Wizualizacja danych w Python

Skrót linku do tego pliku z materiałami:

- http://bit.ly/2BdPtIK

Linki do repozytoriów z kodem:

- https://github.com/ksopyla/numpy-pandas-tutorial
- https://github.com/ksopyla/Matplotlib_examples

Link do pliku z pytaniami od uczestników kursu:

- https://docs.google.com/document/d/1CY5NasdJpVfGIBQ5MqlGUJiEXMQrYAdxpX7SbBCE208/edit?usp=sharing

Warm up

- Pomysł na sposób wizualizacji jest równie ważny
  https://www.ted.com/talks/hans_rosling_shows_the_best_stats_you_ve_ever_seen

- Kontekst w wizualizacji danych jest ważny
  https://www.ted.com/talks/david_mccandless_the_beauty_of_data_visualization?language=pl

- Wydatki domowe na przestrzeni lat
  https://flowingdata.com/2015/04/02/how-we-spend-our-money-a-breakdown/
- Najgorszy wykres roku 2019 (i poprzednich)
  http://smarterpoland.pl/index.php/2019/12/najgorszy-wykres-2019/
- Najlepsze wizualizacje 2019
  https://flowingdata.com/2019/12/19/best-data-visualization-projects-of-2019/

Materiały dodatkowe:

- Blog o analizie i wizualizacji danych http://szychtawdanych.pl/
- About Data - blog o uczeniu maszynowym https://ksopyla.com
- Tutorial 10 minutes to pandas
  http://pandas.pydata.org/pandas-docs/stable/10min.html

- Pandas cookbook - z przykładami jak co zrobić (bardzo polecam)
  http://pandas.pydata.org/pandas-docs/stable/cookbook.html
- Biblioteka do tworzenia wykresów w Python SeaBorn - https://seaborn.pydata.org/

Tworzenie środowiska

Edytor IDE

Zalecanym środowiskiem jest **Visual Studio Code** wraz z pakietem “Python”:

- Jest szybkie i lekkie
- Duża społeczność oraz wiele pakietów
- Ma dobre wsparcie dla Linux/mac/windows
- Natywne wsparcie dla git’a

Popularnym i godnym polecenia IDE jest także **PyCharm:**

- Wieloplatformowy
- Darmowa wersja community oraz jako student można postarać się o licencje professional

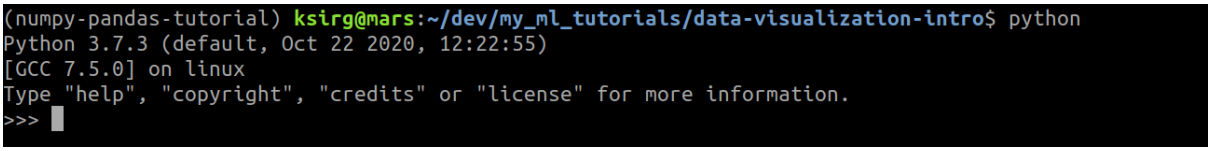
- Dobre wsparcie do intellisense
- Wsparcie dla git'a

Instalacja python

Należy zainstalować min. Python 3.7, dobrze to zrobić z strony https://www.python.org/ dla twojego systemu.
Instalacja ta powinna już zawierać pakiet pip (gdy go nie ma to może to rodzić szereg problemów)
Upewnij się że po zainstalowaniu python możesz w konsoli/terminalu/cmd wpisać

```
python --version
```

I uzyskać w ten sposób numer wersji python (przypominam min. 3.7)
Po wpisaniu samego python, powinien ukazać się



Tworzenie środowiska wirtualnego dla projektu

Polecam jako literaturę dodatkową: http://snoozy.ninja/python/Package-Management/

Pip - instalator pakietów python. Pozwala na automatyczną instalację pakietów opublikowanych w repozytoriach PyPI
Sprawdźcie czy macie go zainstalowanego w systemie (w konsoli/terminalu wpiszcie)

```
>pip --help

Usage:
  pip <command> [options]

Commands:
  install                     Install packages.
  download                    Download packages.
  uninstall                   Uninstall packages.
  freeze                      Output installed packages in requirements
format.
  list                        List installed packages.
  show                        Show information about installed packages.
  check                       Verify installed packages have compatible
```

```
dependencies.
  config                      Manage local and global configuration.
  search                      Search PyPI for packages.
  wheel                       Build wheels from your requirements.
  hash                        Compute hashes of package archives.
  completion                  A helper command used for command
completion.
  help                        Show help for commands.
```

Uwaga! System nie wykrywa zainstalowanego pip!

Wraz z instalacją python'a w systemie pakiet pip powinien być zainstalowany. W rzadkich przypadkach zdarza się, że tak nie jest. Wtedy trzeba go zainstalować oddzielnie.

