

ЦЕЛЬ И ЗАДАЧИ РАБОТЫ, ТРЕБОВАНИЯ К РЕЗУЛЬТАТАМ ЕЕ ВЫПОЛНЕНИЯ

Целью выполнения лабораторной работы является формирование практических навыков работы с библиотеками `pumpru` и `pandas`, а также применения средств визуализации данных языка Python.

Основными задачами выполнения лабораторной работы являются:

1. Ознакомиться с функциональными возможностями библиотек `pumpru` и `pandas`.
2. Изучить средства визуализации языка Python.

Результатами работы являются:

1. Правильно обработанный массив данных.
2. Построенные графики.
3. Подготовленный отчет.

КРАТКАЯ ХАРАКТЕРИСТИКА ОБЪЕКТА ИЗУЧЕНИЯ, ИССЛЕДОВАНИЯ

Мы живем в мире, в котором наблюдается переизбыток данных. Веб-сайты отслеживают любое нажатие любого пользователя. Смартфоны накапливают сведения о местоположении и скорости в ежедневном и ежесекундном режиме. Умные авто собирают сведения о манерах вождения своих владельцев, умные дома - об образе жизни своих обитателей, а маркетологи – о покупательских привычках. Сам Интернет представляет собой огромный граф знаний, который, среди всего прочего, содержит обширную гипертекстовую энциклопедию, специализированные базы данных о фильмах, музыке, спортивных результатах и много статистических отчетов. В этих данных кроются ответы на бесчисленные вопросы, которые никто даже не думает задавать.

В последнее время для анализа данных все чаще используется язык программирования Python, как в науке, так и коммерческой сфере. Этому способствует простота языка, а также большое разнообразие открытых библиотек. Далее в данном методическом указании будут рассмотрены две из них – это библиотеки `numpy` и `pandas`, а также средства, с помощью которых можно визуализировать обработанные данные.

Библиотека `numpy`.

Numerical Python, или сокращенно NumPy - базовый пакет для высокопроизводительных научных расчетов и анализа данных. Это фундамент, на котором возведены многие высокоуровневые инструменты. Основной функционал библиотеки `numpy` следующий:

- `ndarray`. быстрый и потребляющий мало памяти многомерный массив, предоставляющий векторные арифметические операции.
- стандартные математические функции для выполнения быстрых операций над целыми массивами без явного выписывания циклов.

- средства для чтения массива данных с диска и записи его на диск, а также для работы с проецируемыми на память файлами.
- алгоритмы линейной алгебры, генерация случайных чисел и преобразование Фурье.

- средства для интеграции с кодом, написанным на C/C++ или Fortran

Одна из ключевых особенностей NumPy - объект `ndarray` для представления N-мерного массива; это быстрый и гибкий контейнер для хранения больших наборов данных в Python. Массивы позволяют выполнять математические операции над целыми блоками данных, применяя такой же синтаксис, как для соответствующих операций над скалярами. Ниже представлены основные типы данных библиотеки `numpy` (табл.1):

Функции	Код типа	Описание
<code>int 8. uint8</code>	<code>i1, u1</code>	Знаковое и беззнаковое 8-разрядное (1 байт) целое
<code>int 16. uint16</code>	<code>i2, u2</code>	Знаковое и беззнаковое 16 разрядное (2 байта) целое
<code>int32. uint32</code>	<code>i4, u4</code>	Знаковое и беззнаковое 32-разрядное (4 байта) целое
<code>int64. uint64</code>	<code>i8, u8</code>	Знаковое и беззнаковое 64-разрядное (8 байт) целое
<code>float16</code>	<code>f2</code>	С плавающей таксой половинной точности
<code>float32</code>	<code>f4</code>	Стандартный тип с плавающей точкой одинарной точности Совместим с типом C <code>float</code>
<code>float 64</code>	<code>f8 или d</code>	Стандартный тип с плавающей точкой двойной точности. Совместим с типом C <code>double</code> и с типом

