# -\*- coding: utf-8 -\*-

"""

Created on Tue Oct 9 21:28:14 2018

@author: user

"""

from datetime import datetime

from elasticsearch import helpers, Elasticsearch

es = Elasticsearch()

import csv

import requests

from lxml import html

import collections

from urllib.parse import urljoin

from bs4 import BeautifulSoup

from elasticsearch import Elasticsearch

import json

import pandas as pd

import os

import nltk

from textblob import TextBlob

import matplotlib.pyplot as plt

#directory = r'C:\Users\user\Documents\year 2\text minng'

res = requests.get('http://localhost:9200')

print (res.content)

es = Elasticsearch([{'host': 'localhost', 'port': '9200'}])

file = r'C:\Users\user\Documents\year 2\text minng\Video\_Games\_5.json'

df = pd.read\_json(file, lines = True)

df['reviewTime'] = pd.to\_datetime(df['reviewTime'], format='%m %d, %Y')

print (df.columns)

df = pd.read\_json(file, lines = True)

#data = df.to\_json(orient='records', lines=True)

cols = list(df.columns)

print (cols)

df['reviewTime'] = pd.to\_datetime(df['reviewTime'], format='%m %d, %Y')

print (df.columns)

from textblob import TextBlob

#first create new columns, they are empty for now

df['polarity'] = ""

df['subjectivity'] = ""

df.columns

#start loop for filling in the columns with values

i = 0

while i < len(df):

comm\_blob = TextBlob(df.iloc[i,3])

df.iloc[i,9]= comm\_blob.sentiment[0]

df.iloc[i,10] = comm\_blob.sentiment[1]

i += 1

if i % 10000 == 0:

print(i)

else:

continue

df.columns

csvPath = r'C:\Users\user\Documents\year 2\text minng\groupproject.csv'

data = df.to\_csv(csvPath, index = False)

with open(csvPath, encoding='utf-8') as x:

READER = csv.DictReader(x)

helpers.bulk(es, READER, index='mallikav5', doc\_type='mallikag5')

df.head(data)

df.head(20)

#I am not very good at hashtags or emojis or abbreviations for texts/tweets/facebook posts. Like when I used to think "smfh" meant "smacking my forehead." So I thought it would be fun to see what hashtags are used in videogame reviews (and how many people use, and if the ones who use a lot are crazy, etc)

#most number of hashtags

df['hastags'] = df['reviewText'].apply(lambda x: len([x for x in x.split() if x.startswith('#')]))

#df[['reviewText','hastags']].head(10)

df.head(10)

df.columns

dfsorted = df.sort\_values(by=['hastags'], ascending=False)

dfsorted.head(10)

dfsorted[['reviewText','hastags']].head(10)

#view the actual text in Kibana.

csvPath2 = r'C:\Users\user\Documents\year 2\text minng\groupproject\_hashytags.csv'

data = dfsorted.to\_csv(csvPath2, index = 135523)

with open(csvPath, encoding='utf-8') as y:

READER = csv.DictReader(y)

helpers.bulk(es, READER, index='mallikav55', doc\_type='mallikag55')

#turns out many, most, or all of these "#" are not being used as hashtags but are kindof crazy to read anyway.

#no, not useful at all, nor entertaining. Couldn't make my way through the entirety of a single one. moving on...

#most reviewed items, yes boring

df2 = df.groupby('asin').count()

df2.head

df2.columns

df2.sort\_values(by=['reviewText'], ascending=False)

#most recently reviewed items, still boring

df3 = df.sort\_values(by=['reviewTime'], ascending=False)

df3.head

df3[['asin','reviewTime','overall']]

#and now, for something REALLY useful (drumroll)

# I wondered if reviewers had installation problems, like I do for everything python or textmining related

#searched in Kibana’s discover tab for 'reviewText : "install\*"' and made a csv out of it.

file2 = r'C:\Users\user\Documents\year 2\text minng\New Saved Search.csv'

WANTED = 40

with open(file2) as searchfile:

for line in searchfile:

left,sep,right = line.partition('Install')

if sep:

print(sep,right[:WANTED])

#unfortunately I can't tell which review these strings go with. back to Kibana to search one at a time. #haven't figured out how to make the output a csv or something, either. just lives in my console.

#BUT I can read about other people getting angry when they can't install a thing...and that makes me feel better about this class.

#THE END