

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В. И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра математического обеспечения и применения ЭВМ

ОТЧЕТ
по лабораторной работе №2
по дисциплине «Машинное обучение»

Тема: Понижение размерности пространства признаков

Студент гр. 1310

Комаров Д. Е.

Преподаватель

Жангиров Т.Р.

Санкт-Петербург

2025

Постановка задачи

Цель работы: ознакомление с методами понижения размерности данных из библиотеки Scikit Learn.

Выполнение лабораторной работы

Загрузка данных

Для выполнения лабораторной работы используем набор данных «Glass Classification». В данном наборе данных содержит 214 наблюдений содержания 9 различных элементов в 7 видах стекла. Приведем значения каждого признака к интервалу [0;1] и построим диаграммы рассеяния пар признаков. Полученные диаграммы представлены на рисунке 1. Цвет точек показывает, к какому виду стекла относится наблюдение.

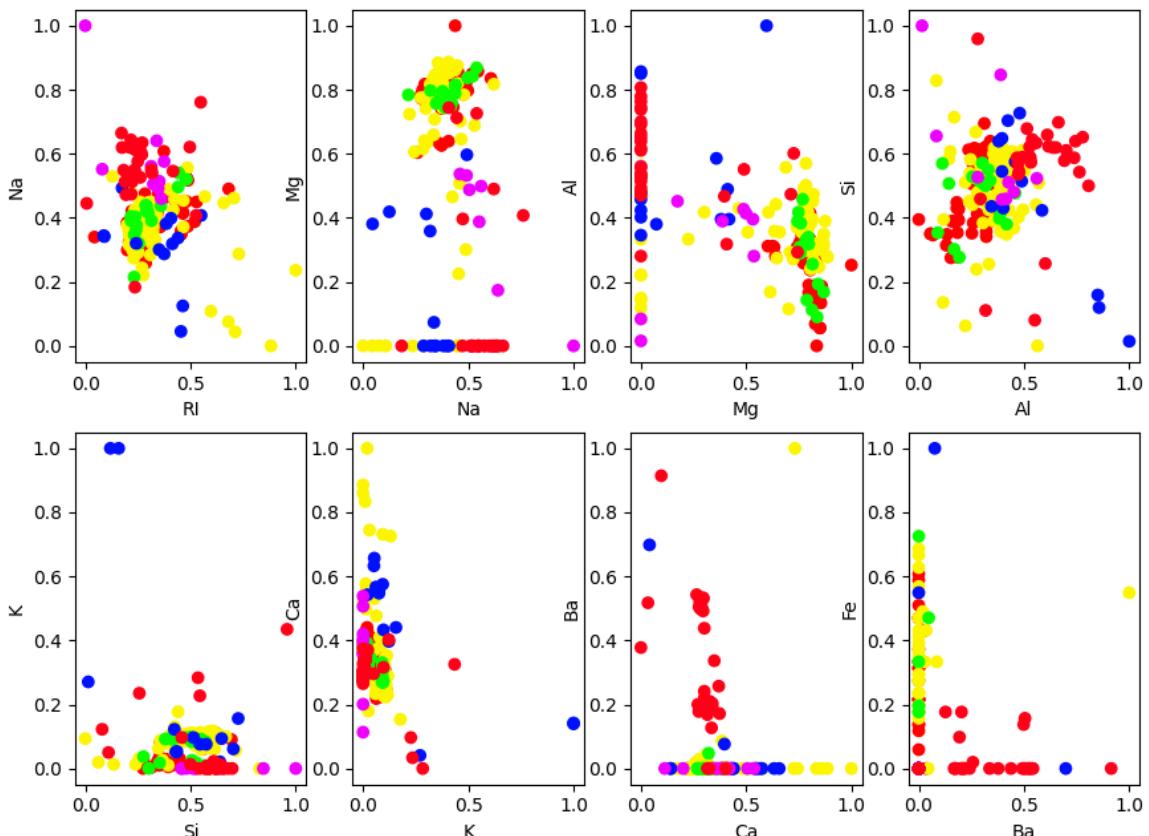


Рисунок 1 – Диаграммы рассеяния пар признаков

Метод главных компонент

Суть метода главных компонент заключается в линейном преобразовании данных в пространство меньшей размерности так, чтобы компоненты пространства меньшей размерности максимально сохраняли

дисперсию исходных данных. Воспользуемся методом главных компонент для понижения размерности пространства признаков до 2. Диаграмма рассеяния после применения данного метода представлена на рисунке 2.

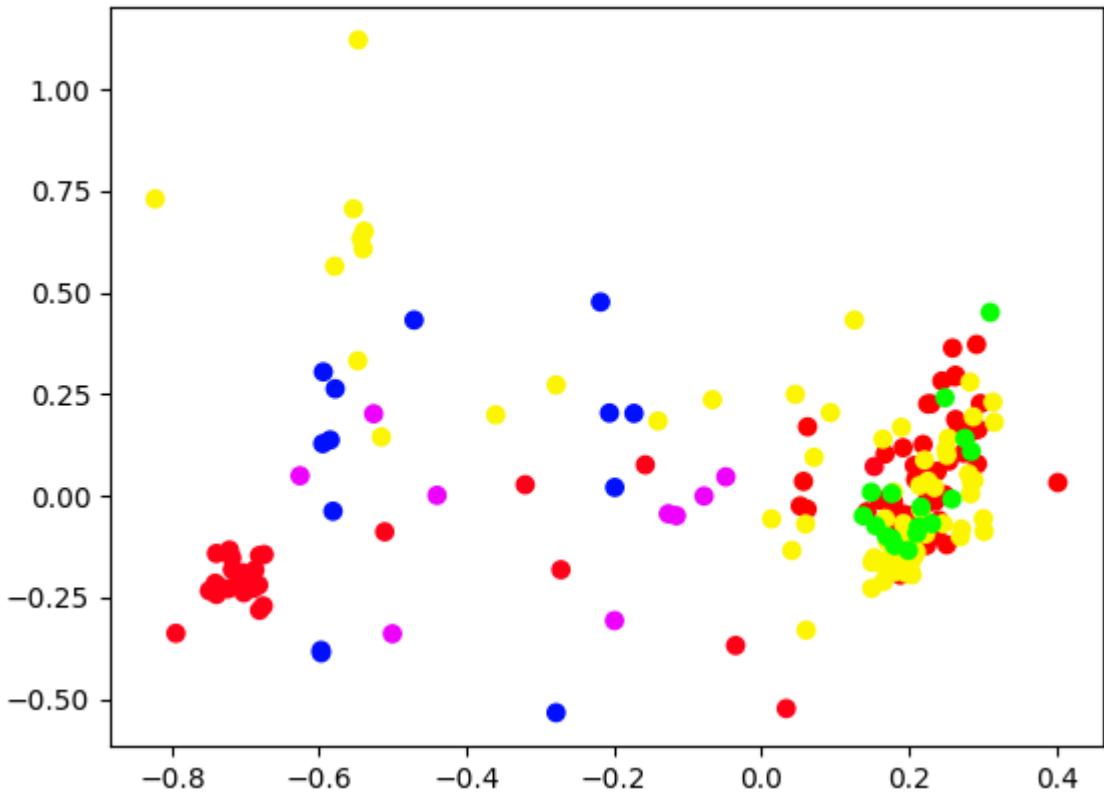


Рисунок 2 – Диаграмма рассеяния после понижения размерности до 2 методом главных компонент

Значения объясненной дисперсии в процентах для двух получившихся компонент равны 45% и 18%. Собственные числа же равны 5.1 и 3.21. Таким образом, две компоненты сохранили только 63% суммарной дисперсии исходных данных. Подберём такую размерность, при которой сохраняется более 85% суммарной дисперсии исходных данных. Такой размерностью является 4 компоненты, поскольку при понижении размерности до 4 компонент сохраняется 86% суммарной дисперсии исходных данных. На рисунке 3 представлены диаграммы рассеяния после понижения размерности до 4 компонент.