		Python float
float128	f16	С плавающей точкой расширенной точности
complex64 complex128	c8, c16 c32	Комплексные числа, вещественная и мнимая части которых представлены соответственно типами float32, float64 и float128
bool	7	Булев тип, способный хранить значение True и False
object	O	Тип объекта Python
string	S	Тип строки
unicode_	u	Тип Unicode

Табл. 1 Типы данных numpy

Массив ndarray содержит следующие статистические методы (табл.2):

Метод	Описание
sum	Сумма элементов всего массива или вдоль одной оси Для массивов нулевой длины функция вин возвращает 0
mean	Среднее арифметическое Для массивов нулевой длины равно NaN
std, var	Стандартное отклонение и дисперсия, соответственно Может быть задано число степеней свободы (по умолчанию знаменатель равен n)
min, max	Минимум и максимум
argmin, argmax	Индексы минимального и максимального элемента

cumsum	Нарастающая сумма с начальным значением 0
cumprod	Нарастающее произведение с начальным значением 1

Табл. 2 Статистические методы массива ndarray

Кроме того, над массивами можно выполнять теоретико-множественные операции (табл. 3):

Метод	Описание
unique(x)	Вычисляет отсортированное множество уникальных элементов
intersect1d(x, y)	Вычисляет отсортированное множество элементов, общих для x и y
union1d(x, y)	Вычисляет отсортированное объединение элементов
in1d(x, y)	Вычисляет булев массив, показывающий, какие элементы x встречаются в y
setdiff1d(x, y)	Вычисляет разность множеств, т. е. элементы, принадлежащие x, но не принадлежащие y
setxor1d(x, y)	Симметрическая разность множеств; элементы, принадлежащие одному массиву, но не обоим сразу

Табл. 3 Теоретико-множественные операции

Модуль numpy.random дополняет встроенный модуль random функциями, которые генерируют целые массивы случайных чисел с различными распределениями вероятности. Некоторые из этих функций приведены в табл. 4:

Функции	Описание
seed	Задаёт начальное значение генератора случайных чисел
permutation	Возвращает случайную перестановку последовательности диапазона

shuffle	Случайным образом переставляет последовательность на месте
rand	Случайная выборка с равномерным распределением
randint	Случайного выборка целого числа из заданного диапазона
randn	Случайная выборка с нормальным распределением со средним 0 и стандартным отклонением 1 (интерфейс похож на MATLAB)
binomial	Случайная выборка с биномиальным распределением
normal	Случайная выборка с нормальным (гауссовым) распределением
beta	Случайная выборка с бета-распределением
chisquare	Случайная выборка с распределением хи-квадрат
gamma	Случайная выборка с гамма-распределением
uniform	Случайная выборка с равномерным распределением на полуинтервале (0,1)

Табл. 4 Некоторые функции модуля numpy.random

Библиотека pandas

Библиотека pandas содержит высокоуровневые структуры данных и средства манипуляции ими, спроектированные так, чтобы обеспечить простоту и высокую скорость анализа данных на Python. Эта библиотека построена поверх NumPy, потому ей легко пользоваться в приложениях, ориентированных на NumPy.

Библиотека pandas была разработана для того, чтобы обеспечить следующие возможности:

- Структуры данных с помеченными осями, поддерживающие автоматическое или явное выравнивание.
- Встроенная функциональность для работы с временными рядами.
- Гибкая обработка отсутствующих данных.
- Объединение и другие реляционные операции.

Чтобы начать работу с pandas, необходимо освоить две основные структуры данных: Series и Data Frame. Они, конечно, не являются универсальным решением любой задачи, но все же образуют солидную и простую для использования основу большинства приложений.

Series - одномерный похожий на массив объект, содержащий массив данных (любого типа, поддерживаемого NumPy) и ассоциированный с ним массив меток, который называется индексом. Простейший объект Series состоит только из массива данных.

В строковом представлении Series, отображаемом в интерактивном режиме, индекс находится слева, а значения справа. Имея объект Series, можно получить представление самого массива и его индекса можно с помощью атрибутов values, и index соответственно.

