5.4 Natural Language Processing with Python

Overview

In this tutorial you will be using a small dataset consisting of customer reviews of Video games that are available in the Amazon web store.

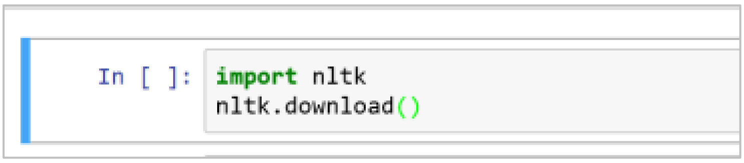
- Your task is to identify named entities occurring in Video Game reviews and to analyse the sentiment of the reviews
- The dataset is in a tab separated format where each line corresponds to a different Video game review
- Moreover, the tab separated file consists of three columns:
    - The first column corresponds to the review id
    - the second column to the movie id
    - the third column to the actual movie review

Topics Covered:

- Installing NER in NLTK
- Loading dataset using panda
- Sentiment analysis
- Named entity recognition

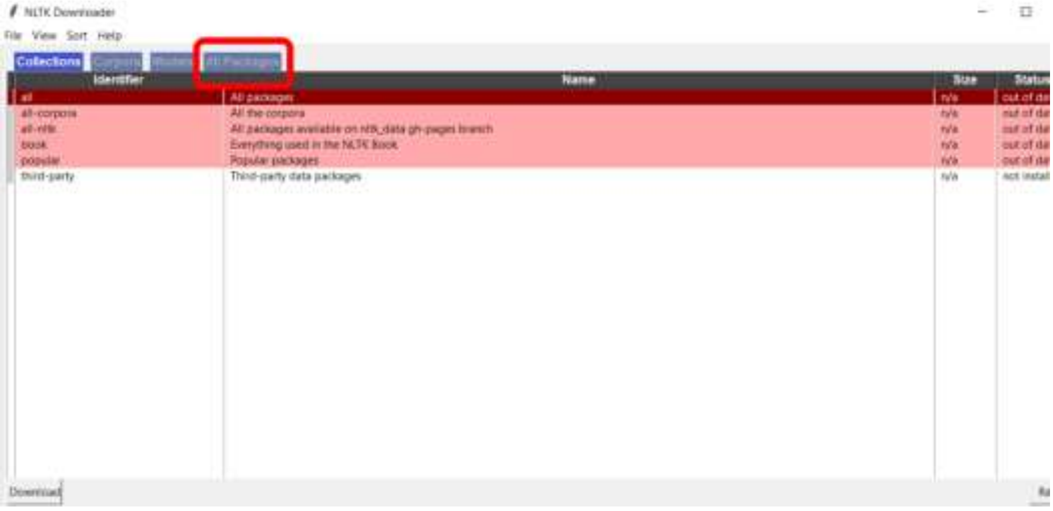
Activity 1: Installing NER in NLTK

- NLTK offers a machine learning-based named entity recogniser (NER).
- We will firstly need to install the NER module within our NLTK
- For this, create a new Python3 file in your Jupyter and enter the commands as shown below:



