wbs
WARWICK BUSINESS SCHOOL
THE UNIVERSITY OF WARWICK

For the Change Makers

Advanced Programming for Data Science

Week 3: Data Visualization
Information Systems and Management
Warwick Business School

Exercise 1: London Bike Hire

- Santander Cycles is a public bicycle hire scheme in London, Swansea, Milton Keynes and Brunel University in the United Kingdom. The scheme's bicycles used to be popularly known as Boris Bikes, after then-Mayor of London, Boris Johnson. (Wikipedia)
- Please connect Tableau with your Google BigQuery account, add ' bigquery-public-data' project, and find the dataset "london_bicycles". Try to identify potential improvement through data visualization. You may need to write SQL queries to generate the analyzable dataset.
- E.g. stations need more docks, or stations need more bikes.

Tableau with Python

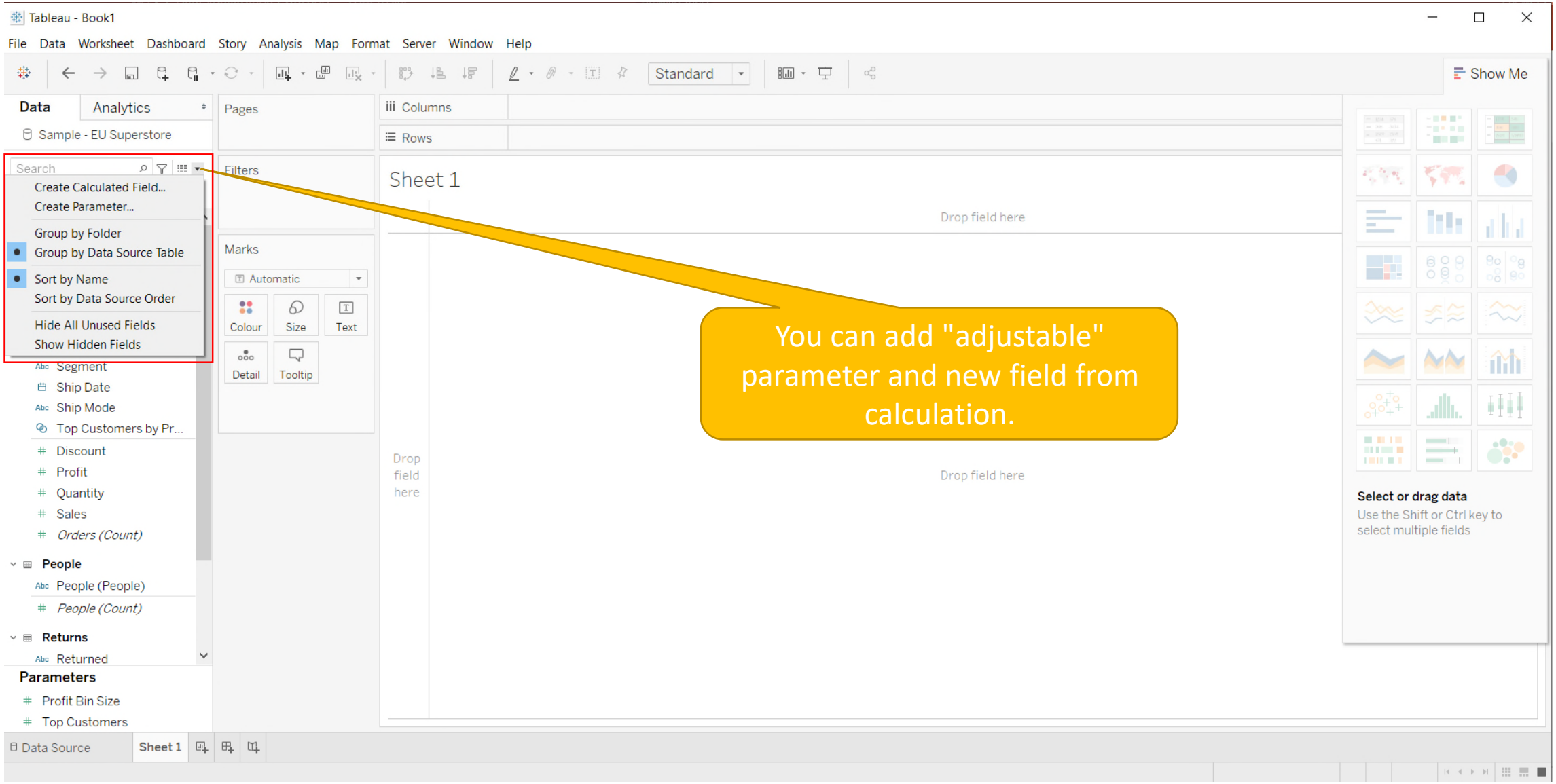
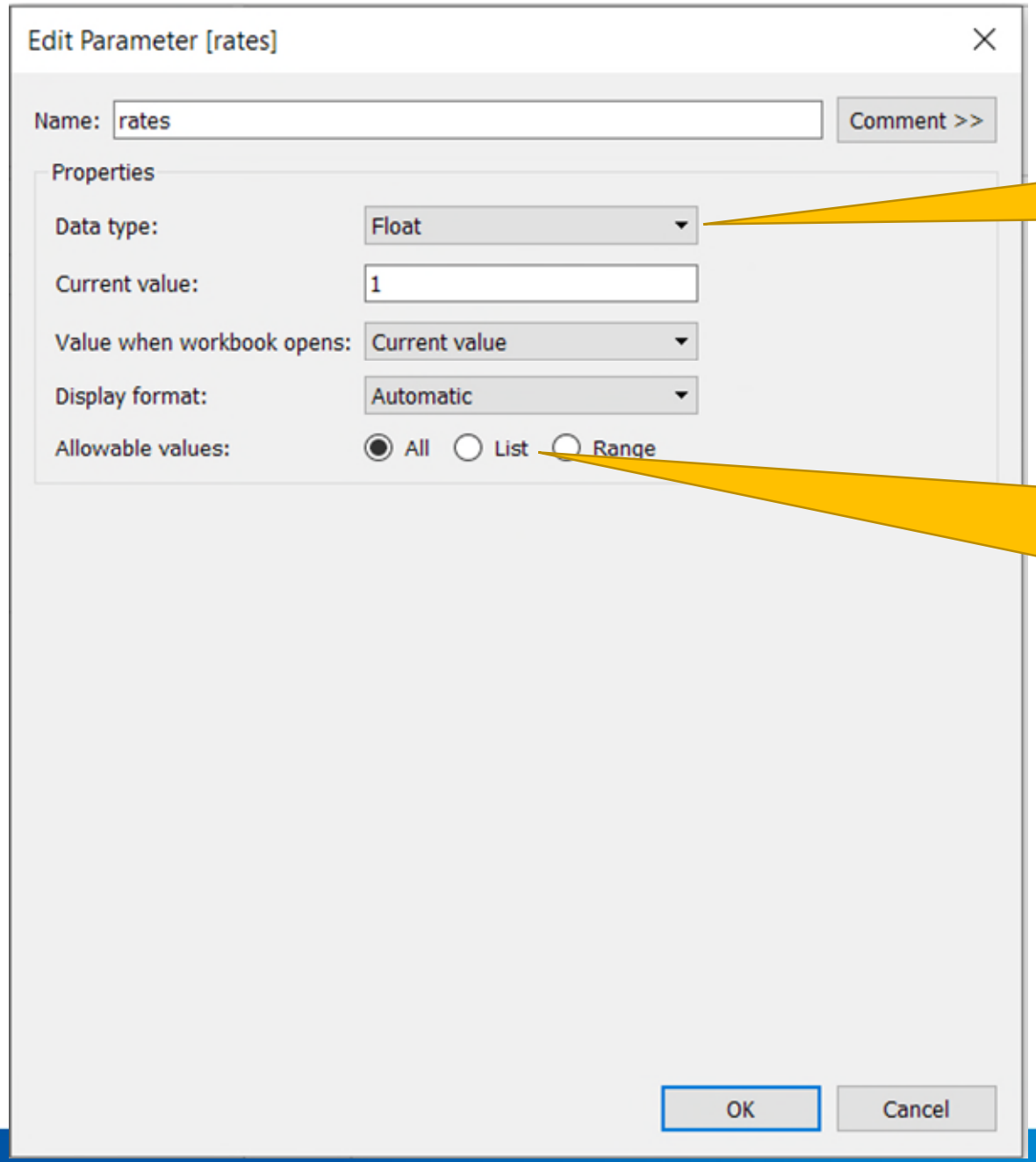


