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

ОТЧЕТ

по лабораторной работе №2

по дисциплине «Машинное обучение»

Тема: Понижение размерности пространства признаков

Студент гр. 6304 _____ Ваганов Н.

Преподаватель _____ Жангиров Т. Р.

Санкт-Петербург

2020

Цель работы

Ознакомиться с методами понижения размерности данных из библиотеки Scikit Learn

Ход работы

Загрузка данных

1. Скачан и загружен датасет в датафрейм. Данные разделены на описательные признаки и признак, отображающий класс (рис. 1).

```
df = pd.read_csv('glass.csv')
var_names = list(df.columns) #получение имен признаков
labels = df.to_numpy('int')[:,-1] #метки классов
data = df.to_numpy('float')[:,:-1] #описательные признаки
```

	RI	Na	Mg	Al	Si	K	Ca	Ba	Fe	Type
0	1.52101	13.64	4.49	1.10	71.78	0.06	8.75	0.00	0.0	1
1	1.51761	13.89	3.60	1.36	72.73	0.48	7.83	0.00	0.0	1
2	1.51618	13.53	3.55	1.54	72.99	0.39	7.78	0.00	0.0	1
3	1.51766	13.21	3.69	1.29	72.61	0.57	8.22	0.00	0.0	1
4	1.51742	13.27	3.62	1.24	73.08	0.55	8.07	0.00	0.0	1
...	...	...	...	...	...	...	...	...	...	...
209	1.51623	14.14	0.00	2.88	72.61	0.08	9.18	1.06	0.0	7
210	1.51685	14.92	0.00	1.99	73.06	0.00	8.40	1.59	0.0	7
211	1.52065	14.36	0.00	2.02	73.42	0.00	8.44	1.64	0.0	7
212	1.51651	14.38	0.00	1.94	73.61	0.00	8.48	1.57	0.0	7
213	1.51711	14.23	0.00	2.08	73.36	0.00	8.62	1.67	0.0	7

214 rows × 10 columns

Рис. 1 — Загруженный датасет

2. Выполнена нормировка данных к интервалу [0, 1]

```
from sklearn import preprocessing
data = preprocessing.minmax_scale(data)
```

3. Построены диаграммы рассеяния для пар признаков. С помощью colorbar из matplotlib было определено соответствие цвета на диаграмме и класса (рис. 2 и 3)

