**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ**

**Московский государственный технический университет**

**им. Н.Э. Баумана**

**(МГТУ им. Н.Э. Баумана)**

**Кафедра «Информационная безопасность» (ИУ8)**

Лабораторная работа № 1

По дисциплине: «Машинное обучение»

Тема: «Входной контроль»

Выполнил: Веденеев А.А.,

Студент группы ИУ8-92

Проверила: Коннова Н.С.,

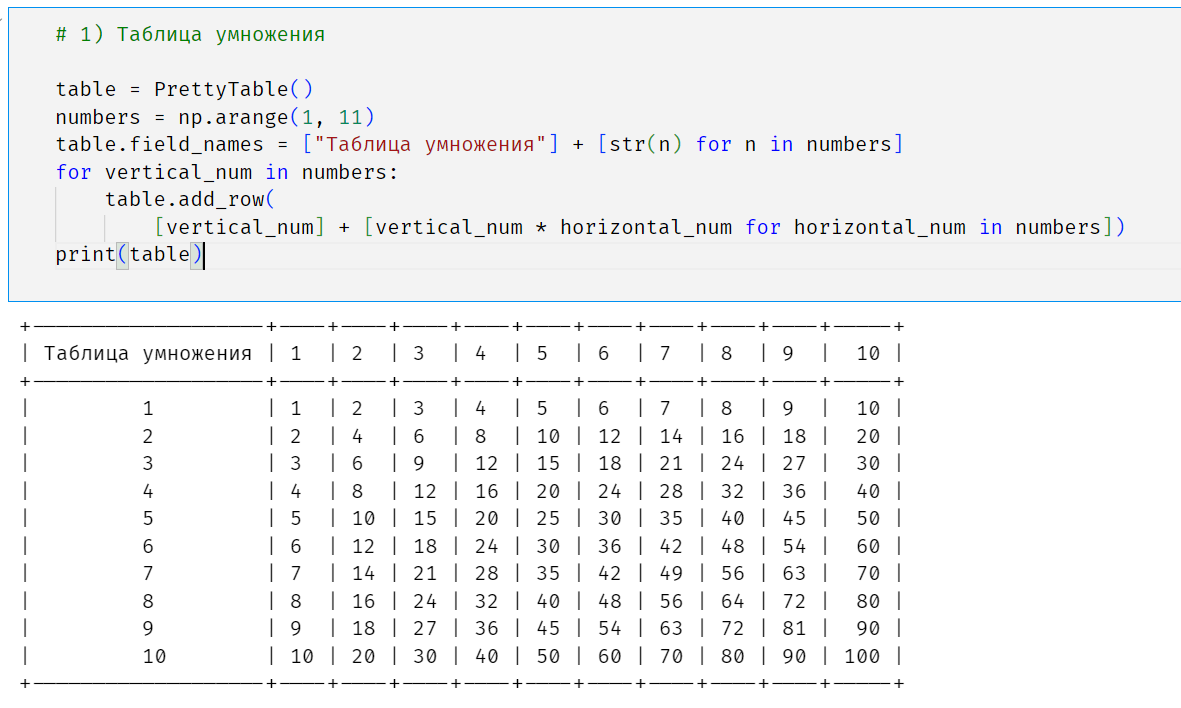
Преподаватель каф. ИУ8

г. Москва 2024 г.

# Практическая часть лабораторной работы 1

**Цель работы:** проверить знания базовых инструментов, которые используются для обращения с данными в проектах по машинному обучению - библиотек numpy, pandas, средств визуализации matplotlib, seaborn.