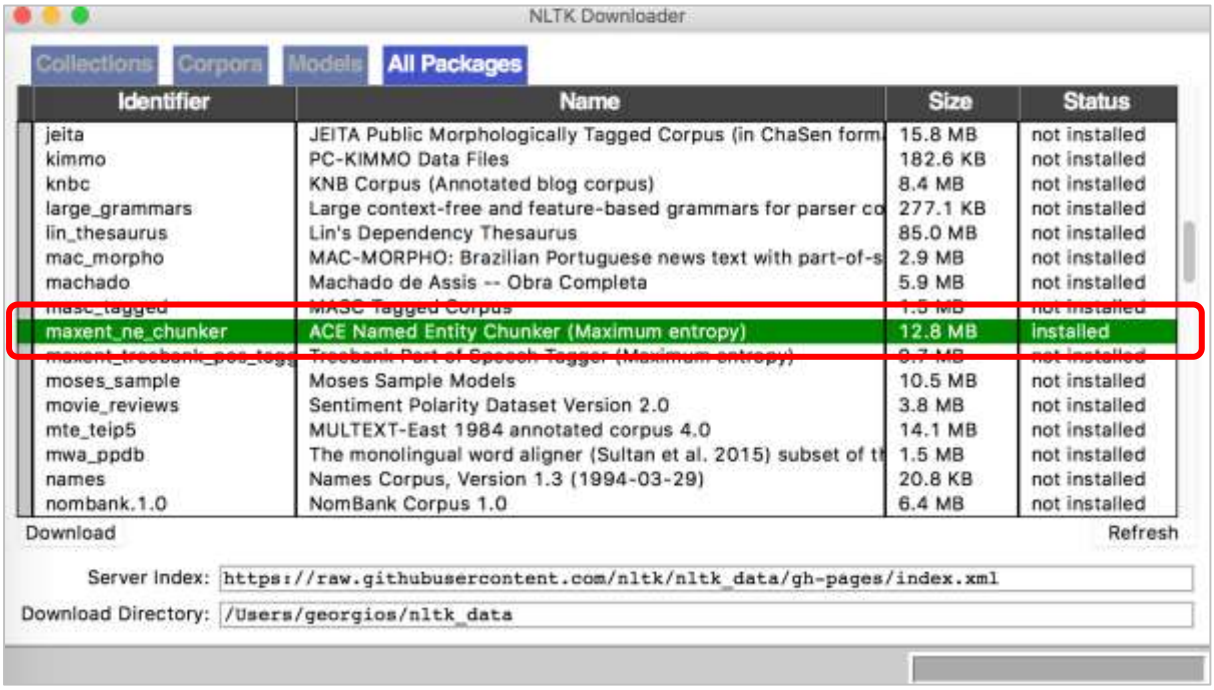
```
import nltk
nltk.download()
```

- In the new window that it appears, go to the ‘All Packages’ tab



Identifier	Name	Size	Status
all	All packages	n/a	out of da
all-corpora	All the corpora	n/a	out of da
all-nltk	All packages available on nltk_data gh-pages branch	n/a	out of da
book	Everything used in the NLTK Book	n/a	out of da
popular	Popular packages	n/a	out of da
third-party	Third-party data packages	n/a	not instal

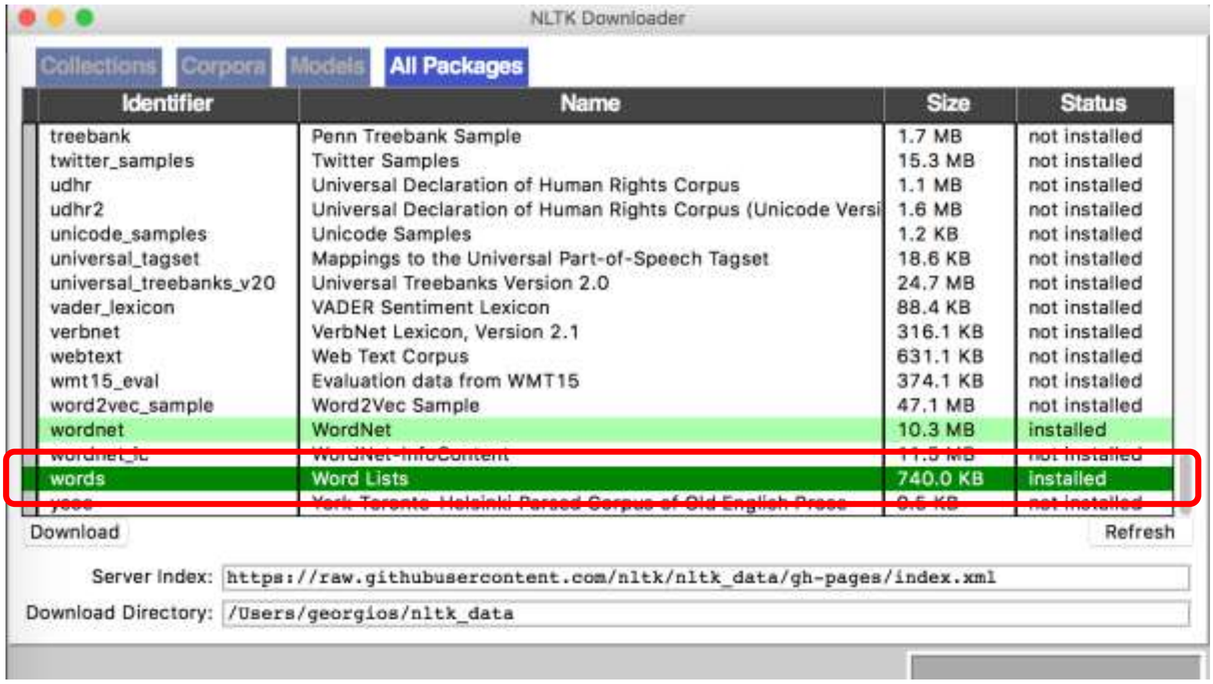
- And then download the following two packages: 1. ‘maxent_ne_chunker’ 2. ‘words’



