Creating a Sentiment Analysis Web App

Using PyTorch and SageMaker

Deep Learning Nanodegree Program | Deployment

Now that we have a basic understanding of how SageMaker works we will try to use it to construct a complete project from end to end. Our goal will be to have a simple web page which a user can use to enter a movie review. The web page will then send the review off to our deployed model which will predict the sentiment of the entered review.

Instructions

Some template code has already been provided for you, and you will need to implement additional functionality to successfully complete this notebook. You will not need to modify the included code beyond what is requested. Sections that begin with '**TODO**' in the header indicate that you need to complete or implement some portion within them. Instructions will be provided for each section and the specifics of the implementation are marked in the code block with a `# TODO: ...` comment. Please be sure to read the instructions carefully!

In addition to implementing code, there will be questions for you to answer which relate to the task and your implementation. Each section where you will answer a question is preceded by a '**Question:**' header. Carefully read each question and provide your answer below the '**Answer:**' header by editing the Markdown cell.

> **Note**: Code and Markdown cells can be executed using the **Shift+Enter** keyboard shortcut. In addition, a cell can be edited by typically clicking it (double-click for Markdown cells) or by pressing **Enter** while it is highlighted.

General Outline

Recall the general outline for SageMaker projects using a notebook instance.

1. Download or otherwise retrieve the data.
2. Process / Prepare the data.
3. Upload the processed data to S3.
4. Train a chosen model.
5. Test the trained model (typically using a batch transform job).
6. Deploy the trained model.
7. Use the deployed model.

For this project, you will be following the steps in the general outline with some modifications.

First, you will not be testing the model in its own step. You will still be testing the model, however, you will do it by deploying your model and then using the deployed model by sending the test data to it. One of the reasons for doing this is so that you can make sure that your deployed model is working correctly before moving forward.

In addition, you will deploy and use your trained model a second time. In the second iteration you will customize the way that your trained model is deployed by including some of your own code. In addition, your newly deployed model will be used in the sentiment analysis web app.

In [1]:
```
# Make sure that we use SageMaker 1.x
!pip install sagemaker==1.72.0
```

```
Collecting sagemaker==1.72.0
  Downloading sagemaker-1.72.0.tar.gz (297 kB)
     |████████████████████████████████| 297 kB 24.3 MB/s eta 0:00:01
Requirement already satisfied: boto3>=1.14.12 in /home/ec2-user/anaconda3/envs/amazonei_pytorch_latest_p36/lib/python3.6/site-packages (from sagemaker==1.72.0) (1.18.45)
Requirement already satisfied: numpy>=1.9.0 in /home/ec2-user/anaconda3/envs/amazonei_pytorch_latest_p36/lib/python3.6/site-packages (from sagemaker==1.72.0) (1.19.5)
Requirement already satisfied: protobuf>=3.1 in /home/ec2-user/anaconda3/envs/amazonei_pytorch_latest_p36/lib/python3.6/site-packages (from sagemaker==1.72.0) (3.18.0)
Requirement already satisfied: scipy>=0.19.0 in /home/ec2-user/anaconda3/envs/amazonei_pytorch_latest_p36/lib/python3.6/site-packages (from sagemaker==1.72.0) (1.5.3)
Requirement already satisfied: protobuf3-to-dict>=0.1.5 in /home/ec2-user/anaconda3/envs/amazonei_pytorch_latest_p36/lib/python3.6/site-packages (from sagemaker==1.72.0) (0.1.5)
Collecting smdebug-rulesconfig==0.1.4
  Downloading smdebug_rulesconfig-0.1.4-py2.py3-none-any.whl (10 kB)
Requirement already satisfied: importlib-metadata>=1.4.0 in /home/ec2-user/anaconda3/envs/amazonei_pytorch_latest_p36/lib/python3.6/site-packages (from sagemaker==1.72.0) (4.5.0)
Requirement already satisfied: packaging>=20.0 in /home/ec2-user/anaconda3/envs/amazonei_pytorch_latest_p36/lib/python3.6/site-packages (from sagemaker==1.72.0) (20.9)
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /home/ec2-user/anaconda3/envs/amazonei_pytorch_latest_p36/lib/python3.6/site-packages (from boto3>=1.14.12->sagemaker==1.72.0) (0.10.0)
Requirement already satisfied: s3transfer<0.6.0,>=0.5.0 in /home/ec2-user/anaconda3/envs/amazonei_pytorch_latest_p36/lib/python3.6/site-packages (from boto3>=1.14.12->sagemaker==1.72.0) (0.5.0)
Requirement already satisfied: botocore<1.22.0,>=1.21.45 in /home/ec2-user/anaconda3/envs/amazonei_pytorch_latest_p36/lib/python3.6/site-packages (from boto3>=1.14.12->sagemaker==1.72.0) (1.21.45)
Requirement already satisfied: urllib3<1.27,>=1.25.4 in /home/ec2-user/anaconda3/envs/amazonei_pytorch_latest_p36/lib/python3.6/site-packages (from botocore<1.22.0,>=1.21.45->boto3>=1.14.12->sagemaker==1.72.0) (1.26.5)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /home/ec2-user/anaconda3/envs/amazonei_pytorch_latest_p36/lib/python3.6/site-packages (from botocore<1.22.0,>=1.21.45->boto3>=1.14.12->sagemaker==1.72.0) (2.8.1)
Requirement already satisfied: zipp>=0.5 in /home/ec2-user/anaconda3/envs/amazonei_pytorch_latest_p36/lib/python3.6/site-packages (from importlib-metadata>=1.4.0->sagemaker==1.72.0) (3.4.1)
Requirement already satisfied: typing-extensions>=3.6.4 in /home/ec2-user/anaconda3/envs/amazonei_pytorch_latest_p36/lib/python3.6/site-packages (from importlib-metadata>=1.4.0->sagemaker==1.72.0) (3.10.0.0)
Requirement already satisfied: pyparsing>=2.0.2 in /home/ec2-user/anaconda3/envs/amazonei_pytorch_latest_p36/lib/python3.6/site-packages (from packaging>=20.0->sagemaker==1.72.0) (2.4.7)
Requirement already satisfied: six in /home/ec2-user/anaconda3/envs/amazonei_pytorch_latest_p36/lib/python3.6/site-packages (from protobuf>=3.1->sagemaker==1.72.0) (1.16.0)
Building wheels for collected packages: sagemaker
  Building wheel for sagemaker (setup.py) ... done
  Created wheel for sagemaker: filename=sagemaker-1.72.0-py2.py3-none-any.whl size=388327 sha256=83aa49be54a2c57bf844ee144f53395354e128a3e28e7e7c2cb36bd96c864fc
  Stored in directory: /home/ec2-user/.cache/pip/wheels/c3/58/70/85faf4437568bfaa4c4
Successfully built sagemaker
Installing collected packages: smdebug-rulesconfig, sagemaker
  Attempting uninstall: smdebug-rulesconfig
    Found existing installation: smdebug-rulesconfig 1.0.1
    Uninstalling smdebug-rulesconfig-1.0.1:
      Successfully uninstalled smdebug-rulesconfig-1.0.1
  Attempting uninstall: sagemaker
    Found existing installation: sagemaker 2.59.3
    Uninstalling sagemaker-2.59.3:
      Successfully uninstalled sagemaker-2.59.3
Successfully installed sagemaker-1.72.0 smdebug-rulesconfig-0.1.4
WARNING: You are using pip version 21.2.4; however, version 21.3 is available.
You should consider upgrading via the '/home/ec2-user/anaconda3/envs/amazonei_pytorch_
latest_p36/bin/python -m pip install --upgrade pip' command.
```

Step 1: Downloading the data

As in the XGBoost in SageMaker notebook, we will be using the IMDb dataset

> Maas, Andrew L., et al. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2011.

In [2]:
```
%mkdir ../data
!wget -O ../data/aclImdb_v1.tar.gz http://ai.stanford.edu/~amaas/data/sentiment/aclImdb_v1.tar.gz
!tar -zxf ../data/aclImdb_v1.tar.gz -C ../data
```

```
mkdir: cannot create directory '../data': File exists
--2021-10-15 15:16:13--  http://ai.stanford.edu/~amaas/data/sentiment/aclImdb_v1.tar.gz
Resolving ai.stanford.edu (ai.stanford.edu)... 171.64.68.10
Connecting to ai.stanford.edu (ai.stanford.edu)|171.64.68.10|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 84125825 (80M) [application/x-gzip]
Saving to: '../data/aclImdb_v1.tar.gz'

../data/aclImdb_v1. 100%[===================>]  80.23M  23.7MB/s    in 3.6s

2021-10-15 15:16:17 (22.5 MB/s) - '../data/aclImdb_v1.tar.gz' saved [84125825/84125825]
```

Step 2: Preparing and Processing the data

Also, as in the XGBoost notebook, we will be doing some initial data processing. The first few steps are the same as in the XGBoost example. To begin with, we will read in each of the reviews and combine them into a single input structure. Then, we will split the dataset into a training set and a testing set.

In [3]:
```
import os
import glob

def read_imdb_data(data_dir='../data/aclImdb'):
    data = {}
    labels = {}

    for data_type in ['train', 'test']:
        data[data_type] = {}
        labels[data_type] = {}

        for sentiment in ['pos', 'neg']:
            data[data_type][sentiment] = []
            labels[data_type][sentiment] = []

            path = os.path.join(data_dir, data_type, sentiment, '*.txt')
            files = glob.glob(path)

            for f in files:
                with open(f) as review:
                    data[data_type][sentiment].append(review.read())
                    # Here we represent a positive review by '1' and a negative review by '0'
                    labels[data_type][sentiment].append(1 if sentiment == 'pos' else 0)

            assert len(data[data_type][sentiment]) == len(labels[data_type][sentiment]), \
                    "{}/{} data size does not match labels size".format(data_type, sentiment)

    return data, labels
```

In [4]:
```
data, labels = read_imdb_data()
print("IMDB reviews: train = {} pos / {} neg, test = {} pos / {} neg".format(
            len(data['train']['pos']), len(data['train']['neg']),
            len(data['test']['pos']), len(data['test']['neg'])))
```

```
IMDB reviews: train = 12500 pos / 12500 neg, test = 12500 pos / 12500 neg
```

Now that we've read the raw training and testing data from the downloaded dataset, we will combine the positive and negative reviews and shuffle the resulting records.

In [5]:
```
from sklearn.utils import shuffle

def prepare_imdb_data(data, labels):
    """Prepare training and test sets from IMDb movie reviews."""

    #Combine positive and negative reviews and labels
    data_train = data['train']['pos'] + data['train']['neg']
    data_test = data['test']['pos'] + data['test']['neg']
    labels_train = labels['train']['pos'] + labels['train']['neg']
    labels_test = labels['test']['pos'] + labels['test']['neg']

    #Shuffle reviews and corresponding labels within training and test sets
    data_train, labels_train = shuffle(data_train, labels_train)
    data_test, labels_test = shuffle(data_test, labels_test)

    # Return a unified training data, test data, training labels, test labets
    return data_train, data_test, labels_train, labels_test
```

In [6]:
```
train_X, test_X, train_y, test_y = prepare_imdb_data(data, labels)
print("IMDb reviews (combined): train = {}, test = {}".format(len(train_X), len(test_X
```

```
IMDb reviews (combined): train = 25000, test = 25000
```

Now that we have our training and testing sets unified and prepared, we should do a quick check and see an example of the data our model will be trained on. This is generally a good idea as it allows you to see how each of the further processing steps affects the reviews and it also ensures that the data has been loaded correctly.

