Sentiment Analysis

In [1]:

```
!pip install -U nltk scikit-learn pandas matplotlib numpy wordcloud
```

```
Collecting nltk
  Downloading https://files.pythonhosted.org/packages/87/16/4d247e27c55a7b64
12e7c4c86f2500ae61afcbf5932b9e3491f8462f8d9e/nltk-3.4.4.zip (https://files.p
ythonhosted.org/packages/87/16/4d247e27c55a7b6412e7c4c86f2500ae61afcbf5932b9
e3491f8462f8d9e/nltk-3.4.4.zip) (1.5MB)
Requirement already up-to-date: scikit-learn in c:\program files (x86)\micro
soft visual studio\shared\anaconda3_64\lib\site-packages (0.21.3)
Requirement already up-to-date: pandas in c:\program files (x86)\microsoft v
isual studio\shared\anaconda3_64\lib\site-packages (0.25.0)
Requirement already up-to-date: matplotlib in c:\program files (x86)\microso
ft visual studio\shared\anaconda3_64\lib\site-packages (3.1.1)
Requirement already up-to-date: numpy in c:\program files (x86)\microsoft vi
sual studio\shared\anaconda3_64\lib\site-packages (1.17.0)
Requirement already up-to-date: wordcloud in c:\program files (x86)\microsof
t visual studio\shared\anaconda3_64\lib\site-packages (1.5.0)
Requirement already satisfied, skipping upgrade: six in c:\program files (x8
6)\microsoft visual studio\shared\anaconda3_64\lib\site-packages (from nltk)
(1.11.0)
Requirement already satisfied, skipping upgrade: joblib>=0.11 in c:\program
 files (x86)\microsoft visual studio\shared\anaconda3_64\lib\site-packages
 (from scikit-learn) (0.13.2)
Requirement already satisfied, skipping upgrade: scipy>=0.17.0 in c:\program
files (x86)\microsoft visual studio\shared\anaconda3_64\lib\site-packages (f
rom scikit-learn) (1.1.0)
Requirement already satisfied, skipping upgrade: python-dateutil>=2.6.1 in
 c:\program files (x86)\microsoft visual studio\shared\anaconda3_64\lib\site
-packages (from pandas) (2.7.3)
Requirement already satisfied, skipping upgrade: pytz>=2017.2 in c:\program
 files (x86)\microsoft visual studio\shared\anaconda3_64\lib\site-packages
 (from pandas) (2018.4)
Requirement already satisfied, skipping upgrade: pyparsing!=2.0.4,!=2.1.2,!=
2.1.6,>=2.0.1 in c:\program files (x86)\microsoft visual studio\shared\anaco
nda3_64\lib\site-packages (from matplotlib) (2.2.0)
Requirement already satisfied, skipping upgrade: kiwisolver>=1.0.1 in c:\pro
gram files (x86)\microsoft visual studio\shared\anaconda3_64\lib\site-packag
es (from matplotlib) (1.0.1)
Requirement already satisfied, skipping upgrade: cycler>=0.10 in c:\program
 files (x86)\microsoft visual studio\shared\anaconda3_64\lib\site-packages
 (from matplotlib) (0.10.0)
Requirement already satisfied, skipping upgrade: pillow in c:\program files
 (x86)\microsoft visual studio\shared\anaconda3_64\lib\site-packages (from w
ordcloud) (5.1.0)
Requirement already satisfied, skipping upgrade: setuptools in c:\program fi
les (x86)\microsoft visual studio\shared\anaconda3_64\lib\site-packages (fro
m kiwisolver>=1.0.1->matplotlib) (41.0.1)
Building wheels for collected packages: nltk
  Building wheel for nltk (setup.py): started
  Building wheel for nltk (setup.py): finished with status 'done'
  Created wheel for nltk: filename=nltk-3.4.4-cp36-none-any.whl size=1449699
sha256=5c5726dc54150078054965eb806558d0dcc8033e826d3d49b0123867fb1403bd
  Stored in directory: C:\Users\a.lima.laurentino\AppData\Local\pip\Cache\wh
eels\41\c8\31\48ace4468e236e0e8435f30d33e43df48594e4d53e367cf061
Successfully built nltk
Installing collected packages: nltk
  Found existing installation: nltk 3.4.3
    Uninstalling nltk-3.4.3:
```

```
ERROR: Could not install packages due to an EnvironmentError: [WinError 5] A
ccess is denied: 'c:\\program files (x86)\\microsoft visual studio\\shared
\\anaconda3_64\\lib\\site-packages\\nltk-3.4.3.dist-info\\INSTALLER'
Consider using the `--user` option or check the permissions.

WARNING: You are using pip version 19.2.1, however version 19.2.2 is availab
le.
You should consider upgrading via the 'python -m pip install --upgrade pip'
command.
```