Linux, MacOs:

- https://pip.pypa.io/en/stable/installing/

Windows:

- Krok po kroku jak zainstalwać pip na Windows https://phoenixnap.com/kb/install-pip-windows

1. Ściągnij plik https://bootstrap.pypa.io/get-pip.py i zapisz go na dysku np. w folderze projketu
2. Otwórz cmd
3. Przejdź w konsoli do folderu w którym jest zapisany plik **get-pip.py**

   `cd c:/Projekty/moj_projekt`
4. Wykonaj skrypy get-pip.py, wpisując w konsoli
   `python get-pip.py`
   `Pip --version`

Aby zainstalować pakiet można wpisać w konsoli wpisać (ale nie rób tego teraz, na instalację pakietów jest lepszy sposób) :

```
pip install numpy
pip install matplotlib
pip install pandas
```

Virtualenv lub venv - narzędzie do tworzenia wirtualnych środowisk z pythonem. Pomaga w utrzymaniu wielu środowisk z różnymi wersjami bibliotek i pakietów. Zazwyczaj tworzymy środowisko per projekt.

```
# tworzymy nowe srodowisko
ksirg@mars:$ cd ~/dev/data-visualization-intro/
ksirg@mars:~/dev/data-visualization-intro$python3 -m venv .venv

#aktywujemy srodowisko, od tego momentu instalacja
#pakietów będzie w ramach środowiska
ksirg@mars:~/dev/data-visualization-intro$. .venv/bin/activate

#deaktywacja srodowiska
ksirg@mars:~/dev/data-visualization-intro$ deactivate
```

Zalecany sposób - Pipenv - pip i venv razem

Jest to obecnie zalecane podejście dla python>3.6. Narzędzie to łączy w sobie pip oraz venv oraz pozwala na rozdzielenie środowiska produkcyjnego oraz dev.

1. Trzeba zainstalować pipenv w systemie
    a. Configure a Pipenv environment (EN) https://www.jetbrains.com/help/pycharm/pipenv.html

Alternatywnym i obecnie zalecanym podejściem jest wykorzystanie pipenv i pliku pipfile

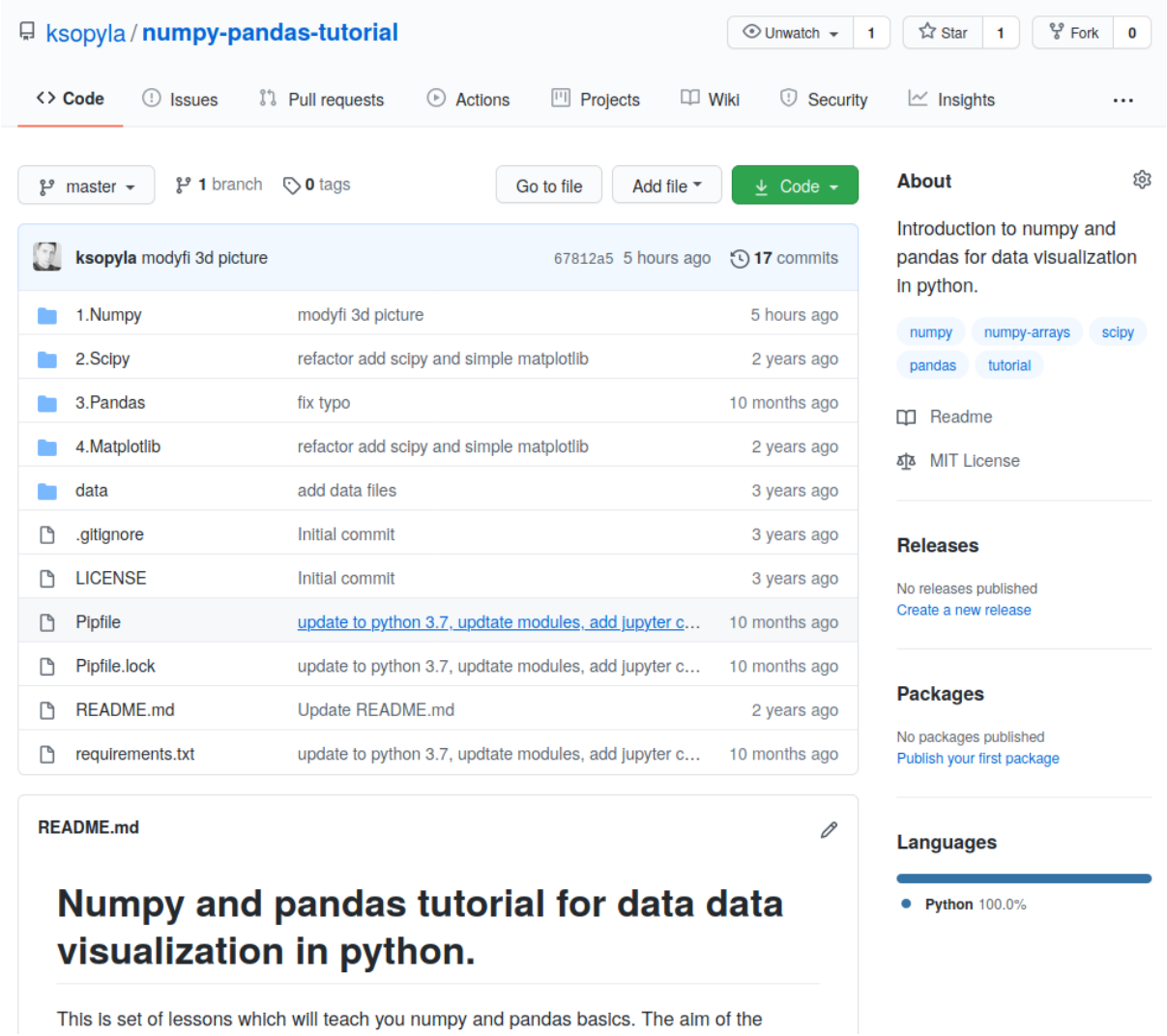
```
pipenv --python 3.7 #stworzy wirtualne środowisko oraz plik pipfile

pipenv install numpy
pipenv install --dev pylint
```