In [7]:
```
print(train_X[100])
print(train_y[100])
```

```
STAR RATING: ***** Saturday Night **** Friday Night *** Friday Morning ** Sunday Night
* Monday Morning <br /><br />Long time inmate Twitch (Kurupt) gets himself transfered
to a tougher prison than the re-opened Alcatraz. He claims it's to be closer to his la
dy but his real motives are a bit more grandiose. There he crosses paths with Burke (B
ill Goldberg) a bulky prisoner who can take care of himself. Twitch, despite being les
s muscular, is just as mouthy and is pretty much the same. But there is a gang war bre
wing between the black and hispanic inmates that explodes into a hostile takeover of t
he prison when the black's gang leader is shot dead and the finger points at Burke. Bu
t the sh!t really hits the fan when the real killer and leader of the hispanics, Corte
z (Robert Madrid) takes Twitch's girlfriend and Burke's daughter hostage.<br /><br />S
teven Seagal doesn't do sequels (reportedly very opposed to the idea of Under Siege 2
and only agreeing to do it on the condition the film company he was with at the time l
et direct his own movie) so despite this being a DVD sequel, the lead role this time r
ound goes to Bill Golberg (Steve doesn't even appear in some of the stock footage from
the first film that appears towards the end.) But there's a reason he hasn't done much
work since Universal Soldier 2 and that's because he's not much of an actor, and not m
uch of an action star either, managing a character that begins as very dark and broodi
ng but unsubtly turns into a standard action hero awkwardly quipping off dull one-line
rs. Support wise, veterans from the first film, Kurupt and Tony Plana, have merely jum
ped at the chance of extra work.<br /><br />This is a film that's tried to copy the st
yle of the original quite well, with the dim lighting, dark shadows and rap music play
ing over a lot of it. It does this quite well, unfortunately it can't contend with an
unengaging hero, an equally cardboard villain and an apathetic story that the makers d
o very much seem to have made up as they went along. **
0
```

The first step in processing the reviews is to make sure that any html tags that appear should be removed. In addition we wish to tokenize our input, that way words such as *entertained* and *entertaining* are considered the same with regard to sentiment analysis.

In [8]:
```
import nltk
from nltk.corpus import stopwords
from nltk.stem.porter import *

import re
from bs4 import BeautifulSoup

def review_to_words(review):
    nltk.download("stopwords", quiet=True)
    stemmer = PorterStemmer()

    text = BeautifulSoup(review, "html.parser").get_text() # Remove HTML tags
    text = re.sub(r"[^a-zA-Z0-9]", " ", text.lower()) # Convert to lower case
    words = text.split() # Split string into words
    words = [w for w in words if w not in stopwords.words("english")] # Remove stopwo
    words = [PorterStemmer().stem(w) for w in words] # stem

    return words
```

The `review_to_words` method defined above uses `BeautifulSoup` to remove any html tags that appear and uses the `nltk` package to tokenize the reviews. As a check to ensure we know how everything is working, try applying `review_to_words` to one of the reviews in the training set.

In [9]:
```
# TODO:review_to_words(train_X[100]) Apply review_to_words to a review (train_X[100]
review_to_words(train_X[100])
```

Out[9]:
```
['star',
 'rate',
 'saturday',
 'night',
 'friday',
 'night',
 'friday',
 'morn',
 'sunday',
 'night',
 'monday',
 'morn',
 'long',
 'time',
 'inmat',
 'twitch',
 'kurupt',
 'get',
 'transfer',
 'tougher',
 'prison',
 'open',
 'alcatraz',
 'claim',
 'closer',
 'ladi',
 'real',
 'motiv',
 'bit',
 'grandios',
 'cross',
 'path',
 'burk',
 'bill',
 'goldberg',
 'bulki',
 'prison',
 'take',
 'care',
 'twitch',
 'despit',
 'less',
 'muscular',
 'mouthi',
 'pretti',
 'much',
 'gang',
 'war',
 'brew',
 'black',
 'hispan',
 'inmat',
 'explod',
 'hostil',
 'takeov',
 'prison',
 'black',
 'gang',
 'leader',
 'shot',
 'dead',
 'finger',
 'point',
 'burk',
 'sh',
 'realli',
 'hit',
 'fan',
 'real',
 'killer',
 'leader',
 'hispan',
 'cortez',
 'robert',
 'madrid',
 'take',
 'twitch',
 'girlfriend',
 'burk',
 'daughter',
 'hostag',
 'steven',
 'seagal',
 'sequel',
 'reportedli',
 'oppos',
 'idea',
 'siege',
 '2',
 'condit',
 'film',
 'compani',
 'time',
 'let',
 'direct',
 'movi',
 'despit',
 'dvd',
 'sequel',
 'lead',
 'role',
 'time',
 'round',
 'goe',
 'bill',
 'golberg',
 'steve',
 'even',
 'appear',
 'stock',
 'footag',
 'first',
 'film',
 'appear',
 'toward',
 'end',
 'reason',
 'done',
 'much',
 'work',
 'sinc',
 'univers',
 'soldier',
 '2',
 'much',
 'actor',
 'much',
 'action',
 'star',
 'either',
 'manag',
 'charact',
 'begin',
 'dark',
 'brood',
 'unsubtli',
 'turn',
 'standard',
 'action',
 'hero',
 'awkwardli',
 'quip',
 'dull',
 'one',
 'liner',
 'support',
 'wise',
 'veteran',
 'first',
 'film',
 'kurupt',
 'toni',
 'plana',
 'mere',
 'jump',
 'chanc',
 'extra',
 'work',
 'film',
 'tri',
 'copi',
 'style',
 'origin',
 'quit',
 'well',
 'dim',
 'light',
 'dark',
 'shadow',
 'rap',
 'music',
 'play',
 'lot',
 'quit',
 'well',
 'unfortun',
 'contend',
 'unengag',
 'hero',
 'equal',
 'cardboard',
 'villain',
 'apathet',
 'stori',
 'maker',
 'seem',
 'made',
 'along']
```

Question: Above we mentioned that `review_to_words` method removes html formatting and allows us to tokenize the words found in a review, for example, converting *entertained* and *entertaining* into *entertain* so that they are treated as though they are the same word. What else, if anything, does this method do to the input?

Answer:

The review_to_words removes none related words

example : 'name', 'also' ...

The review_to_words removes punctuation marks

for example '!'

The method below applies the `review_to_words` method to each of the reviews in the training and testing datasets. In addition it caches the results. This is because performing this processing step can take a long time. This way if you are unable to complete the notebook in the current session, you can come back without needing to process the data a second time.

In [10]:
```
import pickle

cache_dir = os.path.join("../cache", "sentiment_analysis")  # where to store cache fi
os.makedirs(cache_dir, exist_ok=True)  # ensure cache directory exists

def preprocess_data(data_train, data_test, labels_train, labels_test,
                    cache_dir=cache_dir, cache_file="preprocessed_data.pkl"):
    """Convert each review to words; read from cache if available."""

    # If cache_file is not None, try to read from it first
    cache_data = None
    if cache_file is not None:
        try:
            with open(os.path.join(cache_dir, cache_file), "rb") as f:
                cache_data = pickle.load(f)
            print("Read preprocessed data from cache file:", cache_file)
        except:
            pass  # unable to read from cache, but that's okay

    # If cache is missing, then do the heavy lifting
    if cache_data is None:
        # Preprocess training and test data to obtain words for each review
        #words_train = list(map(review_to_words, data_train))
        #words_test = list(map(review_to_words, data_test))
        words_train = [review_to_words(review) for review in data_train]
        words_test = [review_to_words(review) for review in data_test]

        # Write to cache file for future runs
        if cache_file is not None:
            cache_data = dict(words_train=words_train, words_test=words_test,
                              labels_train=labels_train, labels_test=labels_test)
            with open(os.path.join(cache_dir, cache_file), "wb") as f:
                pickle.dump(cache_data, f)
            print("Wrote preprocessed data to cache file:", cache_file)
    else:
        # Unpack data loaded from cache file
        words_train, words_test, labels_train, labels_test = (cache_data['words_train'],
                cache_data['words_test'], cache_data['labels_train'], cache_data['labe

    return words_train, words_test, labels_train, labels_test
```

In [11]:
```
# Preprocess data
train_X, test_X, train_y, test_y = preprocess_data(train_X, test_X, train_y, test_y)
```

```
Read preprocessed data from cache file: preprocessed_data.pkl
```

Transform the data

In the XGBoost notebook we transformed the data from its word representation to a bag-of-words feature representation. For the model we are going to construct in this notebook we will construct a feature representation which is very similar. To start, we will represent each word as an integer. Of course, some of the words that appear in the reviews occur very infrequently and so likely don't contain much information for the purposes of sentiment analysis. The way we will deal with this problem is that we will fix the size of our working vocabulary and we will only include the words that appear most frequently. We will then combine all of the infrequent words into a single category and, in our case, we will label it as `1`.

Since we will be using a recurrent neural network, it will be convenient if the length of each review is the same. To do this, we will fix a size for our reviews and then pad short reviews with the category 'no word' (which we will label `0`) and truncate long reviews.

(TODO) Create a word dictionary

To begin with, we need to construct a way to map words that appear in the reviews to integers. Here we fix the size of our vocabulary (including the 'no word' and 'infrequent' categories) to be `5000` but you may wish to change this to see how it affects the model.

> **TODO:** Complete the implementation for the `build_dict()` method below. Note that even though the vocab_size is set to `5000`, we only want to construct a mapping for the most frequently appearing `4998` words. This is because we want to reserve the special labels `0` for 'no word' and `1` for 'infrequent word'.