Tableau - Book1
File Data Worksheet Dashboard Story Analysis Map Format Server Window Help
Standard
Show Me
Data
Analytics
Sample - EU Superstore
Search
Create Calculated Field...
Create Parameter...
Group by Folder
Group by Data Source Table
Sort by Name
Sort by Data Source Order
Hide All Unused Fields
Show Hidden Fields
Segment
Ship Date
Ship Mode
Top Customers by Pr...
Discount
Profit
Quantity
Sales
Orders (Count)
People
People (People)
People (Count)
Returns
Returned
Parameters
Profit Bin Size
Top Customers
Pages
Filters
Marks
Automatic
Colour
Size
Text
Detail
Tooltip
Columns
Rows
Sheet 1
Drop field here
Drop field here
Drop field here
You can add "adjustable" parameter and new field from calculation.
Select or drag data
Use the Shift or Ctrl key to select multiple fields
Data Source
Sheet 1



You may change the data type of the parameter

You can also how the values of parameter can be adjusted.
All: manually specify the value
List: provide a list of possible values.
Range: provide a way to generate a range of possible values.

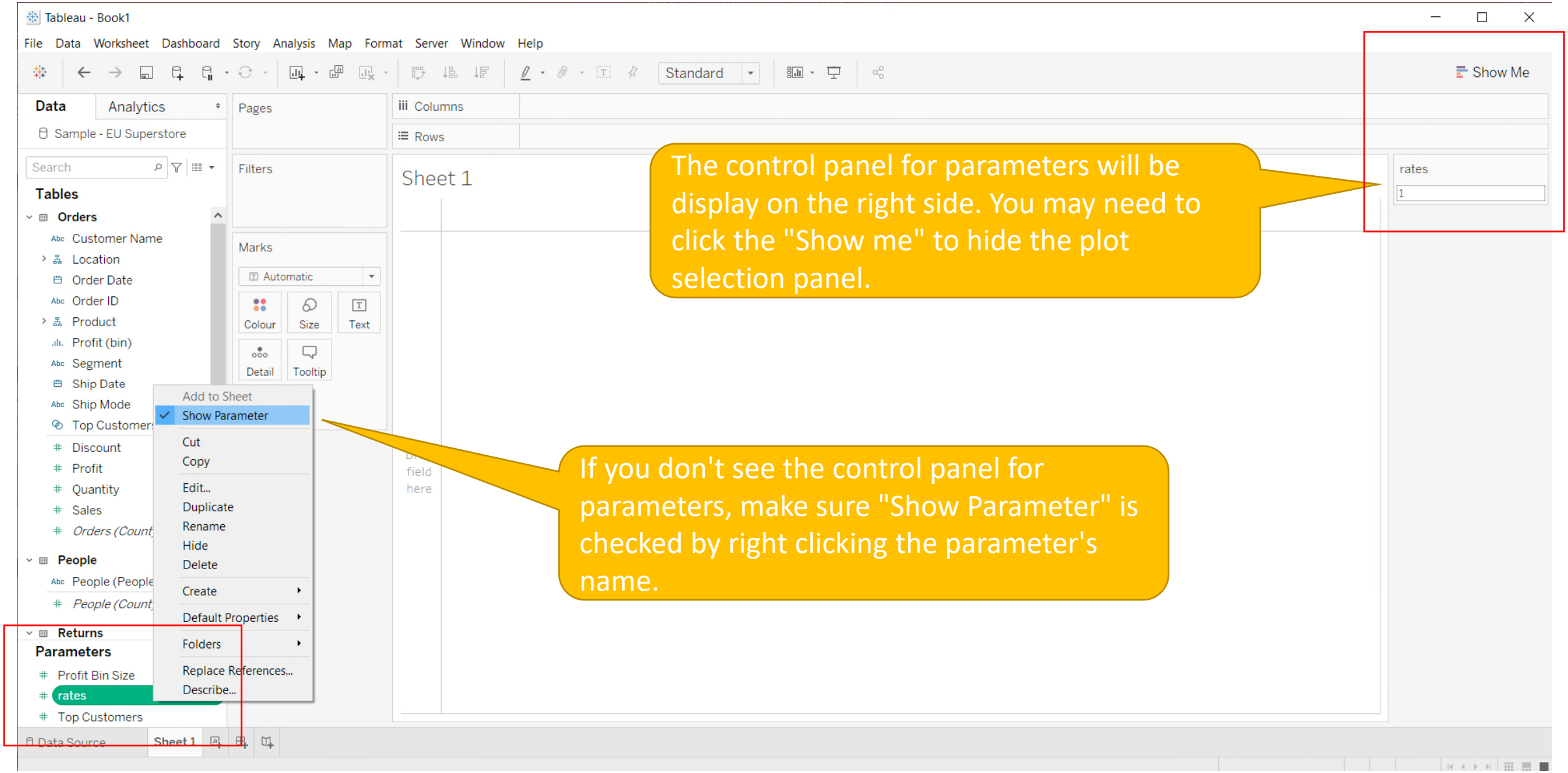


Tableau - Book1
File Data Worksheet Dashboard Story Analysis Map Format Server Window Help
Standard
Show Me
Data
Analytics
Sample - EU Superstore
Search
Tables
Orders
Customer Name
Location
Order Date
Order ID
Product
Profit (bin)
Segment
Ship Date
Ship Mode
Top Customers
Discount
Profit
Quantity
Sales
Orders (Count)
People
People (People)
People (Count)
Returns
Parameters
Profit Bin Size
rates
Top Customers
Pages
Filters
Marks
Automatic
Colour
Size
Text
Detail
Tooltip
Columns
Rows
Sheet 1
Drop field here
rates
1
Add to Sheet
Show Parameter
Cut
Copy
Edit...
Duplicate
Rename
Hide
Delete
Create
Default Properties
Folders
Replace References...
Describe...
Data Source
Sheet 1
The control panel for parameters will be display on the right side. You may need to click the "Show me" to hide the plot selection panel.
If you don't see the control panel for parameters, make sure "Show Parameter" is checked by right clicking the parameter's name.

