Python assumed knowledge

1. Data is held in structured data frames
2. Python is a programming language that can be used for data analysis
3. How to open a csv file in a Jupyter notebook
4. How to use a Jupyter notebook to write, edit and run Python code

Summarising datasets in Python (Part 1)

Learning intentions

We will be learning to summarise datasets in Python, specifically to

- summarise **complete datasets**
- perform summary calculations for **single variables**, such as the **total, count**, **min/max** and **average** values
- perform summary calculations for **multiple variables**

Background

When trying to solve a problem in data science, understanding the data you have is fundamental.

Rows of data can be **filtered** and **sorted** to help you understand your data.

In this lesson we will look at how you can also **summarise and group rows of data.**

Why this is important?

Some benefits of summarising and grouping data are,

Makes the data **easier to work** with

Focus on the important messages

Allows you to **simplify** your dataset

Helps you **describe** your data, e.g., What is the total? How many rows to you have?

Definition

Summarise

To condense the rows in a dataset (often to a single value) by performing a calculation on the data items within a variable.

Summarising data

In the similar way that you can perform calculations on columns of data, you can **perform calculations on rows of data**.

The most common calculations performed on rows of data are,

- Count (number of rows)
- Total
- Average

Show me…

When you are summarising a dataset you can **select the required variables** and then **summarise them.**

month	number_sold	price	income
Jan	6	41	246
Feb	5	27	135
Mar	4	46	184
Apr	6	28	168
May	2	41	82

Summarise

total_number_sold	total_income
23	851

Example

This dataset shows the test results for 5 pupils. Each row shows the % score a pupil achieved in 3 tests.

Summarise the test results in this dataset by calculating,

- Count
- Average (mean)
- Maximum test score
- Minimum test score

pupil_ID	test_1_%	test_2_%	test_3_%
GH1254	50	36	72
SE1547	45	64	94
DM4758	90	48	78
KL4758	32	93	52
PM4575	85	86	92

Example

Summarise the test results in this dataset.

Original dataset

pupil_ID	test_1_%	test_2_%	test_3_%
GH1254	50	36	72
SE1547	45	64	94
DM4758	90	48	78
KL4758	32	93	52
PM4575	85	86	92

Summary

	test_1_%	test_2_%	test_3_%
Count	5	5	5
Average	60	65	78
Maximum	90	93	94
Minimum	32	36	52

Your turn…

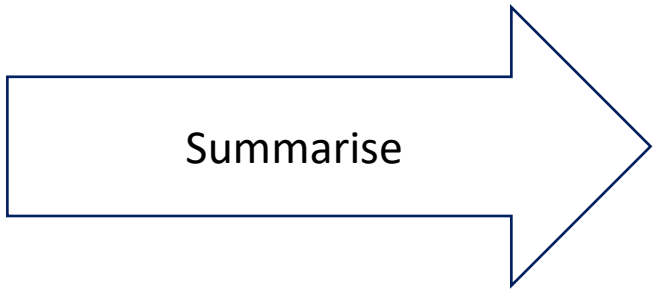
What do you think the **count, maximum and minimum** would be in this dataset?

ocean	depth_m
Pacific	3970
Atlantic	3646
Indian	3741
Arctic	1205

Your turn…

What do you think the **count, maximum and minimum** would be in this dataset?

ocean	depth_m
Pacific	3970
Atlantic	3646
Indian	3741
Arctic	1205



summary	depth_m
Count	4
Maximum	3970
Minimum	1205

Summarise a Complete Data Frame in Python

The pandas **describe()** function can be used to provide a **set of summary calculations** for each variable in a data frame. This is useful when you want an **overview** of a data frame.

The summary calculations it reports depends on the data type of each variable (numeric or text) in the data frame.

