HS19 Data Visualization Concepts

Exercise Session 1
Basic Data Manipulation

Haiyan Yang

haiyan@ifi.uzh.ch
Visualization and MultiMedia Lab
Department of Informatics
University of Zürich

University of Zurich[UZH]

VISUALIZATION AND MULTIMEDIA LAB

Exercise Sessions

Exercise Sessions
General Overview

- Fundamental concepts of data visualization
  - Digitization, Color and perception, Multivariate data analysis and visualization, Spatial and geospatial data visualization, Trees, Graphs and networks
- Programming exercises using Python
  - Python familiarity for this course will be needed
- Takes place on specific Thursdays throughout the semester
  - 5 exercise sessions are scheduled, with exercise point distribution: 2 - 3 - 2 - 5 - 3
  - A minimum points of 5 must be achieved out of exercise 1, 2 and 4
- Contact (for exercises): Kate Gadola <kate.gadola@uzh.ch>
  - Discussions and questions will be handled in OLAT forum (technical questions) and VisGuides (theoretical questions)
  - For meeting in person, arrange first an appointment
  - Please use the platforms as much as possible such that all the students can benefit from them

Exercise Sessions
Session Structure

- Introduction to new theoretical and technical aspects related to new assignment
  - Including examples, code structure, demos, sample results if necessary, etc.
- Announcement of new assignment (one in every session)
  - Discussion of requirements, solution hints, etc.
  - Clarification of point requirement.
- Discussion
  - Including comments on submissions, answering questions regarding the grading, etc.
  - Questions regarding the newly assigned exercise can be handled shortly after the lecture (15mins) or on the OLAT later on.
- Takes place in the second half of the lecture on Thursdays

Exercise Sessions
Assignment Guidelines

- Assignments related only to Data Visualization course.
- For every assignment there will be a .zip file in OLAT, including:
  - Exercise's announcement (***what to do*** — *assignment requirements in details*)
  - Presentation of theoretical and technical aspects (***how to do*** — *technical guidance*)
  - Data files and code skeleton (with needed libraries and hints) will be provided when necessary
  - **Attention!** Uploaded .zip file will **NOT** include additional material presented in lab session , such as demos, code snippets, solutions from previous sessions, etc.
- Grading will be ***incomplete*** or ***complete***
  - Only a **complete** will result in the corresponding awarded points
- Exercises MUST be submitted before each deadline, otherwise it will be a FAIL.

Exercise Sessions
Assignment Submission

- Project files must be zipped and the .zip archive has to be named: dvc_ex1_MATRIKELNUMBER.zip
  - e.g. dvc_ex1_01234567.zip
- Follow exercise instructions and provide an answer sheet when necessary (e.g. a “readme.txt” file)
- No trash / debug output
- Use OLAT to submit files
  - Deadlines are typically Wednesdays at 23:59
- Use OLAT forum to ask questions about the technical questions from exercises

Instructions

Instructions
Dataset

- eBird Basic Dataset (EBD)
  - For more information: http://strimas.com/ebird-best-practices/ebird.html
- It is a multidimensional dataset
  - **Dataset type**: table data
  - **Dimensionality**: 1939*47
  - **row**: each record item
  - **column**: different attributes describing the data, e.g., category of the bird, name of the bird, country of the observation, observation date and time and geographic information of the location, etc.
  - **File format**: .txt (can be read by using **pandas.read_csv()**)