Объект DataFrame представляет табличную структуру данных, состоящую из упорядоченной коллекции столбцов, причем типы значений (числовой, строковый, булев и т. д.) в разных столбцах могут различаться. В объекте DataFrame хранятся два индекса: по строкам и по столбцам. Можно считать, что это словарь объектов Series. Внутри объекта данные хранятся в виде одного или нескольких двумерных блоков, а не в виде списка, словаря или еще какой-нибудь коллекции одномерных массивов.

Хотя в DataFrame данные хранятся в двумерном формате, в виде таблицы, нетрудно представить и данные более высокой размерности, если воспользоваться иерархическим индексированием.

Есть много способов сконструировать объект DataFrame, один из самых распространенных - на основе словаря списков одинаковой длины или массивов NumPy.

На первом этапе обработки данные, хранящиеся в объекте pandas, будь то Series, DataFrame или что-то еще, разделяются на группы по

одному или нескольким указанным ключам. Разделение производится вдоль одной оси объекта. Например, в DataFrame можно группировать по строкам (axis=0) или по столбцам (axis=1). Затем к каждой группе применяется некоторая функция, которая порождает новое значение. Наконец, результаты применения всех функций объединяются в результирующий объект. Форма результирующего объекта обычно зависит от того, что именно проделывается с данными.

Ключи группировки могут задаваться по-разному и необязательно должны быть одного типа:

- список или массив значений той же длины, что ось по которой производится группировка;
- значение, определяющее имя столбца объекта DataFrame;
- словарь или объект Series, определяющий соответствие между значениями на оси группировки и именами групп;
- функция, которой передается индекс оси или отдельные метки из этого индекса.

Для группирования данных применяется специальный метод – groupby(). Идея этого метода в том, что объект GroupBy хранит всю информацию, необходимую для последующего применения некоторой операции к каждой группе. Например, чтобы вычислить средние по группам, можно вызвать метод mean объекта GroupBy. Список функций агрегирования приведен в табл.5:

Имя функции	Описание
count	Количество отличных от NA значений в группе
sum	Сумма отличных от NA значений
mean	Среднее отличных от NA значений
median	Медиана отличных от NA значений
std, var	Несмещенное (со знаменателем n-1) стандартное отклонение и дисперсия
min, max	Минимальное и максимальное отличное от NA значение
prod	Произведение отличных от NA значений
first, last	Первое и последнее отличное от NA значение

Табл. 5 Функции агрегирования

СРЕДСТВА ВИЗУАЛИЗАЦИИ PYTHON

Библиотека `matplotlib`

`Matplotlib` - это пакет для построения графиков (главным образом, двумерных) полиграфического качества. При использовании в сочетании с какой-нибудь библиотекой ГИП (например, внутри IPython), `matplotlib` приобретает интерактивные возможности: панорамирование, масштабирование и другие. Этот пакет поддерживает разнообразные системы ГИП во всех операционных системах, а также умеет экспортировать графические данные во всех векторных и растровых форматах: PDF, SVG, JPG, PNG, BMP и т. д.

Для `matplotlib` имеется целый ряд дополнительных библиотек, например, `mplot3d` для построения трехмерных графиков и `basemap` для построения карт и проекций.

Если все настроено правильно, то появится новое окно с линейным графиком. Окно можно закрыть мышью или введя команду `close()`. Все функции `matplotlib` API, в частности `plot` и `close`, находятся в модуле `matplotlib.pyplot`, при импорте которого обычно придерживаются следующего соглашения:

```
import matplotlib.pyplot as plt
```

Графики в `matplotlib` «находятся» внутри объекта рисунка `Figure`. Создать новый рисунок можно методом `plt.figure`:

```
fig=plt.figure()
```

У метода `plt.figure` много параметров, в частности `figsize` гарантирует, что при сохранении рисунка на диске у него будут определенные размер и отношение сторон. Рисунки в `matplotlib` поддерживают схему нумерации (например, `plt.figure(2)`). Для получения ссылки на активный рисунок служит метод `plt.gcf()`.

