Министерство науки и образования РФ

Федеральное государственное бюджетное учреждение

высшего образования

**«Тверской государственный технический университет»**

(ТвГТУ)

Кафедра программного обеспечения

**Отчет по лабораторной работе №2**

По дисциплине: «Анализ больших данных»

Тема: «Исследовательский анализ данных. Постановка гипотез Категориальные данные»

|  |
| --- |
| Выполнила:  студентка группы  Б.ПИН.РИС - 21.06  Александрова М.И. |
| Проверила:  старший преподаватель  кафедры ПО  Корнеева Е.И. |

Тверь 2025

Оглавление

[Введение 3](#_Toc190087525)

[Общее задание на работу 3](#_Toc190087526)

[Задание для самостоятельного выполнения 4](#_Toc190087527)

[Описание проделанной работы 5](#_Toc190087528)

[**1.** **Ознакомьтесь с набором данных mpg из библиотеки Seaborn.** 5](#_Toc190087529)

[2. **Посчитайте количество строк и столбцов.** 5](#_Toc190087530)

[3. **Проведите разведочный анализ, то есть:** 5](#_Toc190087531)

[**4. Сформулируйте и проверьте минимум 2 статистические гипотезы.** 8](#_Toc190087532)

[5. **Закодируйте категориальные переменные, необходимые для анализа**, если требуется. Методом OneHotEncoding или LabelEncoding. 9](#_Toc190087533)

[**6. Постройте таблицу корреляции признаков и целевого столбца**. 9](#_Toc190087534)

[7. **Реализуйте стохастический и обычный градиентный спуск вручную, можно использовать ноутбук с лекции ссылка. Для этих данных: y = ’mpg’ и x = ’horsepower’ или ’weight’.** 11](#_Toc190087535)

[Задание для самостоятельного выполнения 12](#_Toc190087536)

[Краткий вывод по работе 16](#_Toc190087537)

[Ссылки на используемые материалы. Документация 18](#_Toc190087538)

[Листинг кода 19](#_Toc190087539)

# **Введение**

Цель: провести исследовательский анализ данных, поставить гипотезы и выявить основные статистики.

# **Общее задание на работу**

1. Ознакомьтесь с набором данных mpg из библиотеки Seaborn.

(загрузка через df = sns.load\_dataset(’mpg’))

1. Посчитайте количество строк и столбцов.
2. Проведите разведочный анализ, то есть:

(a) для каждой числовой переменной вычислите:

• Долю пропусков

• Максимальное и минимальное значение

• Среднее значение

• Медиану

• Дисперсию

• Квантиль 0.1 и 0.9

• Квартиль 1 и 3,

(b) для каждой категориальной переменной вычислите:

• Долю пропусков

• Количество уникальных значений

• Моду

1. Сформулируйте и проверьте минимум 2 статистические гипотезы. Выбор критериев для проверки гипотез требуется обосновать. Сделать выводы в терминах предметной области.
2. Закодируйте категориальные переменные, необходимые для анализа, если требуется. Методом OneHotEncoding или LabelEncoding.
3. Постройте таблицу корреляции признаков и целевого столбца. Обоснуйте, какой столбец является целевым, а какие признаками.
4. Реализуйте стохастический и обычный градиентный спуск вручную, можно использовать ноутбук с лекции ссылка. Для этих данных: y = ’mpg’ и x = ’horsepower’ или ’weight’.

# **Задание для самостоятельного выполнения**

Реализовать задание для данных вашего варианта

1. Преобразовать категориальные переменные в числовые, если это необходимо. Добавить вычисляемые столбцы.

2. Посчитайте количество строк и столбцов.

3. Проведите разведочный анализ, то есть:

(a) для каждой числовой переменной вычислите:

• Долю пропусков

• Максимальное и минимальное значение

• Среднее значение

• Медиану

• Дисперсию

• Квантиль 0.1 и 0.9

• Квартиль 1 и 3

(b) для каждой категориальной переменной вычислите:

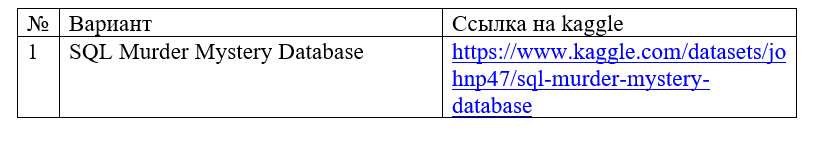
• Долю пропусков

• Количество уникальных значений

• Моду

1. Сформулируйте и проверьте минимум 2 статистические гипотезы. Выбор критериев для проверки гипотез требуется обосновать. Сделать выводы в терминах предметной области.
2. Постройте таблицу корреляции признаков и целевого столбца. Обоснуйте, какой столбец является целевым, а какие признаками.