Identifier	Name	Size	Status
jeita	JEITA Public Morphologically Tagged Corpus (in ChaSen form	15.8 MB	not installed
kimmo	PC-KIMMO Data Files	182.6 KB	not installed
knbc	KNB Corpus (Annotated blog corpus)	8.4 MB	not installed
large_grammars	Large context-free and feature-based grammars for parser co	277.1 KB	not installed
lin_thesaurus	Lin's Dependency Thesaurus	85.0 MB	not installed
mac_morpho	MAC-MORPHO: Brazilian Portuguese news text with part-of-s	2.9 MB	not installed
machado	Machado de Assis -- Obra Completa	5.9 MB	not installed
masc_tagged	MASC Tagged Corpus	1.5 MB	not installed
maxent_ne_chunker	ACE Named Entity Chunker (Maximum entropy)	12.8 MB	installed
maxent_treebank_pos_tagg	Treebank Part of Speech Tagger (Maximum entropy)	8.7 MB	not installed
moses_sample	Moses Sample Models	10.5 MB	not installed
movie_reviews	Sentiment Polarity Dataset Version 2.0	3.8 MB	not installed
mte_teip5	MULTEXT-East 1984 annotated corpus 4.0	14.1 MB	not installed
mwa_ppdb	The monolingual word aligner (Sultan et al. 2015) subset of th	1.5 MB	not installed
names	Names Corpus, Version 1.3 (1994-03-29)	20.8 KB	not installed
nombank.1.0	NomBank Corpus 1.0	6.4 MB	not installed

Server Index: `https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/index.xml`

Download Directory: `/Users/georgios/nltk_data`



Identifier	Name	Size	Status
treebank	Penn Treebank Sample	1.7 MB	not installed
twitter_samples	Twitter Samples	15.3 MB	not installed
udhr	Universal Declaration of Human Rights Corpus	1.1 MB	not installed
udhr2	Universal Declaration of Human Rights Corpus (Unicode Versi	1.6 MB	not installed
unicode_samples	Unicode Samples	1.2 KB	not installed
universal_tagset	Mappings to the Universal Part-of-Speech Tagset	18.6 KB	not installed
universal_treebanks_v20	Universal Treebanks Version 2.0	24.7 MB	not installed
vader_lexicon	VADER Sentiment Lexicon	88.4 KB	not installed
verbnet	VerbNet Lexicon, Version 2.1	316.1 KB	not installed
webtext	Web Text Corpus	631.1 KB	not installed
wmt15_eval	Evaluation data from WMT15	374.1 KB	not installed
word2vec_sample	Word2Vec Sample	47.1 MB	not installed
wordnet	WordNet	10.3 MB	installed
wordnet_ic	WordNet-InfoContent	11.5 MB	not installed
words	Word Lists	740.0 KB	installed
ycoe	York-Toronto-Helsinki Parsed Corpus of Old English Prose	8.5 KB	not installed

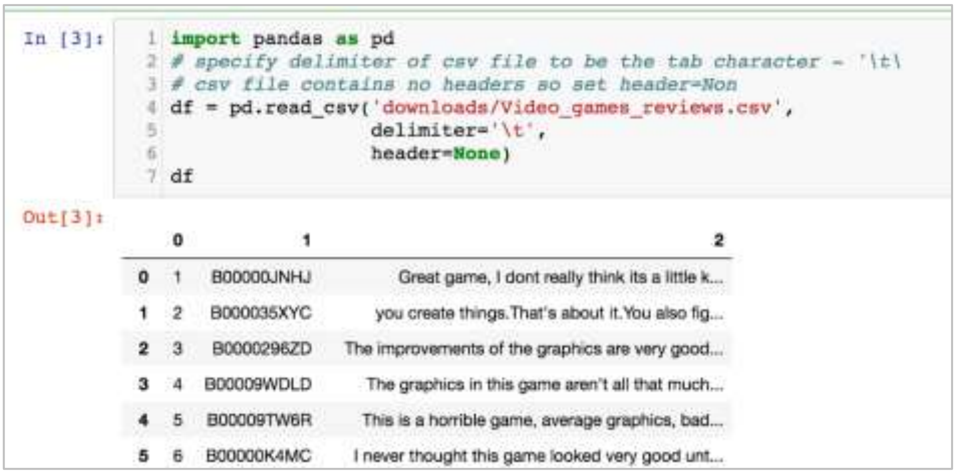
Server Index: `https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/index.xml`

Download Directory: `/Users/georgios/nltk_data`

<u>Activity 2</u>: Loading dataset using Pandas

- We will use the read_csv function of the Pandas library to load the Video games dataset into a Pandas DataFrame
- Note that the dataset is in a tab separated format so we will need to specify that the delimiter is the tab character

