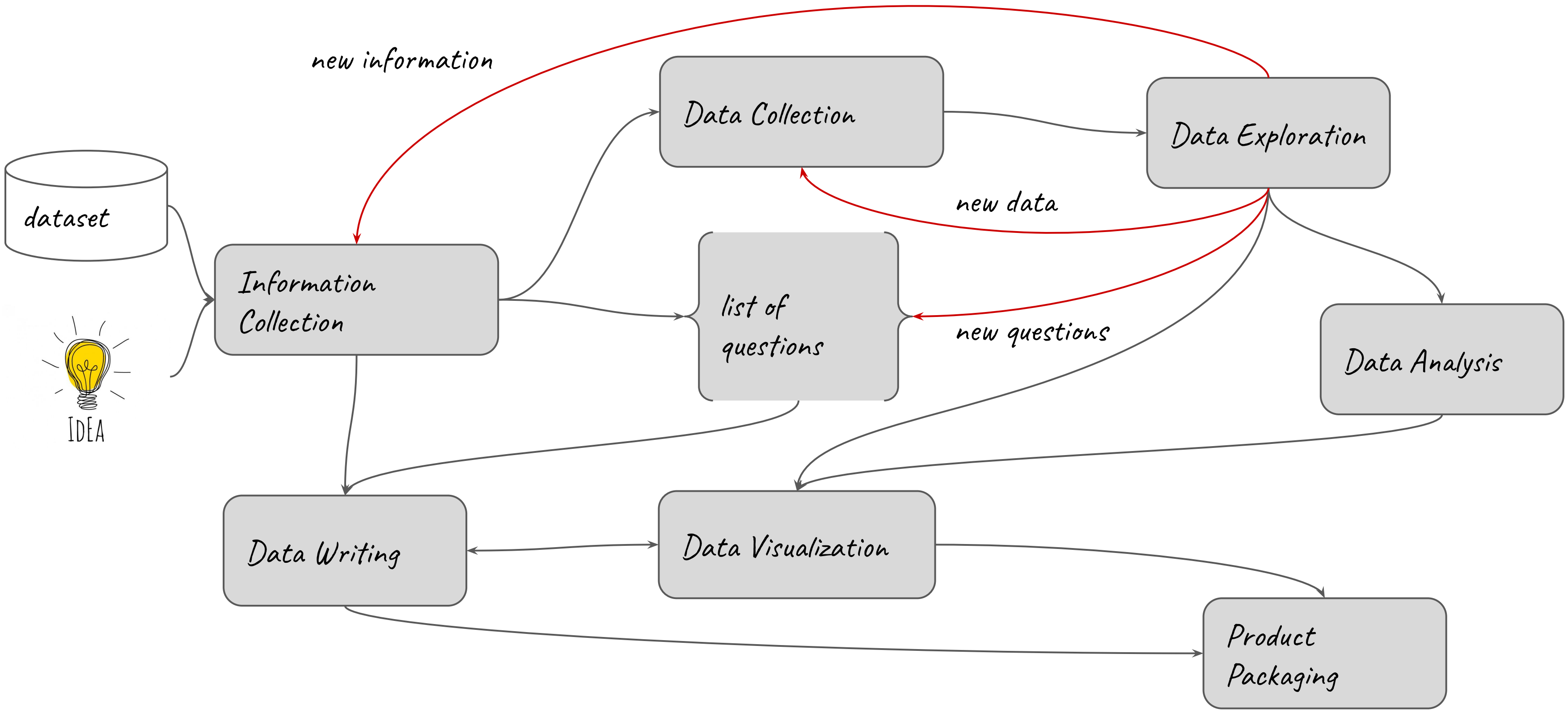
Data Journalism

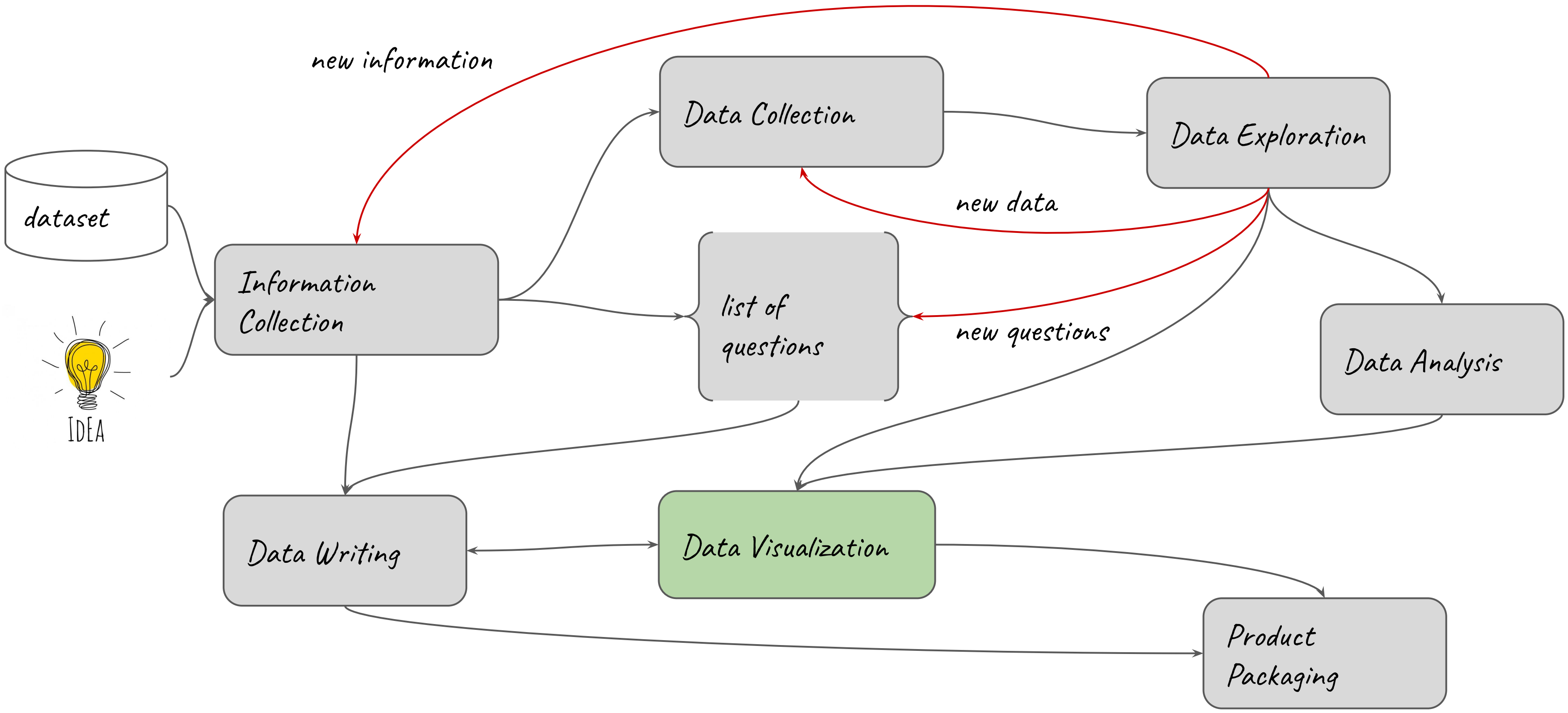
Data Visualization
Altair

Angelica Lo Duca
angelica.loduca@iit.cnr.it

Data Journalism Workflow



Data Journalism Workflow



Altair

The Vega-Altair library (Altair, for short) is a declarative Python library for statistical visualization based on the Vega and Vega-Lite visualization grammars.

Declarative libraries specify what we want to see in a chart. We can specify the data and the type of visualization we want, and the library creates the visualization for us automatically.

