Reading Data

GITAA
Transforming careers

In this lecture

- File formats
- Commonly used file formats
- Read data from
  - .csv format
  - .xlsx format
  - .txt format

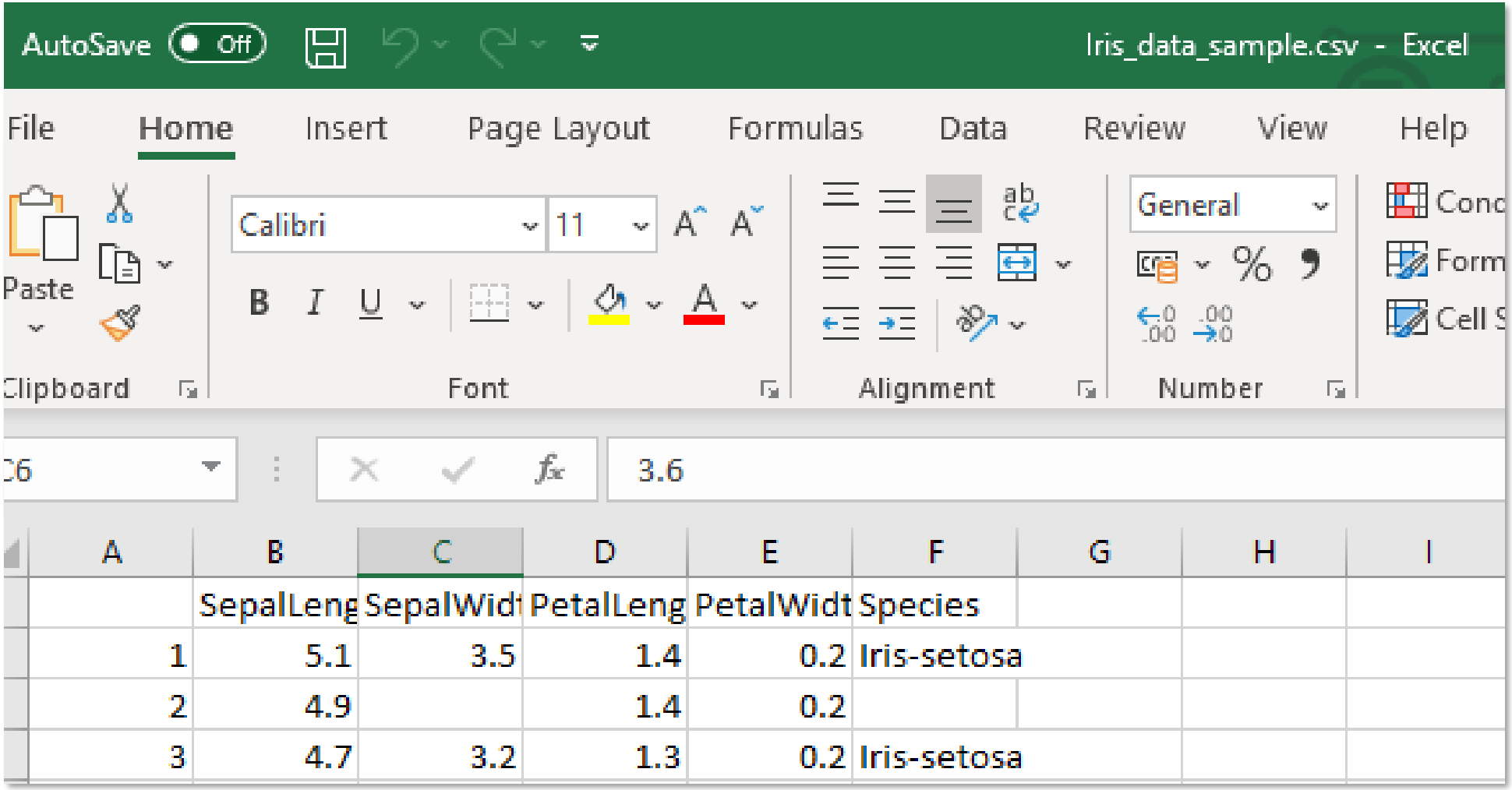
File format

- Standard way in which data is collected and stored
- Most commonly used format for storing data is the spreadsheet format where data is stored in rows and columns
  - Each row is called a record
  - Each column in a spreadsheet holds data belonging to same data type
- Commonly used spreadsheet formats are comma separated values and excel sheets
- Other formats include plain text, json, html, mp3,mp4 etc.