In [12]:
```
import numpy as np

def build_dict(data, vocab_size = 5000):
    """Construct and return a dictionary mapping each of the most frequently appearin

    # TODO: Determine how often each word appears in `data`. Note that `data` is a li
    #       sentence is a list of words.

    word_count = {} # A dict storing the words that appear in the reviews along with

    # TODO: Sort the words found in `data` so that sorted_words[0] is the most freque
    for review in data:
        for word in review:
            if word in word_count:
                word_count[word] += 1
            else:
                word_count[word] = 1

    #       sorted_words[-1] is the least frequently appearing word.
    word_count_sorted = sorted(word_count.items(), key=lambda item: item[1], reverse
    sorted_words = [item[0] for item in word_count_sorted]

    word_dict = {} # This is what we are building, a dictionary that translates words
    for idx, word in enumerate(sorted_words[:vocab_size - 2]): # The -2 is so that we
        word_dict[word] = idx + 2                              # 'infrequent' labels

    return word_dict
```

In [13]:
```
word_dict = build_dict(train_X)
```

In [14]:
```
word_dict
```

Out[14]:
```
{'movi': 2,
 'film': 3,
 'one': 4,
 'like': 5,
 'time': 6,
 'good': 7,
 'make': 8,
 'charact': 9,
 'get': 10,
 'see': 11,
 'watch': 12,
 'stori': 13,
 'even': 14,
 'would': 15,
 'realli': 16,
 'well': 17,
 'scene': 18,
 'look': 19,
 'show': 20,
 'much': 21,
 'end': 22,
 'peopl': 23,
 'bad': 24,
 'go': 25,
 'great': 26,
 'also': 27,
 'first': 28,
 'love': 29,
 'think': 30,
 'way': 31,
 'act': 32,
 'play': 33,
 'made': 34,
 'thing': 35,
 'could': 36,
 'know': 37,
 'say': 38,
 'seem': 39,
 'work': 40,
 'plot': 41,
 'two': 42,
 'actor': 43,
 'year': 44,
 'come': 45,
 'mani': 46,
 'seen': 47,
 'take': 48,
 'life': 49,
 'want': 50,
 'never': 51,
 'littl': 52,
 'best': 53,
 'tri': 54,
 'man': 55,
 'ever': 56,
 'give': 57,
 'better': 58,
 'still': 59,
 'perform': 60,
 'find': 61,
 'feel': 62,
 'part': 63,
 'back': 64,
 'use': 65,
 'someth': 66,
 'director': 67,
 'actual': 68,
 'interest': 69,
 'lot': 70,
 'real': 71,
 'old': 72,
 'cast': 73,
 'though': 74,
 'live': 75,
 'star': 76,
 'enjoy': 77,
 'guy': 78,
 'anoth': 79,
 'new': 80,
 'role': 81,
 'noth': 82,
 '10': 83,
 'funni': 84,
 'music': 85,
 'point': 86,
 'start': 87,
 'set': 88,
 'girl': 89,
 'origin': 90,
 'day': 91,
 'world': 92,
 'everi': 93,
 'believ': 94,
 'turn': 95,
 'quit': 96,
 'direct': 97,
 'us': 98,
 'thought': 99,
 'fact': 100,
 'minut': 101,
 'horror': 102,
 'kill': 103,
 'action': 104,
 'comedi': 105,
 'pretti': 106,
 'young': 107,
 'wonder': 108,
 'happen': 109,
 'around': 110,
 'got': 111,
 'effect': 112,
 'right': 113,
 'long': 114,
 'howev': 115,
 'big': 116,
 'line': 117,
 'famili': 118,
 'enough': 119,
 'seri': 120,
 'may': 121,
 'need': 122,
 'fan': 123,
 'bit': 124,
 'script': 125,
 'beauti': 126,
 'person': 127,
 'becom': 128,
 'without': 129,
 'must': 130,
 'alway': 131,
 'friend': 132,
 'tell': 133,
 'reason': 134,
 'saw': 135,
 'last': 136,
 'final': 137,
 'kid': 138,
 'almost': 139,
 'put': 140,
 'least': 141,
 'sure': 142,
 'done': 143,
 'whole': 144,
 'place': 145,
 'complet': 146,
 'kind': 147,
 'differ': 148,
 'expect': 149,
 'shot': 150,
 'far': 151,
 'mean': 152,
 'anyth': 153,
 'book': 154,
 'laugh': 155,
 'might': 156,
 'name': 157,
 'sinc': 158,
 'begin': 159,
 '2': 160,
 'probabl': 161,
 'woman': 162,
 'help': 163,
 'entertain': 164,
 'let': 165,
 'screen': 166,
 'call': 167,
 'tv': 168,
 'moment': 169,
 'away': 170,
 'read': 171,
 'yet': 172,
 'rather': 173,
 'worst': 174,
 'run': 175,
 'fun': 176,
 'lead': 177,
 'hard': 178,
 'audienc': 179,
 'idea': 180,
 'anyon': 181,
 'episod': 182,
 'american': 183,
 'found': 184,
 'appear': 185,
 'bore': 186,
 'especi': 187,
 'although': 188,
 'hope': 189,
 'cours': 190,
 'keep': 191,
 'anim': 192,
 'job': 193,
 'goe': 194,
 'move': 195,
 'sens': 196,
 'dvd': 197,
 'version': 198,
 'war': 199,
 'money': 200,
 'someon': 201,
 'mind': 202,
 'mayb': 203,
 'problem': 204,
 'true': 205,
 'hous': 206,
 'everyth': 207,
 'nice': 208,
 'second': 209,
 'rate': 210,
 'three': 211,
 'night': 212,
 'face': 213,
 'follow': 214,
 'recommend': 215,
 'main': 216,
 'product': 217,
 'worth': 218,
 'leav': 219,
 'human': 220,
 'special': 221,
 'excel': 222,
 'togeth': 223,
 'wast': 224,
 'sound': 225,
 'everyon': 226,
 'john': 227,
 'hand': 228,
 '1': 229,
 'father': 230,
 'later': 231,
 'eye': 232,
 'said': 233,
 'view': 234,
 'instead': 235,
 'review': 236,
 'boy': 237,
 'high': 238,
 'hour': 239,
 'miss': 240,
 'classic': 241,
 'talk': 242,
 'wife': 243,
 'understand': 244,
 'left': 245,
 'care': 246,
 'black': 247,
 'death': 248,
 'open': 249,
 'murder': 250,
 'write': 251,
 'half': 252,
 'head': 253,
 'rememb': 254,
 'chang': 255,
 'viewer': 256,
 'fight': 257,
 'gener': 258,
 'surpris': 259,
 'short': 260,
 'includ': 261,
 'die': 262,
 'fall': 263,
 'less': 264,
 'els': 265,
 'entir': 266,
 'piec': 267,
 'involv': 268,
 'pictur': 269,
 'simpli': 270,
 'top': 271,
 'power': 272,
 'home': 273,
 'total': 274,
 'usual': 275,
 'budget': 276,
 'attempt': 277,
 'suppos': 278,
 'releas': 279,
 'hollywood': 280,
 'terribl': 281,
 'song': 282,
 'men': 283,
 'possibl': 284,
 'featur': 285,
 'portray': 286,
```

'disappoint': 287,
'3': 288,
'poor': 289,
'coupl': 290,
'camera': 291,
'stupid': 292,
'dead': 293,
'wrong': 294,
'produc': 295,
'low': 296,
'video': 297,
'either': 298,
'aw': 299,
'definit': 300,
'except': 301,
'rest': 302,
'given': 303,
'absolut': 304,
'women': 305,
'lack': 306,
'word': 307,
'writer': 308,
'titl': 309,
'talent': 310,
'decid': 311,
'full': 312,
'perfect': 313,
'along': 314,
'style': 315,
'close': 316,
'truli': 317,
'school': 318,
'save': 319,
'emot': 320,
'age': 321,
'sex': 322,
'next': 323,
'bring': 324,
'mr': 325,
'case': 326,
'killer': 327,
'heart': 328,
'comment': 329,
'sort': 330,
'creat': 331,
'perhap': 332,
'came': 333,
'brother': 334,
'sever': 335,
'joke': 336,
'art': 337,
'dialogu': 338,
'game': 339,
'small': 340,
'base': 341,
'flick': 342,
'written': 343,
'sequenc': 344,
'meet': 345,
'earli': 346,
'often': 347,
'other': 348,
'mother': 349,
'develop': 350,
'humor': 351,
'actress': 352,
'consid': 353,
'dark': 354,
'guess': 355,
'amaz': 356,
'unfortun': 357,
'lost': 358,

'light': 359,
'exampl': 360,
'cinema': 361,
'drama': 362,
'white': 363,
'ye': 364,
'experi': 365,
'imagin': 366,
'mention': 367,
'stop': 368,
'natur': 369,
'forc': 370,
'manag': 371,
'felt': 372,
'present': 373,
'cut': 374,
'children': 375,
'fail': 376,
'son': 377,
'qualiti': 378,
'car': 379,
'support': 380,
'ask': 381,
'hit': 382,
'side': 383,
'voic': 384,
'extrem': 385,
'impress': 386,
'wors': 387,
'evil': 388,
'stand': 389,
'went': 390,
'certainli': 391,
'basic': 392,
'oh': 393,
'overal': 394,
'favorit': 395,
'horribl': 396,
'mysteri': 397,
'number': 398,
'type': 399,
'danc': 400,
'wait': 401,
'hero': 402,
'alreadi': 403,
'5': 404,
'learn': 405,
'matter': 406,
'4': 407,
'michael': 408,
'genr': 409,
'fine': 410,
'despit': 411,
'throughout': 412,
'walk': 413,
'success': 414,
'histori': 415,
'question': 416,
'zombi': 417,
'town': 418,
'relationship': 419,
'realiz': 420,
'child': 421,
'past': 422,
'daughter': 423,
'late': 424,
'b': 425,
'wish': 426,
'credit': 427,
'hate': 428,
'event': 429,
'theme': 430,
'touch': 431,

'citi': 432,
'today': 433,
'sometim': 434,
'behind': 435,
'god': 436,
'twist': 437,
'sit': 438,
'deal': 439,
'stay': 440,
'annoy': 441,
'abl': 442,
'rent': 443,
'pleas': 444,
'edit': 445,
'blood': 446,
'deserv': 447,
'anyway': 448,
'comic': 449,
'appar': 450,
'soon': 451,
'gave': 452,
'etc': 453,
'level': 454,
'slow': 455,
'chanc': 456,
'score': 457,
'bodi': 458,
'brilliant': 459,
'incred': 460,
'figur': 461,
'situat': 462,
'major': 463,
'self': 464,
'stuff': 465,
'decent': 466,
'element': 467,
'dream': 468,
'return': 469,
'obvious': 470,
'order': 471,
'continu': 472,
'pace': 473,
'ridicul': 474,
'happi': 475,
'add': 476,
'highli': 477,
'group': 478,
'thank': 479,
'ladi': 480,
'novel': 481,
'speak': 482,
'pain': 483,
'career': 484,
'shoot': 485,
'strang': 486,
'heard': 487,
'sad': 488,
'polic': 489,
'husband': 490,
'import': 491,
'break': 492,
'took': 493,
'strong': 494,
'cannot': 495,
'robert': 496,
'predict': 497,
'violenc': 498,
'hilari': 499,
'recent': 500,
'countri': 501,
'known': 502,
'particularli': 503,
'pick': 504,

'documentari': 505,
'season': 506,
'critic': 507,
'jame': 508,
'compar': 509,
'obviou': 510,
'alon': 511,
'told': 512,
'state': 513,
'visual': 514,
'rock': 515,
'offer': 516,
'exist': 517,
'theater': 518,
'opinion': 519,
'gore': 520,
'hold': 521,
'crap': 522,
'result': 523,
'realiti': 524,
'hear': 525,
'room': 526,
'clich': 527,
'effort': 528,
'thriller': 529,
'caus': 530,
'explain': 531,
'sequel': 532,
'serious': 533,
'king': 534,
'local': 535,
'ago': 536,
'hell': 537,
'none': 538,
'note': 539,
'allow': 540,
'david': 541,
'sister': 542,
'simpl': 543,
'femal': 544,
'deliv': 545,
'ok': 546,
'convinc': 547,
'class': 548,
'check': 549,
'suspens': 550,
'win': 551,
'oscar': 552,
'buy': 553,
'huge': 554,
'valu': 555,
'sexual': 556,
'scari': 557,
'cool': 558,
'similar': 559,
'excit': 560,
'provid': 561,
'exactli': 562,
'apart': 563,
'shown': 564,
'avoid': 565,
'seriou': 566,
'english': 567,
'taken': 568,
'whose': 569,
'cinematographi': 570,
'shock': 571,
'polit': 572,
'spoiler': 573,
'offic': 574,
'across': 575,
'middl': 576,