- Also the dataset contains no headers so we need to set the Header argument to None



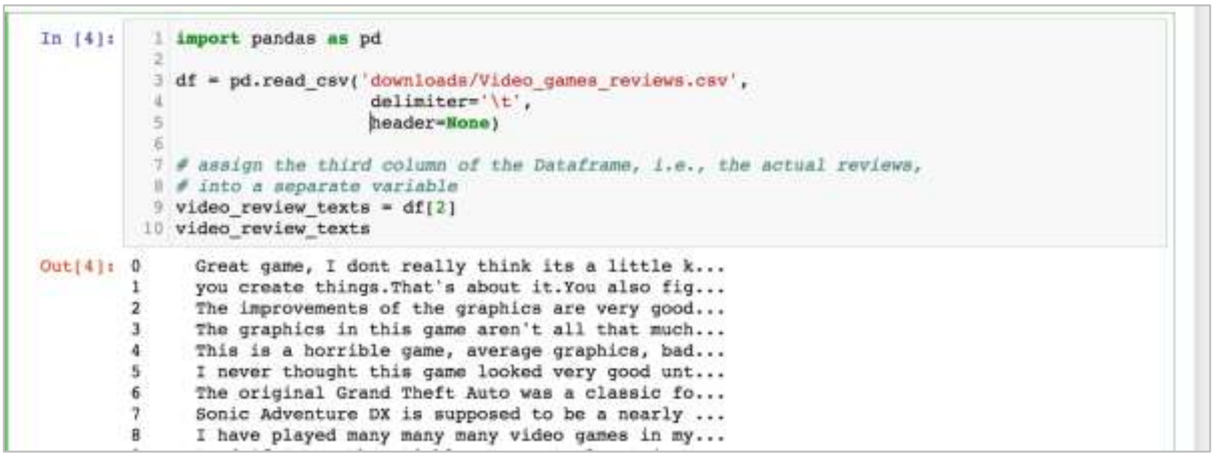
```
In [3]:
import pandas as pd
# specify delimiter of csv file to be the tab character - '\t\
# csv file contains no headers so set header=Non
df = pd.read_csv('downloads/Video_games_reviews.csv',
                 delimiter='\t',
                 header=None)
df
```

Out[3]:

	0	1	2
0	1	B00000JNHJ	Great game, I dont really think its a little k...
1	2	B000035XYC	you create things.That's about it.You also fig...
2	3	B0000296ZD	The improvements of the graphics are very good...
3	4	B00009WDLD	The graphics in this game aren't all that much...
4	5	B00009TW6R	This is a horrible game, average graphics, bad...
5	6	B00000K4MC	I never thought this game looked very good unt...

2.1 Extracting the Review Column from the Data Frame

- The DataFrame consists of three columns corresponding to the three columns of the Video game dataset
- However, we are only interested in analysing the third column which is the actual review of a Video game
- It is thus useful to assign the third column for the Dataframe into a separate variable

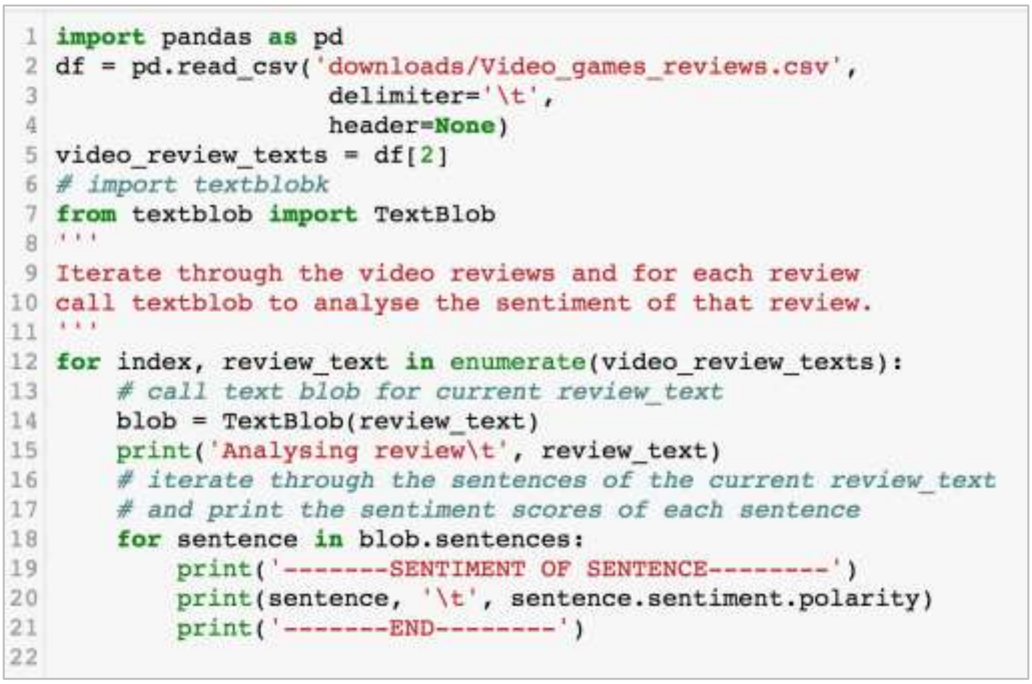


```
In [4]:
import pandas as pd

df = pd.read_csv('downloads/Video_games_reviews.csv',
                 delimiter='\t',
                 header=None)

# assign the third column of the Dataframe, i.e., the actual reviews,
# into a separate variable
video_review_texts = df[2]
video_review_texts
```

```
Out[4]: 0      Great game, I dont really think its a little k...
        1      you create things.That's about it.You also fig...
        2      The improvements of the graphics are very good...
        3      The graphics in this game aren't all that much...
        4      This is a horrible game, average graphics, bad...
        5      I never thought this game looked very good unt...
        6      The original Grand Theft Auto was a classic fo...
        7      Sonic Adventure DX is supposed to be a nearly ...
        8      I have played many many many video games in my...
```

