UMBC

DATA 601-W6

Data Loading and Wrangling

Data Wrangling

- We will learn ways and techniques to getting data into Python
- We already see one of these ways: Working with .csv files

Types of Data: Structured Data

- Conforms a data model
- Fields of data will be stored
- How that data will be stored: data type (numeric, alphabetic, date, Boolean)
- Any restrictions on the data input (number of characters, etc.).

Types of Data: Semi-Structured

- There is a structure in the data in terms of tags, rows, columns, hierarchies, fields.
- But not necessarily there are unified restrictions and pre-defined types for fields and tags.

Types of Data: Unstructured Data

- No pre-defined data model
- Not organized in fields, tags, attributes, etc.

Examples:

- social media posts, images, texts.

File Formats (How we record data)

What is a file format?

A file format is a standard way that information is encoded for storage in a computer file. It specifies how bits are used to encode information in a digital storage medium. File formats may be either proprietary or free and may be either unpublished or open.

Why do we need formatting the files?

- Share (conventions)
- Store - Preserve (efficient ways - speed, different needs, loose of data)
- Access (speed)

Why to use so many different formats?

- Speed
- Different nature of the data
- Each has different advantages and disadvantages
- Each has different use cases

File Formats: CSV Files

- Uses .csv extension.
- Data is organized in columns and each column is separated with a column.
- Very common format in data science.
- Looks very similar to spreadsheet files however note that you cannot encode formulas and format.
- Because there is no formatting and formulas it is easier to work with this format for bigger file sizes.
- If we use tab instead of , then the file format is called tsv (tab separated values)

Libraries to work with CSV files

- Pandas
  - Faster to load for larger datasets
  - Mainly for analyzing, applying functions to data
- CSV module in Python
  - For reading and writing csv files only.
  - Built in Python, you don’t have to install.
  - Faster with smaller datasets (<1K rows)

File Formats: JSON Files

- **J**ava**S**cript **O**bject **N**otation
- Based on JavaScript but it is completely language independent
- Consists of name-value pairs
- It is a very common format especially in data interchange of web applications.
- JSON is **not** a programming language

Libraries to work with JSON files

- Python has a built in library called json to read and parse json files.
- We can also use pandas library to read json files.

Other File Formats…(but we will not be covering today)

- XML
- HTML
- Audio (mp4, wav, etc.)
- Image (png, jpeg, jpg, etc.)
- Other (pickle, hdf5, parquet, etc,.)

Character Encoding

- Do we know bytes? Bits?
- Character encoding is basically mapping each character to bytes (Simple huh?)
- As anything this is not also that straightforward!!

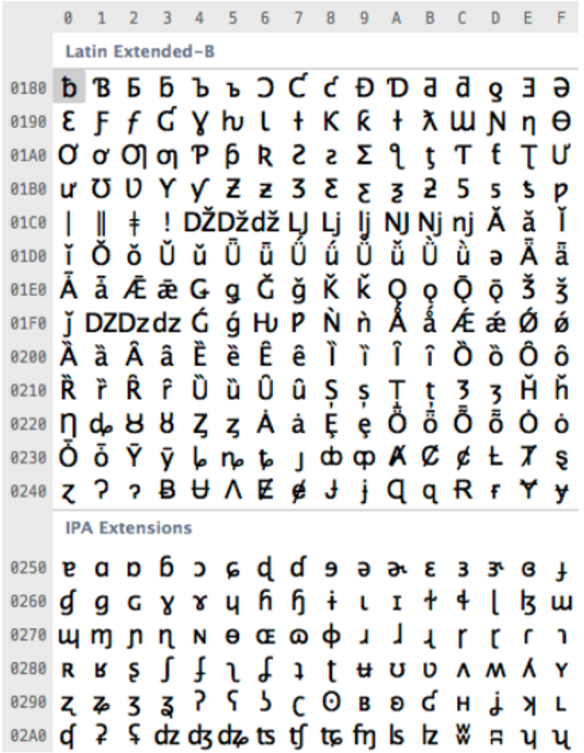
Character Encoding: ASCII

- ASCII Character Set

bits	character
01000001	A
01000010	B
01000011	C
01000100	D
01000101	E
01000110	F

Then What Happened?

- The need for way bigger character sets needed!
- Can you see why?
- Let's take a look at the wikipedia again!