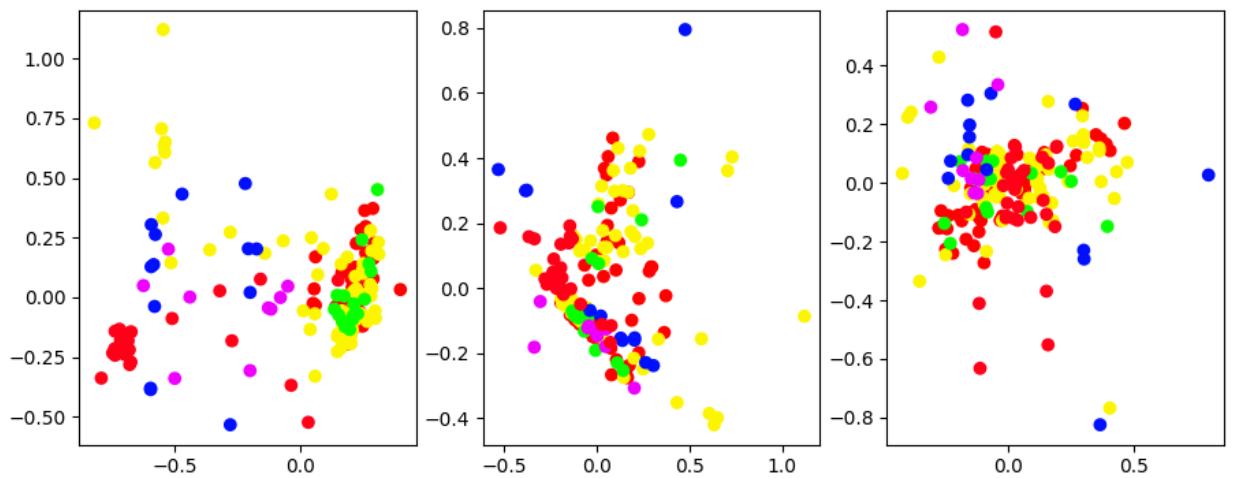


Рисунок 3 – Диаграммы рассеяния после понижения размерности до 4 методом главных компонент

Воспользуемся методом *inverse_transform* для восстановления исходных данных из данных, полученных после понижения размерности до 4 методом главных компонент. Диаграммы рассеяния для полученных данных представлены на рисунке 4.

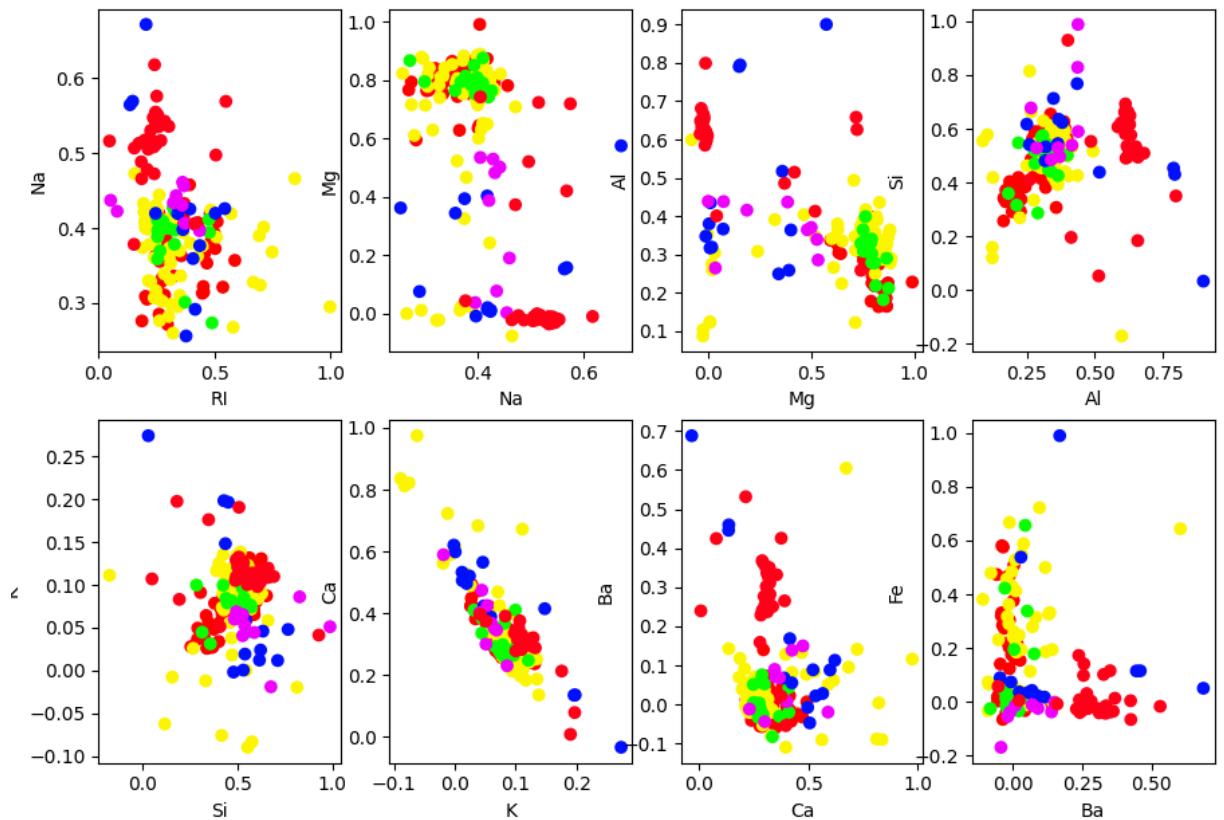


Рисунок 4 – Диаграммы рассеяния пар исходных признаков после обратного преобразования данных с пониженной до 4 размерностью

Сравнив рисунки 1 и 4, можно заметить, что, после понижения размерности до 4 и обратного преобразования, данные не вернулись к исходному состоянию, поскольку при понижении размерности частичная потеря в точности неизбежна. Однако можно заметить, что в целом, данные после преобразований распределены схоже с исходными, из чего следует, что пространство пониженной размерности по большей части достаточно для описания распределения данных в исходном пространстве.

Помимо количества компонент, в методе главных компонент можно использовать различные методы сингулярного разложения. По умолчанию он выбирается автоматически, однако его можно задать вручную. Протестируем понижение размерности методом главных компонент до 2 при `svd_solver='full'`. На рисунке 5 представлены полученные диаграммы рассеяния.

