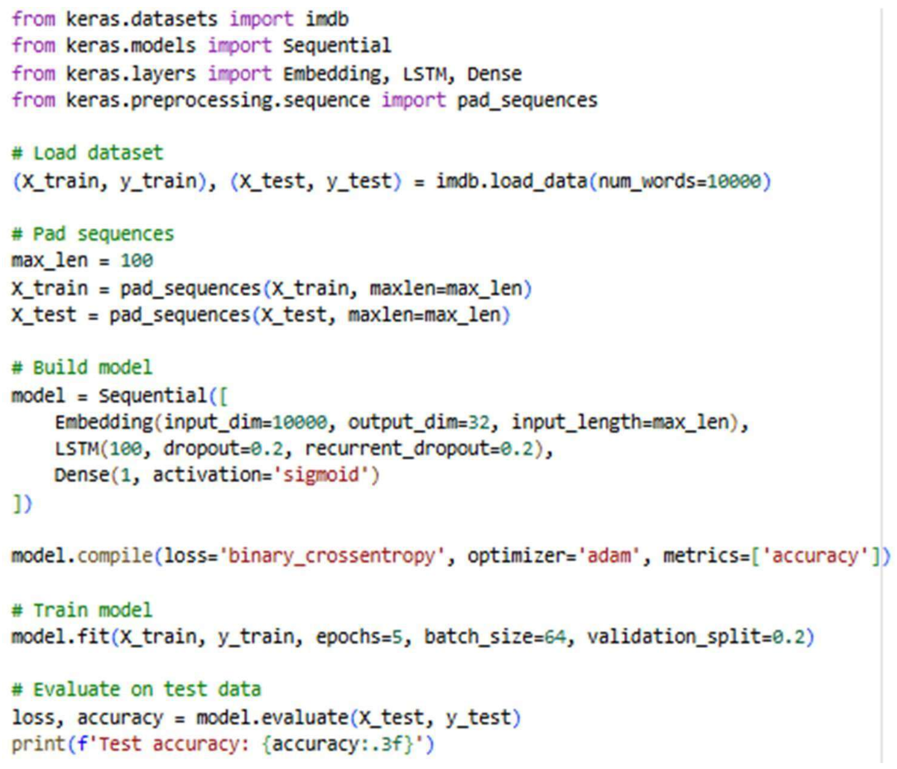
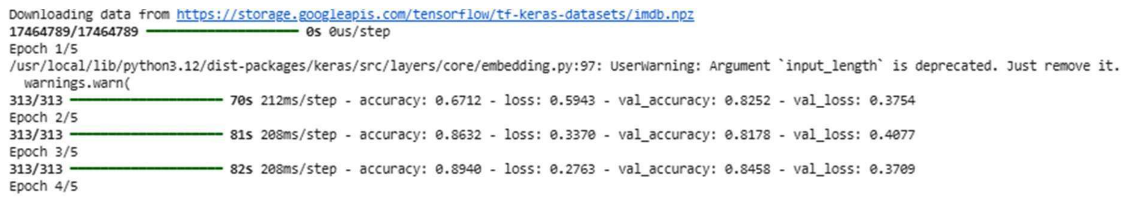
EXP NO: 5