Here are some of the summary calculations **describe()** provides for numeric and text data types.

numeric variables
• **count** - a count of the number of values • **min** - the minimum value • **max** - the maximum value • **mean** - the average (mean) value

text variables
• **count** - a count of the number of values • **unique** - the number of unique values • **top** - the most common value • **freq** - how often the most common value occurs

Summarise a Complete Data Frame in Python

fruit_and_veg

product	colour	price_£	sales_£
apple	pink	1.00	53.00
banana	yellow	0.50	40.50
carrot	orange	0.50	37.00
dragon fruit	pink	1.20	12.00
apple	green	1.00	20.00

Pass the argument **include = 'all'** to describe **all** of the variables.

```
fruit_and_veg.describe(include='all')
```

data frame

summary

	product	colour	price_£	sales_£
count	5	5	5	5
unique	4	4	NaN	NaN
top	apple	pink	NaN	NaN
freq	2	2	NaN	NaN
mean	NaN	NaN	0.84	32.50
min	NaN	NaN	0.50	15.00
max	NaN	NaN	1.20	53.00

NaN means '**N**ot **a** **N**umber' and is used to show missing data.

*QUESTION: why has the **mean** not been calculated for **product** and **unique** not been calculated for **price_£**?*

Next steps

Complete the **Setup** and **Summarise a complete data frame** sections of the

‘Summarising Datasets Part 1' Jupyter Notebook.

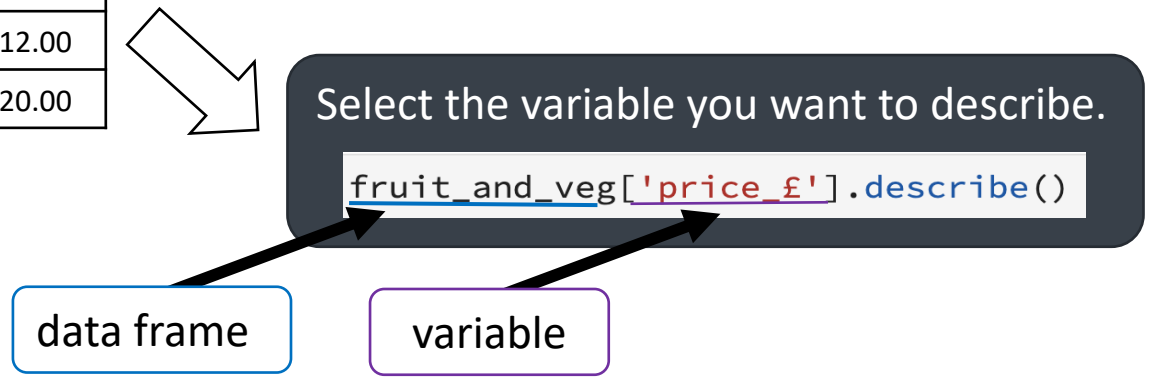
Summarise a Single Variable in Python using describe()

As well as being used on a complete data frame, the **describe()** function can also be used with a **single variable**.

This is useful when you want to quickly see multiple summary calculations for a single variable.

fruit_and_veg

product	colour	price_£	sales_£
apple	pink	1.00	53.00
banana	yellow	0.50	40.50
carrot	orange	0.50	37.00
dragon fruit	pink	1.20	12.00
apple	green	1.00	20.00