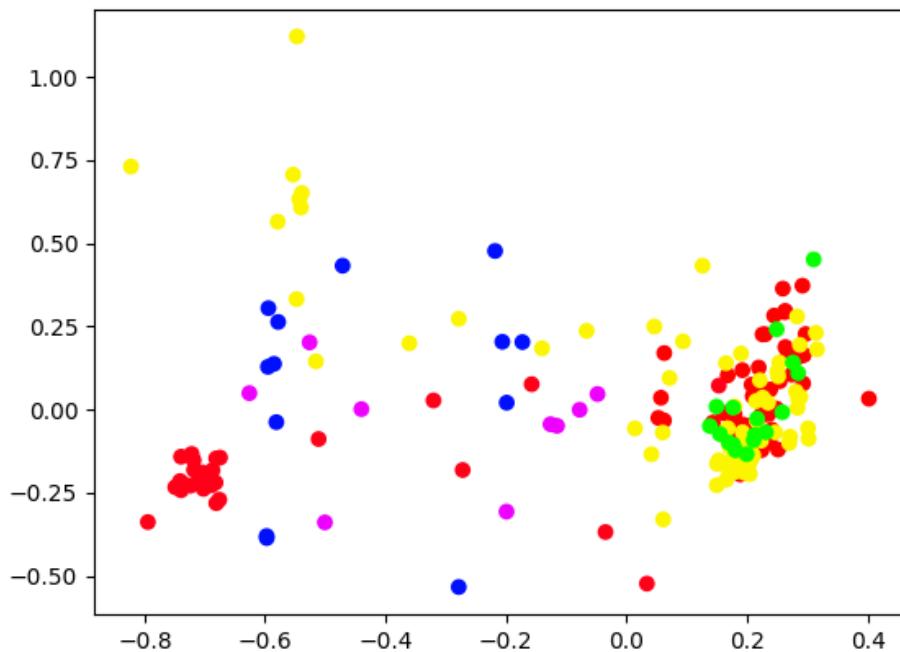


Рисунок 5 – Диаграммы рассеяния после понижения размерности до 2 методом главных компонент при `svd_solver='full'`

На рисунке 6 представлены Диаграммы рассеяния после понижения размерности до 2 методом главных компонент при `svd_solver='covariance_eigh'`.

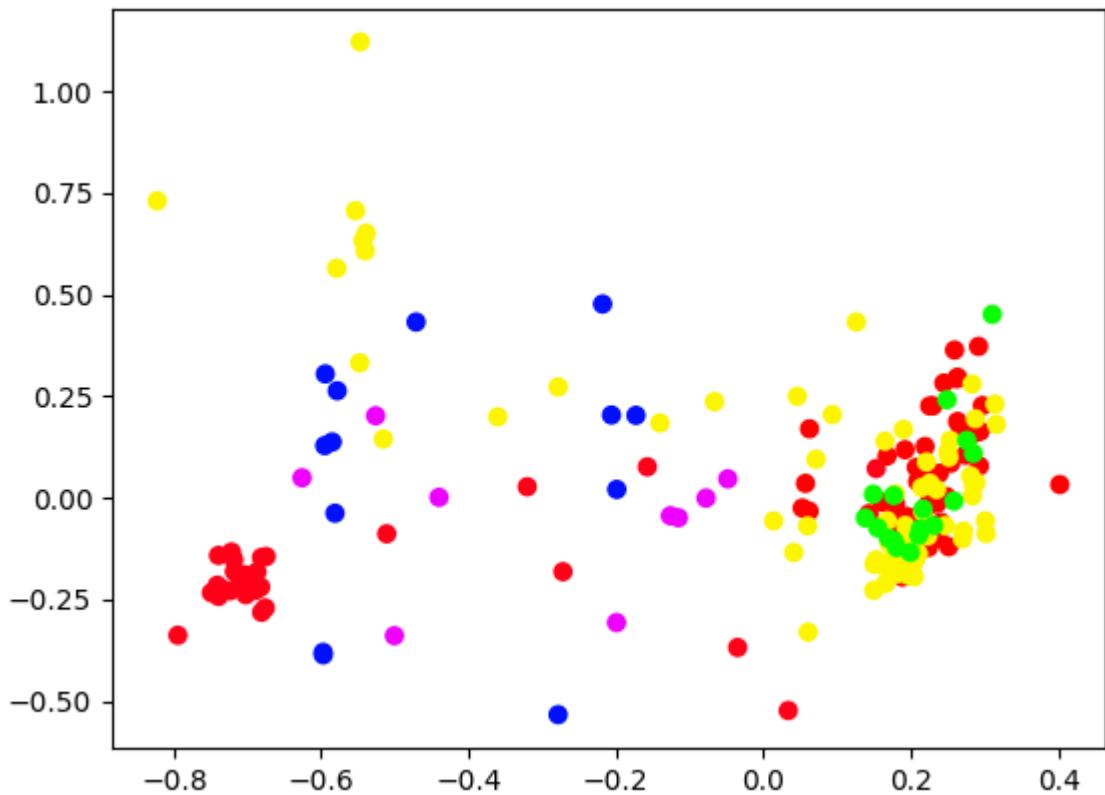


Рисунок 6 – Диаграммы рассеяния после понижения размерности до 2 методом главных компонент при $svd_solver='covariance_eigh'$

На рисунке 7 представлены диаграммы рассеяния после понижения размерности до 2 методом главных компонент при $svd_solver='arpack'$.

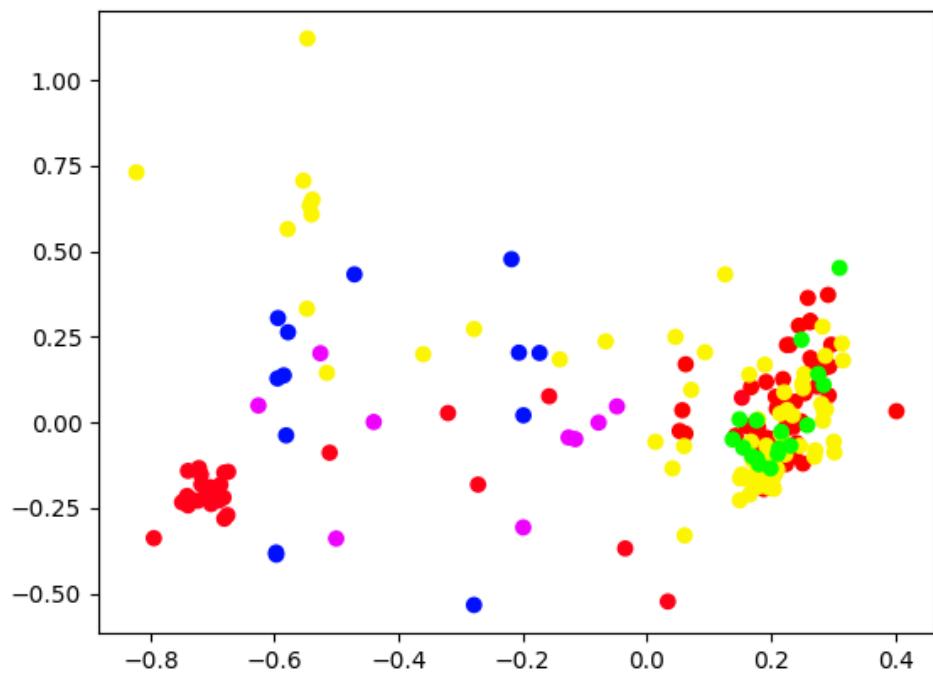


Рисунок 7 – Диаграммы рассеяния после понижения размерности до 2 методом главных компонент при $svd_solver='arpack'$

На рисунке 8 представлены Диаграммы рассеяния после понижения размерности до 2 методом главных компонент при $svd_solver='randomized'$.

