How to read most commonly used file formats in Data Science (using Python)?

analyticsvidhya.com/blog/2017/03/read-commonly-used-formats-using-python/

March 2, 2017

Ankit Gupta — March 2, 2017

Introduction

If you have been part of the data science (or any data!) industry, you would know the challenge of working with different data types. Different formats, different compression, different parsing on different systems – you could be quickly pulling your hair! Oh and I have not talked about the unstructured data or semi-structured data yet.

For any data scientist or data engineer, dealing with different formats can become a tedious task. In real-world, people rarely get neat tabular data. Thus, it is mandatory for any data scientist (or a data engineer) to be aware of different file formats, common challenges in handling them and the best / efficient ways to handle this data in real life.

This article provides common formats a data scientist or a data engineer must be aware of. I will first introduce you to different common file formats used in the industry. Later, we'll see how to read these file formats in Python.

P.S. *In rest of this article, I will be referring to a data scientist, but the same applies to a data engineer or any data science professional.*

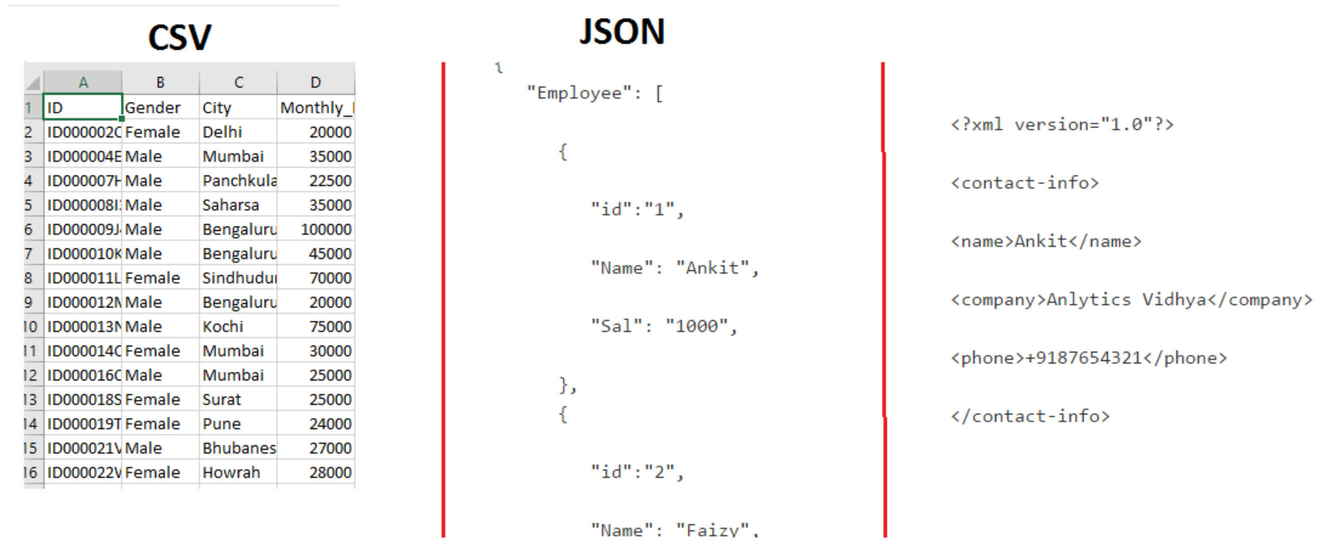
Table of Contents

3. Different file formats and how to read them in Python?
    1. Comma-separated values
    2. XLSX
    3. ZIP
    4. Plain Text (txt)
    5. JSON
    6. XML
    7. HTML
    8. Images
    9. Hierarchical Data Format
    10. PDF
    11. DOCX
    12. MP3
    13. MP4

1. What is a file format?

A file format is a standard way in which information is encoded for storage in a file. First, the file format specifies whether the file is a binary or ASCII file. Second, it shows how the information is organized. For example, comma-separated values (CSV) file format stores tabular data in plain text.

To identify a file format, you can usually look at the file extension to get an idea. For example, a file saved with name “Data” in “CSV” format will appear as “Data.csv”. By noticing “.csv” extension we can clearly identify that it is a “CSV” file and data is stored in a tabular format.



