# Sentiment Analysis on the IMDB Movie Review Dataset

## Data Science Capstone Project

## Exploratory Data Analytics Report

Date:

[02/17/2021]

Team Members: 4

Name: Lawrence Love

Name: Gustavo Ferreira

Name: Frank Zhao

Name: Yan Li

## Analysis of the basic metrics and variables

| | review | sentiment |
|---|---|---|
| **0** | One of the other reviewers has mentioned that ... | positive |
| **1** | A wonderful little production. <br /><br />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 |
| **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 movies to be high... | negative |

With our first look at the data, we see that we have 50,000 rows of data with two columns: ‘review’ and ‘sentiment’.

```
# Check for balanced dataset
df['sentiment'].value_counts()
```

```
negative    25000
positive    25000
```

The dataset is balanced having an equal number of positive and negative reviews at 25,000 each.

### Insights from a datapoint

```
print(df['review'][3])
print('')
print('Sentiment:',df['sentiment'][3])
```

```
Basically there's a family where a little boy (Jake) thinks there's a zombie in his clos
et & his parents are fighting all the time.<br /><br />This movie is slower than a soap
opera... and suddenly, Jake decides to become Rambo and kill the zombie.<br /><br />OK,
first of all when you're going to make a film you must Decide if its a thriller or a dra
ma! As a drama the movie is watchable. Parents are divorcing & arguing like in real lif
e. And then we have Jake with his closet which totally ruins all the film! I expected to
see a BOOGEYMAN similar movie, and instead i watched a drama with some meaningless thril
ler spots.<br /><br />3 out of 10 just for the well playing parents & descent dialogs. A
s for the shots with Jake: just ignore them.

Sentiment: negative
```

With a look at this review, we can see that we will have a few different types of data pre-processing to go through even before tokenization:

- HTML coding removal
- Remove punctuation and other symbols
- Stop-word removal
- Lowercasing of all words

After the steps above, we apply Lemmatization to remove similar words such as singlular and plural words.

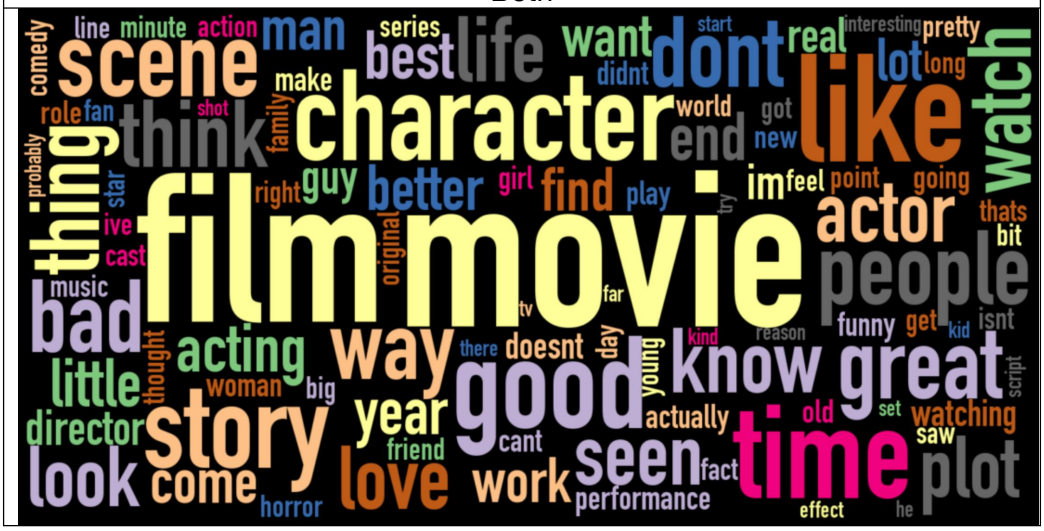
## Non-Graphical and Graphical Univariate Analysis

First of all, we generate ngram word clouds for the most frequent 100 words for positive, negative and total reviews.

We combine the stop words dictionaries from spacy, nltk and wordclouds. For the unigram, we also remove ‘not’. For bigram and trigram, we don’t update words to the stop words dictionary to avoid the combination result. For example, people may comment ‘not good’, ‘not bad’, etc.