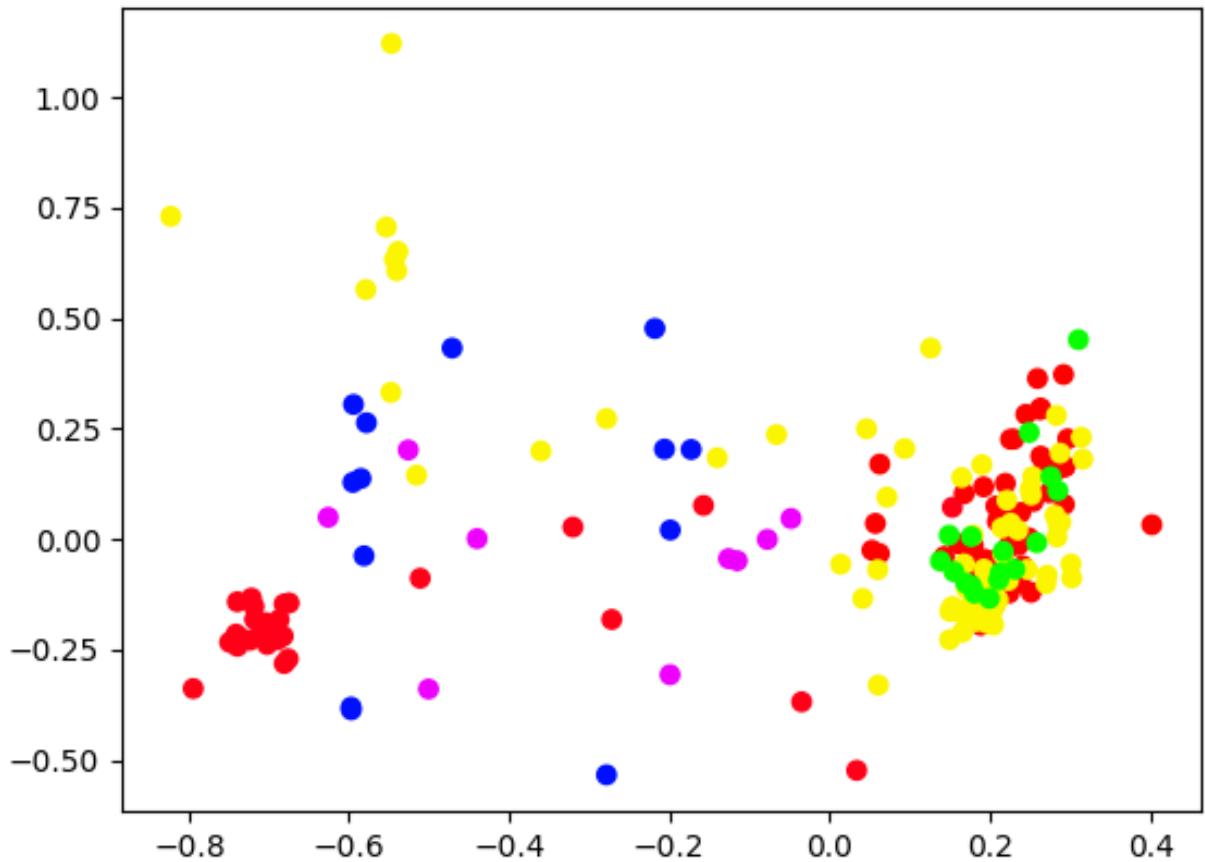


Рисунок 8 – Диаграммы рассеяния после понижения размерности до 2 методом главных компонент при $svd_solver='randomized'$

Модификации метода главных компонент

Воспользуемся ядерным методом главных компонент для понижения размерности. Данный метод отличается от обычного метода главных компонент тем, что собственные числа и собственные вектора вычисляются для ядерной матрицы. За счет этого преобразование данных в пространство меньшей размерности нелинейно. В данном методе для построения ядерной матрицы могут использоваться различные ядра с различными параметрами.

Линейное ядро представляет собой функцию

$$K(x_1, x_2) = x_1^T x_2.$$

На рисунке 9 представлена диаграмма рассеяния после понижения размерности до 2 ядерным методом главных компонент с применением линейного ядра.

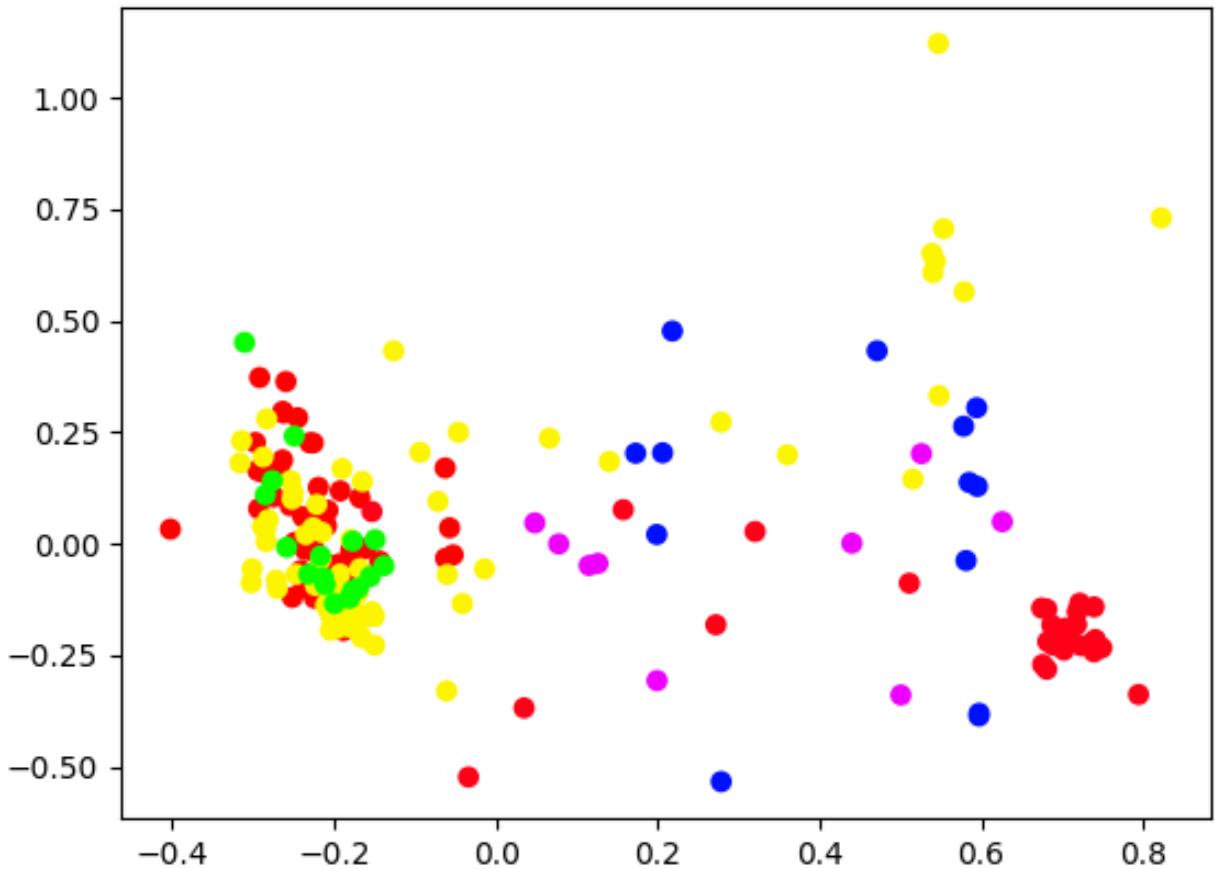


Рисунок 9 – Диаграмма рассеяния после понижения размерности до 2 ядерным методом главных компонент с применением линейного ядра

Сравнив рисунки 2 и 9 можно прийти к выводу, что результаты, полученные ядерным методом главных компонент с применением линейного ядра и обычным методом главных компонент, аналогичны, за исключением того, что данные на одной из осей пространства пониженной размерности в обоих методах отличаются знаком, что не является существенным.

Полиномиальное ядро представляет собой функцию

$$K(x_1, x_2) = (\gamma * x_1^T x_2 + \text{coef0})^{\text{degree}},$$

где параметры γ , coef0 , degree могут варьироваться.