Вариант и условия задачи:



Сложность: Medium

**Сложность: Medium**

* Реализовать общую и самостоятельную часть задания
* Лабораторная работа 1 реализована на **Medium**

**Ссылка на репозиторий с программной реализацией**: <https://github.com/milana-cat/BigData>

**Описание проделанной работы**

1. **Ознакомьтесь с набором данных mpg из библиотеки Seaborn.**

(загрузка через df = sns.load\_dataset(’mpg’))

def load\_mpg\_dataset():

    #Загрузка набора данных 'mpg' из библиотеки Seaborn.

    try:

        df = sns.load\_dataset('mpg')

        print("Набор данных 'mpg' успешно загружен")

        return df

    except Exception as e:

        print("Ошибка загрузки набора данных 'mpg':", e)

        exit()

2. **Посчитайте количество строк и столбцов.**

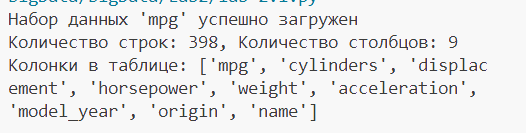
def analyze\_dataset(df):

    #Анализ набора данных: подсчёт строк и столбцов.

    rows, columns = df.shape

    print(f"Количество строк: {rows}, Количество столбцов: {columns}")

    print("Колонки в таблице:", df.columns.tolist())



3. **Проведите разведочный анализ, то есть:**

(a) для каждой числовой переменной вычислите:

• Долю пропусков

• Максимальное и минимальное значение

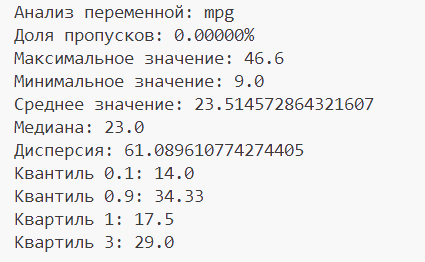
• Среднее значение

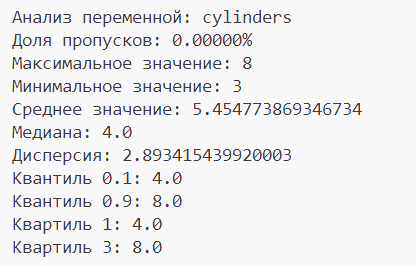
• Медиану

• Дисперсию

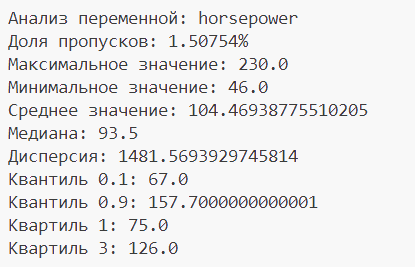
• Квантиль 0.1 и 0.9

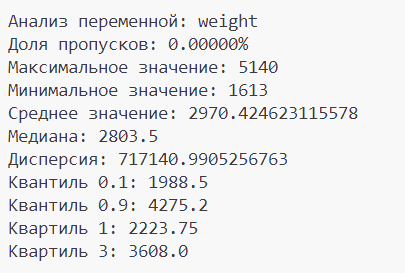
• Квартиль 1 и 3

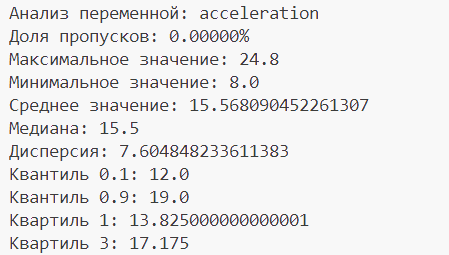


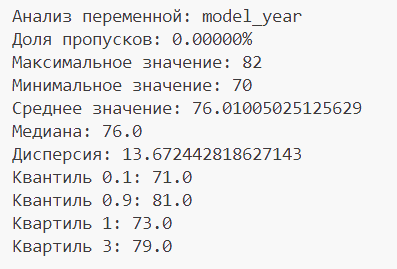










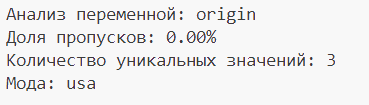


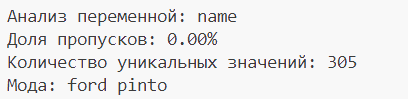
(b) для каждой категориальной переменной вычислите:

• Долю пропусков

• Количество уникальных значений

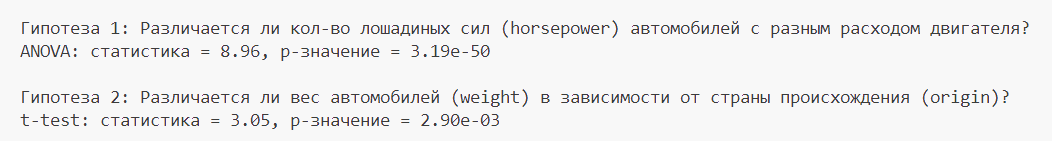
• Моду





**4. Сформулируйте и проверьте минимум 2 статистические гипотезы.**

Выбор критериев для проверки гипотез требуется обосновать. Сделать выводы в терминах предметной области.



**Обоснование выбора критериев:**

Гипотеза 1: Различается ли кол-во лошадиных сил (horsepower) автомобилей с разным числом цилиндров?

Для проверки используется ANOVA (дисперсионный анализ), так как:

* Переменная horsepower (кол-во лошадиных сил) является числовой.
* Группы, по которым происходит разделение (cylinders), являются категориальными (разные количества цилиндров).
* Мы хотим проверить наличие различий между средними значениями расхода топлива в нескольких группах (по числу цилиндров).

ANOVA позволяет оценить, есть ли значимые различия между средними значениями в более чем двух группах.

Гипотеза 2: Различается ли количество лошадиных сил автомобилей (horsepower) в зависимости от страны происхождения (origin)?

Для проверки используется t-тест для двух независимых выборок, так как:

* Переменная horsepower (количество лошадиных сил) является числовой.
* Группы (origin, например, USA и Europe) представляют две независимые категории.
* Мы сравниваем средние значения количества лошадиных сил автомобилей из двух разных стран происхождения.