	A	B	C	D
1	ID	Gender	City	Monthly_I
2	ID000002C	Female	Delhi	20000
3	ID000004E	Male	Mumbai	35000
4	ID000007H	Male	Panchkula	22500
5	ID000008I	Male	Saharsa	35000
6	ID000009J	Male	Bengaluru	100000
7	ID000010K	Male	Bengaluru	45000
8	ID000011L	Female	Sindhudur	70000
9	ID000012M	Male	Bengaluru	20000
10	ID000013N	Male	Kochi	75000
11	ID000014O	Female	Mumbai	30000
12	ID000016O	Male	Mumbai	25000
13	ID000018S	Female	Surat	25000
14	ID000019T	Female	Pune	24000
15	ID000021V	Male	Bhubanes	27000
16	ID000022V	Female	Howrah	28000

```
{
   "Employee": [

      {

         "id":"1",

         "Name": "Ankit",

         "Sal": "1000",

      },
      {

         "id":"2",

         "Name": "Faizy",
```

```
<?xml version="1.0"?>

<contact-info>

<name>Ankit</name>

<company>Anlytics Vidhya</company>

<phone>+9187654321</phone>

</contact-info>
```

2. Why should a data scientist understand different file formats?

Usually, the files you will come across will depend on the application you are building. For example, in an image processing system, you need image files as input and output. So you will mostly see files in jpeg, gif or png format.

As a data scientist, you need to understand the underlying structure of various file formats, their advantages and dis-advantages. Unless you understand the underlying structure of the data, you will not be able to explore it. Also, at times you need to make decisions about how to store data.

Choosing the optimal file format for storing data can improve the performance of your models in data processing.

Now, we will look at the following file formats and how to read them in Python:

- Comma-separated values
- XLSX
- ZIP
- Plain Text (txt)
- JSON
- XML
- HTML
- Images
- Hierarchical Data Format
- PDF
- DOCX
- MP3
- MP4

3. Different file formats and how to read them in Python

3.1 Comma-separated values

Comma-separated values file format falls under spreadsheet file format.

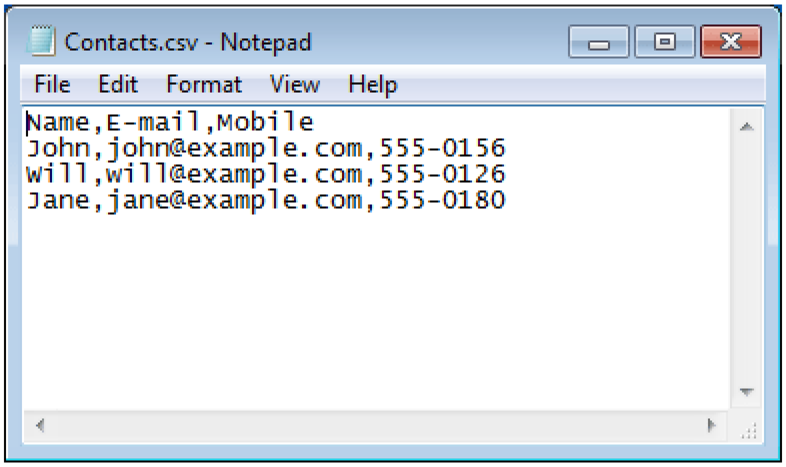
What is Spreadsheet File Format?

In spreadsheet file format, data is stored in cells. Each cell is organized in rows and columns. A column in the spreadsheet file can have different types. For example, a column can be of string type, a date type or an integer type. Some of the most popular spreadsheet file formats are Comma Separated Values (CSV), Microsoft Excel Spreadsheet (xls) and Microsoft Excel Open XML Spreadsheet (xlsx).

Each line in CSV file represents an observation or commonly called a record. Each record may contain one or more fields which are separated by a comma.

Sometimes you may come across files where fields are not separated by using a comma but they are separated using tab. This file format is known as TSV (Tab Separated Values) file format.

The below image shows a CSV file which is opened in Notepad.



```
Name,E-mail,Mobile
John,john@example.com,555-0156
Will,will@example.com,555-0126
Jane,jane@example.com,555-0180
```

Reading the data from CSV in Python

Let us look at how to read a CSV file in Python. For loading the data you can use the "pandas" library in python.

```
import pandas as pd
```

df = pd.read_csv("/home/Loan_Prediction/train.csv")

