Classification

Use the model to classify a given text sequence.

Chapter Goals:

- Classify input text using the BiLSTM model

A. Calculating predictions

our BiLSTM is used to classify sentiment in text as either positive or negative. Since there are only two classes, this is binary classification. For binary classification, we apply the sigmoid function on the model's logits to obtain a probability for the input text. The probability refers to the likelihood that the input text is labeled "positive" (i.e. label `1`), and we can simply round the probability to the nearest integer to obtain the model's prediction.

Time to Code!

In this chapter you'll be completing the `logits_to_predictions` function, which converts the model's logits to binary classification predictions.

To calculate the model's prediction, we first need to obtain probabilities from the logits. We do this by applying the sigmoid function.

Set `probs` equal to `tf.nn.sigmoid` applied to `logits`.

Our model's predictions now become the probabilities rounded to the nearest integer.

Set `preds` equal to `tf.round` applied to `probs`. Then return `preds`.

```
import tensorflow as tf
tf_fc = tf.contrib.feature_column

# Text classification model
class ClassificationModel(object):
    # Model initialization
    def __init__(self, vocab_size, max_length, num_lstm_units):
        self.vocab_size = vocab_size
```

```
        self.max_length = max_length
        self.num_lstm_units = num_lstm_units
        self.tokenizer = tf.keras.preprocessing.text.Tokenizer(num_words=self.vocab_size)


    # Convert logits to predictions
    def logits_to_predictions(self, logits):
        # CODE HERE
        pass
```