Выводы в терминах предметной области:

1. Гипотеза 1 (ANOVA):

* Если гипотеза отвергается (p-value < 0.05), это означает, что расход топлива существенно различается в зависимости от количества лошадиных сил. Это важно для понимания, как технические характеристики двигателя (количество лошадиных сил) влияют на экономичность автомобиля.
* Если гипотеза не отвергается, можно сделать вывод, что расход топлива не зависит от количества лошадиных сил.

1. Гипотеза 2 (t-тест):

* Если гипотеза отвергается (p-value < 0.05), это указывает на значительную разницу в мощности автомобилей из разных стран. Это может отражать различия в подходах к проектированию и производству автомобилей в США и Европы.
* Если гипотеза не отвергается, это значит, что различий в мощности автомобилей между этими странами нет, что может указывать на схожесть конструктивных подходов.

5. **Закодируйте категориальные переменные, необходимые для анализа**, если требуется. Методом OneHotEncoding или LabelEncoding.

def encode\_categorical\_variables(df):

    #Кодирование категориальных переменных.

    categorical\_columns = df.select\_dtypes(include=['object', 'category']).columns

    label\_encoders = {}

    for col in ['origin']:

        le = LabelEncoder()

        df[col] = le.fit\_transform(df[col].astype(str))

        label\_encoders[col] = le

        print(f"Столбец '{col}' закодирован методом LabelEncoding.")

    one\_hot\_cols = ['name']

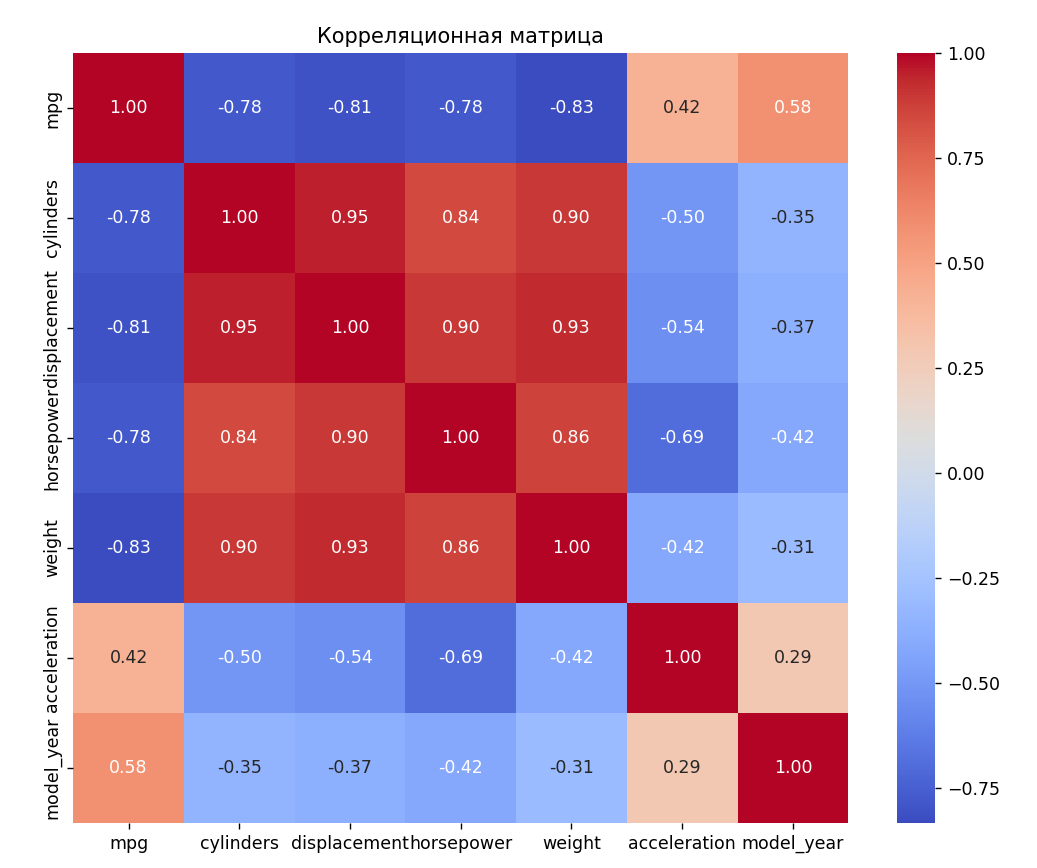
    df = pd.get\_dummies(df, columns=one\_hot\_cols, drop\_first=True)

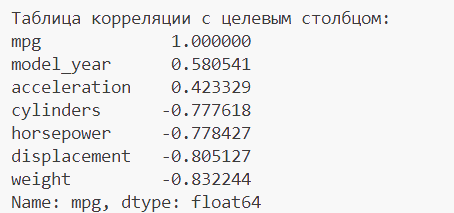
    print("Закодированы OneHotEncoding: 'name'")

    return df

**6. Постройте таблицу корреляции признаков и целевого столбца**.

Обоснуйте, какой столбец является целевым, а какие признаками.





**Целевой столбец:** mpg (расход топлива). Он является ключевым показателем эффективности автомобиля.

**Признаки:** остальные числовые переменные (например, horsepower, weight, acceleration), которые могут влиять на расход топлива.

Корреляция между переменными:

* Значения от -1 до 1 показывают силу и направление линейной связи между переменными.
* Значения ближе к 1 указывают на сильную положительную корреляцию.
* Значения ближе к -1 указывают на сильную отрицательную корреляцию.
* Значения около 0 указывают на слабую или отсутствующую корреляцию.

