Data Shaping & Visualization

Agenda

- **Midterm**
- Final Project
- Data Shaping
  - Merge & joins
  - Concat
  - Clean
- Tableau
- Matplotlib

Midterm

Midterm – 2 parts

- In-class (20 questions, 1 hour)
- Take-home (1 dataset)
  - Will be in Camino by Sunday
  - Due date next Saturday

Agenda

- Midterm
- **Final Project**
- Data Shaping
  - Merge & joins
  - Concat
  - Clean
- Tableau
- Matplotlib

Your final project : guidelines

- Goal: **apply** what you have learned in this class to a realistic data science challenge + exercise your creativity + have fun!
- This is meant to be a significant **individual effort** to learn by practicing what you are learning to a real-world data science problem.
- The **writeup** of your final project is in the form of a Jupyter notebook and associated data – to be uploaded to the final project assignment in Camino.
- You are to submit your final notebook by September 3 @ 11:59pm.

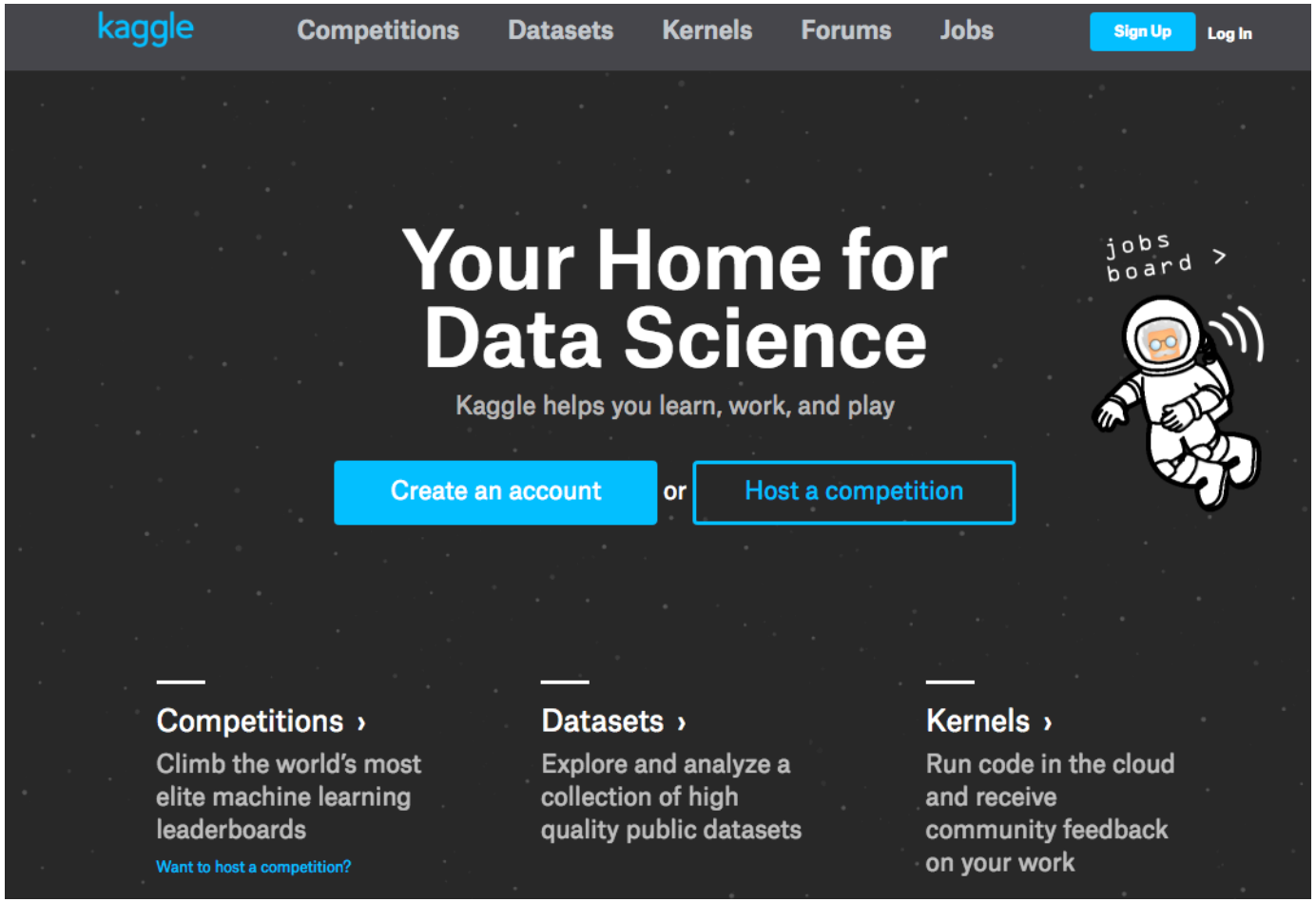
Your final project : topic selection

- Goal: **apply** what you have learned in this class to a realistic data science challenge + exercise your creativity + have fun!
- You can choose any ”significant” data set via downloadable sites, APIs, or use any of the datasets from the class.
- You need to propose an interesting data insight investigation that you would like to explore, analyze the data, visualize the data, and finally write up your conclusion on what insights you have reached.
- Grading of your final project will be based on the following rubric.

