Preprocessing

In [ ]:

```
!pip install smart_open
```

In [20]:

```
from smart_open import open

import json
import pprint
import pandas as pd
import numpy as np
```

In [52]:

```
%%time
line_count = 0
with open('http://deepyeti.ucsd.edu/jianmo/amazon/categoryFilesSmall/Video_Games_5.
json.gz') as fin:
    with open('s3://checkpoint1-inputfiles/reviews_videogames.json', 'w') as fout:
        for line in fin:
            fout.write(line)
            line_count = line_count + 1
print('JSON extracted to S3: ' + str(line_count) + ' lines!')
```

```
JSON extracted to S3: 497577 lines!
CPU times: user 6.61 s, sys: 890 ms, total: 7.5 s
Wall time: 12 s
```

In [4]:

```
with open('s3://checkpoint1-inputfiles/reviews_videogames.json') as f:
    first_line = f.readline()
print(first_line)
```

```
{"overall": 5.0, "verified": true, "reviewTime": "10 17, 2015", "review
erID": "A1HP7NVNPFMA4N", "asin": "0700026657", "reviewerName": "Ambrosi
a075", "reviewText": "This game is a bit hard to get the hang of, but w
hen you do it's great.", "summary": "but when you do it's great.", "uni
xReviewTime": 1445040000}
```

In [5]:

```
pprint.pprint(json.loads(first_line))
```

```
{'asin': '0700026657',
 'overall': 5.0,
 'reviewText': 'This game is a bit hard to get the hang of, but when yo
u do '
               "it's great.",
 'reviewTime': '10 17, 2015',
 'reviewerID': 'A1HP7NVNPFMA4N',
 'reviewerName': 'Ambrosia075',
 'summary': "but when you do it's great.",
 'unixReviewTime': 1445040000,
 'verified': True}
```

In [6]:

```
%%time
with open('s3://checkpoint1-inputfiles/reviews_videogames.json') as f:
        content = [json.loads(line) for line in f]
```

```
CPU times: user 6.4 s, sys: 973 ms, total: 7.37 s
Wall time: 10.2 s
```

In [7]:

```
df = pd.json_normalize(content)
```

In [8]:

```
df.shape
```

Out[8]:

```
(497577, 30)
```

In [9]:

```
df.columns = df.columns.str.replace(':','')
df.columns = df.columns.str.replace(' ','-')
df.columns = df.columns.str.replace('.','_')
```

In [10]:

```
df['vote'] = df['vote'].replace(np.nan, 0)
df['vote'] = df['vote'].str.replace(',', '')
df['vote'] = df['vote'].apply(pd.to_numeric)
```

In [11]:

```
df.head(1)
```

Out[11]:

	overall	verified	reviewTime	reviewerID	asin	reviewerName	reviewText	sumr
0	5.0	True	10 17, 2015	A1HP7NVNPFMA4N	0700026657	Ambrosia075	This game is a bit hard to get the hang of, bu...	but v yc it's g

1 rows × 30 columns

In [12]:

```
df.dtypes
```

Out[12]:

```
overall                        float64
verified                          bool
reviewTime                      object
reviewerID                      object
asin                            object
reviewerName                    object
reviewText                      object
summary                         object
unixReviewTime                   int64
vote                           float64
style_Format                    object
image                           object
style_Platform                  object
style_Edition                   object
style_Color                     object
style_Size                      object
style_Style                     object
style_Length                    object
style_Subscription-Length       object
style_Content                   object
style_Package-Type              object
style_Package-Quantity          object
style_Item-Package-Quantity     object
style_Pattern                   object
style_Platform-for-Display      object
style_Style-Name                object
style_Denomination              object
style_Offer-Type                object
style_Configuration             object
style_Color-Name                object
dtype: object
```

In [13]:

```
df.columns
```

Out[13]:

```
Index(['overall', 'verified', 'reviewTime', 'reviewerID', 'asin',
       'reviewerName', 'reviewText', 'summary', 'unixReviewTime', 'vot
e',
       'style_Format', 'image', 'style_Platform', 'style_Edition',
       'style_Color', 'style_Size', 'style_Style', 'style_Length',
       'style_Subscription-Length', 'style_Content', 'style_Package-Typ
e',
       'style_Package-Quantity', 'style_Item-Package-Quantity',
       'style_Pattern', 'style_Platform-for-Display', 'style_Style-Nam
e',
       'style_Denomination', 'style_Offer-Type', 'style_Configuration',
       'style_Color-Name'],
      dtype='object')
```

In [14]:

```
%%time
spl_chars = [' \n', '"', "'"]
for char in spl_chars:
    for column in df.columns:
        if df[column].dtypes == 'object':
            df[column] = df[column].str.replace(char, ' ')
```

```
CPU times: user 14.4 s, sys: 300 ms, total: 14.7 s
Wall time: 14.8 s
```