**Ход работы:**

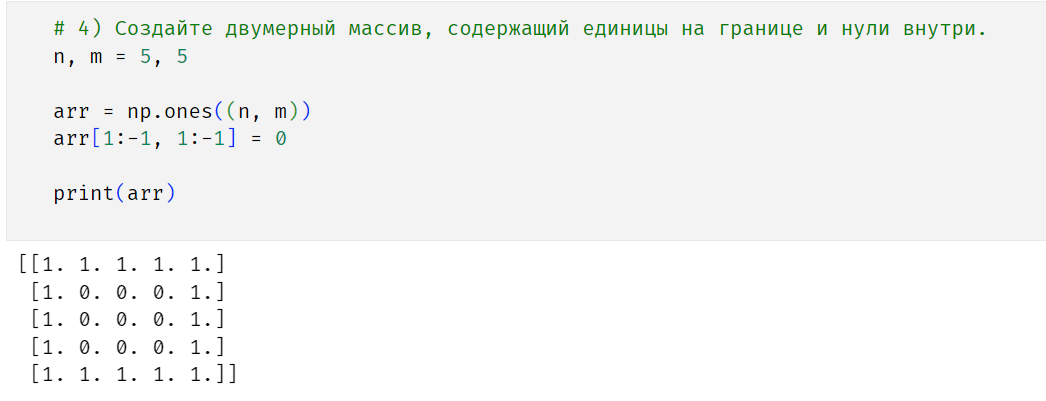


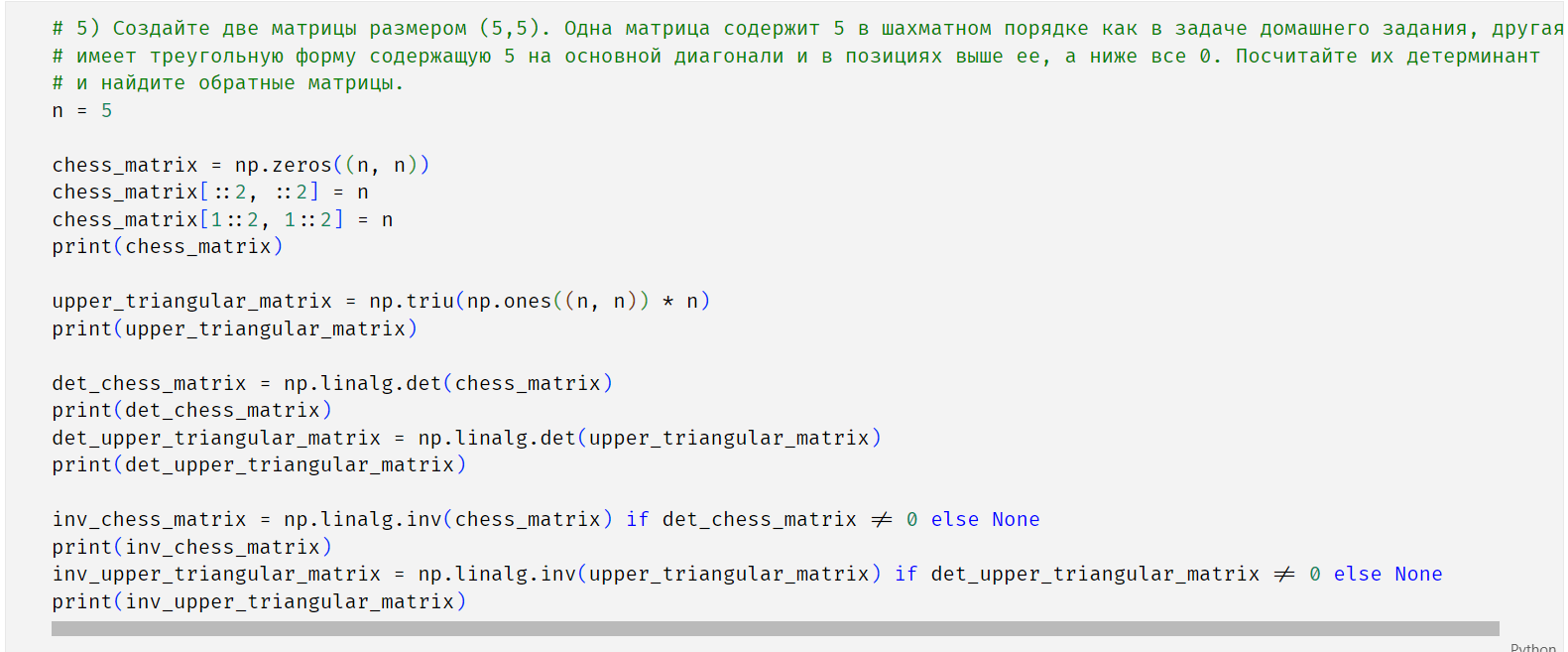
A screenshot of a computer

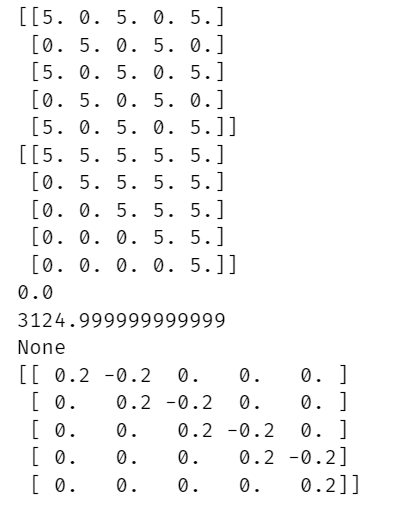
Description automatically generated