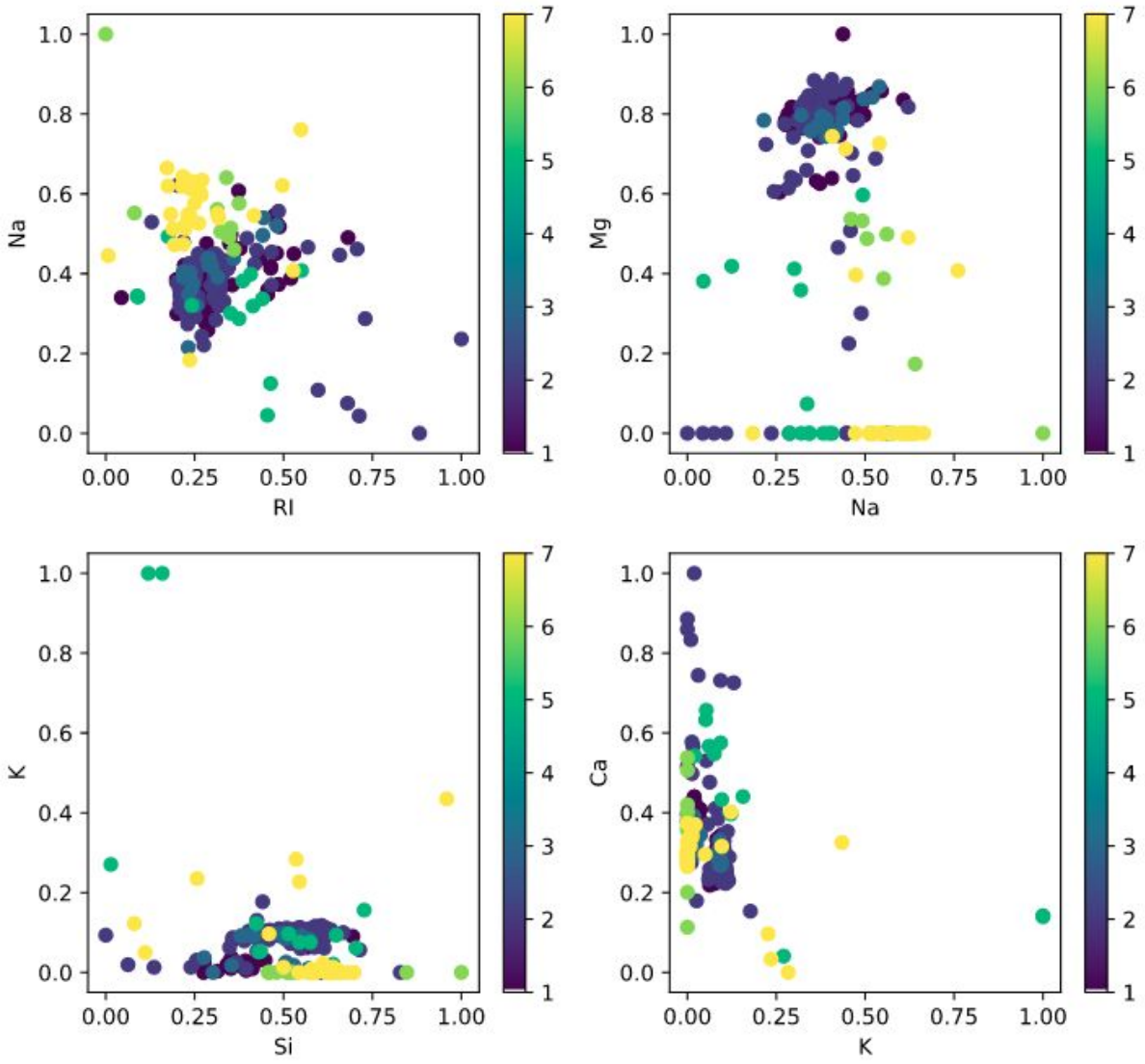

Рис. 2 — Диаграммы рассеивания

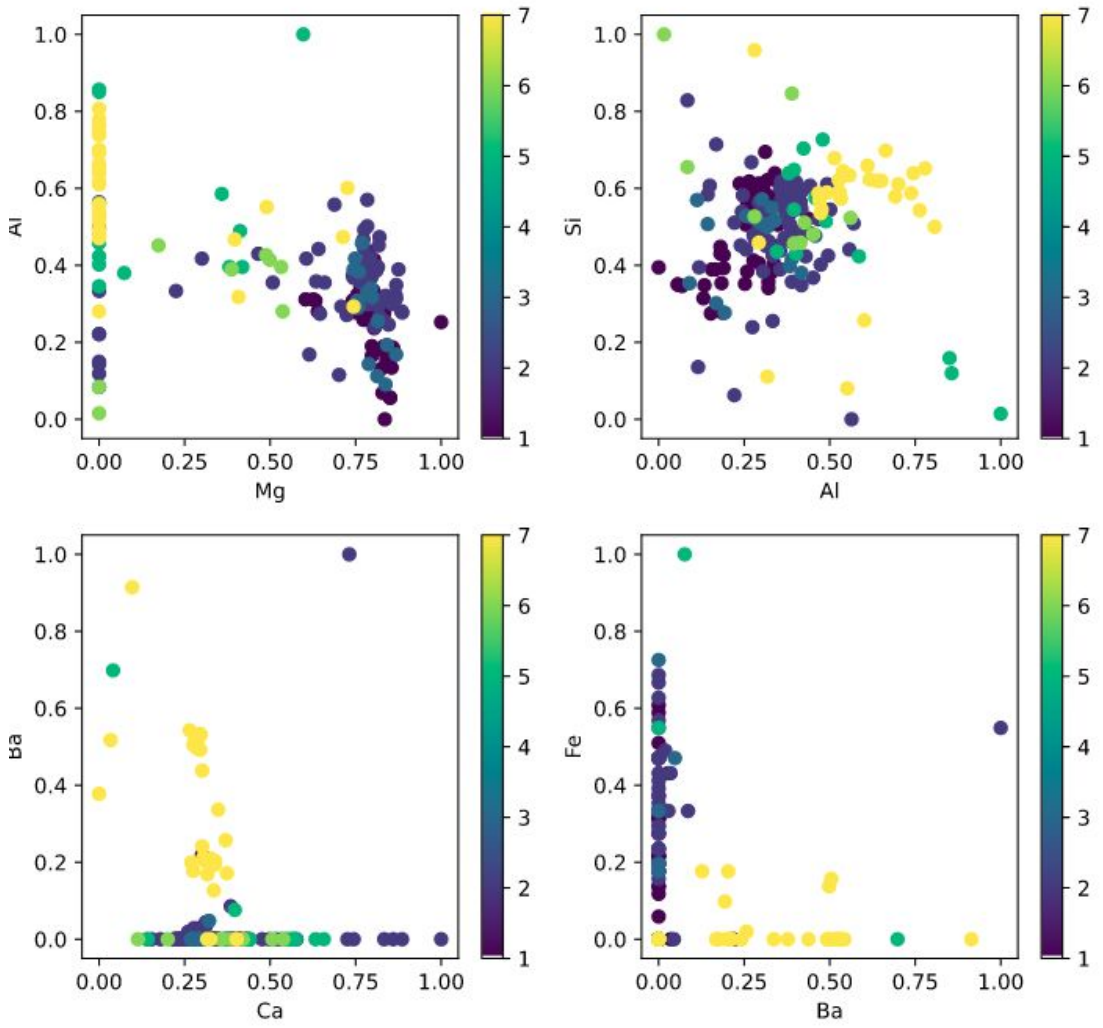

Рис. 3 — Диаграммы рассеивания

Метод главных компонент

1. Используя метод главных компонент, размерность данных была понижена до 2.

```
from sklearn.decomposition import PCA
pca = PCA(n_components = 2)
pca_data = pca.fit(data).transform(data)
```

