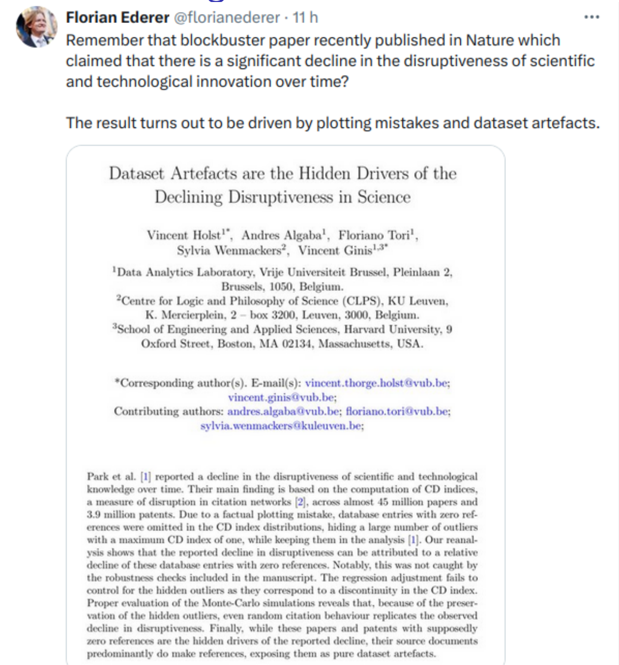
Introduction to Python

Programa de Doctorat en Economia, Universitat de Barcelona

David Moriña

Before we start: BREAKING NEWS.

Free software dangers



Florian Ederer @florianederer · 11 h

Remember that blockbuster paper recently published in Nature which claimed that there is a significant decline in the disruptiveness of scientific and technological innovation over time?

The result turns out to be driven by plotting mistakes and dataset artefacts.

Dataset Artefacts are the Hidden Drivers of the Declining Disruptiveness in Science

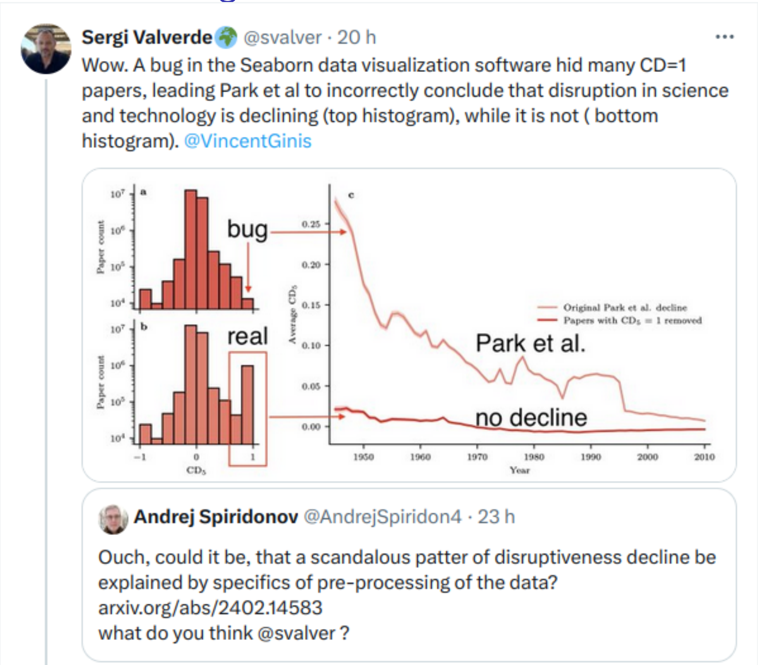
Vincent Holst[1*], Andres Algaba[1], Floriano Tori[1], Sylvia Wenmackers[2], Vincent Ginis[1,3*]

[1]Data Analytics Laboratory, Vrije Universiteit Brussel, Pleinlaan 2, Brussels, 1050, Belgium.
[2]Centre for Logic and Philosophy of Science (CLPS), KU Leuven, K. Mercierplein, 2 – box 3200, Leuven, 3000, Belgium.
[3]School of Engineering and Applied Sciences, Harvard University, 9 Oxford Street, Boston, MA 02134, Massachusetts, USA.