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	Latin Extended-B															
0180	ƀ	Ɓ	Ƃ	ƃ	Ƅ	ƅ	Ɔ	Ƈ	ƈ	Ɖ	Ɗ	Ƌ	ƌ	ƍ	Ǝ	Ə
0190	Ɛ	Ƒ	ƒ	Ɠ	ƔƔ	ƕ	Ɩ	Ɨ	Ƙ	ƙ	ƚ	ƛ	Ɯ	Ɲ	ƞ	Ɵ
01A0	Ơ	ơ	Ƣ	ƣ	Ƥ	ƥ	Ʀ	Ƨ	ƨ	Σ	ƫ	ƫ	Ƭ	ƭ	Ʈ	Ư
01B0	ư	Ʊ	Ʋ	Ƴ	ƴ	Ƶ	ƶ	Ʒ	Ƹ	ƹ	ƺ	ƻ	Ƽ	ƽ	ƾ	ƿ
01C0	|	‖	ǂ	!	DŽ	Dž	dž	LJ	Lj	lj	NJ	Nj	nj	Ǎ	ǎ	Ǐ
01D0	ǐ	Ǒ	ǒ	Ǔ	ǔ	Ǖ	ǖ	Ǘ	ǘ	Ǚ	ǚ	Ǜ	ǜ	ə	Ǟ	ǟ
01E0	Ǡ	ǡ	Ǽ	ǽ	Ǥ	ǥ	Ǧ	ǧ	Ǩ	ǩ	Ǫ	ǫ	Ǭ	ǭ	Ǯ	ǯ
01F0	ǰ	DZ	Dz	dz	Ǵ	ǵ	Ƕ	Ƿ	Ǹ	ǹ	Ǻ	ǻ	Ǽ	ǽ	Ǿ	ǿ
0200	Ȁ	ȁ	Ȃ	ȃ	Ȅ	ȅ	Ȇ	ȇ	Ȉ	ȉ	Ȋ	ȋ	Ȍ	ȍ	Ȏ	ȏ
0210	Ȑ	ȑ	Ȓ	ȓ	Ȕ	ȕ	Ȗ	ȗ	Ș	ș	Ț	ț	Ȝ	ȝ	Ȟ	ȟ
0220	Ƞ	ȡ	Ȣ	ȣ	Ȥ	ȥ	Ȧ	ȧ	Ȩ	ȩ	Ȫ	ȫ	Ȭ	ȭ	Ȯ	ȯ
0230	Ȱ	ȱ	Ȳ	ȳ	ȴ	ȵ	ȶ	ȷ	ȸ	ȹ	Ⱥ	Ȼ	ȼ	Ƚ	Ⱦ	ȿ
0240	ɀ	ɂ	ɂ	Ƀ	Ʉ	Ʌ	Ɇ	ɇ	Ɉ	ɉ	Ɋ	ɋ	Ɍ	ɍ	Ɏ	ɏ
	IPA Extensions															
0250	ɐ	ɑ	ɒ	ɓ	ɔ	ɕ	ɖ	ɗ	ɘ	ə	ɚ	ɛ	ɜ	ɝ	ɞ	ɟ
0260	ɠ	ɡ	ɢ	ɣ	ɤ	ɥ	ɦ	ɧ	ɨ	ɩ	ɪ	ɫ	ɬ	ɭ	ɮ	ɯ
0270	ɰ	ɱ	ɲ	ɳ	ɴ	ɵ	ɶ	ɷ	ɸ	ɹ	ɺ	ɻ	ɼ	ɽ	ɾ	ɿ
0280	ʀ	ʁ	ʂ	ʃ	ʄ	ʅ	ʆ	ʇ	ʈ	ʉ	ʊ	ʋ	ʌ	ʍ	ʎ	ʏ
0290	ʐ	ʑ	ʒ	ʓ	ʔ	ʕ	ʖ	ʗ	ʘ	ʙ	ʚ	ʛ	ʜ	ʝ	ʞ	ʟ
02A0	ʠ	ʡ	ʢ	ʣ	ʤ	ʥ	ʦ	ʧ	ʨ	ʩ	ʪ	ʫ	ʬ	ʭ	ʮ	ʯ

Why Should we care?

```
pd.read_csv('/content/PoliceShootingsUS.csv')
```

```
---------------------------------------------------------------------------
UnicodeDecodeError                        Traceback (most recent call last)
pandas/_libs/parsers.pyx in pandas._libs.parsers.TextReader._convert_tokens()

pandas/_libs/parsers.pyx in pandas._libs.parsers.TextReader._convert_with_dtype()

pandas/_libs/parsers.pyx in pandas._libs.parsers.TextReader._string_convert()

pandas/_libs/parsers.pyx in pandas._libs.parsers._string_box_utf8()

UnicodeDecodeError: 'utf-8' codec can't decode byte 0x96 in position 2: invalid start byte

During handling of the above exception, another exception occurred:

UnicodeDecodeError                        Traceback (most recent call last)
<ipython-input-180-04c68a038be7> in <module>()
----> 1 pd.read_csv('/content/PoliceShootingsUS.csv')
```

Solution?

- Chardet library
- We will discuss in the second part of the lecture...