Comma separated values

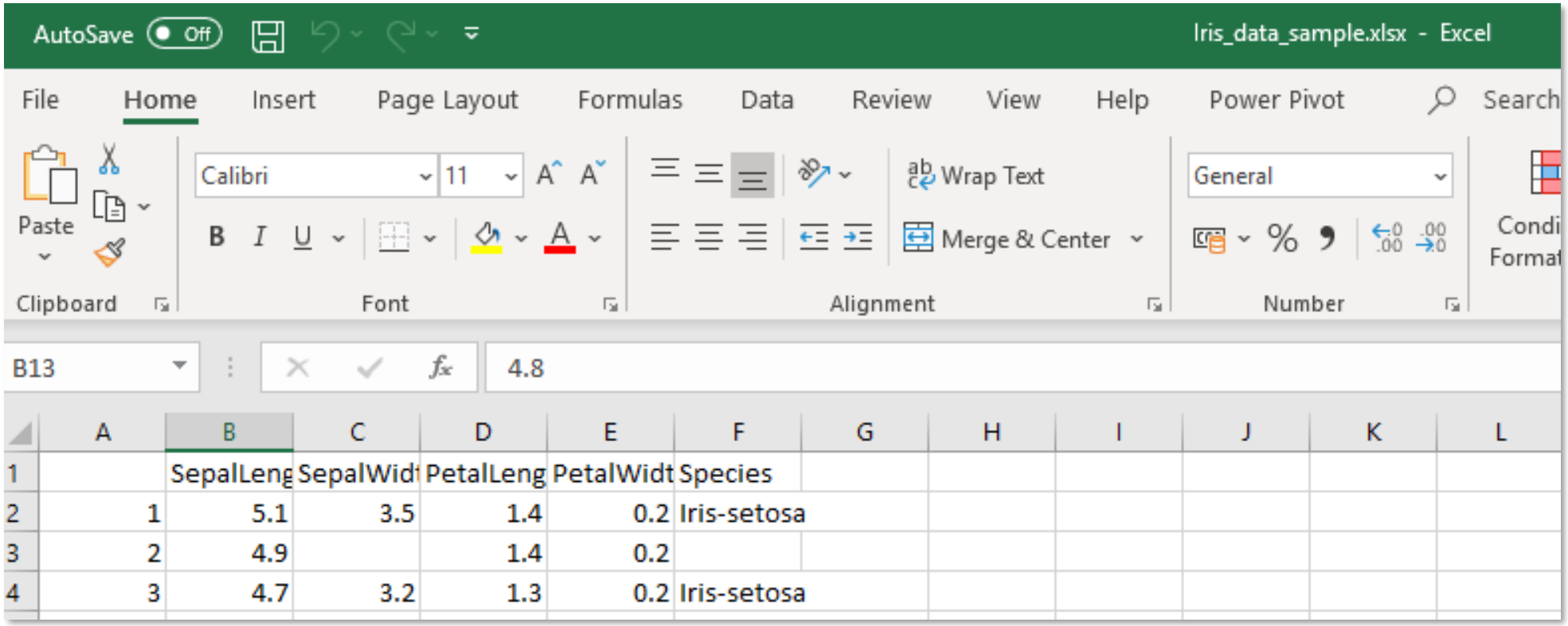
- Spreadsheet format
- Format ***'.csv'***
- Each record is separated by a comma
- Files where records are separated using a tab are called tab separated values
- .csv files can be opened with notepad or Microsoft excel

Comma separated values



	A	B	C	D	E	F
		SepalLeng	SepalWidt	PetalLeng	PetalWidt	Species
	1	5.1	3.5	1.4	0.2	Iris-setosa
	2	4.9		1.4	0.2	
	3	4.7	3.2	1.3	0.2	Iris-setosa

Excel spreadsheets



	A	B	C	D	E	F	G	H	I	J	K	L
1		SepalLeng	SepalWid	PetalLeng	PetalWidt	Species						
2	1	5.1	3.5	1.4	0.2	Iris-setosa						
3	2	4.9		1.4	0.2							
4	3	4.7	3.2	1.3	0.2	Iris-setosa						

Text format

- Consists of plain text or records
- Format ***'.txt'***

SherlockHolmes.txt - Notepad

File Edit Format View Help

```
Project Gutenberg's The Adventures of Sherlock Holmes, by Arthur
Conan Doyle

This eBook is for the use of anyone anywhere at no cost and with
almost no restrictions whatsoever.  You may copy it, give it away
or
re-use it under the terms of the Project Gutenberg License included
with this eBook or online at www.gutenberg.net
```

Importing Data

Importing data into Spyder

- Importing necessary libraries

```
import os
```

'os' library to change the working directory

```
import pandas as pd
```

'pandas' library to work with dataframes

- Changing the working directory

```
os. chdir("D:\Pandas")
```

Comma separated values

- Importing data

```
data_csv=pd.read_csv('Iris_data_sample.csv')
```

- Blank cells read as ‘nan’