summary

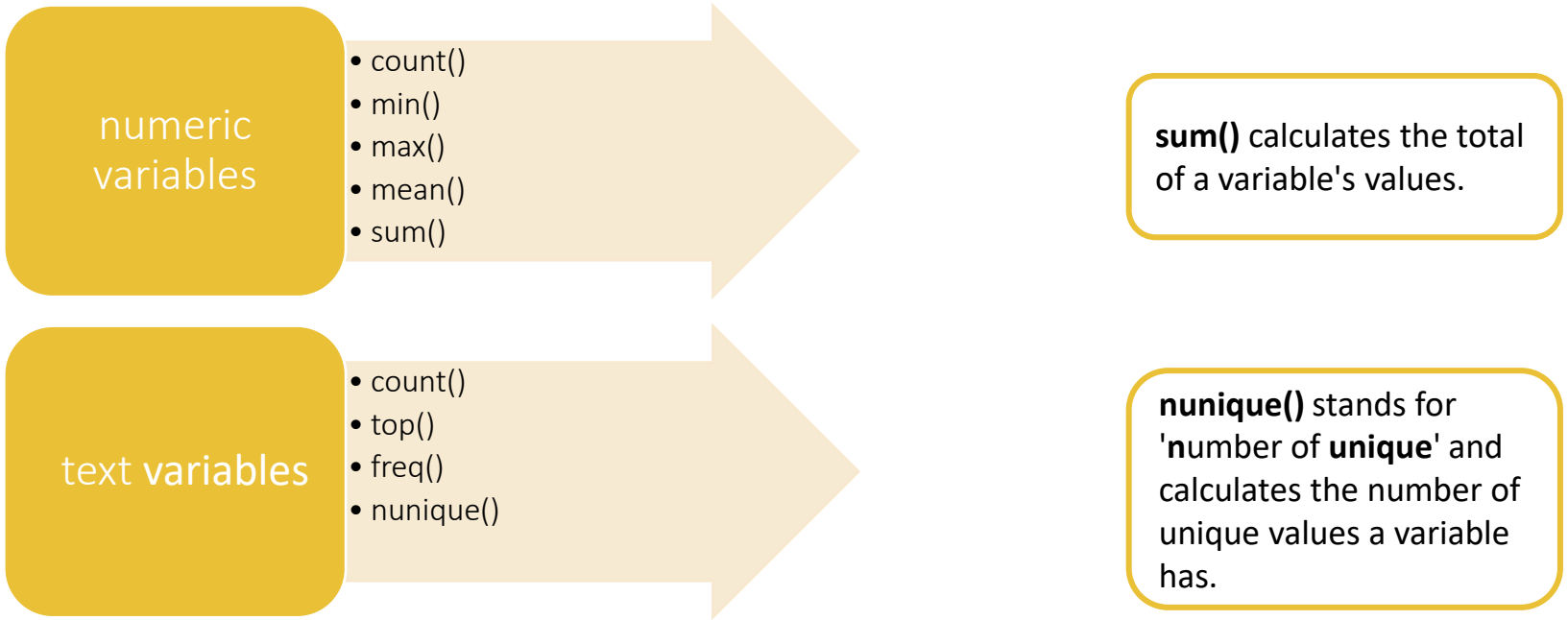
	price_£
count	5
mean	0.84
min	0.50
max	1.20

Summarise a Single Variable in Python

Sometimes you only want to perform a **single summary calculation** for a variable.

Pandas provides a variety of functions to let you do this.

As with **describe()**, the summary calculations that can be performed depend on the data type of the variable.