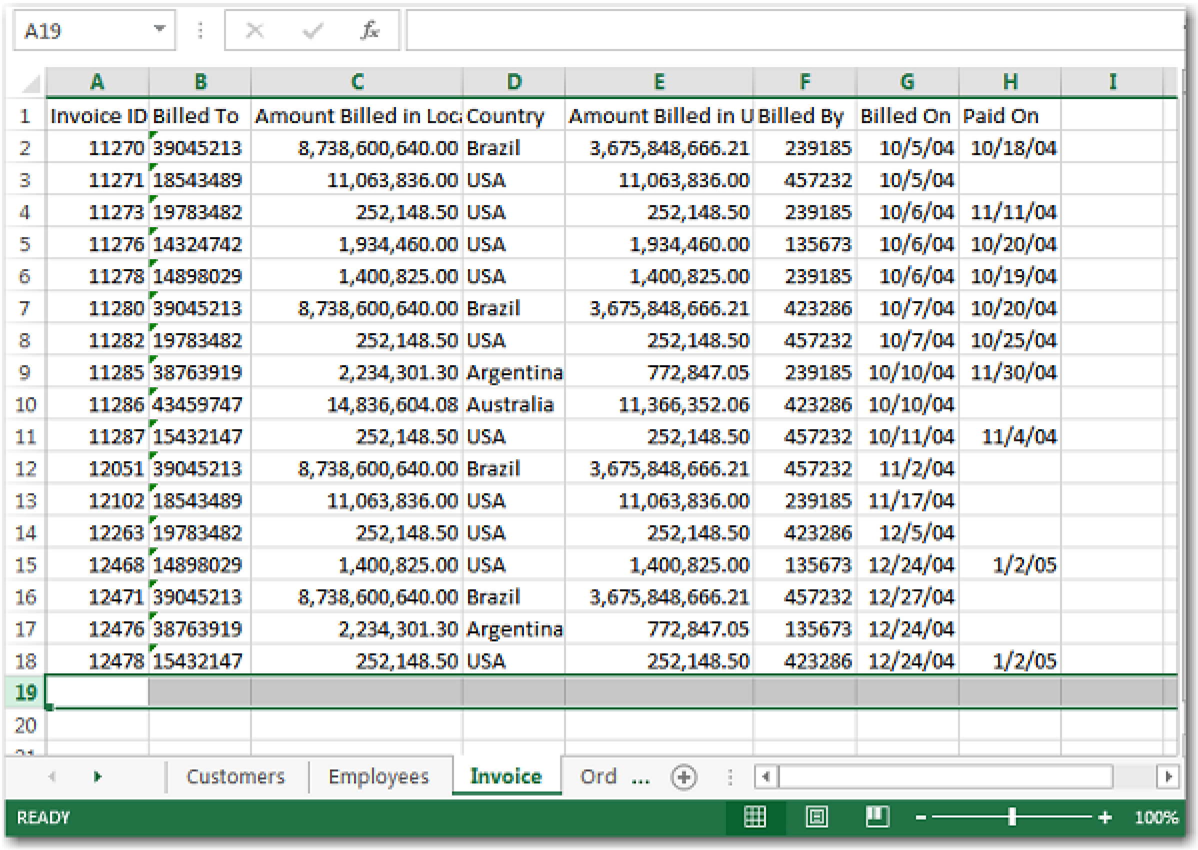
Above code will load the train.csv file in DataFrame df.

3.2 XLSX files

XLSX is a Microsoft Excel Open XML file format. It also comes under the Spreadsheet file format. It is an XML-based file format created by Microsoft Excel. The XLSX format was introduced with Microsoft Office 2007.

In XLSX data is organized under the cells and columns in a sheet. Each XLSX file may contain one or more sheets. So a workbook can contain multiple sheets.

The below image shows a "xlsx" file which is opened in Microsoft Excel.



	A	B	C	D	E	F	G	H	I
1	Invoice ID	Billed To	Amount Billed in Loc	Country	Amount Billed in U	Billed By	Billed On	Paid On	
2	11270	39045213	8,738,600,640.00	Brazil	3,675,848,666.21	239185	10/5/04	10/18/04	
3	11271	18543489	11,063,836.00	USA	11,063,836.00	457232	10/5/04		
4	11273	19783482	252,148.50	USA	252,148.50	239185	10/6/04	11/11/04	
5	11276	14324742	1,934,460.00	USA	1,934,460.00	135673	10/6/04	10/20/04	
6	11278	14898029	1,400,825.00	USA	1,400,825.00	239185	10/6/04	10/19/04	
7	11280	39045213	8,738,600,640.00	Brazil	3,675,848,666.21	423286	10/7/04	10/20/04	
8	11282	19783482	252,148.50	USA	252,148.50	457232	10/7/04	10/25/04	
9	11285	38763919	2,234,301.30	Argentina	772,847.05	239185	10/10/04	11/30/04	
10	11286	43459747	14,836,604.08	Australia	11,366,352.06	423286	10/10/04		
11	11287	15432147	252,148.50	USA	252,148.50	457232	10/11/04	11/4/04	
12	12051	39045213	8,738,600,640.00	Brazil	3,675,848,666.21	457232	11/2/04		
13	12102	18543489	11,063,836.00	USA	11,063,836.00	239185	11/17/04		
14	12263	19783482	252,148.50	USA	252,148.50	423286	12/5/04		
15	12468	14898029	1,400,825.00	USA	1,400,825.00	135673	12/24/04	1/2/05	
16	12471	39045213	8,738,600,640.00	Brazil	3,675,848,666.21	457232	12/27/04		
17	12476	38763919	2,234,301.30	Argentina	772,847.05	135673	12/24/04		
18	12478	15432147	252,148.50	USA	252,148.50	423286	12/24/04	1/2/05	
19									
20									