*Corresponding author(s). E-mail(s): vincent.thorge.holst@vub.be; vincent.ginis@vub.be;
Contributing authors: andres.algaba@vub.be; floriano.tori@vub.be; sylvia.wenmackers@kuleuven.be;

Park et al. [1] reported a decline in the disruptiveness of scientific and technological knowledge over time. Their main finding is based on the computation of CD indices, a measure of disruption in citation networks [2], across almost 45 million papers and 3.9 million patents. Due to a factual plotting mistake, database entries with zero references were omitted in the CD index distributions, hiding a large number of outliers with a maximum CD index of one, while keeping them in the analysis [1]. Our reanalysis shows that the reported decline in disruptiveness can be attributed to a relative decline of these database entries with zero references. Notably, this was not caught by the robustness checks included in the manuscript. The regression adjustment fails to control for the hidden outliers as they correspond to a discontinuity in the CD index. Proper evaluation of the Monte-Carlo simulations reveals that, because of the preservation of the hidden outliers, even random citation behaviour replicates the observed decline in disruptiveness. Finally, while these papers and patents with supposedly zero references are the hidden drivers of the reported decline, their source documents predominantly do make references, exposing them as pure dataset artefacts.

Free software dangers



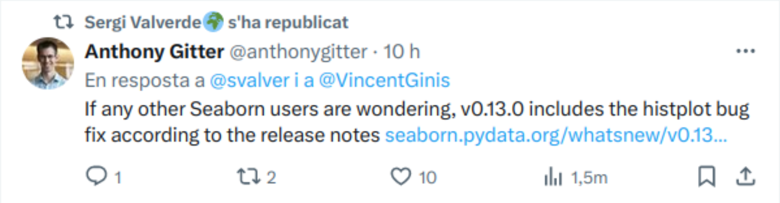
Sergi Valverde @svalver · 20 h ···

Wow. A bug in the Seaborn data visualization software hid many CD=1 papers, leading Park et al to incorrectly conclude that disruption in science and technology is declining (top histogram), while it is not (bottom histogram). @VincentGinis

Andrej Spiridonov @AndrejSpiridon4 · 23 h

Ouch, could it be, that a scandalous patter of disruptiveness decline be explained by specifics of pre-processing of the data?
arxiv.org/abs/2402.14583
what do you think @svalver ?

Free software dangers



Sergi Valverde🌍 s'ha republicat

Anthony Gitter @anthonygitter · 10 h

En resposta a @svalver i a @VincentGinis

If any other Seaborn users are wondering, v0.13.0 includes the histplot bug fix according to the release notes seaborn.pydata.org/whatsnew/v0.13...

Free software dangers

```
pip show seaborn
```

```
Name: seaborn
Version: 0.13.2
Summary: Statistical data visualization
Home-page:
Author:
Author-email: Michael Waskom <mwaskom@gmail.com>
License:
Location: /home/dmorina/.local/lib/python3.10/site-packages
Requires: matplotlib, numpy, pandas
Required-by: segregation, splot, spvcm
Note: you may need to restart the kernel to use updated pac
```

Python progamming language

Integrated Development Environments

Python can be used in *batch* mode, but there are a number of excellent integrated development environments (IDEs) for Python, among the most used:

- JupyterLab / Jupyter Notebook
- Spyder
- PyCharm
- RStudio

Docker file for the course

- Download Docker Desktop from https://www.docker.com/products/docker-desktop/
- Get everything you need to follow the course as a Docker image

```
docker pull dmorinya/python-econ-ub:2024
```

- Run the image

```
docker run --rm -ti -p 8888:8888 -v /home/dmorinya/
 ↪  dmorinya/python-econ-ub:2024
```

JupyterLab installation (Windows)

- Download latest python release (3.12.1 at the time of writing this document) from Microsoft Store
- Install it the usual way
- Run *system symbol*
  - Install the python package manager *pip*:

```
python -m ensurepip -upgrade
```

- Install JupyterLab with pip:

```
pip install jupyterlab
```

- Set a new System environment variable to the path (adapt it to fit your particular installation):

```
C:\Users\dmori\AppData\Local\Packages\PythonSoftwareFoundat
```

JupyterLab installation (Windows)

- Run *jupyterlab* from the system symbol:

```
jupyter lab
```

A new instance of the default browser will launch. If it returns an error, go to the system symbol and copy and paste the address that looks similar to:

```
http://localhost:8888/lab?token=e1c50f3b897c86536a6ecd7b7d
```

