Assignment 8

NLP Sentiment Analysis

Name : Atharva Ramgirkar
Registration Number: 19BCE0114
Submission Date : 14 July, 2021
Program : VIT-AI Industry Certifiation
Email : atharva.ramgirkar2019@vitstudent.ac.in

Other Assignments can be found in the link:
https://drive.google.com/drive/folders/1QGOLHyZykoj_CroTJu6-YkZWf32JZ-QH?usp=sharing

Table of Content

In [ ]:

In [ ]:

In [ ]:

1. Importing Libraries

Back to Top

In [1]:
```
import pandas as pd
import numpy as np

# For NLP
import re
import nltk
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from sklearn.feature_extraction.text import CountVectorizer

# For Train Test Split
from sklearn.model_selection import train_test_split

# For Neural Network
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

# For handling Missing Values
import missingno as ms
import matplotlib.pyplot as plt

# Model Evaluation
from sklearn.metrics import accuracy_score
```

1.1 Initailizing Objects

In [2]:
```
ps = PorterStemmer()
cv = CountVectorizer(max_features=4000)
```

2. Reading Data

Back to Top

In [3]:
```
df = pd.read_csv("winemag-data-130k-v2.csv")
```

In [ ]:

3. Understanding Data

Back to Top

In [4]:
```
df.head()
```

Out[4]:

	Unnamed: 0	country	description	designation	points	price	province	region_1	region_2	taster_name
0	0	Italy	Aromas include tropical fruit, broom, brimston...	Vulkà Bianco	87	NaN	Sicily & Sardinia	Etna	NaN	Kerin O'Keefe
1	1	Portugal	This is ripe and fruity, a wine that is smooth...	Avidagos	87	15.0	Douro	NaN	NaN	Roger Voss
2	2	US	Tart and snappy, the flavors of lime flesh and...	NaN	87	14.0	Oregon	Willamette Valley	Willamette Valley	Paul Gregutt
3	3	US	Pineapple rind, lemon pith and orange blossom ...	Reserve Late Harvest	87	13.0	Michigan	Lake Michigan Shore	NaN	Alexander Peartree
4	4	US	Much like the regular bottling from 2012, this...	Vintner's Reserve Wild Child Block	87	65.0	Oregon	Willamette Valley	Willamette Valley	Paul Gregutt

In [5]:
```
df.columns
```

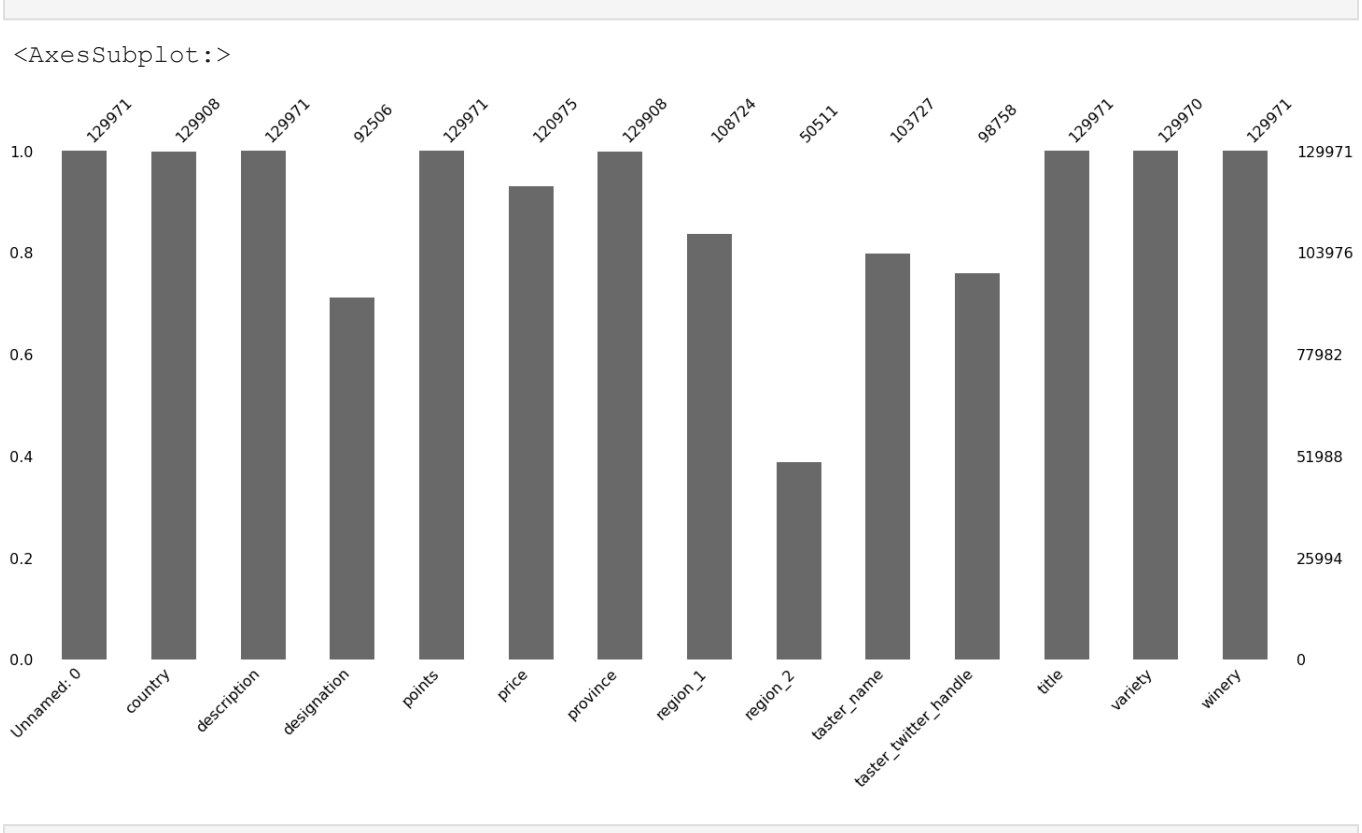
Out[5]:
```
Index(['Unnamed: 0', 'country', 'description', 'designation', 'points',
       'price', 'province', 'region_1', 'region_2', 'taster_name',
       'taster_twitter_handle', 'title', 'variety', 'winery'],
      dtype='object')
```