In above image, you can see that there are multiple sheets present (bottom left) in this file, which are Customers, Employees, Invoice, Order. The image shows the data of only one sheet – "Invoice".

Reading the data from XLSX file

Let's load the data from XLSX file and define the sheet name. For loading the data you can use the Pandas library in python.

```
import pandas as pd
```

df = pd.read_excel("/home/Loan_Prediction/train.xlsx", sheetname = "Invoice")

Above code will load the sheet "Invoice" from "train.xlsx" file in DataFrame df.

3.3 ZIP files

ZIP format is an archive file format.

What is Archive File format?

In Archive file format, you create a file that contains multiple files along with metadata. An archive file format is used to collect multiple data files together into a single file. This is done for simply compressing the files to use less storage space.

There are many popular computer data archive format for creating archive files. Zip, RAR and Tar being the most popular archive file format for compressing the data.

So, a ZIP file format is a lossless compression format, which means that if you compress the multiple files using ZIP format you can fully recover the data after decompressing the ZIP file. ZIP file format uses many compression algorithms for compressing the documents. You can easily identify a ZIP file by the .zip extension.

Reading a .ZIP file in Python

You can read a zip file by importing the “zipfile” package. Below is the python code which can read the “train.csv” file that is inside the “T.zip”.

```
import zipfile
archive = zipfile.ZipFile('T.zip', 'r')
df = archive.read('train.csv')
```

Here, I have discussed one of the famous archive format and how to open it in python. I am not mentioning other archive formats. If you want to read about different archive formats and their comparisons you can refer this link.

3.4 Plain Text (txt) file format

In Plain Text file format, everything is written in plain text. Usually, this text is in unstructured form and there is no meta-data associated with it. The txt file format can easily be read by any program. But interpreting this is very difficult by a computer program.

Let’s take a simple example of a text File.

The following example shows text file data that contain text:

```
“In my previous article, I introduced you to the basics of Apache Spark, different
data representations
(RDD / DataFrame / Dataset) and basics of operations (Transformation and Action). We
even solved a machine
learning problem from one of our past hackathons. In this article, I will continue
from the place I left in
my previous article. I will focus on manipulating RDD in PySpark by applying
operations
(Transformation and Actions).”
```

Suppose the above text written in a file called text.txt and you want to read this so you can refer the below code.

```
text_file = open("text.txt", "r")
lines = text_file.read()
```

3.5 JSON file format

JavaScript Object Notation(JSON) is a text-based open standard designed for exchanging the data over web. JSON format is used for transmitting structured data over the web. The JSON file format can be easily read in any programming language because it is language-independent data format.

Let's take an example of a JSON file

The following example shows how a typical JSON file stores information of employees.

```
{
   "Employee": [

      {

         "id":"1",

         "Name": "Ankit",

         "Sal": "1000",

      },
      {

         "id":"2",

         "Name": "Faizy",

         "Sal": "2000",

      }

   ]

}
```

Reading a JSON file

Let's load the data from JSON file. For loading the data you can use the pandas library in python.

```
import pandas as pd
```

df = pd.read_json("/home/kunal/Downloads/Loan_Prediction/train.json")

3.6 XML file format

XML is also known as Extensible Markup Language. As the name suggests, it is a markup language. It has certain rules for encoding data. XML file format is a human-readable and machine-readable file format. XML is a self-descriptive language designed for sending information over the internet. XML is very similar to HTML, but has some differences. For example, XML does not use predefined tags as HTML.

Let’s take the simple example of XML File format.