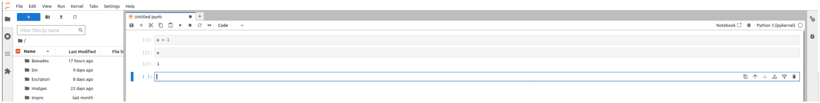
JupyterLab installation (MacOS)

- Install homebrew https://brew.sh/
- Install python and JupyterLab from *brew* using the Terminal app:

```
brew install python
python -m ensurepip --upgrade
brew install jupyterlab
```

JupyterLab

A Jupyter notebook is divided into individual, vertically arranged cells, which can be executed separately:



Name	Last Modified	File Si
Baixades	17 hours ago	
bin	9 days ago	
Escriptori	8 days ago	
Imatges	22 days ago	
Insync	last month	

```
[1]: a = 1
[2]: a
[2]: 1
[ ]:
```

Figure 1: JupyterLab screenshot

Essential concepts

Python evolution

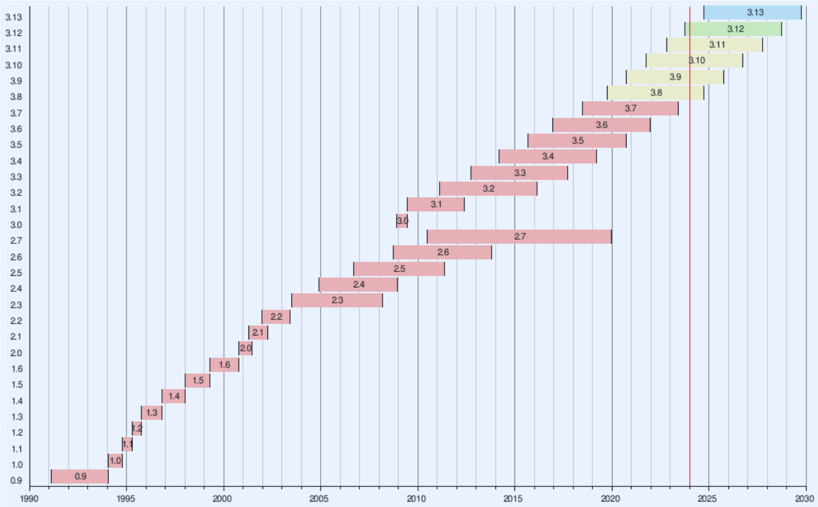


Figure 2: Python timeline

Getting help

Information on Python objects can be obtained quickly in an interactive environment:

```python
help(len)
```

```
Help on built-in function len in module builtins:

len(obj, /)
    Return the number of items in a container.
```

Basic programming

Programs can be implemented very quickly – this is a pretty minimal example. You can write this command to a text file of your choice and run it directly on your system:

```
print("Hello there!")
```

```
Hello there!
```

- Only one function *print()* (shown here as a keyword),
- Function displays argument (a string) on screen,
- Arguments are passed to the function in parentheses,
- A string must be wrapped in ” ” or ’ ’,
- No semicolon at the end.

Basic programming

Types of objects

- Numbers (integers, floating-point numbers, and complex numbers)
- Booleans
- The “null” type (*NA* in some other languages)
- Strings
- Lists

Basic programming

Operations

- Sum: x + y
- Difference: x - y
- Product: x * y
- Quotient: x / y
- Remainder: x % y
- Power: x ** y
- Absolute value: abs(x)

Basic programming

Comparison

- x == y
- x != y
- x > y / x >= y
- x < y / x <= y
- And: &
- Or: |

Basic programming

Importing additional packages

```
import pandas as pd
```

Importing only a function from a package

```
from datetime import datetime
```

Basic programming

Operations (extension)

```python
import math

math.factorial(6)
```

```
720
```

Basic programming

Operations (extension)

```
import math

math.log(1)
```

```
0.0
```

Basic programming

Operations (extension)

```python
import math

math.exp(1)
```

```
2.718281828459045
```

Basic programming

Comments in python (jupyterLab)

```
# This is
# a multiline comment
```

Basic programming

Declaring new variables

```
a = 2
vec = [] * 1000 # Array of size 1000
```

Basic programming

Changing the working directory

```python
import os

os.getcwd()
```

```
'/home/dmorina/Insync/dmorina@ub.edu/OneDrive Biz/Docència/
```

```python
import os

os.chdir('/home/dmorina/')
os.getcwd()
```

```
'/home/dmorina'
```