2. Значения объясненной дисперсии и собственные числа:

```
[0.45429569 0.17990097]
[5.1049308  3.21245688]
```

3. Диаграмма рассеяния для данных после применения метода главных компонент (рис. 4)

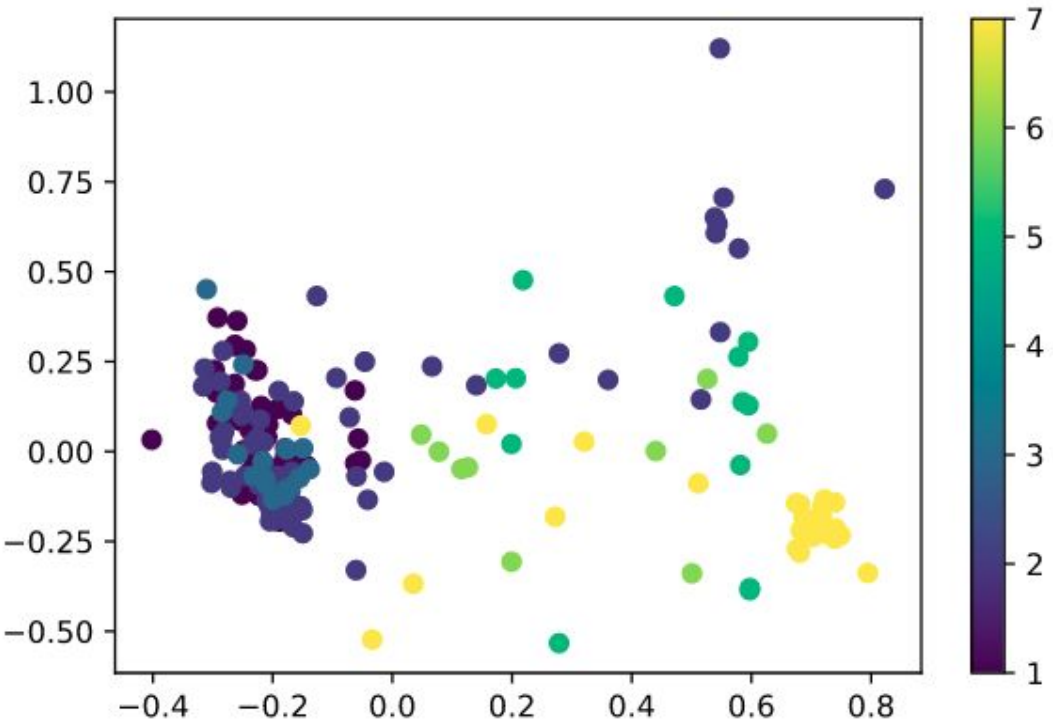

Рис. 4 — Диаграмма рассеивания после применения PCA

4. Согласно полученным данным, двух компонент в этом случае недостаточно (объясненная дисперсия 45% и 18%)

