nkxqosbud

October 26, 2023

Naan Mudhalva Phase4 Development: Part2

Sentiment Analysis for Marketing

[32]:
```
pip install transformers
```

```
Requirement already satisfied: transformers in /usr/local/lib/python3.10/dist-
packages (4.34.1)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-
packages (from transformers) (3.12.4)
Requirement already satisfied: huggingface-hub<1.0,>=0.16.4 in
/usr/local/lib/python3.10/dist-packages (from transformers) (0.17.3)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-
packages (from transformers) (1.23.5)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.10/dist-packages (from transformers) (23.2)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-
packages (from transformers) (6.0.1)
Requirement already satisfied: regex!=2019.12.17 in
/usr/local/lib/python3.10/dist-packages (from transformers) (2023.6.3)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-
packages (from transformers) (2.31.0)
Requirement already satisfied: tokenizers<0.15,>=0.14 in
/usr/local/lib/python3.10/dist-packages (from transformers) (0.14.1)
Requirement already satisfied: safetensors>=0.3.1 in
/usr/local/lib/python3.10/dist-packages (from transformers) (0.4.0)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.10/dist-
packages (from transformers) (4.66.1)
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages
(from huggingface-hub<1.0,>=0.16.4->transformers) (2023.6.0)
Requirement already satisfied: typing-extensions>=3.7.4.3 in
/usr/local/lib/python3.10/dist-packages (from huggingface-
hub<1.0,>=0.16.4->transformers) (4.5.0)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests->transformers) (3.3.0)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests->transformers) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests->transformers) (2.0.7)
```

```
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from requests->transformers)
(2023.7.22)
```

[33]:
```
from PIL import Image
from sklearn import svm
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics import accuracy_score, classification_report
from sklearn.metrics import roc_curve
from sklearn.naive_bayes import MultinomialNB
from sklearn.neighbors import KNeighborsClassifier
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
from transformers import BertTokenizer, BertForSequenceClassification, pipeline
from sklearn.model_selection import train_test_split
from transformers import pipeline

import collections
import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np
import operator
import pandas as pd
import torch

from transformers import BertTokenizer, BertForSequenceClassification, AdamW
from torch.utils.data import DataLoader, Dataset
from torch.nn import functional as F
import matplotlib.pyplot as plt
```

[34]:
```
tweets = pd.read_csv('Tweets.csv')
```

[35]:
```
tweets.head(5)
```

[35]:
```
             tweet_id airline_sentiment  airline_sentiment_confidence  \
0  570306133677760513           neutral                        1.0000
1  570301130888122368          positive                        0.3486
2  570301083672813571           neutral                        0.6837
3  570301031407624196          negative                        1.0000
4  570300817074462722          negative                        1.0000

  negativereason  negativereason_confidence         airline  \
0            NaN                        NaN  Virgin America
1            NaN                     0.0000  Virgin America
2            NaN                        NaN  Virgin America
3     Bad Flight                     0.7033  Virgin America
4     Can't Tell                     1.0000  Virgin America
```

```
  airline_sentiment_gold        name negativereason_gold  retweet_count  \
0                    NaN     cairdin                 NaN              0
1                    NaN    jnardino                 NaN              0
2                    NaN  yvonnalynn                 NaN              0
3                    NaN    jnardino                 NaN              0
4                    NaN    jnardino                 NaN              0

                                              text tweet_coord  \
0              @VirginAmerica What @dhepburn said.         NaN
1  @VirginAmerica plus you've added commercials t…        NaN
2  @VirginAmerica I didn't today… Must mean I n…        NaN
3  @VirginAmerica it's really aggressive to blast…        NaN
4  @VirginAmerica and it's a really big bad thing…        NaN

               tweet_created tweet_location               user_timezone
0  2015-02-24 11:35:52 -0800            NaN  Eastern Time (US & Canada)
1  2015-02-24 11:15:59 -0800            NaN  Pacific Time (US & Canada)
2  2015-02-24 11:15:48 -0800      Lets Play  Central Time (US & Canada)
3  2015-02-24 11:15:36 -0800            NaN  Pacific Time (US & Canada)
4  2015-02-24 11:14:45 -0800            NaN  Pacific Time (US & Canada)
```