Your final project : grading rubric

Area	Details	Grading %
Topic Selection	Did you create a reasonably interesting data insight hypothesis for your investigation?	10%
Packaging	Did you create a Jupyter project packaging that looks professional and understandable?	10%
Analysis Competence	Does your notebook show competence in using the data science tools we learned in class?	40%
Insight	Does your project show useful or interesting insights from the data analysis you have done?	40%

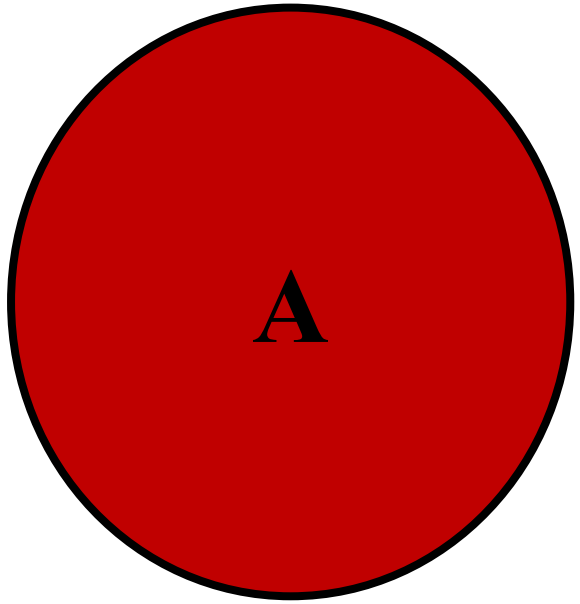
Final Project

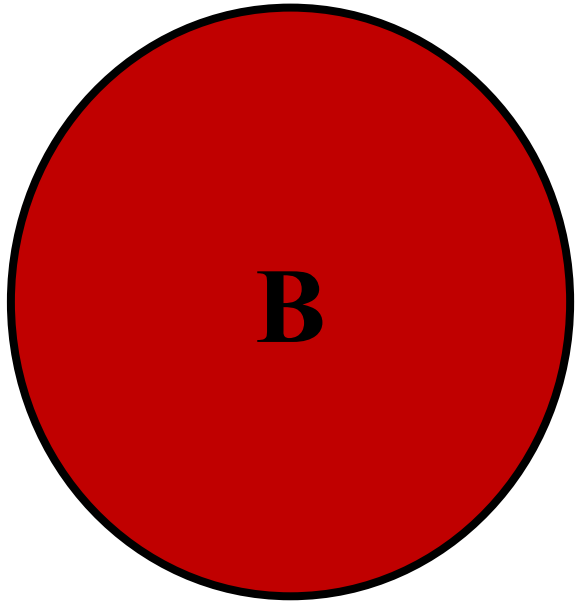


Agenda

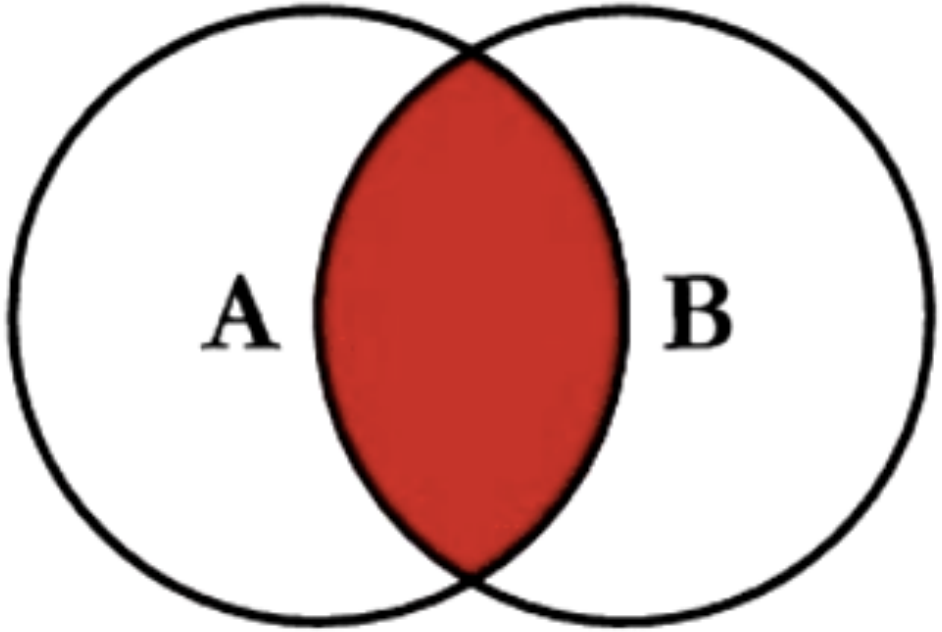
- Midterm
- Final Project
- **Data Shaping**
  - **Merge & joins**
  - **Concat**
  - **Clean**
- Tableau
- Matplotlib

