

## Курс “Python для DataScience”

### Практическое задание

#### Инструкция к сдаче

1. Настоятельно рекомендуем сдавать практическое задание в виде ссылки на личный репозиторий на github.
2. Рекомендуемый способ организации данных в репозитории: создать отдельные папки по темам и помещать в них отдельные файлы для каждой задачи с правильным расширением.

Ссылка на инструкцию по работе с git и сдачу практики:

[https://docs.google.com/document/d/1RAT\\_ukE39iOfbz1xa39QXae2hBUEZ4U6Fko\\_wFDdrsM/edit](https://docs.google.com/document/d/1RAT_ukE39iOfbz1xa39QXae2hBUEZ4U6Fko_wFDdrsM/edit)

Ссылка на видеокурс по Git:

<https://geekbrains.ru/courses/66>

Если остались сложности с системой git, то обратитесь к преподавателю или наставнику.

## Тема “Визуализация данных в Matplotlib”

#### Задание 1

Загрузите модуль pyplot библиотеки matplotlib с псевдонимом plt, а также библиотеку numpy с псевдонимом np.

Примените магическую функцию %matplotlib inline для отображения графиков в Jupyter Notebook и настройки конфигурации ноутбука со значением 'svg' для более четкого отображения графиков.

Создайте список под названием x с числами 1, 2, 3, 4, 5, 6, 7 и список y с числами 3.5, 3.8, 4.2, 4.5, 5, 5.5, 7.

С помощью функции plot постройте график, соединяющий линиями точки с горизонтальными координатами из списка x и вертикальными - из списка y.

Затем в следующей ячейке постройте диаграмму рассеяния (другие названия - диаграмма разброса, scatter plot).

#### Задание 2

С помощью функции linspace из библиотеки Numpy создайте массив t из 51 числа от 0 до 10 включительно.

Создайте массив Numpy под названием `f`, содержащий косинусы элементов массива `t`.

Постройте линейную диаграмму, используя массив `t` для координат по горизонтали, а массив `f` - для координат по вертикали. Линия графика должна быть зеленого цвета.

Выведите название диаграммы - 'График  $f(t)$ '. Также добавьте названия для горизонтальной оси - 'Значения  $t$ ' и для вертикальной - 'Значения  $f$ '.

Ограничьте график по оси  $x$  значениями 0.5 и 9.5, а по оси  $y$  - значениями -2.5 и 2.5.

### **\*Задание 3**

С помощью функции `linspace` библиотеки Numpy создайте массив `x` из 51 числа от -3 до 3 включительно.

Создайте массивы `y1`, `y2`, `y3`, `y4` по следующим формулам:

$$y1 = x^2$$

$$y2 = 2 * x + 0.5$$

$$y3 = -3 * x - 1.5$$

$$y4 = \sin(x)$$

Используя функцию `subplots` модуля `matplotlib.pyplot`, создайте объект `matplotlib.figure.Figure` с названием `fig` и массив объектов `Axes` под названием `ax`, причем так, чтобы у вас было 4 отдельных графика в сетке, состоящей из двух строк и двух столбцов. В каждом графике массив `x` используется для координат по горизонтали. В левом верхнем графике для координат по вертикали используйте `y1`, в правом верхнем - `y2`, в левом нижнем - `y3`, в правом нижнем - `y4`. Дайте название графикам: 'График `y1`', 'График `y2`' и т.д.

Для графика в левом верхнем углу установите границы по оси  $x$  от -5 до 5.

Установите размеры фигуры 8 дюймов по горизонтали и 6 дюймов по вертикали.

Вертикальные и горизонтальные зазоры между графиками должны составлять 0.3.

### **\*Задание 4**

В этом задании мы будем работать с датасетом, в котором приведены данные по мошенничеству с кредитными данными: Credit Card Fraud Detection (информация об авторах: Andrea Dal Pozzolo, Olivier Caelen, Reid A. Johnson and Gianluca Bontempi. Calibrating Probability with Undersampling for Unbalanced Classification. In Symposium on Computational Intelligence and Data Mining (CIDM), IEEE, 2015).

Ознакомьтесь с описанием и скачайте датасет `creditcard.csv` с сайта Kaggle.com по ссылке:

<https://www.kaggle.com/mlg-ulb/creditcardfraud>

Данный датасет является примером несбалансированных данных, так как мошеннические операции с картами встречаются реже обычных.

