Лабораторная работа 3

Задание 1. Построение диаграммы рассеяния

Вариант 2

**Цель работы:**

Построение диаграммы рассеяния для визуализации взаимосвязи между двумя факторами (возраст и индекс массы тела) в наборе данных о диабете, предоставленном библиотекой scikit-learn, с разделением данных на классы на основе категорий целевой переменной (уровень заболевания). Целью является визуальное исследование взаимосвязи между выбранными факторами и категориями, а также получение представления о структуре данных.

**Задачи:**

1. Загрузить набор данных diabetes из библиотеки sklearn.datasets.

2. Преобразовать данные в структуру pandas.DataFrame для удобства обработки.

3. Создать новую категориальную переменную bmi\_category на основе значений целевой переменной (target), используя предоставленную функцию categorize\_bmi.

4. Выбрать два фактора для построения диаграммы рассеяния: age (возраст) и bmi (индекс массы тела).

5. Определить классы на основе уникальных значений в столбце bmi\_category.

6. Построить диаграмму рассеяния с использованием библиотеки matplotlib.pyplot, где каждая категория представлена маркерами разного цвета.

7. Добавить подписи осей, заголовок и легенду для улучшения читаемости графика.

8. Отобразить график.

**Инструменты:**

• **Python 3:** Язык программирования.

• **matplotlib.pyplot:** Библиотека для создания графиков и диаграмм.

• **scikit-learn (sklearn):** Библиотека машинного обучения, используемая для загрузки набора данных diabetes.

• **pandas:** Библиотека для работы с табличными данными (DataFrame).

**Алгоритмы:**

1. Загрузка данных: Используется функция datasets.load\_diabetes() для загрузки набора данных из scikit-learn.

2. Преобразование в DataFrame: Создается pandas.DataFrame из загруженных данных.

3. Категоризация BMI: Определяется функция categorize\_bmi, которая классифицирует значения целевой переменной на три категории: "меньше 100", "от 100 до 200" и "больше 200".

4. Построение диаграммы рассеяния:

• Для каждой категории (класса) выбираются данные из DataFrame.

• Функция ax.scatter() используется для построения диаграммы рассеяния для каждой категории, где x - значения фактора age, y - значения фактора bmi, label - название категории, а c - цвет маркеров.

5. Оформление графика:

• Устанавливаются подписи осей с помощью ax.set\_xlabel() и ax.set\_ylabel().

• Устанавливается заголовок графика с помощью ax.set\_title().

• Отображается легенда с помощью ax.legend().

**Реализация кода (Python):**

import matplotlib.pyplot as plt

from sklearn import datasets

import pandas as pd

diabetes = datasets.load\_diabetes()

df = pd.DataFrame(data=diabetes.data, columns=diabetes.feature\_names)

df['target'] = diabetes.target

def categorize\_bmi(bmi):

if bmi < 100:

return "меньше 100"

elif 100 <= bmi <= 200:

return "от 100 до 200"

else:

return "больше 200"

df['bmi\_category'] = df['target'].apply(categorize\_bmi)

factor\_x = 'age'

factor\_y = 'bmi'

classes = df['bmi\_category'].unique()

fig, ax = plt.subplots()

colors = ['g', 'r', 'b']

for i, cls in enumerate(classes):

class\_data = df[df['bmi\_category'] == cls]

ax.scatter(class\_data[factor\_x], class\_data[factor\_y],

label=cls, c=colors[i])

ax.set\_xlabel(factor\_x)

ax.set\_ylabel(factor\_y)

ax.set\_title("Диаграмма рассеяния для Diabetes (по BMI)")

ax.legend()

plt.show()

**Выводы:**

• Программа успешно строит диаграмму рассеяния для выбранных факторов и классов в наборе данных о диабете.

• Визуализация позволяет получить представление о взаимосвязи между возрастом, индексом массы тела и уровнем заболевания.

• Анализ диаграммы рассеяния может помочь выявить закономерности и тенденции в данных, которые могут быть полезны для дальнейших исследований.

• Выбранный подход может быть применен для визуализации и анализа других наборов данных и факторов. Необходимо только изменить названия столбцов, функцию категоризации (если требуется) и цвета.