Активный рисунок можно сохранить в файле методом `plt.savefig`. Этот метод эквивалентен методу экземпляра рисунка `savefig`. Например, чтобы сохранить рисунок в формате SVG, достаточно указать только имя файла:

```
plt.savefig('figpath.svg')
```

Формат выводится из расширения имени файла. Если бы был задан файл с расширением .pdf, то рисунок был бы сохранен в формате PDF. Чтобы получить тот же самый график в формате PNG с минимальным обрамлением и разрешением 400 DPI, нужно было бы написать:

```
plt.savefig('figpath.png'. dpi=400, bbox_inches='tight')
```

В библиотеке pandas также имеется инструмент для того, чтобы построить график из DataFrame. Параметры этого метода приведены в табл. 6:

Аргумент	Описание
subplots	Рисовать график каждого столбца DataFrame в отдельном подграфике
sharex	Если subplots=True, то совместно использовать ось X, объединяя риски и границы
sharey	Если subplots=True, то совместно использовать ось Y
figsize	Размеры создаваемого рисунка в виде кортежа
title	Название графика в виде строки
legend	Помещать в подграфик пояснительную надпись (по умолчанию True)
sort_columns	Строить графики столбцов в алфавитном порядке, по умолчанию используется существующий порядок столбцов

Табл. 6 Параметры функции DataFrame.plot()

Инструмент IPython

Проект IPython в 2001 году основал Фернандо Перес как побочный продукт по ходу создания усовершенствованного интерактивного интерпретатора Python. Впоследствии он превратился в один из самых важных инструментов в арсенале ученых, работающих на Python. Сам по себе он не предлагает ни вычислительных, ни аналитических средств, но изначально спроектирован с целью повысить продуктивность интерактивных вычислений и разработки ПО. В его основе лежит последовательность действий «выполни и посмотри» вместо типичной для многих языков «отредактируй, откомпилируй и запусти». Он также очень тесно интегрирован с оболочкой

операционной системы и с файловой системой. Поскольку анализ данных подразумевает исследовательскую работу, применение метода проб и ошибок и итеративный подход, то IPython почти во всех случаях позволяет ускорить выполнение работы.

Разумеется, сегодняшний IPython - это куда больше, чем просто усовершенствованная интерактивная оболочка Python. В его состав входит развитая графическая консоль с встроенными средствами построения графиков, интерактивный веб-блокнот и облегченный движок для быстрых параллельных вычислений. И подобно многим другим инструментам, созданным программистами для программистов, он настраивается в очень широких пределах.

ПОРЯДОК УСТАНОВКИ БИБЛИОТЕК И ЗАПУСКА IPYTHON

Установка библиотек осуществляется при помощи следующих команд, которые необходимо набрать в терминале PyCharm:

```
conda install -n lab1 -c conda-forge matplotlib
```

```
conda install -n lab1 -c anaconda pandas
```

```
conda install -n lab1 -c anaconda numpy
```

Кроме того, нужно установить библиотеки jupyter и sympy. Для этого необходимо перейти в раздел Project Interpreter и нажать + (Install), после чего ввести названия пакетов.

Запуск IPython осуществляется следующим образом:

1. Создать Jupyter Notebook файла
2. Нажать кнопку запуск (зеленая стрелка)
3. В появившемся окне нажать Cancel
4. На всплывающем сообщении нажать Run Jupyter Notebook
5. Скопировать первую ссылку в окне Run, вставить её в адресную строку браузера и перейти по ней
6. Откроется страница Home, в которой будут отображаться файлы проекта
7. Нажать на созданный Jupyter Notebook файл. В открывшейся странице можно осуществлять написание кода и его запуск
8. То же самое можно осуществлять непосредственно в среде PyCharm

