Министерство образования Республики Беларусь
Учреждение образования
«Брестский государственный технический университет»
Кафедра ИИТ

Лабораторная работа №1

По дисциплине: «Языковые процессы интеллектуальных систем»

Тема: «**РСА**»

Выполнил:
Студент 4 курса
Группы ИИ-23
Ежевский Е.Р.
Проверила:
Андренко К.В.

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:

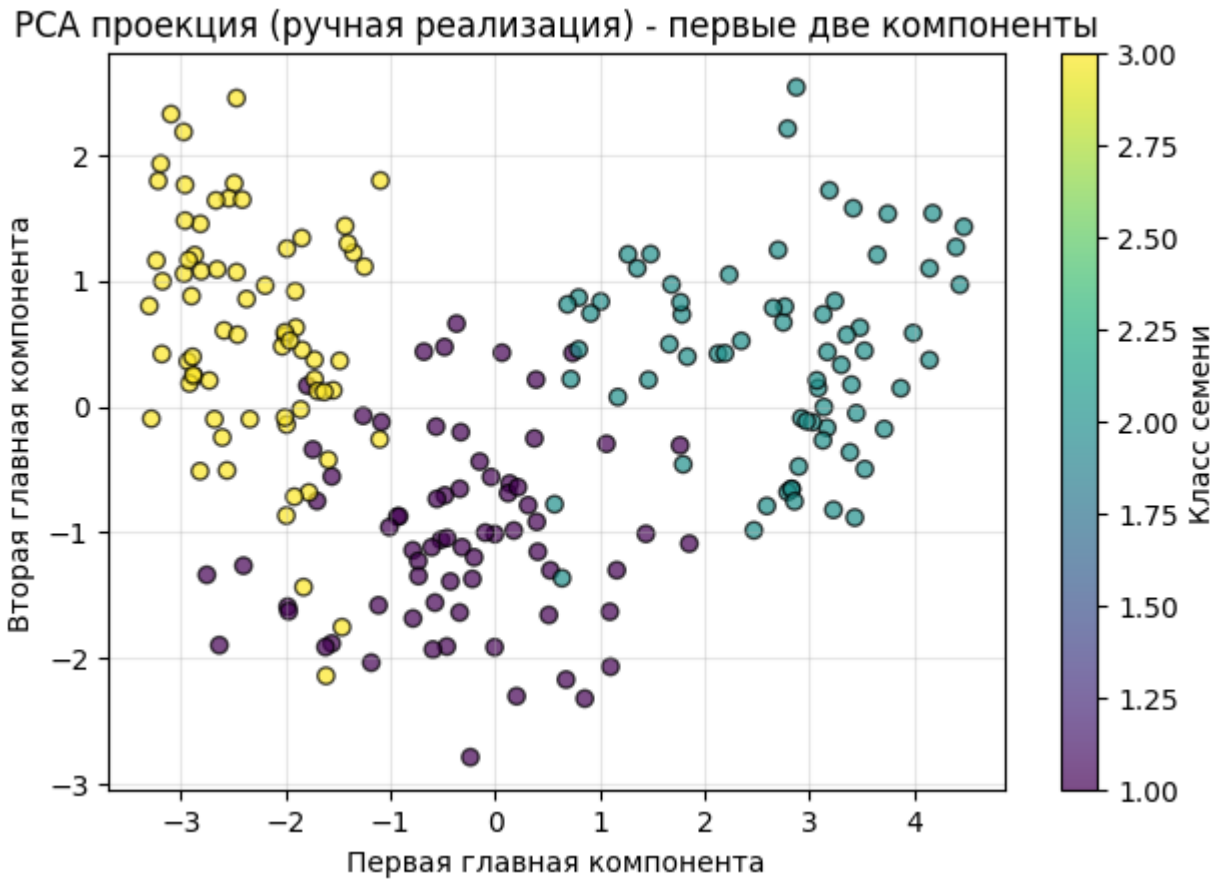
№ варианта	Выборка	Класс
1	seeds.zip	Последняя колонка

