Bryce Smith
5/10/2023
STAA 578 – Assignment 4

The goal of this assignment was to predict the column Cat_code, using the product name and brand name, with the ibotta dataset. The training data consists of 8000 rows while the test data consists of 1999 rows.

Using a Long Short-Term Memory (LSTM) recurrent neural network (RNN) I was able to achieve an accuracy of 0.89148 on the Kaggle leaderboard. The libraries used for this model were pandas, Keras, and scikit-learn to train it for this classification prediction. My model setup began with setting a variable *max_words* and *max_*len to set the maximum number of words to keep based on word fequency and to define the maximum length of the sequences passed to the RNN layer. Next I concantenated the columns Name and Brand_name to create a new column called *combined*. I then created a variables *text* that converts the column combined into a NumPy array. I then used Tokenizer to create a vocabulary index based on work frequency. Using the word indices that was constructed when the tokenizer was fitted, I created a 2D NumPy array. LabelEncoder was then used to create an object to transform category labels into numerical values. This object was fitted on the Cat_code column of the training dataframe. To use categorical crossentropy to compile my model, I use a *to_categorical* object.

The RNN model was built using a sequential structure using Keras. I then added an Embedding layer to the model. The next layer adds a Long Short-Term Memory layer to the model with 64 units. I see the dropout parameters to 0.2 and the recurrent_dropout parameters to 0.2. The last layer was a Dense layer with 7 units and a softmax activation function. When compiling the model I see loss to categorical_crossentropy, the optimizer to Adam, and evaluation to ‘accuracy’ to measure the performance during training and testing. I fit the model with 5 epochs with a validation split of 0.2. Lastly, I performed the same operations of code to the testing dataset, ultimately generating predictions for the test data.

Some variations of this model were to add a bidirectional LSTM as a layer with drpout with parameter of 0.5. This model achieved a lower accuracy of 0.836. I also attempted to use different optimizers when compiling the model. The optimizers I tried were *rmsprop*, *adam*, *AdaGrad*, and *SGD*. The optimizer that performed the best was the Adam optimizer with default parameters.

In [1]:

```
#Load the necessary packages
import numpy as np
import pandas as pd
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, LSTM, Dense, Dropout
from tensorflow.keras.optimizers import Adam
from sklearn.preprocessing import LabelEncoder
from tensorflow.keras.utils import to_categorical
```

In [ ]:

```
#Read in data
df_train = pd.read_csv("ibotta_train.csv")
df_test = pd.read_csv("ibotta_test.csv")
```

In [ ]:

```
max_words = 10000
max_len = 100

#Combine 'Name' and 'Brand_name' fields
df_train['Brand_name'].fillna('unknown', inplace=True)
df_test['Brand_name'].fillna('unknown', inplace=True)
df_train['combined'] = df_train['Name'] + ' ' + df_train['Brand_name']
df_test['combined'] = df_test['Name'] + ' ' + df_test['Brand_name']

#Tokenize words to integer sequences
text = df_train['combined'].values
tokenizer = Tokenizer(num_words=max_words)
tokenizer.fit_on_texts(text)
sequences = tokenizer.texts_to_sequences(text)

#Index linking words to integers
word_index = tokenizer.word_index

#Pad to a common dimension
data = pad_sequences(sequences, maxlen = max_len)

#Convert class vectors to binary class matrices
label_encoder = LabelEncoder()
y_train = label_encoder.fit_transform(df_train['Cat_code'])
y_train = to_categorical(y_train)

#RNN model
model = Sequential()
model.add(Embedding(max_words, 64, input_length = max_len))
model.add(LSTM(64, dropout=0.2, recurrent_dropout=0.2))
model.add(Dense(7, activation='softmax'))  # 7 classes

#Compile the model
model.compile(loss='categorical_crossentropy',
              optimizer='Adam',
              metrics=['accuracy'])

#Fit the model
model.fit(data, y_train, epochs=5, batch_size=32, validation_split=0.2)

#Prepare the test data
test_text = df_test['combined'].values
test_sequences = tokenizer.texts_to_sequences(test_text)
```

```
X_test = pad_sequences(test_sequences, maxlen=max_len)

#Predict
predictions = model.predict(X_test)

#Convert predictions from one-hot encoded to labels
predictions = np.argmax(predictions, axis=1)
predictions = label_encoder.inverse_transform(predictions)
```

In [ ]:

```
#Create a DataFrame for submission
submission = pd.DataFrame({
    'Id': df_test['Id'],
    'Cat_code': predictions
})

#Save the submission DataFrame to a CSV file
submission.to_csv('Smith_Submission.csv', index=False)
```