Instructions

Dataset

GLOBAL UNIQUE IDENTIFIER	LAST EDITED DATE	TAXONOMIC ORDER	CATEGORY	COMMON NAME	SCIENTIFIC NAME	SUBSPECIES COMMON NAME	SUBSPECIES SCIENTIFIC NAME	OBSERVATION COUNT	E
URN:CornellLabOfOrnithology:EBIRD:OBS570779581	2018-01-24 14:41:34.0	19073	species	American Crow	Corvus brachyrhynchos			2	
URN:CornellLabOfOrnithology:EBIRD:OBS563756637	2018-01-03 22:54:39.0	19073	species	American Crow	Corvus brachyrhynchos			2	
URN:CornellLabOfOrnithology:EBIRD:OBS563333659	2018-01-03 22:54:39.0	19073	species	American Crow	Corvus brachyrhynchos			2	
URN:CornellLabOfOrnithology:EBIRD:OBS563148351	2018-01-09 21:46:14.0	19073	species	American Crow	Corvus brachyrhynchos			1	
URN:CornellLabOfOrnithology:EBIRD:OBS566089279	2018-01-10 16:17:09.0	19073	species	American Crow	Corvus brachyrhynchos			1	
URN:CornellLabOfOrnithology:EBIRD:OBS563757539	2018-01-09 21:46:14.0	19073	species	American Crow	Corvus brachyrhynchos			1	
URN:CornellLabOfOrnithology:EBIRD:OBS573616617	2018-05-01 13:28:08.0	19073	species	American Crow	Corvus brachyrhynchos			3	
URN:CornellLabOfOrnithology:EBIRD:OBS570204089	2018-01-22 13:06:57.0	19073	species	American Crow	Corvus brachyrhynchos			1	
URN:CornellLabOfOrnithology:EBIRD:OBS565036961	2018-01-07 13:31:06.0	19073	species	American Crow	Corvus brachyrhynchos			1	
URN:CornellLabOfOrnithology:EBIRD:OBS565027324	2018-01-07 13:11:33.0	19073	species	American Crow	Corvus brachyrhynchos			3	
URN:CornellLabOfOrnithology:EBIRD:OBS563757396	2018-01-06 18:34:34.0	19073	species	American Crow	Corvus brachyrhynchos			3	
URN:CornellLabOfOrnithology:EBIRD:OBS563135906	2018-01-06 18:34:34.0	19073	species	American Crow	Corvus brachyrhynchos			3	
URN:CornellLabOfOrnithology:EBIRD:OBS570775035	2018-01-24 14:20:02.0	19073	species	American Crow	Corvus brachyrhynchos			4	
URN:CornellLabOfOrnithology:EBIRD:OBS570986312	2018-01-26 08:08:07.0	19073	species	American Crow	Corvus brachyrhynchos			4	
URN:CornellLabOfOrnithology:EBIRD:OBS566304678	2018-01-11 11:32:41.0	19073	species	American Crow	Corvus brachyrhynchos			2	
URN:CornellLabOfOrnithology:EBIRD:OBS571034544	2018-01-25 16:59:50.0	19073	species	American Crow	Corvus brachyrhynchos			1	
URN:CornellLabOfOrnithology:EBIRD:OBS570975325	2018-01-25 16:59:50.0	19073	species	American Crow	Corvus brachyrhynchos			1	
URN:CornellLabOfOrnithology:EBIRD:OBS563337268	2018-01-03 16:14:41.0	19073	species	American Crow	Corvus brachyrhynchos			4	
URN:CornellLabOfOrnithology:EBIRD:OBS566343995	2018-01-11 13:40:10.0	19073	species	American Crow	Corvus brachyrhynchos			1	
URN:CornellLabOfOrnithology:EBIRD:OBS571610540	2018-01-27 15:19:02.0	19073	species	American Crow	Corvus brachyrhynchos			1	
URN:CornellLabOfOrnithology:EBIRD:OBS572716949	2018-01-30 11:29:50.0	31039	species	American Goldfinch	Spinus tristis			1	
URN:CornellLabOfOrnithology:EBIRD:OBS563085699	2018-01-02 09:30:05.0	31039	species	American Goldfinch	Spinus tristis			1	

...

⋮

Instructions
Grading Scheme

- Tasks
  - **Data preprocessing**: data reading, basic data manipulation like deleting, filtering, extraction, etc.
  - **Data visualization**: bar chart plot for specific columns of data, specified in the assignment document

- Grading
  - Points: {0, 2}
  - **2**: only if you complete **all two** tasks and submit the requirement documents **before deadline**
  - **0**: partially complete all the tasks or miss of the deadline

Instructions
Data Preprocessing

- Data reading
  - Use the pandas library package to read the file into a data frame
  - Handle the errors by properly using the links provided in the code skeleton
- Data cleaning
  - Data frame provides the most commonly needed functions for data preprocessing
  - https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html