Joining Datasets



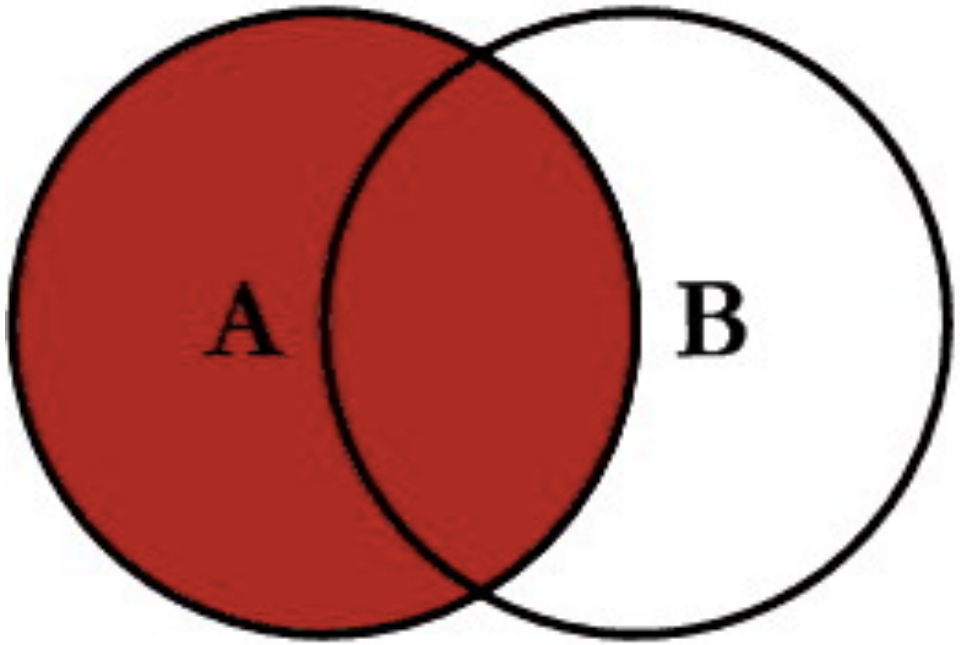


Joining Datasets : Inner Join



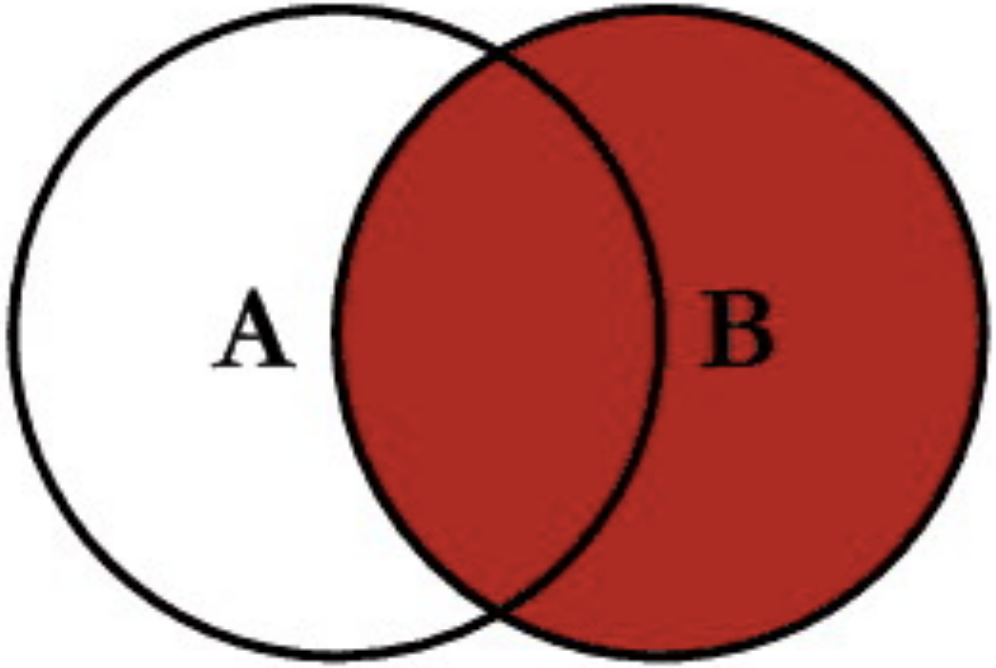

```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```

Joining Datasets : Left Outer Join



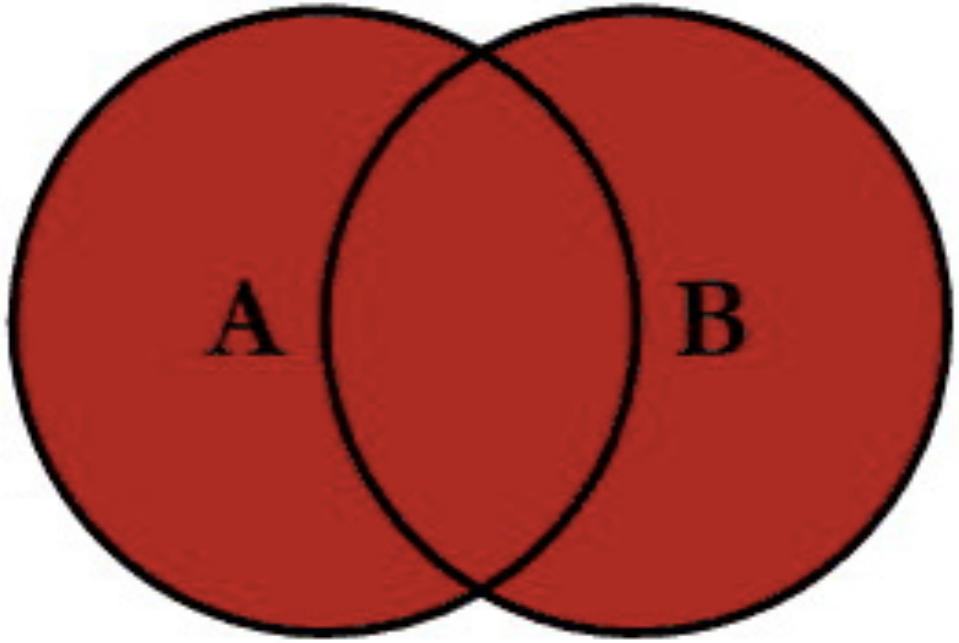

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```

Joining Datasets : Right Outer Join