Ключевые связи:

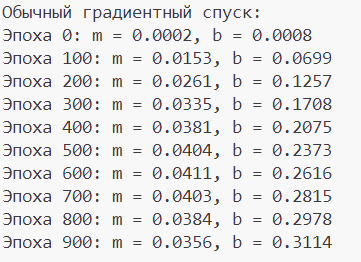
Переменная mpg (расход топлива) имеет:

1. Сильную отрицательную корреляцию с весом (-0.832244), мощностью (-0.778427), и объемом двигателя (-0.80). Следовательно, чем больше вес, мощность и объем двигателя, тем меньше расход топлива.
2. Умеренную положительную корреляцию с model\_year (0.58). Это указывает на то, что чем новее автомобиль, тем меньше расход топлива, а, следовательно, лучше экономичность.
3. Количество цилиндров положительно коррелируют с литражом (0.95) и количеством лошадиных сил (0.90), так как большее количество цилиндров связано с более мощными двигателями (у них больше литраж и больше лошадиных сил).

Целевой столбец:

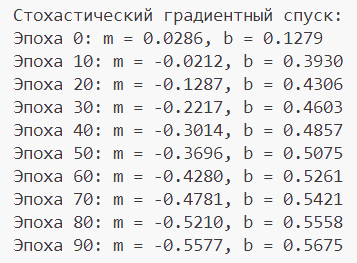
* Выбран mpg как целевой столбец, так как это ключевой показатель эффективности автомобиля.
* Остальные числовые переменные являются признаками, влияющими на расход топлива.

7. **Реализуйте стохастический и обычный градиентный спуск вручную, можно использовать ноутбук с лекции ссылка. Для этих данных: y = ’mpg’ и x = ’horsepower’ или ’weight’.**



Анализ:

* В начале коэффициенты m и b маленькие, но постепенно увеличиваются.
* Коэффициент наклона m растёт, достигая 0.0478 к 600-й эпохе, после чего начинает слегка снижаться.
* Смещение b стабильно увеличивается, что говорит о приближении к оптимальному значению.



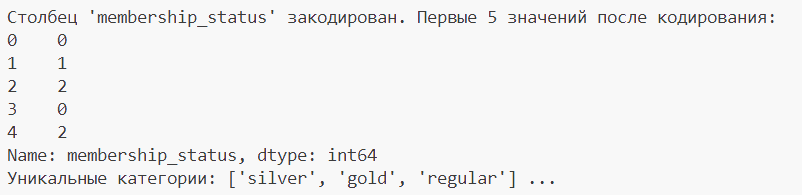
Анализ:

* В отличие от обычного градиентного спуска, здесь коэффициент m сначала уменьшается, а затем начинает колебаться.
* Стохастический градиентный спуск обновляет коэффициенты чаще, но каждое обновление основано на одном случайном примере, поэтому колебания сильнее.
* По мере роста числа эпох b (смещение) растёт, а m уменьшается, что может говорить о нестабильной или переобученной модели.

**Задание для самостоятельного выполнения**

**Вариант 1: SQL Murder Mystery Database**

1. Преобразовать категориальные переменные в числовые, если это необходимо.



Текстовые или категориальные данные, и они были преобразованы в числовые значения с помощью pd.factorize().

* Первая колонка (слева) — индексы строк (0, 1, 2, 3, ...).
* Вторая колонка (справа) — закодированные значения (0, 0, 1, 2, 3 и т. д.), где каждое уникальное значение из исходного столбца заменяется на уникальный идентификатор.

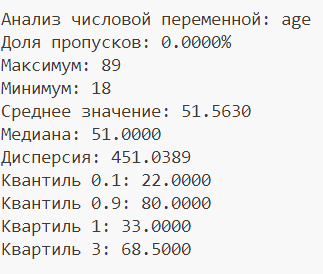
2. Посчитайте количество строк и столбцов.

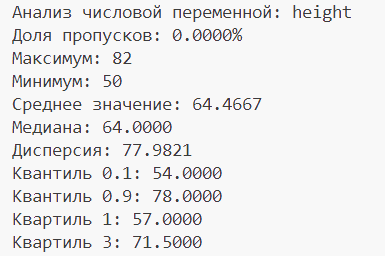


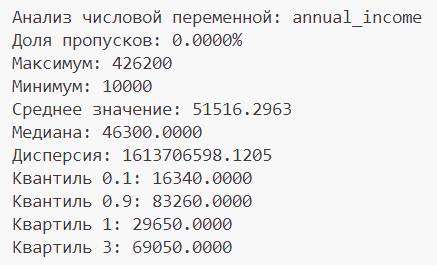
3. Проведите разведочный анализ, то есть:

(a) для каждой числовой переменной вычислите:

* Долю пропусков
* Максимальное и минимальное значение
* Среднее значение
* Медиану
* Дисперсию
* Квантиль 0.1 и 0.9
* Квартиль 1 и 3







Другие анализы переменных в таблице Excel, прикреплённая в папке с программой

(b) для каждой категориальной переменной вычислите:

* Долю пропусков
* Количество уникальных значений
* Моду

Другие анализы переменных в таблице Excel, прикреплённая в папке с программой

4. Сформулируйте и проверьте минимум 2 статистические гипотезы. Выбор критериев для проверки гипотез требуется обосновать. Сделать выводы в терминах предметной области.