На рисунке 10 представлена диаграмма рассеяния после понижения размерности до 2 ядерным методом главных компонент с применением полиномиального ядра с параметрами $\gamma=2$, $\text{coef0}=1$, $\text{degree}=4$.

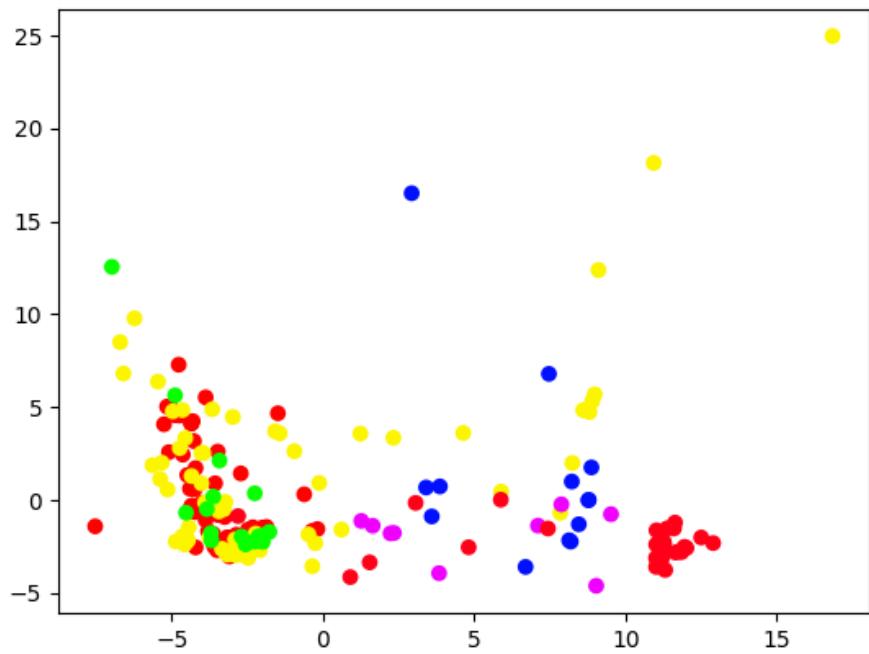


Рисунок 10 – Диаграмма рассеяния после понижения размерности до 2 ядерным методом главных компонент с применением полиномиального ядра

Изменим параметры на $gamma=1$, $coef0=0$, $degree=5$. Результаты после изменения параметров представлены на рисунке 11.

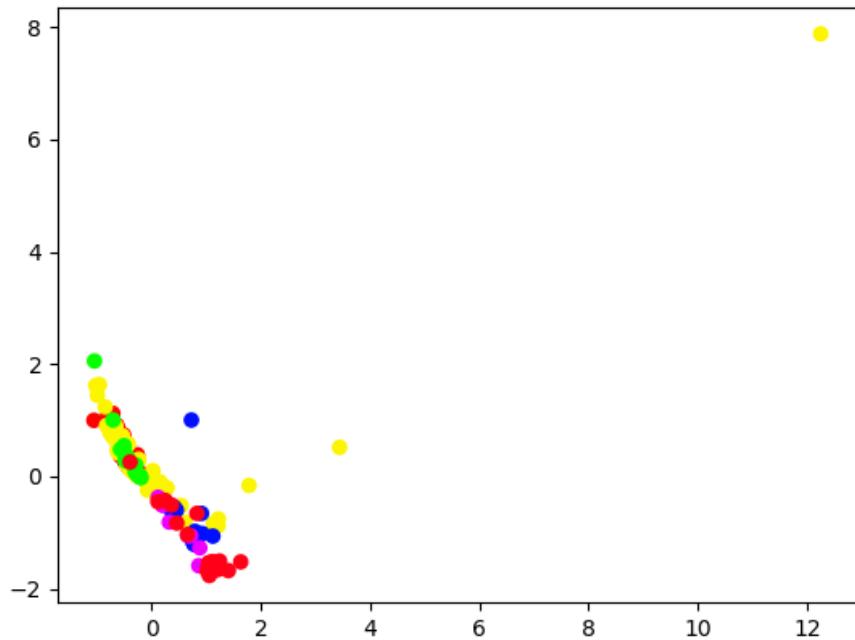


Рисунок 11 – Диаграмма рассеяния после понижения размерности до 2 ядерным методом главных компонент с применением полиномиального ядра после изменения параметров

Ядро может быть представлено радиальной базисной функцией

$$K(x_1, x_2) = e^{-gamma * \|x_1 - x_2\|^2},$$

где параметры $gamma$ может варьироваться.

На рисунке 12 представлена диаграмма рассеяния после понижения размерности до 2 ядерным методом главных компонент с применением ядра в виде радиальной базисной функции с параметром $gamma=1$.

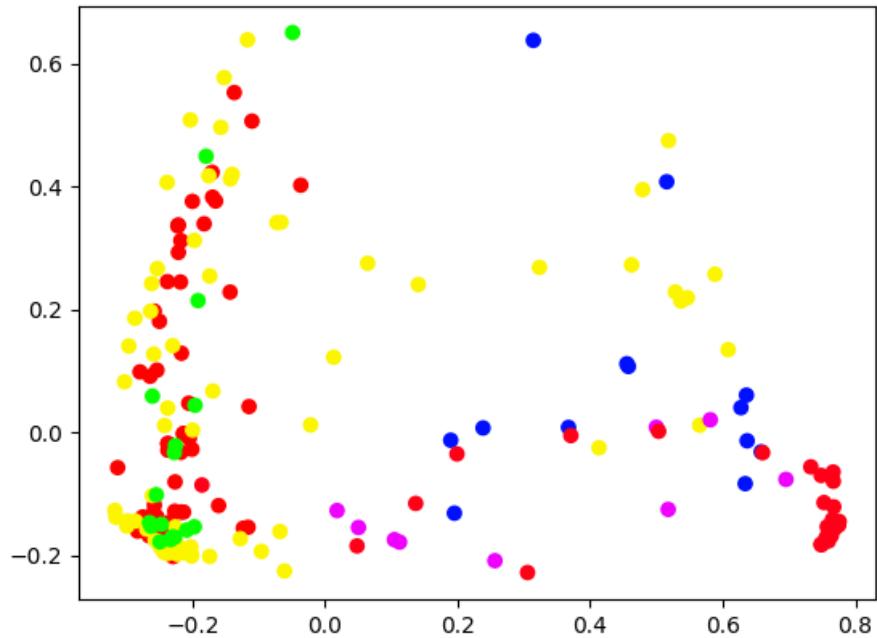


Рисунок 12 – Диаграмма рассеяния после понижения размерности до 2 ядерным методом главных компонент с применением ядра в виде радиальной базисной функции

Изменим параметр на $gamma=2$. Результаты после изменения параметров представлены на рисунке 13.