The following example shows an xml document that contains the information of an employee.

```
<?xml version="1.0"?>
<contact-info>

<name>Ankit</name>

<company>Anlytics Vidhya</company>

<phone>+9187654321</phone>

</contact-info>
```

The “<?xml version=”1.0″?>” is a XML declaration at the start of the file (it is optional). In this deceleration, v*ersion* specifies the XML version and *encoding* specifies the character encoding used in the document. <contact-info> is a tag in this document. Each XML-tag needs to be closed.

Reading XML in python

For reading the data from XML file you can import xml.etree. ElementTree library.

Let’s import an xml file called train and print its root tag.

```
import xml.etree.ElementTree as ET
tree = ET.parse('/home/sunilray/Desktop/2 sigma/train.xml')
root = tree.getroot()
print root.tag
```

3.7 HTML files

HTML stands for Hyper Text Markup Language. It is the standard markup language which is used for creating Web pages. HTML is used to describe structure of web pages using markup. HTML tags are same as XML but these are predefined. You can easily identify HTML document subsection on basis of tags such as <head> represent the heading of HTML document. <p> “paragraph” paragraph in HTML. HTML is not case sensitive.

The following example shows an HTML document.

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body><h1>My First Heading</h1>
<p>My first paragraph.</p></body>
</html>
```

Each tag in HTML is enclosed under the angular bracket(<>). The <!DOCTYPE html> tag defines that document is in HTML format. <html> is the root tag of this document. The <head> element contains heading part of this document. The <title>, <body>, <h1>, <p> represent the title, body, heading and paragraph respectively in the HTML document.

Reading the HTML file

For reading the HTML file, you can use BeautifulSoup library. Please refer to this tutorial, which will guide you how to parse HTML documents. Beginner's guide to Web Scraping in Python (using BeautifulSoup)

3.8 Image files

Image files are probably the most fascinating file format used in data science. Any computer vision application is based on image processing. So it is necessary to know different image file formats.

Usual image files are 3-Dimensional, having RGB values. But, they can also be 2-Dimensional (grayscale) or 4-Dimensional (having intensity) – an Image consisting of pixels and meta-data associated with it.

Each image consists one or more frames of pixels. And each frame is made up of two-dimensional array of pixel values. Pixel values can be of any intensity. Meta-data associated with an image, can be an image type (.png) or pixel dimensions.

Let's take the example of an image by loading it.

```
from scipy import misc
f = misc.face()
misc.imsave('face.png', f) # uses the Image module (PIL)
import matplotlib.pyplot as plt
plt.imshow(f)
plt.show()
```

Now, let’s check the type of this image and its shape.

```
type(f) , f.shape
```

```
numpy.ndarray,(768, 1024, 3)
```

If you want to read about image processing you can refer this article. This article will teach you image processing with an example – Basics of Image Processing in Python

3.9 Hierarchical Data Format (HDF)

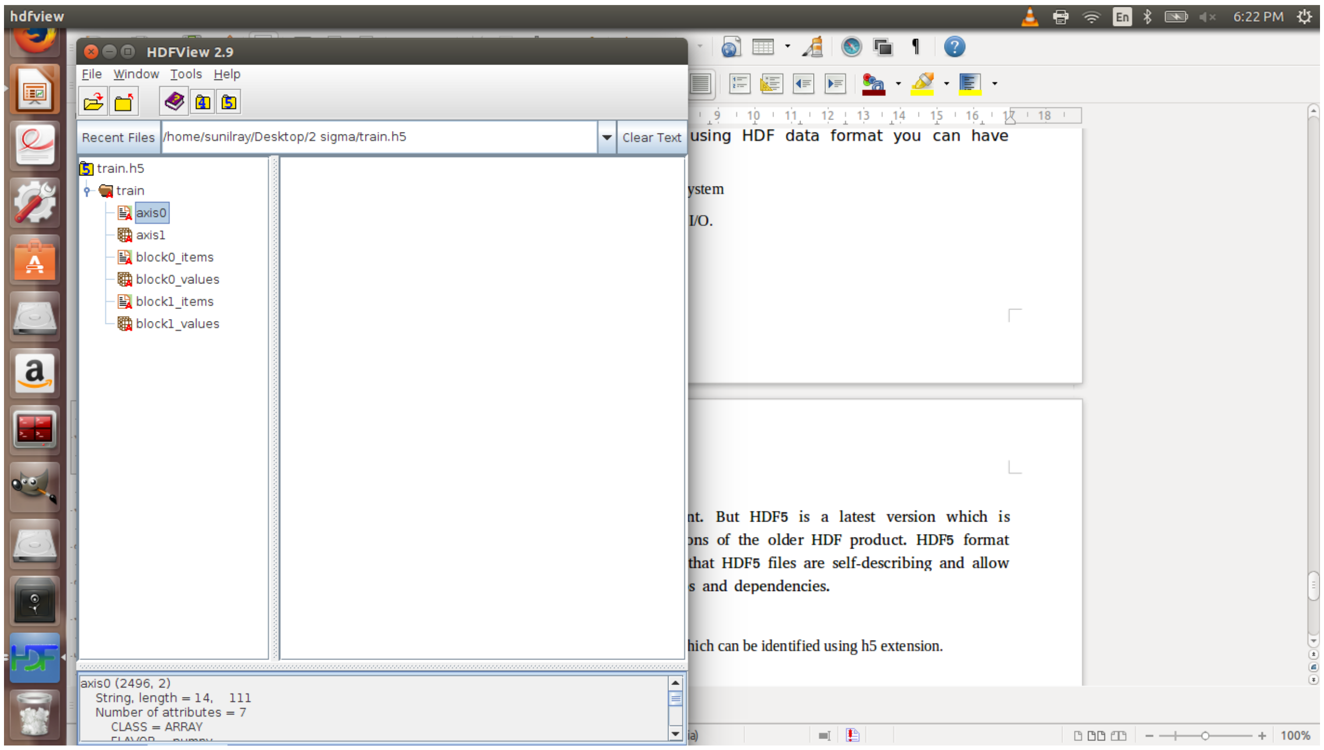
In Hierarchical Data Format (HDF), you can store a large amount of data easily. It is not only used for storing high volumes or complex data but also used for storing small volumes or simple data.

The advantages of using HDF are as mentioned below:

- It can be used in every size and type of system
- It has flexible, efficient storage and fast I/O.
- Many formats support HDF.

There are multiple HDF formats present. But, HDF5 is the latest version which is designed to address some of the limitations of the older HDF file formats. HDF5 format has some similarity with XML. Like XML, HDF5 files are self-describing and allow users to specify complex data relationships and dependencies.

Let’s take the example of an HDF5 file format which can be identified using .h5 extension.



