String Features

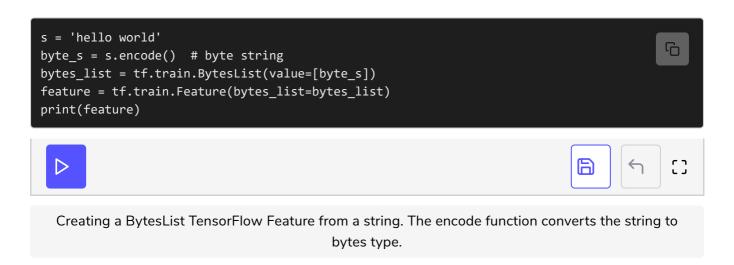
Learn about the string features used in the dataset.

Chapter Goals:

Add the string features of a DataFrame's row to a feature dictionary

A. Adding string features

The third type of TensorFlow Feature object that can be used in an Example object is a <code>BytesList</code> TensorFlow Feature. This can represent either byte values (e.g. image data) or string values. For string values, we need to convert them to the <code>bytes</code> type prior to initializing a <code>BytesList</code> Feature object.



From the analysis of our dataset, we know that the only feature containing string values is 'Type'.

Time to Code!

In this chapter you'll be completing the create_example function, which creates an Example object from a row of the dataset.

We've already initialized a feature dictionary and added the integer and float features, using the functions from the previous two chapters. The only feature that contains string values is 'Type', so we need to convert the 'Type' value in dataset_row to bytes, and then create a BytesList.

Set byte_type equal to dataset_row['Type'], converted to bytes.

Set list_val equal to tf.train.BytesList initialized with the value keyword argument set to a singleton list containing byte_type.

We can now complete the **feature_dict** by mapping 'Type' to its corresponding TensorFlow Feature object.

Put 'Type' as a key in feature_dict, and map it to a tf.train.Feature object initialized with the bytes_list keyword argument set to list_val.

Using the completed feature_dict dictionary, we'll create and return a TensorFlow Example object containing the values in dataset_row.

Set features_obj equal to tf.train.Features initialized with the feature keyword argument set to feature_dict.

Return a tf.train.Example object initialized with the features keyword argument set to features_obj.