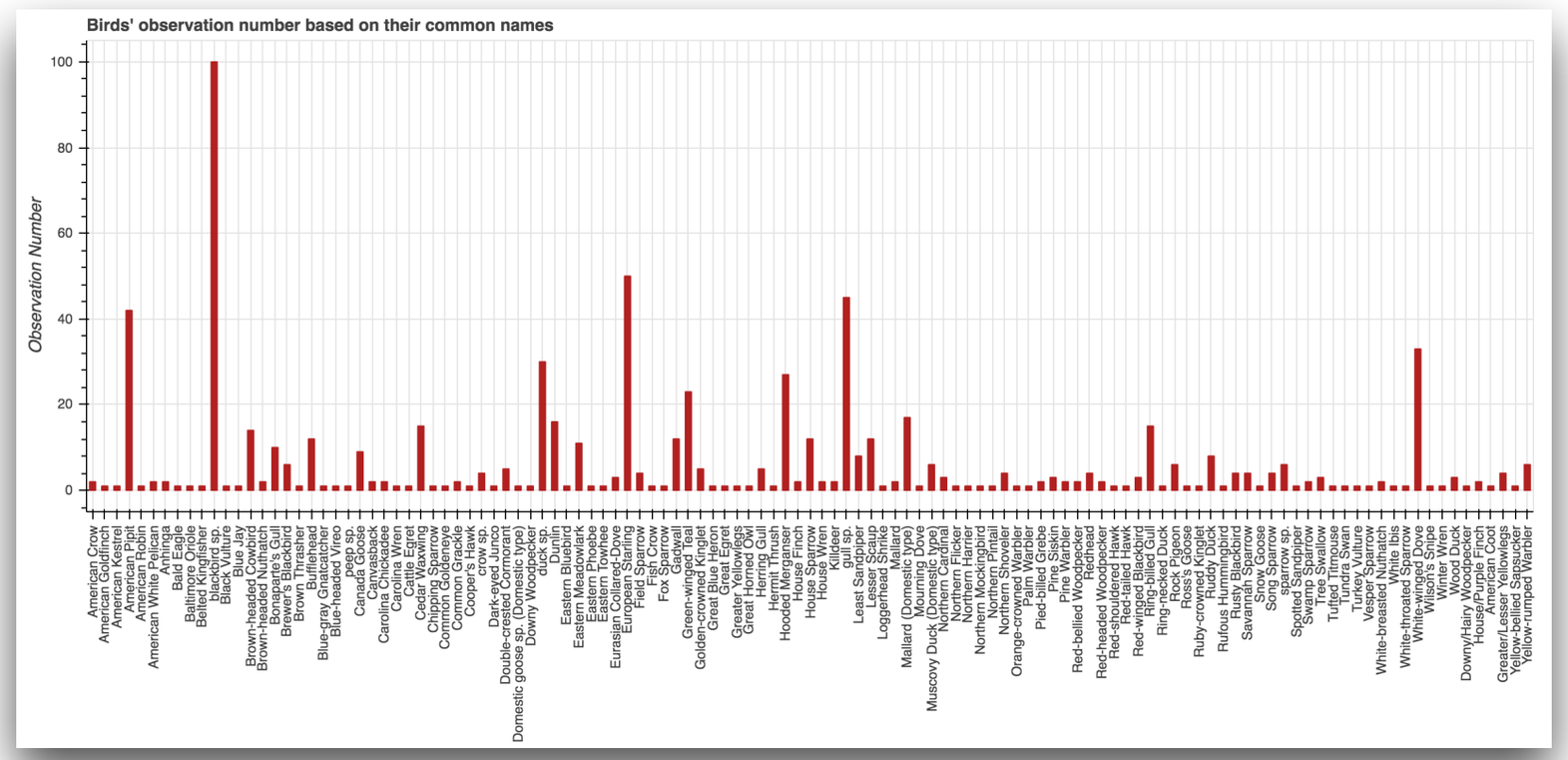
Instructions
Data Visualization

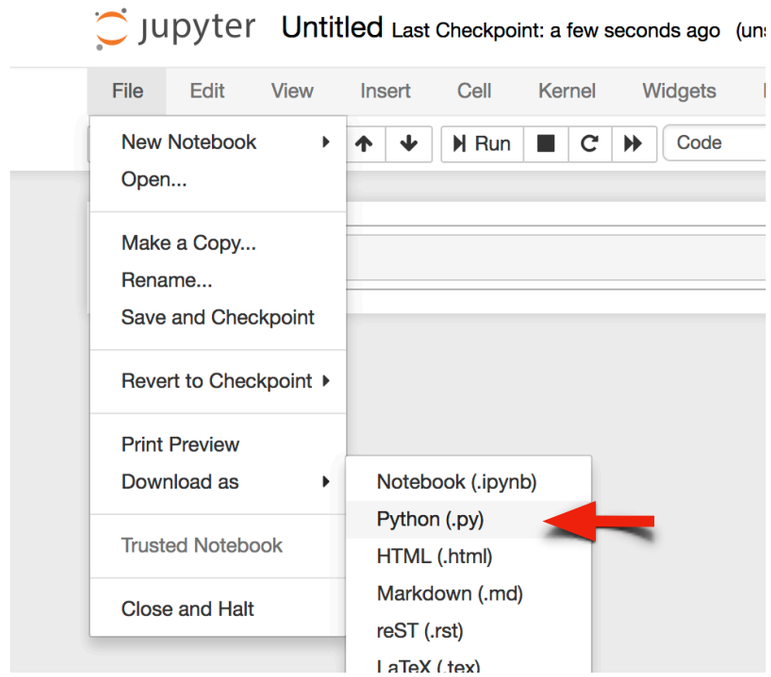
- Visualize the observation number based on the common names of the birds
  - construct a **ColumnDataSource** from extracted data
  - create the plot and specify the width, height, x_range, axis label and title using **figure()**.
  - add hovering tooltip using **HoverTool()**
  - save the plot into a html file using **output_file()**

Instructions

Data Visualization

- Final result should be something **similar** as shown below
  - Detailed requirements and hints are described in the “exercise assignment” document
  - Recommended IDE: Jupyter notebook (friendly for debugging and step by step implementation and visualizing the intermediate outputs)
  - **DO NOT** submit the .ipynb file from Jupyter notebook, save it as .py