```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```

Joining Datasets : Full Outer Join




```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
```

Merging (inner join)

data1	key
0	b
1	b
2	a
3	c
4	a
5	a
6	b

merge

data2	key
0	a
1	b
2	d

→

data1	key	data2
0	b	1
1	b	1
6	b	1
2	a	0
4	a	0
5	a	0

Merging (inner join)

data1	key
0	b
1	b
2	a
3	c
4	a
5	a
6	b

inner join

data2	key
0	a
1	b
2	d

→

data1	key	data2
0	b	1
1	b	1
6	b	1
2	a	0
4	a	0
5	a	0

Merging (left outer join)

data1	key
0	b
1	b
2	a
3	c
4	a
5	a
6	b

outer join

data2	key
0	a
1	b
2	d

→

data1	key	data2
0	b	1.0
1	b	1.0
2	a	0.0
3	c	NaN
4	a	0.0
5	a	0.0
6	b	1.0

Merging (right outer join)

data1	key
0	b
1	b
2	a
3	c
4	a
5	a
6	b

outer join

data2	key
0	a
1	b
2	d

→

data1	key	data2
0.0	b	1
1.0	b	1
6.0	b	1
2.0	a	0
4.0	a	0
5.0	a	0
NaN	d	2

Merging (full outer join)

data1	key
0	b
1	b
2	a
3	c
4	a
5	a
6	b

outer join

data2	key
0	a
1	b
2	d

→

data1	key	data2
0.0	b	1.0
1.0	b	1.0
6.0	b	1.0
2.0	a	0.0
4.0	a	0.0
5.0	a	0.0
3.0	c	NaN
NaN	d	2.0

GroupBy : setup

```
import pandas as pd
scores = pd.DataFrame({
        'name'    : ['Avery', "Bill", "Cathy", "Dave"],
        'age'     : [32, 45, 33, 29],
        'test1'   : [92, 82, 65, 79],
        'test2'   : [99, 89, 98, 60],
        'teacher': ['Mandy', 'Nancy', 'Mandy', 'Nancy']
    })
scores
```

	age	name	teacher	test1	test2
0	32	Avery	Mandy	92	99
1	45	Bill	Nancy	82	89
2	33	Cathy	Mandy	65	98
3	29	Dave	Nancy	79	60

GroupBy : by teacher

```
scores.groupby('teacher').median()
```

	age	test1	test2
teacher			
Mandy	32.5	78.5	98.5
Nancy	37.0	80.5	74.5

GroupBy : by teacher

```
scores.groupby('teacher').median()[['test1', 'test2']]
```

	test1	test2
teacher		
Mandy	78.5	98.5
Nancy	80.5	74.5

GroupBy : specific aggregations

```
scores.groupby(['teacher', 'age']).agg([min, max])
```

		name		test1		test2	
		min	max	min	max	min	max
teacher	age						
Mandy	32	Avery	Avery	92	92	99	99
	33	Cathy	Cathy	65	65	98	98
Nancy	29	Dave	Dave	79	79	60	60
	45	Bill	Bill	82	82	89	89

Stacking & Unstacking

- **stack**: this “rotates” or pivots from the columns in the data to the rows
- **unstack**: this pivots from the rows into the columns

Stacking

```
data = DataFrame(np.arange(6).reshape((2, 3)),
                 index=pd.Index(['Ohio', 'Colorado'], name='state'),
                 columns=pd.Index(['one', 'two', 'three'], name='number'))
data
```

number	one	two	three
state			
Ohio	0	1	2
Colorado	3	4	5

Stacking

```
result = data.stack()
result
```

```
state     number
Ohio      one       0
          two       1
          three     2
Colorado  one       3
          two       4
          three     5
dtype: int64
```

Stacking

```
result.unstack()
```

number	one	two	three
state			
Ohio	0	1	2
Colorado	3	4	5

Stacking

```
result.unstack(0)
```

state	Ohio	Colorado
number		
one	0	3
two	1	4
three	2	5

Stacking

```
result.unstack('state')
```

state	Ohio	Colorado
number		
one	0	3
two	1	4
three	2	5

Stacking : may introduce Missing Values

```
s1 = Series([0, 1, 2, 3], index=['a', 'b', 'c', 'd'])
s2 = Series([4, 5, 6], index=['c', 'd', 'e'])
data2 = pd.concat([s1, s2], keys=['one', 'two'])
data2.unstack()
```

	a	b	c	d	e
one	0.0	1.0	2.0	3.0	NaN
two	NaN	NaN	4.0	5.0	6.0

Stacking : may introduce Missing Values

```
data2.unstack().stack()
```

```
one   a    0.0
      b    1.0
      c    2.0
      d    3.0
two   c    4.0
      d    5.0
      e    6.0
dtype: float64
```

