Collaborative Document

2025-06-03 Introduction to Python

Welcome to The Workshop Collaborative Document.

This Document is synchronized as you type, so that everyone viewing this page sees the same text. This allows you to collaborate seamlessly on documents.

This is the Document for today: https://edu.nl/vehjx

Collaborative Document day 1: https://edu.nl/k8axf

Collaborative Document day 2: https://edu.nl/vehjx

Code of Conduct

Participants are expected to follow these guidelines:

- Use welcoming and inclusive language.
- Be respectful of different viewpoints and experiences.
- Gracefully accept constructive criticism.
- Focus on what is best for the community.
- Show courtesy and respect towards other community members.

If you feel that the code of conduct is breached, please talk to one of the instructors (if the complaint is for one of the participants) or send an email to training@esciencecenter.nl (if the complaint is for one of the instructors).

Certificate of attendance

If you attend the full workshop you can request a certificate of attendance by emailing to training@esciencecenter.nl.
Please request your certificate within 8 months after the workshop, as we will delete all personal identifyable information after this period.

⚖ License

Getting help

To ask a question, raise your hand in zoom. Click on the icon labeled "Reactions" in the toolbar on the bottom center of your screen,
then click the button 'Raise Hand '. For urgent questions, just unmute and speak up!

You can also ask questions or type 'I need help' in the chat window and helpers will try to help you.
Please note it is not necessary to monitor the chat - the helpers will make sure that relevant questions are addressed in a plenary way.
(By the way, off-topic questions will still be answered in the chat)

Workshop website

link (https://esciencecenter-digital-skills.github.io/2025-06-03-dc-socsci-python-odissei/)

Setup

link (https://datacarpentry.github.io/python-socialsci/index.html#setup)

Download files (https://github.com/esciencecenter-digital-skills/2025-06-03-dc-socsci-python-odissei/raw/refs/heads/main/files/data.zip)

Instructors

Helpers

□□□□□□□□□□♂□ Roll Call

Name/ pronouns (optional) / job, role / social media (twitter, github, ...) / background or interests (optional) / city

Icebreaker

Show us something green on camera!

□ Agenda

Time	Topic
09:00	Welcome and icebreaker
09:15	Reading data from a file using Pandas
10:15	Coffee break
10:30	Extracting row and columns and data aggregation using Pandas
11:30	Coffee break
11:45	Data visualization using Matplotlib
12:45	Wrap-up
13:00	END

Exercises

Exercise: Reading in data using pandas

Pandas sep

1. What happens if you forget to specify `sep='\t'` when reading a tab delimited dataset?
2. (Bonus): Use the help of the `read_csv` function. How do you read in the first 10 lines of the .tab file? How do you read in a subset of the columns?

Exercise: Print all columns

1. When we asked for the column names and their data types, the output was abridged, i.e. we didn't get the values for all of the columns. Can you write a small piece of code which will print all of the values on separate lines. Paste your code in the collaborative document.
2. (Bonus) Using if statements, write a piece of code that prints 'big dataset' if the number of columns is larger than 10, and 'small dataset' if the number of columns is smaller.
3. (Bonus) Write a function that returns whether a dataset has more than 10 columns. (it should return a boolean value)

Extracting row and columns

Exercise: Pandas columns

What happens if you:

1. List the columns you want out of order from the way they appear in the file?
2. Put the same column name in twice?
3. Put in a non-existing column name? (a.k.a Typo)

Exercise: Number of values

Compare the count values returned for the `B_no_membrs` and the `E19_period_use` variables.

1. Why do you think they are different?
2. How does this affect the calculation of the mean values?
3. (optional): Use your search engine to find how to deal with missing values in pandas.

Answers

Exercise: aggregation

Discuss and write the answers in the collaborative document.

1. Read in the SAFI_results.csv dataset.
2. Get a list of the different E26_affect_conflicts values.
3. Groupby E26_affect_conflicts and describe the results.
4. (optional) How many of the respondents never had any conflicts?
5. (optional) Using groupby find out whether farms that use water ('E01_water_use') have more plots ('D_plots_count') than farms that do not use water.