In [6]:
```
df.isnull().sum()
```

Out[6]:
```
Unnamed: 0                   0
country                     63
description                  0
designation              37465
points                       0
price                     8996
province                    63
region_1                 21247
region_2                 79460
taster_name              26244
taster_twitter_handle    31213
title                        0
variety                      1
winery                       0
dtype: int64
```

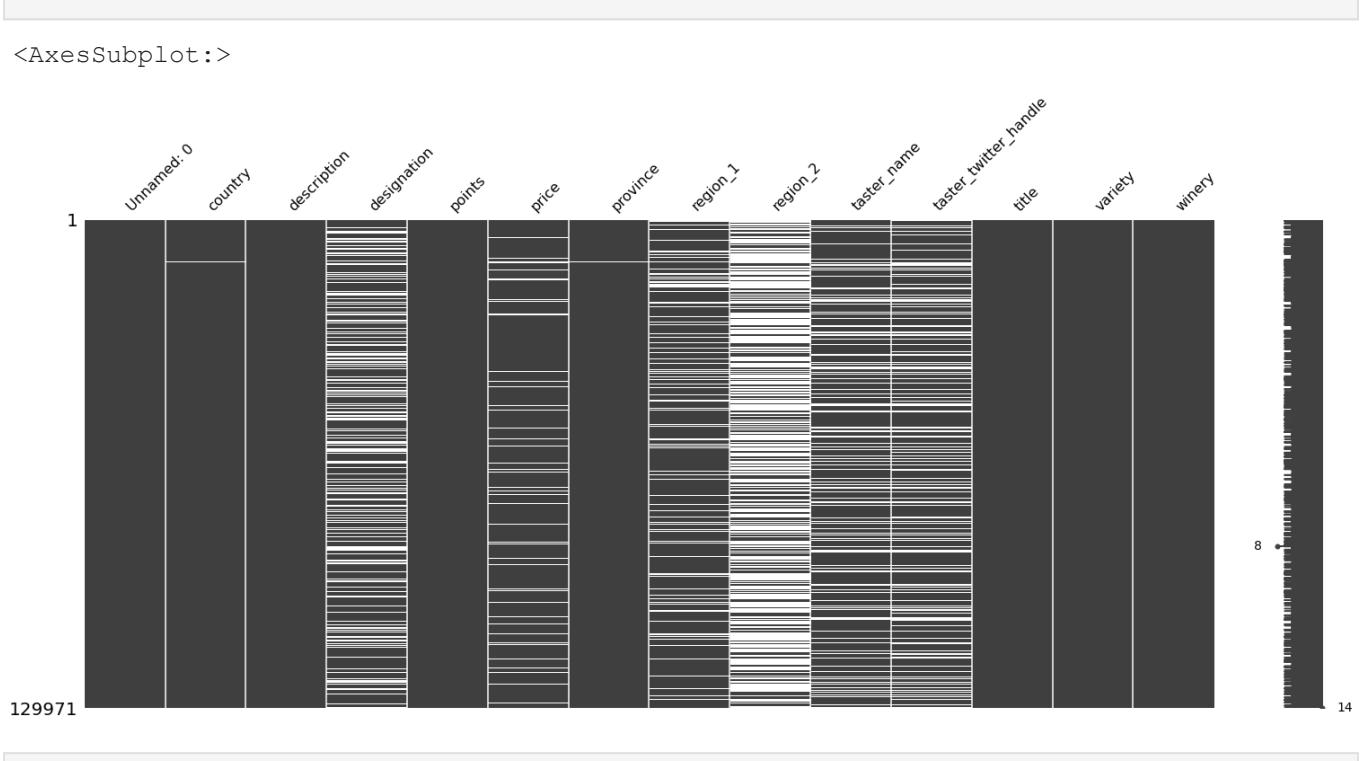
In [7]:
```
ms.bar(df)
```

Out[7]: <AxesSubplot:>



In [8]:
```
ms.matrix(df)
```

Out[8]: <AxesSubplot:>

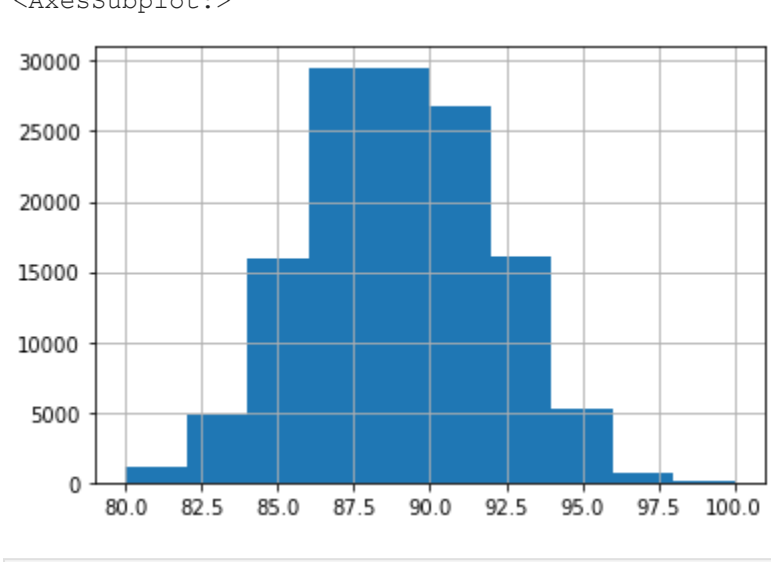


In [9]:
```
df.shape
```

Out[9]: (129971, 14)

In [10]:
```
df['points'].hist()
```

Out[10]: <AxesSubplot:>



In [11]:
```
df['country'].value_counts()
```

Out[11]:
```
US                        54504
France                    22093
Italy                     19540
Spain                      6645
Portugal                   5691
Chile                      4472
Argentina                  3800
Austria                    3345
Australia                  2329
Germany                    2165
New Zealand                1419
South Africa               1401
Israel                      505
Greece                      466
Canada                      257
Hungary                     146
Bulgaria                    141
Romania                     120
Uruguay                     109
Turkey                       90
Slovenia                     87
Georgia                      86
England                      74
Croatia                      73
Mexico                       70
Moldova                      59
Brazil                       52
Lebanon                      35
Morocco                      28
Peru                         16
Ukraine                      14
Macedonia                    12
Czech Republic               12
Serbia                       12
Cyprus                       11
India                         9
Switzerland                   7
Luxembourg                    6
Bosnia and Herzegovina        2
Armenia                       2
China                         1
Egypt                         1
Slovakia                      1
Name: country, dtype: int64
```

In [ ]:

In [ ]:

4. Dropping Unnecessary Columns

Back to Top

In [12]:
```
df.drop(['Unnamed: 0',
         'price',
         'taster_twitter_handle',
         'title',
         'designation',
         'region_1',
         'taster_name',
         'region_2'],
        axis=1,
        inplace=True)
```

5. Dropping Empty Rows

Back to Top

In [13]:
```
df = df.dropna()
```

In [ ]:

6. Modfying Columns

Back to Top

6.1 Encoding "points" Column

In [14]:
```
df['points'] = np.where(df['points']>90,1,0)
```

6.2 "country" Column

In [15]:
```
low_country = list(df['country'].value_counts().index)[12:]
```

In [16]:
```
for index, row in df.iterrows():
    if row['country'] in low_country:
        row['country'] = "Other_Countries"
```

6.3 "province" Column

In [17]:
```
low_pro = list(df['province'].value_counts().index)[40:]
```

In [18]:
```
for index, row in df.iterrows():
    if row['province'] in low_pro:
        row['province'] = "Other_Provinces"
```

In [19]:
```
df['province'] = np.where(df['province']=="Other","Other_Provinces",df['province'])
```

6.4 "province" Column

In [20]:
```
good_win = list(df['winery'].value_counts().index)[:2000]
```

In [21]:
```
for index, row in df.iterrows():
    if row['winery'] in good_win:
        row['winery'] = "Good_Winery"
    else:
        row['winery'] = "Bad_Winery"
```

6.5 "variety" Column

In [22]:
```
low_variety = list(df['variety'].value_counts().index)[30:]
```

In [23]:
```
for index, row in df.iterrows():
    if row['variety'] in low_variety:
        row['variety'] = "Other_Variety"
```

In [ ]:

7. Setting Feature and Target Columns

Back to Top

In [24]:
```
y = df['points'].values
```

In [25]:
```
y
```

Out[25]: array([0, 0, 0, ..., 0, 0, 0])

In [26]:
```
y
```

Out[26]: array([0, 0, 0, ..., 0, 0, 0])

In [27]:
```
X=df.drop('points',axis=1)
```

In [28]:
```
X
```

Out[28]:

	country	description	province	variety	winery
0	Italy	Aromas include tropical fruit, broom, brimston...	Sicily & Sardinia	White Blend	Nicosia
1	Portugal	This is ripe and fruity, a wine that is smooth...	Douro	Portuguese Red	Quinta dos Avidagos
2	US	Tart and snappy, the flavors of lime flesh and...	Oregon	Pinot Gris	Rainstorm
3	US	Pineapple rind, lemon pith and orange blossom ...	Michigan	Riesling	St. Julian
4	US	Much like the regular bottling from 2012, this...	Oregon	Pinot Noir	Sweet Cheeks
...	...	...	...	...	...
129966	Germany	Notes of honeysuckle and cantaloupe sweeten th...	Mosel	Riesling	Dr. H. Thanisch (Erben Müller-Burggraef)
129967	US	Citation is given as much as a decade of bottl...	Oregon	Pinot Noir	Citation
129968	France	Well-drained gravel soil gives this wine its c...	Alsace	Gewürztraminer	Domaine Gresser
129969	France	A dry style of Pinot Gris, this is crisp with ...	Alsace	Pinot Gris	Domaine Marcel Deiss
129970	France	Big, rich and off-dry, this is powered by inte...	Alsace	Gewürztraminer	Domaine Schoffit

129907 rows × 5 columns

In [29]:
```
df.reset_index(drop=True,inplace=True)
```

In [30]:
```
df.shape
```

Out[30]: (129907, 6)

In [31]:
```
df.tail()
```

Out[31]:

	country	description	points	province	variety	winery
129902	Germany	Notes of honeysuckle and cantaloupe sweeten th...	0	Mosel	Riesling	Dr. H. Thanisch (Erben Müller-Burggraef)
129903	US	Citation is given as much as a decade of bottl...	0	Oregon	Pinot Noir	Citation
129904	France	Well-drained gravel soil gives this wine its c...	0	Alsace	Gewürztraminer	Domaine Gresser
129905	France	A dry style of Pinot Gris, this is crisp with ...	0	Alsace	Pinot Gris	Domaine Marcel Deiss
129906	France	Big, rich and off-dry, this is powered by inte...	0	Alsace	Gewürztraminer	Domaine Schoffit

8. NLP on Data

Back to Top

In [ ]:

In [32]:
```
df['description'][129906]
```

Out[32]:
```
'Big, rich and off-dry, this is powered by intense spiciness and rounded texture. Lych
ees dominate the fruit profile, giving an opulent feel to the aftertaste. Drink now.'
```

In [ ]:

In [33]:
```
data = []
for i in range(0,129906):
    rev = df['description'][i]

    # Removing Special Characters
    rev = re.sub('[^a-zA-Z]'," ",rev)

    # Converting to Lower Case
    rev = rev.lower()

    # Spliting Sentences to List of Words
    rev = rev.split()

    # Stemming and Stop Word Removal
    rev = [ps.stem(word) for word in rev if not word in set(stopwords.words('english')

    # Re-Forming Sentence
    rev = " ".join(rev)

    # Appending to Corpus
    data.append(rev)
```