```
train.h5
  train
    axis0
    axis1
    block0_items
    block0_values
    block1_items
    block1_values
```

```
axis0 (2496, 2)
   String, length = 14,    111
   Number of attributes = 7
      CLASS = ARRAY
```

Read the HDF5 file

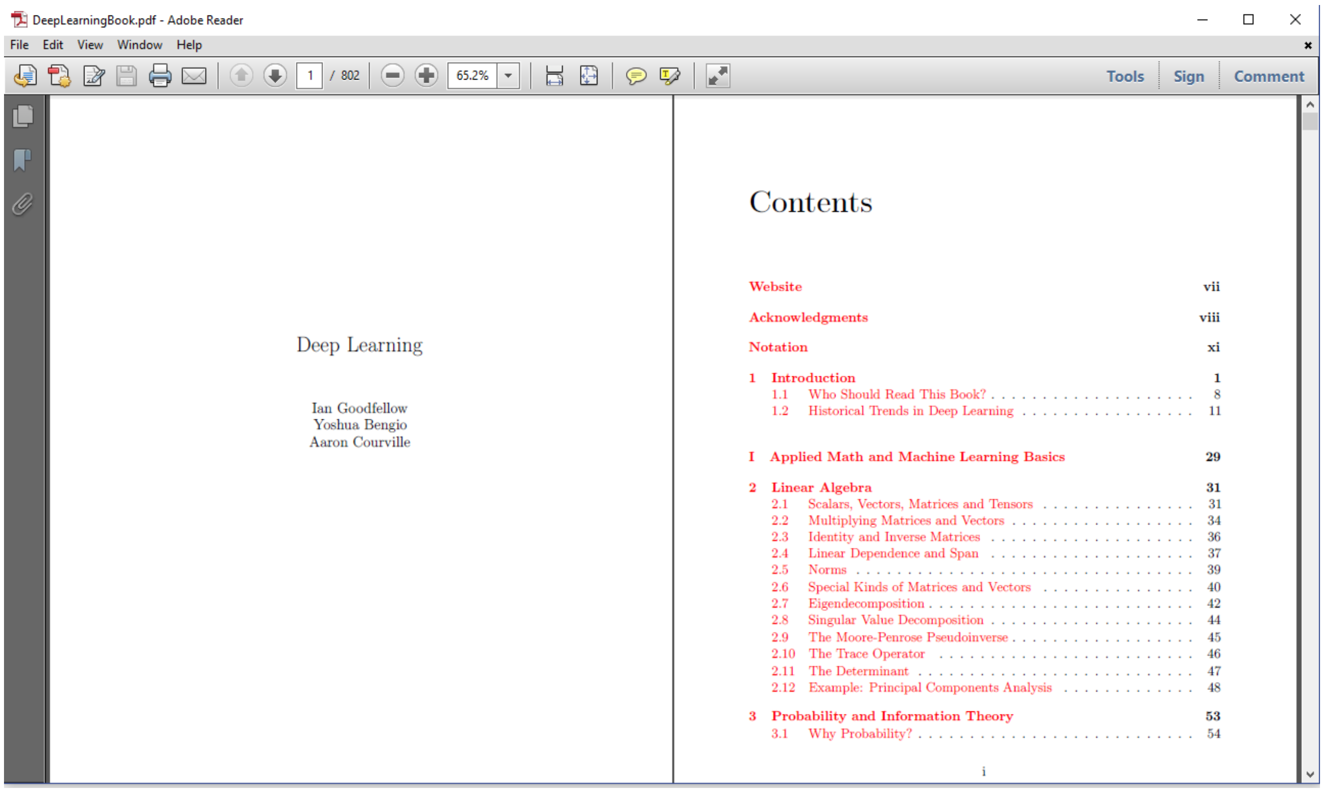
You can read the HDF file using pandas. Below is the python code can load the train.h5 data into the “t”.

t = pd.read_hdf(‘train.h5’)

3.10 PDF file format

PDF (Portable Document Format) is an incredibly useful format used for interpretation and display of text documents along with incorporated graphics. A special feature of a PDF file is that it can be secured by a password.

Here’s an example of a pdf file.


DeepLearningBook.pdf - Adobe Reader
File Edit View Window Help
1 / 802
65.2%
Tools
Sign
Comment

Deep Learning

Ian Goodfellow
Yoshua Bengio
Aaron Courville

Contents

i

Reading a PDF file

On the other hand, reading a PDF format through a program is a complex task. Although there exists a library which do a good job in parsing PDF file, one of them is PDFMiner. To read a PDF file through PDFMiner, you have to:

- Download PDFMiner and install it through the website