ОБРАЗЕЦ ВЫПОЛНЕНИЯ ЗАДАНИЯ

```
import pandas as pd
import numpy as np
from matplotlib.pyplot import figure, savefig, rc
from mpl_toolkits.mplot3d import axes3d, Axes3D

# считываем CSV файл с данными
apps = pd.read_csv('data/googleplaystore.csv', ",")

# выбираем нужные столбцы
installs = apps[['Genres', 'Installs', 'Android Ver']]

# убираем '+' на конце
installs.Installs = installs.Installs.str[:-1]

# убираем запятые
installs.Installs = installs.Installs.str.replace(',', '',
regex=False)

# преобразуем в int
installs.Installs = pd.to_numeric(installs.Installs,
errors='coerce').fillna(0).astype(int)

# группируем и суммируем скачивания по жанрам
installs_genres = installs.groupby(['Genres'], as_index =
False).sum()

# группируем и суммируем скачивания по версии Android и по жанрам
installs_android_and_genres = installs.groupby(['Genres', 'Android
Ver'], as_index = False).sum()

# получение 5 наибольших и наименьших значений
largest = installs_genres.nlargest(5, 'Installs')
smallest = installs_genres.nsmallest(5, 'Installs')

# построение графиков с помощью pandas.plot() и сохранение в pdf-файл
rc('font', size=90)

largest.plot(x='Genres', y='Installs', figsize=(100, 100),
fontsize=90, linewidth=10.0,
            title="Зависимость скачиваемости приложений от их
жанра")
savefig('largest.pdf')

smallest.plot(x='Genres', y='Installs', figsize=(100, 100),
fontsize=90, linewidth=10.0,
            title="Зависимость скачиваемости приложений от их
жанра")
savefig('smallest.pdf')

# параметры гиперболоида
a = 5
```

```
b = 4
c = 7
```

```
# вычисление значений точек гиперboloида с помощью библиотеки numpy
X = np.arange(-5, 5, 0.05)
Y = np.arange(-5, 5, 0.05)
X, Y = np.meshgrid(X, Y)
Z1 = np.sqrt(c**2/a**2 * X**2 + c**2/b**2 * Y**2 - 1)
Z2 = -np.sqrt(c**2/a**2 * X**2 + c**2/b**2 * Y**2 - 1)
```

```
# построение гиперboloида с помощью matplotlib
rc('font', size=15)
fig = figure()
ax = Axes3D(fig)
ax.plot_surface(X, Y, Z1, linewidth=1)
ax.plot_surface(X, Y, Z2, linewidth=1)
savefig('hyperboloid.pdf')
```

В результате выполнения программы будут построены следующие графики (рис. 1):

