# Analysis of Amazon review content

90 percent of the consumers read online reviews before they decide to purchase any Mobile phone from any e-commerce website .Online Mobile applications has revolutionised the way consumers purchase mobile phones online as these apps have all the information regarding any mobile phone at users finger tips. Amazon is one of the best mobile applications which is considered as a treasure trove of all mobile reviews, and their review system is accessible across all channels presenting reviews in an easy-to-use format. So,There should be a system which analyses thousands of reviews of unlocked mobile phones sold on Amazon.com to find insights with respect to reviews, ratings, price and their relationships.

**Solution:**

This project aims at building a model to predict the helpfulness of the review and the rating based on the review text. Corpus-based and knowledge-based methods can be used to determine the semantic similarity of review text. We will be using Natural language processing to analyse the sentiment ( positive or a negative) of the given review . A sample web application is integrated to the model built.

# ->started building NLP model .

import re

import pandas as pd

import numpy as np

dataset=pd.read\_csv("20191226-reviews.csv")

dataset.tail()

dataset.isnull().sum()

dataset=dataset.drop(columns=['asin','name','helpfulVotes','date'],axis=1)

dataset.isnull().sum()

#dataset.dropna(inplace=True)

dataset.isnull().sum()

dataset.head()

a=dataset['rating'].tolist()

a

d=[]

for i in range(len(a)):

if a[i]>=3:

d.append(1)

else:

d.append(0)

d

dt=pd.DataFrame(d,columns=['emotion'])

dt

data1=pd.concat([dataset,dt],axis=1)

data1.isnull().sum()

data1.dropna(inplace=True)

data1.isnull().sum()

data1.head()

data1['verified'].isin([False]).sum()

data1['verified'].replace([False],[np.nan],inplace=True)

data1['verified'].isnull().sum()

data1.dropna(inplace=True)

data1.head()

data1.drop(['verified'],axis=1,inplace=True)

#df["period"] = df["Year"] + df["quarter"] another method

data1['Review'] = data1[['title', 'body',]].agg(' '.join, axis=1)

data1.drop(['title','body','rating'],axis=1,inplace=True)

data1.tail()

len(list(data1['emotion']))

x=data1.iloc[:,1].values

len(x)

from nltk.corpus import stopwords

from nltk.stem.porter import PorterStemmer

from nltk.stem import WordNetLemmatizer

ps = PorterStemmer()

wordnet=WordNetLemmatizer()

len(x)

corpus=[]

for i in range(len(x)):

temp=re.sub('[^a-zA-Z]',' ',x[i])

temp=temp.lower()

temp=temp.split()

temp=[wordnet.lemmatize(word) for word in temp if not word in set(stopwords.words('english'))]

temp=' '.join(temp)

corpus.append(temp)

corpus

afgfggbnkfgd print if else #from sklearn.feature\_extraction.text import TfidfVectorizer

#cv=TfidfVectorizer(max\_features= 2500)

#z=cv.fit\_transform(corpus).toarray()

from sklearn.feature\_extraction.text import CountVectorizer

cv=CountVectorizer(max\_features= 100)

z=cv.fit\_transform(corpus).toarray()

import pickle

pickle.dump(cv,open('count\_vec.pkl','wb'))

type(z)

z

y=data1.iloc[:,0].values

y

from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test=train\_test\_split(z,y,test\_size=0.33)

x\_train.shape

import keras

from keras.models import Sequential

from keras.layers import Dense

model=Sequential()

model.add(Dense(input\_dim=100,init="random\_uniform",activation="relu",output\_dim=30))

model.add(Dense(init="random\_uniform",activation="relu",output\_dim=25))

model.add(Dense(init="random\_uniform",activation="sigmoid",output\_dim=1))

model.compile(optimizer="adam",loss="binary\_crossentropy",metrics=["accuracy"])

model.fit(x\_train,y\_train,batch\_size=32,epochs=100)

model.save('phone.h5')

corpus=[]

with open("20191226-reviews.csv\count\_vec.pkl','rb') as file:

cv=pickle.load(file)

inp = "the phone is bad"

x=cv.transform([inp])

y=model.predict(x)

if(y>0.5):

print('good review')

else:

print('bad review')

from keras.models import load\_model

import numpy as np

import pickle

model=load\_model('phone.h5')

corpus=[]

with open('count\_vec.pkl','rb') as file:

cv=pickle.load(file)

inp = "the phone is bad"

x=cv.transform([inp])

y=model.predict(x)

if(y>0.5):

print('good review')

else:

print('bad review')

corpus=[]

with open('count\_vec.pkl','rb') as file:

cv=pickle.load(file)

inp = "the phone is ok"

x=cv.transform([inp])

y=model.predict(x)

if(y>0.5):

print('good review')

else:

print('bad review')

corpus=[]

with open('count\_vec.pkl','rb') as file:

cv=pickle.load(file)

inp = "bad and good"

x=cv.transform([inp])

y=model.predict(x)

if(y>0.5):

print('good review')

else:

print('bad review')

corpus=[]

with open('count\_vec.pkl','rb') as file:

cv=pickle.load(file)

inp = "bad and good"

x=cv.transform([inp])

y=model.predict(x)

if(y>0.5):

print('good review')

else:

print('bad review')

## ->our model predicted correctly , later built python code

from flask import render\_template, Flask, request,url\_for

from keras.models import load\_model

import pickle

import tensorflow as tf

graph = tf.get\_default\_graph()

with open(r'count\_vec.pkl','rb') as file:

cv=pickle.load(file)

cla = load\_model('phone.h5')

cla.compile(optimizer='adam',loss='binary\_crossentropy')

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

return render\_template('index.html')

@app.route('/predict')

@app.route('/', methods = ['GET','POST'])

def page2():

if request.method == 'GET':

img\_url = url\_for('static',filename = 'style/123.jpg')

return render\_template('index.html',url=img\_url)

if request.method == 'POST':

topic = request.form['tweet']

print("Hey " +topic)

topic=cv.transform([topic])

print("\n"+str(topic.shape)+"\n")

with graph.as\_default():

y\_pred = cla.predict(topic)

print("pred is "+str(y\_pred))

if(y\_pred > 0.5):

img\_url = url\_for('static',filename = 'style/1.jpg')

topic = "Positive Review"

elif(y\_pred < 0.5):

img\_url = url\_for('static',filename = 'style/2.jpg')

topic = "Negative Review"

return topic

if \_\_name\_\_ == '\_\_main\_\_':

app.run(host = 'localhost', debug = False , threaded = False)

### ->built HTML code for it!

%%HTML

<html>

<head>

<title>review amazon</title>

<style>

body {background-color:#2E86C1 ;

background-image: linear-gradient(-90deg,#2E86C1 ,#C02BB4 );}

h1{color:white;

font-size:48px;

float:left;

margin-left:60px;

text-decoration-line: underline;

text-decoration-style: solid;}

p{color:white;

font-size:20px;

}

input[type=text] {

border: 4px solid black;

border-radius: 20px;

}

body{

border: 8px solid black;

border-radius: 20px;

}

</style>

</head>

<body>

<form action="/login" method="POST">

<div class="home">

<h1>review amazon</h1>

<br><br><br>

<div class="image">

</div>

<center>

<div class="login">

<p><input type="text" name="review" required="required" style="height:200px;width:320px;"/></p>

<p>Enter the text message</p>

<div class="button">

<p><button type="submit">Predict</button>

</div>

</form>

<div>

<b><font color="black" size='10'></font></b>

</center>

</body>

</html>

return review

#### \* project done..model prediction successful ! \*