VISUALIZATIONAND MULTIMEDIALAB

Instructions
Code Skeleton

- Code provided
  - ***ALL*** necessary libraries are provided
  - Reference links and hints for each task
- Code missing
  - Data cleaning mini-tasks
  - Plotting function for the data visualization
  - Hovering tool for the plot
  - Visualization output

Instructions
Do It Yourself

- Try to get familiar with python coding style
- Read carefully the instructions in the skeleton and finish the missing parts
- Write proper comments for your code
- Use forum to ask and/or answer questions
- Make sure to produce desired results with Bug-Free code
- Submit it before deadline

Python in Practice

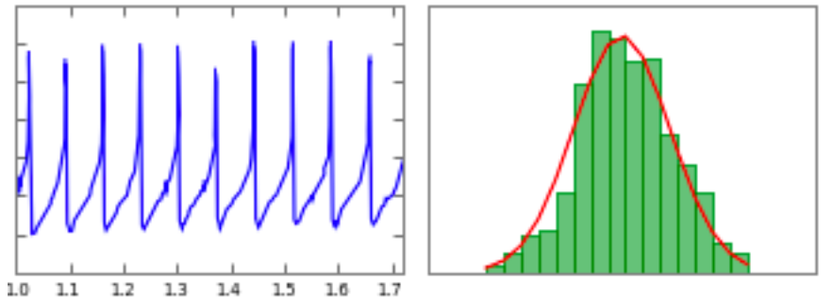
Python in Practice
Preliminaries

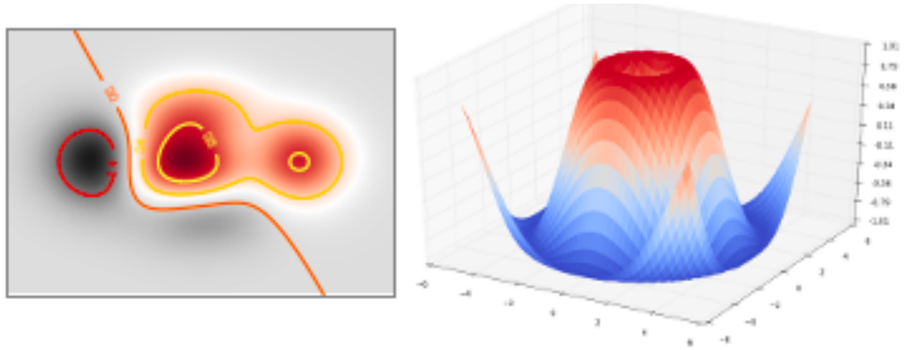
- Why Python?
- Essential Python libraries & Components
  - NumPy
  - Pandas
  - Matplotlib
  - IPython
  - SciPy
  - Bokeh
- Installation & Setup
- IDEs

Python in Practice

Why Python

- Proper for scientific computing
  - The scientist's needs: get data, process data, visualize & communicate with results
- Very rich scientific computing and data analysis libraries
  - Python Package Index (PyPI) hosts thousands of third-party modules
- Friendly & easy to learn
- Many libraries for other tasks
  - e.g. DB management, web applications, serial port access, etc.
- Free and open-source
- Capabilities for 2D & 3D visualization





Python in Practice

Essential Python Libraries & Components

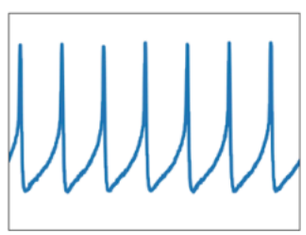
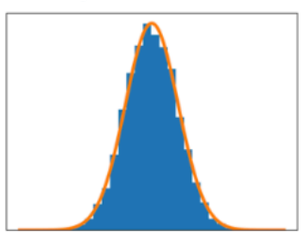
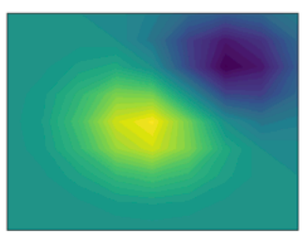
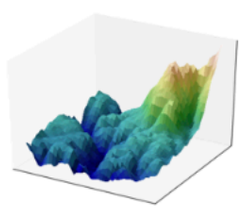
- NumPy (Numerical Python)
  - Scientific computing & data analysis
    - Multidimensional array processing, read/write datasets to disk, linear algebra operations, random number generation , code integration, etc.
  - Basis for other libraries

- Pandas
  - Provides rich data structures and functions for data analysis
    - e.g. *DataFrame* object for 2D tabular and column-oriented data
  - Combined with NumPy
  - Many functionalities
    - e.g. time series, DB managements, etc.

Python in Practice

Essential Python Libraries & Components

- Matplotlib
  - Library for producing “publication-ready” plots and other 2D data visualizations
  - Integrates well with IPython and provides user interactivity

- IPython
  - An advanced Python shell
  - Provides an environment for interactive and exploratory computing
  - Jupyter Notebook (web-based interactive computational environment for creating, executing, and visualizing Jupyter Notebooks)

- SciPy
  - a collection of packages addressing a number of different standard problems in scientific computing (e.g. optimization, regression, interpolation, etc.)