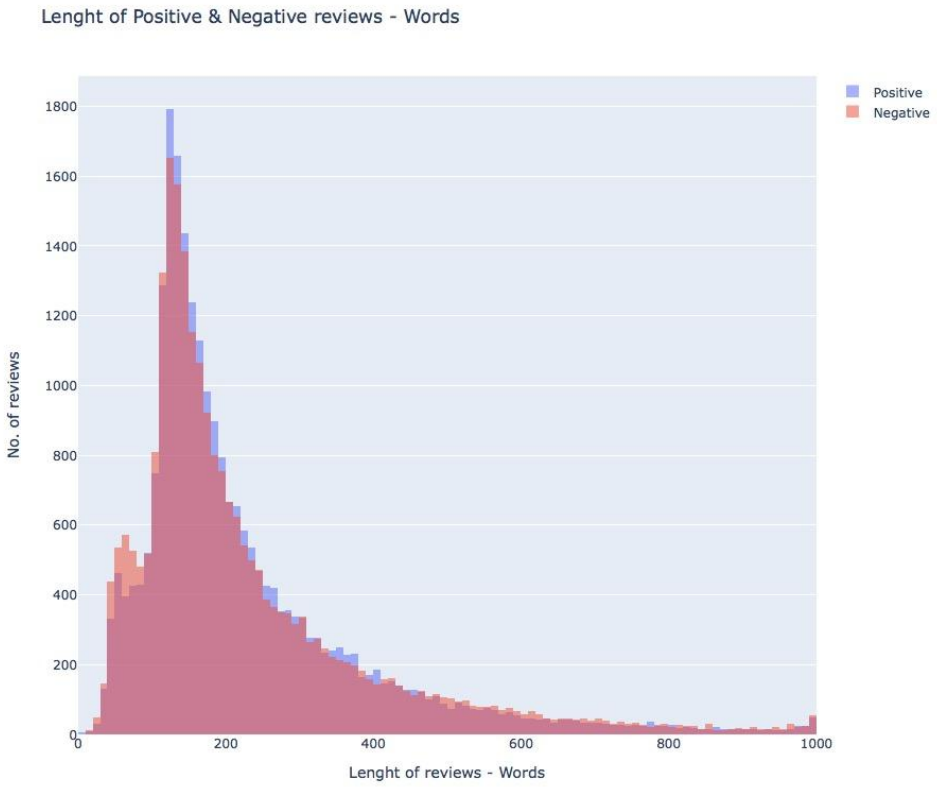
## Unigram

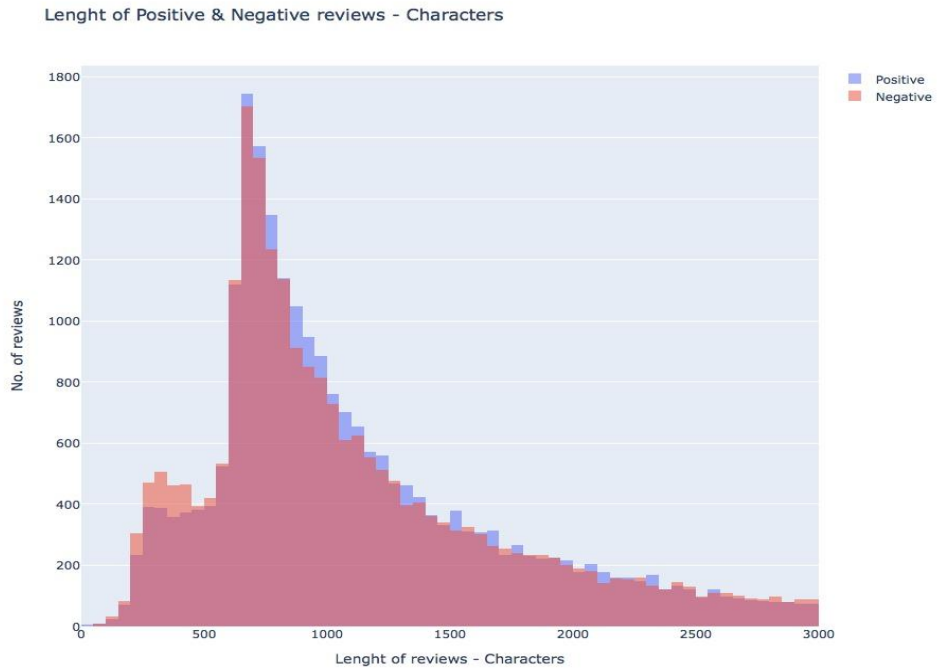
| Positive | Negative |
| --- | --- |
|  |  |