A screenshot of a computer program

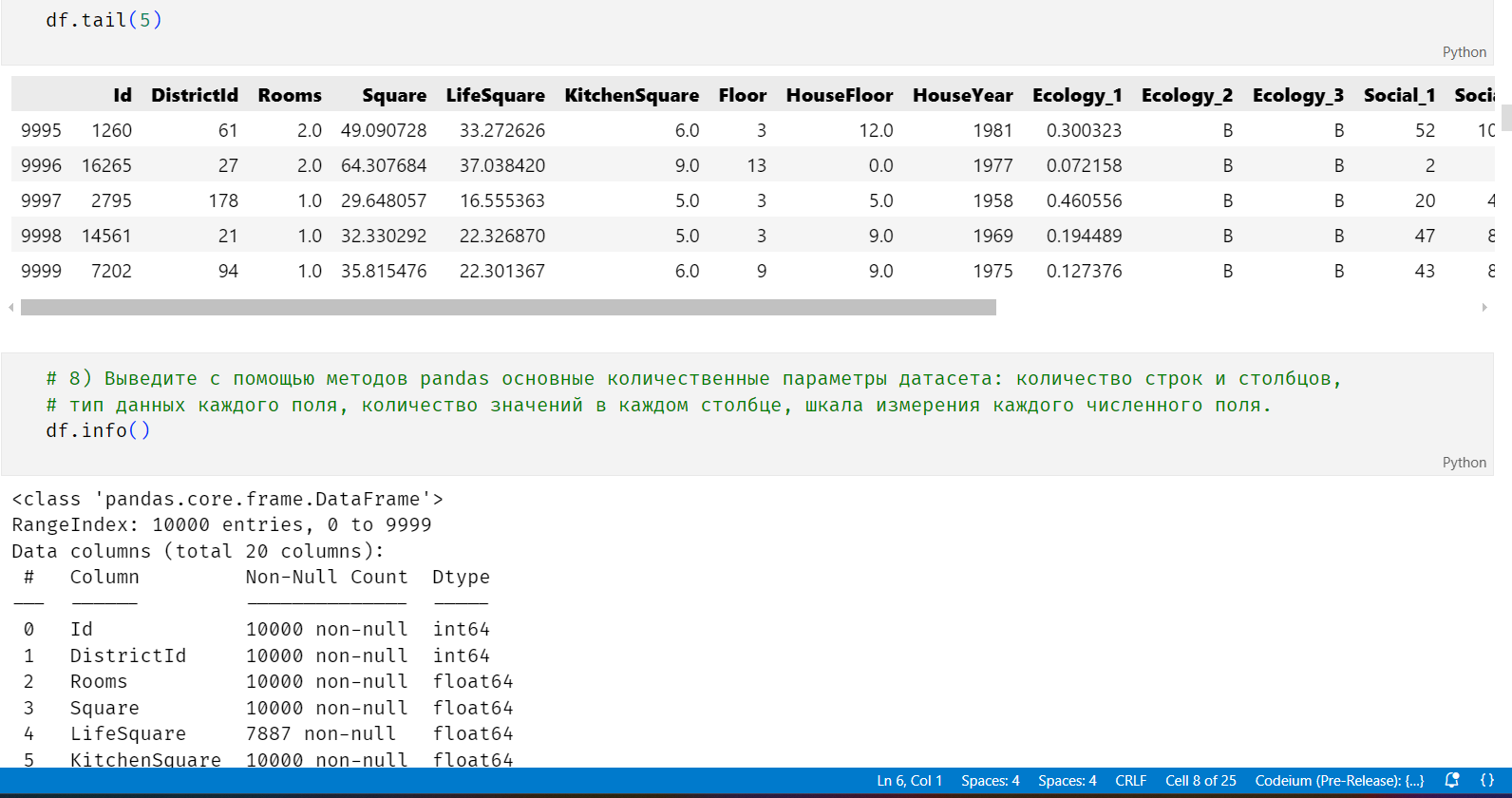
Description automatically generated









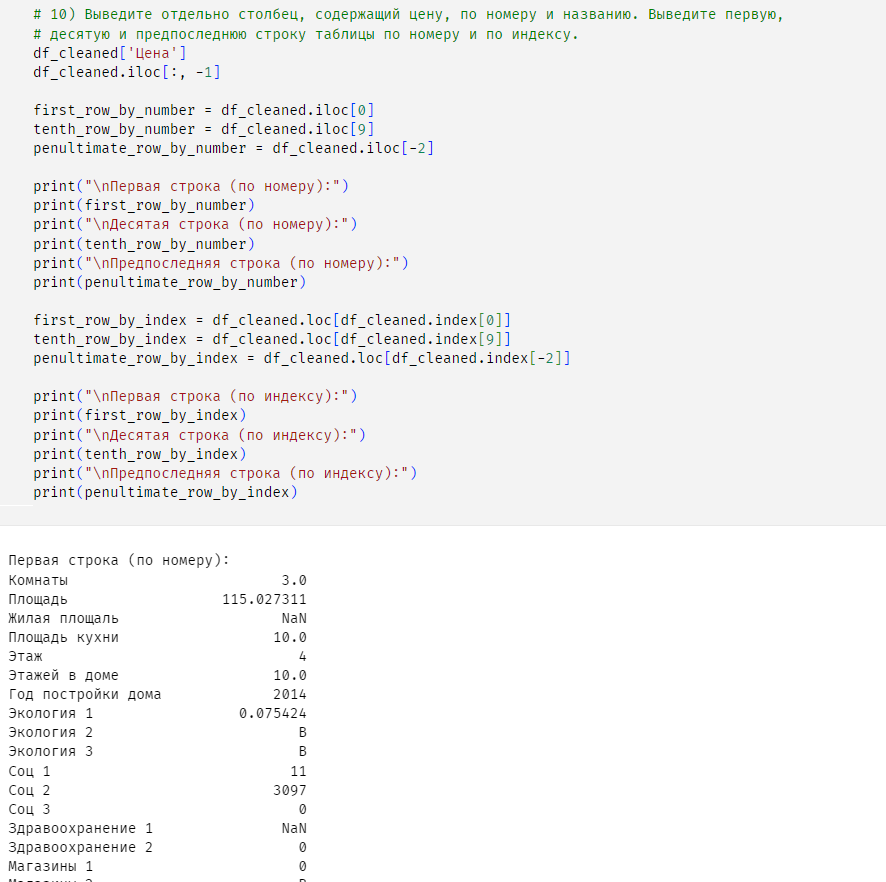


A screenshot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generated