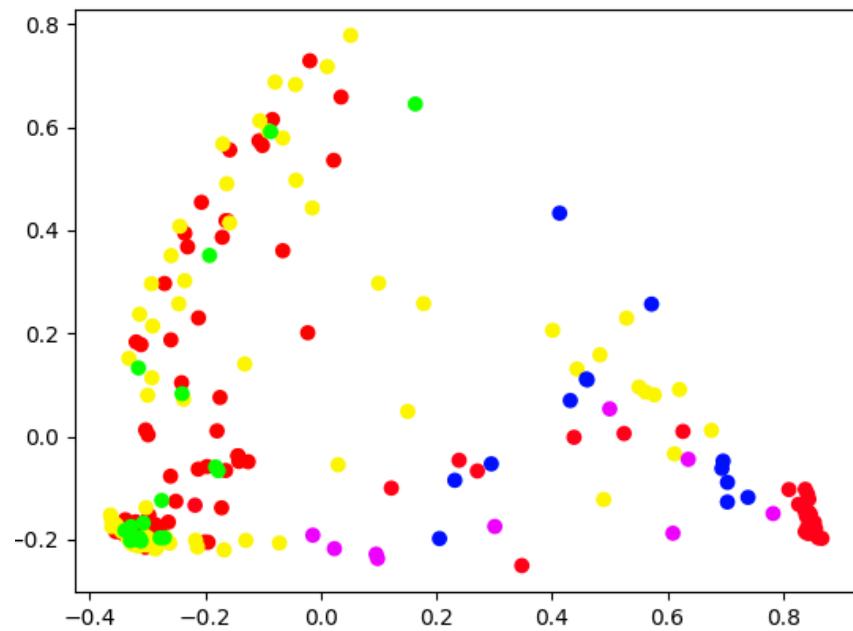


Рисунок 13 – Диаграмма рассеяния после понижения размерности до 2 ядерным методом главных компонент с применением ядра в виде радиальной базисной функции после изменения параметров

Сигмоидальное ядро представляет собой функцию

$$K(x_1, x_2) = \tanh(\gamma * x_1^T x_2 + \text{coef0}),$$

где параметры γ , coef0 могут варьироваться.

На рисунке 14 представлена диаграмма рассеяния после понижения размерности до 2 ядерным методом главных компонент с применением сигмоидального ядра с параметром $\gamma=1$ и $\text{coef0}=0$.

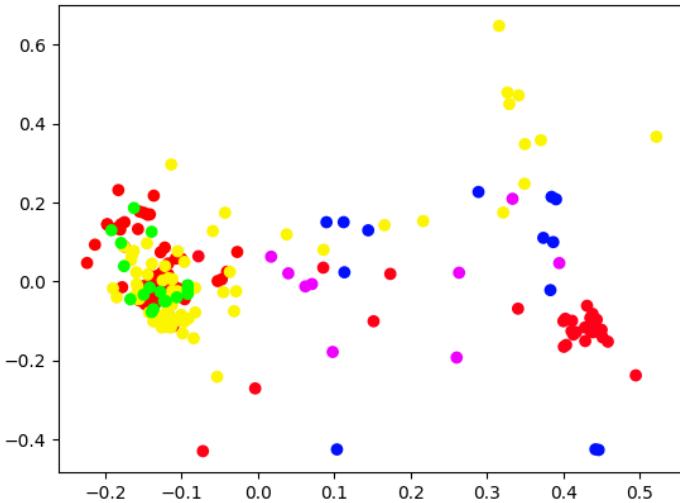


Рисунок 14 – Диаграмма рассеяния после понижения размерности до 2 ядерным методом главных компонент с применением сигмоидального ядра

Изменим параметры на $\gamma=2$ и $\text{coef0}=3$. Результаты после изменения параметров представлены на рисунке 15.

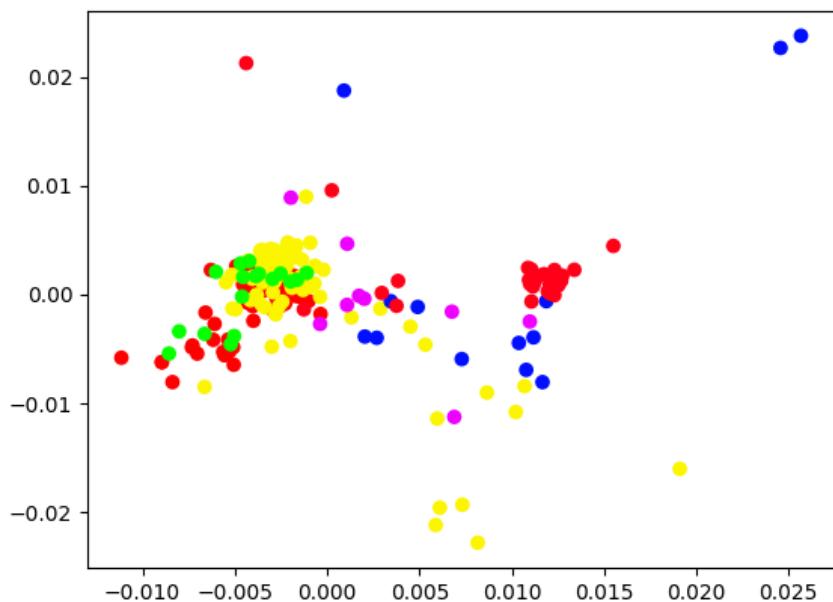


Рисунок 15 – Диаграмма рассеяния после понижения размерности до 2 ядерным методом главных компонент с применением сигмоидального ядра после изменения параметров

Ядро на основе косинусного сходства представляет собой функцию

$$K(x_1, x_2) = \frac{x_1^T x_2}{\|x_1\| \|x_2\|}.$$

На рисунке 16 представлена диаграмма рассеяния после понижения размерности до 2 ядерным методом главных компонент с применением ядра на основе косинусного сходства.

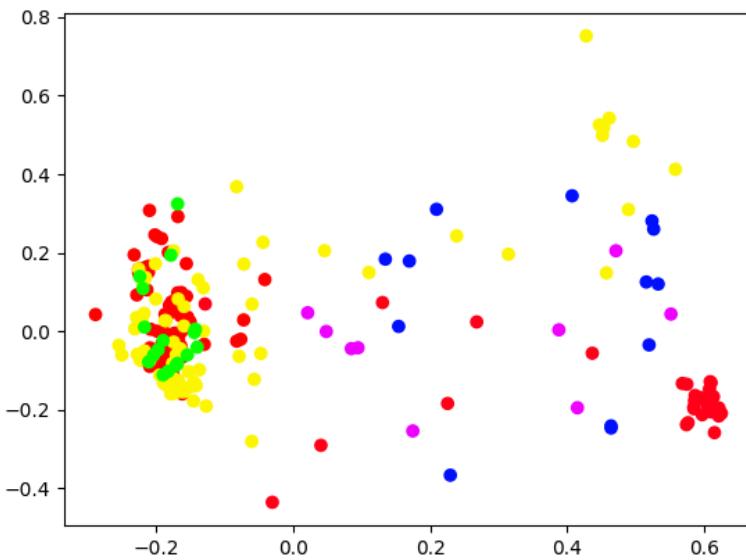


Рисунок 16 – Диаграмма рассеяния после понижения размерности до 2 ядерным методом главных компонент с применением ядра на основе косинусного сходства

Воспользуемся разряженным методом главных компонент для понижения размерности. Данная модификация отличается тем, что выражает каждую компоненту пространства пониженной размерности не через все исходные признаки, а только через некоторые. За счет этого легче понять, за что отвечает каждая из компонент пространства пониженной размерности. Данный метод, помимо количества компонент имеет следующие основные параметры такие параметры как *alpha*, отвечающий за степень разреженности компонент, и *method*, отвечающий используемый за метод оптимизации. На рисунке 17 представлена диаграмма рассеяния после понижения размерности до 2 разряженным методом главных компонент с параметрами *alpha=1* и *method='lars'*.