Tworzymy środowisko dla kursu

Zadania oraz kod dostępny jest na

https://github.com/ksopyla/numpy-pandas-tutorial



Całość ma formę plików z kodem python, wraz z interaktywnymi komórkami na podobieństwo jupyter notebook

- VS code - code cells - https://code.visualstudio.com/docs/python/jupyter-support-py#_jupyter-code-cells
- Pychram code cell - https://www.jetbrains.com/help/pycharm/running-jupyter-notebook-cells.html#clear-output

```
git clone https://github.com/ksopyla/numpy-pandas-tutorial.git
cd numpy-pandas-tutorial
pipenv install
```

W ten sposób powinno zostać stworzone środowisko virtualne dla projektu, doinstalowany python oraz niezbędne biblioteki.

Biblioteka NUMPY

NumPy jest biblioteką Pythona służącą do obliczeń naukowych. Obecnie jest standardem i fundamentem dla innych bibliotek, szczególnie typy danych takie jak array (macierze). Wprowadza uproszczony i bardzo kompletny interfejs do operacji na macierzach.
NumPy dostarcza listę funkcji użytecznych w takich zagadnieniach jak:

- algebra liniowa: wektory, macierze, obliczanie wyznaczników, dodawanie macierzy, wartości własne i wiele innych operacji macierzowych
- transformacje Fouriera,
- generowanie liczb losowych,
- I wiele innych

Polecane materiały:

- http://cs231n.github.io/python-numpy-tutorial/

1. Tworzenie macierzy i wektorów

W tym module zapoznamy się z typem danych `array` jest on fundamentem do wszystkich operacji numerycznych. Macierze (1D, 2D, 3D) są wykorzystywane w wielu dziedzinach: obliczenia numeryczne, przetwarzanie sygnałów, przetwarzanie obrazów, przetwarzanie tekstu, sztuczna inteligencja, grafika komputerowe itp.

W ramach zadań zaznajomimy się z podstawowymi funkcjami do budowy wektorów oraz macierzy.

Numpy zapewnia funkcje do tworzenia:

- Wektor też ~~człowiek~~ macierz - wektor kolumnowy, wierszowy
  - `np.arange, np.linspace`
- Macierz zerowa: `np.zeros`
- Macierz jedynkowa `np.ones`
- Macierz diagonalna np.eye
- Macierz losowa `np.random.random`

Rozpoczynając pracę z numpy należy zaimportować pakiet

```
import numpy as np
```

Skrót `np` jest przyjętym standardem.

Proszę otworzyć plik i prześledzić poszczególne przykłady z pliku:

"1.Numpy/1.vectors_arrays.py"

2. Dostęp do elementów macierzy

Ogromną siłą numpy jest szeroki zakres funkcji i ułatwień pozwalających na dostęp do elementów macierzy, w biblioteka wspiera:

- wybór poszczególnych wierszy i kolumn
- Poprzez wykorzystanie operatora zakresu **":"** a[1:5] - możemy wybrać podzbiór wierszy lub kolumn
- Indeksowanie przy pomocy tablic liczb całkowitych (integer indexing). Na podstawie tablicy indeksów można wybrać wiersze lub kolumny z innej tablicy
- Indeksowanie przy pomocy tablic wartości boolowskich (boolean indexing). Można w ten sposób wybrać wiersze lub kolumny spełniające zadane

Proszę otworzyć plik i prześledzić poszczególne przykłady z pliku:
"1.Numpy/2.matrix_element_access.py"

3. Operacje na macierzach

W tym module zapoznamy się z podstawowymi operacjami macierzowymi oraz typami danych trzymanych w macierzach.

Numpy pozwala na intuicyjne:

- Dodawanie macierzy
- Dodawanie wektora do macierzy
- Mnożenie macierzy
- Mnożenie wektorów
- Transpozycję macierzy
- Obliczanie wartości funkcji dla całych tablic: sin, cos, exp, power, sqrt itp.

z wykorzystaniem operatorów oraz funkcji.
Lista funkcji w numpy https://numpy.org/doc/stable/reference/routines.math.html

Proszę otworzyć plik i prześledzić poszczególne przykłady z pliku:

1.Numpy/3.matrix_math.py”

4. Manipulacja macierzami

W tym module przyjrzymy się, w jaki sposób macierze są ułożone. Jak zmieniać ich rozmiary, jak scalać wraz z innymi macierzami oraz w jaki sposób dzielić.
Etap ten jest bardzo istotny z uwagi na częste manipulacje wynikające ze scalania wyników lub przekształcania macierzy na potrzeby innych bibliotek.