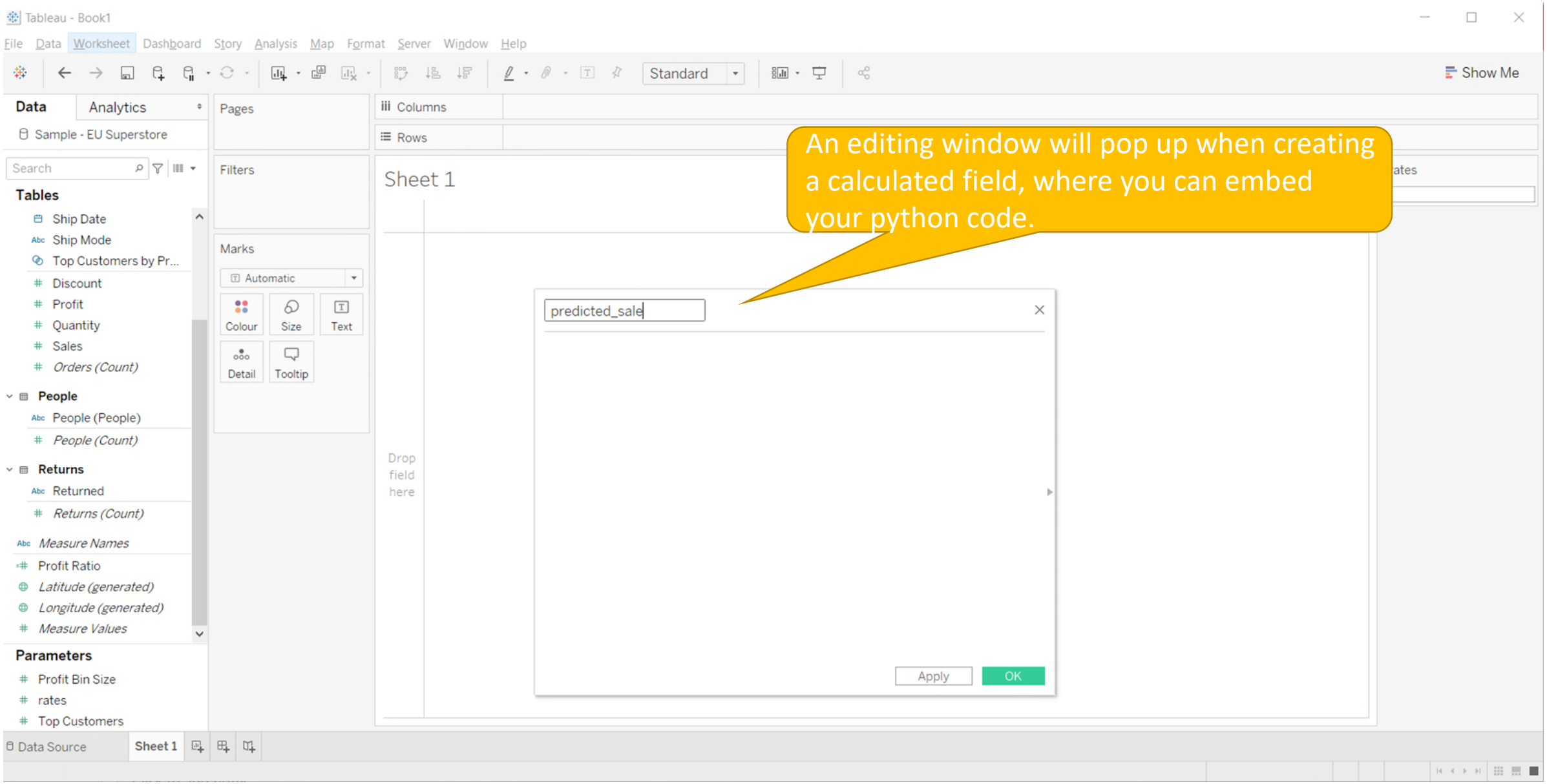


Tableau - Book1
File Data Worksheet Dashboard Story Analysis Map Format Server Window Help
Standard
Show Me
Data
Analytics
Sample - EU Superstore
Search
Tables
Ship Date
Ship Mode
Top Customers by Pr...
Discount
Profit
Quantity
Sales
Orders (Count)
People
People (People)
People (Count)
Returns
Returned
Returns (Count)
Measure Names
Profit Ratio
Latitude (generated)
Longitude (generated)
Measure Values
Parameters
Profit Bin Size
rates
Top Customers
Pages
Filters
Marks
Automatic
Colour
Size
Text
Detail
Tooltip
Columns
Rows
Sheet 1
Drop field here
An editing window will pop up when creating a calculated field, where you can embed your python code.
predicted_sale
Apply
OK
Data Source
Sheet 1

Sample Python Function

```
def predict_sales(sales,year,rate):
    predicted_sales = []
    for i in sales:
        prediction = i * rate ** year
        predicted_sales.append(prediction)
    return predicted_sales
```

Sample Code in Tableau

```
SCRIPT_INT(
"
rate = 1.2
predicted_sales = []
for i in _arg1:
    prediction = i * _arg2[0] ** _arg3[0]
    predicted_sales.append(prediction)
return predicted_sales
",
SUM([Sales]),
[rate],
[years]
)
```

General Rules to Embed Python Code

1. Make sure Python backend is running and connected to Tableau by executing "tabpy" in your prompt or terminal and setting up connection in Tableau.
2. Python code needs to be embedded in the Tableau's SCRIPT_* function. Depending on the data type of returned results, it could be SCRIPT_INT, SCRIPT_BOOL, SCRIPT_REAL, or SCRIPT_STR.
3. Python code needs to be embedded as a string wrapped in quotes.
4. Python function parameters will be named as _argN, depending on the order of arguments specified.
5. Embedded Python code string will be followed by arguments corresponding to the parameter _argN, separated by comma. For example, SUM([Sales]) will be passed to _arg1, rate will be passed to _arg2, etc.
6. Measures from Tableau dataset should be passed in the aggregated form, such as SUM(*) or AVG(*), because the calculation is across table.
7. The calculation is performed for all relevant cells in the table, therefore, measures, such as Sales in this case, will be passed as a list of values (i.e. the column of Sales). As a result, you need to write iteration loops to perform the calculation for each individual items in the list. The returned result thus should also be a list of calculated values, such as predicted_sales.
8. Tableau parameters, such as rates and years, will be passed as iterables to your Python code. Therefore, it may need to be referred using index, such as _arg2[0], when single value calculation is required.