In [15]:

```
df['reviewTime'] = df['reviewTime'].str.replace(',', '')
df['reviewTime'] = df['reviewTime'].str.replace(' ', '/')
df['reviewTime'] = pd.to_datetime(df['reviewTime'])
```

In [16]:

```
df.to_csv("s3://checkpoint1-inputfiles/reviews_videogames.csv", index = False)
```

Read the csv file from S3

In [63]:

```
from smart_open import open

import json
import pprint
import pandas as pd
import numpy as np

from sagemaker import get_execution_role

%matplotlib inline

import os, re

import boto3
import matplotlib.pyplot as plt

np.set_printoptions(precision=3, suppress=True)

import sagemaker
from sagemaker.amazon.common import RecordSerializer
from sagemaker.serializers import CSVSerializer
from sagemaker.deserializers import JSONDeserializer
```

In [64]:

```
# Read the csv file saved in s3 bucket
%%time
videogames = pd.read_csv("s3://checkpoint1-inputfiles/reviews_videogames.csv",low_m
emory = False)
```

```
CPU times: user 6.87 s, sys: 993 ms, total: 7.86 s
Wall time: 25.6 s
```

In [65]:

```
videogames.head(2)
```

Out[65]:

	overall	verified	reviewTime	reviewerID	asin	reviewerName	reviewText	sumr
0	5.0	True	2015-10-17	A1HP7NVNPFMA4N	0700026657	Ambrosia075	This game is a bit hard to get the hang of, bu...	but v you s g
1	4.0	False	2015-07-27	A1JGAP0185YJI6	0700026657	travis	I played it a while but it was alright. The st...	E spi tl was lik

2 rows × 30 columns

In [66]:

```
videogames[videogames.asin == '0700026657'].columns
```

Out[66]:

```
Index(['overall', 'verified', 'reviewTime', 'reviewerID', 'asin',
       'reviewerName', 'reviewText', 'summary', 'unixReviewTime', 'vot
e',
       'style_Format', 'image', 'style_Platform', 'style_Edition',
       'style_Color', 'style_Size', 'style_Style', 'style_Length',
       'style_Subscription-Length', 'style_Content', 'style_Package-Typ
e',
       'style_Package-Quantity', 'style_Item-Package-Quantity',
       'style_Pattern', 'style_Platform-for-Display', 'style_Style-Nam
e',
       'style_Denomination', 'style_Offer-Type', 'style_Configuration',
       'style_Color-Name'],
      dtype='object')
```

Word2vec

In [67]:

```
session = sagemaker.Session()
role = get_execution_role()
bucket = session.default_bucket()
prefix = "sagemaker/lda-videogames-review"

print("Training input/output will be stored in {}/{}".format(bucket, prefix))
print("\nIAM Role: {}".format(role))
```

```
Training input/output will be stored in sagemaker-us-east-1-61149064076
0/sagemaker/lda-videogames-review

IAM Role: arn:aws:iam::611490640760:role/LabRole
```

In [68]:

```
reviewers = set(videogames.reviewerID.unique())
```

In [69]:

```
user_to_products = pd.DataFrame(columns = [['reviewerID','products']])
```

Loading dict

In [9]:

```
%%time
with open('s3://checkpoint1-inputfiles/products_file.txt') as f:
        contents =  eval(f.read())
```

```
CPU times: user 1.23 s, sys: 217 ms, total: 1.45 s
Wall time: 1.97 s
```

In [75]:

```
df = pd.DataFrame(list(contents.items()),columns = ['reviewerID','asin'])
```

In [76]:

```
# clean the asin columns

df['asin'] = df['asin'].astype(str).str.replace('[', '', regex = True).replace(']',
'', regex = True).replace({'\'': ''}, regex = True).replace(',','', regex=True).rep
lace('\\n',' ', regex = True)
```

In [77]:

```
df.head(3)
```

Out[77]:

	reviewerID	asin
0	A2T7YFEAI0X74W	B00009WAUH B000O62OS6 B000P5FEJC B000QW9D14 B0...
1	A3043WYL272JIK	B000067DPM B000087L4G B00008XKZM B0008GJRQ4 B0...
2	A257QFK8MAYEUC	B003S9WJ9A B00503E8S2 B0054IUY22 B006RJ373K B0...