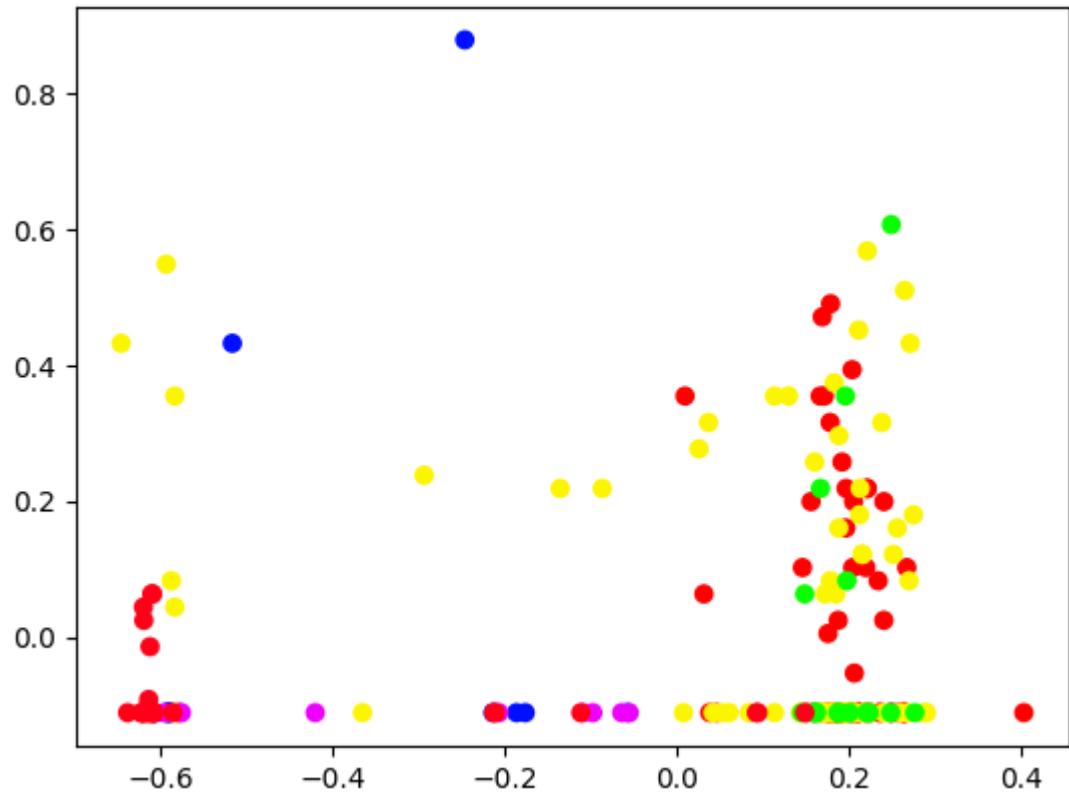


Рисунок 17 – Диаграмма рассеяния после понижения размерности до 2 разряженным методом главных компонент

На рисунке 18 представлена диаграмма рассеяния после понижения размерности до 2 разряженным методом главных компонент с параметрами $alpha=0.1$ и $method='cd'$.

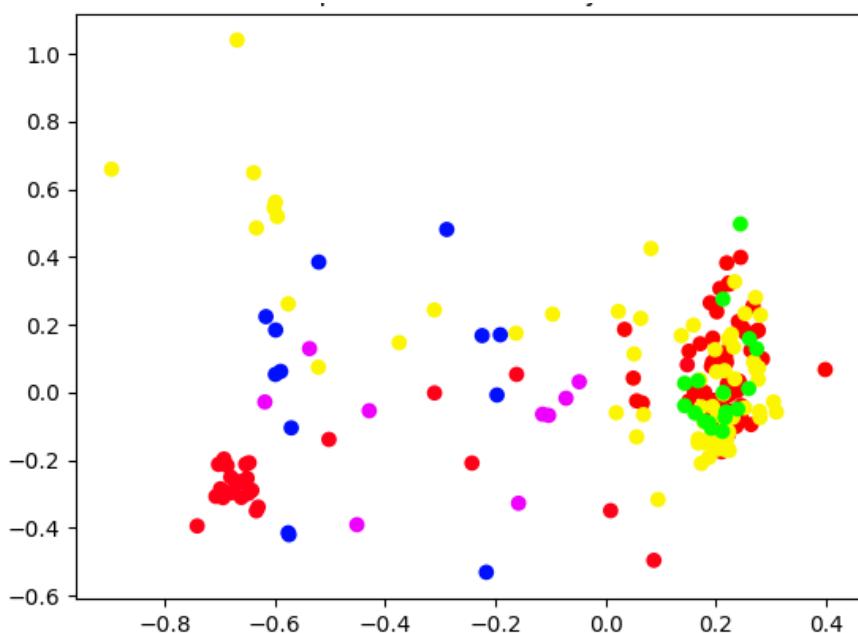


Рисунок 18 – Диаграмма рассеяния после понижения размерности до 2 разряженным методом главных компонент после изменения параметров

Факторный анализ

Альтернативным методом понижения размерности является факторный анализ. В отличие от метода главных компонент, который просто изменяет систему координат в исходном пространстве признаков, данный метод предполагает, что могут существовать скрытые признаки, в пространстве большей размерности, объясняющие корреляцию признаков исходных данных, и ставит целью найти такие признаки.

На рисунке 19 представлена диаграмма рассеяния, отражающая 2 скрытых признака, найденных методом факторного анализа.

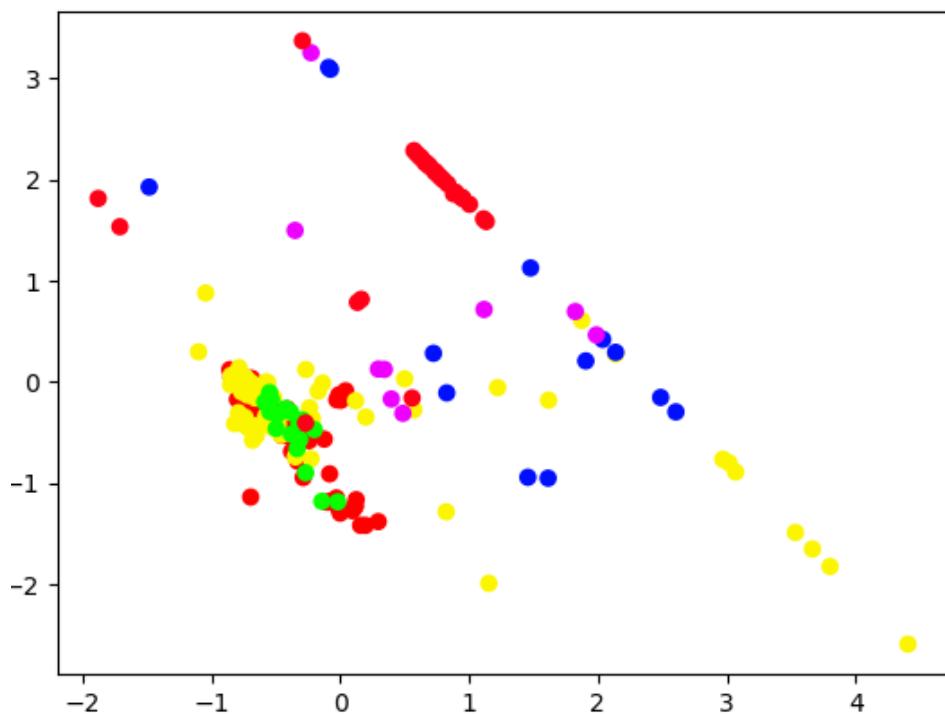


Рисунок 19 – Диаграмма рассеяния, отражающая 2 скрытых признака, найденных методом факторного анализа

Выводы

В результате выполнения лабораторной работы было проведено ознакомление с методами понижения размерности данных из библиотеки Scikit Learn.

