МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«БРЕСТСКИЙГОСУДАРСТВЕННЫЙТЕХНИЧЕСКИЙУНИВЕРСИТЕТ»

ФАКУЛЬТЕТ ЭЛЕКТРОННО-ИНФОРМАЦИОННЫХ СИСТЕМ

Кафедра интеллектуальных информационных технологий

Отчет по лабораторной работе №1

Специальность ИИ-23

Выполнил

А.П. Романюк,

студент группы ИИ-23

Проверила

К.В. Андреенко,

« » 2025 г.

Брест 2025

Цель работы: научиться применять метод PCA для осуществления визуализации данных

**Общее задание**

1. Используя выборку по варианту, осуществить проецирование данных на плоскость первых двух и трех главных компонент (двумя способами: 1. вручную через использование numpy.linalg.eig для вычисления собственных значений и собственных векторов и 2. с помощью sklearn.decomposition.PCA для непосредственного применения метода PCA – два независимых варианта решения);

2. Выполнить визуализацию полученных главных компонент с использованием средств библиотеки matplotlib, обозначая экземпляры разных классов с использованием разных цветовых маркеров;

3. Используя собственные значения, рассчитанные на этапе 1, вычислить потери, связанные с преобразованием по методу PCA. Сделать выводы;

4. Оформить отчет по выполненной работе, загрузить исходный код и отчет в соответствующий репозиторий на github.

**Задание по варианту:**

| **№ варианта** | **Выборка** | **Класс** |
| --- | --- | --- |
| 9 | heart+failure+clinical+records.zip | death\_event |

Ход работы:

import pandas as pd  
import numpy as np  
import matplotlib.pyplot as plt  
from sklearn.preprocessing import StandardScaler  
  
raw\_data = pd.read\_csv('data.csv')  
raw\_data.head()  
  
labels = raw\_data['DEATH\_EVENT']  
data = raw\_data.drop(columns=['DEATH\_EVENT'])  
data  
  
scaler = StandardScaler()  
data = scaler.fit\_transform(data)  
data  
  
data\_centred = data - data.mean()  
cov\_matrix = np.cov(data\_centred, rowvar=False)  
eig\_values, eig\_vectors = np.linalg.eig(cov\_matrix)  
idx = np.argsort(eig\_values)[::-1]  
eig\_vectors = eig\_vectors[:,idx]  
  
data\_2d = data\_centred.dot(eig\_vectors[:,:2])  
data\_3d = data\_centred.dot(eig\_vectors[:,:3])  
  
plt.figure(figsize=(8, 6))  
scatter = plt.scatter(data\_2d[:, 0], data\_2d[:, 1], c=labels, cmap='viridis')  
plt.colorbar(scatter)  
plt.xlabel('Первая главная компонента')  
plt.ylabel('Вторая главная компонента')  
plt.title('PCA проекция на плоскость первых двух компонент')  
plt.show()  
Изображение выглядит как снимок экрана, Красочность, текст

Содержимое, созданное искусственным интеллектом, может быть неверным.  
fig = plt.figure(figsize=(8, 6))  
ax = fig.add\_subplot(111, projection='3d')  
scatter = ax.scatter(data\_3d[:, 0], data\_3d[:, 1], data\_3d[:, 2], c=labels, cmap='viridis')  
fig.colorbar(scatter)  
ax.set\_xlabel('Первая главная компонента')  
ax.set\_ylabel('Вторая главная компонента')  
ax.set\_zlabel('Третья главная компонента')  
ax.set\_title('PCA проекция на плоскость первых трёх компонент')  
plt.show()  
Изображение выглядит как снимок экрана, текст, диаграмма

Содержимое, созданное искусственным интеллектом, может быть неверным.  
from sklearn.decomposition import PCA  
  
pca\_2d = PCA(n\_components=2)  
data\_2d = pca\_2d.fit\_transform(data)  
  
pca\_3d = PCA(n\_components=3)  
data\_3d = pca\_3d.fit\_transform(data)  
  
plt.figure(figsize=(8, 6))  
scatter = plt.scatter(data\_2d[:, 0], data\_2d[:, 1], c=labels, cmap='viridis')  
plt.colorbar(scatter)  
plt.xlabel('Первая главная компонента')  
plt.ylabel('Вторая главная компонента')  
plt.title('PCA проекция на плоскость первых двух компонент')  
plt.show()  
Изображение выглядит как снимок экрана, текст, Красочность, фиолетовый

Содержимое, созданное искусственным интеллектом, может быть неверным.  
fig = plt.figure(figsize=(8, 6))  
ax = fig.add\_subplot(111, projection='3d')  
scatter = ax.scatter(data\_3d[:, 0], data\_3d[:, 1], data\_3d[:, 2], c=labels, cmap='viridis')  
fig.colorbar(scatter)  
ax.set\_xlabel('Первая главная компонента')  
ax.set\_ylabel('Вторая главная компонента')  
ax.set\_zlabel('Третья главная компонента')  
ax.set\_title('PCA проекция на плоскость первых трёх компонент')  
plt.show()  
  
total\_var = np.sum(eig\_values)  
  
explained\_var\_ratio = np.cumsum(sorted(eig\_values, reverse=True)) / total\_var  
  
loss\_2d = 1 - explained\_var\_ratio[1]  
loss\_3d = 1 - explained\_var\_ratio[2]  
  
print("Потери при 2D проекции:", loss\_2d)  
print("Потери при 3D проекции:", loss\_3d)

Изображение выглядит как текст, снимок экрана, диаграмма

Содержимое, созданное искусственным интеллектом, может быть неверным.

Вывод: научился применять метод PCA для осуществления визуализации данных