Гипотеза 1: Различается ли доход за год (annual\_income) для с статусом членства (membership\_status)

Критерий проверки: Двухвыборочный t-тест для независимых выборок.

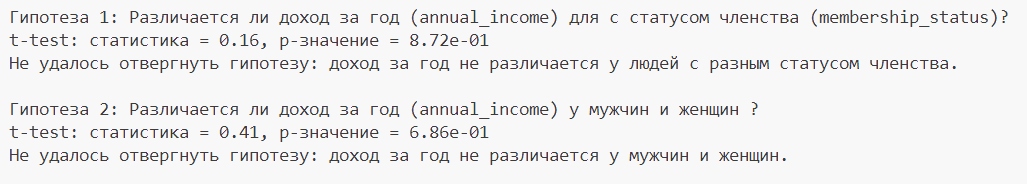
* Сравниваем средние значения одной числовой переменной (annual\_income) между двумя группами (silver и gold в membership\_status).
* t-тест подходит для сравнения двух групп, даже если они имеют разные размеры выборок.

Гипотеза 2: Различается ли доход за год (annual\_income) у мужчин и женщин?

Критерий проверки: Двухвыборочный t-тест для независимых выборок.

* Сравниваем средние значения одной числовой переменной (annual\_income) между двумя группами (woman и man в gender).
* t-тест подходит для сравнения двух групп, даже если они имеют разные размеры выборок.

Результаты гипотез:



5. Постройте таблицу корреляции признаков и целевого столбца. Обоснуйте, какой столбец является целевым, а какие признаками.

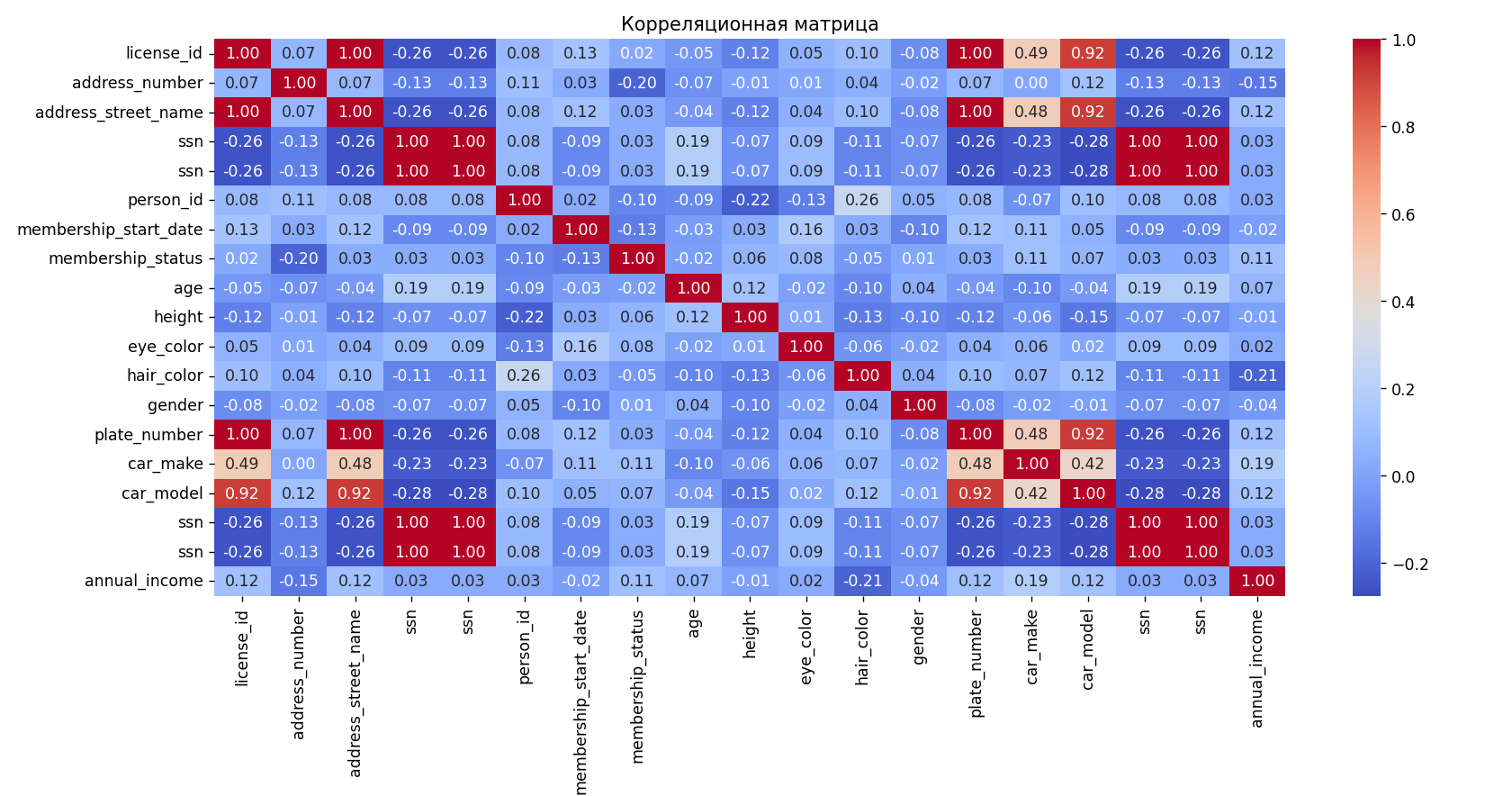


Таблица корреляции с целевым столбцом (membership\_status):

membership\_status 1.000000

annual\_income 0.112690

car\_make 0.110612

eye\_color 0.078850

car\_model 0.065193

height 0.061095

plate\_number 0.032051

address\_street\_name 0.032051

ssn 0.031141

ssn 0.031141

ssn 0.031141

ssn 0.031141

license\_id 0.020116

gender 0.007620

age -0.020868

hair\_color -0.046562

person\_id -0.101791

membership\_start\_date -0.126588

address\_number -0.202307

Name: membership\_status, dtype: float64

**Целевой столбец:** membership\_status (статус членства), поскольку представляет собой интересующую нас переменную, которую могут попытаться объяснить. **Признаки:** Все числовые переменные.