W szczególności zapoznamy się z:

- Ułożenie elementów (row, column order)
- Transpozycja
- Zmiana wymiarów macierzy (reshape)
- Scalanie macierzy

Proszę otworzyć plik i prześledzić poszczególne przykłady z pliku:

Plik “1.Numpy/4.matrix_manipulation.py”

5. Rozgłaszanie w numpy (broadcasting)

W tym module zapoznamy się z metodami rozgłaszania wyników i “wnioskowania” przez numpy o docelowych rozmiarach wektorów i macierzy. Broadcasting pozwala na pracę z macierzami o różnych rozmiarach podczas dokonywania obliczeń. Często mamy mniejszą macierz, której wartości chcemy “przyłożyć” do dużej wielokrotnie. Operacje te pojawiają się w filtrowaniu sygnału np. Operacja konwolucji obrazu czy dźwięku.

Odpowiemy sobie na pytania co się stanie gdy:

- Dodamy wektor do macierzy?
- Kiedy możemy wykonać operację na dwóch macierzach o różnych wymiarach?

Proszę otworzyć plik i prześledzić poszczególne przykłady z pliku:
“1.Numpy/5.matrix_vector_broadcasting.py”

Zadania do samodzielnego wykonania

Zad1. Tworzenie tablic

1. Utwórz tablicę liczb od 1 do 100.
2. Utwórz tablicę liczb dodatnich, podzielnych przez 11 do 1000.
3. Utwórz tablice liczb od -5 do 5 składającą się z 70 liczb dzielących ten odcinek na równe części

Zad2.
Utwórz **macierze** a=[1 2 3] i b=[4 5 6] upewnij się co do wymiarów (shape, powinien być 1x3). Oblicz:

- sumę, różnicę, iloczyn, iloraz, transpozycję tablic
- oraz pierwiastki elementów obu tablic.
- a+b',
- a*b,
- a*b',
- a'*b,
- a'*b'

Zad3.
Wykonaj obliczenia:

- Utwórz tablicę sinusów, cosinusów od 0 do pi co 1/10.
- Oblicz wartości sześcianów liczb z tablicy o elementach -100 do 100 co 2, [-100, -98, -96 … 98, 100]
- Oblicz pierwsze 20 potęg liczby 2 2^0, 2^1,...2^19
- Utwórz tablicę liczb, w której na pozycji "i" znajduje się suma elementu "i"+ "i+1" [0+1, 1+2, 2+3, ……...99+100]

Zad4.
Utwórz po 100 wyrazów ciągów z poszczególnych przykładów dla n=1...100 i oblicz ich sumę (nie możesz używać pętli)

- 1/n (wynik = 5.187377517639621)
    - Utwórz wektor o elementach [1/1, 1/2, 1/3 … 1/100]
    - Oblicz sumę elementów tego ciągu
- 1/sqt(n) (wynik = 18.589603824784156)
- (1+1/n)^n (wynik =?)

Zad5.
Utwórz macierz A o wymiarach 7x7 (elementy dowolne)

- Znajdź element z 3 wiersza 4 kolumny
- Podstaw za element z 6 wiersza 3 kolumny liczbę π.
- Utwórz macierz B składającą się z 3 pierwszych wierszy i kolumn od 2 do 6
- Utwórz macierz C – z trzech pierwszych kolumn,
- Utwórz macierz D – z czterech ostatnich wierszy
-

Zad6.
Utwórz tablicę o 36 elementach, następnie zmień ją na macierz o wymiarach:

- 2x18
- 3x12
- 4x9

Zwróć uwagę na ułożenie elementów

Zad7.
Utwórz macierz o wymiarach 100x100 o następującej strukturze
[
[1, 1 ….1]
[2,2, ….2]
…
[100,100, … 100]
]

Zad8.
Stwórz 3 macierze 2D o wymiarach 2x5 o elementach [0..9],[10,..19], [20, ..29]. Połącz je w macierz 3d kolejno wzdłuż osi 0,1,2.

- Wypisz macierz i wybierz z niej elementy [0,0,0], [0,1,1]
- Wypisz rozmiar macierzy
- Policz sumę elementów wzdłuż poszczególnych osi

Biblioteka matplotlib

Jest to najpopularniejsza biblioteka do tworzenia wykresów w python. Wzorowana na bibliotece z matlaba. Pozwala na tworzenie wykresów 2D, 3D manipulację wykresami oraz odpowiednie ich opisywanie.

Przykłady w ramach tej części znajdują się w repozytorium
https://github.com/ksopyla/Matplotlib_examples

Matplotlib gallery

- https://matplotlib.org/stable/gallery/index.html

Biblioteki bazujące na Matplotlib:

- Bokeh https://docs.bokeh.org/en/latest/docs/gallery.html
- Seaborn - https://seaborn.pydata.org/examples/index.html

W tym module nauczysz się:

- Rysować proste funkcje geometryczne na wykresie np. Sin, cos
- Nakładać wykresy funkcji na siebie
- Dodawać atrybuty do wykresów: tytuł, podpisy osi, legendę, punkty na wykresach
- Rysować wiele niezależnych wykresów w ramach jednego okna
- Tworzyć wykresy aktualizujące się w trakcie wykonywania obliczeń
- Tworzyć wykresy słupkowe, kołowe, kropkowe, konturowe itp.
- Tworzyć wykresy 3D

Ściągnij i otwórz repozytorium. Następnie uruchom kolejno skrypty i zapoznaj się z ich treścią.