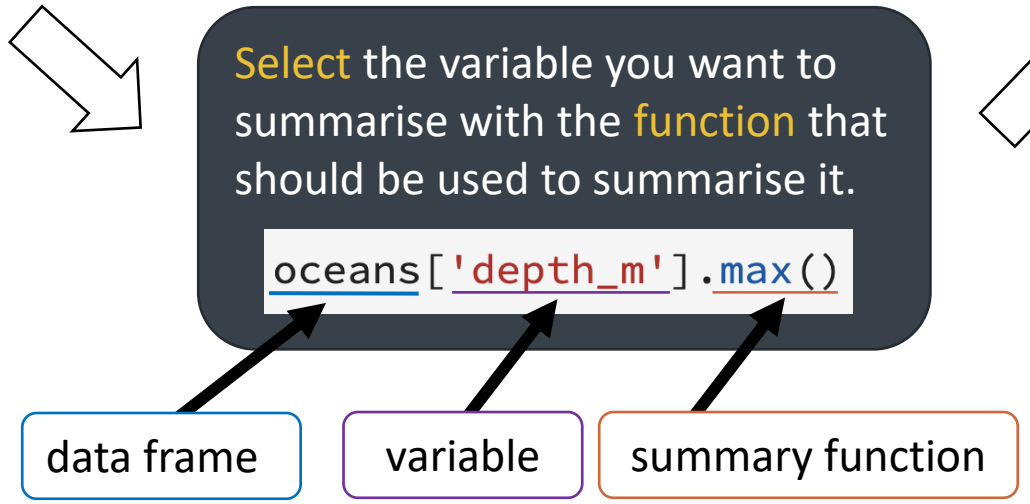


Summarise a Numeric Variable in Python

As an example, we will find the **maximum** value of the **depth_m** variable in the **oceans** data frame.

oceans

ocean	depth_m
Pacific	3970
Atlantic	3646
Indian	3741
Southern	3270
Arctic	1205



3970

max() returns a number.

min(), **mean()** and **count()** do the same.

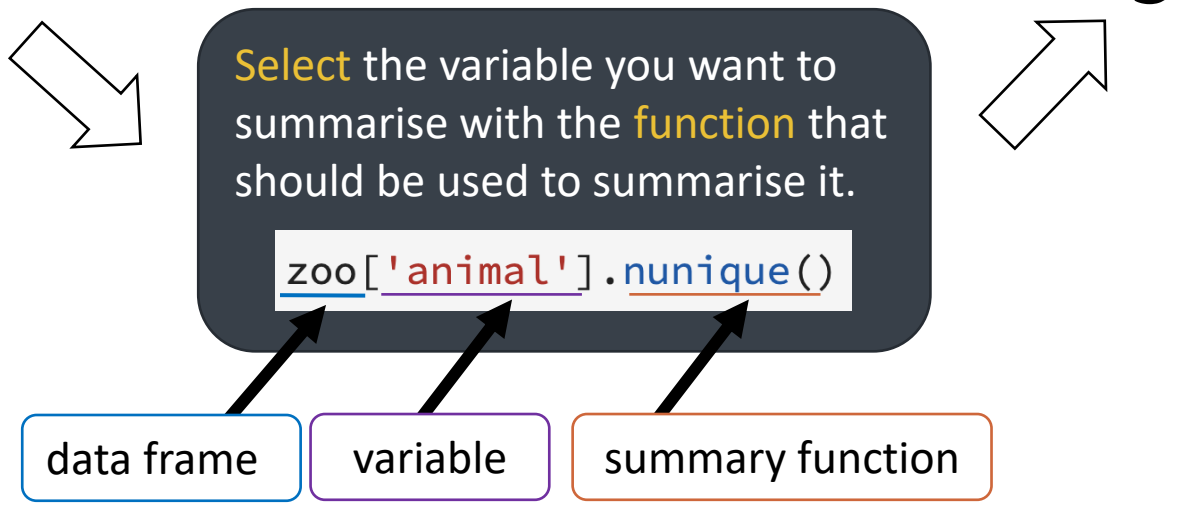
min(), **mean()** and **count()** can all be used in the same way as shown here.

Summarise a Text Variable in Python

As an example, we will find the **number of unique values** in the **animal** variable in the **zoo** data frame.

zoo

animal	name	age_yrs
lion	Lisa	10
tiger	Ted	2
elephant	Elise	15
penguin	Peter	3
parrot	Peizhi	5
lion	Laasya	1
penguin	Priti	49



5

nunique() returns a number.

count() does the same.

count() can be used in the same way as shown here.

Next steps

Complete the **Summarise a single variable** section of the ‘Summarising Datasets Part 1' Jupyter Notebook.

Summarise Multiple Variables in Python

You can also perform a **single summary calculation** for **more than one** variable at a time.

