Input Layer

Combine data features into an input layer for a neural network.

Chapter Goals:

- Learn how to use feature columns to create an input layer for a machine learning model
- Implement a function that iterates through a protocol buffer dataset and creates a list of input layers

A. Combining features

If you recall, a parsed protocol buffer is a dictionary that maps feature names to either a `tf.Tensor` or `tf.SparseTensor` object. Using the parsed protocol buffer and corresponding feature columns, we can combine the feature values into an input layer for a machine learning model. To do this, we use the `tf.feature_column.input_layer` function.

task.py

majors.txt

```
import numpy as np
import tensorflow as tf

# Get a parsed TFRecordDataset
dataset = get_dataset()

it = dataset.make_one_shot_iterator()
next_elem = it.get_next()
# Get feature columns for the dataset
feature_columns = get_feature_columns()
for i in range(len(feature_columns)):
    print(feature_columns[i])
    print() # Newline
inputs = tf.feature_column.input_layer(next_elem, feature_columns)
print(inputs)
```

In the example above, the `inputs` tensor represents an input layer created from a batch of parsed protocol buffers. Notice that the second dimension of the tensor's shape is 21. This means that the features specified in `feature_columns` have a combined value length of 21.

B. Extracting data

We use `tf.Session` to extract values from the input layer tensor. The process of iterating through the dataset and extracting data is nearly identical to the one described in chapter 9. The only difference is that, when using categorical feature columns, we need to set up a tables initializer with the `tf.tables_initializer` operation.

task.py

majors.txt

```
import numpy as np
import tensorflow as tf

# Get a parsed TFRecordDataset
dataset = get_dataset()

it = dataset.make_one_shot_iterator()
next_elem = it.get_next()
# Get feature columns for the dataset
feature_columns = get_feature_columns()
inputs = tf.feature_column.input_layer(next_elem, feature_columns)

table_init = tf.tables_initializer()
with tf.Session() as sess:
    sess.run(table_init) # Initialize vocab table
    for i in range(2):
        print(repr(sess.run(inputs)))
```

This code will extract values from the input layer tensor

Since there are vocabulary features in `feature_columns`, we need to use the table initializer in the example above. We call `sess.run` on `table_init` to perform the table initialization.