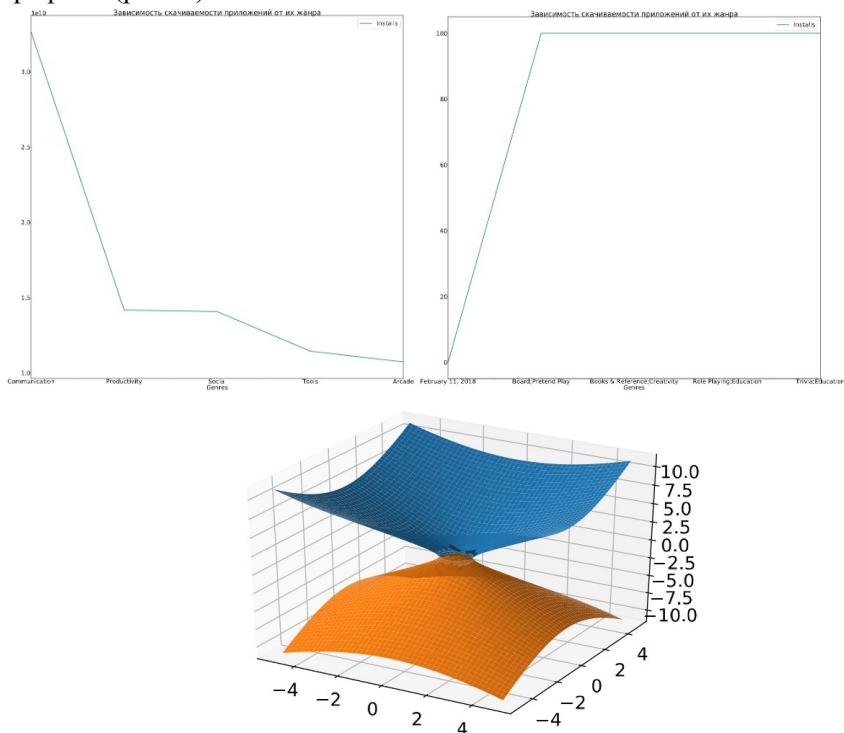


Рис. 1 Примеры графиков

ЗАДАНИЕ НА ЛАБОРАТОРНУЮ РАБОТУ

Для всех вариантов необходимо осуществить обработку данных из файла формата CSV, применяя методы агрегации и групповые операции. Визуализировать эти данные в виде графика зависимости. Построить трехмерную фигуру согласно заданию, указанному в варианте. Файлы для заданий имеют названия, соответствующие варианту.

ТРЕБОВАНИЯ К РЕАЛИЗАЦИИ

В качестве результата работы необходимо построить графики и отобразить их в Jupyter Notebook. Также необходимо сохранить эти графики в pdf файле.

ВАРИАНТЫ ЗАДАНИЙ

Вариант 1

Считать данные из CSV файла в структуру DataFrame. Добавить в DataFrame еще один столбец, в котором содержится отношение числа просмотров к числу загруженных видео. Определить название канала, имеющего оценку B+, который имеет наибольшее число загруженных видео. Построить график зависимости среднего арифметического количества просмотров от рейтинга канала. Вывести результаты для 5 самых популярных каналов и 5 самых непопулярных (по среднему числу просмотров). Построить трехмерную поверхность – куб (размеры куба задать произвольно).

Вариант 2

Считать данные из CSV файла в структуру DataFrame. Добавить в DataFrame еще один столбец, в котором содержится отношение заработной платы футболиста к его стоимости ($K = 1000$, $M = 1000000$). Определить футболиста из Бразилии, имеющего правую рабочую ногу, стоимость которого максимальна. Построить график зависимости среднего общего рейтинга от страны. Вывести

результаты для 5 самых высоких рейтингов и 5 самых низких (по среднему значению). Построить трехмерную поверхность – шар (размеры шара задать произвольно).

Вариант 3

Считать данные из CSV файла в структуру DataFrame. Добавить в DataFrame еще один столбец, в котором содержится количество символов в строке адреса. Определить ID камеры, зафиксировавшую наибольшее число нарушений. Вычислить стандартное отклонение числа нарушений. Построить график зависимости количества нарушений от адреса расположения камеры. Вывести результаты для 5 адресов с самым большим числом нарушений и 5 с самым маленьким. Построить трехмерную поверхность – двуполостный гиперболоид (параметры гиперболоида задать произвольно).

Вариант 4

Считать данные из CSV файла в структуру DataFrame. Добавить в DataFrame еще один столбец, в котором содержится разница (по модулю) между средним возрастом и возрастом каждого из покупателей. Определить ID покупательницы, не моложе 40 лет, которая имеет самый большой доход. Построить график зависимости дохода покупателя (среднее значение) от его возраста. Вывести результаты для 5 покупателей с самым большим средним доходом и 5 с самым маленьким. Построить трехмерную поверхность – прямоугольный параллелепипед (параметры параллелепипеда задать произвольно).

Вариант 5

Считать данные из CSV файла в структуру DataFrame. Добавить в DataFrame еще один столбец, в котором содержится количество символов в строке идентификатора продукта. Определить возрастную группу покупателя (мужчина), который приобрел больше всего товаров (по стоимости). Построить график зависимости потраченных средств покупателей (среднее значение) от возрастной группы.

Вывести результаты для 5 покупателей с самыми большими расходами и 5 с самыми маленькими. Построить трехмерную поверхность – цилиндр (параметры цилиндра задать произвольно).

