text-1

April 22, 2023

1 Portfolio Assignment: Text Classification

1.0.1 Abdullah Hasani - AHH190004

Text classification using the Yelp Reviews Sentiment Dataset to gain experience with deep learning model variations and embeddings.

This data set classifies Yelp reviews based on their general sentiment. The model should be able to accurately predict the sentiment of a text based on the review, for instance, if a review is very negative the model should predict the result to be 0, otherwise if the review is overwhelmingly positive it should be able to confidently predict 1. If it is a neutral review, the classification will change depending on the language used in the review.

[5]:
```
import pandas as pd
import csv
import numpy as np
import tensorflow as tf
from tensorflow.keras import datasets, layers, models
from sklearn.preprocessing import LabelEncoder
import warnings

# warnings.filterwarnings('ignore')

# Load the dataset
df_train = pd.read_csv('train.csv', error_bad_lines=False)
df_test = pd.read_csv('test.csv', error_bad_lines=False)

# remove rows with null or missing values
df_train.dropna(subset=['text'], inplace=True)
df_test.dropna(subset=['text'], inplace=True)

print(df_test.head())
```

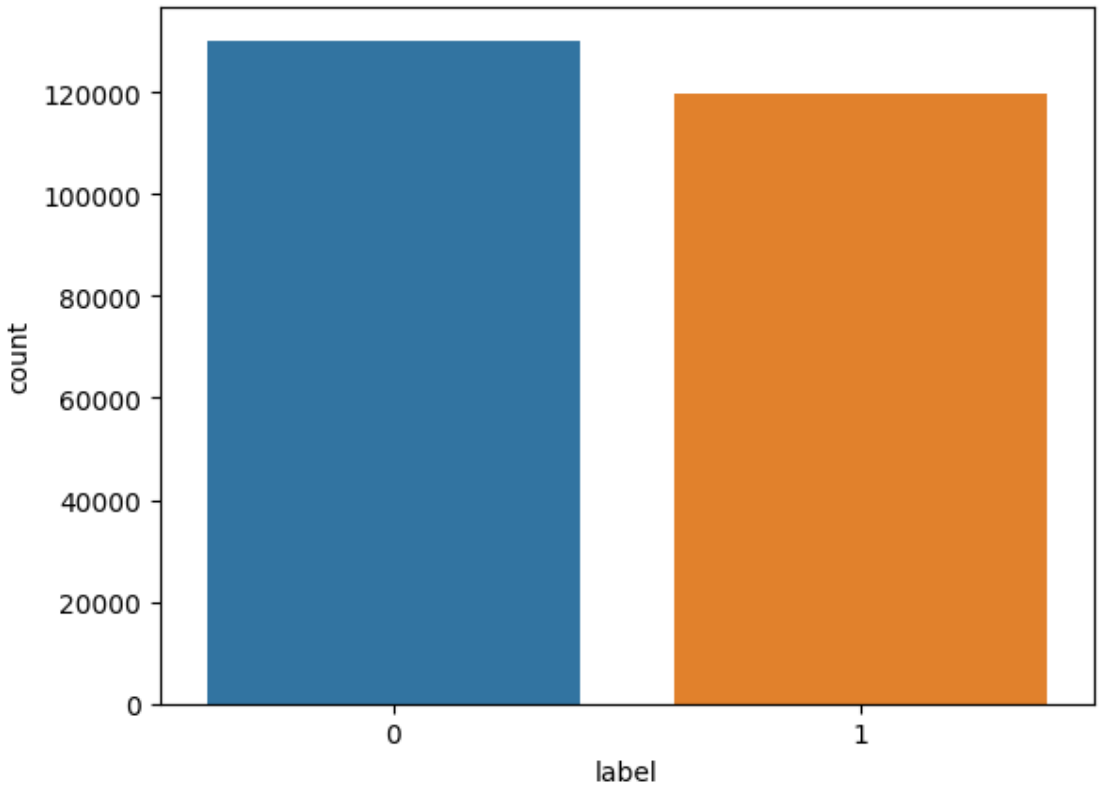
```
                                                text  label
0  Contrary to other reviews, I have zero complai…      1
1  Last summer I had an appointment to get new ti…      0
2  Friendly staff, same starbucks fair you get an…      1
3  The food is good. Unfortunately the service is…      0
4  Even when we didn't have a car Filene's Baseme…      1
```