[36]:
```
tweets['negativereason_gold'].value_counts()
```

[36]:
```
Customer Service Issue                      12
Late Flight                                  4
Can't Tell                                   3
Cancelled Flight                             3
Cancelled Flight\nCustomer Service Issue     2
Late Flight\nFlight Attendant Complaints     1
Late Flight\nLost Luggage                    1
Bad Flight                                   1
Lost Luggage\nDamaged Luggage                1
Late Flight\nCancelled Flight                1
Flight Attendant Complaints                  1
Customer Service Issue\nLost Luggage         1
Customer Service Issue\nCan't Tell           1
Name: negativereason_gold, dtype: int64
```

[37]:
```
tweets['airline_sentiment_gold'].value_counts()
```

[37]:
```
negative    32
positive     5
neutral      3
Name: airline_sentiment_gold, dtype: int64
```

[38]:
```
tweets['retweet_count'].value_counts()
```

[38]:
```
0     13873
1       640
2        66
3        22
4        17
5         5
7         3
6         3
22        2
8         1
32        1
28        1
9         1
18        1
11        1
31        1
15        1
44        1
Name: retweet_count, dtype: int64
```

[39]:
```
tweets.drop('negativereason_gold', axis=1, inplace=True)
tweets.drop('airline_sentiment_gold', axis=1, inplace=True)
tweets.drop('retweet_count', axis=1, inplace=True)
tweets.drop('tweet_coord', axis=1, inplace=True)

tweets.drop('tweet_location', axis=1, inplace=True)
tweets.drop('tweet_created', axis=1, inplace=True)
tweets.drop('user_timezone', axis=1, inplace=True)
tweets.drop('name', axis=1, inplace=True)

list(tweets.columns)
```

[39]:
```
['tweet_id',
 'airline_sentiment',
 'airline_sentiment_confidence',
 'negativereason',
 'negativereason_confidence',
 'airline',
 'text']
```

[40]:
```
unmeaningful = ['i', 'you', 'me', 'to', 'the', 'a', 'my', 'is', 'in', 'and',␣
 ↪'for', 'on', 'of',
                'your', 'so', 'was', 'have', 'it', 'at', 'with', 'that',␣
 ↪'from', 'do', 'get',
                'but', 'this', 'can', 'just', 'they', 'we', 'are', 'an', 'be',␣
 ↪"i'm", 'will',
                'if', 'had', 'our', 'about', 'there', 'has', 'been', '-', 'by',␣
 ↪'like', 'or',
                'as', 'he', 'she', 'it', 'us', 'has' ,"i've", "it's", "don't",␣
 ↪'would', 'am',
                'flight', 'customer', 'any', 'very', "didn't", "you've",␣
 ↪'thing', 'take',
                'other', 'u', '', ' ']
```

[41]:
```
def clean_text(str_in):

    res = ""
    str_in = str_in.lower()
    str_arr = str_in.split(' ')
    for word in str_arr:

        word = word.lower()

        if '@' in word or word == '' or word[:1] == '&':
            continue
        if word.lower() in unmeaningful:
            continue
        if word.isnumeric():
            continue
        res = res + " " + word
    return res
```

[42]:
```
tweets["text"] = tweets["text"].apply(clean_text)
```

[43]:
```
tweets.head(5)
```

[43]:
```
             tweet_id airline_sentiment  airline_sentiment_confidence  \
0  570306133677760513           neutral                        1.0000
1  570301130888122368          positive                        0.3486
2  570301083672813571           neutral                        0.6837
3  570301031407624196          negative                        1.0000
4  570300817074462722          negative                        1.0000

  negativereason  negativereason_confidence         airline  \
0            NaN                        NaN  Virgin America
1            NaN                     0.0000  Virgin America
2            NaN                        NaN  Virgin America
3    Bad Flight                      0.7033  Virgin America
4    Can't Tell                      1.0000  Virgin America

                                         text
0                                  what said.
1        plus added commercials experience… tacky.
```

```
2               today… must mean need another trip!
3   really aggressive blast obnoxious "entertainm…
4                                       really big bad
```