When selecting the variables to perform a summary calculation for, you can either select them by:

- their **names**, or
- their **data type**

Selecting Variables By Name

Reminder:

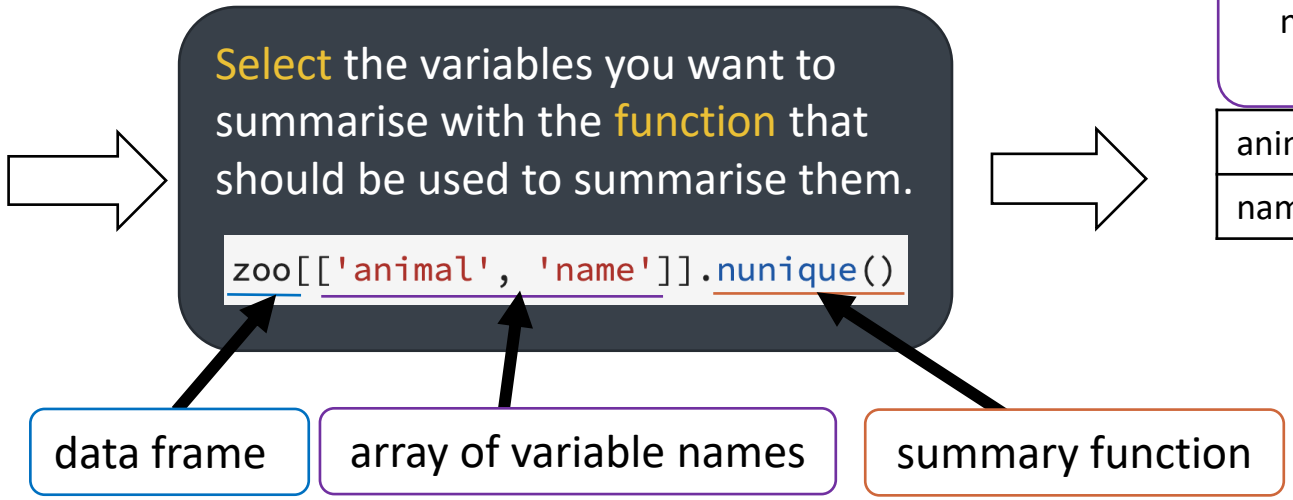
To select a **single** variable: `zoo['animal']`

To select **multiple** variables: `zoo[['animal', 'name']]` (notice the double brackets [[ ]])

Performing a summary calculation works the same way for both.

zoo

animal	name	age_yrs
lion	Lisa	10
tiger	Ted	2
elephant	Elise	15
penguin	Peter	3
parrot	Peizhi	5
lion	Laasya	1
penguin	Priti	49



summary

variable name	number of unique values
animal	5
name	7

Selecting Variables By Data Type

To select all variables that have a **specific data type** and perform a summary calculation for each of these variables, use the pandas function **select_dtypes()**. **select_dtypes()** stands for '**select d**ata **types'.**

To find the **number of unique values** for all of the **text variables** in the zoo data frame...

zoo

animal	name	age_yrs
lion	Lisa	10
tiger	Ted	2
elephant	Elise	15
penguin	Peter	3
parrot	Peizhi	5
lion	Laasya	1
penguin	Priti	49

Select the variables you want to summarise by data type with the function that should be used to summarise them.

```
zoo.select_dtypes(include='object').nunique()
```

data frame

select only text variables

summary function

summary

variable name	number of unique values
animal	5
name	7

The pandas **object** data type is the same as **string/text**.

To select **numeric** variables, use **include='number'.**

Next steps

Complete the **Summarise multiple variables** section of the ‘Summarising Datasets Part 1' Jupyter Notebook.

Additional information

Some useful resources:

- Official pandas documentation

Learning checklist

I can *describe* how to summarise variables.

I can *summarise* a complete data frame in Python.

I can *summarise* a single variable in Python.

I can *summarise* multiple variables in Python.