Вариант 6

Считать данные из CSV файла в структуру DataFrame. Добавить в DataFrame еще один столбец, в котором содержится отношение числа просмотров к числу подписчиков. Определить название канала, имеющего оценку В или В+, который имеет наибольшее число подписчиков. Построить график зависимости среднего арифметического количества загруженных видео от рейтинга канала. Вывести результаты для 5 активных каналов и 5 самых неактивных (по среднему числу загруженных видео). Построить трехмерную поверхность – шар (размеры шара задать произвольно).

Вариант 7

Считать данные из CSV файла в структуру DataFrame. Добавить в DataFrame еще один столбец, в котором содержится отношение заработной платы футболиста к его стоимости ($K = 1000$, $M = 1000000$). Определить футболиста из Аргентины, имеющего общий рейтинг не менее 88, стоимость которого минимальна. Построить график зависимости среднего общего рейтинга от страны. Вывести результаты для 5 самых высоких рейтингов и 5 самых низких (по среднему значению). Построить трехмерную поверхность – куб (размеры куба задать произвольно).

Вариант 8

Считать данные из CSV файла в структуру DataFrame. Добавить в DataFrame еще один столбец, в котором содержится количество символов в строке адреса. Определить ID камеры, зафиксировавшую наибольшее число нарушений 17 января 2019 года. Построить график зависимости количества нарушений от адреса расположения камеры. Вывести результаты для 5 адресов с самым большим числом нарушений и 5 с самым маленьким. Построить трехмерную

поверхность – двуполостный гиперboloид (параметры гиперboloида задать произвольно).

Вариант 9

Считать данные из CSV файла в структуру DataFrame. Добавить в DataFrame еще один столбец, в котором содержится разница (по модулю) между средним возрастом и возрастом каждого из покупателей. Определить ID покупательницы, старше 25 лет и моложе 40 лет, которая имеет самый маленький доход. Построить график зависимости дохода покупателя (среднее значение) от его возраста. Вывести результаты для 5 покупателей с самым большим средним доходом и 5 с самым маленьким. Построить трехмерную поверхность – цилиндр (параметры цилиндра задать произвольно).

Вариант 10

Считать данные из CSV файла в структуру DataFrame. Добавить в DataFrame еще один столбец, в котором содержится количество символов в строке идентификатора продукта. Определить возрастную группу покупательницы, которая приобрела меньше всего товаров (по стоимости). Построить график зависимости потраченных средств покупателей (среднее значение) от возрастной группы. Вывести результаты для 5 покупателей с самыми большими расходами и 5 с самыми маленькими. Построить трехмерную поверхность – прямоугольный параллелепипед (параметры параллелепипеда задать произвольно).

Вариант 11

Считать данные из CSV файла в структуру DataFrame. Добавить в DataFrame еще один столбец, в котором содержится отношение заработной платы футболиста к его стоимости ($K = 1000$, $M = 1000000$). Определить футболиста (-ов) из ФК Chelsea, имеющего (-их) левую рабочую ногу, стоимость которого (-ых) находится в интервале от 50 до 70 млн. евро. Построить график зависимости общего рейтинга футболиста от его зарплаты (по всем данным).

Вывести результаты для 5 самых высоких зарплат и 5 самых низких. Построить трехмерную поверхность – шар (размеры шара задать произвольно).

Вариант 12

Считать данные из CSV файла в структуру DataFrame. Добавить в DataFrame еще один столбец, в котором содержится среднее арифметическое показателей math score, reading score и writing score. Определить студента(-ов), прошедшего курс подготовки к экзаменам, у которого все показатели выше среднего значения math score (для всех студентов). Построить график зависимости среднего арифметического показателей от показателя math score. Вывести результаты для 5 самых высоких значений math score и 5 самых низких. Построить трехмерную поверхность – прямоугольный параллелепипед (параметры параллелепипеда задать произвольно).