'pass': 577,
'street': 578,
'messag': 579,
'somewhat': 580,
'silli': 581,
'charm': 582,
'modern': 583,
'confus': 584,
'filmmak': 585,
'form': 586,
'tale': 587,
'singl': 588,
'jack': 589,
'mostli': 590,
'carri': 591,
'attent': 592,
'william': 593,
'sing': 594,
'subject': 595,
'five': 596,
'prove': 597,
'richard': 598,
'stage': 599,
'team': 600,
'unlik': 601,
'cop': 602,
'georg': 603,
'monster': 604,
'televis': 605,
'earth': 606,
'villain': 607,
'cover': 608,
'pay': 609,
'marri': 610,
'toward': 611,
'build': 612,
'parent': 613,
'pull': 614,
'due': 615,
'fill': 616,
'respect': 617,
'dialog': 618,
'four': 619,
'remind': 620,
'futur': 621,
'typic': 622,
'weak': 623,
'7': 624,
'cheap': 625,
'intellig': 626,
'british': 627,
'atmospher': 628,
'80': 629,
'clearli': 630,
'non': 631,
'paul': 632,
'dog': 633,
'fast': 634,
'artist': 635,
'8': 636,
'knew': 637,
'crime': 638,
'easili': 639,
'escap': 640,
'doubt': 641,
'adult': 642,
'detail': 643,
'date': 644,
'fire': 645,
'romant': 646,
'member': 647,
'drive': 648,

'gun': 649,
'straight': 650,
'fit': 651,
'beyond': 652,
'attack': 653,
'imag': 654,
'upon': 655,
'posit': 656,
'whether': 657,
'fantast': 658,
'peter': 659,
'aspect': 660,
'captur': 661,
'appreci': 662,
'ten': 663,
'plan': 664,
'discov': 665,
'remain': 666,
'period': 667,
'near': 668,
'realist': 669,
'air': 670,
'mark': 671,
'red': 672,
'dull': 673,
'adapt': 674,
'within': 675,
'spend': 676,
'lose': 677,
'materi': 678,
'color': 679,
'chase': 680,
'mari': 681,
'storylin': 682,
'forget': 683,
'bunch': 684,
'clear': 685,
'lee': 686,
'victim': 687,
'nearli': 688,
'box': 689,
'york': 690,
'inspir': 691,
'match': 692,
'mess': 693,
'finish': 694,
'standard': 695,
'easi': 696,
'truth': 697,
'suffer': 698,
'busi': 699,
'bill': 700,
'dramat': 701,
'space': 702,
'western': 703,
'e': 704,
'list': 705,
'battl': 706,
'notic': 707,
'de': 708,
'french': 709,
'ad': 710,
'9': 711,
'tom': 712,
'larg': 713,
'among': 714,
'eventu': 715,
'accept': 716,
'train': 717,
'agre': 718,
'soundtrack': 719,
'spirit': 720,
'third': 721,

'teenag': 722,
'soldier': 723,
'adventur': 724,
'suggest': 725,
'drug': 726,
'famou': 727,
'sorri': 728,
'babi': 729,
'normal': 730,
'cri': 731,
'ultim': 732,
'troubl': 733,
'contain': 734,
'certain': 735,
'cultur': 736,
'romanc': 737,
'rare': 738,
'lame': 739,
'somehow': 740,
'mix': 741,
'disney': 742,
'gone': 743,
'cartoon': 744,
'student': 745,
'fear': 746,
'reveal': 747,
'suck': 748,
'kept': 749,
'attract': 750,
'appeal': 751,
'premis': 752,
'secret': 753,
'greatest': 754,
'design': 755,
'shame': 756,
'throw': 757,
'scare': 758,
'copi': 759,
'wit': 760,
'america': 761,
'admit': 762,
'relat': 763,
'particular': 764,
'brought': 765,
'screenplay': 766,
'whatev': 767,
'pure': 768,
'70': 769,
'averag': 770,
'harri': 771,
'master': 772,
'describ': 773,
'male': 774,
'treat': 775,
'20': 776,
'fantasi': 777,
'issu': 778,
'warn': 779,
'inde': 780,
'background': 781,
'forward': 782,
'free': 783,
'project': 784,
'japanes': 785,
'memor': 786,
'poorli': 787,
'award': 788,
'locat': 789,
'potenti': 790,
'amus': 791,
'struggl': 792,
'weird': 793,
'magic': 794,

'societi': 795,
'okay': 796,
'imdb': 797,
'doctor': 798,
'accent': 799,
'hot': 800,
'water': 801,
'dr': 802,
'30': 803,
'alien': 804,
'express': 805,
'odd': 806,
'crazi': 807,
'choic': 808,
'fiction': 809,
'studio': 810,
'becam': 811,
'control': 812,
'masterpiec': 813,
'difficult': 814,
'fli': 815,
'joe': 816,
'scream': 817,
'costum': 818,
'lover': 819,
'refer': 820,
'uniqu': 821,
'remak': 822,
'girlfriend': 823,
'vampir': 824,
'prison': 825,
'execut': 826,
'wear': 827,
'jump': 828,
'wood': 829,
'unless': 830,
'creepi': 831,
'cheesi': 832,
'superb': 833,
'otherwis': 834,
'parti': 835,
'roll': 836,
'ghost': 837,
'mad': 838,
'public': 839,
'depict': 840,
'week': 841,
'earlier': 842,
'badli': 843,
'moral': 844,
'jane': 845,
'dumb': 846,
'fi': 847,
'flaw': 848,
'grow': 849,
'deep': 850,
'sci': 851,
'cat': 852,
'maker': 853,
'footag': 854,
'older': 855,
'connect': 856,
'plenti': 857,
'bother': 858,
'outsid': 859,
'stick': 860,
'gay': 861,
'catch': 862,
'plu': 863,
'co': 864,
'popular': 865,
'equal': 866,
'social': 867,

'quickli': 868,
'disturb': 869,
'perfectli': 870,
'dress': 871,
'90': 872,
'era': 873,
'mistak': 874,
'lie': 875,
'previou': 876,
'ride': 877,
'combin': 878,
'concept': 879,
'band': 880,
'rich': 881,
'answer': 882,
'surviv': 883,
'front': 884,
'sweet': 885,
'christma': 886,
'insid': 887,
'eat': 888,
'concern': 889,
'bare': 890,
'listen': 891,
'beat': 892,
'ben': 893,
'c': 894,
'serv': 895,
'term': 896,
'meant': 897,
'la': 898,
'german': 899,
'hardli': 900,
'stereotyp': 901,
'law': 902,
'innoc': 903,
'desper': 904,
'memori': 905,
'promis': 906,
'intent': 907,
'cute': 908,
'steal': 909,
'variou': 910,
'inform': 911,
'brain': 912,
'post': 913,
'tone': 914,
'island': 915,
'amount': 916,
'nuditi': 917,
'track': 918,
'compani': 919,
'store': 920,
'claim': 921,
'50': 922,
'flat': 923,
'hair': 924,
'univers': 925,
'land': 926,
'kick': 927,
'danger': 928,
'fairli': 929,
'scott': 930,
'player': 931,
'plain': 932,
'crew': 933,
'step': 934,
'toni': 935,
'share': 936,
'centuri': 937,
'tast': 938,
'engag': 939,
'achiev': 940,

```
 'travel': 941,
 'cold': 942,
 'record': 943,
 'suit': 944,
 'rip': 945,
 'sadli': 946,
 'manner': 947,
 'spot': 948,
 'tension': 949,
 'wrote': 950,
 'intens': 951,
 'fascin': 952,
 'familiar': 953,
 'burn': 954,
 'depth': 955,
 'remark': 956,
 'histor': 957,
 'destroy': 958,
 'sleep': 959,
 'purpos': 960,
 'languag': 961,
 'ruin': 962,
 'ignor': 963,
 'delight': 964,
 'unbeliev': 965,
 'italian': 966,
 'abil': 967,
 'soul': 968,
 'collect': 969,
 'detect': 970,
 'clever': 971,
 'violent': 972,
 'rape': 973,
 'reach': 974,
 'door': 975,
 'trash': 976,
 'scienc': 977,
 'liter': 978,
 'reveng': 979,
 'caught': 980,
 'commun': 981,
 'creatur': 982,
 'trip': 983,
 'approach': 984,
 'fashion': 985,
 'intrigu': 986,
 'skill': 987,
 'paint': 988,
 'introduc': 989,
 'channel': 990,
 'complex': 991,
 'camp': 992,
 'christian': 993,
 'extra': 994,
 'hole': 995,
 'limit': 996,
 'immedi': 997,
 'ann': 998,
 'mental': 999,
 '6': 1000,
 'comput': 1001,
 ...}
```

TODO: Use this space to determine the five most frequently appearing words in the training set.