5. Для 4 компонент объясненная дисперсия составит примерно 85%
   `[0.45429569 0.17990097 0.12649459 0.09797847]`

6. Используя метод *inverse_transform* был получен следующий результат:

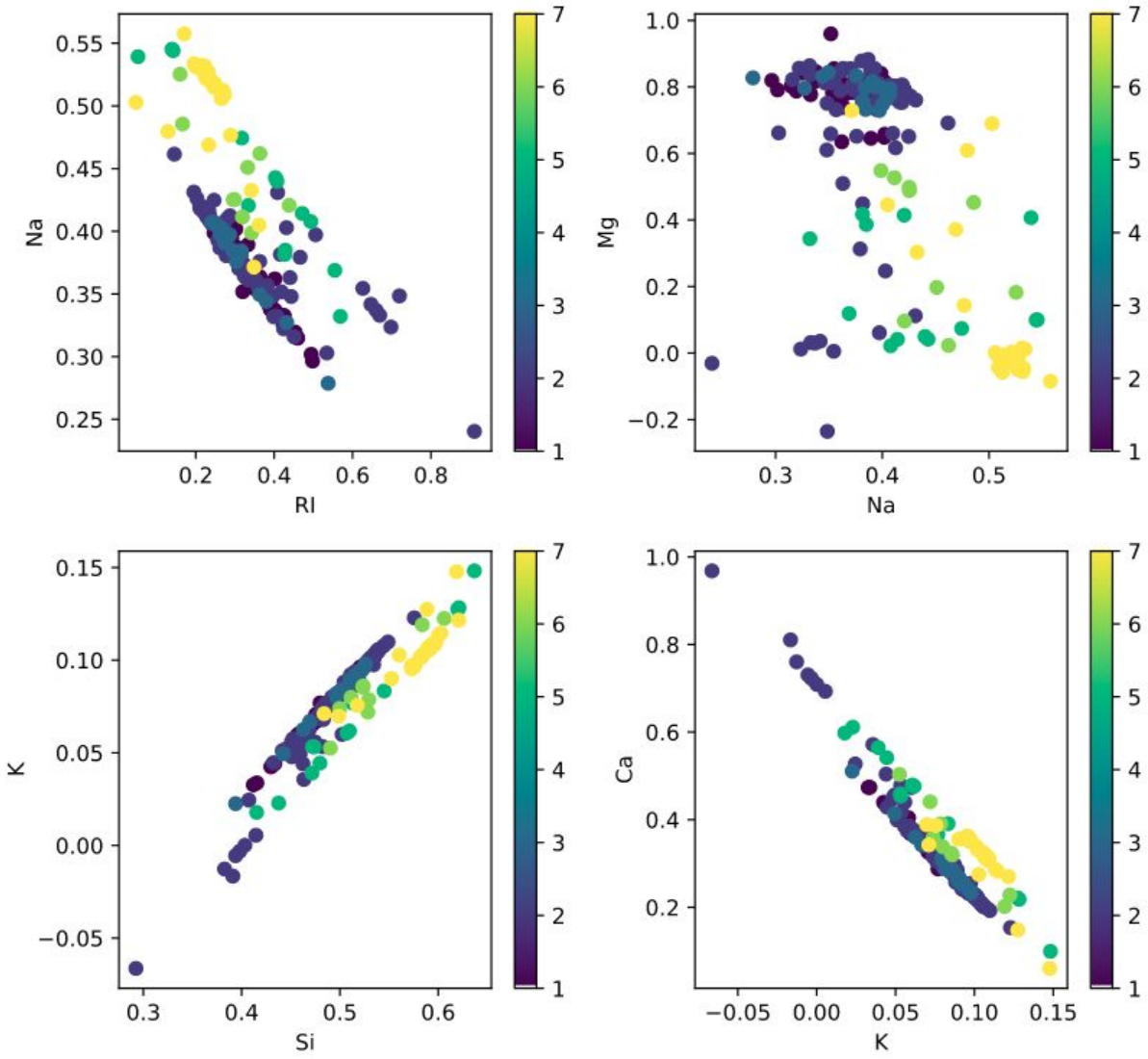

