Таким образом, удалось решить задачу ЛП и графически визуализировать решениепри помощи программы MathCad15. Практическая ценность исследования заключается в возможности применения этого математического пакета для решения задач линейного программирования оптимизации, где необходимо найти минимум или максимум целевой функции при заданных линейных ограничениях. Можно заметить, что анимация применяется и при решении задач параметрического линейного программирования, где целевая функция имеет видF($x_1 x_2$,t)=(2-t)x_1-(1-2t)x_2, при этом t принадлежит некоторому промежутку, напримерt ∈ [0; 4].

Литература

1. Автоматизация решение задач линейного программирования: Пособие для студентов дневной формы обучения экономических специальностей / Авторы-составители: В.В. Бондарева, О.И. Еськова – Гомель: УО «Белорусский торгово-экономический университет потребительской кооперации», 2003. – 68 с. (дата обращения 20.04.2025)

2. Как заштриховать область под графиком в MathCAD[Электронный ресурс]. Электронные данные. – Режим доступа: http://blog.kislenko.net/show.php?id=1439(дата обращения 20.04.2025)

УДК 519.237.7:004.85

ПРИМЕНЕНИЕ МАТРИЧНЫХ ПРЕОБРАЗОВАНИЙ В ПРЕДОБРАБОТКЕ ДАННЫХ: МЕТОД ГЛАВНЫХ КОМПОНЕНТ

Лобань И. Д.
Научный руководитель – Юринок В. И., к.т.н., доцент

В науке о данных (Datascience)очень важным этапом является предобработка данных. Она нужна для того, чтобы увидеть распределение признаков, узнать их главные характеристики, такие как среднее или медиана, визуализировать данные,перед тем какиспользовать их для обучения модели.

Здесь возникает проблема: зачастую объекты имеют десятки, а то и сотни признаков, человек просто не сможет представить такое многомерное пространство. Также некоторые признаки могут быть частично или полностью зависимы от других, как например вес человека от его роста или площадь квадрата от его стороны.Тогда на помощь приходит очень мощный инструмент – метод главных компонент (PrincipalComponentAnalysis, PCA).

Суть метода заключается в том, что он переводит исходную матрицу в базис с новыми осями координат (которые называются главными

компонентами), причем дисперсия для этих осей расположена в порядке убывания, т. е. на первую главную компоненту приходится наибольшая дисперсия, на вторую – наибольшая дисперсия из оставшейся и т. д.

Благодаря PCA,мы можем взять первые kосей и спроецировать на них наши данные, потеряв при этом совсем незначительный процент информации. То есть мы уменьшим размерность, избавившись при этом от сильно коррелированных признаков.

С математической точки зрения, метод находит собственные вектора и собственные числа ковариационной матрицы, причем чем больше собственное число, тем больше информации приходится на соответствующий ему собственный вектор.

Рассмотрим применение метода на конкретном примере:

Пусть дана вещественная матрица$A_{(200\times3)} = (X \quad Y \quad Z)$, где X, Y, Z–векторы-столбцы, причем $x_i \in [-5; 5], y_i \in [-5; 5], z_i = 0.5x_i + 0.2y_i + \varepsilon_i, \varepsilon_i \in [-1; 1], i = \overline{1, 200}$, то есть, мы имеем матрицу из 200 точек, которые лежат около плоскости $0.5x + 0.2y - z = 0$, с некоторой погрешностью ε.

В основном вместо построения ковариационной матрицы и нахождения её собственных векторов применяется сингулярное разложение (SingularValueDecomposition, SVD) исходной матрицы:$A_{m\times n} = USV^T$, где $U_{m\times m}$–матрица левых сингулярных векторов, $S_{m\times n}$–диагональная матрица сингулярных чисел, $V_{n\times n}$ – матрица правых сингулярных векторов, причем матрицы Uи V–ортогональны.При этом, матрица правых сингулярных векторов как раз и будет содержать в себе собственные вектора ковариационной матрицы.

Так как $C = A^T A$, то, подставив её в SVD, получим:

$$A^T A = (USV^T)^T USV^T = VSU^T USV^T$$

Исходя из ортогональности матрицы U, $U^T U = E$, где E–единичная матрица. Получим $C = VS^2V^T$, что является спектральным разложением матрицыC.

Рассмотрим реализацию нашего примера на Python.