Python in Practice

Essential Python Libraries & Components

- Bokeh
  - Bokeh enables high-performance interactive visualizations of large datasets in modern web browsers
  - Different interfaces:
    - Bokeh.plotting
    - Bokeh.models
    - Bokeh.layout

VISUALIZATIONAND MULTIMEDIALAB

Python in Practice
Installation & Setup

- Get a complete scientific Python environment
  - e.g. Anaconda (https://www.anaconda.com/)
    - works for Win, Mac, Linux and includes Python IDE, IPython and more

- Once conda is installed in your system, download the *environment.yml* file and run the following commands:
  - For Jupyter Notebook, run the following commands:

    $ conda env create -f environment.yml

    $ conda env list
  - For other missing libraries that you need later on, install them using **conda** or **pip**

- Working with the new environment named “dv2019”
  - For Jupyter Notebook, run the following commands:

    $ source activate dv2019 *# to activate the environment*

    $ jupyter notebook *# to launch jupyter notebook interface inside the current folder*
  - In Pycharm, go to Pycharm/Preference —> Project Interpreter —> add conda environment

Python in Practice
IDEs

- Many Python IDEs
  - e.g. VS Code (with Python extension), Spyder, PyCharm, Sublime Text, etc.
- Recommended: PyCharm Community Edition (free)

- Browser-based Jupyter Notebook
  - Jupyter is an evolution of IPython notebooks (http://jupyter.org/)
  - Web-based application to build interactive computational notebook

Python in Practice
Reading/Writing Files

- In Python, basically you don’t need to import any library to read and write files

- However, to handle dataset more efficiently, we recommend the IO Tools of Pandas, and use the absolute path for the input file(s)
  - In []:
```
import pandas as pd
__file__ = ‘example.csv'
my_absolute_dirpath = os.path.abspath(os.path.dirname(__file__))
data = pd.read_csv(os.path.join(my_absolute_dirpath,__file__))
```

- pandas.read_csv(‘filename’, sep=‘’, lineterminator=‘’, error_bad_lines=False)
  - return: a dataframe which represent the data as a structured ….

Python in Practice

Data: First Impression

- dataframe.head()
  - shows the first five rows of the dataframe
  - alternatively: dataframe.tail(), which will show the last five rows of the dataframe
- df.describe()

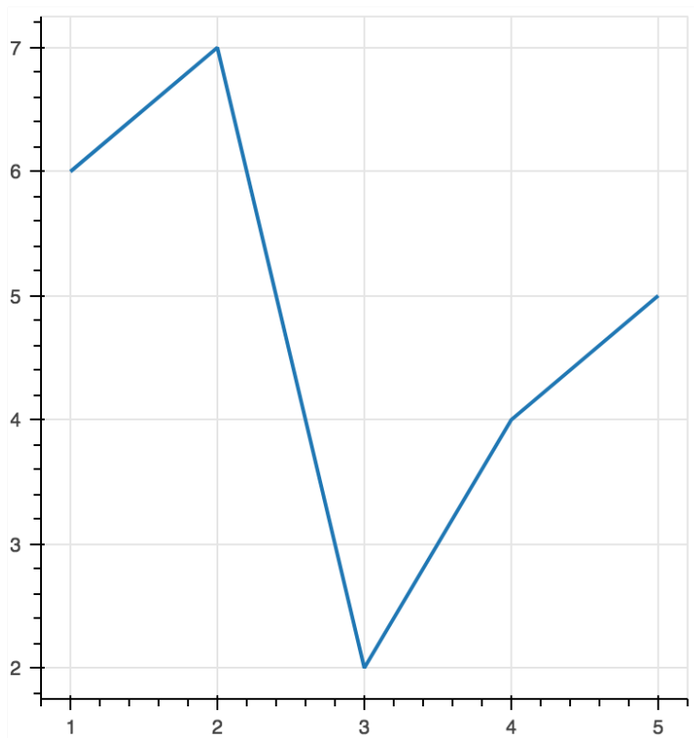
	TAXONOMIC ORDER	OBSERVATION COUNT	IBA CODE	BCR CODE	USFWS CODE	ATLAS BLOCK	LATITUDE	LONGITUDE	DURATION MINUTES	EFFORT DISTANCE KM
count	1939.000000	1939.000000	0.0	1939.0	0.0	0.0	1939.000000	1939.000000	1907.000000	1245.000000
mean	16351.371187	17.193914	NaN	27.0	NaN	NaN	32.324641	-86.255219	33.413214	2.159374
std	11925.127382	134.346903	NaN	0.0	NaN	NaN	0.069665	0.108199	60.105680	3.428520
min	254.000000	1.000000	NaN	27.0	NaN	NaN	32.033745	-86.399918	1.000000	0.112000
25%	2959.000000	1.000000	NaN	27.0	NaN	NaN	32.262419	-86.345224	10.000000	0.692000
50%	19073.000000	2.000000	NaN	27.0	NaN	NaN	32.345463	-86.289228	19.000000	1.174000
75%	27939.000000	6.000000	NaN	27.0	NaN	NaN	32.380433	-86.229222	35.000000	1.963000
max	31345.000000	3000.000000	NaN	27.0	NaN	NaN	32.440901	-85.995435	420.000000	24.140000

