```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MultiLabelBinarizer
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.callbacks import EarlyStopping
import matplotlib.pyplot as plt
import seaborn as sns
from nltk.corpus import stopwords
import re
import nltk



# Download NLTK data
nltk.download('stopwords')

# Task 1: Data Preprocessing
def preprocess_text(text):
    # Convert to lowercase
    text = text.lower()
    # Remove punctuation and special characters
    text = re.sub(r'[^\w\s]', '', text)
    # Remove stopwords
    stop_words = set(stopwords.words('english'))
    text = ' '.join([word for word in text.split() if word not in stop_words])
    return text
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

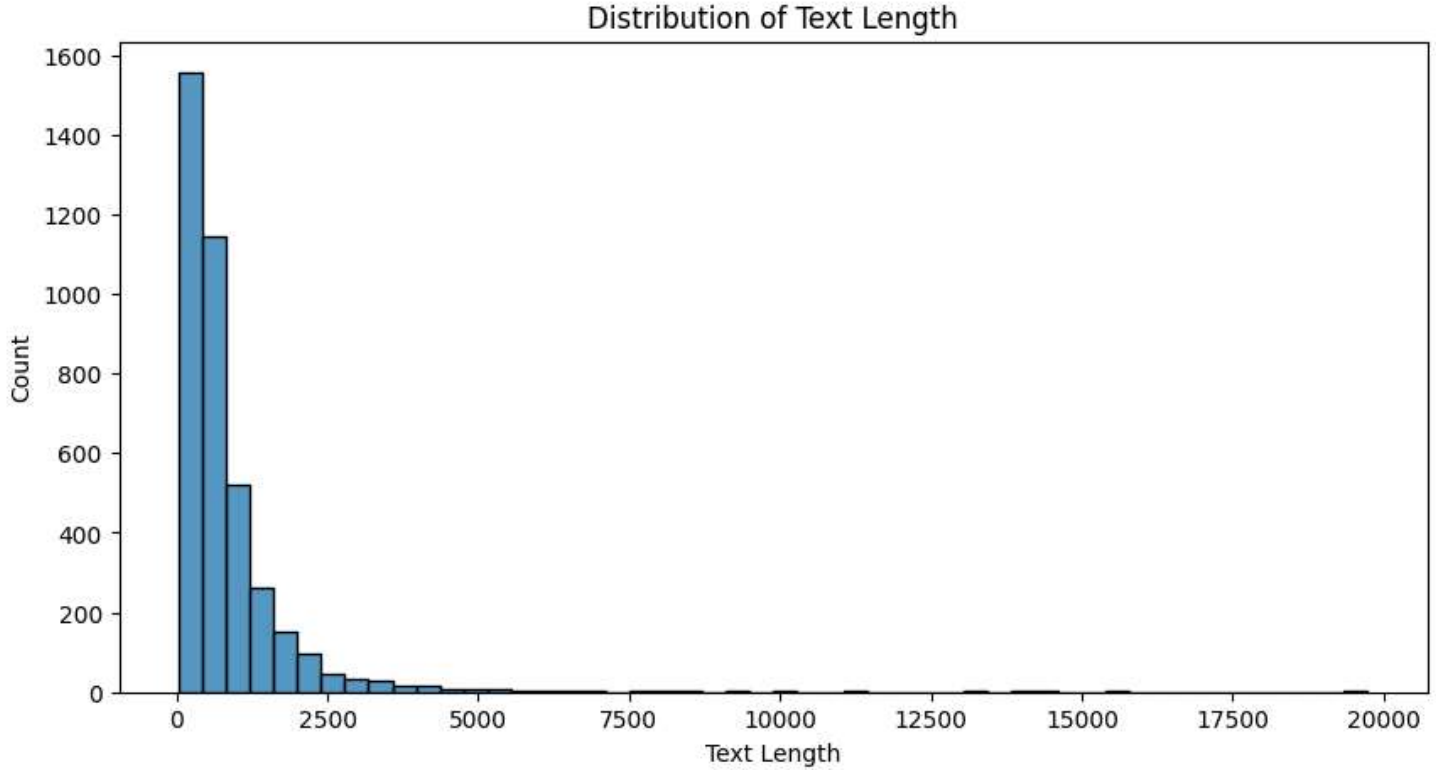
```
data = pd.read_csv('/content/StackOverflow_questions_2009.csv')
data.head()
```

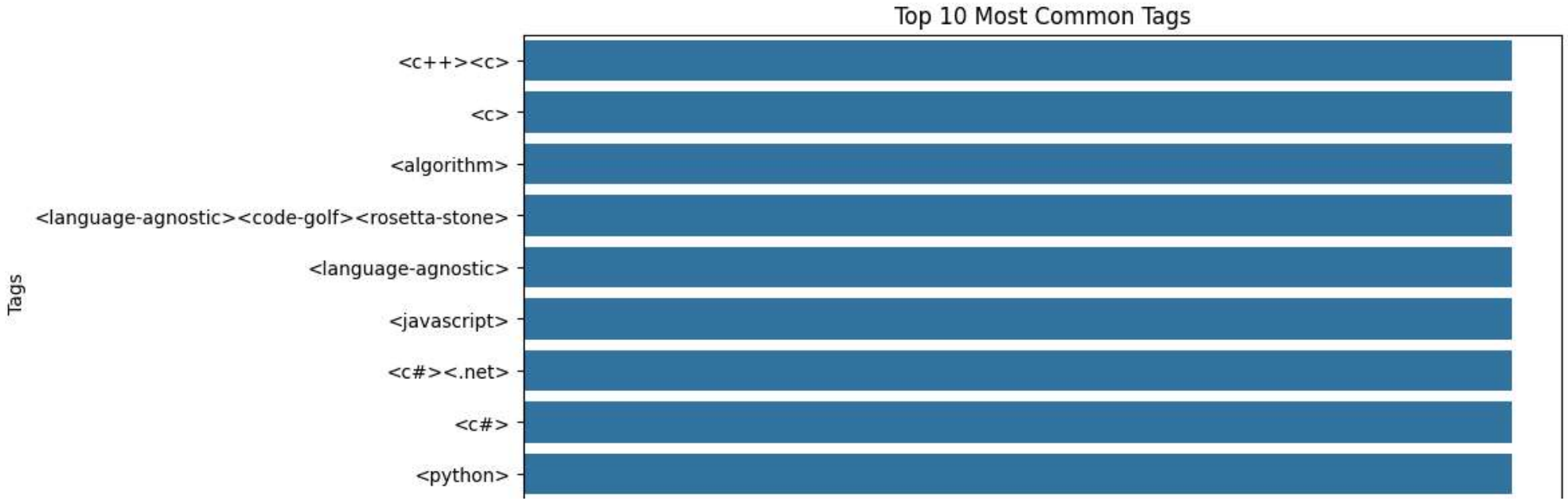
	Id	CreationDate	Title	Body	Tags	ViewCount	CommentCount	AnswerCount	Score
0	414336	2009-01-05 20:09:49	Why does the SqlServer optimizer get so confus...	<p>I know this has something to do with parame...	<sql><sql-server><optimization>	3368	6	9	7
1	414605	2009-01-05 21:33:21	Fluent interface for rendering HTML	<p>Rendering HTML with the HtmlTextWriter isn'...	<.net><html><render><web-controls><htmltextwri...	4557	7	4	10
2	414714	2009-01-05 22:21:04	Compiling with g++ using multiple cores	<p>Quick question: what is the compiler flag t...	<c++><compiler-construction><makefile><g++><mu...	159848	7	8	181
3	415452	2009-01-06 04:26:16	Object-orientation in C	<p>What would be a set of nifty preprocessor h...	<c><oop><object>	82624	6	23	161
		2009-01-06	How can I read and	<p>Pretty self-explanatory. I					

Next steps: Generate code with data | View recommended plots | New interactive sheet