In [80]:

```
# save product vocabulary file

df.asin.to_csv('s3://checkpoint1-inputfiles/train_data/product_vocab.txt', header=F
alse, index=False, sep='\t')
```

In [81]:

```
# load the file with product vocabulary

#%%time
with open('s3://checkpoint1-inputfiles/train_data/product_vocab.txt', 'r') as f:
    c2 = f.readlines()

c3 = [s.rstrip('\n') for s in c2]
```

In [82]:

```
total_words = [" ".join(c3)]
```

In [83]:

```
import nltk
nltk.download('punkt')
from nltk.tokenize import word_tokenize
```

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

In [84]:

```
# tokenize the words in products files

tokenized_sents = [word_tokenize(i) for i in total_words]
```

In [85]:

```
# find the unique words

vocabulary = np.unique(tokenized_sents[0])
```

In [97]:

```
# create a file with prodcuts details to use for training

with open('s3://checkpoint1-inputfiles/train_data/products.txt', 'w') as file:
    strings = df.asin.to_string(header=False, index=False)
    file.write(strings)
```

In [98]:

```
%%time
with open('s3://checkpoint1-inputfiles/train_data/products.txt') as f:
    c =  f.readlines()
```

```
CPU times: user 44.5 ms, sys: 3.65 ms, total: 48.2 ms
Wall time: 125 ms
```

Word2Vec

In [100]:

```
train_data = 's3://checkpoint1-inputfiles/train_data/products.txt'
```

In [101]:

```
s3_output_location = 's3://sagemaker-studio-1orda0h8hkx/outputfiles-word2vec'
```

In [102]:

```
region_name = boto3.Session().region_name
sess = sagemaker.Session()
```

In [103]:

```
container = sagemaker.image_uris.retrieve(
    region=region_name, framework="blazingtext"
)
print("Using SageMaker BlazingText container: {} ({})".format(container, region_nam
e))
```

```
Using SageMaker BlazingText container: 811284229777.dkr.ecr.us-east-1.a
mazonaws.com/blazingtext:1 (us-east-1)
```

In [104]:

```
bt_model = sagemaker.estimator.Estimator(
    container,
    role,
    instance_count=1,
    instance_type="ml.m5.large",
    volume_size=30,
    max_run=360000,
    input_mode="File",
    output_path=s3_output_location,
    sagemaker_session=sess,
)
```

In [105]:

```
bt_model.set_hyperparameters(
    mode="skipgram",
    epochs=5,
    min_count=5,
    sampling_threshold=0.0001,
    learning_rate=0.05,
    window_size=5,
    vector_dim=100,
    negative_samples=5,
    subwords=True,  # Enables learning of subword embeddings for OOV word vector ge
neration
    min_char=3,  # min length of char ngrams
    max_char=6,  # max length of char ngrams
    batch_size=11,  #  = (2*window_size + 1) (Preferred. Used only if mode is batch
_skipgram)
    evaluation=True,
)  # Perform similarity evaluation on WS-353 dataset at the end of training
```

Train

In [106]:

```
train_data = sagemaker.inputs.TrainingInput(
    train_data,
    distribution="FullyReplicated",
    content_type="text/plain",
    s3_data_type="S3Prefix",
)
data_channels = {"train": train_data}
```