Imperative libraries focus on building a visualization manually, for example specifying the desired axis, size, legend, and labels (e.g. Matplotlib)

Altair parameters

Marks: define the type of chart we want to build (e.g. bar chart, line chart, …)

Encodings: mapping of visual properties (**channels**) to data columns in the DataFrame

Visual properties include axes, colors, size, …

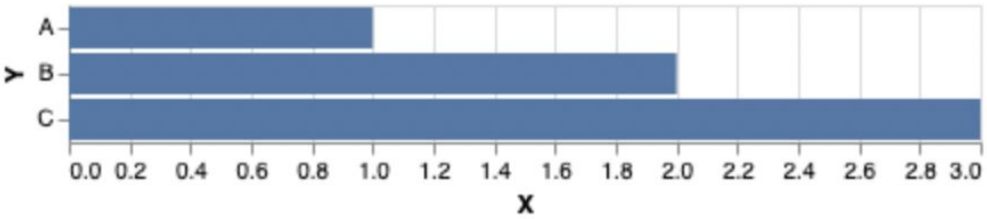
```
pip install altair
```

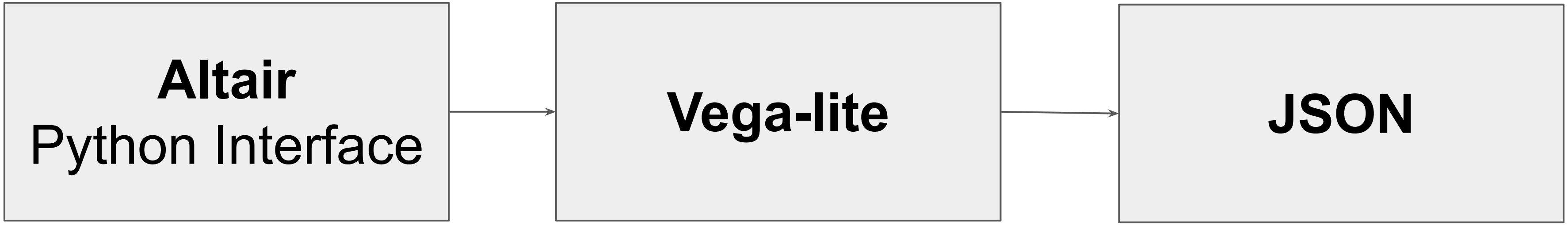
A first example

```
import pandas as pd
import altair as alt

df = pd.DataFrame({
    'X' : [3,2,4],
    'Y' : ['M','N','O']
})


chart = alt.Chart(df).mark_bar(
).encode(
     x = 'X:Q',
     y = 'Y:N'
)
```





Altair
Python Interface

Vega-lite

JSON

Vega and Vega-lite

Vega and Vega-lite provide a declarative language for creating interactive visualizations.

```
{
  "$schema": "https://vega.github.io/schema/vega/v5.json",
  "description": "An example of vega specification.",
  "width": 600,
  "height": 400,

  "data": [],
  "scales": [],
  "axes": [],
  "marks": [],
  "signals": []
}
```

Vega-lite

```
{ "config": {
    "mark": "bar",
    "encoding": {
        "x": {
            "field": "X",
            "type": "quantitative"
        },
        "y": {
            "field": "Y",
            "type": "nominal"
        }
    },
    "$schema":
"https://vega.github.io/schema/vega-lite/v4.17.0.json",
    "datasets": {...}}
```

Marks

Method	Description
`mark_arc()`	A pie chart
`mark_area()`	A filled area chart
`mark_bar()`	A bar chart
`mark_circle()`	A scatter plot with filled points
`mark_geoshape()`	A geographic shape
`mark_line()`	A line chart
`mark_point()`	A scatter plot with configurable points
`mark_rect()`	A filled rectangle for heatmaps
`mark_rule()`	A vertical or horizontal line spanning the axis
`mark_text()`	A text

Channels

Position Channels

Channel	Description
`x`, `y`	Horizontal and vertical positions
`longitude`, `latitude`	Geographical coordinates
`xError`, `xError`	Error values for x and y
`x2`,`y2`, `longitude2`, `latitude2`	Second positions for ranges
`theta`, `theta2`	Start and end arc angles

Mark Property Channels

Channel	Description
`angle`, `radius`	The angle and the radius of the mark
`color`, `fill`	The color and the fill of the mark
`opacity`, `fillOpacity`	The opacity and the fill opacity of the mark
`shape`, `size`	The shape and the size of the mark
`stroke`, `strokeDash`, `strokeOpacity`, `strokeWidth`	Stroke properties of the mark

Text and Tooltip Channels

Channel	Description
`text`	The text of the mark
`tooltip`	The tooltip of the mark