In [34]:
```
len(data)
```

Out[34]: 129906

In [35]:
```
data[0:5]
```

Out[35]:
```
['aroma includ tropic fruit broom brimston dri herb palat overli express offer unripen
appl citru dri sage alongsid brisk acid',
 'ripe fruiti wine smooth still structur firm tannin fill juici red berri fruit freshe
n acid alreadi drinkabl although certainli better',
 'tart snappi flavor lime flesh rind domin green pineappl poke crisp acid underscor fl
avor wine stainless steel ferment',
 'pineappl rind lemon pith orang blossom start aroma palat bit opul note honey drizzl
guava mango give way slightli astring semidri finish',
 'much like regular bottl come across rather rough tannic rustic earthi herbal charact
erist nonetheless think pleasantli unfussi countri wine good companion hearti winter s
tew']
```

In [36]:
```
for i in range(129907):
    try:
        data[i] = data[i] +" "+df.loc[i,('country')]+" "+df.loc[i,('province')]+" "+d
    except:
        pass
```

In [37]:
```
data[:5]
```

Out[37]:
```
['aroma includ tropic fruit broom brimston dri herb palat overli express offer unripen
appl citru dri sage alongsid brisk acid Italy Sicily & Sardinia White Blend Nicosia',
 'ripe fruiti wine smooth still structur firm tannin fill juici red berri fruit freshe
n acid alreadi drinkabl although certainli better Portugal Douro Portuguese Red Quinta
dos Avidagos',
 'tart snappi flavor lime flesh rind domin green pineappl poke crisp acid underscor fl
avor wine stainless steel ferment US Oregon Pinot Gris Rainstorm',
 'pineappl rind lemon pith orang blossom start aroma palat bit opul note honey drizzl
guava mango give way slightli astring semidri finish US Michigan Riesling St. Julian',
 'much like regular bottl come across rather rough tannic rustic earthi herbal charact
erist nonetheless think pleasantli unfussi countri wine good companion hearti winter s
tew US Oregon Pinot Noir Sweet Cheeks']
```

In [38]:
```
X = cv.fit_transform(data).toarray()
```

In [ ]:

9. Train Test Split

Back to Top

In [41]:
```
X.shape
```

Out[41]: (129906, 4000)

In [48]:
```
y = y[:129906]
```

In [49]:
```
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.3,random_state=114,st
```

In [ ]:

10. Building Model

Back to Top

10.1 Initializing Model

In [50]:
```
model = Sequential()
```

10.2 Adding Hidden Layers

In [51]:
```
model.add(Dense(units = 4000,
                kernel_initializer="random_uniform",
                activation="relu"))

model.add(Dense(units = 3000,
                kernel_initializer="random_uniform",
                activation="relu"))

model.add(Dense(units = 1000,
                kernel_initializer="random_uniform",
                activation="relu"))
```

10.3 Adding Output Layer

In [52]:
```
model.add(Dense(units = 1,
                kernel_initializer="random_uniform",
                activation="sigmoid"))
```

In [ ]:

11. Compiling the Model

Back to Top

In [53]:
```
model.compile(optimizer="adam",
              loss="binary_crossentropy",
              metrics=['accuracy'])
```

In [ ]:

12. Training the Model

Back to Top

In [54]:
```
model.fit(X_train,y_train,epochs=1)
```

```
2842/2842 [==============================] - 647s 228ms/step - loss: 0.3334 - accurac
y: 0.8511
```

Out[54]: <tensorflow.python.keras.callbacks.History at 0x25eb0ee9b50>

In [ ]:

13. Testing the Model

Back to Top

In [55]:
```
pred = model.predict(X_test)
```

In [56]:
```
pred = pred>0.5
```

14. Model Accurecy

Back to Top

In [57]:
```
accuracy_score(y_test, pred)
```

Out[57]: 0.8661603202299086

In [ ]:

15. Single Predictions

Back to Top

In [58]:
```
model.predict(cv.transform(["Italy Bordeaux-style Red Blend Bad_Winery"]))
```

Out[58]: array([[0.04960445]], dtype=float32)

In [59]:
```
model.predict(cv.transform(["Wine India bad"]))
```

Out[59]: array([[0.08979273]], dtype=float32)

In [60]:
```
model.predict(cv.transform(["Wine Italy Best"]))
```

Out[60]: array([[0.15743548]], dtype=float32)