Задание 2 Построение графика динамики временных рядов

Вариант1 Построение графика динамики временных рядов CO2 (1958-1980)

**Цель работы:**

Визуализация динамики временного ряда концентрации CO2 в атмосфере за период с 1958 по 1980 год. Цель состоит в том, чтобы создать график, отображающий изменение концентрации CO2 во времени, используя данные из библиотеки statsmodels.

**Задачи:**

1. Загрузить набор данных co2: Использовать statsmodels.api.datasets.co2.load\_pandas() для загрузки данных в формате pandas DataFrame.

2. Преобразовать индекс в datetime: Преобразовать индекс DataFrame, представляющий даты, в формат datetime с использованием pd.to\_datetime(), чтобы облегчить работу с временными рядами.

3. Выбрать промежуток времени: Отфильтровать DataFrame, чтобы включить только данные за период с 1958 по 1980 год, используя срезы.

4. Построить график: Создать график временного ряда с использованием matplotlib.pyplot.plot(), отображающий значения CO2 по оси Y и время по оси X.

5. Настроить график: Добавить подписи к осям, заголовок, легенду и сетку для улучшения читаемости графика.

6. Отобразить график: Использовать matplotlib.pyplot.show() для отображения созданного графика.

**Инструменты:**

• **Python 3:** Основной язык программирования.

• matplotlib.pyplot**:** Библиотека для создания графиков.

• statsmodels.api**:** Библиотека для статистического моделирования и загрузки наборов данных.

• pandas**:** Библиотека для работы с данными в формате DataFrame.

**Алгоритмы:**

1. **Загрузка данных:**

• Использована функция sm.datasets.co2.load\_pandas() для загрузки данных о CO2 из библиотеки statsmodels в формате pandas DataFrame.

2. **Преобразование данных:**

• Индекс DataFrame, содержащий даты, был преобразован в формат datetime с использованием pd.to\_datetime(). Это необходимо для правильной интерпретации и отображения временных рядов.

3. **Выбор промежутка времени:**

• DataFrame был отфильтрован для выбора данных за период с 1958 по 1980 год с использованием среза df['1958':'1980'].

4. **Построение графика:**

• Был создан объект matplotlib.pyplot.figure для задания размеров графика.

• Использована функция plt.plot() для построения графика временного ряда. Индекс DataFrame (даты) использован для оси X, а значения CO2 - для оси Y. Добавлена метка для легенды.

5. **Оформление графика:**

• Добавлены подписи к осям X и Y с помощью plt.xlabel() и plt.ylabel(), чтобы пояснить, что отображено на каждой оси.

• Установлен заголовок графика с помощью plt.title() для указания темы графика.

• Добавлена легенда с помощью plt.legend() для пояснения, какие данные отображены на графике.

• Добавлена сетка с помощью plt.grid(True) для улучшения читаемости графика.

• Использована функция plt.tight\_layout() для автоматической регулировки размещения элементов графика.

6. **Отображение графика:**

• Функция plt.show() использована для отображения созданного графика.

**Реализация кода (Python):**

import matplotlib.pyplot as plt

import statsmodels.api as sm

import pandas as pd

# Загрузка данных

data = sm.datasets.co2.load\_pandas()

df = data.data

# Преобразование индекса в datetime

df.index = pd.to\_datetime(df.index)

# Выбор промежутка времени

df = df['1958':'1980']

# Построение графика

plt.figure(figsize=(12, 6))

plt.plot(df.index, df['co2'], label='CO2')

# Добавление подписей и заголовка

plt.xlabel("Год")

plt.ylabel("CO2 (ppm)")

plt.title("Динамика CO2 (1958-1980)")

plt.legend()

# Отображение графика

plt.grid(True)

plt.tight\_layout()

plt.show()

**Ошибки и их исправления:**

1. ModuleNotFoundError (отсутствие библиотеки statsmodels): Эта ошибка была исправлена путем установки библиотеки statsmodels с помощью pip install statsmodels.

2. Неправильный формат даты: Ошибка была устранена путем преобразования индекса DataFrame в формат datetime с использованием pd.to\_datetime(). Это обеспечило правильную интерпретацию временных рядов.

**Выводы:**

• В результате работы был создан график динамики временного ряда концентрации CO2 в атмосфере за период с 1958 по 1980 год.

• График позволяет визуально оценить изменение концентрации CO2 со временем и выявить тренды и закономерности.