Python in Practice
Data Cleaning

- data filtering by matching the attribute name
  - dataframe[dataframe[‘column_cell_name’] == ‘attribute_name’]
- reset the index into {0,1,2,3, …} after data filtering
  - dataframe.reset_index()
- data filtering by matching multiple conditions
  - dataframe.loc[(condition1) & (condition2) & …]
- data deleting
  - dataframe.drop([‘attribute_name1’, …], axis=1)
  - axis=1, deleting columns
  - axis=0, deleting rows

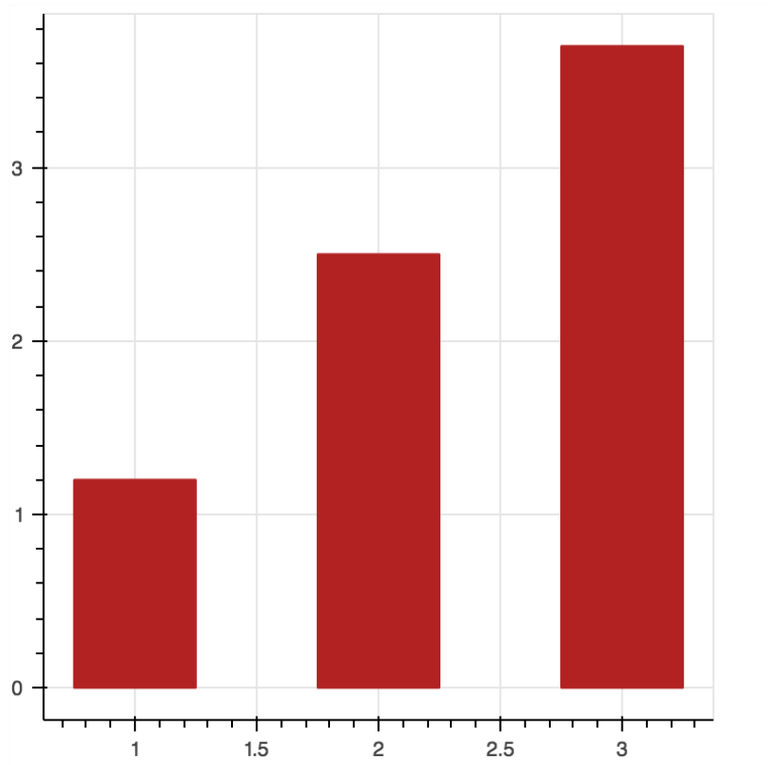
Python in Practice
Bokeh Plotting

- Single lines (using bokeh line() function)
  - In []:
```
from bokeh.plotting import figure, show

p = figure(plot_width=400, plot_height=400)

# add a line renderer
p.line([1, 2, 3, 4, 5], [6, 7, 2, 4, 5], line_width=2)
show(p)
```



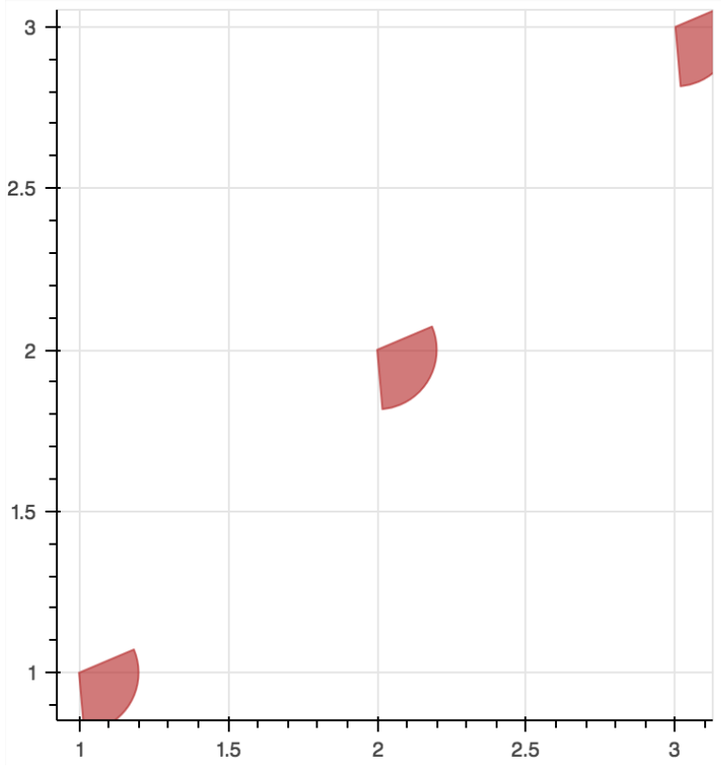
Python in Practice
Bokeh Plotting

- Plot bars (function vbar() or hbar())
  - In []:
```
from bokeh.plotting import figure, show
p = figure(plot_width=400, plot_height=400)
p.vbar(x=[1, 2, 3], width=0.5, bottom=0,
       top=[1.2, 2.5, 3.7], color="firebrick")
show(p)
```