Рис. 5 — Диаграммы рассеивания после применения *inverse_transform*

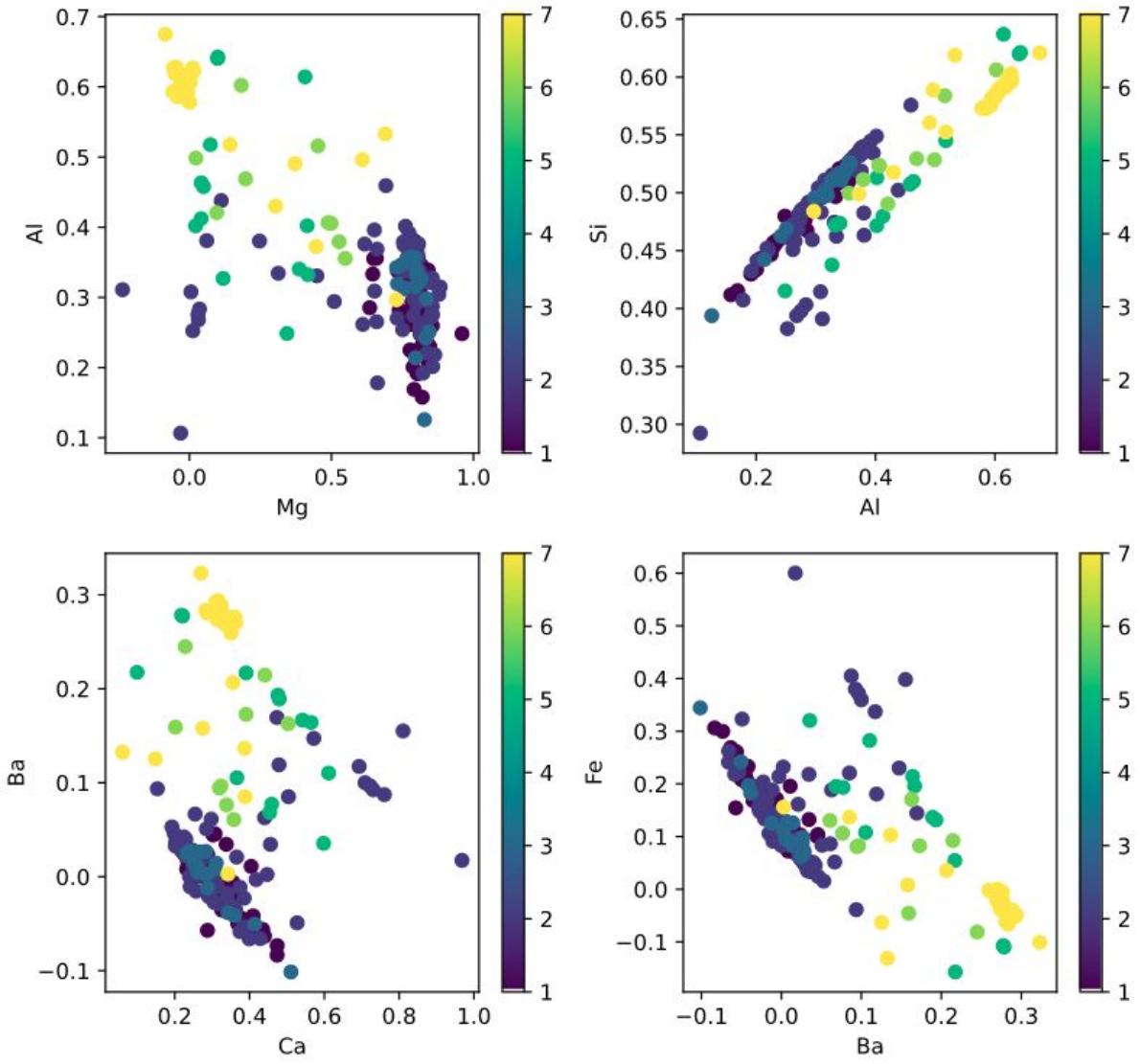

Рис. 6 — Диаграммы рассеивания после применения *inverse_transform*

“Восстановленные” данные сильно отличаются от исходных, что заметно на графиках.