Read the csv

In [2]:

```
import pandas as pd

reviews = pd.read_csv('./files/imdb-reviews.csv')
reviews['sentiment'] = reviews['sentiment'].map({'neg': 0, 'pos': 1})
reviews.drop(columns=["text_pt", "id"], inplace=True)
reviews.head(10)
```

Out[2]:

	text_en	sentiment
0	Once again Mr. Costner has dragged out a movie...	0
1	This is an example of why the majority of acti...	0
2	First of all I hate those moronic rappers, who...	0
3	Not even the Beatles could write songs everyon...	0
4	Brass pictures movies is not a fitting word fo...	0
5	A funny thing happened to me while watching "M...	0
6	This German horror film has to be one of the w...	0
7	Being a long-time fan of Japanese film, I expe...	0
8	"Tokyo Eyes" tells of a 17 year old Japanese g...	0
9	Wealthy horse ranchers in Buenos Aires have a ...	0

Preprocessing

In [3]:

```
import re
from nltk.tokenize import WhitespaceTokenizer
```

1. Cleaning:

```
Clear and get only the main part from the dataset
Ex: remove the tags of the html.
Ex: filter the texts in PDF and etc.
```

In [4]:

```
def clean(text):
    # Remove the HTML tags
    text = re.sub("<!--?.*?-->","",text)
    text = re.sub("<.*?>","",text)

    return text
```

2. Normalization:

```
Remove the pontuation, tags, put everything in same case and etc.
```

In [5]:

```
def normalize(text):
    # Convert to lower case
    text = text.lower()

    # remove special characters and digits
    text = re.sub("(\\d|\\W)+", " ",text)

    text = text.replace('  ', ' ')

    return text
```

3. Tokenization:

```
Split the text in words spliting by the whitespaces.
```

In [6]:

```
def tokenizer(text):
    tokenizer = WhitespaceTokenizer()
    tokens = tokenizer.tokenize(text)

    return tokens
```

4. Stop Words:

```
They are words witch don't get no one meaning, they are just used to complement th
e context,
and to connect the terms.
Ex: 'i', 'you', 'in', 'out', 'are', 'the'
```

In [7]:

```
from nltk.corpus import stopwords

# Remove the stop words, they are words witch don't give no one especific meaning
def remove_stopwords(tokens):
    return [w for w in tokens if w not in stopwords.words("english")]
```

5. Stemming:

```
Takes of the variation of the words and remove the finally to combine than.
Ex: 'change', 'changing', 'changes' => 'chang'
```

6. Lemmatization:

```
Takes the variation of the same word and convert to the same one (Noun).
Ex: 'is', 'were', 'was' => 'be'
Ex: 'ones' => 'one'

Part of Speech(PoS) (Verb):
Ex: 'bored' => 'bore'
Ex: 'stating' => 'start'
```

In [8]:

```
from nltk.stem.porter import PorterStemmer
from nltk.stem.wordnet import WordNetLemmatizer

# Remove the pural
# Remove the verb conjugation
def stem(words):
    return [PorterStemmer().stem(w) for w in words]

# Remove the personality
def lem(words):
    return [WordNetLemmatizer().lemmantize(w) for w in words]
```

