sentimentana

June 28, 2024

[1]:
```
import numpy as np
import pandas as pd
import os
import tensorflow as tf
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.models import Sequential
```

[2]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

[3]:
```
df = pd.read_csv("/content/drive/MyDrive/zomato_senti/zom_rev.csv")
```

[4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 103183 entries, 0 to 103182
Data columns (total 3 columns):
 #   Column      Non-Null Count   Dtype
---  ------      --------------   -----
 0   Unnamed: 0  103183 non-null  int64
 1   review      103183 non-null  object
 2   rating      103183 non-null  float64
dtypes: float64(1), int64(1), object(1)
memory usage: 2.4+ MB
```

[5]:
```
null_values = df.isnull().sum()
print("Null values in the entire Data:")
print(null_values)
```

```
Null values in the entire Data:
Unnamed: 0    0
review        0
rating        0
dtype: int64
```

[6]:
```
df.dropna(inplace=True)
```

[7]:
```
null_values = df.isnull().sum()
null_values
```

[7]:
```
Unnamed: 0    0
review        0
rating        0
dtype: int64
```

[8]:
```
df.drop_duplicates(inplace=True)
```

[9]:
```
import string
df['review'] = df['review'].apply(lambda x: x.lower())
df['review'] = df['review'].apply(lambda x: x.translate(str.maketrans('',␣
 ↪'',string.punctuation)))
```

[10]:
```
df['review']
```

[10]:
```
0         one stop place for every foodie as you get var…
1         punjab sweets corner is a place where youll fi…
2         customer misbehave with me and the sandwich wa…
3         i had worst pizza from this food corner qualit…
4         i recently visited this place in karol bagh an…
                              …
103178    pretty underrated they serve really great food…
103179    tried their family veg combo  the dal and pane…
103180    today i tasted thalli from street food it was …
103181    random sunday afternoon at ambiance mall lunch…
103182    if they would have name this place as punjab g…
Name: review, Length: 103183, dtype: object
```

[11]:
```
from sklearn.feature_extraction.text import CountVectorizer
# Assuming 'df' is your Data containing text data
text_data = df['review']
vectorizer = CountVectorizer()
feature_matrix = vectorizer.fit_transform(text_data)
feature_names = vectorizer.get_feature_names_out()
```

[12]:
```
feature_names
```

[12]:
```
array(['00', '000', '01', …, '   ', '    ', '   '], dtype=object)
```

[13]:
```
import sklearn.feature_extraction.text as text
count_vectorizer = text.CountVectorizer()
```

[14]:
```
count_vectorizer.fit(df.review)
```

[14]: CountVectorizer()

[15]:
```
data_features = count_vectorizer.transform(df.review)
```

[16]:
```
density = [(data_features.getnnz() * 100) / (data_features.shape[0] *␣
↪data_features.shape[1])]
print("Density of the matrix: ", density)
```

```
Density of the matrix:  [0.045246976413245765]
```

[17]:
```
feature_counts = df['review'].value_counts()
feature_counts
```

[17]:
```
review
good
486
good food
155
nice
133
very good
67
awesome
67
                                                …
i ordered a fresh fruit cake for my spouse’s birthday and i must say the cake
was too awesome and very refreshing as the name suggests it was loaded with
fresh seasonal fruits pricing is reasonable for the taste delivered will
definitely recommend it
1
such a big cheater this brand is i had ordered veg burgers as you can see in the
pics i have shared and i was delivered a chickenmutton burger on 14th sept 2020
just imagine that i am a pure veg brahmin and during sharad time they did this
blunder of delivering non veg to my house my younger brother ate half of it when
we noticed this no one was ready to support or take any action even be smart and
just dont order from this place or platform      1
the only restaurant  bakery house i go for any kind of celebration let it be a
cake or burgers to pizza to chinese to mughlai they make everything so
great\nthe staff is really helpful and the cakes n brownies just to die for\n
will definitely recommend others for sure\n  cakes burgers mughlai  chinese
chaat top recommendations\n  2 thumbs up for sure
1
i have been ordering from this place since my childhood  im so happy that they
are serving their best since then  have maintained their taste n standards \n
recommendations \n  cakes pastry burgers  chinese chaat\ntwo thumbs up from my
side too
1
```

```
if they would have name this place as punjab grill express it would have really
justified the name  the food is similar to what you get in punjab grill and this
is wrong business decision to occupy a food court spot  they are selling their
product at a discounted rate and loosing the exclusivity  lite food bites
already dominates the mall with 4 restaurants
1
Name: count, Length: 99674, dtype: int64
```

[18]:
```
features = vectorizer.get_feature_names_out() # Replace with the variable that␣
 ↪holds feature names
features_counts = np.sum(data_features.toarray(), axis=0)
features_counts_df = pd.DataFrame({'features': features, 'counts':
 ↪features_counts})
```

[19]:
```
count_of_single_occurrences␣
 ↪=len(features_counts_df[features_counts_df['counts'] == 1])
count_of_single_occurrences
```

[19]:
```
41047
```

[20]:
```
count_vectorizer = CountVectorizer(max_features=10000)
feature_vector = count_vectorizer.fit_transform(df['review'])
features = count_vectorizer.get_feature_names_out()
data_features = feature_vector.toarray()
features_counts = np.sum(data_features, axis=0)
feature_counts = pd.DataFrame({'features': features, 'counts': features_counts})
```

[21]:
```
top_features_counts = feature_counts.sort_values('counts', ascending=False).
 ↪head(15)
```

[22]:
```
top_features_counts
```

[22]:

	features	counts
8943	the	178962
525	and	160064
9654	was	98264
4593	is	74757
9094	to	69172
3475	food	66285
6135	of	62019
4606	it	53472
4437	in	47664
3840	good	46586
3541	for	43922
8993	this	42477
9810	with	39589
6693	place	38985