На тепловой карте корреляции представлены взаимосвязи между числовыми признаками, включая целевой столбец membership\_status. Цветовая шкала указывает степень корреляции:

* Коэффициенты корреляции близкие к 1 (тёмно-красный цвет) означают сильную положительную связь между переменными (при увеличении одной переменной увеличивается другая).
* Коэффициенты близкие к -1 (тёмно-синий цвет) показывают сильную отрицательную связь (при увеличении одной переменной другая уменьшается).
* Значения около 0 (светлый цвет) означают отсутствие или слабую корреляцию.

**Выводы по целевому столбцу saleprice**

1.Самая высокая корреляция с annual\_income (0.113). Следовательно, статус членства напрямую зависит (хоть и слабо) от размера годового дохода.

2. Примерно такая же корреляция с car\_make (0.111). Что достаточно ожидаемо, поскольку люди с более высоким статусом членства имеют немного более высокий доход и могут позволить себе более дорогие марки автомобилей.

# **Краткий вывод по работе**

Работа успешно выполнена, выявлены ключевые зависимости в данных, проверены гипотезы и применены методы машинного обучения для анализа

**Функциональность программы**

1. Загрузка данных

* Функция load\_mpg\_dataset() загружает набор данных mpg.
* analyze\_dataset(df) анализирует структуру данных (количество строк и столбцов).

1. Разведочный анализ данных

* exploratory\_analysis\_numeric(df): анализ числовых переменных (доля пропусков, максимум, минимум, среднее значение, медиана, дисперсия, квантили).
* exploratory\_analysis\_categorical(df): анализ категориальных переменных (доля пропусков, количество уникальных значений, мода).

1. Обработка данных

* encode\_categorical\_variables(df): кодирование категориальных переменных.
* normalize(data): нормализация числовых переменных с использованием Min-Max Scaling.

1. Проверка статистических гипотез
2. Градиентный спуск

* gradient\_descent(x, y): реализация стандартного градиентного спуска.
* stochastic\_gradient\_descent(x, y): реализация стохастического градиентного спуска.

1. Анализ корреляций

* correlation\_analysis(df, target\_column): построение таблицы корреляций и визуализация корреляционной матрицы с помощью тепловой карты.

# **Ссылки на используемые материалы. Документация**

1. Клиент-серверные приложения Сокеты. Создание клиента // METANIT.COM URL:https://metanit.com/python/network/1.1.php (дата обращения: 15.01.2025).
2. Python Documentation contents // python.org URL: https://docs.python.org/3/contents.html (дата обращения: 15.01.2025).
3. Визуализация данных с matplotlib // NSU Programming URL: https://nsu-programming.github.io/textbook/python/plotting.html (дата обращения: 15.01.2025).

**Листинг кода**

Общее задание:

import pandas as pd

import seaborn as sns

from scipy.stats import ttest\_ind, f\_oneway

import matplotlib.pyplot as plt

import numpy as np

from sklearn.preprocessing import OneHotEncoder, LabelEncoder

def load\_mpg\_dataset():

    #Загрузка набора данных 'mpg' из библиотеки Seaborn.

    try:

        df = sns.load\_dataset('mpg')

        print("Набор данных 'mpg' успешно загружен")

        return df

    except Exception as e:

        print("Ошибка загрузки набора данных 'mpg':", e)

        exit()

def analyze\_dataset(df):

    #Анализ набора данных: подсчёт строк и столбцов.

    rows, columns = df.shape

    print(f"Количество строк: {rows}, Количество столбцов: {columns}")

    print("Колонки в таблице:", df.columns.tolist())

def normalize(data):

    #Нормализация данных (Min-Max Scaling)

    return (data - np.min(data)) / (np.max(data) - np.min(data))

def exploratory\_analysis\_numeric(df):

    #Разведочный анализ числовых переменных.

    numeric\_columns = df.select\_dtypes(include=['float64', 'int64']).columns

    for col in numeric\_columns:

        print(f"\nАнализ переменной: {col}")

        print(f"Доля пропусков: {df[col].isnull().mean():.5%}")

        print(f"Максимальное значение: {df[col].max()}")

        print(f"Минимальное значение: {df[col].min()}")

        print(f"Среднее значение: {df[col].mean()}")

        print(f"Медиана: {df[col].median()}")

        print(f"Дисперсия: {df[col].var()}")

        print(f"Квантиль 0.1: {df[col].quantile(0.1)}")

        print(f"Квантиль 0.9: {df[col].quantile(0.9)}")

        print(f"Квартиль 1: {df[col].quantile(0.25)}")

        print(f"Квартиль 3: {df[col].quantile(0.75)}")

def exploratory\_analysis\_categorical(df):

    #Разведочный анализ категориальных переменных.