Index	Unnamed: 0	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-set…
1	2	4.9	nan	1.4	0.2	nan
2	3	4.7	3.2	1.3	0.2	Iris-set…
3	4	??	3.1	1.5	0.2	Iris-set…
4	5	5	3.6	###	0.2	Iris-set…

Comma separated values

- Removing the extra id column by passing `index_col=0`

```
data_csv=pd.read_csv('Iris_data_sample.csv',index_col=0)
```

Index	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
1	5.1	3.5	1.4	0.2	Iris-set…
2	4.9	nan	1.4	0.2	nan
3	4.7	3.2	1.3	0.2	Iris-set…
4	??	3.1	1.5	0.2	Iris-set…
5	5	3.6	###	0.2	Iris-set…

- Replacing '??' and '# # #' as missing values

Comma separated values

- Junk values can be converted to missing values by passing them as a list to the parameter '`na_values`'

```
data_csv=pd.read_csv('Iris_data_sample.csv',
                     index_col=0,na_values=["??"])
```

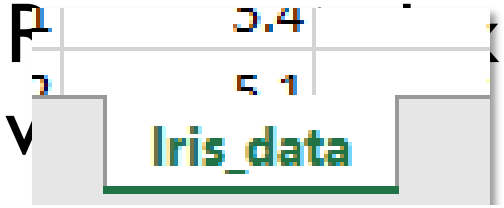
```
data_csv=pd.read_csv('Iris_data_sample.csv',
                     index_col=0,na_values=["??","###"])
```

Excel spreadsheets

- Importing data

```
data_xlsx=pd.read_excel('Iris_data_sample.xlsx',
                        sheet_name='Iris_data')
```

- F column and replace '??' and '# # #' as missing v



Sheet name

```
data_xlsx=pd.read_excel('Iris_data_sample.xlsx',index_col=0,
                        na_values=["??","###"])
```

Text format

- Importing data

```
data_txt1=pd.read_table('Iris_data_sample.txt')
```

Name	Type
data_txt1	DataFrame

Index	"SepalLengthCm" "SepalWidthCm" "PetalLengthCm" "PetalWidthCm" "Species"
0	1 1 5.1 3.5 1.4 0.2 "Iris-setosa"
1	2 2 4.9 3 1.4 0.2 "Iris-setosa"
2	3 3 4.7 3.2 1.3 0.2 "Iris-setosa"

- All columns read and stored in a single column of dataframe
- In order to avoid this, provide a delimiter to the parameters '`sep`' or '`delimiter`'

Text format

- Default delimiter is tab represented by '\t'

```
data_txt1=pd.read_table('Iris_data_sample.txt',sep='\t')
data_txt1=pd.read_table('Iris_data_sample.txt',delimiter="\t")
```

- Tab delimiter might not always work

Index	"SepalLengthCm" "SepalWidthCm" "PetalLengthCm" "PetalWidthCm" "Species"
0	1 1 5.1 3.5 1.4 0.2 "Iris-setosa"
1	2 2 4.9 3 1.4 0.2 "Iris-setosa"
2	3 3 4.7 3.2 1.3 0.2 "Iris-setosa"

Text format

- Other commonly used delimiters are commas and blanks
- In this case using a comma as a delimiter also gives the earlier output
-

Name	Type
data_txt1	DataFrame

Index	"SepalLengthCm" "SepalWidthCm" "PetalLengthCm" "PetalWidthCm" "Species"
0	1 1 5.1 3.5 1.4 0.2 "Iris-setosa"
2	3 3 4.7 3.2 1.3 0.2 "Iris-setosa"

```
data_txt1=pd.read_table('Iris_data_sample.txt',delimiter=" ")
```

Index	Unnamed: 0	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
1	1	5.1	3.5	1.4	0.2	Iris-set…
2	2	4.9	3	1.4	0.2	Iris-set…

Text format

- Remove index column and replace '??' and '# # #' as missing values
- Instead of using `read_table()`, `read_csv()` can also be used to read ***.txt*** files

```
data_txt2=pd.read_csv('Iris_data_sample.txt',delimiter=" ")
```


operation == "MIRROR_X":
mirror_mod.use_x = True
mirror_mod.use_y = False
mirror_mod.use_z = False
_operation == "MIRROR_Y":
mirror_mod.use_x = False
mirror_mod.use_y = True
mirror_mod.use_z = False
_operation == "MIRROR_Z":
mirror_mod.use_x = False
mirror_mod.use_y = False
mirror_mod.use_z = True
mirror_ob.select = 0
bpy.context.selected_ob
int("please select exactly

THANK YOU