```
9526     very   33329
```

[23]:
```
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')
english_stop_words = stopwords.words('english')
df['review'][0:10]
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data…
[nltk_data]   Unzipping corpora/stopwords.zip.
```

[23]:
```
0    one stop place for every foodie as you get var…
1    punjab sweets corner is a place where youll fi…
2    customer misbehave with me and the sandwich wa…
3    i had worst pizza from this food corner qualit…
4    i recently visited this place in karol bagh an…
5    v bad khana and payment 2plat but delvari 1 pl…
6    unfresh base less cheese and not up to the mar…
7    not even sufficient for single personvery poor…
8    we recently visited this place and had an amaz…
9    we recently visited this place and had an time…
Name: review, dtype: object
```

[28]:
```
# ... (Your code for model training and prediction)

# Assuming you have a trained model called 'model' and test data 'X_test'
y_pred = model.predict(X_test)

# Assuming 'y_test' contains the true labels for the test data
cm = confusion_matrix(y_test, y_pred)

# ... (Rest of your code for plotting the confusion matrix)

import seaborn as sns
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, cmap='Blues', fmt='d')
plt.title('Confusion Matrix')
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.show()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-28-99578a501892> in <cell line: 4>()
```

```
      2
      3 # Assuming you have a trained model called 'model' and test data 'X_tes'
----> 4 y_pred = model.predict(X_test)
      5
      6 # Assuming 'y_test' contains the true labels for the test data

NameError: name 'X_test' is not defined
```

[27]:
```
# ... (Your code for model training and prediction)

# Define and train your model here
from sklearn.linear_model import LogisticRegression  # Example model
model = LogisticRegression()
model.fit(X_train, y_train)  # Assuming you have training data 'X_train' and␣
 ↪'y_train'

# Now you can predict using the trained model
y_pred = model.predict(X_test)

# Assuming 'y_test' contains the true labels for the test data
cm = confusion_matrix(y_test, y_pred)

# ... (Rest of your code for plotting the confusion matrix)

import seaborn as sns
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, cmap='Blues', fmt='d')
plt.title('Confusion Matrix')
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.show()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-27-2ed1734243b9> in <cell line: 6>()
      4 from sklearn.linear_model import LogisticRegression  # Example model
      5 model = LogisticRegression()
----> 6 model.fit(X_train, y_train)  # Assuming you have training data 'X_train␣
 ↪and 'y_train'
      7
      8 # Now you can predict using the trained model

NameError: name 'X_train' is not defined
```

[33]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report
X_train, X_test, y_train, y_test = train_test_split(df['review'],df['rating'],␣
 ↪test_size=0.2, random_state=42)
vectorizer = CountVectorizer()
X_train_vectorized = vectorizer.fit_transform(X_train)
X_test_vectorized = vectorizer.transform(X_test)
model = RandomForestClassifier()

y_pred = model.predict(X_test_vectorized)

accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)
print("Accuracy: ", accuracy)
print("Classification Report:\n", report)
```

```
---------------------------------------------------------------------------
NotFittedError                            Traceback (most recent call last)
<ipython-input-33-b5a327f003d5> in <cell line: 10>()
      8 model = RandomForestClassifier()
      9
---> 10 y_pred = model.predict(X_test_vectorized)
     11
     12 accuracy = accuracy_score(y_test, y_pred)

/usr/local/lib/python3.10/dist-packages/sklearn/ensemble/_forest.py in␣
 ↪predict(self, X)
    818             The predicted classes.
    819         """
--> 820         proba = self.predict_proba(X)
    821
    822         if self.n_outputs_ == 1:

/usr/local/lib/python3.10/dist-packages/sklearn/ensemble/_forest.py in␣
 ↪predict_proba(self, X)
    858             classes corresponds to that in the attribute :term:
 ↪`classes_`.
    859         """
--> 860         check_is_fitted(self)
    861         # Check data
    862         X = self._validate_X_predict(X)

/usr/local/lib/python3.10/dist-packages/sklearn/utils/validation.py in␣
 ↪check_is_fitted(estimator, attributes, msg, all_or_any)
   1388
   1389     if not fitted:
```

```
-> 1390         raise NotFittedError(msg % {"name": type(estimator).__name__})
   1391
   1392

NotFittedError: This RandomForestClassifier instance is not fitted yet. Call␣
 ↪'fit' with appropriate arguments before using this estimator.
```

[ ]:
```
import matplotlib.pyplot as plt
import seaborn as sns
sns.histplot(df['product_price'])
plt.title('Product Price')
plt.show()
```

[ ]:
```
sns.countplot(data=df, x='Sentiment')
plt.title('Sentiment Anaysis')
plt.show()
```

[ ]:
```
import matplotlib.pyplot as plt
plt.figure(figsize=(12, 5))
plt.hist(features_counts_df['counts'], bins=50, range=(0, 5000))
plt.xlabel('Frequency of Words')
plt.ylabel('Density')
plt.show()
```