7. По умолчанию параметр *svd_solver* принимает значение *auto*. Ниже представлены результаты работы метода PCA при значениях *full*, *arpack* и *randomized.*

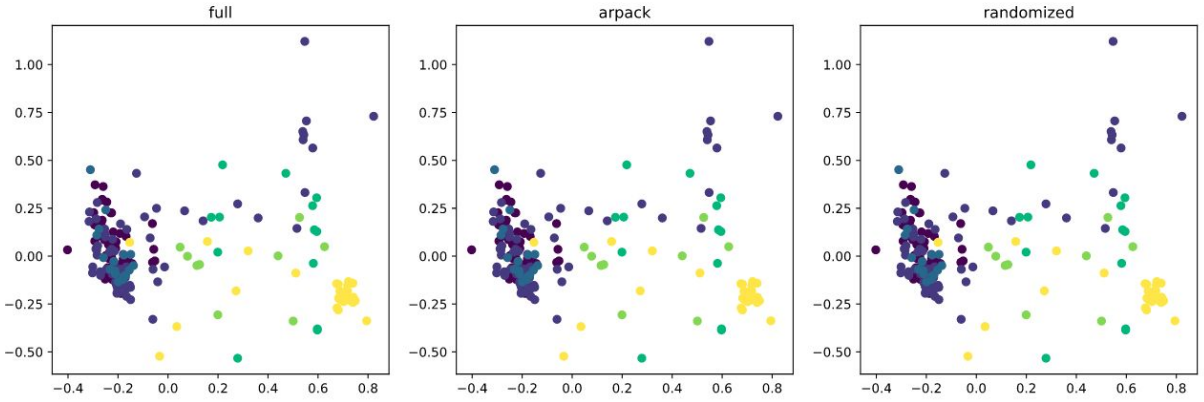

Рис. 7 — Диаграммы рассеивания с различными параметрами *svd_solver*

Модификации метода главных компонент

KernelPCA - вычисление главных компонент с помощью ядер. Возможные ядра - линейное, полиномиальное, радиально-базисная функция (RBF), сигмоида, косинус.

KernelPCA работает аналогично PCA, при значении параметра kernel равного “linear”. После настройки, в поле lambdas_ содержатся собственные числа (26.06, 10.32, 7.26, 5.62), являющиеся квадратами сингулярных чисел, хранящихся в поле singular_values: (5.11, 3.21, 2.69, 2.37). Сравнение объясненных дисперсий для различных ядер приведено в таблице 1.

Таблица 1 - Сравнение различных ядер KernelPCA

Ядро	PC1	PC2	PC3	PC4	Сумма	Объясненная дисперсия
linear	26.06	10.32	7.26	5.62	49.26	0.859
poly	10.92	4.32	3.12	2.37	20.72	0.849
rbf	5.35	2.02	1.50	1.11	9.98	0.850
sigmoid	1.01	0.40	0.27	0.22	1.90	0.862
cos	18.31	6.48	4.70	3.58	33.06	0.860

SparsePCA - анализ разреженных компонент, которые могут оптимально восстановить данные. Могут использоваться два метода: наименьшая угловая регрессия (least angle regression) или координатный спуск (coordinate descent) для решения задачи Lasso (метод регрессионного анализа, который выполняет как выбор переменных, так и регуляризацию, чтобы повысить точность прогнозирования и интерпретируемость создаваемой статистической модели).

Применение разных методов не дает визуально различимых отличий, кроме того, компоненты получаются одинаковые. Однако изменение параметра alpha, который влияет на разреженность данных, дает видимые различия. Также получаются различные компоненты.