A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A white screen with text

Description automatically generated

A screenshot of a computer

Description automatically generated

A pie chart with numbers and a few percentages

Description automatically generated

A graph with blue lines

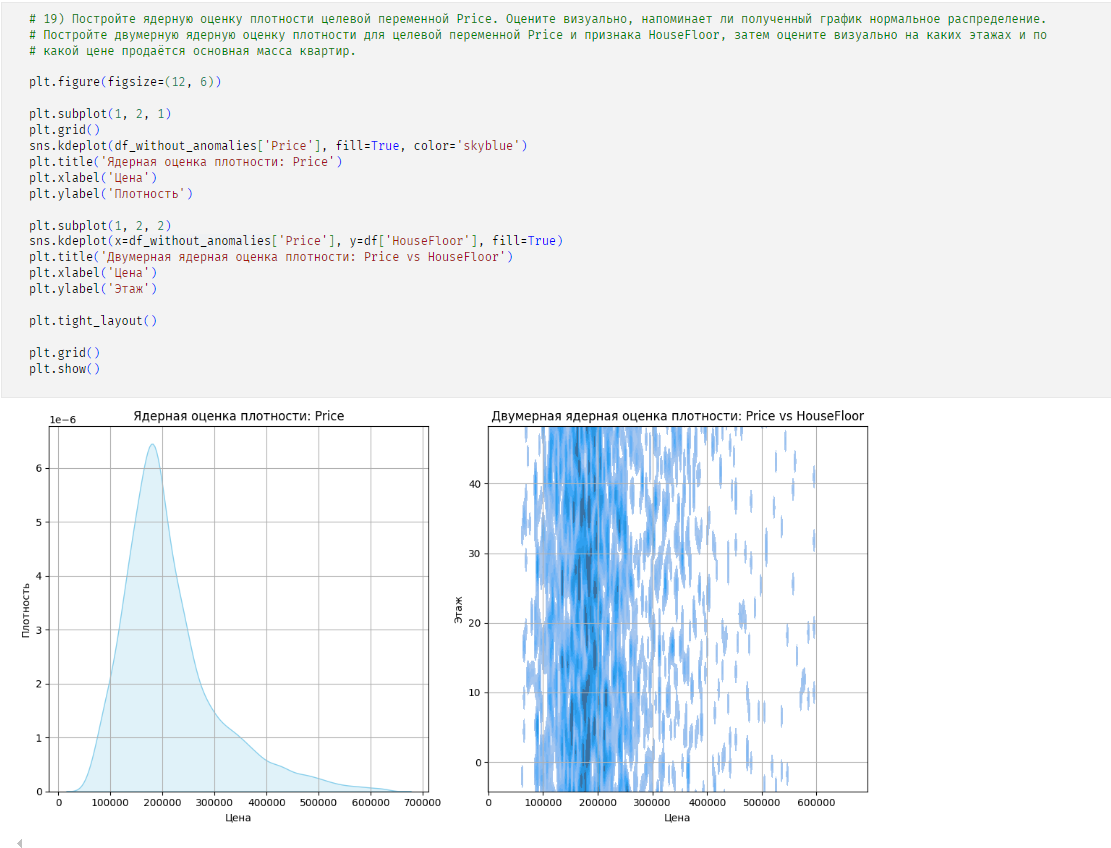
Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated

A group of blue and white graphs

Description automatically generated

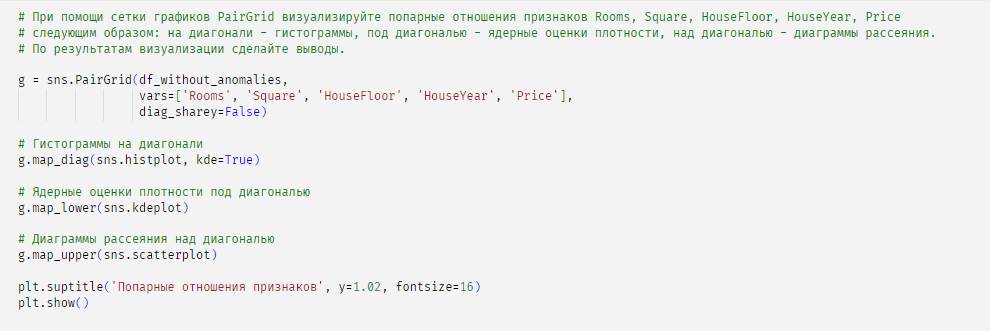


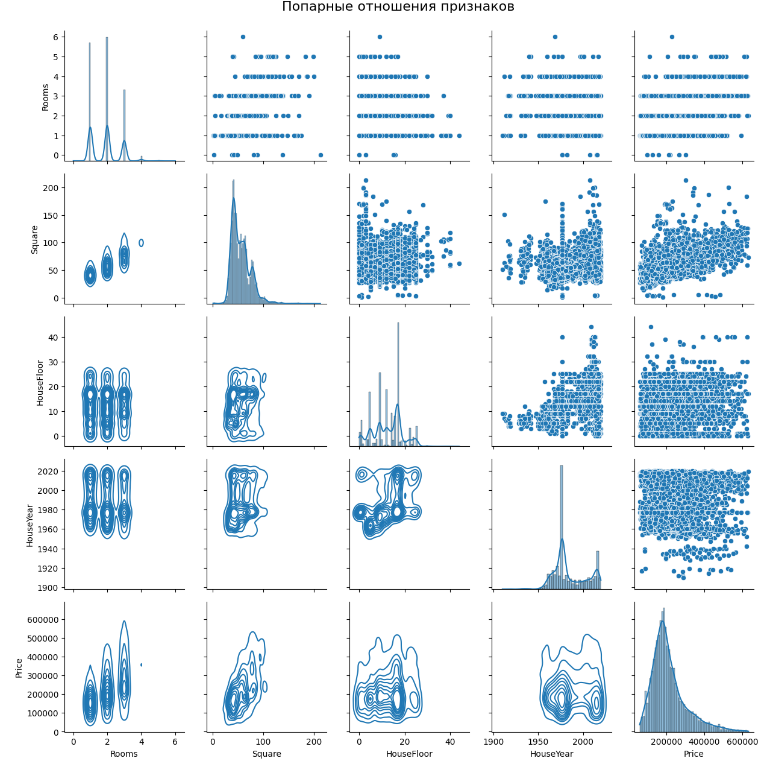
A graph of a graph of a graph

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated





A screenshot of a computer

Description automatically generated

**Контрольные вопросы**

1. *Вопрос от преподавателя.* Что означают отрицательные значения на тепловой карте?

Отрицательные значения сигнализируют об обратной пропорциональности величин. При увеличении одной другая уменьшается и наоборот.

1. Какие структуры данных используются в Numpy? В чем их отличие от списков Python?

В NumPy используется структура данных массив (ndarray). Она отличается от списков Python следующим:

* Гомогенность - все элементы массива одного типа, тогда как список может содержать элементы разных типов.
* Эффективность - массивы используют меньше памяти благодаря фиксированной длине элементов.
* Скорость - операции с массивами выполняются быстрее благодаря векторизации и оптимизированному коду на C.
* Векторизация - можно выполнять операции сразу на весь массив, избегая циклов.
* Размерность - поддерживаются многомерные массивы, тогда как списки одномерные.

Массивы NumPy предпочтительны для работы с числовыми данными и вычислениями.

1. Какие функции для генерации массивов использует Numpy?

* numpy.array(): создает массив из существующего списка или другого объекта.
* numpy.zeros(): создает массив, заполненный нулями.
* numpy.ones(): создает массив, заполненный единицами.
* numpy.full(): создает массив, заполненный указанным значением.
* numpy.arange(): создает массив с равномерно распределенными значениями в заданном диапазоне (аналог range).
* numpy.linspace(): создает массив с указанным количеством равномерно распределенных значений между началом и концом диапазона.
* numpy.random.rand(): создает массив случайных значений от 0 до 1 с равномерным распределением.
* numpy.random.randn(): создает массив случайных значений с нормальным распределением.
* numpy.eye(): создает единичную матрицу.
* numpy.empty(): создает массив без инициализации значений (содержит случайные данные).