[57]:
```
data = tweets
data['airline_sentiment'] = data['airline_sentiment'].astype('category')

X = data['text']
y = data['airline_sentiment']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)


tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')


X_train_encodings = tokenizer(list(X_train), padding=True, truncation=True,␣
 ↪return_tensors='pt', max_length=128)
X_test_encodings = tokenizer(list(X_test), padding=True, truncation=True,␣
 ↪return_tensors='pt', max_length=128)
y_train_encodings = torch.tensor(y_train.cat.codes.values)
y_test_encodings = torch.tensor(y_test.cat.codes.values)
```

[47]:
```
class SentimentDataset(Dataset):
    def __init__(self, encodings, labels):
        self.encodings = encodings
        self.labels = labels

    def __getitem__(self, idx):
        item = {key: val[idx] for key, val in self.encodings.items()}
        item['labels'] = self.labels[idx]
        return item

    def __len__(self):
        return len(self.labels)
```

[48]:
```
train_dataset = SentimentDataset(X_train_encodings, y_train_encodings)
test_dataset = SentimentDataset(X_test_encodings, y_test_encodings)
```

[49]:
```
model = BertForSequenceClassification.from_pretrained('bert-base-uncased',␣
 ↪num_labels=3)
optimizer = AdamW(model.parameters(), lr=1e-5)
```

```
Downloading model.safetensors:   0%|          | 0.00/440M [00:00<?, ?B/s]
```

```
Some weights of BertForSequenceClassification were not initialized from the
model checkpoint at bert-base-uncased and are newly initialized:
```

```
['classifier.bias', 'classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use it
for predictions and inference.
/usr/local/lib/python3.10/dist-packages/transformers/optimization.py:411:
FutureWarning: This implementation of AdamW is deprecated and will be removed in
a future version. Use the PyTorch implementation torch.optim.AdamW instead, or
set `no_deprecation_warning=True` to disable this warning
  warnings.warn(
```

[50]:
```
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
model.to(device)
train_loader = DataLoader(train_dataset, batch_size=32, shuffle=True)
model.train()
```

[50]:
```
BertForSequenceClassification(
  (bert): BertModel(
    (embeddings): BertEmbeddings(
      (word_embeddings): Embedding(30522, 768, padding_idx=0)
      (position_embeddings): Embedding(512, 768)
      (token_type_embeddings): Embedding(2, 768)
      (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (encoder): BertEncoder(
      (layer): ModuleList(
        (0-11): 12 x BertLayer(
          (attention): BertAttention(
            (self): BertSelfAttention(
              (query): Linear(in_features=768, out_features=768, bias=True)
              (key): Linear(in_features=768, out_features=768, bias=True)
              (value): Linear(in_features=768, out_features=768, bias=True)
              (dropout): Dropout(p=0.1, inplace=False)
            )
            (output): BertSelfOutput(
              (dense): Linear(in_features=768, out_features=768, bias=True)
              (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
              (dropout): Dropout(p=0.1, inplace=False)
            )
          )
          (intermediate): BertIntermediate(
            (dense): Linear(in_features=768, out_features=3072, bias=True)
            (intermediate_act_fn): GELUActivation()
          )
          (output): BertOutput(
            (dense): Linear(in_features=3072, out_features=768, bias=True)
            (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
            (dropout): Dropout(p=0.1, inplace=False)
          )
        )
      )
    )
    (pooler): BertPooler(
      (dense): Linear(in_features=768, out_features=768, bias=True)
      (activation): Tanh()
    )
  )
  (dropout): Dropout(p=0.1, inplace=False)
  (classifier): Linear(in_features=768, out_features=3, bias=True)
)
```

[53]:
```
model.eval()
test_loader = DataLoader(test_dataset, batch_size=64)
predictions = []
```