Rysowanie wykresów

W celu wykreślenia prostego wykresu funkcyjnego, zależności pomiędzy zmienną X a Y wykorzystujemy funkcję *plot.*

```
# import necessary libraries
import matplotlib.pyplot as pl
import numpy as np

n = 256
X = np.linspace(-np.pi, np.pi, n, endpoint=True)
C, S = np.cos(X), np.sin(X)

# create a new figure with dimensions 10x8 inches, set dpi to 80
pl.figure(figsize=(10, 8), dpi=80)

# plot cosine using blue color, line width of 2 px, dotted
pl.plot(X, C, color="blue", linewidth=3, linestyle=":")

# plot sine using red color, line width of 2.5 px, dashed
pl.plot(X, S, color="red",  linewidth=2.5, linestyle="--")
# show plots
pl.show()
```

Przykłady rysowania wykresów oraz określania ich atrybutów, takich jak: tytył, osie, legenda, dodawnie annotacji do wykresów zawarte są w repozytorium w folderze: *“2. Plot features”*

Proszę zwrócić uwagę na wykorzystanie funkcji:

- *plot() - rysuje zależność funkcyjną, wraz z parametrami*
  - *color=””,*
  - *linewidth=2,*

    - *linestyle=":" ,*
    - *marker='o',*
    - *markersize=20*
-
- *pl.xlim(-5, 5) pl.ylim(-1,10) - określenie zakresów osi X i Y*
- *pl.xticks([-1:0.1:1]) - ustalenie znaczników na osi X*
- *pl.yticks([-1, 0, +1]) - ustalenie znaczników na osi Y*
- *pl.legend(loc='upper left') - dodanie legendy wraz z określeniem jej umiejscowienia, uwaga trzeba do funkcji plot ustawić dodatkowy parametr label='cosine'*
- *pl.title("tytył") - ustawia tytuł wykresu*
- *pl.xlabel('nazwa osi X') - ustawia nazwę dla osi X*
- *pl.ylabel('nazwa osi Y') - ustawia nazwę dla osi Y*

Lisa stylów lini:

- *https://matplotlib.org/gallery/lines_bars_and_markers/line_styles_reference.html*
- *https://matplotlib.org/stable/gallery/lines_bars_and_markers/linestyles.html*

Lista markerów:

- https://matplotlib.org/3.1.3/api/markers_api.html
- https://matplotlib.org/3.1.1/gallery/lines_bars_and_markers/marker_reference.html#sphx-glr-gallery-lines-bars-and-markers-marker-reference-py

Lista map kolorów:

- https://matplotlib.org/stable/tutorials/colors/colormaps.html

Rysowanie wielu wykresów w jednym oknie graficznym

Zadania.

Zad1
Narysuj wykresy dla ciągu funkcji, każdy ciąg umieść na jednym wykresie, n=[1,2,3,4,5]:

$$f_n(x) = \sin(nx),\ x \in [-\pi, \pi]$$

$$f_n(x) = \frac{nx}{1+n^5x^2},\ x \in [-2, 2]$$

Dla każdego wykresu wykonaj:

- Nadaj tytuł,
- ustal zakresy dla osi,
- nazwij osie,
- dodaj legendę

Zad2.
W jednym oknie graficznym utwórz 6 wykresów funkcji rozmieszczonych w trzech wierszach i dwóch kolumnach. W pierwszej kolumnie mają znaleźć się funkcje:

$$f(x) = n * cos(x)\ x \in [-2\pi, 2\pi],\ n \in \{1, 2, 3\}$$

A w drugiej funkcje:

$$f(x) = cos(n * x)\ x \in [-2\pi, 2\pi],\ n \in \{1, 2, 3\}$$

Funkcje w pierwszej kolumnie powinny być narysowane kolorem niebieskim a w drugiej zielonym. Użyj funkcji subplot

Zad3.
Narysuj wykresy funkcji 3D:

$$f(x, y) = \sin(x) * \sin(y) * \exp(-x^2 - y^2)\ ,x, y \in [-\pi, \pi]$$

$$f(x,y) = \sqrt{x^2+y^2},\ x,y \in [-8,8]$$

$$f(x,y) = \sin(\sqrt{x^2+y^2}),\ x,y \in [-8,8]$$

$$f(x,y) = \sin(\sqrt{x^2+y^2})/x,\ x,y \in [-8,8]$$

$$f(x,y) = \cos(x*y),\ x,y \in [-2\pi,2\pi]$$

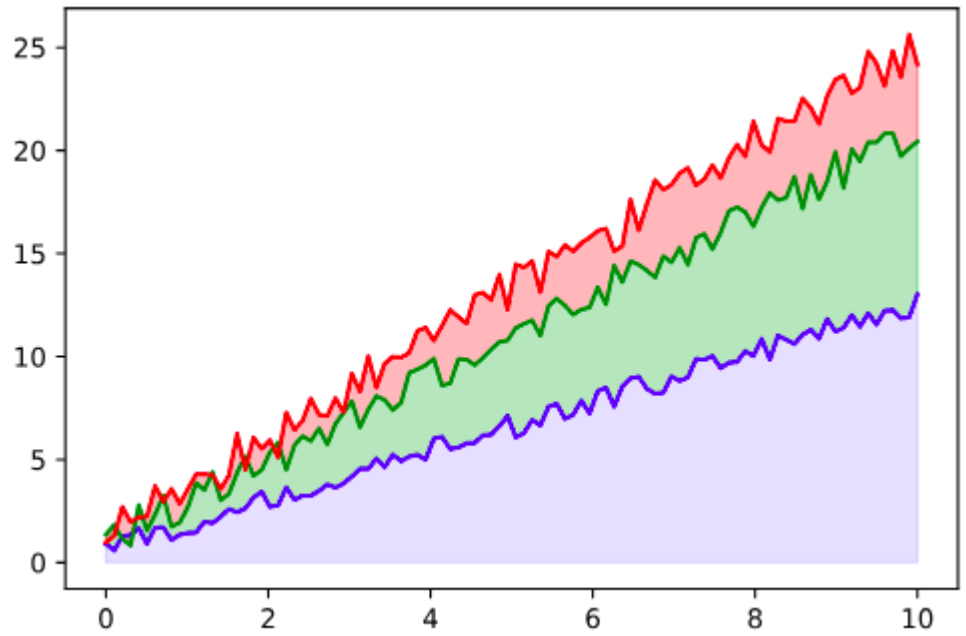
$$f(x,y) = \exp(\sin(x^2+y^2)),\ x,y \in [-8,8]$$

$$f(x,y) = \sin(x)+\cos(y),\ x,y \in [-8,8]$$

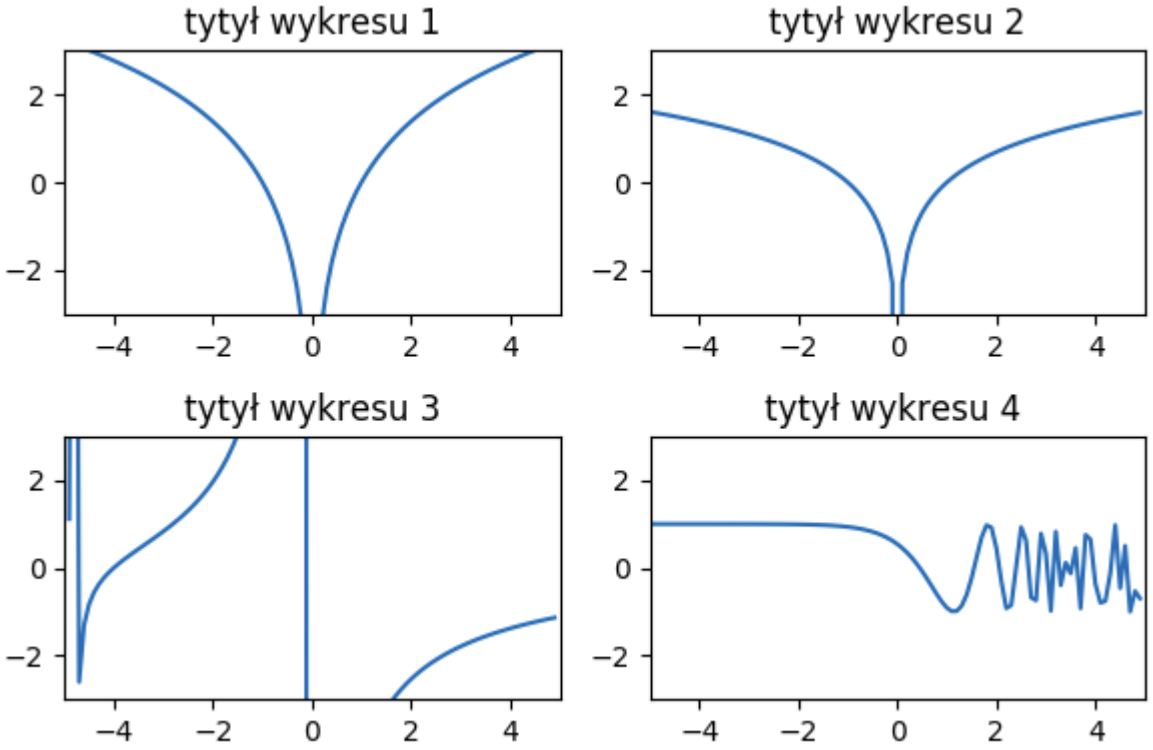
Zad 4.

