school girl carrie wh...	-0.114848	[misunderstood, high, school, carrie, white,, ...	0.562576	-0.114848	-24	

In [ ]:
```
for emotion in emotions:
    books_data_EDA[emotion] = books_data_EDA[emotion] / books_data_EDA['word_count']
```

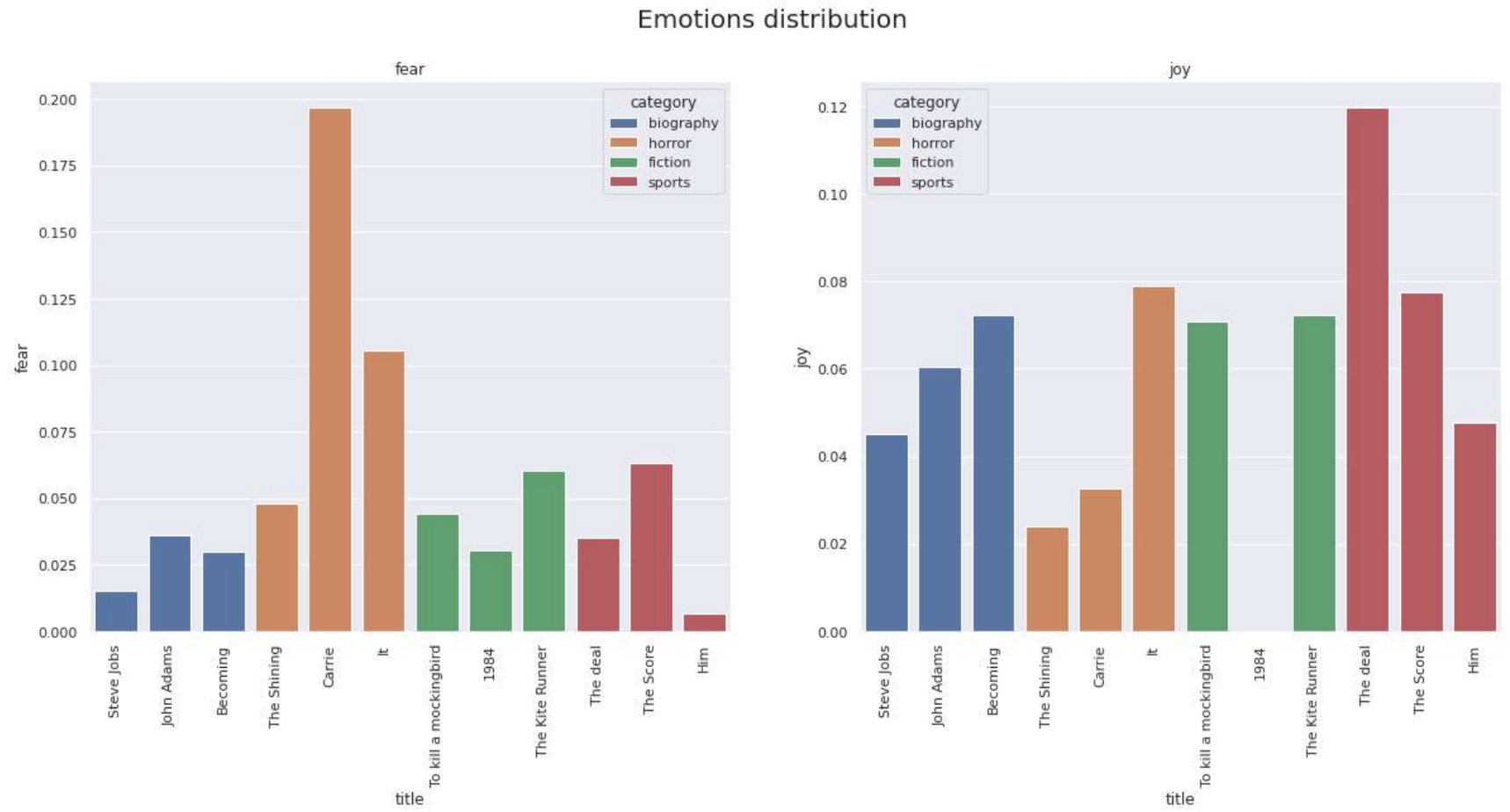
In [ ]:
```
emotions
```

Out[45]:
```
['anger',
 'disgust',
 'fear',
 'joy',
 'surprise',
 'trust',
 'anticipation',
 'sadness',
 'positive',
 'negative']
```

