Formulas like this can be combined to provide a symbolic differentiation system for vectorial and tensorial calculus.

Learning with TensorFlow

In the rest of this chapter, we will cover the concepts that you need to learn basic machine learning models with TensorFlow. We will start by introducing the concept of toy datasets, and will explain how to create meaningful toy datasets using common Python libraries. Next, we will discuss new TensorFlow ideas such as placeholders, feed dictionaries, name scopes, optimizers, and gradients. The next section will show you how to use these concepts to train simple regression and classification models.

Creating Toy Datasets

In this section, we will discuss how to create simple but meaningful synthetic datasets, or toy datasets, that we will use to train simple supervised classification and regression models.

An (extremely) brief introduction to NumPy

We will make heavy use of NumPy in order to define useful toy datasets. NumPy is a Python package that allows for manipulation of tensors (called `ndarrays` in NumPy). Example 3-1 shows some basics.

Example 3-1. Some examples of basic NumPy usage

```
>>> import numpy as np
>>> np.zeros((2,2))
array([[ 0.,  0.],
       [ 0.,  0.]])
>>> np.eye(3)
array([[ 1.,  0.,  0.],
       [ 0.,  1.,  0.],
       [ 0.,  0.,  1.]])
```

You may notice that NumPy `ndarray` manipulation looks remarkably similar to TensorFlow tensor manipulation. This similarity was purposefully designed by TensorFlow's architects. Many key TensorFlow utility functions have similar arguments and forms to analogous functions in NumPy. For this purpose, we will not attempt to introduce NumPy in great depth, and will trust readers to use experimentation to work out NumPy usage. There are numerous online resources that provide tutorial introductions to NumPy.