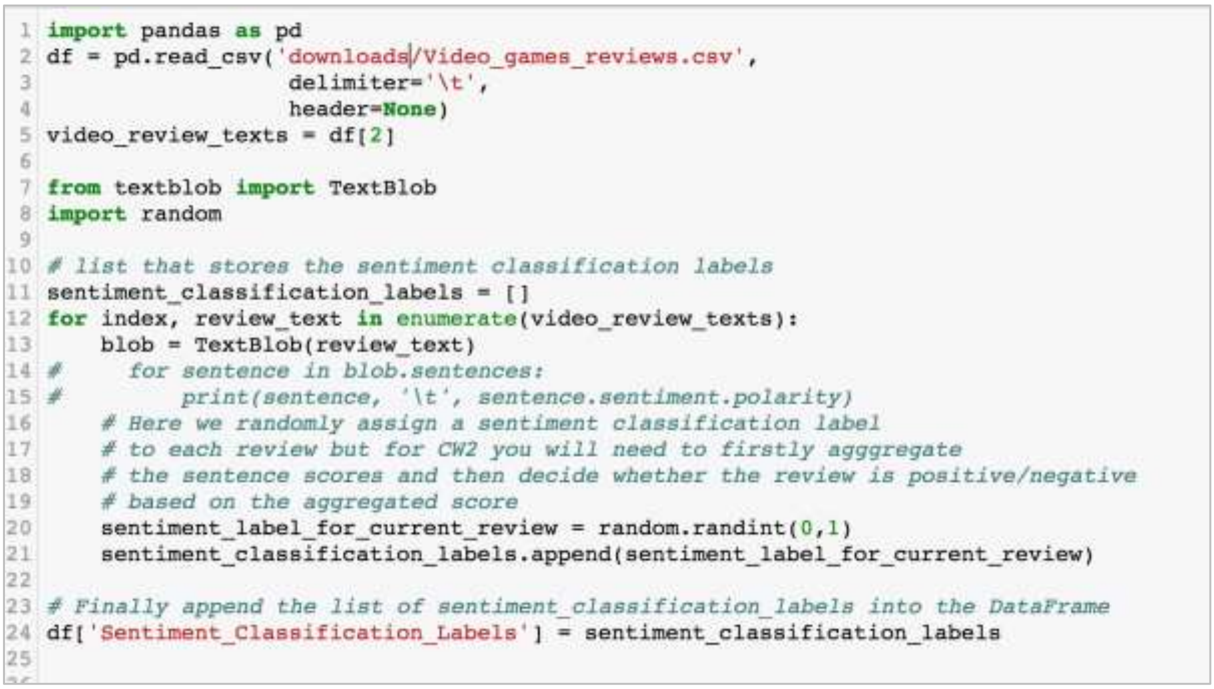
<u>**Activity 3**</u>: Sentiment Analysis

- We are now ready to analyse the sentiment of the Video game reviews using the textblob library
- For this we will iterate through the reviews in the video_review_texts and for each review we will call textblob to analyse the sentiment of that review
- Textblob will assign a sentiment score to each sentence of a review
- You will need to think how you can aggregate the sentence scores to derive an overall sentiment score for each review