Wykorzystując funkcję fill_between wygeneruj wykres podobny do poniższego, dla X [0,10]

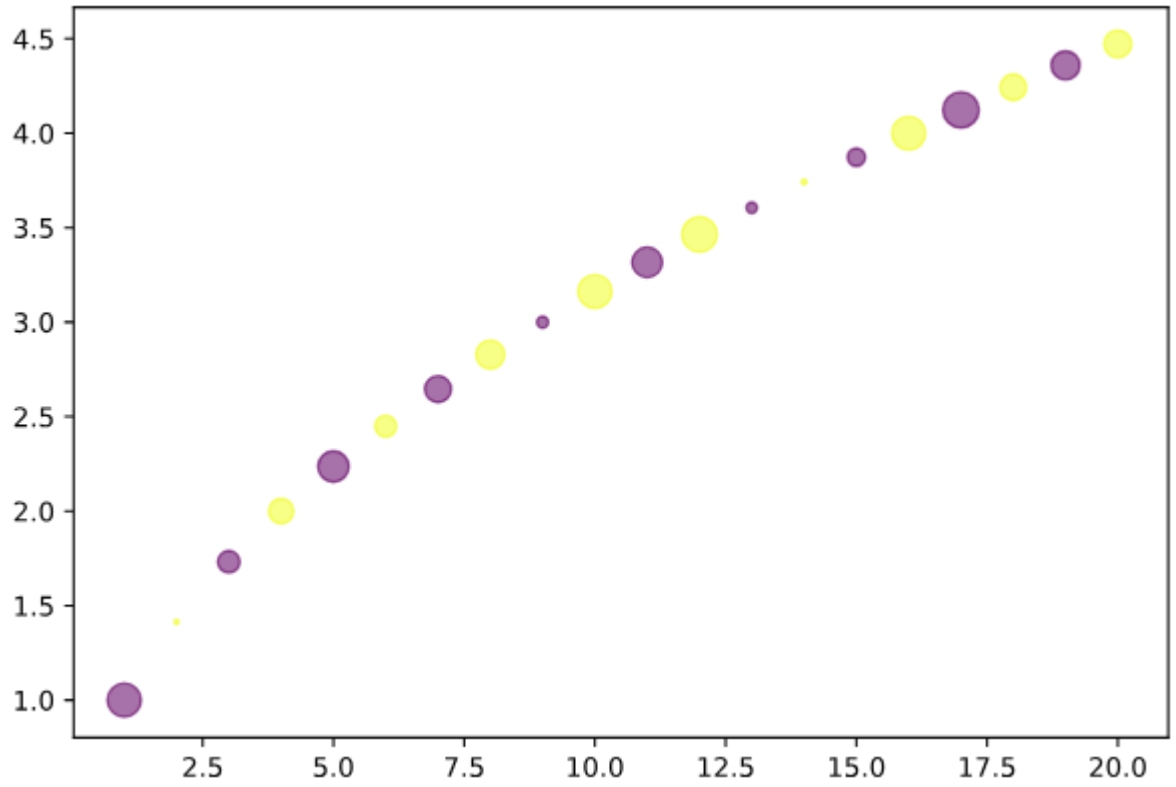
- y1= 1.2x+ random_noise
- y2= 2x+ random_noise
- y3= 2.4x+ random_noise



Zad 5.
Narysuj dowolne wykresy funkcji w 4 pod wykresach



Zad 6.
Narysuj 20 punktów wykorzystując wykres scatter, gdzie punkty są rozmieszczone na wykresie funkcji pierwiastek, wielkość punktu niech będzie losowa a kolory ustalone na przemiennie.



Pandas

Biblioteka Pandas jest open-source'owym narzędziem do analizy danych tabelarycznych. Zawiera funkcje pozwalające operować tablicami danych (DataFrame), ułatwia wczytywanie, czyszczenie oraz wstępną obróbkę danych.

Tutorial 10 minutes to pandas
http://pandas.pydata.org/pandas-docs/stable/10min.html

Pandas cookbook - z przykładami jak co zrobić (bardzo polecam)
http://pandas.pydata.org/pandas-docs/stable/cookbook.html

Przydatne zestawienia

- Date range freq
  https://pandas.pydata.org/pandas-docs/stable/user_guide/timeseries.html#timeseries-offset-aliases

Tworzenie dataframes

Pandas przy przetwarzaniu danych posługuje się 'dataframe', czyli 'ramkami danych'. Można je porównać do tabeli w bazie danych.

Główna klasa obsługująca przechowywanie danych to *DataFrame.* Chcąc się nią posługiwać należy podać tabelę z danymi, wskazać index, czyli co będzie służyło nam za klucz główny oraz nazwy dla kolumn.