Plotting

Exercise: Plotting with Pandas

1. Make a histogram of the number of buildings in the compound (`buildings_in_compound`). Determine the appropriate number of bins, then include the `bins` argument in your function to improve the chart.
2. Make a scatter plot of `years_farm` vs `years_liv` and color the points by `buildings_in_compound`.
3. (Optional) Make a bar plot of the mean number of rooms per wall type (use columns `rooms` and `respondent_wall_type`). Hint: check out the function `plot.bar`, and recall how to use `groupby` to apply statistics to grouped data.

(Optional) Exercise: Customize your plot

Revisit your favorite plot we've made so far, or make one with your own data then:

- add axes labels
- add a title
- save it in two different formats
- make it pretty
-

Collaborative Notes

Datasets we are using today

SN7577

Audit of Political Engagement 11, 2013 (https://beta.ukdataservice.ac.uk/datacatalogue/studies/study?id=7577&type=data%20catalogue#!/details)

SAFI

Studying African Farmer-Led Irrigation (https://figshare.com/articles/dataset/SAFI_Survey_Results/6262019/4)

Quick Recap

Yesterday we learned about types and basic operations in Python. We also learned how to handle string with `len()`, `.split()`, `.upper()`

We also learned about if statements, while and for loops. The basic syntax for these is:

```
# if statement
if condition:
    argument to run
elif condition2:
    other argument
else:
    final argument

# while loops
while condition:
    run this

# for loops
for item in list:
    run this
```

We also learned how to write functions:

```
def function_name(arg1, arg2):
    """Docstrings"""
    code to run
    return what the function outputs
```

Reading in data with Pandas

If Pandas is not installed, you can run the following at the top of the notebook:

```
!pip install pandas
```

First let's import Pandas and look at common functions:

```
import pandas

pd.read_csv
pd.DataFrame
pd.Series
```

Usually we don't create dataframes manually but rather load in data:

```
pd.read_csv("data/SN7577.tab", sep="\t")
```

Make sure you have the correct path to your data.
Since from the extension we can telll that this is tab separated data, we can specify the separator with the sep argument.

To find which directory your notebook is:

```
import os
os.getcwd()
```

Then you can put the data folder to the same path. (You can also specify an absolute path if that is easier for you)

You can explore your dataframe with the following commands:

```
data.head() # gives you the first few lines
data.tail() # gives you the last few lines
len(data) # gives number of rows
data.columns # lists column names
data.shape() # gives shape of the dataframe
data.dtypes # gives types of columns
```

with data.head() and data.tail() you can specify the number of rows you want to show.

```
data.head(5)
data.tail(5)
```

Manipulating data with Pandas

```
import pandas as pd

df_SN7577 = pd.read_csv("data/SN7577.tab", sep="\t")

df_SN7577.describe()
```

Let's only read in some columns, you can use both numbers or column names to choose:

```
df_SN7577_some_cols = pd.read_csv("data/SN7577.tab", sep='\t', usecols=
[0,1,2,173,174,175])
```

or:

```
df_SN7577_some_cols = pd.read_csv("data/SN7577.tab", sep='\t', usecols= ['Q1',
'Q2', 'Q3', 'sex', 'age', 'agegroups'])
```

we can view individual columns with:

```
df_SN7577.Q1
```

For multiple columns:

```
df_SN7577[['Q1', 'Q2', 'Q3']]
```

The columns are taken in the order that you specify them.

You can ask for a range of rows:

```
df_SN7577[0:2]
```

You can also filter rows according to a criteria:

```
df_SN7577[(df_SN7577.Q2 == -1) & (df_SN7577.numage > 60)]
```

In this example we are only keep rows where Q2 is equal to -1 and the age is over 60.

