Data Science Fundamentals - Python

MSc. Juan Antonio Castro Silva

April 2020

Version: 0.1 (20200315_1645)

1 How To Load Machine Learning Data

You must be able to load your data before you can start your machine learning project. The most common format for machine learning data is CSV files. There are a number of ways to load a CSV file in Python. In this lesson you will learn three ways that you can use to load your CSV data in Python:

1. Load CSV Files with the Python Standard Library.
2. Load CSV Files with NumPy.
3. Load CSV Files with Pandas.

1.1 Considerations When Loading CSV Data

There are a number of considerations when loading your machine learning data from CSV files. For reference, you can learn a lot about the expectations for CSV files by reviewing the CSV request for comment titled Common Format and MIME Type for Comma-Separated Values (CSV) Files.

1.1.1 File Header

Does your data have a file header? If so this can help in automatically assigning names to each column of data. If not, you may need to name your attributes manually. Either way, you should explicitly specify whether or not your CSV file had a file header when loading your data.

1.1.2 Comments

Does your data have comments? Comments in a CSV file are indicated by a hash (#) at the start of a line. If you have comments in your file, depending on the method used to load your data, you may need to indicate whether or not to expect comments and the character to expect to signify a comment line.

1.1.3 Delimiter

The standard delimiter that separates values in fields is the comma (,) character. Your file could use a different delimiter like tab or white space in which case you must specify it explicitly.

1.1.4 Quotes

Sometimes field values can have spaces. In these CSV files the values are often quoted. The default quote character is the double quotation marks character. Other characters can be used, and you must specify the quote character used in your file.

1.2 Pima Indians Dataset

The Pima Indians dataset is used to demonstrate data loading in this lesson. It will also be used in many of the lessons to come. This dataset describes the medical records for Pima Indians and whether or not each patient will have an onset of diabetes within five years. As such it is a classification problem. It is a good dataset for demonstration because all of the input attributes are numeric and the output variable to be predicted is binary (0 or 1). The data is available in the GitHub repository https://github.com/juancasi/data_science_fundamentals.

1.3 Load CSV Files with the Python Standard Library

The Python API provides the module CSV and the function reader() that can be used to load CSV files. Once loaded, you can convert the CSV data to a NumPy array and use it for machine learning. For example, you can download the Pima Indians dataset into your local directory with the filename datasets/pima-indians-diabetes.csv. All fields in this dataset are numeric and there is no header line.

```
# Load CSV Using Python Standard Library
import csv
import numpy
filename = '../datasets/pima-indians-diabetes.csv'
raw_data = open(filename, 'r')
reader = csv.reader(raw_data, delimiter=',', quoting=csv.QUOTE_NONE)
x = list(reader)
data = numpy.array(x).astype('float')
print(data.shape)
```

Listing 1: Example of loading a CSV file using the Python standard library.

The example loads an object that can iterate over each row of the data and can easily be converted into a NumPy array. Running the example prints the shape of the array.

```
(768, 9)
```

Listing 2: Output of example loading a CSV file using the Python standard library.

For more information on the csv.reader() function, see CSV File Reading and Writing in the Python API documentation.

1.4 Load CSV Files with NumPy

You can load your CSV data using NumPy and the numpy.loadtxt() function. This function assumes no header row and all data has the same format. The example below assumes that the file pima-indians-diabetes.csv is in the datasets directory.

```
# Load CSV using NumPy
from numpy import loadtxt
filename = '../datasets/pima-indians-diabetes.csv'
raw_data = open(filename, 'r')
data = loadtxt(raw_data, delimiter=",")
```

```
print(data.shape)
```

Listing 3: Example of loading a CSV file using NumPy.

Running the example will load the file as a numpy.ndarray and print the shape of the data:

```
(768, 9)
```

Listing 4: Output of example loading a CSV file using NumPy.

This example can be modified to load the same dataset directly from a URL as follows:

```
# Load CSV from URL using NumPy
from numpy import loadtxt
from urllib.request import urlopen
url = 'https://raw.githubusercontent.com/juancasi/data_science_fundamentals/'
url += 'master/chapter_03/datasets/pima-indians-diabetes.csv'
raw_data = urlopen(url)
dataset = loadtxt(raw_data, delimiter=",")
print(dataset.shape)
```

Listing 5: Example of loading a CSV URL using NumPy.

Again, running the example produces the same resulting shape of the data.

```
(768, 9)
```

Listing 6: Output of example loading a CSV URL using NumPy.

For more information on the numpy.loadtxt() function see the API documentation.

1.5 Load CSV Files with Pandas

You can load your CSV data using Pandas and the pandas.read_csv() function. This function is very flexible and is perhaps my recommended approach for loading your machine learning data. The function returns a pandas.DataFrame that you can immediately start summarizing and plotting. The example below assumes that the pima-indians-diabetes.csv file is in the current working directory.

```
# Load CSV using Pandas
from pandas import read_csv
filename = '../datasets/pima-indians-diabetes.csv'
names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']
data = read_csv(filename, names=names)
print(data.shape)
```

Listing 7: Example of loading a CSV file using Pandas.

Note that in this example we explicitly specify the names of each attribute to the DataFrame. Running the example displays the shape of the data:

```
(768, 9)
```

Listing 8: Output of example loading a CSV file using Pandas.

We can also modify this example to load CSV data directly from a URL

```
# Load CSV using Pandas from URL
from pandas import read_csv
url = 'https://raw.githubusercontent.com/juancasi/data_science_fundamentals/'
url += 'master/chapter_03/datasets/pima-indians-diabetes.csv'
names = ['preg','plas','pres','skin','test','mass','pedi','age','class']
```

```
6 data = read_csv(url, names=names)
7 print(data.shape)
```

Listing 9: Example of loading a CSV URL using Pandas.

Again, running the example downloads the CSV file, parses it and displays the shape of the loaded DataFrame.

```
(768, 9)
```

Listing 10: Output of example loading a CSV URL using Pandas.

To learn more about the pandas.read_csv() function you can refer to the API documentation.

1.6 Summary

In this chapter you discovered how to load your machine learning data in Python. You learned three specific techniques that you can use:

- Load CSV Files with the Python Standard Library.
- Load CSV Files with NumPy.
- Load CSV Files with Pandas.

Generally I recommend that you load your data with Pandas in practice and all subsequent examples in this book will use this method.