In [107]:

```
bt_model.fit(inputs=data_channels, logs=True)
```

```
2021-11-02 19:45:03 Starting - Starting the training job...
2021-11-02 19:45:32 Starting - Launching requested ML instancesProfiler
Report-1635882302: InProgress
......
2021-11-02 19:46:33 Starting - Preparing the instances for trainin
g.........
2021-11-02 19:47:53 Downloading - Downloading input data...
2021-11-02 19:48:33 Training - Training image download completed. Train
ing in progress..Arguments: train
[11/02/2021 19:48:25 WARNING 140519756232320] Loggers have already been
setup.
[11/02/2021 19:48:25 WARNING 140519756232320] Loggers have already been
setup.
[11/02/2021 19:48:25 INFO 140519756232320] nvidia-smi took: 0.025172472
00012207 secs to identify 0 gpus
[11/02/2021 19:48:25 INFO 140519756232320] Running single machine CPU B
lazingText training using skipgram mode.
Number of CPU sockets found in instance is  1
[11/02/2021 19:48:25 INFO 140519756232320] Processing /opt/ml/input/dat
a/train/products.txt . File size: 2.685901641845703 MB
Read 0M words
Number of words:  8483
##### Alpha: 0.0489  Progress: 2.24%  Million Words/sec: 0.04 #####
##### Alpha: 0.0458  Progress: 8.42%  Million Words/sec: 0.10 #####
##### Alpha: 0.0427  Progress: 14.59%  Million Words/sec: 0.13 #####
##### Alpha: 0.0396  Progress: 20.89%  Million Words/sec: 0.15 #####
##### Alpha: 0.0367  Progress: 26.51%  Million Words/sec: 0.16 #####
##### Alpha: 0.0337  Progress: 32.66%  Million Words/sec: 0.17 #####
##### Alpha: 0.0307  Progress: 38.50%  Million Words/sec: 0.18 #####
##### Alpha: 0.0276  Progress: 44.86%  Million Words/sec: 0.19 #####
##### Alpha: 0.0245  Progress: 51.09%  Million Words/sec: 0.20 #####
##### Alpha: 0.0214  Progress: 57.23%  Million Words/sec: 0.20 #####
##### Alpha: 0.0183  Progress: 63.36%  Million Words/sec: 0.20 #####
##### Alpha: 0.0152  Progress: 69.69%  Million Words/sec: 0.21 #####
##### Alpha: 0.0121  Progress: 75.88%  Million Words/sec: 0.21 #####
##### Alpha: 0.0089  Progress: 82.15%  Million Words/sec: 0.22 #####
##### Alpha: 0.0059  Progress: 88.27%  Million Words/sec: 0.22 #####
##### Alpha: 0.0027  Progress: 94.61%  Million Words/sec: 0.22 #####
##### Alpha: -0.0000  Progress: 100.00%  Million Words/sec: 0.22 #####
##### Alpha: 0.0000  Progress: 100.00%  Million Words/sec: 0.22 #####
Training finished.
Average throughput in Million words/sec: 0.22
Total training time in seconds: 7.53
Evaluating word embeddings....
Vectors read from: /opt/ml/model/vectors.txt
{
    "EN-WS-353-ALL.txt": {
        "not_found": 353,
        "spearmans_rho": 0.0,
        "total_pairs": 353
    },
    "EN-WS-353-REL.txt": {
        "not_found": 252,
        "spearmans_rho": 0.0,
        "total_pairs": 252
    },
    "EN-WS-353-SIM.txt": {
        "not_found": 203,
        "spearmans_rho": 0.0,
        "total_pairs": 203
    },
    "mean_rho": 0.0
}
[11/02/2021 19:48:41 INFO 140519756232320] #mean_rho: 0.0

2021-11-02 19:48:53 Uploading - Uploading generated training model
2021-11-02 19:50:54 Completed - Training job completed
ProfilerReport-1635882302: NoIssuesFound
Training seconds: 163
Billable seconds: 163
```

Deploy

In [108]:

```
bt_endpoint = bt_model.deploy(initial_instance_count=1, instance_type="ml.m4.xlarg
e")
```

```
---------!
```

Predict

In [109]:

```
from sagemaker.serializers import JSONSerializer

bt_endpoint.serializer = JSONSerializer()

word1 = ["3828770193"]
word2 = ["6050036071"]
payload1 = {"instances": word1}
payload2 = {"instances": word2}
response1 = bt_endpoint.predict(payload1)
response2 = bt_endpoint.predict(payload2)
vecs1 = json.loads(response1)
vecs2 = json.loads(response2)
```

In [110]:

```
def similarity(v1, v2):
    n1 = np.linalg.norm(v1)
    n2 = np.linalg.norm(v2)
    return np.dot(v1, v2) / n1 / n2
```

In [111]:

```
v1 = vecs1[0]['vector']
v2 = vecs2[0]['vector']
```

In [112]:

```
similarity(v1, v2)
```

Out[112]:

```
-0.19820168174042893
```

Item similarity for recommendation

In [113]:

```
# function to find similar products

vectors = {}

def item_vectors():
    vectors = {}
    payload = {"instances": vocabulary}
    response = bt_endpoint.predict(payload)
    vecs = json.loads(response)
    for i in vecs:
        arr = np.array(i["vector"], dtype=float)
        if np.linalg.norm(arr) == 0:
            continue
        vectors[i["word"]] = arr
    return vectors
```

In [138]:

```
vectors = item_vectors()
```

In [164]:

```
def item_similarity(item, vectors):
    sim = {}
    v1 = vectors[item]
    for i in vocabulary:
        if i != item:
            v2 = vectors[i]
            d = similarity(v1, v2)
            sim[i] = d
    return sim

def recommendations(item, n, vectors):
    sim = item_similarity(item, vectors)
    sorted_sim = {k: v for k, v in sorted(sim.items(), key=lambda item: item[1])}
    n_items = list(sorted_sim)[:n]
    return n_items

def find_reviewer(videogames, reviewerid, n, vectors):
    asin = videogames[videogames.reviewerID == reviewerid].sort_values(['reviewTim
e','unixReviewTime'], ascending=False).iloc[-1].asin
    recos = recommendations(asin,n,vectors)
    return recos
```

In [170]:

```
find_reviewer(videogames, 'A1HP7NVNPFMA4N', 2, vectors)
```

Out[170]:

```
['3828770193', '8565000168']
```

In [ ]:

```

```