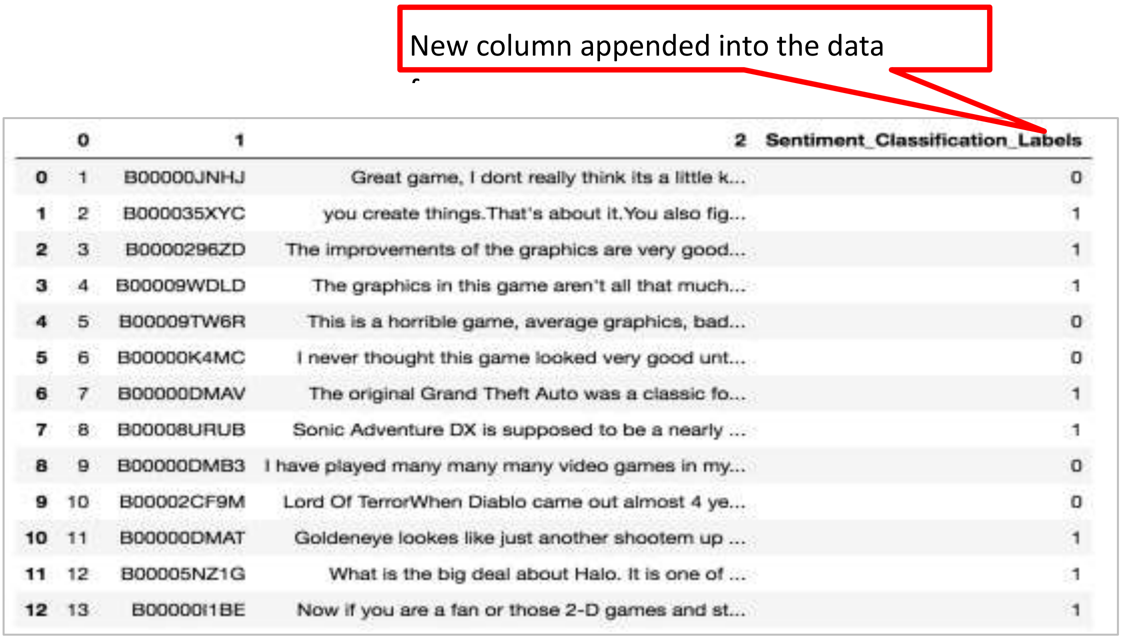


```
import pandas as pd
df = pd.read_csv('downloads/Video_games_reviews.csv',
                  delimiter='\t',
                  header=None)
video_review_texts = df[2]
# import textblobk
from textblob import TextBlob
'''
Iterate through the video reviews and for each review
call textblob to analyse the sentiment of that review.
'''
for index, review_text in enumerate(video_review_texts):
    # call text blob for current review_text
    blob = TextBlob(review_text)
    print('Analysing review\t', review_text)
    # iterate through the sentences of the current review_text
    # and print the sentiment scores of each sentence
    for sentence in blob.sentences:
        print('-------SENTIMENT OF SENTENCE--------')
        print(sentence, '\t', sentence.sentiment.polarity)
        print('-------END--------')

```

- After aggregating the sentiment scores of the constituent sentences of a review you can use a simple threshold to decide whether a review is positive or negative (e.g., if aggregate_score > 0 then review is positive else review is negative)
- Once you have decided whether a review is positive or negative you can store the sentiment classification labels on a separate column of your DataFrame
- The code below randomly decides whether a review is positive/negative and it then stores the sentiment classification labels in the dataframe