    categorical\_columns = df.select\_dtypes(include=['object', 'category']).columns

    for col in categorical\_columns:

        print(f"\nАнализ переменной: {col}")

        print(f"Доля пропусков: {df[col].isnull().mean():.2%}")

        print(f"Количество уникальных значений: {df[col].nunique()}")

        print(f"Мода: {df[col].mode().iloc[0] if not df[col].mode().empty else 'Нет моды'}")

def encode\_categorical\_variables(df):

    #Кодирование категориальных переменных.

    categorical\_columns = df.select\_dtypes(include=['object', 'category']).columns

    label\_encoders = {}

    for col in ['origin']:

        le = LabelEncoder()

        df[col] = le.fit\_transform(df[col].astype(str))

        label\_encoders[col] = le

        print(f"Столбец '{col}' закодирован методом LabelEncoding.")

    one\_hot\_cols = ['name']

    df = pd.get\_dummies(df, columns=one\_hot\_cols, drop\_first=True)

    print("Закодированы OneHotEncoding: 'name'")

    return df

def gradient\_descent(x, y, lr=0.001, epochs=1000):

    #Исправленный градиентный спуск.

    m, b = 0.0, 0.0

    n = len(x)

    x = np.array(x)

    y = np.array(y)

    for epoch in range(epochs):

        y\_pred = m \* x + b

        error = y - y\_pred

        dm = -2 / n \* np.sum(x \* error)

        db = -2 / n \* np.sum(error)

        m -= lr \* dm

        b -= lr \* db

        if epoch % 100 == 0:

            print(f"Эпоха {epoch}: m = {m:.4f}, b = {b:.4f}")

    return m, b

def stochastic\_gradient\_descent(x, y, lr=0.001, epochs=100):

    #Исправленный стохастический градиентный спуск.

    m, b = 0.0, 0.0

    n = len(x)

    x = np.array(x)

    y = np.array(y)

    for epoch in range(epochs):

        for i in range(n):

            xi, yi = x[i], y[i]

            y\_pred = m \* xi + b

            error = yi - y\_pred

            m += lr \* error \* xi

            b += lr \* error

        if epoch % 10 == 0:

            print(f"Эпоха {epoch}: m = {m:.4f}, b = {b:.4f}")

    return m, b

def hypothesis\_testing(df):

    #Проверка статистических гипотез.

    print("\nГипотеза 1: Различается ли кол-во лошадиных сил (horsepower) автомобилей с разным расходом двигателя?")

    groups = [df[df['mpg'] == cyl]['horsepower'].dropna() for cyl in df['mpg'].unique() if not pd.isnull(cyl)]

    stat, p\_value = f\_oneway(\*groups)

    print(f"ANOVA: статистика = {stat:.2f}, p-значение = {p\_value:.2e}")

    print("\nГипотеза 2: Различается ли количество лошадиных сил автомобилей (horsepower) в зависимости от страны происхождения (origin)?")

    usa\_weight = df[df['origin'] == 0]['weight'].dropna()

    europe\_weight = df[df['origin'] == 1]['weight'].dropna()

    stat, p\_value = ttest\_ind(usa\_weight, europe\_weight, equal\_var=False)

    print(f"t-test: статистика = {stat:.2f}, p-значение = {p\_value:.2e}")

def correlation\_analysis(df, target\_column):

    #Построение таблицы корреляции признаков и целевого столбца.

    numeric\_columns = df.select\_dtypes(include=['float64', 'int64']).columns

    correlation\_table = df[numeric\_columns].corr()[target\_column].sort\_values(ascending=False)

    print("\nТаблица корреляции с целевым столбцом:")

    print(correlation\_table)

    plt.figure(figsize=(10, 8))

    sns.heatmap(df[numeric\_columns].corr(), annot=True, cmap='coolwarm', fmt='.2f')

    plt.title('Корреляционная матрица')

    plt.show()

if \_\_name\_\_ == "\_\_main\_\_":

    df = load\_mpg\_dataset()

    analyze\_dataset(df)

    exploratory\_analysis\_numeric(df)

    exploratory\_analysis\_categorical(df)

    df = encode\_categorical\_variables(df)

    df = df.dropna(subset=['mpg', 'horsepower'])

    x = normalize(df['horsepower'].values)

    y = normalize(df['mpg'].values)

    hypothesis\_testing(df)

    print("\nОбычный градиентный спуск:")

    gradient\_descent(x, y)

    print("\nСтохастический градиентный спуск:")

    stochastic\_gradient\_descent(x, y)

    correlation\_analysis(df, target\_column='mpg')

Задание по вариантам:

from sqlalchemy import create\_engine

import pandas as pd

from scipy.stats import f\_oneway, ttest\_ind

import seaborn as sns

import matplotlib.pyplot as plt

def connect\_to\_db(host, port, user, password, database):

    #Подключение к базе данных PostgreSQL."""

    try:

        engine = create\_engine(f"postgresql://{user}:{password}@{host}:{port}/{database}?client\_encoding=utf8")

        print("Подключение успешно")

        return engine

    except Exception as e:

        print("Ошибка подключения:", e)

        exit()

def load\_data(engine, query):

   #Загрузка данных из базы данных в DataFrame.

    try:

        df = pd.read\_sql\_query(query, engine)

        print("Данные успешно загружены в DataFrame")

        return df

    except Exception as e:

        print("Ошибка выполнения запроса:", e)

        exit()

def exploratory\_analysis\_numeric(df, columns):

    #Разведочный анализ числовых переменных.