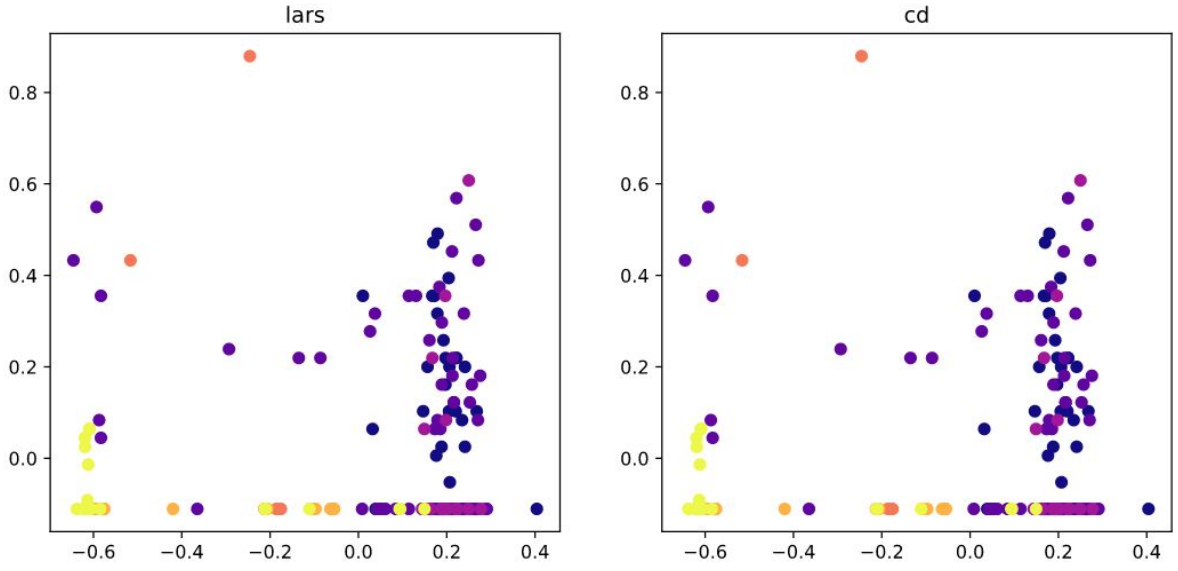

Рис. 8 — Диаграммы рассеивания первых двух главных компонент с использованием различных методов

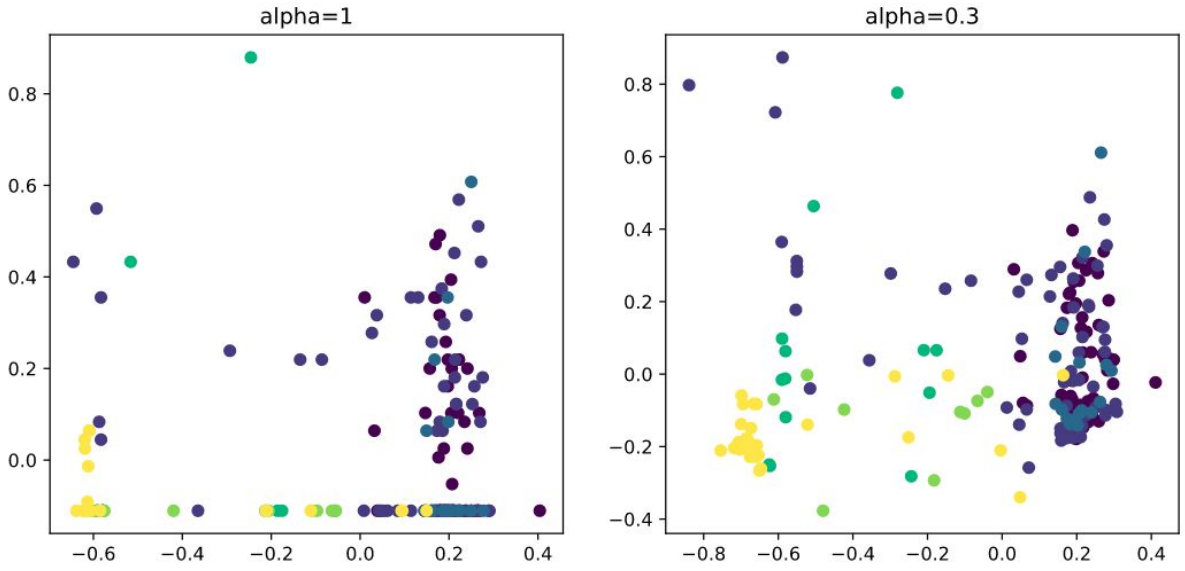

Рис. 9 — Диаграммы рассеивания первых двух главных компонент с различными значениями alpha