7. Tag filtering

```
Filter the words according with the sintaxe definition like a noun, verbs, adverbs
e etc.
```

In [9]:

```
import nltk

# RB | RB | JJ | NN | NNP | JJ | JJS | IN | VB | VBZ | VBD | VBG

# IN = preposition/subordinating conjunction

# RB = adverb very, silently
# RBR = adverb, comparative better
# RBS = adverb, superlative best
# RP = particle give up

# IN = preposition/subordinating conjunction
# JJ = adjective ‘big’
# JJR = adjective, comparative ‘bigger’
# JJS = adjective, superlative ‘biggest’

# VB verb, base form take
# VBD verb, past tense took
# VBG verb, gerund/present participle taking
# VBN verb, past participle taken
# VBP verb, sing. present, non-3d take
# VBZ verb, 3rd person sing. present takes

# Filter using regular array
def filter_tokens(tokens):
    tags = [x[1] for x in nltk.pos_tag(list(tokens))]
    filters = ("RB", "RBR", "RBS", "RP", "JJ", "JJR", "JJS", "JJ", "VB")

    return [tokens[i] for i in range(len(tokens)) if tags[i] in filters]
```

Bag of Words

In [10]:

```
all_words = " ".join(list(reviews.text_en[:1000]))

tokens = tokenizer(normalize(clean(all_words)))
print("> Tokenized!")

tokens = remove_stopwords(tokens)
print("> Removed the stop words!")

tokens = stem(tokens)
# tokens = lem(tokens)
print("> Merged the term by stem or lem!")

tokens = filter_tokens(tokens)
print("> Filtred by tags witch get more meaning!")

print("\nColection [:100]:\n")
print(tokens[:100])
```

```
> Tokenized!
> Removed the stop words!
> Merged the term by stem or lem!
> Filtred by tags witch get more meaning!

Colection [:100]:

['far', 'longer', 'necessari', 'charact', 'forgotten', 'much', 'later', 'bet
ter', 'sign', 'win', 'final', 'half', 'kutcher', 'best', 'prior', 'keep', 't
urn', 'major', 'realli', 'worth', 'tap', 'proven', 'mani', 'well', 'dont',
'bother', 'go', 'see', 'new', 'new', 'undercov', 'n', 'higher', 'friday', 'r
eal', 'clich', 'still', 'wonder', 'alway', 'play', 'exact', 'charact', 'alie
n', 'ive', 'exact', 'irrit', 'least', 'alien', 'somewhat', 'gratifi', 'overa
l', 'second', 'better', 'see', 'practic', 'better', 'better', 'script', 'wor
th', 'decent', 'almost', 'refresh', 'close', 'first', 'hate', 'gun', 'go',
'alreadi', 'warehous', 'also', 'sadler', 'much', 'right', 'peopl', 'everywhe
r', 'pretti', 'much', 'big', 'get', 'deserv', 'black', 'ugli', 'dead', 'sta
y', 'away', 'crap', 'instead', 'lest', 'real', 'even', 'write', 'song', 'mo
p', 'top', 'provok', 'social', 'movi', 'full', 'back', 'seat']
```

Get the frequency of the words

In [11]:

```
import nltk

frequency = nltk.FreqDist(tokens)

# Create the bag of words dataframe
bag_of_words = pd.DataFrame({"words": list(frequency.keys()), "frequency": list(frequency.v

# Order by the Frequency
bag_of_words.sort_values(by="frequency", ascending=False, inplace=True)
bag_of_words.reset_index(drop=True, inplace=True)

# Save the bag of words
bag_of_words.to_csv('./files/bag-of-words.csv', index=True)

print(f"Back of words size: {bag_of_words.shape[0]}")

print(bag_of_words.shape[0])
bag_of_words.head(5)
```

```
Back of words size: 5190
5190
```

Out[11]:

	words	frequency
0	bad	648
1	even	633
2	good	532
3	movi	430
4	much	387

Plot the frequency in Word Cloud

In [12]:

```
def to_single_str(words, frequency):
    words = list(words)
    frequency = list(frequency)

    return " ".join([(words[i] + " ") * frequency[i] for i in range(len(frequency))])
```

In [13]:

```
from wordcloud import WordCloud
import matplotlib.pyplot as plt

word_cloud = WordCloud(width=800, height=500, max_font_size=110, collocations=False).genera

plt.figure(figsize=(13, 13))
plt.imshow(word_cloud)
plt.show()
```

```
<Figure size 1300x1300 with 1 Axes>
```