Вариант 13

Считать данные из CSV файла в структуру DataFrame. Добавить в DataFrame еще один столбец, в котором содержится отношение стоимости автомобиля к его пробегу (км.). Определить автомобиль(-ли), являющийся гибридом с пробегом более 10000 км, у которого год выпуска старше 2005. Построить график зависимости пробега автомобиля от его стоимости. Вывести результаты для 5 самых дорогих и 5 самых дешевых. Построить трехмерную поверхность – шар (параметры шара задать произвольно).

Вариант 14

Считать данные из CSV файла в структуру DataFrame. Добавить в DataFrame еще один столбец, в котором содержится сумма показателей math score, reading score и writing score. Определить студента(-ов), не прошедшего курс подготовки к экзаменам, у которого среднее арифметическое показателей выше среднего значения writing score (для всех студентов). Построить график зависимости суммы показателей от показателя reading score. Вывести результаты для 5

самых высоких значений reading score и 5 самых низких. Построить трехмерную поверхность – двуполостный гиперboloид (параметры гиперboloида задать произвольно).

Вариант 15

Считать данные из CSV файла в структуру DataFrame. Добавить в DataFrame еще один столбец, в котором содержится отношение стоимости автомобиля к разнице (в годах) между текущей датой и датой выпуска. Определить автомобиль(-ли), потребляющий бензин, с пробегом от 10000 до 40000 км, у которого год выпуска расположен в промежутке от 2006 до 2010. Построить график зависимости пробега автомобиля от его года выпуска. Вывести результаты для 5 самых старых и 5 самых новых автомобилей. Построить трехмерную поверхность – цилиндр (параметры цилиндра задать произвольно).

КОНТРОЛЬНЫЕ ВОПРОСЫ И ЗАДАНИЯ

1. Перечислите основные возможности библиотеки `numpy`.
2. Раскройте достоинства массива `ndarray`.
3. Приведите основные типы данных библиотеки `numpy`.
4. Перечислите и раскройте основные статистические методы массива `ndarray`.
5. Перечислите и раскройте основные теоретико-множественные операции с массивами, реализованные в `numpy`.
6. Опишите назначение модуля `numpy.random`.
7. Перечислите и раскройте основные функции модуля `numpy.random`.
8. Опишите назначение объекта `Series` библиотеки `pandas`.
9. Опишите назначение объекта `DataFrame` библиотеки `pandas`.
10. Приведите функции для чтения данных форматов CSV, реализованные в библиотеке `pandas`.
11. Опишите назначение объекта `GroupBy` библиотеки `pandas`.
12. Перечислите основные функции агрегирования библиотеки `pandas`.
13. Приведите основные возможности библиотеки `matplotlib`.
14. Перечислите основные параметры функции `DataFrame.plot()`.
15. Раскройте назначение инструмента IPython (Jupyter Notebook).

ФОРМА ОТЧЕТА ПО ЛАБОРАТОРНОЙ РАБОТЕ

На выполнение лабораторной работы отводится 1 занятие (2 академических часа: 1 час на выполнение и сдачу лабораторной работы и 1 час на подготовку отчета).

Отчет на защиту предоставляется в печатном виде.

Структура отчета (на отдельном листе(-ах)): титульный лист, формулировка задания, описание процесса выполнения лабораторной работы, результаты выполнения работы, выводы.

ОСНОВНАЯ ЛИТЕРАТУРА

1. Маккинли, Уэс Python и анализ данных / Пер. с англ. Слинкин А.А. - М.: ДМК Пресс, 2015. - 482 с.:ил.
2. Грас, Дж. Data Science. Наука о данных с нуля / Пер. с англ. - СПб.: БХВ -Петербург, 2017. - 336с.: ил.

ДОПОЛНИТЕЛЬНАЯ ЛИТЕРАТУРА

3. Henley, A.J. Learn Data Analysis with Python: Lessons in Coding / A.J. Henley, Dave Wolf ISBN 978-1-4842-3486-0

Электронные ресурсы:

4. Научная электронная библиотека <http://eLIBRARY.RU>
5. Электронно-библиотечная система <http://e.lanbook.com>