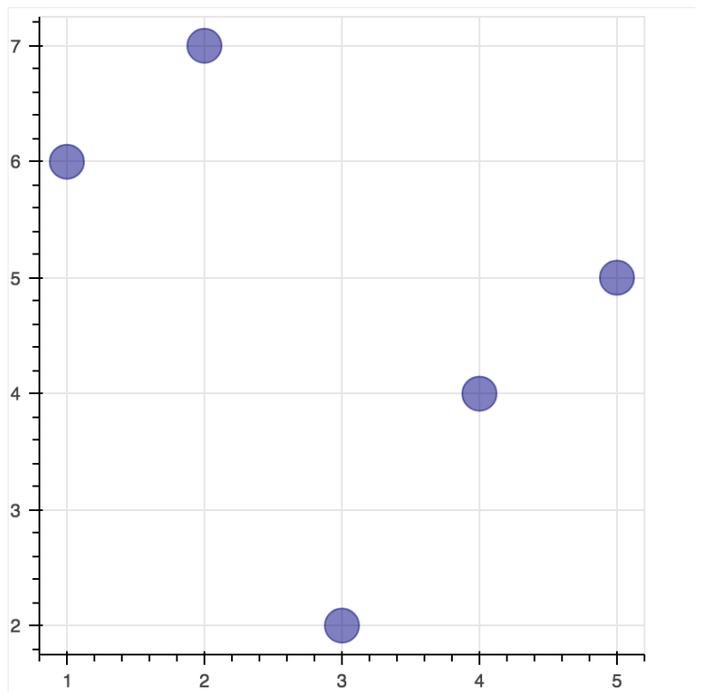
Python in Practice
Bokeh Plotting

- Plot wedges
  - In []:
```
from bokeh.plotting import figure, show
p = figure(plot_width=400, plot_height=400)
p.wedge(x=[1, 2, 3], y=[1, 2, 3], radius=0.2, start_angle=0.4,
end_angle=4.8, color="firebrick", alpha=0.6, direction="clock")
show(p)
```



Python in Practice
Bokeh Plotting

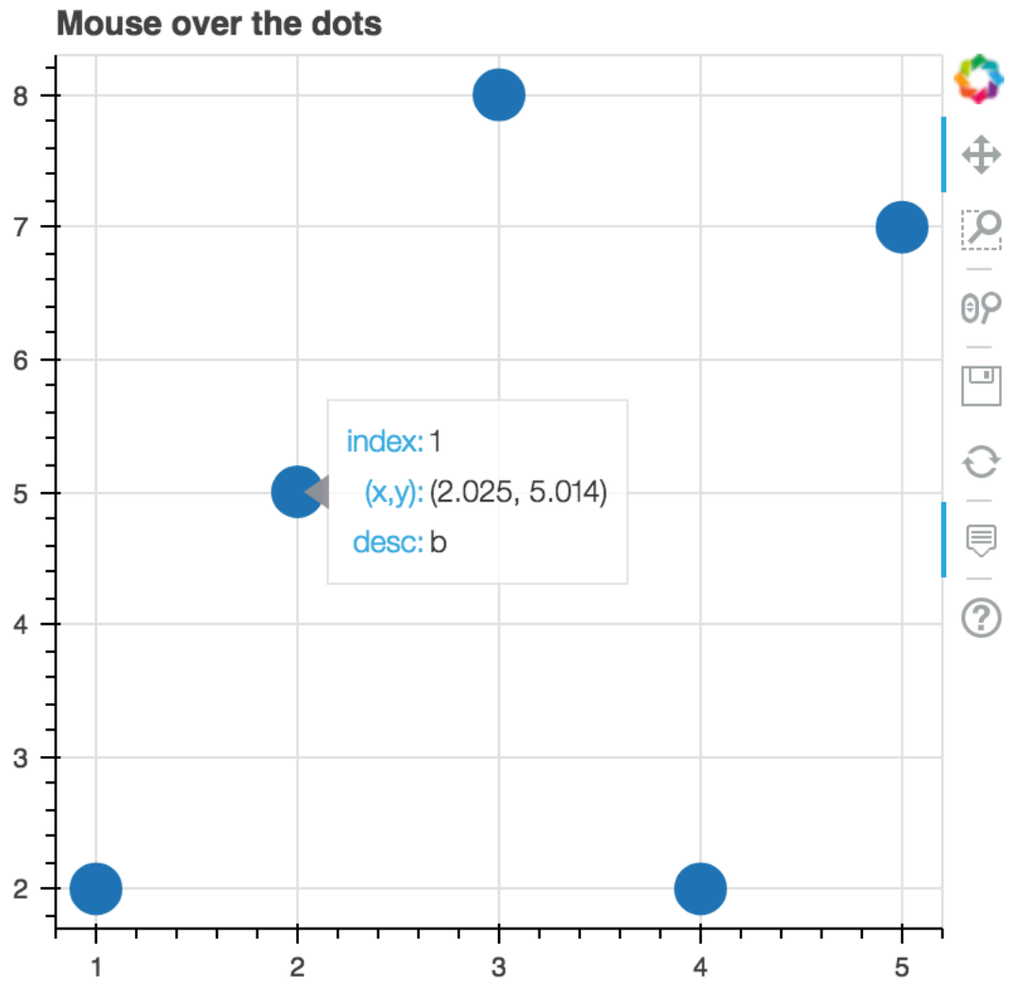
- Plot scatter markers
  - In []:
```
from bokeh.plotting import figure, show

p = figure(plot_width=400, plot_height=400)

# add a circle renderer with a size, color, and alpha
p.circle([1, 2, 3, 4, 5], [6, 7, 2, 4, 5], size=20,
color="navy", alpha=0.5)

show(p)
```