Таблица 2 - Главные компоненты SparsePCA при различных alpha

alpha = 1	PC1	0	0	0.998	-0.037	0	0	0	-0.05	0
	PC2	0	0	0	0	0	0	0	0	1
alpha = 0.3	PC1	0	-0.044	0.946	-0.203	0	0	-0.109	-0.224	0
	PC2	0.367	-0.148	0	-0.163	-0.144	0	0.328	0	0.83

Факторный анализ

1. Было выполнено понижение размерности с использованием факторного анализа

```
transformer = FactorAnalysis(n_components=2)
fa_2 = transformer.fit_transform(data)
```

2. Между результатами PCA и FA есть значимые различия в результирующем виде данных (рис. 10)

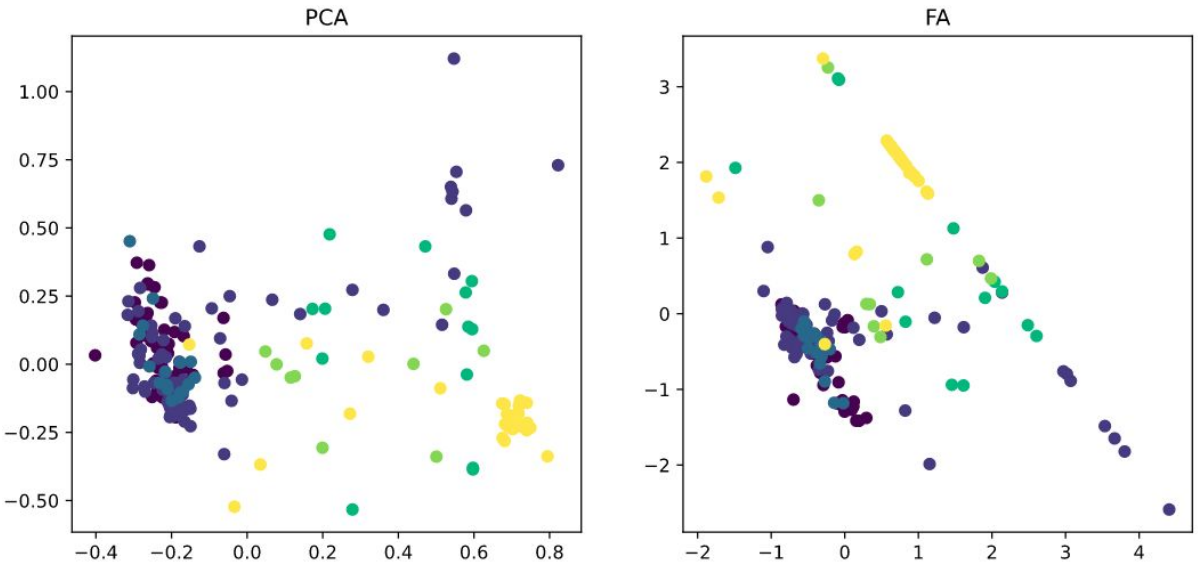

Рис. 10 — Диаграмма рассеяния результатов PCA и FA

3. Различия PCA и FA:
    a. Компоненты PCA объясняют максимальную дисперсию, в то время как факторный анализ объясняет ковариацию данных.
    b. Компоненты PCA полностью ортогональны друг другу, тогда как факторный анализ не требует, чтобы факторы были ортогональными.
    c. Компонент PCA представляет собой линейную комбинацию наблюдаемой переменной, тогда как в FA наблюдаемые переменные представляют собой линейные комбинации ненаблюдаемой переменной или фактора.
    d. PCA - это своего рода метод уменьшения размерности, тогда как факторный анализ - метод поиска скрытых переменных.

Вывод

Были изучены методы понижения размерности данных Factor Analysis и Principal component analysis из библиотеки Scikit Learn.

В ходе работы было выяснено, что разное количество компонент в PCA объясняет разное количество дисперсии данных.

KernelPCA используется для поиска нелинейных зависимостей в данных. Однако на предложенных данных объясненная дисперсия была около 85% независимо от ядра. SparsePCA выгодно использовать на данных большого размера.

Также был изучен факторный анализ и его различия с PCA.