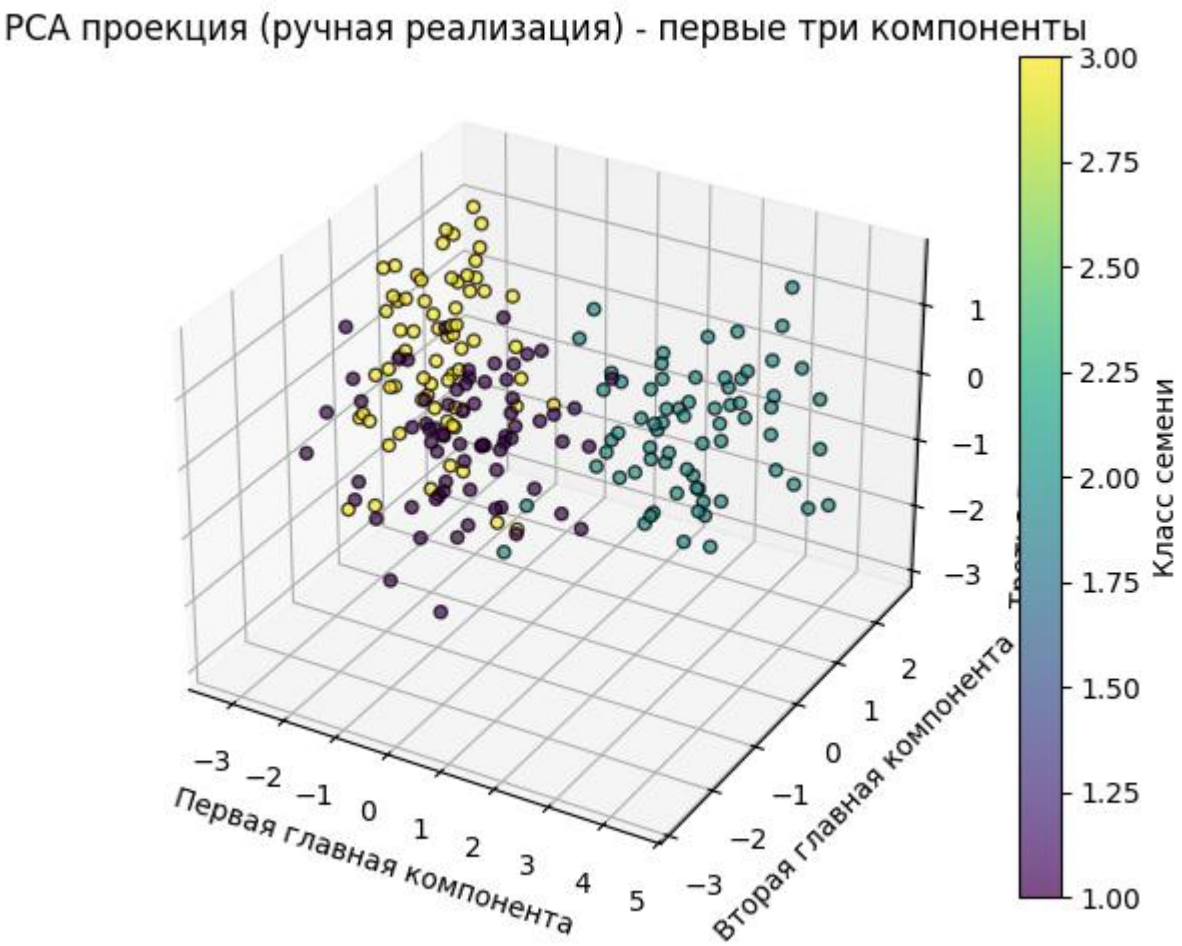
Ход работы:

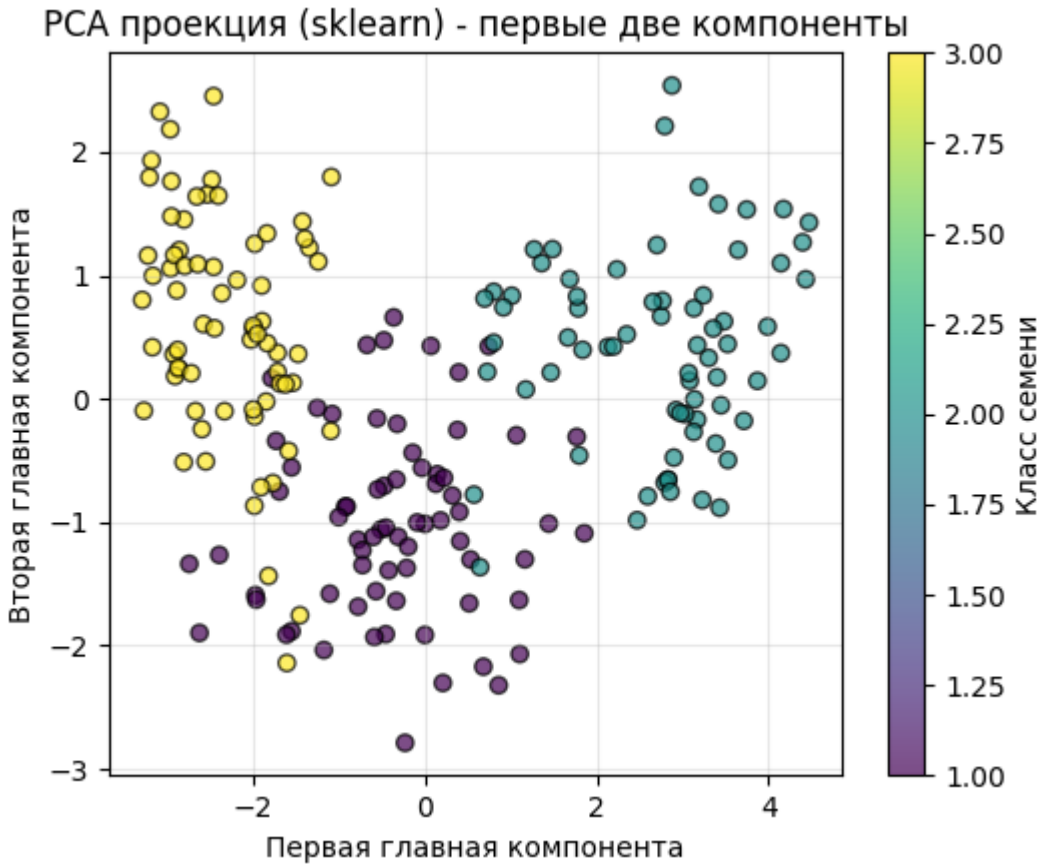
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA

raw_data = pd.read_csv('seeds_dataset.txt', sep='\s+', header=None)
raw_data.head()

labels = raw_data.iloc[:, -1]
data = raw_data.iloc[:, :-1]
print("Размерность данных:", data.shape)
print("Количество классов:", labels.nunique())

scaler = StandardScaler()
data_scaled = scaler.fit_transform(data)

print("\n=== Ручная реализация PCA ===")
data_centred = data_scaled - data_scaled.mean(axis=0)
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]
eig_values = eig_values[idx]

data_2d_manual = data_centred.dot(eig_vectors[:, :2])
```

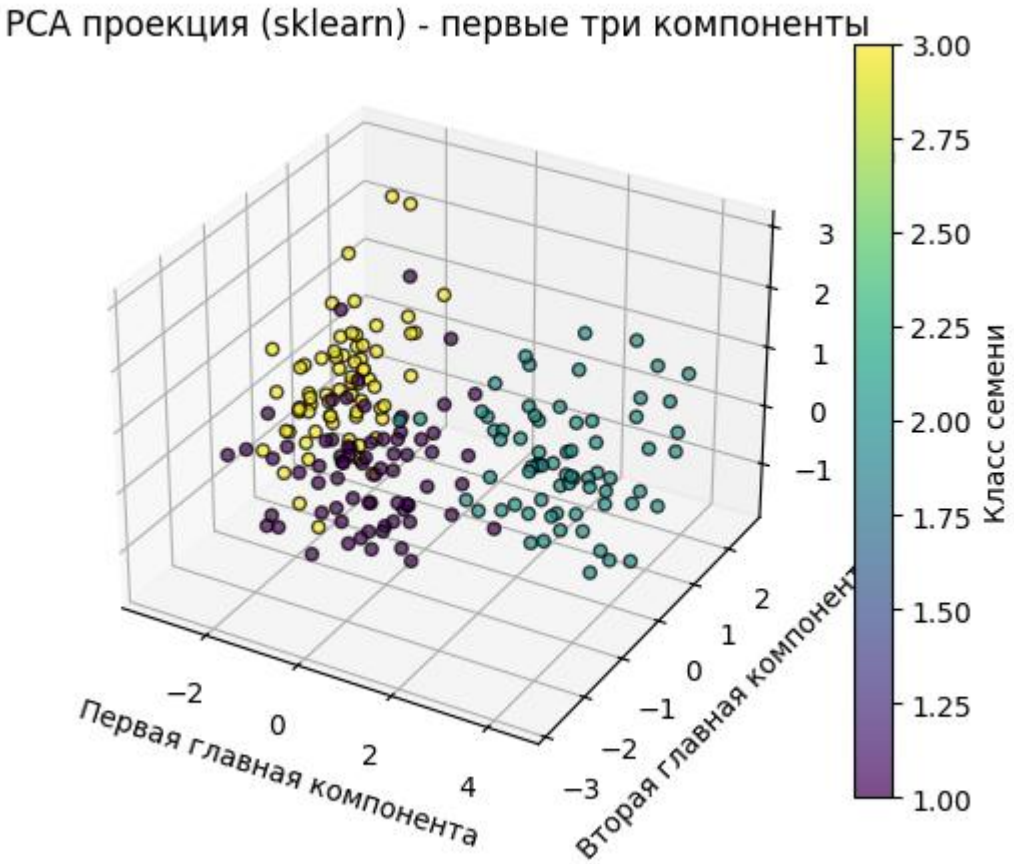
```
data_3d_manual = data_centred.dot(eig_vectors[:, :3])