`.iloc()` let's you select rows (and columns) more easily, and it uses rows (and column) indices:

```
df_SN7577.iloc[1:4]
```

There is also the `.loc` method to index using **labels** (e.g. row and column names):

```
df_SN7577.loc[1:4, ["Q1", "Q2", "Q3"]]
```

Aggregates

We can extract some simple stats from the dataframe:

```
df_SN7577.describe()
```

Load the SAFI dataset:

```
df_SAFI = pd.read_csv("data/SAFI_results.csv")

df_SAFI.describe()
```

```
df_SAFI.B_no_membrs.count() # count number of records
df_SAFI.E19_period_use.dropna() # drop nan values
df_SAFI.B_no_membrs.max() # Extract the max value
df_SAFI.B_no_membrs.mean() # Takes the mean of the column
df_SAFI.B_no_membrs.notnull() # Take all rows that are not null
df_SAFI.B_no_membrs.std() # Takes the standard deviation of the column
df_SAFI.B_no_membrs.sum() # Sum of all rows
```

Here we are doing those stats for the column "B_no_membrs" but you can replace that for any column.

We can drop all nan values:

```
df_SAFI.E19_period_use.dropna(inplace=True)
```

To filter all rows for which a column is not a missing value:

```
df_SAFI[df_SAFI.B_no_membrs.notnull()]
```

Replace NaN values with value of your choice:

```
df_SAFI['E19_period_use'].fillna(0, inplace=True)
```

If we want to extract the unique values from a column with categorical values:

```
pd.unique(df_SAFI.C01_respondent_roof_type)
```

We can group the dataset by these unique values by doing:

```
grouped_data = df_SAFI.groupby("C01_respondent_roof_type")

grouped_data.describe()
```

Visualising data

You need to import matplotlib

We are starting a new notebook:

```
import pandas as pd

df = pd.read_csv("data/SAFI_full_shortname.csv")
df
```

```
df["years_liv"].hist() # Creates a histogram
```

We can modify the number of bins in the histogram

```
df["years_liv"].hist(bins=20)
```

You can also do it at a dataframe level:

```
df.hist(column="years_liv")
```

The advantage of using this method is that we have access to other columns to further group the data etc:

```
df.hist(column="years_liv", by="village")
```

```
df.hist(column="years_liv", by="village", sharex=True, layout=(1,3),figsize=
(12,3))
```

Now let's try a scatterplot:

```
df.plot.scatter(x="gps_Longitude", y="gps_Latitude")
```

We can colour our points with another column:

```
df.plot.scatter(x="gps_Longitude", y="gps_Latitude", c="gps_Altitude",
colormap="viridis")
```

The default colormap is greys so it's nice to specify it. You can also use the "c" argument to just change the colour of the dots e.g. `c='red'`, you can also change the markerstyle with `marker`, there are [several options (https://matplotlib.org/stable/api/markers_api.html#module-matplotlib.markers)](https://matplotlib.org/stable/api/markers_api.html#module-matplotlib.markers) for it such as `-`, `--`, `x` etc.

```
df.boxplot(column="buildings_in_compound")
```

Using matplotlib commands to further customise your plot:

```
import matplotlib.pyplot as plt

df.boxplot(column="buildings_in_compound", by="village")
plt.xlabel("")
plt.ylabel("Number of buildings")
plt.grid(False)
plt.savefig("figure.png") # Save the figure
```

Resources

- [Pandas (https://pandas.pydata.org/)](https://pandas.pydata.org/)
- [Pandas comparison with R (https://pandas.pydata.org/docs/getting_started/comparison/comparison_with_r.html)](https://pandas.pydata.org/docs/getting_started/comparison/comparison_with_r.html)

- JupyterLite (https://esciencecenter-digital-skills.github.io/socsci-python-jupyterlite/)
- Lesson material (https://datacarpentry.github.io/python-socialsci/index.html)
- Matplotlib (includes a gallery of examples) (https://matplotlib.org/)
- Seaborn (alternative plotting library) (https://seaborn.pydata.org/)
- hvPlot (interactive plotting library) (https://hvplot.holoviz.org/)

:+1: :-1: Post-workshop survey link

https://www.surveymonkey.com/r/HCLFQHR