Метод главных компонент линейно преобразует данные из исходного пространства в пространство меньшей размерности так, чтобы оставшиеся

после преобразования компоненты максимально сохраняли дисперсию исходных данных.

Ядерный метод главных компонент является модификацией метода главных компонент. Его особенностью является то, что данные сначала подвергаются нелинейному преобразованию при помощи ядра.

Разряженный метод главных компонент также является модификацией метода главных компонент. Его особенность в том, что компоненты пространства пониженной размерности выражены не через все исходные признаки, а только через некоторые, что помогает проще интерпретировать, за что отвечает каждая из компонент.

Метод факторного анализа, в отличие от метода главных компонент предполагает, что могут существовать скрытые признаки, в пространстве большей размерности, объясняющие корреляцию признаков исходных данных, и ставит целью найти такие признаки.

Код программы, написанной в ходе выполнения лабораторной работы представлен в приложении А.

ПРИЛОЖЕНИЕ А

КОД ПРОГРАММЫ

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

df = pd.read_csv('glass.csv')
var_names = list(df.columns)
labels = df.to_numpy('int')[:, -1]
data = df.to_numpy('float')[:, :-1]
data = preprocessing.minmax_scale(data)
fig, axs = plt.subplots(2, 4)
for i in range(data.shape[1]-1):
    axs[i // 4, i % 4].scatter(data[:, i], data[:, (i+1)], c=labels, cmap='hsv')
    axs[i // 4, i % 4].set_xlabel(var_names[i])
    axs[i // 4, i % 4].set_ylabel(var_names[i+1])
plt.show()

pca = decomposition.PCA(n_components = 2)
pca_data = pca.fit_transform(data)
print(sum(pca.explained_variance_ratio_))
print(pca.singular_values_)
plt.scatter(pca_data[:, 0], pca_data[:, 1], c=labels, cmap='hsv')
plt.title("PCA")
plt.show()

comp=2
while sum(pca.explained_variance_ratio_)<0.85:
    comp+=1
    pca = decomposition.PCA(n_components = comp)
    pca_data = pca.fit_transform(data)
print(pca.explained_variance_ratio_)
print(sum(pca.explained_variance_ratio_))
print(pca.singular_values_)
fig, axs = plt.subplots(1, comp-1)
for i in range(comp-1):
    axs[i].scatter(pca_data[:, i], pca_data[:, (i+1)], c=labels, cmap='hsv')
plt.show()

inversed_data=pca.inverse_transform(pca_data)
fig, axs = plt.subplots(2, 4)
for i in range(inversed_data.shape[1]-1):
    axs[i // 4, i % 4].scatter(inversed_data[:, i], inverted_data[:, (i+1)], c=labels, cmap='hsv')
    axs[i // 4, i % 4].set_xlabel(var_names[i])
    axs[i // 4, i % 4].set_ylabel(var_names[i+1])
plt.show()

pca = decomposition.PCA(n_components = 4, svd_solver='full')
pca_data = pca.fit_transform(data)
print(pca.explained_variance_ratio_)
print(sum(pca.explained_variance_ratio_))
print(pca.singular_values_)
plt.scatter(pca_data[:, 0], pca_data[:, 1], c=labels, cmap='hsv')
plt.title("full SVD")
plt.show()

pca = decomposition.PCA(n_components = 4, svd_solver='covariance_eigh')
pca_data = pca.fit_transform(data)
```

```

print(pca.explained_variance_ratio_)
print(sum(pca.explained_variance_ratio_))
print(pca.singular_values_)
plt.scatter(pca_data[:,0],pca_data[:,1],c=labels,cmap='hsv')
plt.title("covariance_eigh SVD")

plt.show()

pca = decomposition.PCA(n_components = 4,svd_solver='arpack')
pca_data = pca.fit_transform(data)
print(pca.explained_variance_ratio_)
print(sum(pca.explained_variance_ratio_))
print(pca.singular_values_)
plt.scatter(pca_data[:,0],pca_data[:,1],c=labels,cmap='hsv')
plt.title("arpack SVD")
plt.show()

pca = decomposition.PCA(n_components = 4,svd_solver='randomized')
pca_data = pca.fit_transform(data)
print(pca.explained_variance_ratio_)
print(sum(pca.explained_variance_ratio_))
print(pca.singular_values_)
plt.scatter(pca_data[:,0],pca_data[:,1],c=labels,cmap='hsv')
plt.title("randomized SVD")
plt.show()

kpca = decomposition.KernelPCA(n_components = 2,kernel='linear')
kpca_data = kpca.fit_transform(data)
plt.scatter(kpca_data[:,0],kpca_data[:,1],c=labels,cmap='hsv')
plt.title("Linear kernel PCA")
plt.show()

kpca = decomposition.KernelPCA(n_components = 2,kernel='poly',gamma=2, coef0=1,
degree=4)
kpca_data = kpca.fit_transform(data)
plt.scatter(kpca_data[:,0],kpca_data[:,1],c=labels,cmap='hsv')
plt.title("Poly kernel PCA")
plt.show()

kpca = decomposition.KernelPCA(n_components = 2,kernel='poly',gamma=1, coef0=0,
degree=5)
kpca_data = kpca.fit_transform(data)
plt.scatter(kpca_data[:,0],kpca_data[:,1],c=labels,cmap='hsv')
plt.title("Poly kernel PCA")
plt.show()

kpca = decomposition.KernelPCA(n_components = 2,kernel='rbf',gamma=1)
kpca_data = kpca.fit_transform(data)
plt.scatter(kpca_data[:,0],kpca_data[:,1],c=labels,cmap='hsv')
plt.title("RBF kernel PCA")
plt.show()

kpca = decomposition.KernelPCA(n_components = 2,kernel='rbf',gamma=2)
kpca_data = kpca.fit_transform(data)
plt.scatter(kpca_data[:,0],kpca_data[:,1],c=labels,cmap='hsv')
plt.title("RBF kernel PCA")
plt.show()

kpca = decomposition.KernelPCA(n_components = 2,kernel='sigmoid',gamma=1,
coef0=0)
kpca_data = kpca.fit_transform(data)
plt.scatter(kpca_data[:,0],kpca_data[:,1],c=labels,cmap='hsv')
plt.title("Sigmoid kernel PCA")

```

```

plt.show()

kpca = decomposition.KernelPCA(n_components = 2,kernel='sigmoid',gamma=2,
coef0=3)
kpca_data = kpca.fit_transform(data)
plt.scatter(kpca_data[:,0],kpca_data[:,1],c=labels,cmap='hsv')
plt.title("Sigmoid kernel PCA")
plt.show()

kpca = decomposition.KernelPCA(n_components = 2,kernel='cosine')
kpca_data = kpca.fit_transform(data)
plt.scatter(kpca_data[:,0],kpca_data[:,1],c=labels,cmap='hsv')
plt.title("Cosine kernel PCA")
plt.show()

spca=decomposition.SparsePCA(n_components=2,alpha=1,method='lars')
spca_data=spca.fit_transform(data)
plt.scatter(spca_data[:,0],spca_data[:,1],c=labels,cmap='hsv')
plt.title("Sparse PCA data")
plt.show()

spca=decomposition.SparsePCA(n_components=2,alpha=0.1,method='cd')
spca_data=spca.fit_transform(data)
plt.scatter(spca_data[:,0],spca_data[:,1],c=labels,cmap='hsv')
plt.title("Sparse PCA data")
plt.show()

fa=decomposition.FactorAnalysis(n_components=2)
fa_data=fa.fit_transform(data)
plt.scatter(fa_data[:,0],fa_data[:,1],c=labels,cmap='hsv')
plt.title("Factor analysis")
plt.show()

```