Python in Practice
Bokeh Tooltips

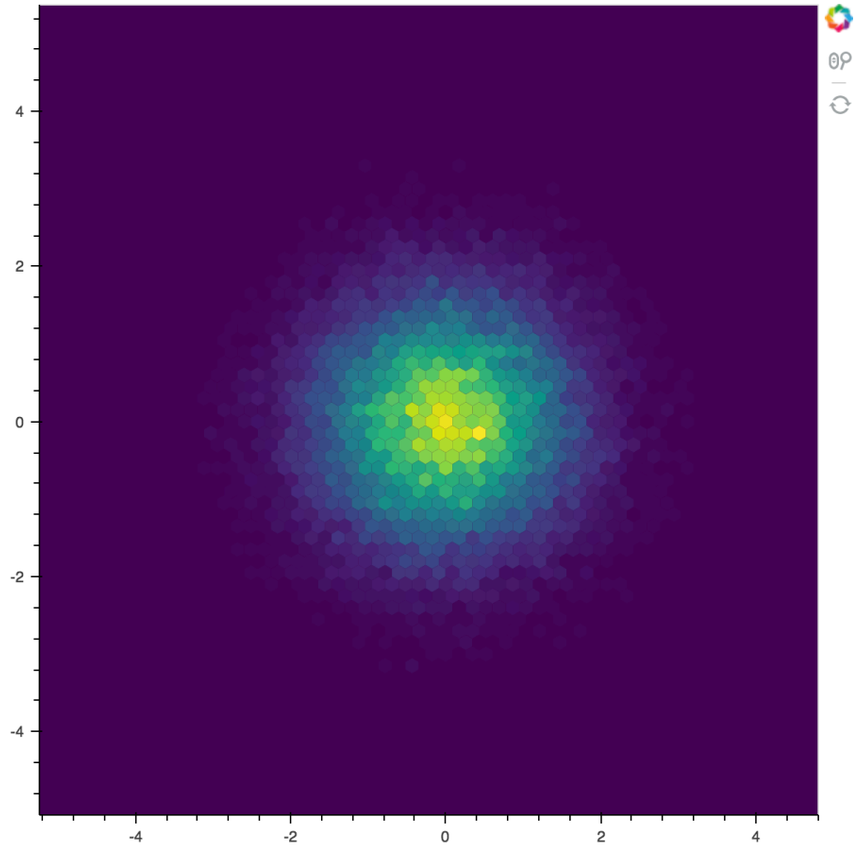
- Add a HoverTool
  - `TOOLTIPS = [`

```
    ("index", “$index"),
    ("(x,y)", "($x, $y)”),
    ("desc", "@desc"),
]
```



Python in Practice
Bokeh Plots

- More plots can be found in:
  - http://bokeh.pydata.org/en/latest/docs/user_guide/plotting.html
  - https://bokeh.pydata.org/en/latest/docs/user_guide/categorical.html
  - http://bokeh.pydata.org/en/latest/docs/user_guide/tools.html



Python in Practice
More Tutorials

- Python 3 Tutorial (programiz.com/python-programming/tutorial)
  - With interactive live demos
- LearnPython.org interactive Python tutorial (https://www.learnpython.org)
  - With more details and exercises
- Python documentation (https://docs.python.org/3/)
  - The official Python documentation
  - With tutorial and much more

Thank you