1. Какие способы предлагает Numpy для извлечения данных из массивов?

* Индексация: доступ к элементам по индексам.
* Срезы: извлечение подмассивов с помощью диапазонов индексов.
* Булева индексация: выбор элементов по условию.
* Fancy индексация: выбор элементов с использованием списков индексов.
* Итерация: обход элементов с помощью циклов.

1. Что такое векторизация кода и почему это ускоряет работу программ?

Векторизация кода — это замена циклов операциями над целыми массивами данных, которые выполняются сразу для всех элементов. В NumPy это реализуется через оптимизированные низкоуровневые функции на C.

Ускорение достигается за счет:

* Избегания циклов в Python, что снижает накладные расходы.
* Параллельной обработки данных на уровне процессора (использование SIMD-инструкций).

1. Какие виды матричных операций реализованы в Numpy?

* Сложение и вычитание: элемент-wise операции для матриц.

np.add(A, B), np.subtract(A, B)

* Умножение:

Элемент-wise умножение: np.multiply(A, B)

Матричное умножение: np.dot(A, B) или A @ B

* Транспонирование: поворот матрицы.

np.transpose(A) или A.T

* Обратная матрица: для квадратных матриц.

np.linalg.inv(A)

* Определитель: вычисление детерминанта.

np.linalg.det(A)

* Собственные значения и векторы:

np.linalg.eig(A)

1. Какие функции используются для преобразования формы, размера и соединения массивов?

* Преобразование формы:

reshape(): изменение формы массива.

ravel(): преобразование многомерного массива в одномерный.

transpose(): транспонирование массива.

* Изменение размера:

resize(): изменение размера массива с возможностью заполнения пустых мест.

* Соединение массивов:

concatenate(): объединение массивов вдоль указанной оси.

stack(): объединение массивов вдоль новой оси.

hstack(), vstack(): горизонтальное и вертикальное соединение массивов.

1. Какие две главные структуры данных используются в pandas? В чем их отличие?

* Series: одномерная структура, похожая на столбец таблицы или массив, с метками (индексами) для каждого элемента.
* DataFrame: двумерная таблица с данными, состоящая из нескольких Series (столбцов), где каждый столбец может содержать данные разного типа.

Отличие:

* Series — одномерный массив с индексами.
* DataFrame — двумерная таблица, состоящая из нескольких Series, где строки и столбцы также индексируются.

1. Как происходит объединение двух таблиц в pandas?

* merge(): объединяет таблицы по общим столбцам, аналогично SQL JOIN (по умолчанию — inner join).
* concat(): объединяет таблицы по строкам или столбцам (сцепление), может работать с несколькими таблицами одновременно.
* join(): удобен для соединения по индексам, но также может объединять по столбцам.

1. Зачем нужны и как работают индексы в pandas.

Индексы в pandas служат для:

* Уникальной идентификации: они позволяют быстро и удобно находить и обращаться к строкам данных.
* Упрощения операций: индексы облегчают слияние, объединение и выборку данных.
* Упорядочивания: они обеспечивают порядок строк, что полезно для сортировки и временных рядов.

Работа с индексами:

* Создание: индексы создаются автоматически, но могут быть заданы вручную.
* Обращение: доступ к строкам осуществляется через индексы, например, df.loc[index].
* Переиндексация: можно изменять индексы с помощью set\_index() и reset\_index().

1. Построение каких основных видов графиков используется при анализе данных в машинном обучении?

* Гистограммы: показывают распределение числовых данных, позволяя визуализировать частоту значений.
* Диаграммы разброса (scatter plots): отображают взаимосвязь между двумя переменными, помогают выявлять корреляцию.
* Линейные графики: используются для отображения изменений переменной во времени или зависимости между переменными.
* Круговые диаграммы: показывают процентное соотношение категориальных данных.
* Корреляционные матрицы: визуализируют корреляции между множеством переменных.
* Ящики с усами (box plots): позволяют увидеть распределение данных, выявить выбросы и сравнить несколько групп.