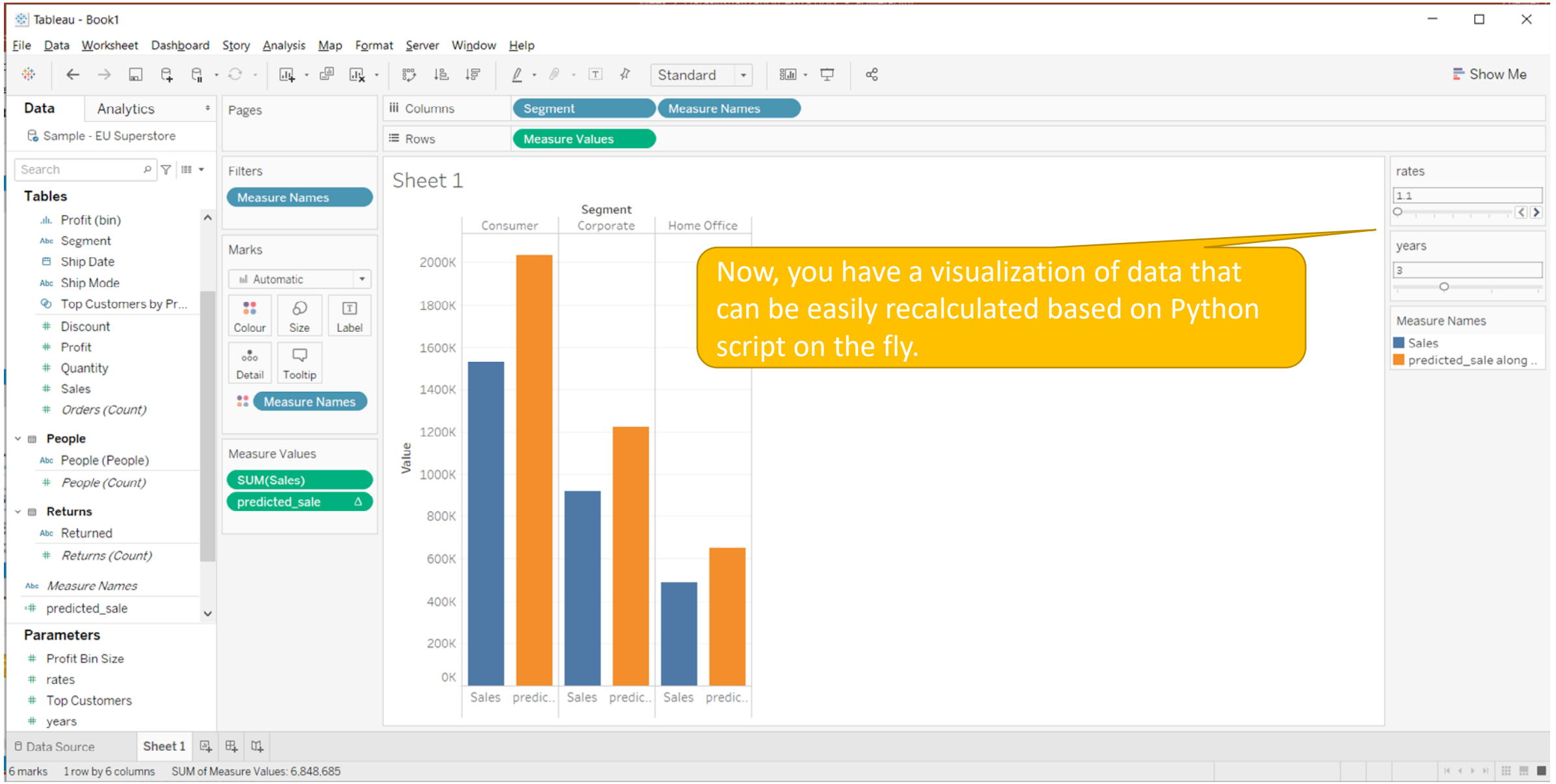


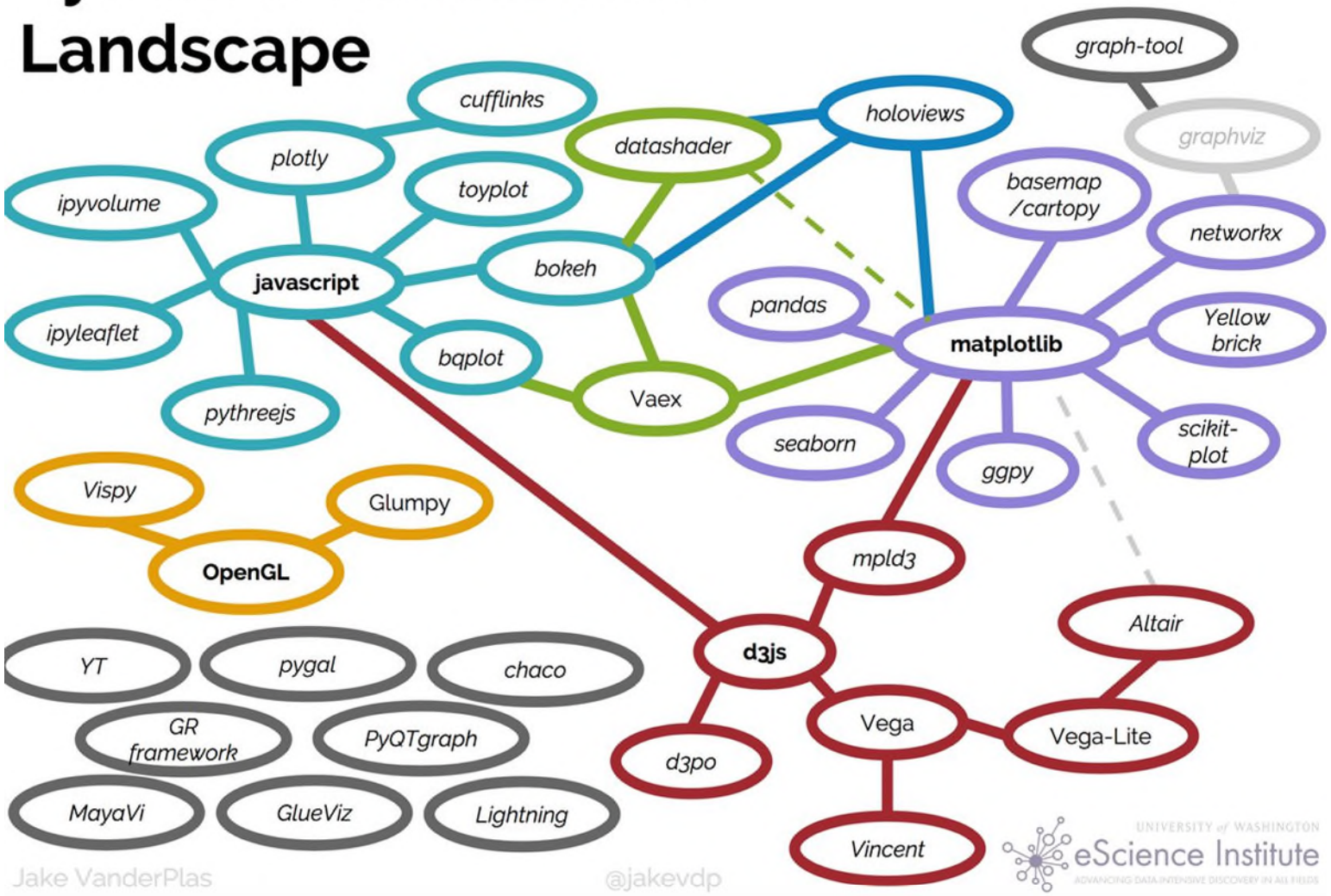
Tableau - Book1
File Data Worksheet Dashboard Story Analysis Map Format Server Window Help
Standard
Show Me
Data
Analytics
Sample - EU Superstore
Search
Tables
Profit (bin)
Segment
Ship Date
Ship Mode
Top Customers by Pr...
Discount
Profit
Quantity
Sales
Orders (Count)
People
People (People)
People (Count)
Returns
Returned
Returns (Count)
Measure Names
predicted_sale
Parameters
Profit Bin Size
rates
Top Customers
years
Pages
Filters
Measure Names
Marks
Automatic
Colour
Size
Label
Detail
Tooltip
Measure Names
Measure Values
SUM(Sales)
predicted_sale
Columns
Segment
Measure Names
Rows
Measure Values
Sheet 1
Segment
Consumer
Corporate
Home Office
Value
2000K
1800K
1600K
1400K
1200K
1000K
800K
600K
400K
200K
0K
Sales
predic..
Sales
predic..
Sales
predic..
Now, you have a visualization of data that can be easily recalculated based on Python script on the fly.
rates
1.1
years
3
Measure Names
Sales
predicted_sale along ..
Data Source
Sheet 1
6 marks
1 row by 6 columns
SUM of Measure Values: 6,848,685

Data Visualization in Python

Data visualization

- Eyeballing is a good way to have some quick initial understanding of your data. It is often used as the first step to start working on the data you just collected.