[6]:
```
print('Size of training and test data:', df_train.shape, df_test.shape)
```

```
Size of training and test data: (249999, 2) (38000, 2)
```

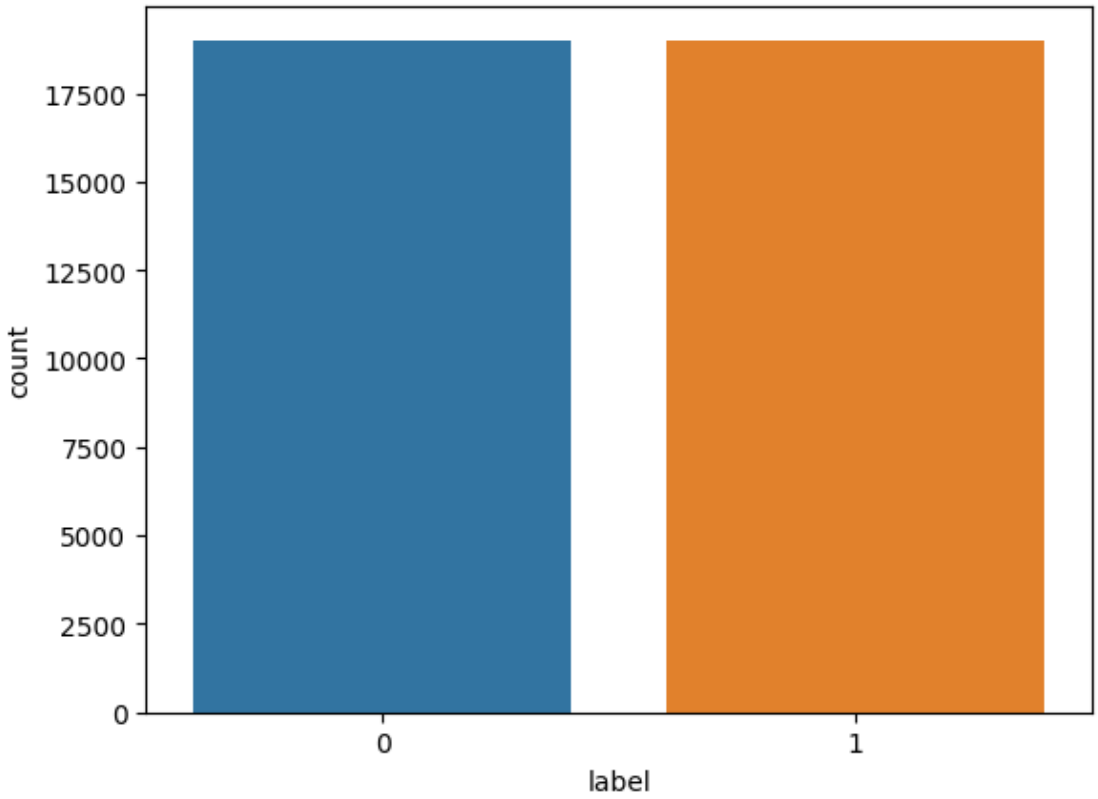
[7]:
```
import seaborn as sb
sb.countplot(x = df_train['label'])
```

[7]: <Axes: xlabel='label', ylabel='count'>



[8]:
```
sb.countplot(x = df_test['label'])
```

[8]: <Axes: xlabel='label', ylabel='count'>



[9]:
```
print(df_train['text'].isnull().sum())
print(df_test['text'].isnull().sum())
```

```
0
0
```