Basic programming

Defining (and using) new functions:

```
def newFunction(x, y):
  return x % y

newFunction(3, 2)
```

```
1
```

Basic programming

Defining (and using) new functions:

```
def newFunction2(x):
  if x > 5:
    return x+5
  elif x == 5:
    return x+10
  else:
    return x+100
```

```
newFunction2(3)
```

```
103
```

```
newFunction2(5)
```

```
15
```

```
newFunction2(20)
```

Basic programming

Defining (and using) new functions:

```
def newFunction3():
  for x in range(6):
    if x == 3: continue
    print(x)
  else:
    print("Finally finished!")
```

```
newFunction3()
```

```
0
1
2
4
5
Finally finished!
```

Basic programming

Defining (and using) new functions:

```
def newFunction4():
  for x in range(6):
    if x == 3: break
    print(x)
  else:
    print("Finally finished!")
```

```
newFunction4()
```

```
0
1
2
```

Basic programming

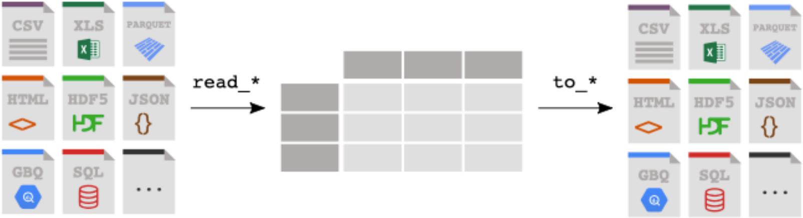


Figure 3: pandas import and export

Basic programming

Reading and basic work with data (pandas!)

```python
import pandas as pd
import os

os.chdir("/home/dmorina/Insync/dmorina@ub.edu/OneDrive Biz/Docència/UB/2023-2024/PyEcon/1. Introduction to Python/examples/")
newData = pd.read_csv("titanic.csv")
```

Basic programming

Reading and basic work with data (pandas!)

```
newData.head(2)
```

	PassengerId	Survived	Pclass	Name
0	1	0	3	Braund, Mr. Owen Harris
1	2	1	1	Cumings, Mrs. John Bradley (Flore

Basic programming

Reading and basic work with data (pandas!)

```python
newData.tail(5)
```

	PassengerId	Survived	Pclass	Name
886	887	0	2	Montvila, Rev. Juozas
887	888	1	1	Graham, Miss. Margaret Edith
888	889	0	3	Johnston, Miss. Catherine Helen
889	890	1	1	Behr, Mr. Karl Howell
890	891	0	3	Dooley, Mr. Patrick

Basic programming

Reading and basic work with data (pandas!)

```
newData.shape
```

```
(891, 12)
```

```
len(newData)
```

```
891
```

```
newData.size
```

```
10692
```

```
newData.ndim
```

```
2
```

```
newData.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

Basic programming

Reading and basic work with data (pandas!)

```
newData.count()
```

```
PassengerId    891
Survived       891
Pclass         891
Name           891
Sex            891
Age            714
SibSp          891
Parch          891
Ticket         891
Fare           891
Cabin          204
Embarked       889
dtype: int64
```

```
newData['Age'].count()
```

Basic programming

Reading and basic work with data (pandas!)

```
newData.describe()
```

	PassengerId	Survived	Pclass	Age	SibSp
count	891.000000	891.000000	891.000000	714.000000	891.0000
mean	446.000000	0.383838	2.308642	29.699118	0.523008
std	257.353842	0.486592	0.836071	14.526497	1.102743
min	1.000000	0.000000	1.000000	0.420000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000
50%	446.000000	0.000000	3.000000	28.000000	0.000000
75%	668.500000	1.000000	3.000000	38.000000	1.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000

Basic programming

Reading and basic work with data (pandas!)