- There are tons of data visualization libraries for Python. Many data analysis libraries also provide built-in visualization functions.

Python's Visualization Landscape



Matplotlib and Seaborn

- **Matplotlib** is a plotting library for Python. It is the basis of many other libraries' visualization functionality. General and powerful yet a bit tricky sometimes.
- **Seaborn** is a data visualization library based on Matplotlib. It focuses on statistical visualization and provides easier, more user-friendly way to generate prettier graphs.
- We use build-in datasets for demonstration. You can explore these datasets here: https://github.com/mwaskom/seaborn-data

Simple Visualization with Seaborn

Tips dataset

total_bill	tip	sex	smoker	day	time	size
16.99	1.01	Female	No	Sun	Dinner	2
10.34	1.66	Male	No	Sun	Dinner	3
21.01	3.5	Male	No	Sun	Dinner	3

```
import seaborn as sns
tips = sns.load_dataset("tips")
```

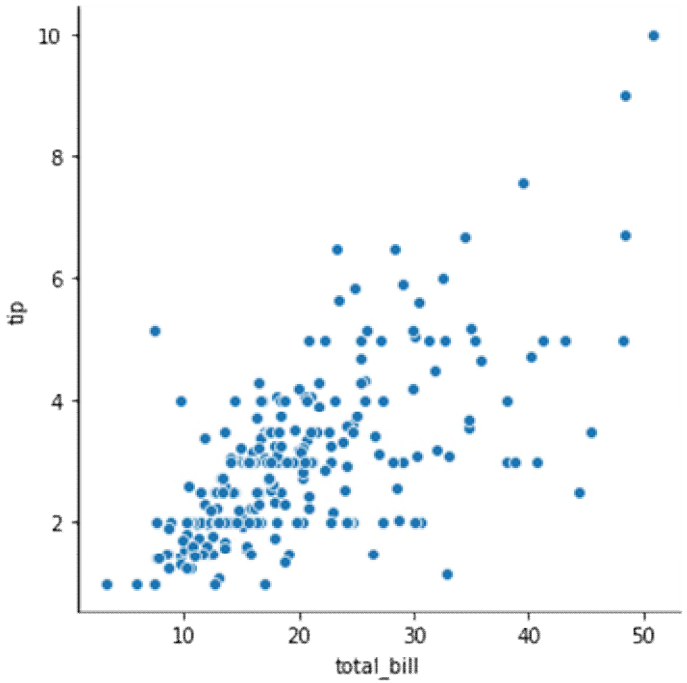
Function names
Important things
Parameter names
Values for argument

Scatter Plots

- Scatter plot is one of the most used visualization to explore the relationships between (two) variables.
- `relplot()` is a seaborn function that draws relational plots.

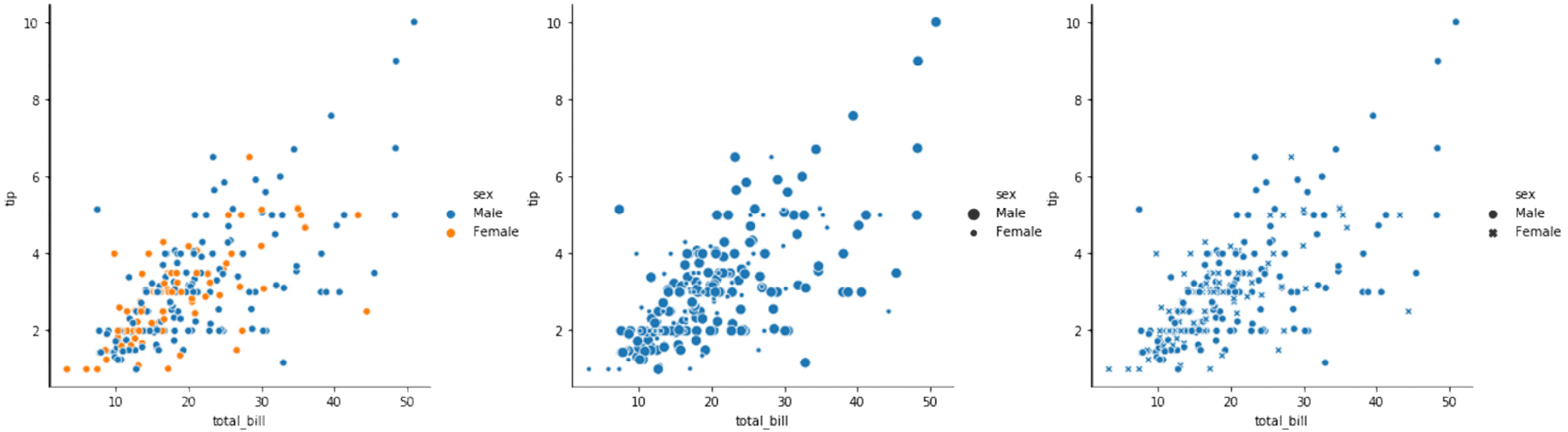
```
sns.relplot(x="total_bill", y="tip", data=tips)
```