In [15]:

```
import operator
```

In [16]:

```
sorted_word_dict = sorted(word_dict.items(), key=operator.itemgetter(1))
print('Dictionary in ascending order by value : ',sorted_word_dict)
sorted_word_dict = dict( sorted(word_dict.items(), key=operator.itemgetter(1),reverse=
```

```
Dictionary in ascending order by value :  [('movi', 2), ('film', 3), ('one', 4), ('lik
e', 5), ('time', 6), ('good', 7), ('make', 8), ('charact', 9), ('get', 10), ('see', 1
1), ('watch', 12), ('stori', 13), ('even', 14), ('would', 15), ('realli', 16), ('wel
l', 17), ('scene', 18), ('look', 19), ('show', 20), ('much', 21), ('end', 22), ('peop
l', 23), ('bad', 24), ('go', 25), ('great', 26), ('also', 27), ('first', 28), ('love',
29), ('think', 30), ('way', 31), ('act', 32), ('play', 33), ('made', 34), ('thing', 3
5), ('could', 36), ('know', 37), ('say', 38), ('seem', 39), ('work', 40), ('plot', 4
1), ('two', 42), ('actor', 43), ('year', 44), ('come', 45), ('mani', 46), ('seen', 4
7), ('take', 48), ('life', 49), ('want', 50), ('never', 51), ('littl', 52), ('best', 5
3), ('tri', 54), ('man', 55), ('ever', 56), ('give', 57), ('better', 58), ('still', 5
9), ('perform', 60), ('find', 61), ('feel', 62), ('part', 63), ('back', 64), ('use', 6
5), ('someth', 66), ('director', 67), ('actual', 68), ('interest', 69), ('lot', 70),
('real', 71), ('old', 72), ('cast', 73), ('though', 74), ('live', 75), ('star', 76),
('enjoy', 77), ('guy', 78), ('anoth', 79), ('new', 80), ('role', 81), ('noth', 82),
('10', 83), ('funni', 84), ('music', 85), ('point', 86), ('start', 87), ('set', 88),
('girl', 89), ('origin', 90), ('day', 91), ('world', 92), ('everi', 93), ('believ', 9
4), ('turn', 95), ('quit', 96), ('direct', 97), ('us', 98), ('thought', 99), ('fact',
100), ('minut', 101), ('horror', 102), ('kill', 103), ('action', 104), ('comedi', 10
5), ('pretti', 106), ('young', 107), ('wonder', 108), ('happen', 109), ('around', 11
0), ('got', 111), ('effect', 112), ('right', 113), ('long', 114), ('howev', 115), ('bi
g', 116), ('line', 117), ('famili', 118), ('enough', 119), ('seri', 120), ('may', 12
1), ('need', 122), ('fan', 123), ('bit', 124), ('script', 125), ('beauti', 126), ('per
son', 127), ('becom', 128), ('without', 129), ('must', 130), ('alway', 131), ('frien
d', 132), ('tell', 133), ('reason', 134), ('saw', 135), ('last', 136), ('final', 137),
('kid', 138), ('almost', 139), ('put', 140), ('least', 141), ('sure', 142), ('done', 1
43), ('whole', 144), ('place', 145), ('complet', 146), ('kind', 147), ('differ', 148),
('expect', 149), ('shot', 150), ('far', 151), ('mean', 152), ('anyth', 153), ('book',
154), ('laugh', 155), ('might', 156), ('name', 157), ('sinc', 158), ('begin', 159),
('2', 160), ('probabl', 161), ('woman', 162), ('help', 163), ('entertain', 164), ('le
t', 165), ('screen', 166), ('call', 167), ('tv', 168), ('moment', 169), ('away', 170),
('read', 171), ('yet', 172), ('rather', 173), ('worst', 174), ('run', 175), ('fun', 17
6), ('lead', 177), ('hard', 178), ('audienc', 179), ('idea', 180), ('anyon', 181), ('e
pisod', 182), ('american', 183), ('found', 184), ('appear', 185), ('bore', 186), ('esp
eci', 187), ('although', 188), ('hope', 189), ('cours', 190), ('keep', 191), ('anim',
192), ('job', 193), ('goe', 194), ('move', 195), ('sens', 196), ('dvd', 197), ('versio
n', 198), ('war', 199), ('money', 200), ('someon', 201), ('mind', 202), ('mayb', 203),
('problem', 204), ('true', 205), ('hous', 206), ('everyth', 207), ('nice', 208), ('sec
ond', 209), ('rate', 210), ('three', 211), ('night', 212), ('face', 213), ('follow', 2
14), ('recommend', 215), ('main', 216), ('product', 217), ('worth', 218), ('leav', 21
9), ('human', 220), ('special', 221), ('excel', 222), ('togeth', 223), ('wast', 224),
('sound', 225), ('everyon', 226), ('john', 227), ('hand', 228), ('1', 229), ('father',
230), ('later', 231), ('eye', 232), ('said', 233), ('view', 234), ('instead', 235),
('review', 236), ('boy', 237), ('high', 238), ('hour', 239), ('miss', 240), ('classi
c', 241), ('talk', 242), ('wife', 243), ('understand', 244), ('left', 245), ('care', 2
46), ('black', 247), ('death', 248), ('open', 249), ('murder', 250), ('write', 251),
('half', 252), ('head', 253), ('rememb', 254), ('chang', 255), ('viewer', 256), ('figh
t', 257), ('gener', 258), ('surpris', 259), ('short', 260), ('includ', 261), ('die', 2
62), ('fall', 263), ('less', 264), ('els', 265), ('entir', 266), ('piec', 267), ('invo
lv', 268), ('pictur', 269), ('simpli', 270), ('top', 271), ('power', 272), ('home', 27
3), ('total', 274), ('usual', 275), ('budget', 276), ('attempt', 277), ('suppos', 27
8), ('releas', 279), ('hollywood', 280), ('terribl', 281), ('song', 282), ('men', 28
3), ('possibl', 284), ('featur', 285), ('portray', 286), ('disappoint', 287), ('3', 28
8), ('poor', 289), ('coupl', 290), ('camera', 291), ('stupid', 292), ('dead', 293),
('wrong', 294), ('produc', 295), ('low', 296), ('video', 297), ('either', 298), ('aw',
299), ('definit', 300), ('except', 301), ('rest', 302), ('given', 303), ('absolut', 30
4), ('women', 305), ('lack', 306), ('word', 307), ('writer', 308), ('titl', 309), ('ta
lent', 310), ('decid', 311), ('full', 312), ('perfect', 313), ('along', 314), ('styl
e', 315), ('close', 316), ('truli', 317), ('school', 318), ('save', 319), ('emot', 32
0), ('age', 321), ('sex', 322), ('next', 323), ('bring', 324), ('mr', 325), ('case', 3
26), ('killer', 327), ('heart', 328), ('comment', 329), ('sort', 330), ('creat', 331),
('perhap', 332), ('came', 333), ('brother', 334), ('sever', 335), ('joke', 336), ('ar
t', 337), ('dialogu', 338), ('game', 339), ('small', 340), ('base', 341), ('flick', 34
2), ('written', 343), ('sequenc', 344), ('meet', 345), ('earli', 346), ('often', 347),
('other', 348), ('mother', 349), ('develop', 350), ('humor', 351), ('actress', 352),
('consid', 353), ('dark', 354), ('guess', 355), ('amaz', 356), ('unfortun', 357), ('lo
st', 358), ('light', 359), ('exampl', 360), ('cinema', 361), ('drama', 362), ('white',
363), ('ye', 364), ('experi', 365), ('imagin', 366), ('mention', 367), ('stop', 368),
('natur', 369), ('forc', 370), ('manag', 371), ('felt', 372), ('present', 373), ('cu
t', 374), ('children', 375), ('fail', 376), ('son', 377), ('qualiti', 378), ('car', 37
9), ('support', 380), ('ask', 381), ('hit', 382), ('side', 383), ('voic', 384), ('extr
em', 385), ('impress', 386), ('wors', 387), ('evil', 388), ('stand', 389), ('went', 39
0), ('certainli', 391), ('basic', 392), ('oh', 393), ('overal', 394), ('favorit', 39
```

5), ('horribl', 396), ('mysteri', 397), ('number', 398), ('type', 399), ('danc', 400),
('wait', 401), ('hero', 402), ('alreadi', 403), ('5', 404), ('learn', 405), ('matter',
406), ('4', 407), ('michael', 408), ('genr', 409), ('fine', 410), ('despit', 411), ('t
hroughout', 412), ('walk', 413), ('success', 414), ('histori', 415), ('question', 41
6), ('zombi', 417), ('town', 418), ('relationship', 419), ('realiz', 420), ('child', 4
21), ('past', 422), ('daughter', 423), ('late', 424), ('b', 425), ('wish', 426), ('cre
dit', 427), ('hate', 428), ('event', 429), ('theme', 430), ('touch', 431), ('citi', 43
2), ('today', 433), ('sometim', 434), ('behind', 435), ('god', 436), ('twist', 437),
('sit', 438), ('deal', 439), ('stay', 440), ('annoy', 441), ('abl', 442), ('rent', 44
3), ('pleas', 444), ('edit', 445), ('blood', 446), ('deserv', 447), ('anyway', 448),
('comic', 449), ('appar', 450), ('soon', 451), ('gave', 452), ('etc', 453), ('level',
454), ('slow', 455), ('chanc', 456), ('score', 457), ('bodi', 458), ('brilliant', 45
9), ('incred', 460), ('figur', 461), ('situat', 462), ('major', 463), ('self', 464),
('stuff', 465), ('decent', 466), ('element', 467), ('dream', 468), ('return', 469),
('obvious', 470), ('order', 471), ('continu', 472), ('pace', 473), ('ridicul', 474),
('happi', 475), ('add', 476), ('highli', 477), ('group', 478), ('thank', 479), ('lad
i', 480), ('novel', 481), ('speak', 482), ('pain', 483), ('career', 484), ('shoot', 48
5), ('strang', 486), ('heard', 487), ('sad', 488), ('polic', 489), ('husband', 490),
('import', 491), ('break', 492), ('took', 493), ('strong', 494), ('cannot', 495), ('ro
bert', 496), ('predict', 497), ('violenc', 498), ('hilari', 499), ('recent', 500), ('c
ountri', 501), ('known', 502), ('particularli', 503), ('pick', 504), ('documentari', 5
05), ('season', 506), ('critic', 507), ('jame', 508), ('compar', 509), ('obviou', 51
0), ('alon', 511), ('told', 512), ('state', 513), ('visual', 514), ('rock', 515), ('of
fer', 516), ('exist', 517), ('theater', 518), ('opinion', 519), ('gore', 520), ('hol
d', 521), ('crap', 522), ('result', 523), ('realiti', 524), ('hear', 525), ('room', 52
6), ('clich', 527), ('effort', 528), ('thriller', 529), ('caus', 530), ('explain', 53
1), ('sequel', 532), ('serious', 533), ('king', 534), ('local', 535), ('ago', 536),
('hell', 537), ('none', 538), ('note', 539), ('allow', 540), ('david', 541), ('siste
r', 542), ('simpl', 543), ('femal', 544), ('deliv', 545), ('ok', 546), ('convinc', 54
7), ('class', 548), ('check', 549), ('suspens', 550), ('win', 551), ('oscar', 552),
('buy', 553), ('huge', 554), ('valu', 555), ('sexual', 556), ('scari', 557), ('cool',
558), ('similar', 559), ('excit', 560), ('provid', 561), ('exactli', 562), ('apart', 5
63), ('shown', 564), ('avoid', 565), ('seriou', 566), ('english', 567), ('taken', 56
8), ('whose', 569), ('cinematographi', 570), ('shock', 571), ('polit', 572), ('spoile
r', 573), ('offic', 574), ('across', 575), ('middl', 576), ('pass', 577), ('street', 5
78), ('messag', 579), ('somewhat', 580), ('silli', 581), ('charm', 582), ('modern', 58
3), ('confus', 584), ('filmmak', 585), ('form', 586), ('tale', 587), ('singl', 588),
('jack', 589), ('mostli', 590), ('carri', 591), ('attent', 592), ('william', 593), ('s
ing', 594), ('subject', 595), ('five', 596), ('prove', 597), ('richard', 598), ('stag
e', 599), ('team', 600), ('unlik', 601), ('cop', 602), ('georg', 603), ('monster', 60
4), ('televis', 605), ('earth', 606), ('villain', 607), ('cover', 608), ('pay', 609),
('marri', 610), ('toward', 611), ('build', 612), ('parent', 613), ('pull', 614), ('du
e', 615), ('fill', 616), ('respect', 617), ('dialog', 618), ('four', 619), ('remind',
620), ('futur', 621), ('typic', 622), ('weak', 623), ('7', 624), ('cheap', 625), ('int
ellig', 626), ('british', 627), ('atmospher', 628), ('80', 629), ('clearli', 630), ('n
on', 631), ('paul', 632), ('dog', 633), ('fast', 634), ('artist', 635), ('8', 636),
('knew', 637), ('crime', 638), ('easili', 639), ('escap', 640), ('doubt', 641), ('adul
t', 642), ('detail', 643), ('date', 644), ('fire', 645), ('romant', 646), ('member', 6
47), ('drive', 648), ('gun', 649), ('straight', 650), ('fit', 651), ('beyond', 652),
('attack', 653), ('imag', 654), ('upon', 655), ('posit', 656), ('whether', 657), ('fan
tast', 658), ('peter', 659), ('aspect', 660), ('captur', 661), ('appreci', 662), ('te
n', 663), ('plan', 664), ('discov', 665), ('remain', 666), ('period', 667), ('near', 6
68), ('realist', 669), ('air', 670), ('mark', 671), ('red', 672), ('dull', 673), ('ada
pt', 674), ('within', 675), ('spend', 676), ('lose', 677), ('materi', 678), ('color',
679), ('chase', 680), ('mari', 681), ('storylin', 682), ('forget', 683), ('bunch', 68
4), ('clear', 685), ('lee', 686), ('victim', 687), ('nearli', 688), ('box', 689), ('yo
rk', 690), ('inspir', 691), ('match', 692), ('mess', 693), ('finish', 694), ('standar
d', 695), ('easi', 696), ('truth', 697), ('suffer', 698), ('busi', 699), ('bill', 70
0), ('dramat', 701), ('space', 702), ('western', 703), ('e', 704), ('list', 705), ('ba
ttl', 706), ('notic', 707), ('de', 708), ('french', 709), ('ad', 710), ('9', 711), ('t
om', 712), ('larg', 713), ('among', 714), ('eventu', 715), ('accept', 716), ('train',
717), ('agre', 718), ('soundtrack', 719), ('spirit', 720), ('third', 721), ('teenag',
722), ('soldier', 723), ('adventur', 724), ('suggest', 725), ('drug', 726), ('famou',
727), ('sorri', 728), ('babi', 729), ('normal', 730), ('cri', 731), ('ultim', 732),
('troubl', 733), ('contain', 734), ('certain', 735), ('cultur', 736), ('romanc', 737),
('rare', 738), ('lame', 739), ('somehow', 740), ('mix', 741), ('disney', 742), ('gon
e', 743), ('cartoon', 744), ('student', 745), ('fear', 746), ('reveal', 747), ('suck',
748), ('kept', 749), ('attract', 750), ('appeal', 751), ('premis', 752), ('secret', 75
3), ('greatest', 754), ('design', 755), ('shame', 756), ('throw', 757), ('scare', 75
8), ('copi', 759), ('wit', 760), ('america', 761), ('admit', 762), ('relat', 763), ('p
articular', 764), ('brought', 765), ('screenplay', 766), ('whatev', 767), ('pure', 76
8), ('70', 769), ('averag', 770), ('harri', 771), ('master', 772), ('describ', 773),
('male', 774), ('treat', 775), ('20', 776), ('fantasi', 777), ('issu', 778), ('warn',