Stacking : may introduce Missing Values

```
data2.unstack().stack(dropna=False)
```

```
one  a    0.0
     b    1.0
     c    2.0
     d    3.0
     e    NaN
two  a    NaN
     b    NaN
     c    4.0
     d    5.0
     e    6.0
dtype: float64
```

Open notebook: “lecture05.data.shaping”

Agenda

- Midterm
- Final Project
- Data Shaping
  - Merge & joins
  - Concat
  - Clean
- **Tableau**
- Matplotlib

Tableau

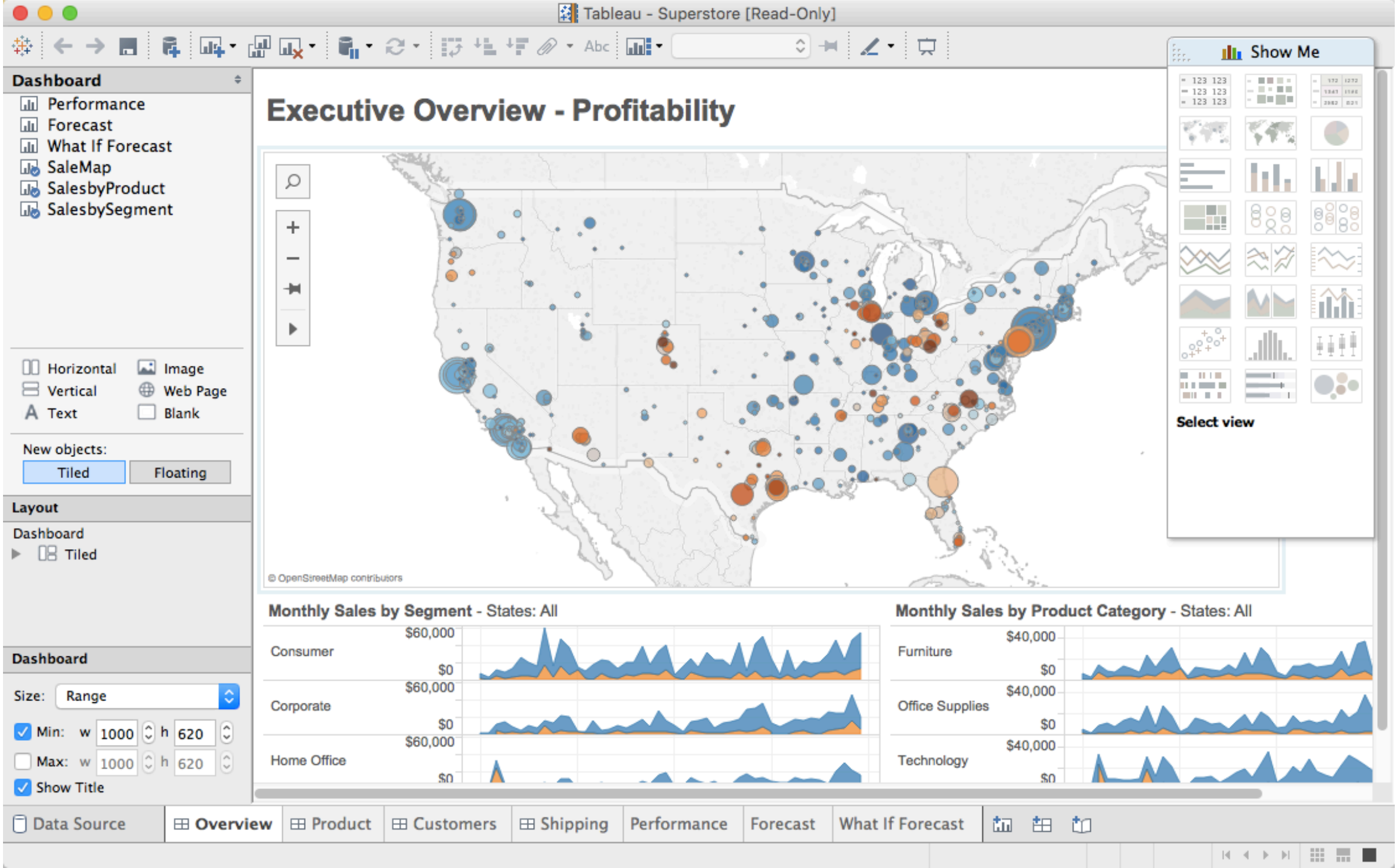


Tableau - Superstore [Read-Only]
Dashboard
Performance
Forecast
What If Forecast
SaleMap
SalesbyProduct
SalesbySegment
Horizontal
Vertical
Text
Image
Web Page
Blank
New objects:
Tiled
Floating
Layout
Dashboard
Tiled
Dashboard
Size:
Range
Min: w 1000 h 620
Max: w 1000 h 620
Show Title
Executive Overview - Profitability
© OpenStreetMap contributors
Show Me
Select view
Monthly Sales by Segment - States: All
Consumer
Corporate
Home Office
$60,000
$0
Monthly Sales by Product Category - States: All
Furniture
Office Supplies
Technology
$40,000
$0
Data Source
Overview
Product
Customers
Shipping
Performance
Forecast
What If Forecast