```
# Combine title and body
data['text'] = data['Title'] + ' ' + data['Body']
data['text'] = data['text'].apply(preprocess_text)

# Encode tags
mlb = MultiLabelBinarizer()
tags_encoded = mlb.fit_transform(data['Tags'].str.split())

# Create vocabulary and convert text to sequences
tokenizer = Tokenizer()
tokenizer.fit_on_texts(data['text'])
sequences = tokenizer.texts_to_sequences(data['text'])

# Pad sequences
max_sequence_length = 200  # Adjust as needed
padded_sequences = pad_sequences(sequences, maxlen=max_sequence_length)



# Task 2: Exploratory Data Analysis
plt.figure(figsize=(10, 5))
sns.histplot(data['text'].str.len(), bins=50)
plt.title('Distribution of Text Length')
plt.xlabel('Text Length')
plt.show()
```

```
# Top 10 most common tags
top_tags = mlb.classes_[np.argsort(tags_encoded.sum(axis=0))[-10:]]
plt.figure(figsize=(10, 5))
sns.barplot(x=tags_encoded.sum(axis=0)[-10:], y=top_tags)
plt.title('Top 10 Most Common Tags')
plt.xlabel('Count')
plt.ylabel('Tags')
plt.show()
```


Distribution of Text Length
Count
Text Length
0
200
400
600
800
1000
1200
1400
1600
0
2500
5000
7500
10000
12500
15000
17500
20000


Top 10 Most Common Tags
Tags
<c++><c>
<c>
<algorithm>
<language-agnostic><code-golf><rosetta-stone>
<language-agnostic>
<javascript>
<c#><.net>
<c#>
<python>


```
# Task 3 & 4: Model Development and Training
X_train, X_test, y_train, y_test = train_test_split(padded_sequences, tags_encoded, test_size=0.2, random_state=42)

vocab_size = len(tokenizer.word_index) + 1
embedding_dim = 100
num_tags = len(mlb.classes_)

model = Sequential([
    Embedding(vocab_size, embedding_dim, input_length=max_sequence_length),
    LSTM(128, return_sequences=True),
    LSTM(64),
    Dense(64, activation='relu'),
    Dropout(0.5),
    Dense(num_tags, activation='sigmoid')
])

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

```
/usr/local/lib/python3.10/dist-packages/keras/src/layers/core/embedding.py:90: UserWarning: Argument `input_length` is deprecated. Just remove it.
  warnings.warn(
```

```
early_stopping = EarlyStopping(monitor='val_loss', patience=3, restore_best_weights=True)

history = model.fit(X_train, y_train,
                    epochs=10,
                    batch_size=32,
                    validation_split=0.2,
                    callbacks=[early_stopping])
```

```
Epoch 1/10
79/79 ━━━━━━━━━━━━━━━━━━━━ 65s 713ms/step - accuracy: 6.3185e-04 - loss: 0.4360 - val_accuracy: 0.0016 - val_loss: 0.0028
Epoch 2/10
79/79 ━━━━━━━━━━━━━━━━━━━━ 50s 640ms/step - accuracy: 0.0032 - loss: 0.0034 - val_accuracy: 0.0016 - val_loss: 0.0027
Epoch 3/10
79/79 ━━━━━━━━━━━━━━━━━━━━ 82s 645ms/step - accuracy: 0.0034 - loss: 0.0030 - val_accuracy: 0.0016 - val_loss: 0.0027
Epoch 4/10
79/79 ━━━━━━━━━━━━━━━━━━━━ 81s 637ms/step - accuracy: 0.0100 - loss: 0.0029 - val_accuracy: 0.0127 - val_loss: 0.0028
Epoch 5/10
```

```
79/79 ━━━━━━━━━━━━━━━━━━━━ 82s 634ms/step - accuracy: 0.0053 - loss: 0.0028 - val_accuracy: 0.0127 - val_loss: 0.0029
```

```
# Task 5: Model Evaluation
from sklearn.metrics import precision_score, recall_score, f1_score, hamming_loss

# Threshold for converting probabilities to binary predictions (adjust as needed)
threshold = 0.5

# Convert probabilities to binary predictions
y_pred = (y_pred > threshold).astype(np.int32)

# Calculate evaluation metrics
precision = precision_score(y_test.argmax(axis=1), y_pred.argmax(axis=1), average='macro')
recall = recall_score(y_test.argmax(axis=1), y_pred.argmax(axis=1), average='macro')
```