Feature Selection

TF-IDF

Configs

In [14]:

```
# Load the dictionary
bag_of_words = pd.read_csv('./files/bag-of-words.csv')
bag_of_words_array = bag_of_words.words.values

# Get the inputs
reviews = pd.read_csv('./files/imdb-reviews.csv')
reviews['sentiment'] = reviews['sentiment'].map({'neg': 0, 'pos': 1})
reviews.drop(columns=["text_pt", "id"], inplace=True)

inputs = reviews.text_en.values
```

Implementation

In [15]:

```
import re
from sklearn.feature_extraction.text import TfidfVectorizer


def tf_idf(txt, vocabulary=None):
    txt = list(txt)

    tf = TfidfVectorizer(smooth_idf=False, sublinear_tf=False, norm=None, analyzer='word',
    txt_transformed = tf.fit(txt).transform(txt)

    return pd.DataFrame(txt_transformed.toarray(), columns=tf.get_feature_names())
```

In [16]:

```
tfidf = tf_idf(inputs, bag_of_words_array)
tfidf.head(10)
```

```
C:\Program Files (x86)\Microsoft Visual Studio\Shared\Anaconda3_64\lib\site-
packages\sklearn\feature_extraction\text.py:1278: RuntimeWarning: divide by
zero encountered in true_divide
  idf = np.log(n_samples / df) + 1
```

Out[16]:

	bad	even	good	movi	much	get	well	first	better	ever
0	0.0	0.00000	0.000000	0.0	2.286672	0.000000	2.212549	0.000000	2.701146	0.000000
1	0.0	0.00000	0.000000	0.0	0.000000	0.000000	2.212549	0.000000	8.103437	0.000000
2	0.0	0.00000	0.000000	0.0	4.573344	2.337195	0.000000	2.398643	0.000000	0.000000
3	0.0	2.11299	0.000000	0.0	0.000000	0.000000	2.212549	0.000000	0.000000	0.000000
4	0.0	0.00000	1.977176	0.0	2.286672	0.000000	2.212549	0.000000	0.000000	0.000000
5	0.0	0.00000	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
6	0.0	0.00000	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
7	0.0	0.00000	1.977176	0.0	2.286672	0.000000	0.000000	0.000000	0.000000	2.657248
8	0.0	0.00000	0.000000	0.0	0.000000	4.674390	0.000000	0.000000	0.000000	0.000000
9	0.0	2.11299	1.977176	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

10 rows × 5190 columns

Modeling

In [17]:

```
from sklearn.model_selection import train_test_split

x = tfidf.values
y = [[x] for x in reviews.sentiment.values]

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.25, random_state=0, s
```

Dummy Classifier

In [18]:

```
from sklearn.dummy import DummyClassifier

model = DummyClassifier()
model.fit(x_train, y_train)

accuracy = model.score(x_test, y_test) * 100
print("Taxa de acerto do algoritimo de Base line: %.2f%%" % accuracy)
```

```
Taxa de acerto do algoritimo de Base line: 50.26%
```

Linear SVC

In [19]:

```
from sklearn.svm import LinearSVC
from sklearn.metrics import accuracy_score
import numpy as np

np.random.seed(5)

# Test a linear model
model = LinearSVC()
model.fit(x_train, y_train)

accuracy = model.score(x_test, y_test) * 100
print("Linear SVC accuracy: %.2f%%" % accuracy)
```

```
C:\Program Files (x86)\Microsoft Visual Studio\Shared\Anaconda3_64\lib\site-
packages\sklearn\utils\validation.py:724: DataConversionWarning: A column-ve
ctor y was passed when a 1d array was expected. Please change the shape of y
to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)

Linear SVC accuracy: 79.80%

C:\Program Files (x86)\Microsoft Visual Studio\Shared\Anaconda3_64\lib\site-
packages\sklearn\svm\base.py:929: ConvergenceWarning: Liblinear failed to co
nverge, increase the number of iterations.
  "the number of iterations.", ConvergenceWarning)
```