| Both |
| --- |
|  |

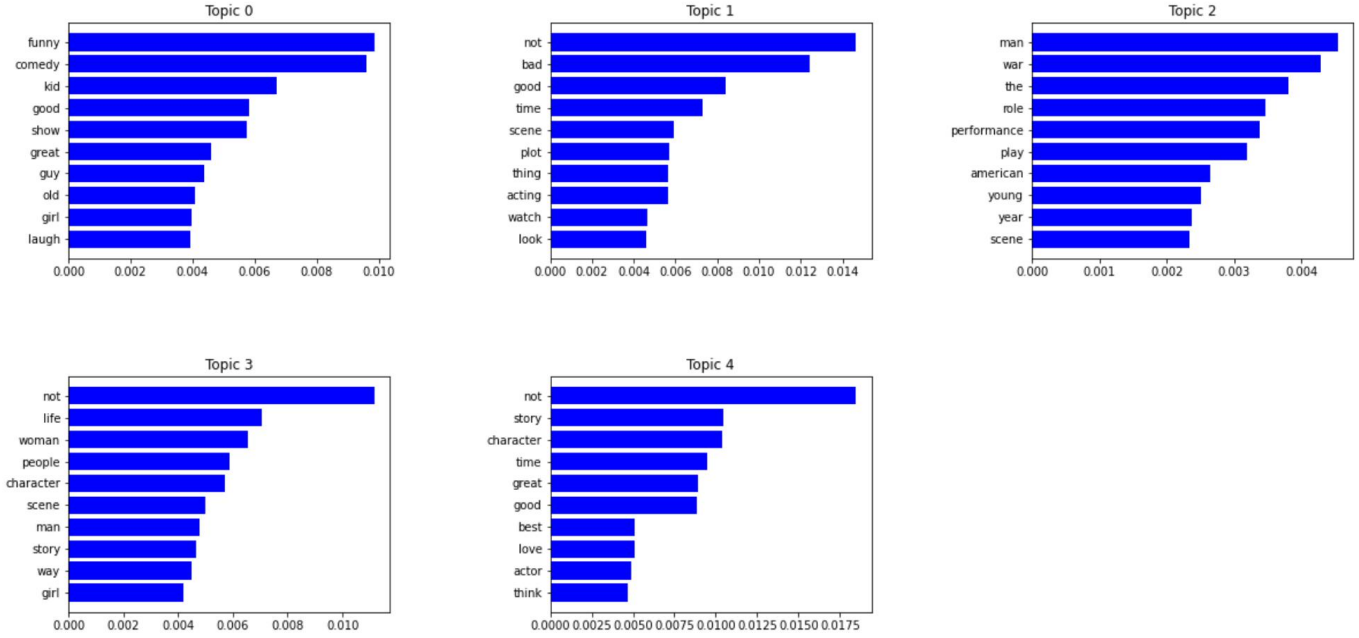
## Length of reviews

We also analyzed the review lengths by words and characters. We can see that the length of the most common reviews is around 120 words, with positive reviews being slightly longer tat negative reviews.


Lenght of Positive & Negative reviews - Words
Positive
Negative
No. of reviews
Lenght of reviews - Words
0
200
400
600
800
1000
1200
1400
1600
1800


Lenght of Positive & Negative reviews - Characters
Positive
Negative
No. of reviews
Lenght of reviews - Characters
0
500
1000
1500
2000
2500
3000
200
400
600
800
1000
1200
1400
1600
1800

### Latent Dirichlet Allocation




## Missing value analysis and outlier analysis

We don't have any missing value and outlier data in our dataset since the dataset that we have is all text data.

## Feature engineering and analysis

### Column Transfer

We convert the categorical column `sentiment` to the numerical values: negative changed to 0 and positive changed to 1 in order to apply the machine learning algorithm to this dataset.

### Sentiment Score

We create sentiment score for each review in order to get more features to apply the machine learning algorithm: Subjectivity, Polarity, Compound, Negative, Neutral, Positive.

## TF–IDF

To analyze the importance of the words in the reviews, we checked the TF-IDF scores. TF-IDF measures the importance of a word to a document, which is part of a collection of documents. In this case, the significance of the words in the revies corpus.