Patrzy przykłady z repozytorium **3.Pandas/1.creating_series_dataframes.py**

Selekcja danych z tabel

Mając stworzoną ‘dataframe’ możemy operować na podzbiorze interesujących nas danych. Podobnie jak w tabeli z bazy danych możemy wybierać dane za pomocą funkcji ‘select’

Szczegóły w dokumentacji https://pandas.pydata.org/pandas-docs/stable/indexing.html

Selekcji możemy dokonywać na wiele sposobów:

- Funkcja .loc() - selekcja na podstawie nazwy (indeks lub nazwa kolumny)
- Funkcja .iloc() “integer location” - selekcja na podstawie numerów wierszy
- Indeksowanie boolowskie

Patrzy przykłady z repozytorium **3.Pandas/dataframes_selection.py**

Importowanie danych z plików

Na dzień pisania (13.12.2017) Pandas wspiera 14 formatów danych, z których możemy czytać dane. Najpopularniejsze to CSV, Excel, Json, SQL, HDF5.
Przydatne funkcje:

- read_csv()
- read_table()

Szczegóły w dokumentacji
http://pandas.pydata.org/pandas-docs/stable/io.html

Funkcja *read_csv()* przydatne argumenty:

- filepath_or_buffer - nazwa pliku, adres url lub buffor z danymi
- usecols - określa, które kolumny mamy wczytać
- dtype - określenie typów danych w kolumnach
- skiprows - ile wierszy ma pominąć
- converters - możemy podać funkcję, która dokona konwersji danych z tabeli na nasz format zanim zostaną one wstawione do dataframe’a
- na_values - możemy podać nasz format braku wartości
- chunksize - określa po ile wierszy będzie wczytywanych, przydatne gdy pracujemy z ogromnymi plikami

-

Patrzy przykłady z repozytorium **3.Pandas/reading_files.py**

Łączenie i agregacja danych w Pandas

Jest to bardzo potężna funkcjonalność w Pandas. W ramach tego kursu omijamy.

- merge
- groupby
- aggregate
- pivot_table
- concat

Wykresy w pandas

W pandas obiekt dataframe posiada funkcję *plot*, która pozwala wygenerować wybrany typ wykresu. Oczywiście należy zwrócić uwagę na ułożenia danych w dataframe. Od tego ułożenia zależy czy:

- Wykres uda się wygenerować, pandas może zwrócić błąd że nie może wygenerować wykresu
- Czy na odpowiednich osiach będą dane o które nam chodzi
- Czy pojawią się serie danych, czyli wiele lini(wykresów) na jednym obrazie

Rodzaje wykresów:

- Liniowe
- Słupkowe
- Histogramy

i wiele innych do sprawdzenia w dokumentacji.

Większość wykresów wygenerujemy przy pomocy funkcji *plot*
https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.plot.html

Szczegóły związane z wizualizacją w dokumentacji
http://pandas.pydata.org/pandas-docs/stable/visualization.html

Zad1 GPD per capita krajów ościennych Polski

Wejdź pod adres
http://databank.worldbank.org/data/reports.aspx?source=world-development-indicators
Skorzystaj z pliku **/data/GDP_Poland_neighbours.csv** z repozytoroum

i utwórz plik csv z danymi o produkcie narodowym brutto dla krajów: Polska, Niemcy, Czechy, Ukraina, Francja. Wybierz wskaźniki:

- GDP per capita (current US$),
- GDP per capita growth (annual %),
- GDP (current US$),
- GDP growth (annual %)

1. Wczytaj dane z csv do dataframe.
2. Poszukaj pustych wartości, zobacz jaki mają format, zamień je na ‘nan’. Możesz wykorzystać funkcję *replace*
3. Sprawdź typy danych i jeżeli będą odstępstwa to skonwertuj na poprawne (zwróć uwagę na daty i liczby). Upewnij się że wartości liczbowe mają typ *float* lub pokrewny. Jeżeli nie to dokonaj konwersji, możesz użyć funkcji *apply i pd.to_numeric*
4. Zmień nazwy kolumn na liczby 1990, 1991, itp
5. Policz podstawowe statystyki: średnia, mediana, min, max dla poszczególnych krajów w latach 2000 - 2010

Zad 2

Wygeneruj wykresy, dla danych wczytanych w zadaniu 1.

- Wykres liniowy przedstawiający zmianę GDP growth na jednym wykresie przedstaw wszystkie państwa w latach 1996-2010
- Wykres słupkowy porównujący pkb per capita, dane pogrupowane w seriach dla wszystkich państw w latach 2006-2014
- Wykres liniowy porównujący GDP, dane kolejno sumowane wszystkich państw w latach 2006-2014

Zad3

Napisz skrypt wyświetlający przebieg temperatury dla okolic San Francisco.
Zaimportuj dane z pliku san_francisko_weather.csv dołączonego do repozytorium przedmiotu. Wskazówki:

- Kolumna ZIP wskazuje region - odnajdź w internecie nazwy regionów
- Dodaj legendę z nazwami regionów
- Temperatura jest w stopniach fahrenheita dokonaj konwersji na stopnie Celsjusza
- Na osi X przedstaw tylko miesiące

Zad4.

Na stronie World Banku odnajdź dane na temat:

- Ilość urodzeń przypadających na kobietę (rodzinę)
- Oczekiwaną długość życia
- GPD

dla państw: Chiny, Indie, USA, Francja, UK,Polska, Ukraina, Czechy

Narysuj wykres typu scatter plot przedstawiający zależność pomiędzy ilością urodzeń, średnią długością życia. Każde koło ma symbolizować oddzielne państwo, wielkość koła GDP.
Narysuj wykresy dla lat 1960 1970 1980 1990 2000, 2010