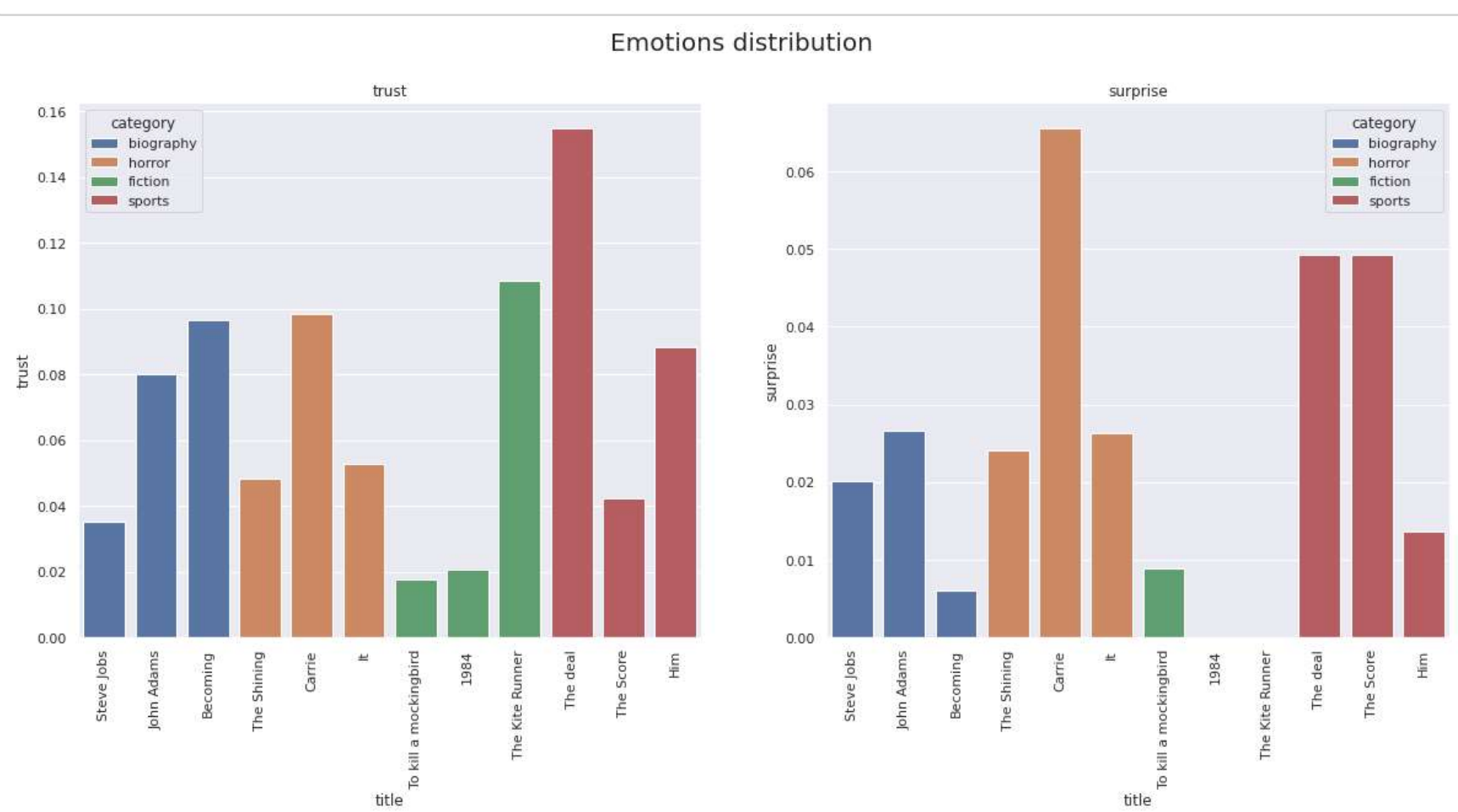
In [ ]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns


def emotions_mapping(em1, em2):
  sns.set(font_scale = 1)
  fig, axes = plt.subplots(1, 2,figsize=(20,8))

  g= sns.barplot(x = 'title', y = em1,data = books_data_EDA, ax = axes[0], hue="category",dodge=False)
  g= sns.barplot(x = 'title', y = em2,data = books_data_EDA, ax = axes[1], hue="category", dodge=False)
  axes[0].set_xticklabels(labels = books_data_EDA["title"].values, rotation=90)
  axes[0].set_title(em1)
  axes[1].set_xticklabels(labels = books_data_EDA["title"].values, rotation=90)
  axes[1].set_title(em2)