First, we checked the IDF (Inverse document frequency) since it is related to the whole corpus.

```
# Words with lowest IDF values
sorted(word_idf, key = lambda x: x[1])[:10]
```

```
[('the', 1.0088185941563523),
 ('and', 1.0345700375594091),
 ('of', 1.0526615711256),
 ('to', 1.0623210259973819),
 ('this', 1.0988463541068978),
 ('is', 1.1114209443936836),
 ('it', 1.1162392587927048),
 ('in', 1.1276034025546802),
 ('that', 1.211284388584135),
 ('but', 1.334262723632932)]
```

```
#Words with highest IDF values
sorted(word_idf, key = lambda x: x[1], reverse=True)[1:10]
```

```
[('kickbr', 11.819798284210286),
 ('popcornmunching', 11.819798284210286),
 ('soaplike', 11.819798284210286),
 ('grudgeboogeymanringsaw', 11.819798284210286),
 ('fascistsbr', 11.819798284210286),
 ('blooddrinking', 11.819798284210286),
 ('eyejam', 11.819798284210286),
 ('syndromeher', 11.819798284210286),
 ('halfsmile', 11.819798284210286)]
```

We also got the TF-IDF scores (Term Frequency - Inverse document frequency). Here is one example of the values.

Review:

```
corpus[6]
```

```
"i sure would like to see a resurrection of a up dated seahunt series with the tech they have today it would bring ba
ck the kid excitement in me.i grew up on black and white tv and seahunt with gunsmoke were my hero's every week.you h
ave my vote for a comeback of a new sea hunt.we need a change of pace in tv and this would work for a world of under
water adventure.oh by the way thank you for an outlet like this to view many viewpoints about tv and the many movies.
so any ole way i believe i've got what i wanna say.would be nice to read some more plus points about sea hunt.if my r
hymes would be 10 lines would you let me submit,or leave me out to be in doubt and have me to quit,if this is so then
i must go so lets do it."
```

Top 10 TF-IDF scores of the review

| | tfidf |
|---|---|
| **seahunt** | 0.408303 |
| **would** | 0.234290 |
| **hunt** | 0.217947 |
| **sea** | 0.205761 |
| **tv** | 0.190001 |
| **gunsmoke** | 0.167182 |
| **me** | 0.163403 |
| **viewpoints** | 0.151636 |
| **outlet** | 0.151119 |
| **rhymes** | 0.148733 |

# Appendix

## 1.Code for Text Preprocessing

**Remove html coding**

```
df['review'] = df['review'].str.replace('<.*?>','')
```

**Remove all puncuation and symbols**

```
df['review'] = df['review'].str.replace('[^\w\s]','')
```

**Make everything lower case**

```
df['review'] = df['review'].str.lower()
```

**Remove stop words**

```
import spacy
sp = spacy.load('en_core_web_sm')
all_stopwords = sp.Defaults.stop_words
# # After seeing the word counts, update stop words
sp.Defaults.stop_words |= {'movie', 'film', 'like'}
```

```
df['review'] = df['review'].apply(lambda x: ' '.join([word for word in x.split() if word not in (all_stopwords)]))
```

**Tokenize**

```
w_tokenizer = nltk.tokenize.WhitespaceTokenizer()
lemmatizer = nltk.stem.WordNetLemmatizer()
```

```
def lemmatize_text(text):
    return [lemmatizer.lemmatize(w) for w in w_tokenizer.tokenize(text)]
```

```
df['lemma_review'] = df.review.apply(lemmatize_text)
```

## 2.Code for Sentiment Score: subjectivity and polarity

```
#Create a function to get subjectivity and polarity
def getSubjectivity(text):

    return TextBlob(text).sentiment.subjectivity

def getPolarity(text):

    return TextBlob(text).sentiment.polarity
```

```
#Create two columns 'Subjectivity' and 'Polarity'
df['Subjectivity'] = df['review'].apply(getSubjectivity)
df['Polarity'] = df['review'].apply(getPolarity)
```

## Table of Contributions

The table below identifies contributors to various sections of this document.

| | Section | Writing | Editing |
|---|---|---|---|
| 1 | Analysis the basic metrics of variables | Lowrance Love | Lowrance Love |
| 2 | Non-graphical and graphical univariate analysis | Lowrance Love, Frank Zhao | Frank Zhao |
| 3 | Missing value analysis and outlier analysis | Yan Li | Yan Li |
| 4 | Feature engineering and analysis | Yan Li, Gustavo Ferreira | Yan Li |
| 5 | Appendix | Yan Li | Yan Li |