779), ('inde', 780), ('background', 781), ('forward', 782), ('free', 783), ('project',
784), ('japanes', 785), ('memor', 786), ('poorli', 787), ('award', 788), ('locat', 78
9), ('potenti', 790), ('amus', 791), ('struggl', 792), ('weird', 793), ('magic', 794),
('societi', 795), ('okay', 796), ('imdb', 797), ('doctor', 798), ('accent', 799), ('ho
t', 800), ('water', 801), ('dr', 802), ('30', 803), ('alien', 804), ('express', 805),
('odd', 806), ('crazi', 807), ('choic', 808), ('fiction', 809), ('studio', 810), ('bec
am', 811), ('control', 812), ('masterpiec', 813), ('difficult', 814), ('fli', 815),
('joe', 816), ('scream', 817), ('costum', 818), ('lover', 819), ('refer', 820), ('uniq
u', 821), ('remak', 822), ('girlfriend', 823), ('vampir', 824), ('prison', 825), ('exe
cut', 826), ('wear', 827), ('jump', 828), ('wood', 829), ('unless', 830), ('creepi', 8
31), ('cheesi', 832), ('superb', 833), ('otherwis', 834), ('parti', 835), ('roll', 83
6), ('ghost', 837), ('mad', 838), ('public', 839), ('depict', 840), ('week', 841), ('e
arlier', 842), ('badli', 843), ('moral', 844), ('jane', 845), ('dumb', 846), ('fi', 84
7), ('flaw', 848), ('grow', 849), ('deep', 850), ('sci', 851), ('cat', 852), ('maker',
853), ('footag', 854), ('older', 855), ('connect', 856), ('plenti', 857), ('bother', 8
58), ('outsid', 859), ('stick', 860), ('gay', 861), ('catch', 862), ('plu', 863), ('c
o', 864), ('popular', 865), ('equal', 866), ('social', 867), ('quickli', 868), ('distu
rb', 869), ('perfectli', 870), ('dress', 871), ('90', 872), ('era', 873), ('mistak', 8
74), ('lie', 875), ('previou', 876), ('ride', 877), ('combin', 878), ('concept', 879),
('band', 880), ('rich', 881), ('answer', 882), ('surviv', 883), ('front', 884), ('swee
t', 885), ('christma', 886), ('insid', 887), ('eat', 888), ('concern', 889), ('bare',
890), ('listen', 891), ('beat', 892), ('ben', 893), ('c', 894), ('serv', 895), ('ter
m', 896), ('meant', 897), ('la', 898), ('german', 899), ('hardli', 900), ('stereotyp',
901), ('law', 902), ('innoc', 903), ('desper', 904), ('memori', 905), ('promis', 906),
('intent', 907), ('cute', 908), ('steal', 909), ('variou', 910), ('inform', 911), ('br
ain', 912), ('post', 913), ('tone', 914), ('island', 915), ('amount', 916), ('nuditi',
917), ('track', 918), ('compani', 919), ('store', 920), ('claim', 921), ('50', 922),
('flat', 923), ('hair', 924), ('univers', 925), ('land', 926), ('kick', 927), ('dange
r', 928), ('fairli', 929), ('scott', 930), ('player', 931), ('plain', 932), ('crew', 9
33), ('step', 934), ('toni', 935), ('share', 936), ('centuri', 937), ('tast', 938),
('engag', 939), ('achiev', 940), ('travel', 941), ('cold', 942), ('record', 943), ('su
it', 944), ('rip', 945), ('sadli', 946), ('manner', 947), ('spot', 948), ('tension', 9
49), ('wrote', 950), ('intens', 951), ('fascin', 952), ('familiar', 953), ('burn', 95
4), ('depth', 955), ('remark', 956), ('histor', 957), ('destroy', 958), ('sleep', 95
9), ('purpos', 960), ('languag', 961), ('ruin', 962), ('ignor', 963), ('delight', 96
4), ('unbeliev', 965), ('italian', 966), ('abil', 967), ('soul', 968), ('collect', 96
9), ('detect', 970), ('clever', 971), ('violent', 972), ('rape', 973), ('reach', 974),
('door', 975), ('trash', 976), ('scienc', 977), ('liter', 978), ('reveng', 979), ('cau
ght', 980), ('commun', 981), ('creatur', 982), ('trip', 983), ('approach', 984), ('fas
hion', 985), ('intrigu', 986), ('skill', 987), ('paint', 988), ('introduc', 989), ('ch
annel', 990), ('complex', 991), ('camp', 992), ('christian', 993), ('extra', 994), ('h
ole', 995), ('limit', 996), ('immedi', 997), ('ann', 998), ('mental', 999), ('6', 100
0), ('comput', 1001), ('mere', 1002), ('million', 1003), ('slightli', 1004), ('slashe
r', 1005), ('conclus', 1006), ('suddenli', 1007), ('imposs', 1008), ('teen', 1009),
('crimin', 1010), ('neither', 1011), ('nation', 1012), ('spent', 1013), ('physic', 101
4), ('respons', 1015), ('planet', 1016), ('fake', 1017), ('receiv', 1018), ('sick', 10
19), ('blue', 1020), ('bizarr', 1021), ('embarrass', 1022), ('indian', 1023), ('15', 1
024), ('ring', 1025), ('pop', 1026), ('drop', 1027), ('drag', 1028), ('haunt', 1029),
('pointless', 1030), ('suspect', 1031), ('search', 1032), ('edg', 1033), ('handl', 103
4), ('biggest', 1035), ('common', 1036), ('hurt', 1037), ('arriv', 1038), ('faith', 10
39), ('technic', 1040), ('angel', 1041), ('dad', 1042), ('genuin', 1043), ('solid', 10
44), ('awesom', 1045), ('f', 1046), ('former', 1047), ('colleg', 1048), ('van', 1049),
('focu', 1050), ('count', 1051), ('tear', 1052), ('heavi', 1053), ('wall', 1054), ('ra
is', 1055), ('visit', 1056), ('laughabl', 1057), ('younger', 1058), ('fair', 1059),
('excus', 1060), ('sign', 1061), ('cult', 1062), ('tough', 1063), ('motion', 1064),
('key', 1065), ('super', 1066), ('desir', 1067), ('addit', 1068), ('stun', 1069), ('cl
oth', 1070), ('exploit', 1071), ('tortur', 1072), ('smith', 1073), ('davi', 1074), ('r
ace', 1075), ('author', 1076), ('cross', 1077), ('jim', 1078), ('compel', 1079), ('con
sist', 1080), ('focus', 1081), ('minor', 1082), ('commit', 1083), ('chemistri', 1084),
('pathet', 1085), ('park', 1086), ('tradit', 1087), ('obsess', 1088), ('frank', 1089),
('grade', 1090), ('asid', 1091), ('60', 1092), ('brutal', 1093), ('steve', 1094), ('so
mewher', 1095), ('grant', 1096), ('depress', 1097), ('opportun', 1098), ('rule', 109
9), ('explor', 1100), ('u', 1101), ('honest', 1102), ('besid', 1103), ('anti', 1104),
('dub', 1105), ('intend', 1106), ('trailer', 1107), ('bar', 1108), ('regard', 1109),
('scientist', 1110), ('west', 1111), ('longer', 1112), ('judg', 1113), ('decad', 111
4), ('silent', 1115), ('creativ', 1116), ('armi', 1117), ('wild', 1118), ('stewart', 1
119), ('south', 1120), ('g', 1121), ('draw', 1122), ('road', 1123), ('govern', 1124),
('boss', 1125), ('ex', 1126), ('practic', 1127), ('surprisingli', 1128), ('motiv', 112
9), ('club', 1130), ('gang', 1131), ('festiv', 1132), ('green', 1133), ('page', 1134),
('redeem', 1135), ('london', 1136), ('militari', 1137), ('idiot', 1138), ('display', 1
139), ('aliv', 1140), ('machin', 1141), ('thrill', 1142), ('repeat', 1143), ('yeah', 1
144), ('nobodi', 1145), ('folk', 1146), ('100', 1147), ('40', 1148), ('journey', 114
9), ('garbag', 1150), ('tire', 1151), ('ground', 1152), ('smile', 1153), ('bought', 11
54), ('mood', 1155), ('cost', 1156), ('stone', 1157), ('sam', 1158), ('mouth', 1159),
('noir', 1160), ('agent', 1161), ('terrif', 1162), ('utterli', 1163), ('requir', 116
4), ('area', 1165), ('honestli', 1166), ('sexi', 1167), ('geniu', 1168), ('report', 11
69), ('enter', 1170), ('humour', 1171), ('investig', 1172), ('glad', 1173), ('serial',
1174), ('narr', 1175), ('occasion', 1176), ('passion', 1177), ('marriag', 1178), ('cli
max', 1179), ('studi', 1180), ('industri', 1181), ('demon', 1182), ('nowher', 1183),
('center', 1184), ('charli', 1185), ('ship', 1186), ('hors', 1187), ('loos', 1188),
('bear', 1189), ('hang', 1190), ('wow', 1191), ('graphic', 1192), ('admir', 1193), ('g
iant', 1194), ('send', 1195), ('loud', 1196), ('damn', 1197), ('profession', 1198),
('rel', 1199), ('subtl', 1200), ('nake', 1201), ('blow', 1202), ('bottom', 1203), ('ba
tman', 1204), ('insult', 1205), ('doubl', 1206), ('boyfriend', 1207), ('kelli', 1208),
('r', 1209), ('initi', 1210), ('frame', 1211), ('gem', 1212), ('opera', 1213), ('draw
n', 1214), ('cinemat', 1215), ('affect', 1216), ('challeng', 1217), ('church', 1218),
('j', 1219), ('seek', 1220), ('nightmar', 1221), ('l', 1222), ('fulli', 1223), ('evi