plt.figure(figsize=(10, 8))
scatter = plt.scatter(data_2d_manual[:, 0], data_2d_manual[:, 1],
                      c=labels, cmap='viridis', alpha=0.7, edgecolors='k')
plt.colorbar(scatter, label='Класс семени')
plt.xlabel('Первая главная компонента')
plt.ylabel('Вторая главная компонента')
plt.title('PCA проекция (ручная реализация) - первые две компоненты')
plt.grid(True, alpha=0.3)
plt.show()

fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111, projection='3d')
scatter = ax.scatter(data_3d_manual[:, 0], data_3d_manual[:, 1],
                     data_3d_manual[:, 2], c=labels, cmap='viridis',
                     alpha=0.7, edgecolors='k')
fig.colorbar(scatter, label='Класс семени')
ax.set_xlabel('Первая главная компонента')
ax.set_ylabel('Вторая главная компонента')
ax.set_zlabel('Третья главная компонента')
ax.set_title('PCA проекция (ручная реализация) - первые три компоненты')
plt.show()

print("\n=== PCA с использованием sklearn ===")
pca_2d = PCA(n_components=2)
data_2d_sklearn = pca_2d.fit_transform(data_scaled)

pca_3d = PCA(n_components=3)
data_3d_sklearn = pca_3d.fit_transform(data_scaled)

plt.figure(figsize=(10, 8))
scatter = plt.scatter(data_2d_sklearn[:, 0], data_2d_sklearn[:, 1],
                      c=labels, cmap='viridis', alpha=0.7, edgecolors='k')
plt.colorbar(scatter, label='Класс семени')
plt.xlabel('Первая главная компонента')
plt.ylabel('Вторая главная компонента')
plt.title('PCA проекция (sklearn) - первые две компоненты')
plt.grid(True, alpha=0.3)
plt.show()

fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111, projection='3d')
scatter = ax.scatter(data_3d_sklearn[:, 0], data_3d_sklearn[:, 1],
                     data_3d_sklearn[:, 2], c=labels, cmap='viridis',
                     alpha=0.7, edgecolors='k')
fig.colorbar(scatter, label='Класс семени')
ax.set_xlabel('Первая главная компонента')
ax.set_ylabel('Вторая главная компонента')
ax.set_zlabel('Третья главная компонента')
ax.set_title('PCA проекция (sklearn) - первые три компоненты')
plt.show()
```

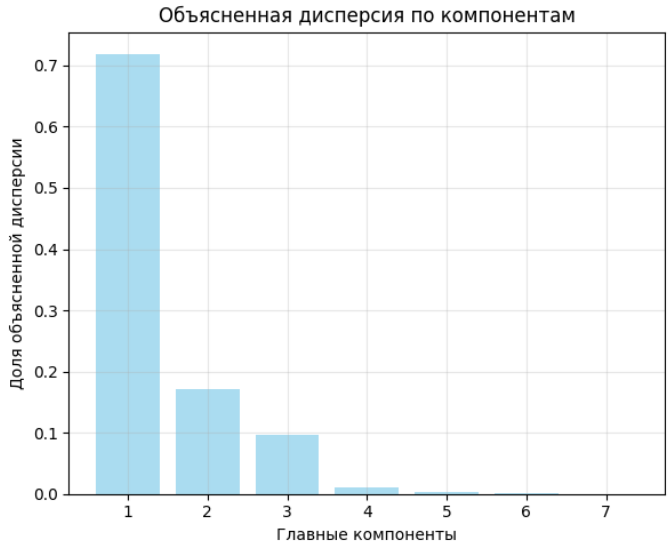
```
print("\n=== Анализ объясненной дисперсии ===")
total_var = np.sum(eig_values)
explained_var_ratio = eig_values / total_var
cumulative_var_ratio = np.cumsum(explained_var_ratio)

