Лабораторная работа №1  
  
**Установка Python на компьютер**

1. Первым делом скачиваем Python c официального сайта: [Welcome to Python.org](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://www.python.org/%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1741314698711616%26amp;usg%3DAOvVaw2ForCiJfrcL08c7ZNufotu&sa=D&source=docs&ust=1741314698741822&usg=AOvVaw2aIRnm2m5tSTOvxhJAXy7d). Рекомендуемая версия python - 3.10, так как версия 3.8 уже закончила свой жизненный цикл в октябре этого года, следовательно версия 3.9 будет следующей, которая в скором времени окажется в такой же ситуации. Поэтому используем версию 3.10.
2. Как установить Python на Windows. Пошаговая инструкция по ссылке: [Как скачать Python на Windows 10 — пошаговая установка](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://pythonru.com/baza-znanij/skachat-i-ustanovit-python-na-windows-10%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1741314698711966%26amp;usg%3DAOvVaw1ZR46F8AQXVZm9OTVMW_36&sa=D&source=docs&ust=1741314698742103&usg=AOvVaw2eUHweZnncQ-Bt-GMXqGX1)
3. Касаемо IDE рекомендуется использовать Community версию [PyCharm](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://www.jetbrains.com/pycharm/download/?section%253Dwindows%2523section%253Dwindows%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1741314698712226%26amp;usg%3DAOvVaw3n2_J_8W3qAmWvqZf8Tfgd&sa=D&source=docs&ust=1741314698742249&usg=AOvVaw1RgNF2h0QYbtfqR16BRBZq)

**Виртуальное окружение Python**

Виртуальное окружение позволяет изолировать зависимости разных проектов, избегая конфликтов между библиотеками. Разберемся как его создать:

1. В командной строке (или через проводник) создайте папку, в которой будет лежать проект:  
           mkdir my\_ml\_project

cd my\_ml\_project

1. В командной строке введите команду для создания виртуального окружения (вместо **<your\_directory\_name>** введите своё название, обычно используется venv):  
           python -m venv <your\_directory\_name>
2. Активируйте виртуальное окружение:
3. Windows:  
           <your\_directory\_name>\Scripts\activate
4. Linux/MacOS:  
           source <your\_directory\_name>/bin/activate

После этих действий командная строка будет выглядеть вот так (вместо venv будет название Вашего виртуального окружения):



**Установка пакетов (библиотек)**

По умолчанию в Python в качестве менеджера пакетов используется pip. Чтобы установить какую-нибудь библиотеку с помощью pip в командной строке выполнить команду:  
        pip install <package\_name>

Например:

        pip install pandas

Эта команда установит последнюю версию этой библиотеки, но если нужна другая версия, то можно установить её вот так:  
  
        pip install pandas==2.2.0

Так же в Python обычно создается файл со всеми зависимостями проекта (обычно называется requirements.txt). Чтобы установить все библиотеки из этого файла, используете такую команду:

        pip install -r requirements.txt

        Если нужно создать свой файл с зависимостями проекта, то используют такую команду:  
  
        pip freeze > requirements.txt

        Эта команда создаст список из всех библиотек, которые установлены для Вашего интерпретатора.

Именно поэтому приветствуется создание **виртуального окружения** перед началом работы над новым проектом, так как оно изолирует пакеты конкретного для этого проекта от основного интерпретатора.

**Начало работы**

Перед тем как обучать модели машинного обучения, необходимо подготовить данные. В реальных задачах датасеты часто содержат пропущенные значения, выбросы и категориальные признаки, которые алгоритмы не могут обработать напрямую. Неподготовленные данные могут привести к ошибочным выводам и низкой точности модели.

В данной лабораторной работе мы разберём основные этапы предобработки данных:

1. **Загрузка данных** (например, с Kaggle).
2. **Обработка пропущенных значений** – замена на моду, медиану, среднее или дисперсию.
3. **Нормализация данных** – приведение числовых данных к единому масштабу.
4. **Преобразование категориальных данных** – кодирование с помощью one-hot encoding.
5. **Сохранение обработанных данных и выгрузка на GitHub**.

## **Загрузка данных**

Для анализа можно использовать датасет **Titanic** с Kaggle, содержащий информацию о пассажирах «Титаника», включая возраст, пол, класс каюты и информацию о спасении.

Данные можно загрузить с Kaggle и открыть в Python с помощью библиотеки pandas:

import pandas as pd

df = pd.read\_csv("titanic.csv")  # путь к файлу

df.head()  # просмотр первых строк

## **Заполнение пропущенных значений**

Пропущенные значения (NaN) могут возникать по разным причинам: ошибки при сборе данных, невозможность получения информации и т. д. Они могут негативно повлиять на работу модели, поэтому их нужно обработать.