[ ]:
```
from tensorflow.keras.preprocessing.text import Tokenizer

# set up X and Y
num_labels = 2
vocab_size = 25000
batch_size = 128

# fit the tokenizer on the training data
tokenizer = Tokenizer(num_words=vocab_size)
tokenizer.fit_on_texts(df_train['text'])

x_train = tokenizer.texts_to_matrix(df_train['text'], mode='tfidf')
x_test = tokenizer.texts_to_matrix(df_test['text'], mode='tfidf')

encoder = LabelEncoder()
```

```
encoder.fit(df_train['label'])

y_train = encoder.transform(df_train['label'])
y_test = encoder.transform(df_test['label'])

# print shapes
print("train shapes:", x_train.shape, y_train.shape)
print("test shapes:", x_test.shape, y_test.shape)
print("test first five labels:", y_test[:5])
```

1.1 Sequential Model

[ ]:
```
from tensorflow.keras import models, layers
vocab_size = 25000

model = models.Sequential()
model.add(layers.Dense(32, input_dim=vocab_size, kernel_initializer='normal',␣
↪activation='relu'))
model.add(layers.Dense(1, kernel_initializer='normal', activation='sigmoid'))

model.compile(loss='binary_crossentropy', optimizer='adam',metrics=['accuracy'])

history = model.
↪fit(x_train,y_train,batch_size=batch_size,epochs=5,verbose=1,validation_split=0.
↪2)
```

[ ]:
```
# evaluate
score = model.evaluate(x_test, y_test, batch_size=batch_size, verbose=1)
print('Accuracy: ', score[1])
```

1.2 Embeddings

[ ]:
```
model = models.Sequential()
model.add(layers.Embedding(vocab_size, 8, input_length=10000))
model.add(layers.Flatten())
model.add(layers.Dense(16, activation='relu'))
model.add(layers.Dense(6, activation='sigmoid'))

model.compile(optimizer='rmsprop',
              loss='categorical_crossentropy',
              metrics=['acc'])
model.summary()

history = model.fit(x_train,
                    y_train,
                    epochs=5,
```

```
                    batch_size=32)
```

[ ]:
```
score = model.evaluate(x_test, y_test)
print('Accuracy: ', score[1])
print(score)
```

1.3 CNN

[ ]:
```
from keras.layers import Dense, Embedding, Conv1D, MaxPooling1D,␣
↪GlobalMaxPooling1D
from tensorflow.keras import datasets, layers, models


model = models.Sequential()
model.add(layers.Embedding(vocab_size, 128, input_length=10000))
model.add(layers.Conv1D(32, 7, activation='relu'))
model.add(layers.MaxPooling1D(4))
model.add(layers.Conv1D(32, 7, activation='relu'))
model.add(layers.GlobalMaxPooling1D())
model.add(layers.Dense(6))

model.summary()
```

[ ]:
```
model.compile(optimizer=tf.keras.optimizers.RMSprop(learning_rate=1e-4),
              loss='categorical_crossentropy',
              metrics=['accuracy'])

history = model.fit(x_train,
                    y_train,
                    epochs=5,
                    batch_size=128,
                    validation_split=0.2)
```

[ ]:
```
score = model.evaluate(x_test, y_test)
print('Accuracy: ', score[1])
print(score)
```

1.4 Summary

Overall, the deep learning techniqes that were used poorly predicted the sentiment of a Yelp review based on the training data. Both accuracy and loss were not very good, the model took a very long time to train, and crashed several times due to using up all the available RAM both on my local machine as well as on Google Collab. The best result was attained from the Sequential model, while the others all crashed the machine I was working on, even afer reducing the test and training data sizes significantly. I ran the others independently and got results that had an accuracy in the high 20% area. In the future, further preprocessing of the data, along with tweaking the number of epochs and other parameters could lead to better results.