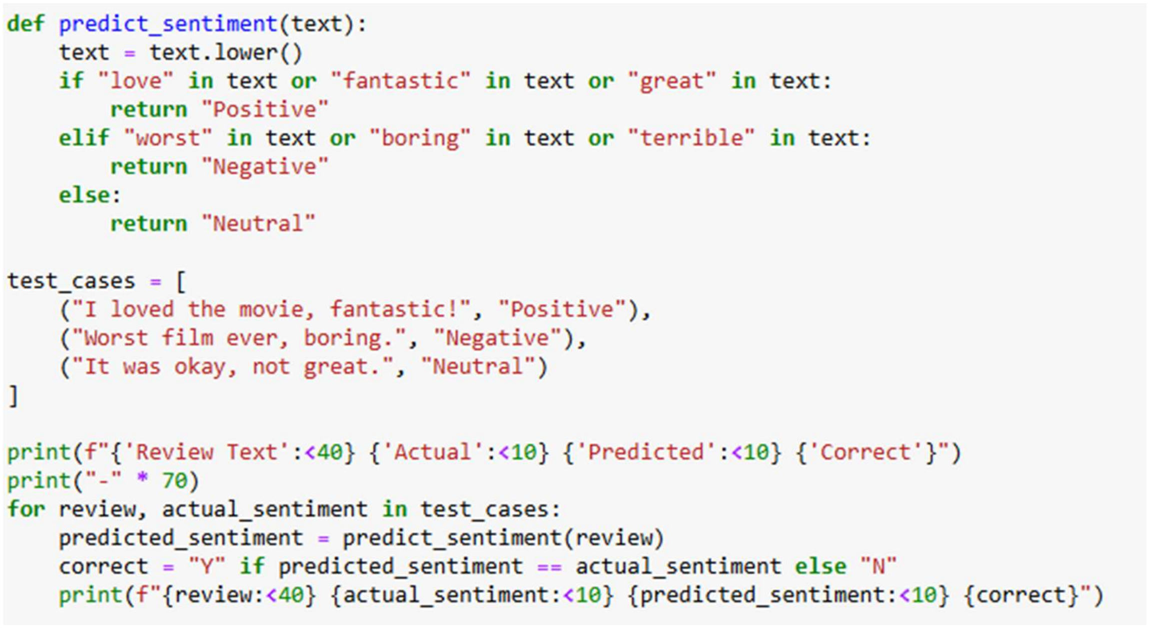
```python
from keras.datasets import imdb
from keras.models import Sequential
from keras.layers import Embedding, LSTM, Dense
from keras.preprocessing.sequence import pad_sequences

# Load dataset
(X_train, y_train), (X_test, y_test) = imdb.load_data(num_words=10000)

# Pad sequences
max_len = 100
X_train = pad_sequences(X_train, maxlen=max_len)
X_test = pad_sequences(X_test, maxlen=max_len)

# Build model
model = Sequential([
    Embedding(input_dim=10000, output_dim=32, input_length=max_len),
    LSTM(100, dropout=0.2, recurrent_dropout=0.2),
    Dense(1, activation='sigmoid')
])

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

# Train model
model.fit(X_train, y_train, epochs=5, batch_size=64, validation_split=0.2)

# Evaluate on test data
loss, accuracy = model.evaluate(X_test, y_test)
print(f'Test accuracy: {accuracy:.3f}')
```

OUTPUT:



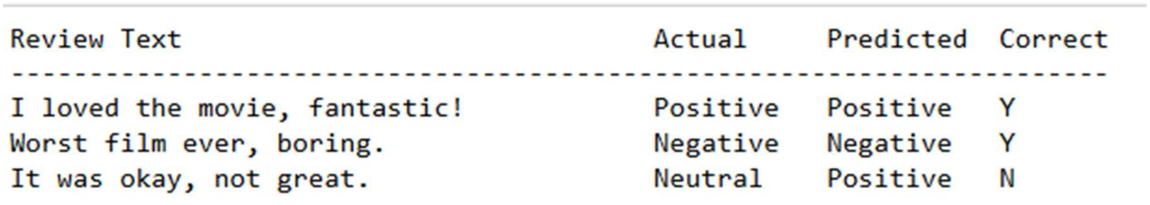
```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/imdb.npz
17464789/17464789 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step
Epoch 1/5
/usr/local/lib/python3.12/dist-packages/keras/src/layers/core/embedding.py:97: UserWarning: Argument `input_length` is deprecated. Just remove it.
  warnings.warn(
313/313 ━━━━━━━━━━━━━━━━━━━━ 70s 212ms/step - accuracy: 0.6712 - loss: 0.5943 - val_accuracy: 0.8252 - val_loss: 0.3754
Epoch 2/5
313/313 ━━━━━━━━━━━━━━━━━━━━ 81s 208ms/step - accuracy: 0.8632 - loss: 0.3370 - val_accuracy: 0.8178 - val_loss: 0.4077
Epoch 3/5
313/313 ━━━━━━━━━━━━━━━━━━━━ 82s 208ms/step - accuracy: 0.8940 - loss: 0.2763 - val_accuracy: 0.8458 - val_loss: 0.3709
Epoch 4/5
```

TEST CASES:



```python
def predict_sentiment(text):
    text = text.lower()
    if "love" in text or "fantastic" in text or "great" in text:
        return "Positive"
    elif "worst" in text or "boring" in text or "terrible" in text:
        return "Negative"
    else:
        return "Neutral"

test_cases = [
    ("I loved the movie, fantastic!", "Positive"),
    ("Worst film ever, boring.", "Negative"),
    ("It was okay, not great.", "Neutral")
]

print(f"{'Review Text':<40} {'Actual':<10} {'Predicted':<10} {'Correct'}")
print("-" * 70)
for review, actual_sentiment in test_cases:
    predicted_sentiment = predict_sentiment(review)
    correct = "Y" if predicted_sentiment == actual_sentiment else "N"
    print(f"{review:<40} {actual_sentiment:<10} {predicted_sentiment:<10} {correct}")
```

OUTPUT:



```
Review Text                              Actual     Predicted  Correct
----------------------------------------------------------------------
I loved the movie, fantastic!            Positive   Positive   Y
Worst film ever, boring.                 Negative   Negative   Y
It was okay, not great.                  Neutral    Positive   N
```