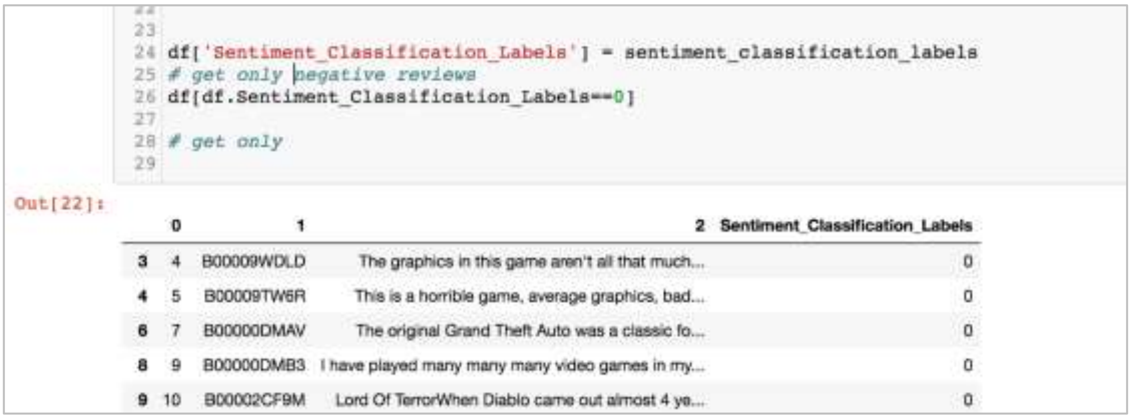


```
import pandas as pd
df = pd.read_csv('downloads/Video_games_reviews.csv',
                  delimiter='\t',
                  header=None)
video_review_texts = df[2]

from textblob import TextBlob
import random

# list that stores the sentiment classification labels
sentiment_classification_labels = []
for index, review_text in enumerate(video_review_texts):
    blob = TextBlob(review_text)
#     for sentence in blob.sentences:
#         print(sentence, '\t', sentence.sentiment.polarity)
    # Here we randomly assign a sentiment classification label
    # to each review but for CW2 you will need to firstly agggregate
    # the sentence scores and then decide whether the review is positive/negative
    # based on the aggregated score
    sentiment_label_for_current_review = random.randint(0,1)
    sentiment_classification_labels.append(sentiment_label_for_current_review)

# Finally append the list of sentiment_classification_labels into the DataFrame
df['Sentiment_Classification_Labels'] = sentiment_classification_labels

```


New column appended into the data

	0	1	2	Sentiment_Classification_Labels
0	1	B00000JNHJ	Great game, I dont really think its a little k...	0
1	2	B000035XYC	you create things.That's about it.You also fig...	1
2	3	B0000296ZD	The improvements of the graphics are very good...	1
3	4	B00009WDLD	The graphics in this game aren't all that much...	1
4	5	B00009TW6R	This is a horrible game, average graphics, bad...	0
5	6	B00000K4MC	I never thought this game looked very good unt...	0
6	7	B00000DMAV	The original Grand Theft Auto was a classic fo...	1
7	8	B00008URUB	Sonic Adventure DX is supposed to be a nearly ...	1
8	9	B00000DMB3	I have played many many many video games in my...	0
9	10	B00002CF9M	Lord Of TerrorWhen Diablo came out almost 4 ye...	0
10	11	B00000DMAT	Goldeneye lookes like just another shootem up ...	1
11	12	B00005NZ1G	What is the big deal about Halo. It is one of ...	1
12	13	B00000I1BE	Now if you are a fan or those 2-D games and st...	1

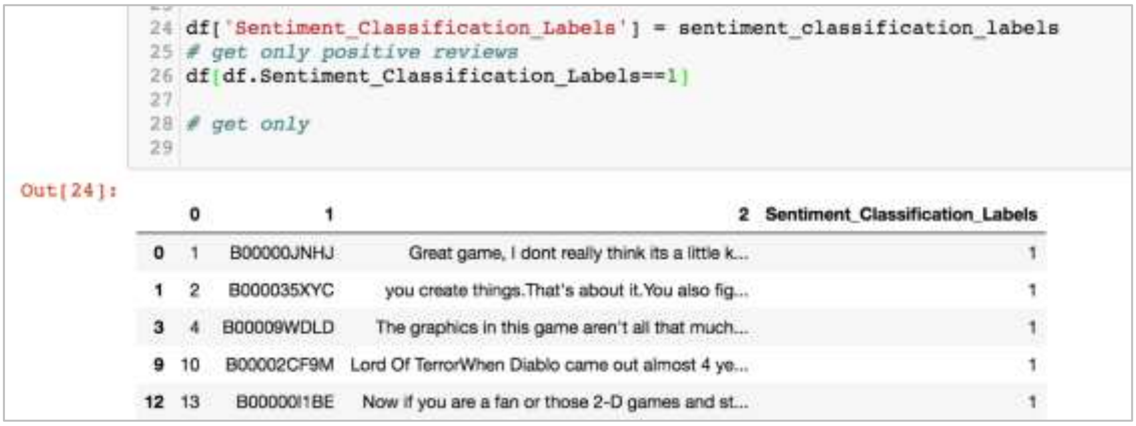
- The sentiment classification labels are now stored in the fourth column of our DataFrame
- We can apply standard Dataframe operations to retrieve only the positive or only the negative reviews



```
df['Sentiment_Classification_Labels'] = sentiment_classification_labels
# get only negative reviews
df[df.Sentiment_Classification_Labels==0]

# get only