#three required arguments: x variable name, y variable name and the dataset name. *Values of x and y need to be numeric*.



Optional arguments

- You may specify additional variables from the dataset to be used as the extra dimensions to group your data points, by passing the variable name to the arguments:
  1. hue: your data will be group in different colours.
     - ❖To change colour themes, you can use parameter "palette".
  2. size: your data will be group in different size.
  3. style: your data will be group in different style.
  4. They can be used together.


tip
total_bill
sex
Male
Female

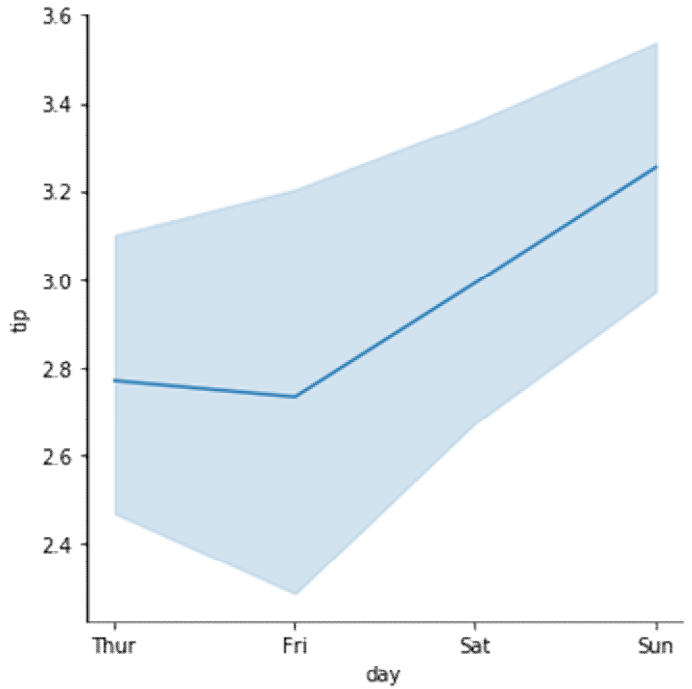
Exercise

- Draw a scatter plotter about the relationship between tip and total bill. Explore how different factors, such as smoker, day, time, size may affect this relationship.

Line plot

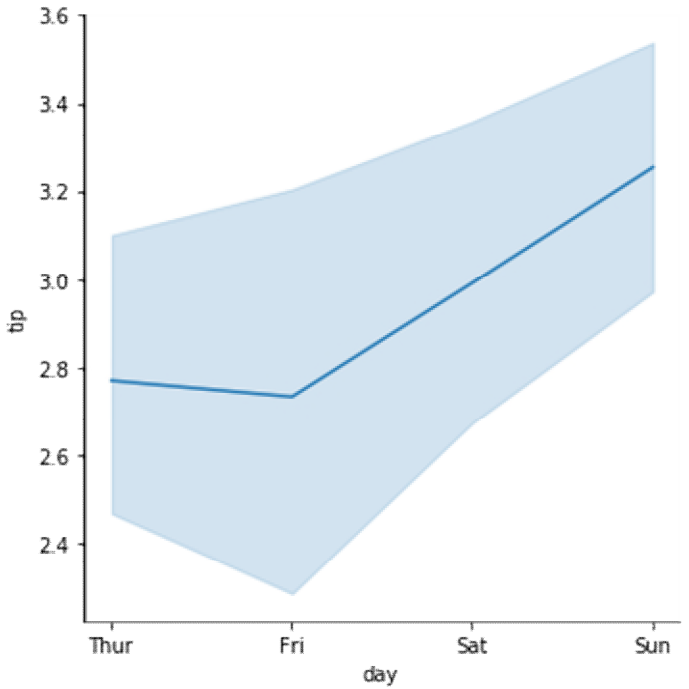
- When you have continuous variables, such as time, it can be a good idea to view the changes with line plot.
- An optional argument in `relplot()` is kind, with default value "scatter" for scatter plot, can be passed with value "line" to draw a line plot.

```
sns.relplot(x="day", y="tip", kind='line', data=tips)
```