Student Tableau License

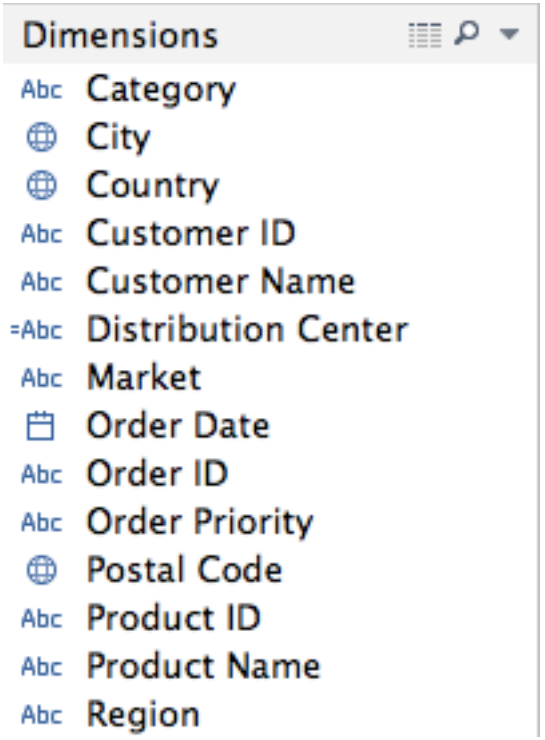
http://tableau.com/students

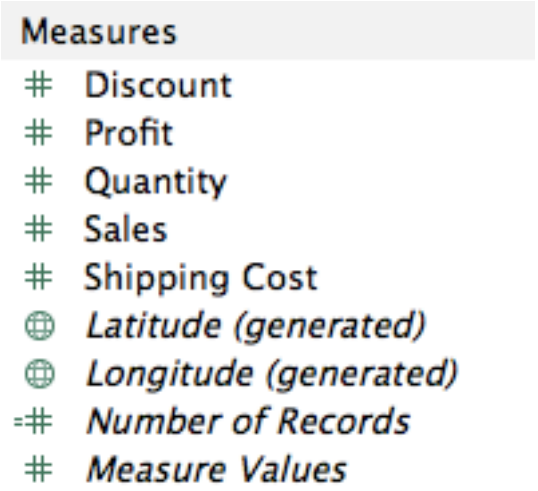
Data Ingestion

- Joins: inner, left, right, full
- Extract – Transform – Load (ETL)
- Field Transformation
- Live / Extract
- Filtering
- Large dataset & role of Tableau