  plt.suptitle("Emotions distribution", fontsize = 20)

emotions_mapping('anger', 'disgust')
```



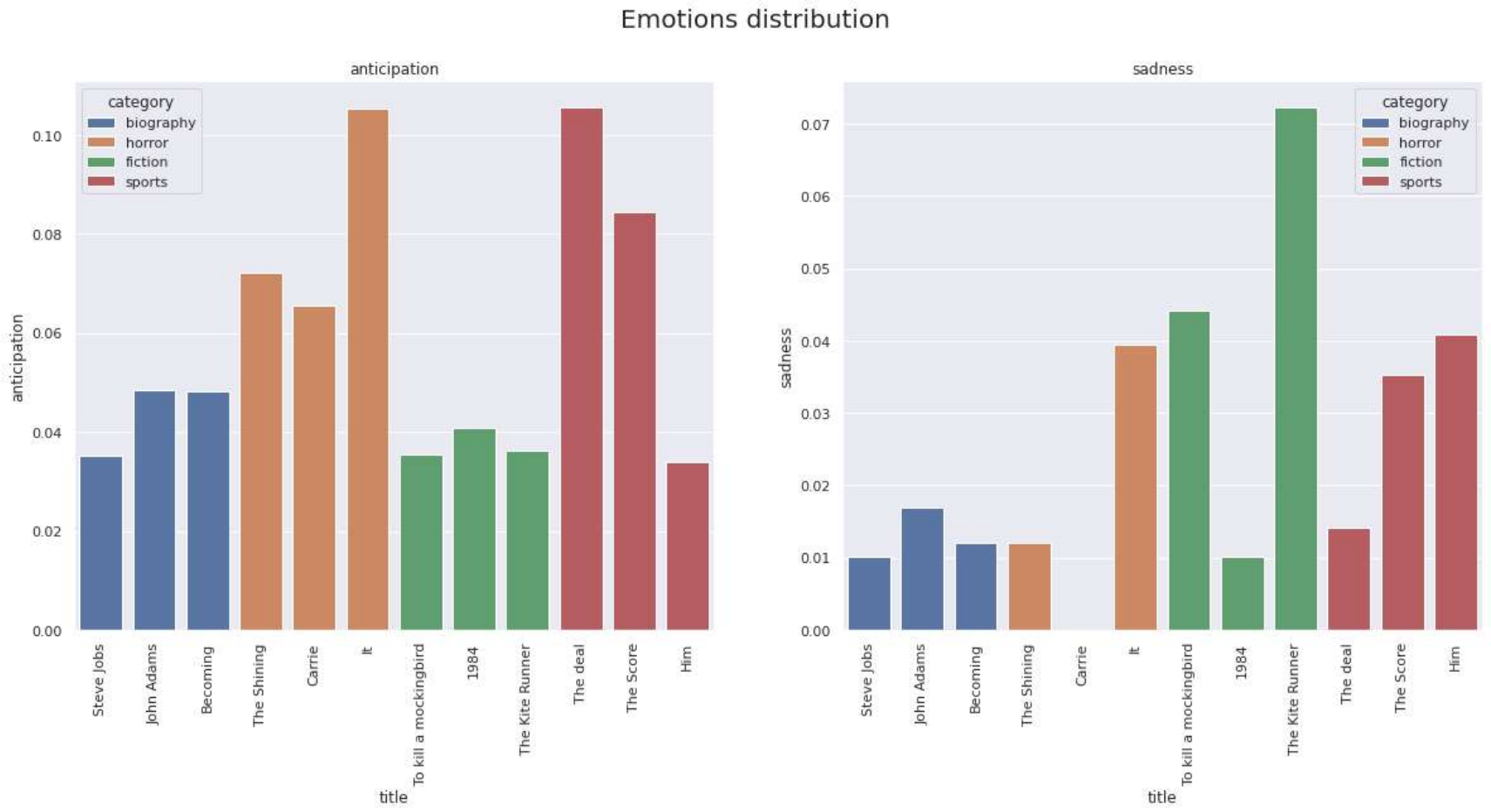
In [ ]:
```
emotions_mapping('fear', 'joy')
```



In [ ]:
```
emotions_mapping('trust', 'surprise')
```



In [ ]:
```
emotions_mapping('anticipation', 'sadness')
```



In [ ]:
```
emotions_mapping('positive', 'negative')
```