Restaurants Reviews Data Sentiment Analysis

August 28, 2023

[1]:
```
import pandas as pd
import numpy as nm
import matplotlib.pyplot as plt
import seaborn as sns
import tensorflow as tf
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.layers import Embedding, LSTM, Dense, Dropout
from tensorflow.keras.models import Sequential
```

[2]:
```
data = pd.read_csv(r"C:\Users\kazit\Downloads\Data\Restaurant_Reviews.csv")
data
```

[2]:

	review_text	sentiment
0	Wow… Loved this place.	1
1	Crust is not good.	0
2	Not tasty and the texture was just nasty.	0
3	Stopped by during the late May bank holiday of…	1
4	The selection on the menu was great and so wer…	1
..	…	…
995	I think food should have flavor and texture an…	0
996	Appetite instantly gone.	0
997	Overall I was not impressed and would not go b…	0
998	The whole experience was underwhelming, and I …	0
999	Then, as if I hadn't wasted enough of my life …	0

[1000 rows x 2 columns]

[3]:
```
# Extract 'review_text' and 'sentiment' columns from the DataFrame
reviews = data['review_text'].tolist()
sentiments = data['sentiment'].tolist()

# Create and fit the tokenizer
tokenizer = Tokenizer()
tokenizer.fit_on_texts(reviews)
vocab_size = len(tokenizer.word_index) + 1

X = tokenizer.texts_to_sequences(reviews)
```

```
X = pad_sequences(X, padding='post')

# Convert sentiments to a numpy array
sentiments = np.array(sentiments)

# Build the model
model = Sequential()
model.add(Embedding(vocab_size, 100, input_length=X.shape[1]))
model.add(LSTM(100))
model.add(Dropout(0.2))
model.add(Dense(1, activation='sigmoid'))

# Compile the model
model.compile(loss='binary_crossentropy', optimizer='adam',␣
 ↪metrics=['accuracy'])

# Train the model
model.fit(X, sentiments, epochs=5, batch_size=64, validation_split=0.2)
```

```
Epoch 1/5
13/13 [==============================] - 6s 147ms/step - loss: 0.6910 -
accuracy: 0.5350 - val_loss: 0.7719 - val_accuracy: 0.2400
Epoch 2/5
13/13 [==============================] - 1s 68ms/step - loss: 0.6856 - accuracy:
0.5650 - val_loss: 0.7444 - val_accuracy: 0.2400
Epoch 3/5
13/13 [==============================] - 1s 70ms/step - loss: 0.6835 - accuracy:
0.5663 - val_loss: 0.7827 - val_accuracy: 0.2400
Epoch 4/5
13/13 [==============================] - 1s 75ms/step - loss: 0.6719 - accuracy:
0.5725 - val_loss: 0.7873 - val_accuracy: 0.4050
Epoch 5/5
13/13 [==============================] - 1s 74ms/step - loss: 0.4654 - accuracy:
0.8100 - val_loss: 1.2458 - val_accuracy: 0.5750
```

[3]:
```
<tensorflow.python.keras.callbacks.History at 0x29e7b929348>
```

[4]:
```
new_review = "Do not waste your money here!"
new_review_seq = tokenizer.texts_to_sequences([new_review])
new_review_seq = pad_sequences(new_review_seq, padding='post', maxlen=X.
 ↪shape[1])
predicted_sentiment = model.predict(new_review_seq)[0][0]

if predicted_sentiment > 0.5:
        prediction = "positive"
        print( prediction )
else:
```

```
        prediction = "negative"
        print( prediction )
```

```
negative
```

[5]:
```
predicted_sentiments = model.predict(X)
positive_percentage = sum(predicted_sentiments > 0.5) /␣
 ↪len(predicted_sentiments) * 100
negative_percentage = 100 - positive_percentage
print( positive_percentage )
print( negative_percentage )
```

```
[65.9]
[34.1]
```

[6]:
```
reviews_with_predictions = [(reviews[i], predicted_sentiments[i][0]) for i in␣
 ↪range(len(reviews))]
reviews_with_predictions.sort(key=lambda x: x[1], reverse=True)

top_positive_reviews = [review for review, _ in reviews_with_predictions[:5]]
top_negative_reviews = [review for review, _ in reviews_with_predictions[-5:]]
print( top_positive_reviews )
print( top_negative_reviews )
```

```
['Great steak, great sides, great wine, amazing desserts.', 'Great atmosphere,
friendly and fast service.', 'Great food and great service in a clean and
friendly setting.', 'Great brunch spot.', 'The staff are great, the ambiance is
great.']
['Waited 2 hours & never got either of our pizzas as many other around us who
came in later did!', "I don't think I'll be running back to Carly's anytime soon
for food.", "Bland… Not a liking this place for a number of reasons and I
don't want to waste time on bad reviewing.. I'll leave it at that…", 'After I
pulled up my car I waited for another 15 minutes before being acknowledged.', "I
don't think we'll be going back anytime soon."]
```

[11]:
```
import pandas as pd
import io
import ipywidgets as widgets
from IPython.display import display

# Load and preprocess data, train model, etc. (Steps 1-5)

# Function to handle file upload
def handle_upload(change):
    uploaded_file = change['new']
    if uploaded_file:
        uploaded_content = uploaded_file[0]['content'].tobytes()
```

```
        string_dataset = uploaded_content.decode('utf-8')  # Convert bytes to␣
↪string

        # Process string_dataset, predict sentiments, calculate percentages,␣
↪and find top reviews (Steps 3-5)

        # Display results
        positive_percentage_widget = widgets.
↪FloatText(value=positive_percentage)
        negative_percentage_widget = widgets.
↪FloatText(value=negative_percentage)

        top_positive_reviews_widget = widgets.Textarea(value="\n".
↪join(top_positive_reviews), description="Top Positive Reviews")
        top_negative_reviews_widget = widgets.Textarea(value="\n".
↪join(top_negative_reviews), description="Top Negative Reviews")

        display("Positive Percentage", positive_percentage_widget)
        display("Negative Percentage" , negative_percentage_widget)
        display(widgets.Label("Top Positive Reviews:"))
        display(top_positive_reviews_widget)
        display(widgets.Label("Top Negative Reviews:"))
        display(top_negative_reviews_widget)

        # Display the processed dataset
        processed_df = pd.read_csv(io.StringIO(string_dataset))  # Convert␣
↪string to DataFrame
        display(processed_df)

# Create a FileUpload widget
upload_button = widgets.FileUpload(description="Choose a file", accept=".csv")

# Attach the handle_upload function to the widget
upload_button.observe(handle_upload, names='value')

# Display the upload button
display(upload_button)
```

```
FileUpload(value=(), accept='.csv', description='Choose a file')
```

```
'Positive Percentage'
```

```
FloatText(value=65.9)
```

```
'Negative Percentage'
```

```
FloatText(value=34.099999999999994)
```

```
Label(value='Top Positive Reviews:')
```

```
Textarea(value='Great steak, great sides, great wine, amazing desserts.\nGreat␣
 ↪atmosphere, friendly and fast s…

Label(value='Top Negative Reviews:')

Textarea(value="Waited 2 hours & never got either of our pizzas as many other␣
 ↪around us who came in later did!…
```

	values
0	awsm
1	nice product and hd picture
2	super
3	good product nice price i
4	good
…	…
99995	not smooth and comfortable to use
99996	it s really value for money
99997	nice
99998	very good product
99999	size is small in comparison to other cones sme…

```
[100000 rows x 1 columns]
```

```
[ ]:
```