1. В чём разница между библиотеками matplotlib и seaborn? Каковы преимущества каждой из них?

Matplotlib и Seaborn — это обе библиотеки для визуализации данных в Python, но они имеют свои особенности и преимущества:

Matplotlib

* Описание: Базовая библиотека для создания графиков, предлагающая широкие возможности настройки.
* Преимущества:
* Гибкость: позволяет создавать разнообразные виды графиков с высокой степенью настройки.
* Широкие возможности: подходит для создания как простых, так и сложных графиков.
* Поддержка анимации и интерактивности.

Seaborn

Описание: Библиотека, построенная на основе Matplotlib, предназначенная для более удобной визуализации статистических данных.

Преимущества:

* Упрощение: предоставляет высокоуровневые интерфейсы для создания сложных графиков (например, распределения, пары и тепловые карты).
* Эстетика: графики по умолчанию имеют более приятный вид и стиль.
* Статистические функции: встроенные функции для визуализации статистики и анализа данных.

Разница

Матplotlib — это основа для построения графиков с низким уровнем абстракции, в то время как Seaborn предлагает более высокоуровневые функции для работы с данными и упрощает процесс создания визуализаций, особенно для статистических графиков.

1. Как задать размер графика в matplotlib?

plt.figure(figsize=(10, 5)) # Ширина 10 дюймов, высота 5 дюймов

1. Как установить стили в seaborn?

В Seaborn стили можно установить с помощью функции set\_style(). Эта функция позволяет выбрать один из предустановленных стилей для графиков, таких как:

* "white"
* "dark"
* "whitegrid"
* "darkgrid"
* "ticks"

Пример:

sns.set\_style("whitegrid")

1. Для чего используют подграфики subplots?

Подграфики (subplots) в Matplotlib используются для отображения нескольких графиков в одном окне, что позволяет сравнивать данные и визуализировать их в более компактном виде.

Основные преимущества:

* Сравнение: удобно сравнивать разные наборы данных или различные аспекты одного и того же набора данных.
* Экономия пространства: позволяет эффективно использовать экранное пространство, избегая необходимости открывать несколько окон.
* Упрощение анализа: помогает видеть общие тенденции и закономерности между различными графиками.

1. Какие основные типы графиков реализованы в matplotlib? Что изображается на ящиковой диаграмме?

В Matplotlib реализованы следующие основные типы графиков:

* Линейные графики: отображают изменение одной переменной относительно другой.
* Гистограммы: показывают распределение данных по интервалам.
* Диаграммы разброса (scatter plots): визуализируют взаимосвязь между двумя переменными.
* Круговые диаграммы: показывают доли частей в целом.
* Ящиковые диаграммы (box plots): показывают распределение данных через квартильные значения и выявляют выбросы.

На ящиковой диаграмме отображаются:

* Ящик: охватывает межквартильный диапазон (IQR), который включает 50% данных.
* Медиана: линия внутри ящика, представляющая центральное значение.
* Усы: линии, идущие от ящика, показывающие диапазон данных (обычно до 1.5 \* IQR от квартилей).
* Выбросы: точки, находящиеся за пределами усов, которые представляют собой аномальные значения.

1. Как поменять палитру цветов у тепловой карты?

В Seaborn палитру цветов у тепловой карты можно изменить с помощью параметра cmap в функции heatmap(). Можно использовать предустановленные палитры или создать собственные.

Пример: sns.heatmap(data, cmap='coolwarm')

## Выводы

В ходе лабораторной работы были изучены основные библиотеки для анализа и визуализации данных в Python: NumPy и pandas для работы с массивами и таблицами, а также Matplotlib и Seaborn для построения графиков.

В результате работы были освоены практические навыки работы с данными, что является важным аспектом в области анализа данных и машинного обучения.