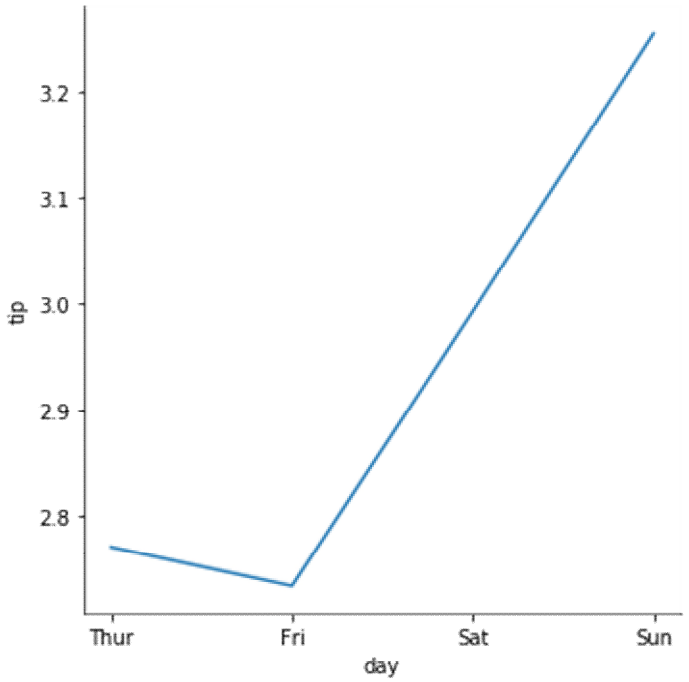


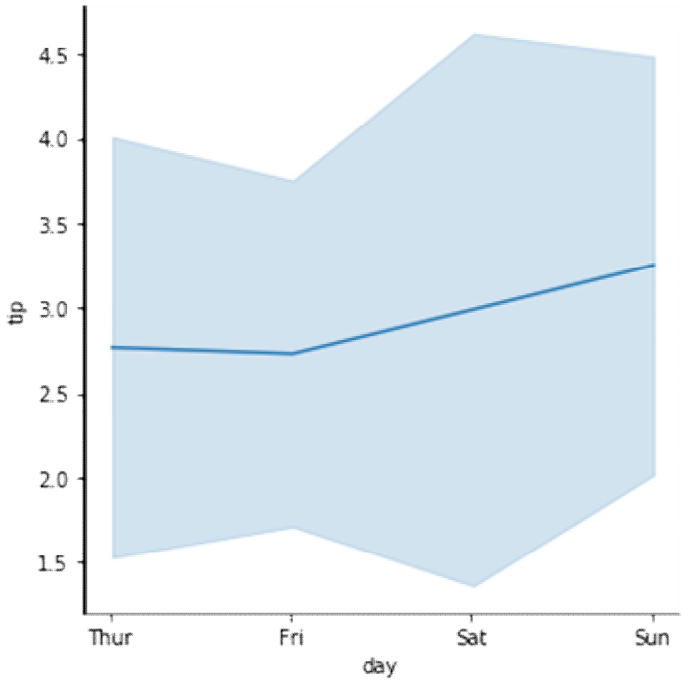
Line plot

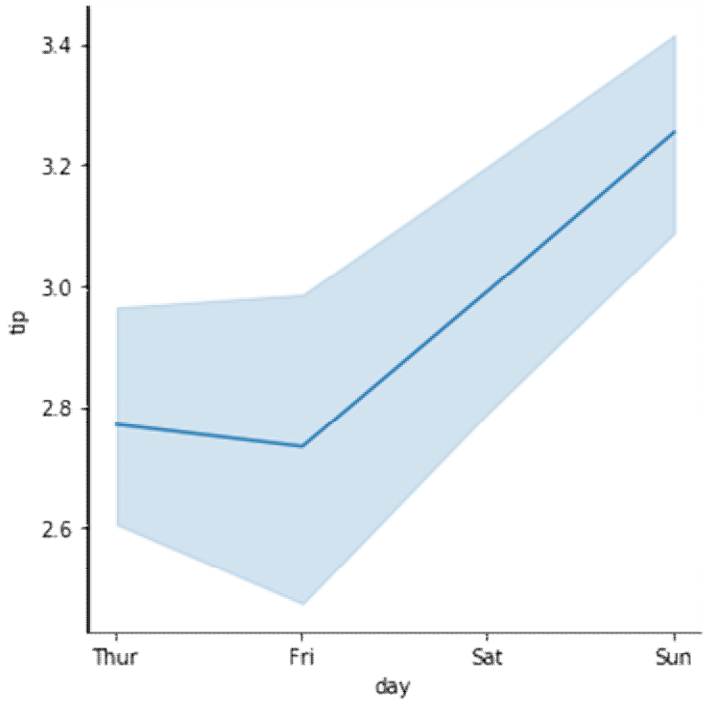
- By default, Seaborn aggregates the multiple measurements at each x value by plotting the mean and the 95% confidence interval around the mean
- You can change both by passing arguments:
  1. ci: value, none, or 'sd'.
  2. estimator: none or other pandas estimator.