loss_2d = 1 - cumulative_var_ratio[1]
loss_3d = 1 - cumulative_var_ratio[2]

print(f"Объясненная дисперсия по компонентам:")
for i, (var, cum_var) in enumerate(zip(explained_var_ratio, cumulative_var_ratio), 1):
    print(f"Компонента {i}: {var:.3%} (кумулятивно: {cum_var:.3%})")

print(f"\nПотери при 2D проекции: {loss_2d:.3%}")
print(f"Потери при 3D проекции: {loss_3d:.3%}")

plt.figure(figsize=(12, 5))

plt.subplot(1, 2, 1)
bars = plt.bar(range(1, len(explained_var_ratio) + 1), explained_var_ratio,
               alpha=0.7, color='skyblue')
plt.xlabel('Главные компоненты')
plt.ylabel('Доля объясненной дисперсии')
plt.title('Объясненная дисперсия по компонентам')
plt.grid(True, alpha=0.3)

plt.subplot(1, 2, 2)
plt.plot(range(1, len(cumulative_var_ratio) + 1), cumulative_var_ratio,
         marker='o', linewidth=2)
plt.axhline(y=0.95, color='r', linestyle='--', alpha=0.7, label='95% дисперсии')
plt.axhline(y=0.90, color='g', linestyle='--', alpha=0.7, label='90% дисперсии')
plt.xlabel('Количество компонент')
plt.ylabel('Кумулятивная доля дисперсии')
plt.title('Кумулятивная объясненная дисперсия')
plt.legend()
plt.grid(True, alpha=0.3)
plt.tight_layout()
plt.show()

print("\n=== Сравнение реализаций ===")
print("Совпадение объясненной дисперсии (первые 2 компоненты):")
print(f"Ручная: {explained_var_ratio[:2].sum():.3%}")
print(f"Sklearn: {pca_2d.explained_variance_ratio_.sum():.3%}")
print(f"Разница: {abs(explained_var_ratio[:2].sum() -
pca_2d.explained_variance_ratio_.sum()):.6%}")
```

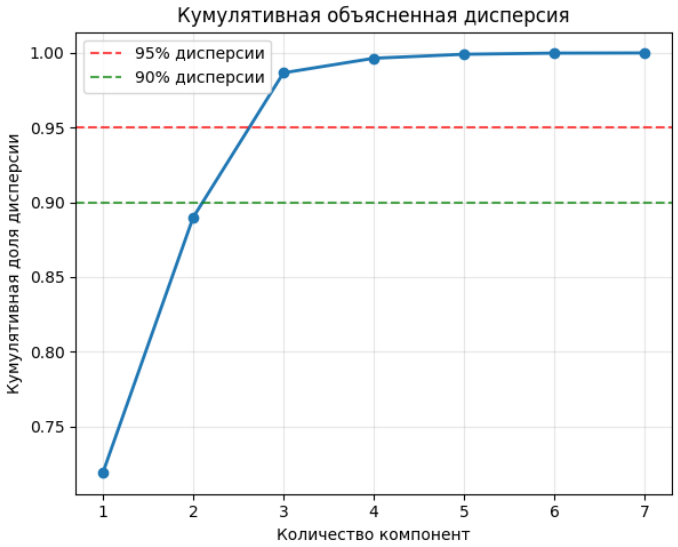

PCA проекция (ручная реализация) - первые две компоненты
Вторая главная компонента
Первая главная компонента
Класс семени
3.00
2.75
2.50
2.25
2.00
1.75
1.50
1.25
1.00


PCA проекция (ручная реализация) - первые три компоненты
Первая главная компонента
Вторая главная компонента
Класс семени
3.00
2.75
2.50
2.25
2.00
1.75
1.50
1.25
1.00


PCA проекция (sklearn) - первые две компоненты
Вторая главная компонента
Первая главная компонента
Класс семени
3.00
2.75
2.50
2.25
2.00
1.75
1.50
1.25
1.00


PCA проекция (sklearn) - первые три компоненты
Первая главная компонента
Вторая главная компонент
Класс семени
3.00
2.75
2.50
2.25
2.00
1.75
1.50
1.25
1.00





Вывод: научился применять метод PCA для осуществления визуализации данных