[illegible]
4760), ('http', 4761), ('articl', 4762), ('foil', 4763), ('mason', 4764), ('flavor', 4765), ('alexandr', 4766), ('feast', 4767), ('breakfast', 4768), ('nope', 4769), ('lucil', 4770), ('trier', 4771), ('toronto', 4772), ('sicken', 4773), ('milo', 4774), ('redneck', 4775), ('wrestler', 4776), ('retriev', 4777), ('ceremoni', 4778), ('swept', 4779), ('leather', 4780), ('huston', 4781), ('deem', 4782), ('highway', 4783), ('cartoonish', 4784), ('knightley', 4785), ('shorter', 4786), ('turd', 4787), ('styliz', 4788), ('clinic', 4789), ('disregard', 4790), ('abund', 4791), ('dudley', 4792), ('gilbert', 4793), ('outdat', 4794), ('howl', 4795), ('ideolog', 4796), ('psych', 4797), ('comprehend', 4798), ('peer', 4799), ('linear', 4800), ('audrey', 4801), ('charlton', 4802), ('glare', 4803), ('oppon', 4804), ('feminin', 4805), ('boyer', 4806), ('smack', 4807), ('gunga', 4808), ('magician', 4809), ('tack', 4810), ('frog', 4811), ('fifth', 4812), ('affleck', 4813), ('nightclub', 4814), ('client', 4815), ('cemeteri', 4816), ('compris', 4817), ('plate', 4818), ('uniformli', 4819), ('spine', 4820), ('cuban', 4821), ('senat', 4822), ('salman', 4823), ('lavish', 4824), ('monument', 4825), ('sleaz', 4826), ('durat', 4827), ('goldsworthi', 4828), ('btw', 4829), ('potter', 4830), ('collector', 4831), ('corn', 4832), ('bastard', 4833), ('newer', 4834), ('energet', 4835), ('tara', 4836), ('deliver', 4837), ('1994', 4838), ('spawn', 4839), ('1991', 4840), ('4th', 4841), ('bogu', 4842), ('braveheart', 4843), ('lighter', 4844), ('breakdown', 4845), ('boo', 4846), ('phenomen', 4847), ('snatch', 4848), ('lifeless', 4849), ('greet', 4850), ('moe', 4851), ('einstein', 4852), ('summar', 4853), ('conrad', 4854), ('chip', 4855), ('evolut', 4856), ('whack', 4857), ('toe', 4858), ('mitch', 4859), ('spectacl', 4860), ('jam', 4861), ('pronounc', 4862), ('neatli', 4863), ('fluid', 4864), ('judd', 4865), ('creek', 4866), ('jare', 4867), ('outright', 4868), ('healthi', 4869), ('undead', 4870), ('capot', 4871), ('bori', 4872), ('wtf', 4873), ('eleven', 4874), ('gilliam', 4875), ('mcqueen', 4876), ('historian', 4877), ('ie', 4878), ('kazan', 4879), ('luxuri', 4880), ('replay', 4881), ('appl', 4882), ('lex', 4883), ('1977', 4884), ('jule', 4885), ('cecil', 4886), ('alleg', 4887), ('ol', 4888), ('firmli', 4889), ('inaccuraci', 4890), ('clara', 4891), ('randolph', 4892), ('undertak', 4893), ('signal', 4894), ('1974', 4895), ('jedi', 4896), ('sorrow', 4897), ('bulk', 4898), ('trauma', 4899), ('belli', 4900), ('constitut', 4901), ('nina', 4902), ('occup', 4903), ('evelyn', 4904), ('kent', 4905), ('embark', 4906), ('liu', 4907), ('armstrong', 4908), ('pepper', 4909), ('decapit', 4910), ('miami', 4911), ('antonio', 4912), ('inaccur', 4913), ('paula', 4914), ('miniseri', 4915), ('goldblum', 4916), ('unsuspect', 4917), ('meal', 4918), ('vain', 4919), ('truman', 4920), ('sacrif', 4921), ('roar', 4922), ('vignett', 4923), ('carmen', 4924), ('sidewalk', 4925), ('blur', 4926), ('propheci', 4927), ('palm', 4928), ('cape', 4929), ('mum', 4930), ('lauren', 4931), ('curtain', 4932), ('walt', 4933), ('galaxi', 4934), ('id', 4935), ('fruit', 4936), ('forgiven', 4937), ('ash', 4938), ('1985', 4939), ('groan', 4940), ('unattract', 4941), ('lanc', 4942), ('pioneer', 4943), ('subtli', 4944), ('kiddi', 4945), ('genet', 4946), ('poker', 4947), ('aussi', 4948), ('congratul', 4949), ('abound', 4950), ('tokyo', 4951), ('comb', 4952), ('inclus', 4953), ('knee', 4954), ('basket', 4955), ('bsg', 4956), ('relentless', 4957), ('porter', 4958), ('bait', 4959), ('conan', 4960), ('comprehens', 4961), ('spray', 4962), ('rosario', 4963), ('reincarn', 4964), ('victorian', 4965), ('mice', 4966), ('bravo', 4967), ('spill', 4968), ('dubiou', 4969), ('jill', 4970), ('sophi', 4971), ('omin', 4972), ('vastli', 4973), ('asset', 4974), ('sparkl', 4975), ('assort', 4976), ('scarfac', 4977), ('weaker', 4978), ('epitom', 4979), ('masterson', 4980), ('ghetto', 4981), ('substanti', 4982), ('playboy', 4983), ('optimist', 4984), ('casper', 4985), ('detach', 4986), ('ingrid', 4987), ('growth', 4988), ('evan', 4989), ('profil', 4990), ('macabr', 4991), ('scariest', 4992), ('bach', 4993), ('rapidli', 4994), ('verg', 4995), ('frontal', 4996), ('modest', 4997), ('orchestr', 4998), ('motorcycl', 4999)]

https://www.w3resource.com/python-exercises/dictionary/python-data-type-dictionary-exercise-1.php

Question: What are the five most frequently appearing (tokenized) words in the training set? Does it makes sense that these words appear frequently in the training set?

In [17]:
```
list(word_dict.keys())[:5]
```

Out[17]: ['movi', 'film', 'one', 'like', 'time']

Answer:

yes it is a common words

- movi
- film
- one
- like
- time

Save `word_dict`

Later on when we construct an endpoint which processes a submitted review we will need to make use of the `word_dict` which we have created. As such, we will save it to a file now for future use.

In [17]:
```
data_dir = '../data/pytorch' # The folder we will use for storing data
if not os.path.exists(data_dir): # Make sure that the folder exists
    os.makedirs(data_dir)
```

In [18]:
```
with open(os.path.join(data_dir, 'word_dict.pkl'), "wb") as f:
    pickle.dump(word_dict, f)
```

Transform the reviews

Now that we have our word dictionary which allows us to transform the words appearing in the reviews into integers, it is time to make use of it and convert our reviews to their integer sequence representation, making sure to pad or truncate to a fixed length, which in our case is `500`.

In [19]:
```
def convert_and_pad(word_dict, sentence, pad=500):
    NOWORD = 0 # We will use 0 to represent the 'no word' category
    INFREQ = 1 # and we use 1 to represent the infrequent words, i.e., words not appe

    working_sentence = [NOWORD] * pad

    for word_index, word in enumerate(sentence[:pad]):
        if word in word_dict:
            working_sentence[word_index] = word_dict[word]
        else:
            working_sentence[word_index] = INFREQ

    return working_sentence, min(len(sentence), pad)

def convert_and_pad_data(word_dict, data, pad=500):
    result = []
    lengths = []

    for sentence in data:
        converted, leng = convert_and_pad(word_dict, sentence, pad)
        result.append(converted)
        lengths.append(leng)

    return np.array(result), np.array(lengths)
```

In [20]:
```
train_X, train_X_len = convert_and_pad_data(word_dict, train_X)
test_X, test_X_len = convert_and_pad_data(word_dict, test_X)
```

As a quick check to make sure that things are working as intended, check to see what one of the reviews in the training set looks like after having been processeed. Does this look reasonable? What is the length of a review in the training set?

In [21]:
```
# Use this cell to examine one of the processed reviews to make sure everything is wo
train_X[100]
```

Out[21]:
```
array([ 347,   11,    2,    1,   35,   25,  572,   59,  215,  432, 1903,
         81, 1290,   33, 1347, 2165,   76,   80,  690, 3542,  441,  485,
        237,   95,   82,   16,  112,    2,    9, 2192,   15,  148,  181,
        268,  572, 4637,  406,   21, 1341,  303, 3199,  121,  642,   12,
          2,  244,  157, 1816,  360,   66,  559,    2,  286,   86,    1,
        386,  273,    2, 1052,  180,  391,    4, 2325,   11,   27,   76,
        227, 2844, 4220, 3318, 1446,    1, 1802,    1,  542,    1,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0])
```

Question: In the cells above we use the `preprocess_data` and `convert_and_pad_data` methods to process both the training and testing set. Why or why not might this be a problem?

Answer:

the method we use to replace no word and infrequent words with 0 and 1 that make the review as abinarry input no matter the word is

Step 3: Upload the data to S3

As in the XGBoost notebook, we will need to upload the training dataset to S3 in order for our training code to access it. For now we will save it locally and we will upload to S3 later on.

Save the processed training dataset locally

It is important to note the format of the data that we are saving as we will need to know it when we write the training code. In our case, each row of the dataset has the form `label`, `length`, `review[500]` where `review[500]` is a sequence of `500` integers representing the words in the review.

In [22]:
```
import pandas as pd

pd.concat([pd.DataFrame(train_y), pd.DataFrame(train_X_len), pd.DataFrame(train_X)], a
        .to_csv(os.path.join(data_dir, 'train.csv'), header=False, index=False)
```

Uploading the training data

Next, we need to upload the training data to the SageMaker default S3 bucket so that we can provide access to it while training our model.

Part B: Create a Lambda function

Now it is time to actually create the Lambda function.

Using the AWS Console, navigate to the AWS Lambda page and click on **Create a function**. When you get to the next page, make sure that **Author from scratch** is selected. Now, name your Lambda function, using a name that you will remember later on, for example `sentiment_analysis_func`. Make sure that the **Python 3.6** runtime is selected and then choose the role that you created in the previous part. Then, click on **Create Function**.

On the next page you will see some information about the Lambda function you've just created. If you scroll down you should see an editor in which you can write the code that will be executed when your Lambda function is triggered. In our example, we will use the code below.

```python
# We need to use the low-level library to interact with SageMaker since the SageMaker API
# is not available natively through Lambda.
import boto3

def lambda_handler(event, context):

    # The SageMaker runtime is what allows us to invoke the endpoint that we've created.
    runtime = boto3.Session().client('sagemaker-runtime')

    # Now we use the SageMaker runtime to invoke our endpoint, sending the review we were given
    response = runtime.invoke_endpoint(EndpointName = '**ENDPOINT NAME HERE**',    # The name of the endpoint we created
                                       ContentType = 'text/plain',                 # The data format that is expected
                                       Body = event['body'])                       # The actual review

    # The response is an HTTP response whose body contains the result of our inference
    result = response['Body'].read().decode('utf-8')

    return {
        'statusCode' : 200,
        'headers' : { 'Content-Type' : 'text/plain', 'Access-Control-Allow-Origin' : '*' },
        'body' : result
    }
```

Once you have copy and pasted the code above into the Lambda code editor, replace the `**ENDPOINT NAME HERE**` portion with the name of the endpoint that we deployed earlier. You can determine the name of the endpoint using the code cell below.

```
In [45]: predictor.endpoint
```

```
Out[45]: 'sagemaker-pytorch-2021-10-14-23-04-35-370'
```

Once you have added the endpoint name to the Lambda function, click on **Save**. Your Lambda function is now up and running. Next we need to create a way for our web app to execute the Lambda function.

