

**BOWEN UNIVERSITY, IWO**

**COLLEGE OF COMPUTING AND COMMUNICATION STUDIES**

**DEPARTMENT OF COMPUTER SCIENCE**

**CSC 404- MACHINE LEARNING WITH PYTHON**

**LECTURE NOTE 5**

Data loading for ML Project

Suppose if you want to start a ML project then what is the first and most important thing you would require? It is the data that we need to load for starting any of the ML project. With respect to data, the most common format of data for ML projects is CSV (comma-separated values).

Basically, CSV is a simple file format which is used to store tabular data (number and text) such as a spreadsheet in plain text. In Python, we can load CSV data into with different ways but before loading CSV data we must have to take care about some considerations.

**Consideration While Loading CSV data**

CSV data format is the most common format for ML data, but we need to take care about following major considerations while loading the same into our ML projects:

**File Header**

In CSV data files, the header contains the information for each field. We must use the same delimiter for the header file and for data file because it is the header file that specifies how should data fields be interpreted.

The following are the two cases related to CSV file header which must be considered:

* **Case-I: When Data file is having a file header:** It will automatically assign the names to each column of data if data file is having a file header.
* **Case-II: When Data file is not having a file header:** We need to assign the names to each column of data manually if data file is not having a file header.

In both the cases, we must need to specify explicitly weather our CSV file contains header or not.

**Comments**

Comments in any data file are having their significance. In CSV data file, comments are indicated by a hash (#) at the start of the line. We need to consider comments while loading CSV data into ML projects because if we are having comments in the file then we may need to indicate, depends upon the method we choose for loading, whether to expect those comments or not.

**Delimiter**

In CSV data files, comma (,) character is the standard delimiter. The role of delimiter is to separate the values in the fields. It is important to consider the role of delimiter while uploading the CSV file into ML projects because we can also use a different delimiter such as a tab or white space. But in the case of using a different delimiter than standard one, we must have to specify it explicitly.

**Quotes**

In CSV data files, double quotation (“ ”) mark is the default quote character. It is important to consider the role of quotes while uploading the CSV file into ML projects because we can also use other quote character than double quotation mark. But in case of using a different quote character than standard one, we must have to specify it explicitly.

**Methods to Load CSV Data File**

While working with ML projects, the most crucial task is to load the data properly into it. The most common data format for ML projects is CSV and it comes in various flavors and varying difficulties to parse. In this section, we are going to discuss about three common approaches in Python to load CSV data file:

**Load CSV with Python Standard Library**

The first and most used approach to load CSV data file is the use of Python standard library which provides us a variety of built-in modules namely **csv module** and the **reader()function**. The following is an example of loading CSV data file with the help of it:

**Example**

In this example, we are using the iris flower data set which can be downloaded into our local directory. After loading the data file, we can convert it into **NumPy** array and use it for ML projects. Following is the Python script for loading CSV data file:

First, we need to import the csv module provided by Python standard library as follows:

import csv

Next, we need to import Numpy module for converting the loaded data into NumPy array.

import numpy as np

Now, provide the full path of the file, stored on our local directory, having the CSV data file:

path = r"c:\iris.csv"

Next, use the csv.reader()function to read data from CSV file:

with open(path,'r') as f:

reader = csv.reader(f,delimiter = ',')

headers = next(reader)

data = list(reader)

data = np.array(data).astype(float)

We can print the names of the headers with the following line of script:

print(headers)

The following line of script will print the shape of the data i.e. number of rows & columns in the file:

print(data.shape)

Next script line will give the first three line of data file:

print(data[:3])

**Output**

['sepal\_length', 'sepal\_width', 'petal\_length', 'petal\_width']

(150, 4)

[[5.1 3.5 1.4 0.2]

[4.9 3. 1.4 0.2]

[4.7 3.2 1.3 0.2]]

**Load CSV with NumPy**

Another approach to load CSV data file is **NumPy** and **numpy.loadtxt() function**. The following is an example of loading CSV data file with the help of it:

**Example**

In this example, we are using the Pima Indians Dataset having the data of diabetic patients. This dataset is a numeric dataset with no header. It can also be downloaded into our local directory. After loading the data file, we can convert it into **NumPy** array and use it for ML projects. The following is the Python script for loading CSV

data file:

from numpy import loadtxt

path = r"C:\pima-indians-diabetes.csv"

datapath= open(path, 'r')

data = loadtxt(datapath, delimiter=",")

print(data.shape)

print(data[:3])

**Output**

(768, 9)

[[ 6. 148. 72. 35. 0. 33.6 0.627 50. 1.]

[ 1. 85. 66. 29. 0. 26.6 0.351 31. 0.]

[ 8. 183. 64. 0. 0. 23.3 0.672 32. 1.]]

**Load CSV with Pandas**

Another approach to load CSV data file is by **Pandas** and **pandas.read\_csv()function**. This is the very flexible function that returns a **pandas.DataFrame** which can be used immediately for plotting. The following is an example of loading CSV data file with the help of it:

**Example**

Here, we will be implementing two Python scripts, first is with Iris data set having headers and another is by using the Pima Indians Dataset which is a numeric dataset with no header. Both the datasets can be downloaded into local directory.

**Script-1**

The following is the Python script for loading CSV data file using **Pandas** on Iris Data set:

from pandas import read\_csv

path = r"C:\iris.csv"

data = read\_csv(path)

print(data.shape)

print(data[:3])

Output:

(150, 4)

sepal\_length sepal\_width petal\_length petal\_width

0 5.1 3.5 1.4 0.2

1 4.9 3.0 1.4 0.2

2 4.7 3.2 1.3 0.2

**Script-2**

The following is the Python script for loading CSV data file, along with providing the headers names too, using Pandas on Pima Indians Diabetes dataset:

from pandas import read\_csv

path = r"C:\pima-indians-diabetes.csv"

headernames = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']

data = read\_csv(path, names=headernames)

print(data.shape)

print(data[:3])

**Output**

(768, 9)

preg plas pres skin test mass pedi age class

0 6 148 72 35 0 33.6 0.627 50 1

1 1 85 66 29 0 26.6 0.351 31 0

2 8 183 64 0 0 23.3 0.672 32 1

The difference between above used three approaches for loading CSV data file can easily be understood with the help of given examples.