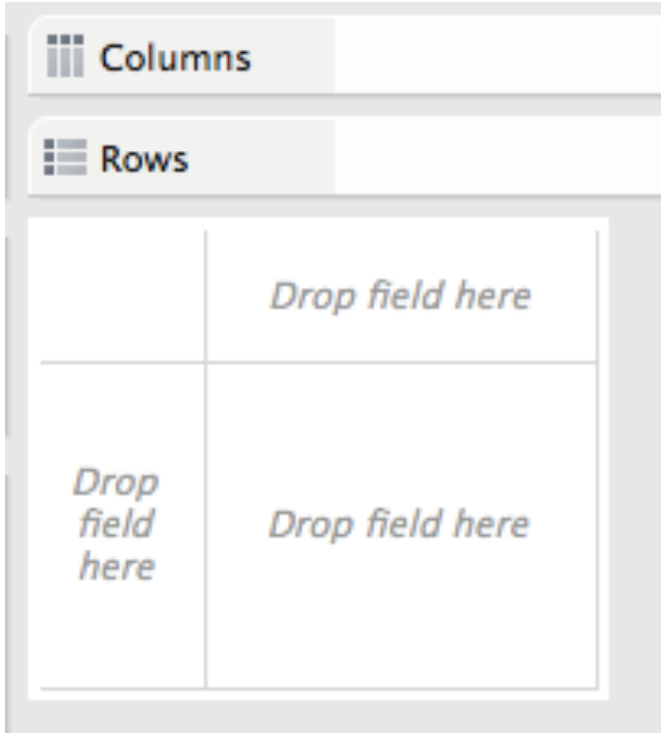
Dimensions & Measures

- Dimensions : categorical
- Measures : numerical

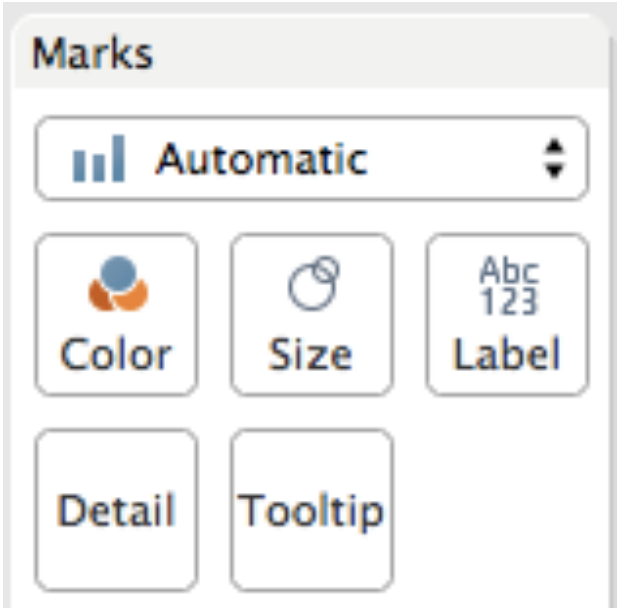




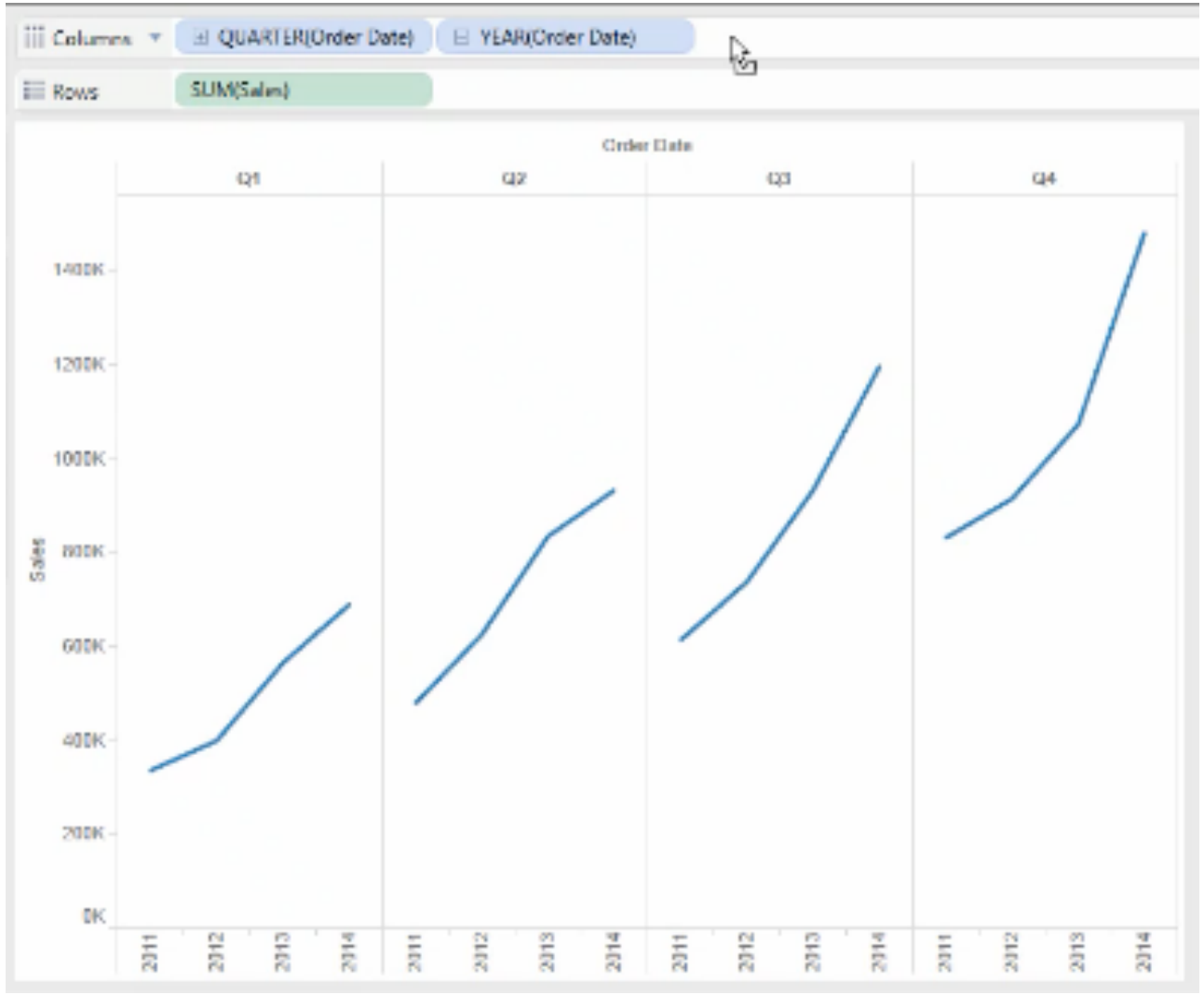
Rows & Columns



Marks



Multi-level Analysis



Group Exercise

- Load the “global_superstore.xls” dataset into Tableau
- Answer the following questions:
  1. Which region has the highest sales?
  2. For (1), which product segment for that region has the highest sales?
  3. Regardless of region, which product segment in a given region has the highest sales?
- Take 15 min in your group to decide on extracting an interesting insight from this dataset to present to the class.

Agenda

- Midterm
- Final Project
- Data Shaping
  - Merge & joins
  - Concat
  - Clean
- Tableau
- **Matplotlib**