```
newData.groupby(["Sex", "Pclass"])["Fare"].describe()
```

		count	mean	std	min	25%
Sex	Pclass					
	1	94.0	106.125798	74.259988	25.9292	57.24480
female	2	76.0	21.970121	10.891796	10.5000	13.00000
	3	144.0	16.118810	11.690314	6.7500	7.85420
	1	122.0	67.226127	77.548021	0.0000	27.72810
male	2	108.0	19.741782	14.922235	0.0000	12.33125
	3	347.0	12.661633	11.681696	0.0000	7.75000

Basic programming

Selecting rows

```
newData.iloc[:3]
```

	PassengerId	Survived	Pclass	Name
0	1	0	3	Braund, Mr. Owen Harris
1	2	1	1	Cumings, Mrs. John Bradley (Flore
2	3	1	3	Heikkinen, Miss. Laina

Basic programming

Selecting rows (conditionally)

```
newData.query('Age>40 & Sex=="female"').head(2)
```

	PassengerId	Survived	Pclass	Name
11	12	1	1	Bonnell, Miss. Elizabeth
15	16	1	2	Hewlett, Mrs. (Mary D Kingcome

Basic programming

Selecting rows (conditionally)

```
newData[(newData.Age > 40) & (newData.Sex ==
 ↪  "female")].head(2)
```

	PassengerId	Survived	Pclass	Name
11	12	1	1	Bonnell, Miss. Elizabeth
15	16	1	2	Hewlett, Mrs. (Mary D Kingcome

Basic programming

Selecting rows (randomly)

```
newData.sample(n=2)
```

	PassengerId	Survived	Pclass	Name	Se
121	122	0	3	Moore, Mr. Leonard Charles	m
30	31	0	1	Uruchurtu, Don. Manuel E	m

```
newData.sample(frac=0.001)
```

	PassengerId	Survived	Pclass	Name	Sex
793	794	0	1	Hoyt, Mr. William Fisher	male

Basic programming

Selecting columns

```
newData[['Age', 'Sex']].head(2)
```

	Age	Sex
0	22.0	male
1	38.0	female

Basic programming

Selecting columns

```
newData.loc[:, 'Age':'Ticket'].head(2)
```

	Age	SibSp	Parch	Ticket
0	22.0	1	0	A/5 21171
1	38.0	1	0	PC 17599

Basic programming

Selecting columns

```
newData[['Age', 'Sex']].head(2)
```

	Age	Sex
0	22.0	male
1	38.0	female

Basic programming

Rename columns

```
newData.rename(columns={'Age': 'age'}).head(3)
```

	PassengerId	Survived	Pclass	Name
0	1	0	3	Braund, Mr. Owen Harris
1	2	1	1	Cumings, Mrs. John Bradley (Flore
2	3	1	3	Heikkinen, Miss. Laina

Basic programming

Drop columns

```
newData.drop(['Age', 'Sex'], axis=1).head(2)
```

	PassengerId	Survived	Pclass	Name
0	1	0	3	Braund, Mr. Owen Harris
1	2	1	1	Cumings, Mrs. John Bradley (Flore

Basic programming

Drop duplicates

```
newData.drop_duplicates().head(3)
```

	PassengerId	Survived	Pclass	Name
0	1	0	3	Braund, Mr. Owen Harris
1	2	1	1	Cumings, Mrs. John Bradley (Flore
2	3	1	3	Heikkinen, Miss. Laina

Basic programming

Create a new column

```
newData["AgeGroup"] = pd.cut(newData.Age, range(0,
 ↪  105, 10), right=False)
newData[['Age', 'AgeGroup']].head(8)
```

	Age	AgeGroup
0	22.0	[20.0, 30.0)
1	38.0	[30.0, 40.0)
2	26.0	[20.0, 30.0)
3	35.0	[30.0, 40.0)
4	35.0	[30.0, 40.0)
5	NaN	NaN
6	54.0	[50.0, 60.0)
7	2.0	[0.0, 10.0)

Basic programming

Join DataFrames vertically

```
less50 = newData[newData.Age <= 50]
over50 = newData[newData.Age > 50]
total = pd.concat([less50, over50])
total.head(2)
```

	PassengerId	Survived	Pclass	Name
0	1	0	3	Braund, Mr. Owen Harris
1	2	1	1	Cumings, Mrs. John Bradley (Flore

Basic programming

Join DataFrames horizontally

```
df1 = pd.DataFrame({
    'A': [1,2,3,4,5],
    'B': [1,2,3,4,5]
})

df2 = pd.DataFrame({
    'C': [1,2,3,4,5],
    'D': [1,2,3,4,5]
})

df_concat = pd.concat([df1, df2], axis=1)
df_concat.head(2)
```

	A	B	C	D
0	1	1	1	1
1	2	2	2	2

Basic programming

Merge DataFrames