```

Out[22]:

	0	1	2	Sentiment_Classification_Labels
3	4	B00009WDLD	The graphics in this game aren't all that much...	0
4	5	B00009TW6R	This is a horrible game, average graphics, bad...	0
6	7	B00000DMAV	The original Grand Theft Auto was a classic fo...	0
8	9	B00000DMB3	I have played many many many video games in my...	0
9	10	B00002CF9M	Lord Of TerrorWhen Diablo came out almost 4 ye...	0



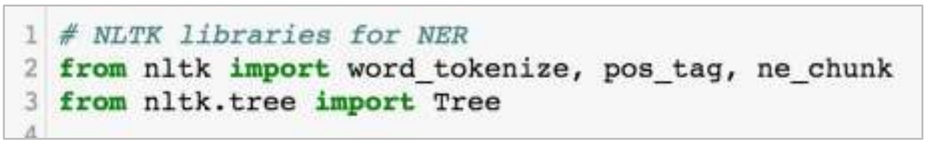
```
df['Sentiment_Classification_Labels'] = sentiment_classification_labels
# get only positive reviews
df[df.Sentiment_Classification_Labels==1]

# get only

```

Out[24]:

	0	1	2	Sentiment_Classification_Labels
0	1	B00000JNHJ	Great game, I dont really think its a little k...	1
1	2	B000035XYC	you create things.That's about it.You also fig...	1
3	4	B00009WDLD	The graphics in this game aren't all that much...	1
9	10	B00002CF9M	Lord Of TerrorWhen Diablo came out almost 4 ye...	1
12	13	B00000I1BE	Now if you are a fan or those 2-D games and st...	1

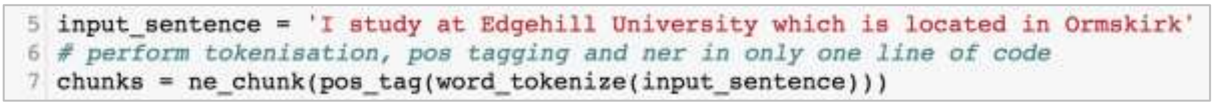
<u>Activity 4</u>: Named Entity Recognition

- Firstly, import the NLTK libraries which will be used for NER



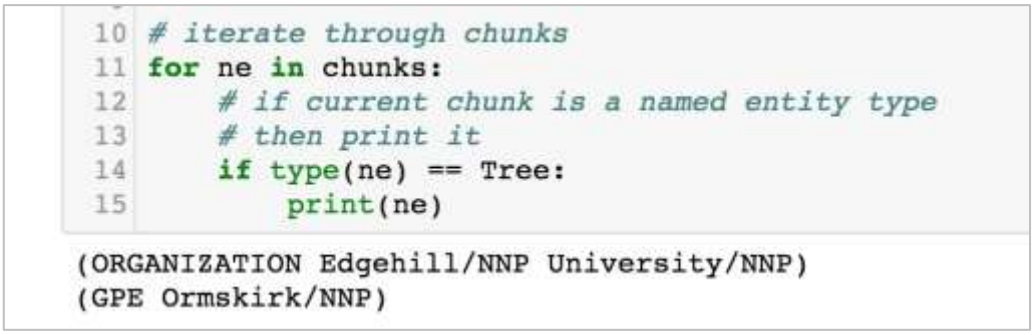
```
# NLTK libraries for NER
from nltk import word_tokenize, pos_tag, ne_chunk
from nltk.tree import Tree
```

- In NLTK we can perform tokenisation, POS tagging and NER in only one line of code
- Note that tokenisation and POS tagging are necessary pre-processing steps for NER



```
input_sentence = 'I study at Edgehill University which is located in Ormskirk'
# perform tokenisation, pos tagging and ner in only one line of code
chunks = ne_chunk(pos_tag(word_tokenize(input_sentence)))
```

- The ne_chunk function of NLTK will return a list of words or phrases occurring in the input sentence together with their POS tag and their named entity label
- The NLTK NER identifies the following named entities:
    - Organisation, Person, Location, Date, Time, Money, Percent, Facility and GPE (geo-political entity)
- In order to print the identified named entities that occur in the input sentence, simply iterate through the list of chunks and print only the chunks that contain a named entity (ignore chunks that contain only a POS tag)



```
# iterate through chunks
for ne in chunks:
    # if current chunk is a named entity type
    # then print it
    if type(ne) == Tree:
        print(ne)
```

```
(ORGANIZATION Edgehill/NNP University/NNP)
(GPE Ormskirk/NNP)
```

Exercise: Apply NER to the Video Game dataset