Импортируйте библиотеку Pandas, а также используйте для графиков стиль "fivethirtyeight".

Посчитайте с помощью метода `value_counts` количество наблюдений для каждого значения целевой переменной `Class` и примените к полученным данным метод `plot`, чтобы построить столбчатую диаграмму. Затем постройте такую же диаграмму, используя логарифмический масштаб.

На следующем графике постройте две гистограммы по значениям признака `V1` - одну для мошеннических транзакций (`Class` равен 1) и другую - для обычных (`Class` равен 0). Подберите значение аргумента `density` так, чтобы по вертикали графика было расположено не число наблюдений, а плотность распределения. Число бинов должно равняться 20 для обеих гистограмм, а коэффициент `alpha` сделайте равным 0.5, чтобы гистограммы были полупрозрачными и не загромождали друг друга. Создайте легенду с двумя значениями: "`Class 0`" и "`Class 1`". Гистограмма обычных транзакций должна быть серого цвета, а мошеннических - красного. Горизонтальной оси дайте название "`Class`".

## **\*\*Задание на повторение материала**

1. Создать одномерный массив Numpy под названием `a` из 12 последовательных целых чисел от 12 до 24 неключительно
2. Создать 5 двумерных массивов разной формы из массива `a`. Не использовать в аргументах метода `reshape` число -1.
3. Создать 5 двумерных массивов разной формы из массива `a`. Использовать в аргументах метода `reshape` число -1 (в трех примерах - для обозначения числа столбцов, в двух - для строк).
4. Можно ли массив Numpy, состоящий из одного столбца и 12 строк, назвать одномерным?
5. Создать массив из 3 строк и 4 столбцов, состоящий из случайных чисел с плавающей запятой из нормального распределения со средним, равным 0 и среднеквадратичным отклонением, равным 1.0. Получить из этого массива одномерный массив с таким же атрибутом `size`, как и исходный массив.
6. Создать массив `a`, состоящий из целых чисел, убывающих от 20 до 0 неключительно с интервалом 2.
7. Создать массив `b`, состоящий из 1 строки и 10 столбцов: целых чисел, убывающих от 20 до 1 неключительно с интервалом 2. В чем разница между массивами `a` и `b`?
8. Вертикально соединить массивы `a` и `b`. `a` - двумерный массив из нулей, число строк которого больше 1 и на 1 меньше, чем число строк двумерного массива `b`, состоящего из единиц. Итоговый массив `v` должен иметь атрибут `size`, равный 10.
9. Создать одномерный массив `a`, состоящий из последовательности целых чисел от 0 до 12. Поменять форму этого массива, чтобы получилась матрица `A` (двумерный массив Numpy), состоящая из 4 строк и 3 столбцов. Получить матрицу `At` путем транспонирования матрицы `A`. Получить матрицу `B`, умножив матрицу `A` на матрицу `At` с помощью матричного умножения. Какой размер имеет матрица `B`? Получится ли вычислить обратную матрицу для матрицы `B` и почему?
10. Инициализируйте генератор случайных чисел с помощью объекта `seed`, равного 42.
11. Создайте одномерный массив `c`, составленный из последовательности 16-ти случайных равномерно распределенных целых чисел от 0 до 16 неключительно.

12. Поменяйте его форму так, чтобы получилась квадратная матрица  $C$ . Получите матрицу  $D$ , поэлементно прибавив матрицу  $B$  из предыдущего вопроса к матрице  $C$ , умноженной на 10. Вычислите определитель, ранг и обратную матрицу  $D_{inv}$  для  $D$ .
13. Приравняйте к нулю отрицательные числа в матрице  $D_{inv}$ , а положительные - к единице. Убедитесь, что в матрице  $D_{inv}$  остались только нули и единицы. С помощью функции `numpy.where`, используя матрицу  $D_{inv}$  в качестве маски, а матрицы  $B$  и  $C$  - в качестве источников данных, получите матрицу  $E$  размером  $4 \times 4$ . Элементы матрицы  $E$ , для которых соответствующий элемент матрицы  $D_{inv}$  равен 1, должны быть равны соответствующему элементу матрицы  $B$ , а элементы матрицы  $E$ , для которых соответствующий элемент матрицы  $D_{inv}$  равен 0, должны быть равны соответствующему элементу матрицы  $C$ .