```
df1 = pd.DataFrame({
    'id': [1,2,3,4,5],
    'col1': [1,2,3,4,5]
})

df2 = pd.DataFrame({
    'id': [1,2,3,4,5],
    'col2': [6,7,8,9,10]
})

df_merge = df1.merge(df2, on='id')
df_merge.head(2)
```

	id	col1	col2
0	1	1	6
1	2	2	7

Basic programming

Exporting data (pandas!)

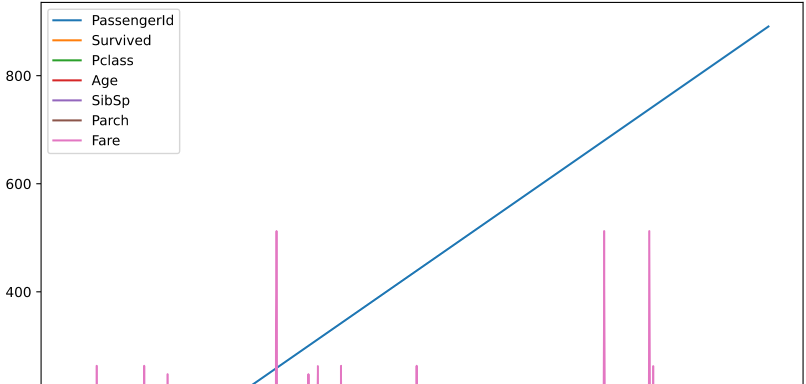
```
import pandas as pd
newData.to_excel("titanic.xlsx",
 ↪  sheet_name="passengers", index=False)
```

Basic programming

Generating basic graphs with pandas

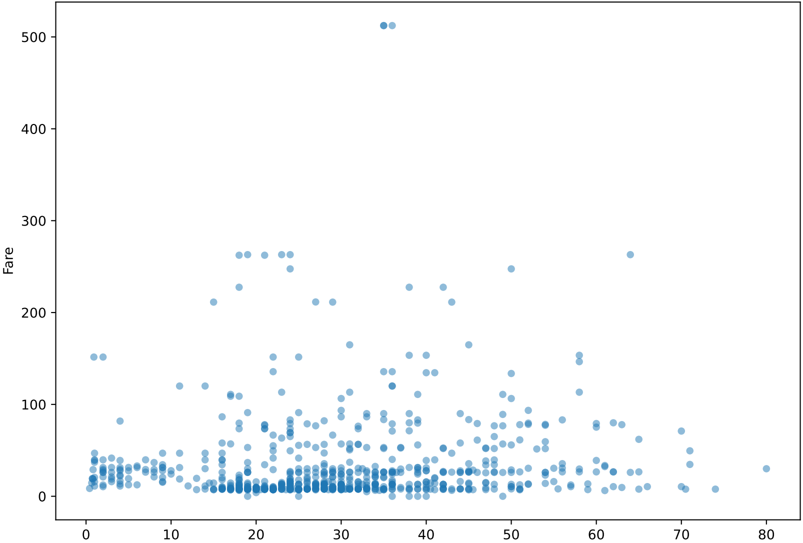
```
import matplotlib.pyplot as plt

newData.plot()
plt.show()
```



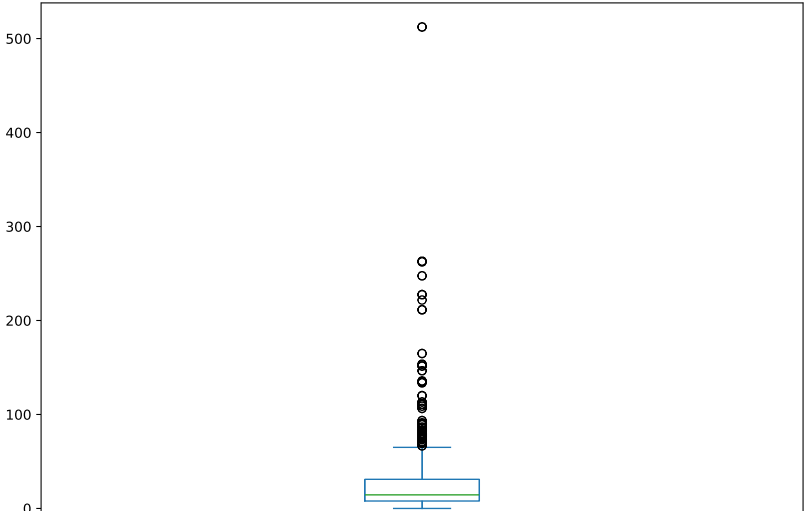
Basic programming