### **Способы заполнения пропущенных данных**

* **Мода (самое частое значение)** – подходит для категориальных признаков.
* **Медиана (среднее значение после сортировки)** – устойчива к выбросам.
* **Среднее значение** – наиболее часто используемый метод для числовых данных.
* **Дисперсия** – редко используется, но может помочь при определённых задачах.

Пример замены пропущенных значений медианой:

df["Age"].fillna(df["Age"].median(), inplace=True)

Заполнение модой для категориальных данных:

df["Embarked"].fillna(df["Embarked"].mode()[0], inplace=True)

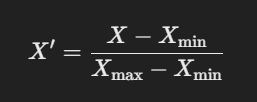
## **Нормализация данных**

Нормализация — это приведение числовых данных к единому масштабу, что улучшает работу алгоритмов, чувствительных к диапазону значений (например, градиентный спуск).

### **Методы нормализации**

1. **Минимально-максимальное масштабирование (MinMaxScaler)**

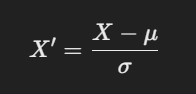
* Преобразует значения в диапазон от 0 до 1.
* Формула:



* Используется, когда важны относительные различия между значениями.

1. **Z-оценка (StandardScaler)**

* Преобразует данные так, чтобы среднее = 0, а стандартное отклонение = 1.
* Формула:



Пример нормализации возраста пассажиров:

from sklearn.preprocessing import MinMaxScaler, StandardScaler

scaler = MinMaxScaler()  # или StandardScaler()

df["Age"] = scaler.fit\_transform(df[["Age"]])

## **Преобразование категориальных данных в численные**

Алгоритмы машинного обучения не могут работать с текстовыми данными, поэтому их нужно закодировать.

### **One-hot encoding**

One-hot encoding (OHE) создает отдельные бинарные столбцы для каждого уникального значения категориального признака.

Пример кодирования порта посадки пассажира (Embarked):

df = pd.get\_dummies(df, columns=["Embarked"], drop\_first=True)

Теперь вместо одного столбца **Embarked** появятся несколько: **Embarked\_C, Embarked\_Q, Embarked\_S**, где 1 – наличие признака, 0 – его отсутствие.

## **Сохранение обработанных данных и выгрузка на GitHub**

После предобработки можно сохранить датасет в CSV и загрузить его в репозиторий GitHub.

### Сохранение данных

df.to\_csv("processed\_titanic.csv", index=False)

### **Выгрузка на GitHub**

1. Создайте репозиторий на [GitHub](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://github.com/%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1741314698722696%26amp;usg%3DAOvVaw2rLAXGF4myQtyKAtpamQam&sa=D&source=docs&ust=1741314698747021&usg=AOvVaw2cw3qyHeHuHSFY7llIugmN).
2. Перейдите в терминал и выполните:  
   git init

git add .

git commit -m "Added lab1 code"

git branch -M main

git remote add origin [https://github.com/ваш-аккаунт/ваш-репозиторий.git](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://github.com/%2525D0%2525B2%2525D0%2525B0%2525D1%252588-%2525D0%2525B0%2525D0%2525BA%2525D0%2525BA%2525D0%2525B0%2525D1%252583%2525D0%2525BD%2525D1%252582/%2525D0%2525B2%2525D0%2525B0%2525D1%252588-%2525D1%252580%2525D0%2525B5%2525D0%2525BF%2525D0%2525BE%2525D0%2525B7%2525D0%2525B8%2525D1%252582%2525D0%2525BE%2525D1%252580%2525D0%2525B8%2525D0%2525B9.git%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1741314698723356%26amp;usg%3DAOvVaw3EdZDjE7gBVABCokDbGegw&sa=D&source=docs&ust=1741314698747355&usg=AOvVaw21v9WP6bIdrCJxNOKVfCP_)

git push -u origin main

1. Отправьте ссылку на репозиторий преподавателю.

**Задание**

1. Загрузить данные с Kaggle – выбрать датасет (например, [Титаник](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://www.kaggle.com/competitions/spaceship-titanic%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1741314698723999%26amp;usg%3DAOvVaw14bKEWOL0gqa9bLWexWAHL&sa=D&source=docs&ust=1741314698747723&usg=AOvVaw3yunIY3xH4attYPsexxpqK))
2. Вывести с помощью python данные из датасета на экран
3. Получить количество пропущенных значений для каждого столбца в датасетах
4. Заполнить пропущенные значения в датасете модой/медианой/средним значением
5. Провести нормализацию данных
6. Преобразовать категориальные данные в численные (one-hot encoding)
7. Выгрузить **весь** код на гитхаб – (сбросить ссылку преподавателю)