Different confidence intervals

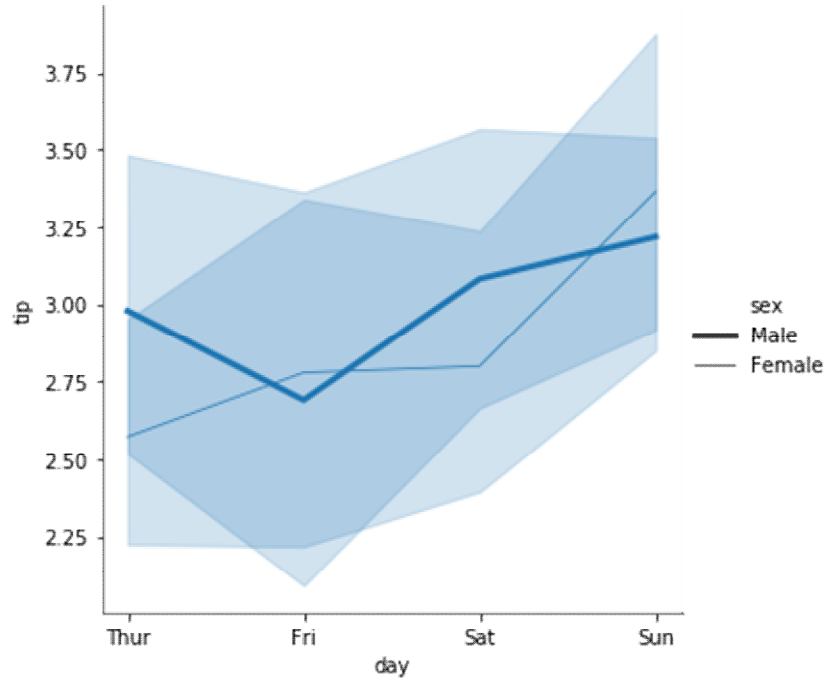


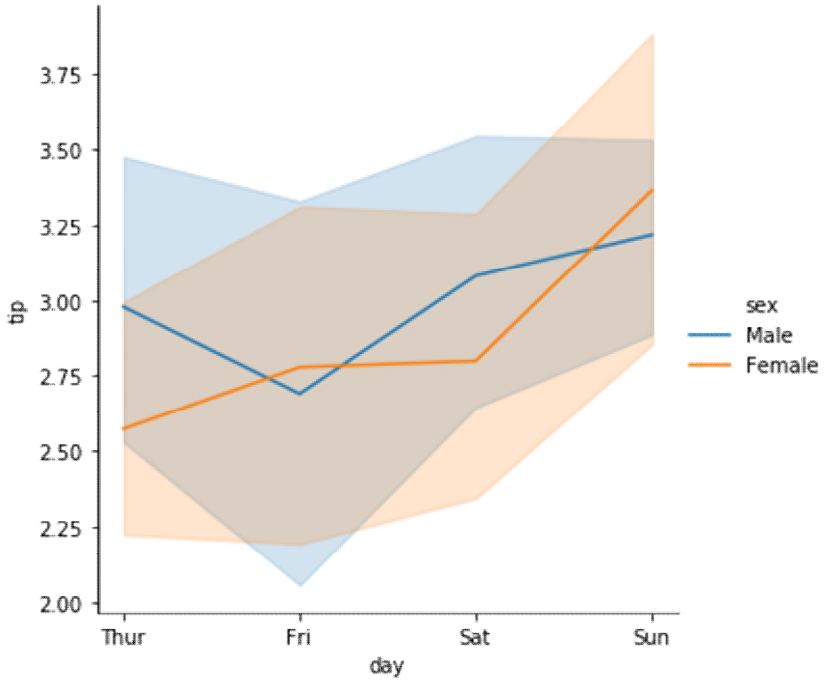


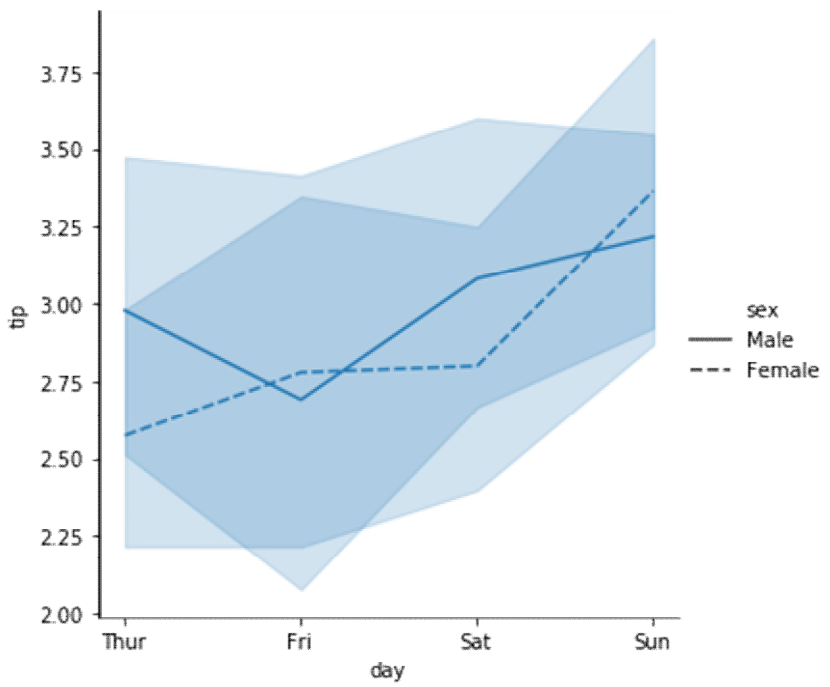


Line plotting by groups

- You can also create multiple lines based on the groups spitted by hue, size and style.



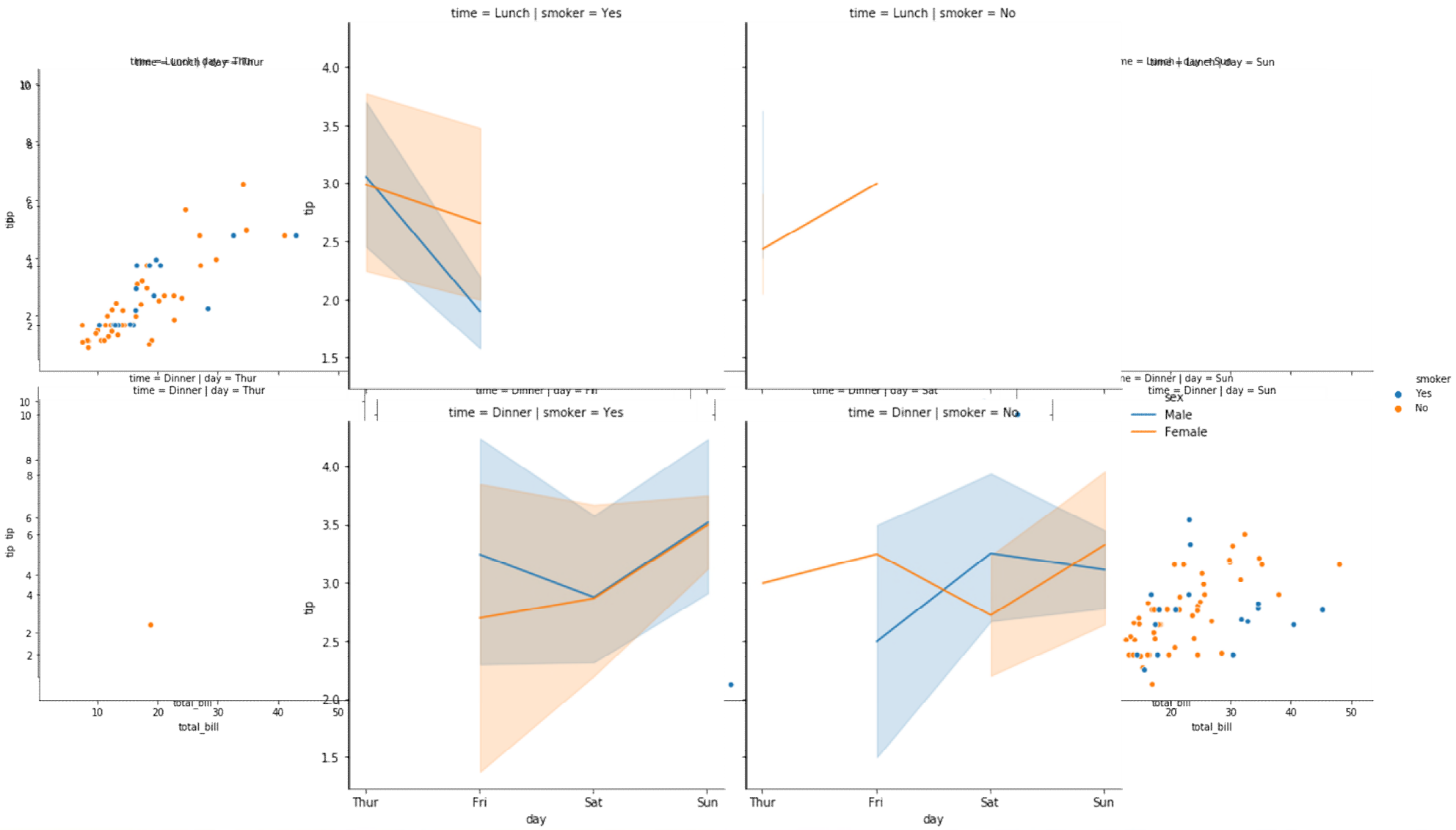




Showing multiple relationships

- Very often, when you explore your dataset, you would like to compare multiple relationships at once.
- You can create a grid of graphs and specify facets (grouping dimensions) by passing arguments:
    1. col: the variable used for splitting in horizontal direction.
    2. row: the variable used for splitting in vertical direction.

```
sns.relplot(x="total_bill", y="tip", col='day',
row='time', data=tips)
```


time = Lunch | smoker = Yes
time = Lunch | smoker = No
time = Dinner | day = Thur
time = Dinner | smoker = Yes
time = Dinner | smoker = No
tip
total_bill
day
Thur
Fri
Sat
Sun
smoker
Yes
No
sex
Male
Female

Plotting categorical variables

- `relplot()`, while is best to plot relationship between numeric variables, is able to handle categorical variables as well.
- `catplot()` is a function dedicated for categorical plotting.
- `sns.catplot(x="day", y="total_bill", data=tips)`