```
newData.plot.scatter(x="Age", y="Fare", alpha=0.5)
plt.show()
```



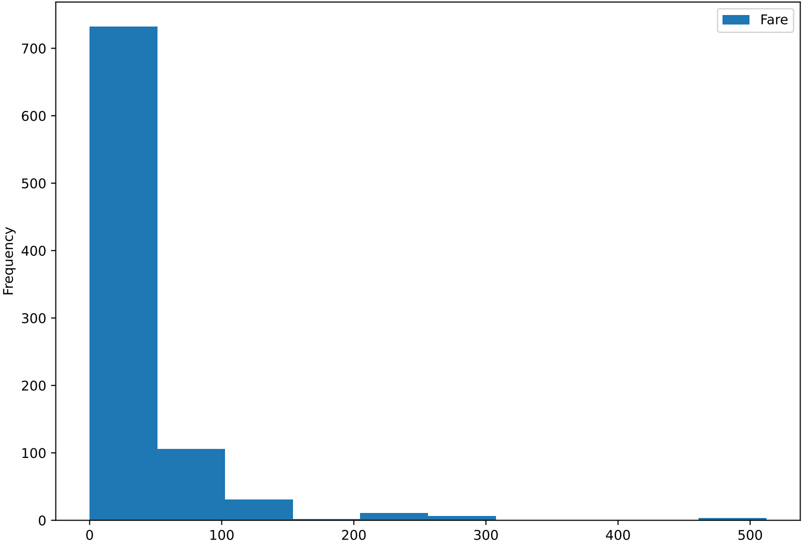
Basic programming

```
newData.plot.box(y="Fare")
plt.show()
```



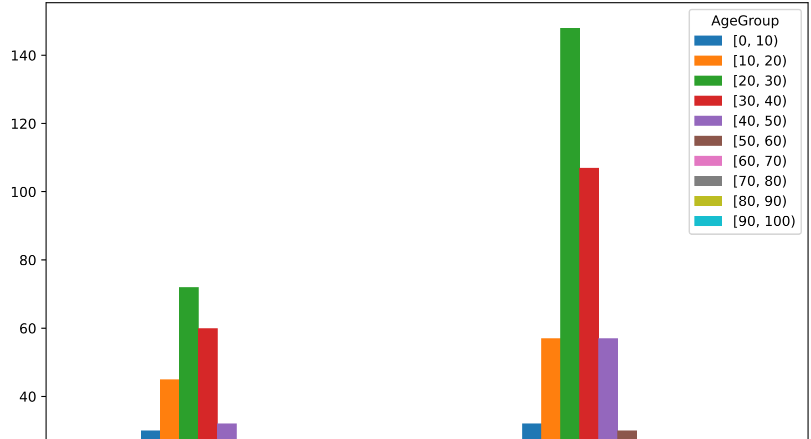
Basic programming

```
newData.plot.hist(y="Fare")
plt.show()
```



Basic programming

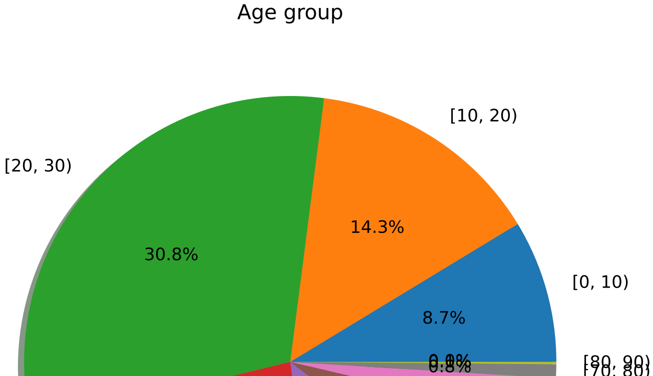
```
newData2 = newData.groupby(['Sex','AgeGroup']).size()
newData2 = newData2.unstack()
newData2.plot.bar()
plt.show()
```



Basic programming

```
newData3 = newData.groupby(['AgeGroup']).size()
newData3.plot.pie(y="AgeGroup", title="Age group",
 ↪  legend=False,
                  autopct='%1.1f%%',
                  shadow=True, startangle=0)
```

```
<Axes: title={'center': 'Age group'}>
```



Basic programming

Remove objects

```
del [[newData, newData2, newData3]]
```

Basic programming

More information on pandas: https://pandas.pydata.org