Matplotlib

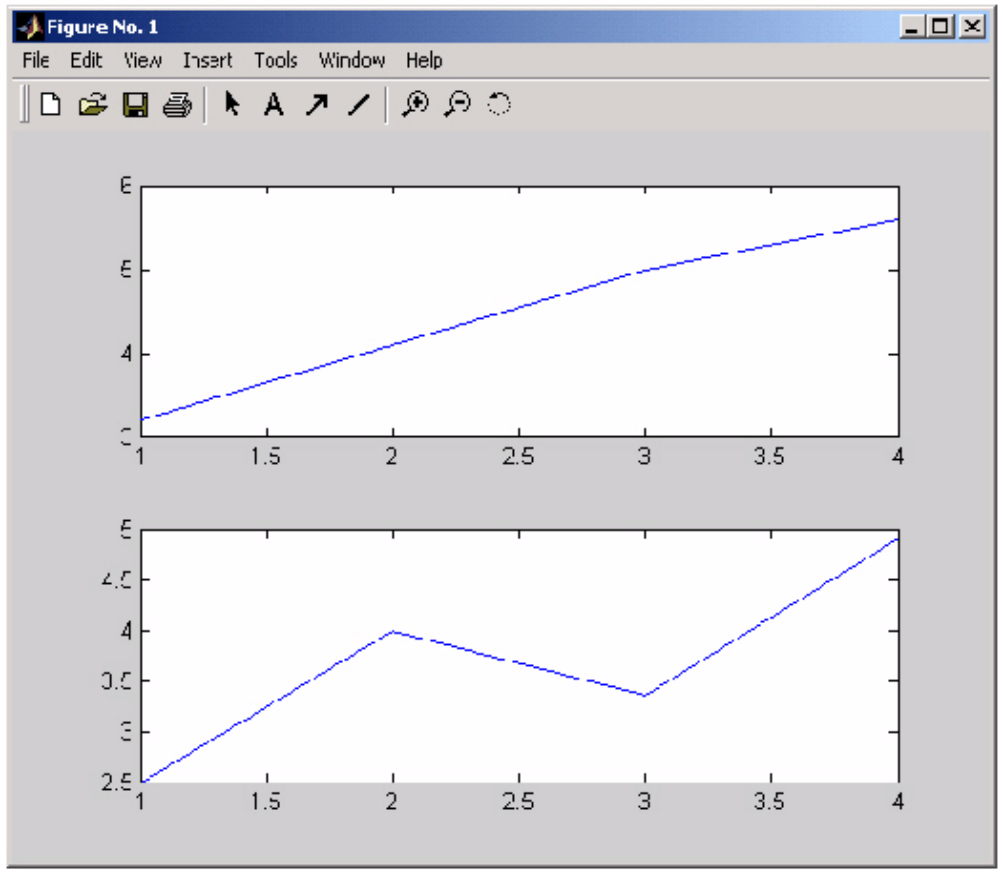
Matplotlib

- **Matplotlib** is a python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms.
- **Matplotlib** is the whole package; **pylab** is a module in matplotlib that gets installed alongside matplotlib; and **matplotlib.pyplot** is a module in matplotlib
- **Pyplot** provides the state-machine interface to the underlying plotting library in matplotlib.

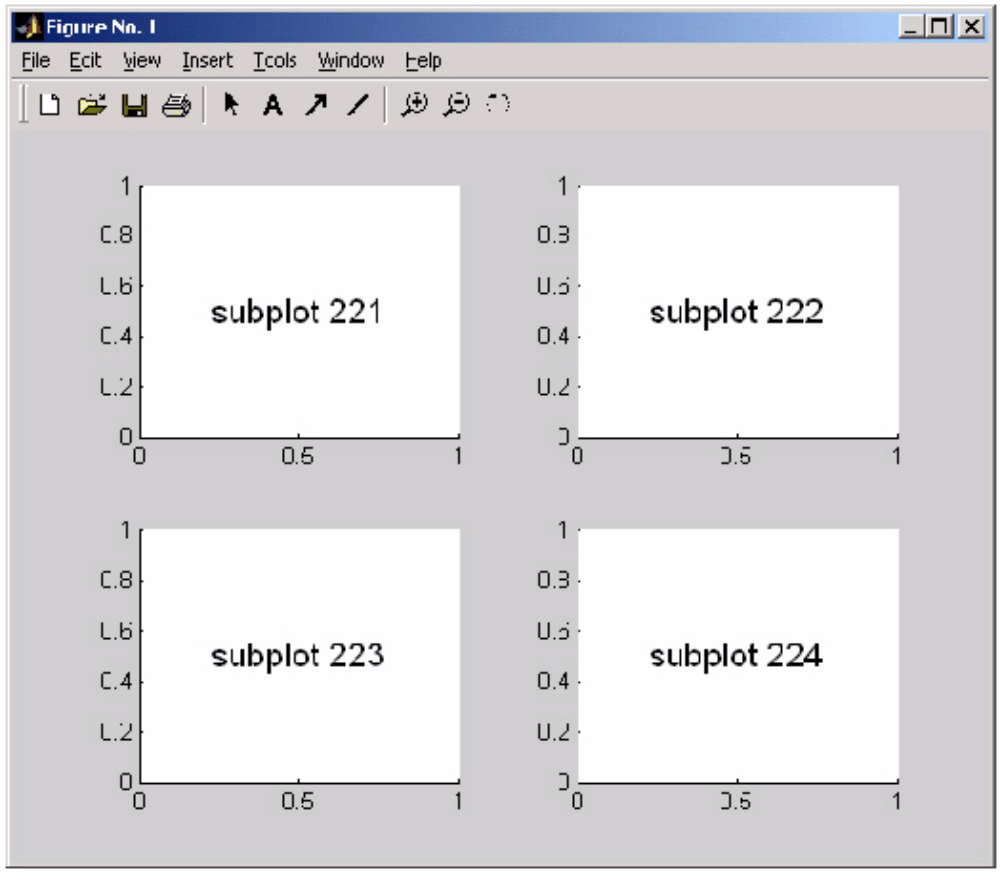
Subplot

- Subplot divides the current figure into rectangular panes that are numbered row wise.
- Each pane contains an axes object. Subsequent plots are output to the current pane.

Subplot grid 2X1



Subplot grid 2x2



Matplotlib Gallery

http://matplotlib.org/examples/index.html