Setting up API Gateway

Now that our Lambda function is set up, it is time to create a new API using API Gateway that will trigger the Lambda function we have just created.

Using AWS Console, navigate to **Amazon API Gateway** and then click on **Get started**.

On the next page, make sure that **New API** is selected and give the new api a name, for example, `sentiment_analysis_api`. Then, click on **Create API**.

Now we have created an API, however it doesn't currently do anything. What we want it to do is to trigger the Lambda function that we created earlier.

Select the **Actions** dropdown menu and click **Create Method**. A new blank method will be created, select its dropdown menu and select **POST**, then click on the check mark beside it.

For the integration point, make sure that **Lambda Function** is selected and click on the **Use Lambda Proxy integration**. This option makes sure that the data that is sent to the API is then sent directly to the Lambda function with no processing. It also means that the return value must be a proper response object as it will also not be processed by API Gateway.

Type the name of the Lambda function you created earlier into the **Lambda Function** text entry box and then click on **Save**. Click on **OK** in the pop-up box that then appears, giving permission to API Gateway to invoke the Lambda function you created.

The last step in creating the API Gateway is to select the **Actions** dropdown and click on **Deploy API**. You will need to create a new Deployment stage and name it anything you like, for example `prod`.

You have now successfully set up a public API to access your SageMaker model. Make sure to copy or write down the URL provided to invoke your newly created public API as this will be needed in the next step. This URL can be found at the top of the page, highlighted in blue next to the text **Invoke URL**.

Step 4: Deploying our web app

Now that we have a publicly available API, we can start using it in a web app. For our purposes, we have provided a simple static html file which can make use of the public api you created earlier.

In the `website` folder there should be a file called `index.html`. Download the file to your computer and open that file up in a text editor of your choice. There should be a line which contains **REPLACE WITH PUBLIC API URL**. Replace this string with the url that you wrote down in the last step and then save the file.

Now, if you open `index.html` on your local computer, your browser will behave as a local web server and you can use the provided site to interact with your SageMaker model.

If you'd like to go further, you can host this html file anywhere you'd like, for example using github or hosting a static site on Amazon's S3. Once you have done this you can share the link with anyone you'd like and have them play with it too!

> **Important Note** In order for the web app to communicate with the SageMaker endpoint, the endpoint has to actually be deployed and running. This means that you are paying for it. Make sure that the endpoint is running when you want to use the web app but that you shut it down when you don't need it, otherwise you will end up with a surprisingly large AWS bill.

TODO: Make sure that you include the edited `index.html` file in your project submission.

Now that your web app is working, trying playing around with it and see how well it works.

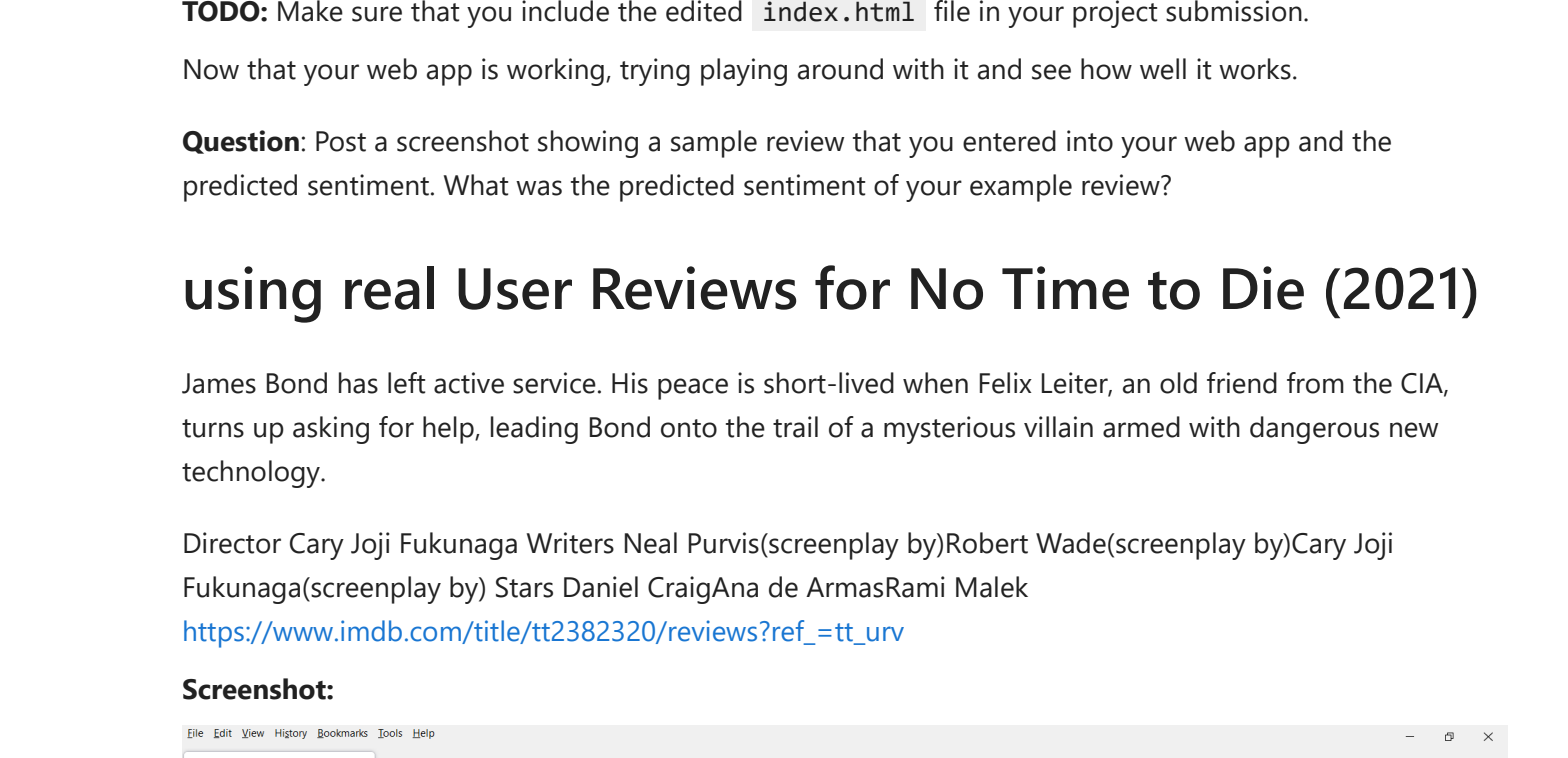
Question: Post a screenshot showing a sample review that you entered into your web app and the predicted sentiment. What was the predicted sentiment of your example review?

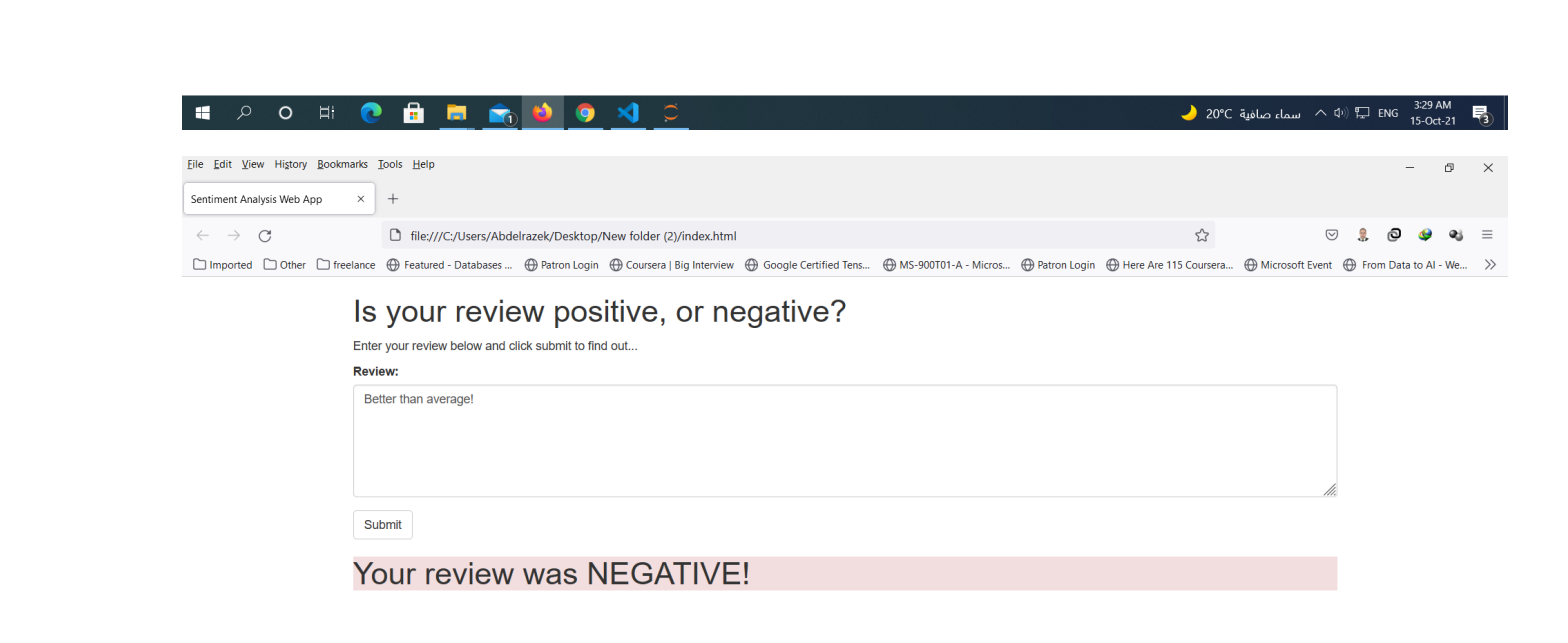
using real User Reviews for No Time to Die (2021)

James Bond has left active service. His peace is short-lived when Felix Leiter, an old friend from the CIA, turns up asking for help, leading Bond onto the trail of a mysterious villain armed with dangerous new technology.

Director Cary Joji Fukunaga Writers Neal Purvis(screenplay by)Robert Wade(screenplay by)Cary Joji Fukunaga(screenplay by) Stars Daniel CraigAna de ArmasRami Malek
https://www.imdb.com/title/tt2382320/reviews?ref_=tt_urv

Screenshot:





Answer:

- the positive predict match with the user review

  "I enjoyed every second of it Gordon-111 October 2021 The story is really engaging. It didn't feel like 2.5 hours! I enjoyed every second of it. I like the bond between Daniel Craig and Lea Seydoux. The ending got me tearful. Who would have thought James Bond would make someone cry? 174 out of 327 found this helpful. Was this review helpful? Sign in to vote. Permalink"

- the negative predict doese not match with user review

"Better than average! mm-3914 October 2021 Warning: Spoilers 0 out of 1 found this helpful. Was this review helpful? Sign in to vote. Permalink"

Delete the endpoint

Remember to always shut down your endpoint if you are no longer using it. You are charged for the length of time that the endpoint is running so if you forget and leave it on you could end up with an unexpectedly large bill.

```
In [ ]: predictor.delete_endpoint()
```

```
In [ ]:
```

```
In [ ]:
```