[54]:
```
for batch in test_loader:
    input_ids = batch['input_ids'].to(device)
    attention_mask = batch['attention_mask'].to(device)
    with torch.no_grad():
        outputs = model(input_ids, attention_mask=attention_mask)
    logits = outputs.logits
    predicted_labels = F.softmax(logits, dim=1).argmax(dim=1)
    predictions.extend(predicted_labels.cpu().numpy())
```

[55]:
```
accuracy = accuracy_score(y_test_encodings, predictions)
print("Accuracy:", accuracy)
print(classification_report(y_test_encodings, predictions))
```

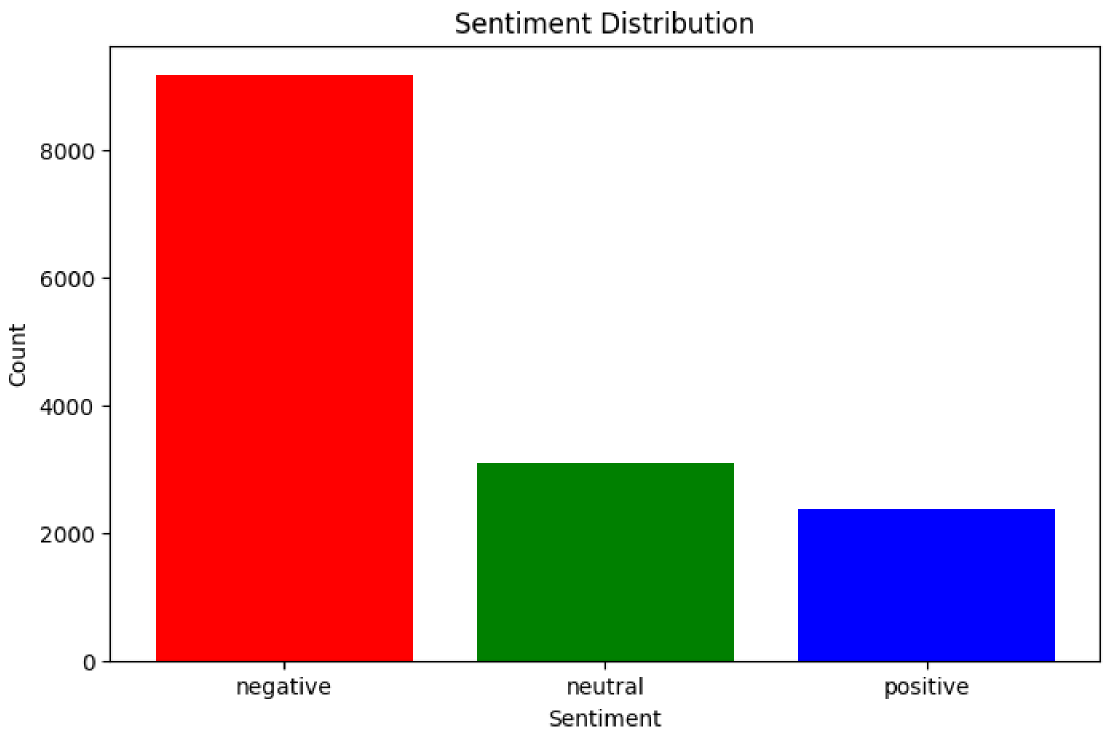
```
Accuracy: 0.6441256830601093
              precision    recall  f1-score   support

           0       0.64      1.00      0.78      1889
           1       0.00      0.00      0.00       580
           2       0.00      0.00      0.00       459

    accuracy                           0.64      2928
   macro avg       0.21      0.33      0.26      2928
weighted avg       0.42      0.64      0.51      2928
```

[56]:
```
sentiment_counts = data['airline_sentiment'].value_counts()
plt.figure(figsize=(8, 5))
plt.bar(sentiment_counts.index, sentiment_counts.values, color=['red', 'green',␣
 ↪'blue'])
plt.title('Sentiment Distribution')
```

```python
plt.xlabel('Sentiment')
plt.ylabel('Count')
plt.show()
```


Sentiment Distribution
Count
negative
neutral
positive
Sentiment