    for col in columns:

        try:

            print(f"\nАнализ числовой переменной: {col}")

            print(f"Доля пропусков: {df[col].isnull().mean():.4%}")

            print(f"Максимум: {df[col].max()}")

            print(f"Минимум: {df[col].min()}")

            print(f"Среднее значение: {df[col].mean():.4f}")

            print(f"Медиана: {df[col].median():.4f}")

            print(f"Дисперсия: {df[col].var():.4f}")

            print(f"Квантиль 0.1: {df[col].quantile(0.1):.4f}")

            print(f"Квантиль 0.9: {df[col].quantile(0.9):.4f}")

            print(f"Квартиль 1: {df[col].quantile(0.25):.4f}")

            print(f"Квартиль 3: {df[col].quantile(0.75):.4f}")

        except Exception as e:

            print(f"Ошибка анализа переменной {col}: {e}")

def exploratory\_analysis\_categorical(df, columns):

    #Разведочный анализ категориальных переменных.

    for col in columns:

        print(f"\nАнализ категориальной переменной: {col}")

        print(f"Доля пропусков: {df[col].isnull().mean():.4%}")

        print(f"Количество уникальных значений: {df[col].nunique()}")

        print(f"Мода: {df[col].mode()[0] if not df[col].mode().empty else 'Нет моды'}")

def encode\_categorical\_variables(df):

    #Пошаговое кодирование категориальных переменных.

    categorical\_columns = df.select\_dtypes(include=['object', 'category']).columns

    for col in categorical\_columns:

        if col !='name' and col !='id':

            df[col], unique\_values = pd.factorize(df[col])

            print(f"\nСтолбец '{col}' закодирован. Первые 5 значений после кодирования:")

            print(df[col].head())

            print(f"Уникальные категории: {list(unique\_values[:5])} ...")

    return df

def analyze\_dataset(df):

    #Подсчёт строк и столбцов.

    rows, columns = df.shape

    print(f"Количество строк: {rows}, Количество столбцов: {columns}")

def test\_hypotheses(df):

    #Проверка статистических гипотез.

    print("\nГипотеза 1: Различается ли доход за год (annual\_income) для с статусом членства (membership\_status)?")

    silver\_weight = df[df['membership\_status'] == 0]['annual\_income'].dropna()

    gold\_weight = df[df['membership\_status'] == 1]['annual\_income'].dropna()

    stat, p\_value = ttest\_ind(gold\_weight , silver\_weight, equal\_var=False)

    print(f"t-test: статистика = {stat:.2f}, p-значение = {p\_value:.2e}")

    if p\_value < 0.05:

        print("Отвергаем гипотезу: доход за год различается у людей с разным статусом членства.")

    else:

        print("Не удалось отвергнуть гипотезу: доход за год не различается у людей с разным статусом членства.")

    print("\nГипотеза 2: Различается ли доход за год (annual\_income) у мужчин и женщин ?")

    man\_weight = df[df['gender'] == 0]['annual\_income'].dropna()

    woman\_weight = df[df['gender'] == 1]['annual\_income'].dropna()

    stat, p\_value = ttest\_ind(man\_weight , woman\_weight , equal\_var=False)

    print(f"t-test: статистика = {stat:.2f}, p-значение = {p\_value:.2e}")

    if p\_value < 0.05:

        print("Отвергаем гипотезу: доход за год различается у мужчин и женщин.")

    else:

        print("Не удалось отвергнуть гипотезу: доход за год не различается у мужчин и женщин.")

def correlation\_table(df, target\_column):

    #Построение таблицы корреляции.

    df = df.drop('id', axis=1)

    numeric\_columns = df.select\_dtypes(include=['float64', 'int64']).columns

    #numeric\_columns.delete("id")

    #del df["id"]

    correlation = df[numeric\_columns].corr()[target\_column].sort\_values(ascending=False)

    print("\nТаблица корреляции с целевым столбцом:")

    print(correlation)

    # Построение тепловой карты корреляций

    plt.figure(figsize=(10, 8))

    sns.heatmap(df[numeric\_columns].corr(), annot=True, cmap="coolwarm", fmt=".2f")

    plt.title("Корреляционная матрица")

    plt.show()

if \_\_name\_\_ == "\_\_main\_\_":

    # Данные для подключения

    host = "povt-cluster.tstu.tver.ru"

    port = 5432

    user = "mpi"

    password = "135a1"

    database = "Criminal"

    # Подключение к базе данных

    engine = connect\_to\_db(host, port, user, password, database)

    # Загрузка данных

    query = 'SELECT \* FROM person JOIN get\_fit\_now\_member ON person.id = get\_fit\_now\_member.person\_id JOIN drivers\_license ON person.license\_id = drivers\_license.id JOIN income ON income.ssn = person.ssn  LIMIT 1000'

    df = load\_data(engine, query)

    # Определение признаков

    nominal\_columns = [   "address\_street\_name",  "membership\_status", "eye\_color", "hair\_color", "gender", "car\_make", "car\_model"]

    ordinal\_columns = ["membership\_start\_date"]

    quantitative\_columns = ["age", "height","annual\_income"]

    exploratory\_analysis\_categorical(df,nominal\_columns )

    # Кодирование категориальных переменных

    df = encode\_categorical\_variables(df)

    # Анализ набора данных

    analyze\_dataset(df)

    exploratory\_analysis\_numeric(df, quantitative\_columns)

    # Проверка статистических гипотез

    test\_hypotheses(df)

    # Построение таблицы корреляции

    correlation\_table(df, target\_column="membership\_status")