- Extract PDF file by the following code

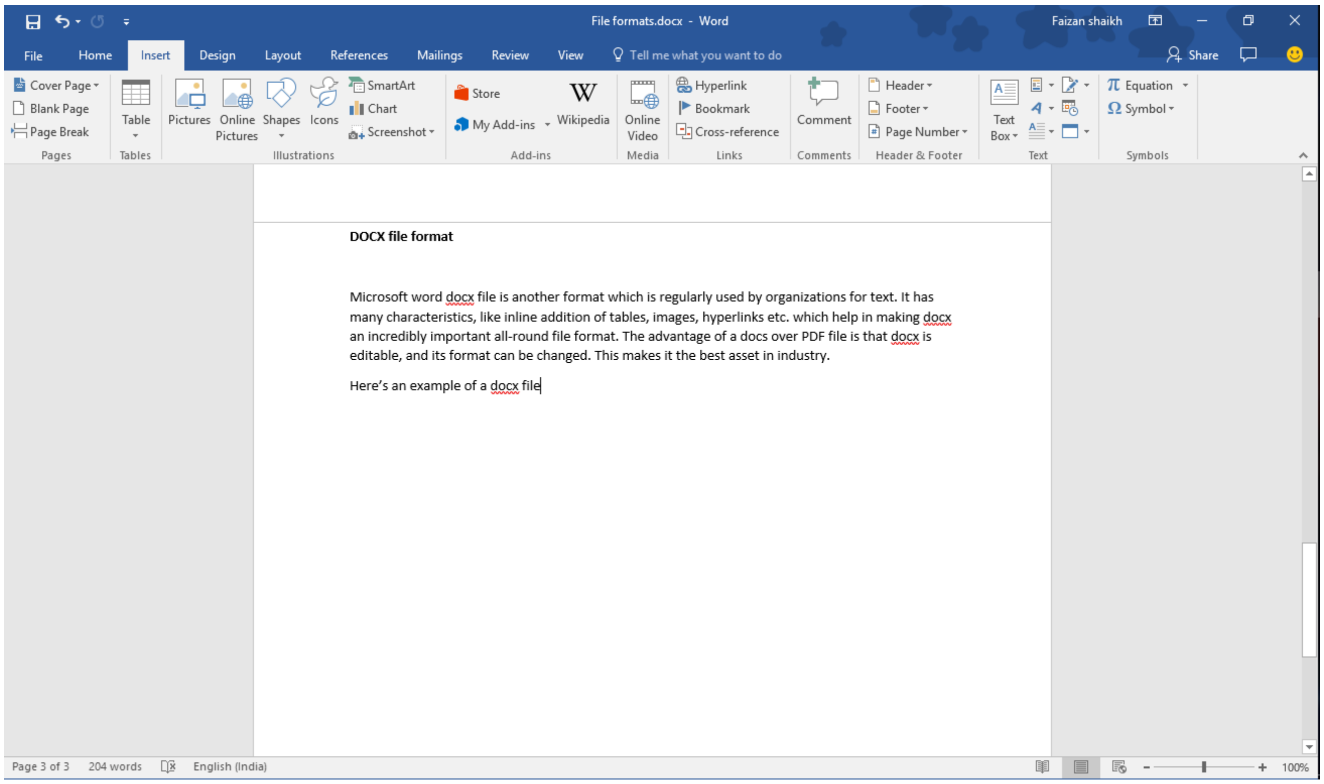
```
pdf2txt.py <pdf_file>.pdf
```

3.11 DOCX file format

Microsoft word docx file is another file format which is regularly used by organizations for text based data. It has many characteristics, like inline addition of tables, images, hyperlinks, etc. which helps in making docx an incredibly important file format.

The advantage of a docx file over a PDF file is that a docx file is editable. You can also change a docx file to any other format.

Here's an example of a docx file:



DOCX file format

Microsoft word docx file is another format which is regularly used by organizations for text. It has many characteristics, like inline addition of tables, images, hyperlinks etc. which help in making docx an incredibly important all-round file format. The advantage of a docs over PDF file is that docx is editable, and its format can be changed. This makes it the best asset in industry.

Here's an example of a docx file

Reading a docx file

Similar to PDF format, python has a community contributed library to parse a docx file. It is called python-docx2txt.

Installing this library is easy through pip by:

```
pip install docx2txt
```

To read a docx file in Python use the following code:

```
import docx2txt
text = docx2txt.process("file.docx")
```

3.12 MP3 file format

MP3 file format comes under the multimedia file formats. Multimedia file formats are similar to image file formats, but they happen to be one the most complex file formats.

In multimedia file formats, you can store variety of data such as text image, graphical, video and audio data. For example, A multimedia format can allow text to be stored as Rich Text Format (RTF) data rather than ASCII data which is a plain-text format.

MP3 is one of the most common audio coding formats for digital audio. A mp3 file format uses the MPEG-1 (Moving Picture Experts Group – 1) encoding format which is a standard for lossy compression of video and audio. In lossy compression, once you have compressed the original file, you cannot recover the original data.

A mp3 file format compresses the quality of audio by filtering out the audio which can not be heard by humans. MP3 compression commonly achieves 75 to 95% reduction in size, so it saves a lot of space.

mp3 File Format Structure

A mp3 file is made up of several frames. A frame can be further divided into a header and data block. We call these sequence of frames an elementary stream.

A header in mp3 usually, identify the beginning of a valid frame and a data blocks contain the (compressed) audio information in terms of frequencies and amplitudes. If you want to know more about mp3 file structure you can refer this link.

Reading the multimedia files in python

For reading or manipulating the multimedia files in Python you can use a library called PyMedia.

3.13 MP4 file format

MP4 file format is used to store videos and movies. It contains multiple images (called frames), which play in form of a video as per a specific time period. There are two methods for interpreting a mp4 file. One is a closed entity, in which the whole video is considered as a single entity. And other is mosaic of images, where each image in the video is considered as a different entity and these images are sampled from the video.

Here's is an example of mp4 video

FRIENDS - Final Thoughts (2004) FULL VIDEO.mp4 - VLC media player
Media Playback Audio Video Subtitle Tools View Help
FRIENDS - Final Thoughts (2004)
00:04
25:49
80%

Reading an mp4 file

MP4 also has a community built library for reading and editing mp4 files, called MoviePy.

You can install the library from this link. To read a mp4 video clip, in Python use the following code.

```
from moviepy.editor import VideoFileClip
clip = VideoFileClip('<video_file>.mp4')
```

You can then display this in jupyter notebook as below

```
ipython_display(clip)
```

End Notes

In this article, I have introduced you to some of the basic file formats, which are used by data scientist on a day to day basis. There are many file formats I have not covered. Good thing is that I don't need to cover all of them in one article.

I hope you found this article helpful. I would encourage you to explore more file formats. Good luck! If you still have any difficulty in understanding a specific data format, I'd like to interact with you in comments. If you have any more doubts or queries feel free to drop in your comments below.