Для начала сгенерируем матрицу Аи центрируем её. Центрирование – это стандартный шаг перед PCA:

```
n = 200

x = np.random.uniform(low=-5, high=5, size=n)
y = np.random.uniform(low=-5, high=5, size=n)
eps = np.random.uniform(low=-1, high=1, size=n)

z = 0.5*x + 0.2*y + eps

A = np.column_stack((x, y, z))
A = A - np.mean(A, axis=0)
```

Затем отобразим распределение точек в трёхмерном пространстве(Рис.1)

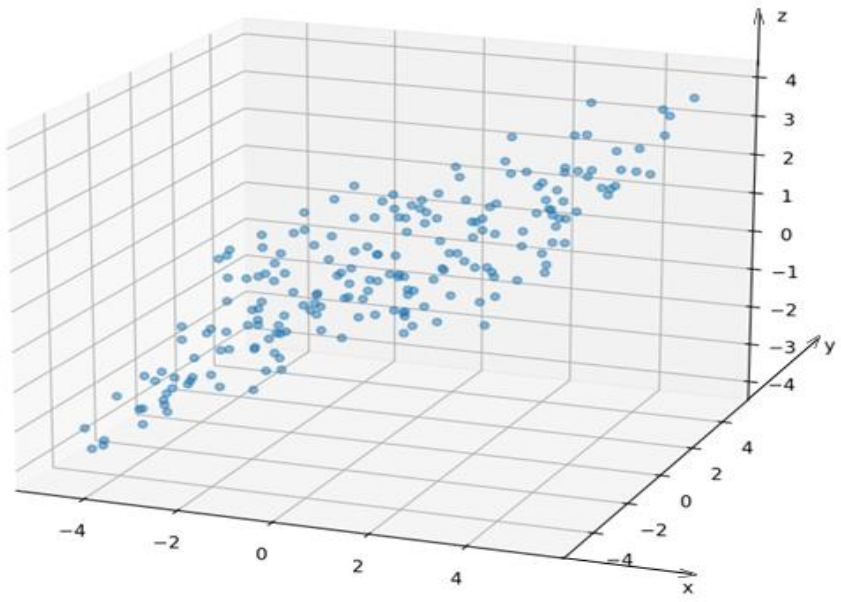

Рис.1. Исходная матрица в 3D.

Мы видим, что наши данные лежат около некоторой плоскости, и т. к. координата z выражается через x и y,следовательно, не несёт дополнительной информации. Теперь применим сингулярное разложение и возьмем только первые 2 главные компоненты, тем самым спроецировав точки на плоскость:

```
U, S, Vt = np.linalg.svd(A)
V = Vt.T
A_proj = A @ V[:, :2]
```

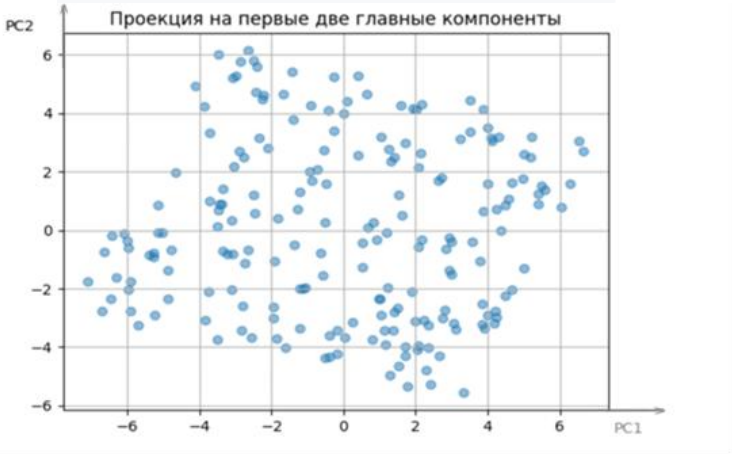

Рис.2. Проекция на первые две главные компоненты.

Из Рис.2 видно, что мы уменьшили размерность наших данных, при этом отобразив их с сохранением максимальной информации об их взаимном расположении.

Рис.3. Процент информации, приходящийся на каждую компоненту.

Получается, что на первые 2 компоненты приходится более 98% информации (Рис.3).

Таким образом, метод главных компонентпоказал себя как эффективный инструмент снижения размерности данных при минимальной потере информации. Среди его основных преимуществ – устранение избыточности признаков, упрощение анализа и визуализации, а также ускорение работы алгоритмов машинного обучения.

К недостаткам можно отнести его чувствительность к масштабированию, линейность и затруднённую интерпретацию новых признаков.

Несмотря на это, PCA остаётся одним из ключевых методов предобработки данных и активно применяется как в исследовательских, так и в прикладных задачах анализа данных.

Литература

1.Метод главных компонент [Электронный ресурс]. – Режим доступа: https://practicum.yandex.ru/blog/metod-glavnyh-komponent/. – Дата доступа: 10.04.2025.