Понижение размерности

Паточенко Евгений

НИУ ВШЭ

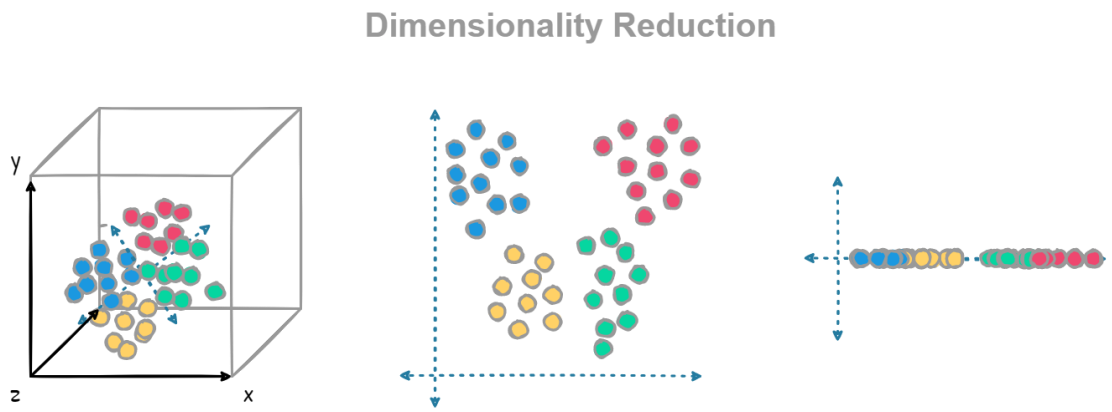
План занятия

- Задача понижения размерности
- Отбор и извлечение признаков
- PCA
- SVD
- Визуализация данных

Задача понижения размерности

Понижение размерности (dimensionality reduction) — задача машинного обучения, направленная на уменьшение числа признаков (измерений) в данных при сохранении их информативности.

Количество признаков снижается за счет отбрасывания слабо информативных и неинформативных, избыточных (коррелирующих) и шумовых признаков.



Источник: https://blog.roboflow.com/what-is-dimensionality-reduction/

Задача понижения размерности

Цели:

- повысить скорость обучения модели, снизить риск переобучения
- за счет повышения скорости увеличить возможное количество проводимых за то же время экспериментов
- снизить объемы хранимой и обрабатываемой информации
- повысить наглядность представления и интерпретируемость данных
- упростить модель и, следовательно, ускорить инференс

Задача понижения размерности

Проклятие размерности

Чем выше размерность (то есть чем больше признаков), тем:

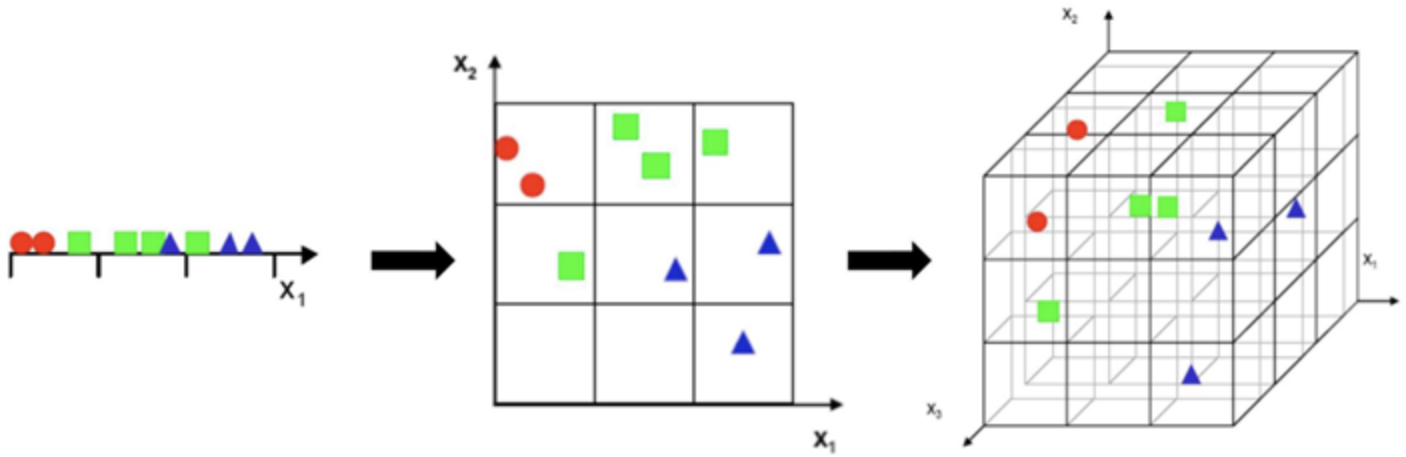
- более разреженными становятся данные и тем более сложными становятся вычисления,
- сильнее растет риск переобучения,
- менее информативным становится расстояние между точками ,
- большее количество данных требуется

Задача понижения размерности

Проклятие размерности

Чем выше размерность (то есть чем больше признаков), тем:

- **экспоненциально** большее количество данных требуется!

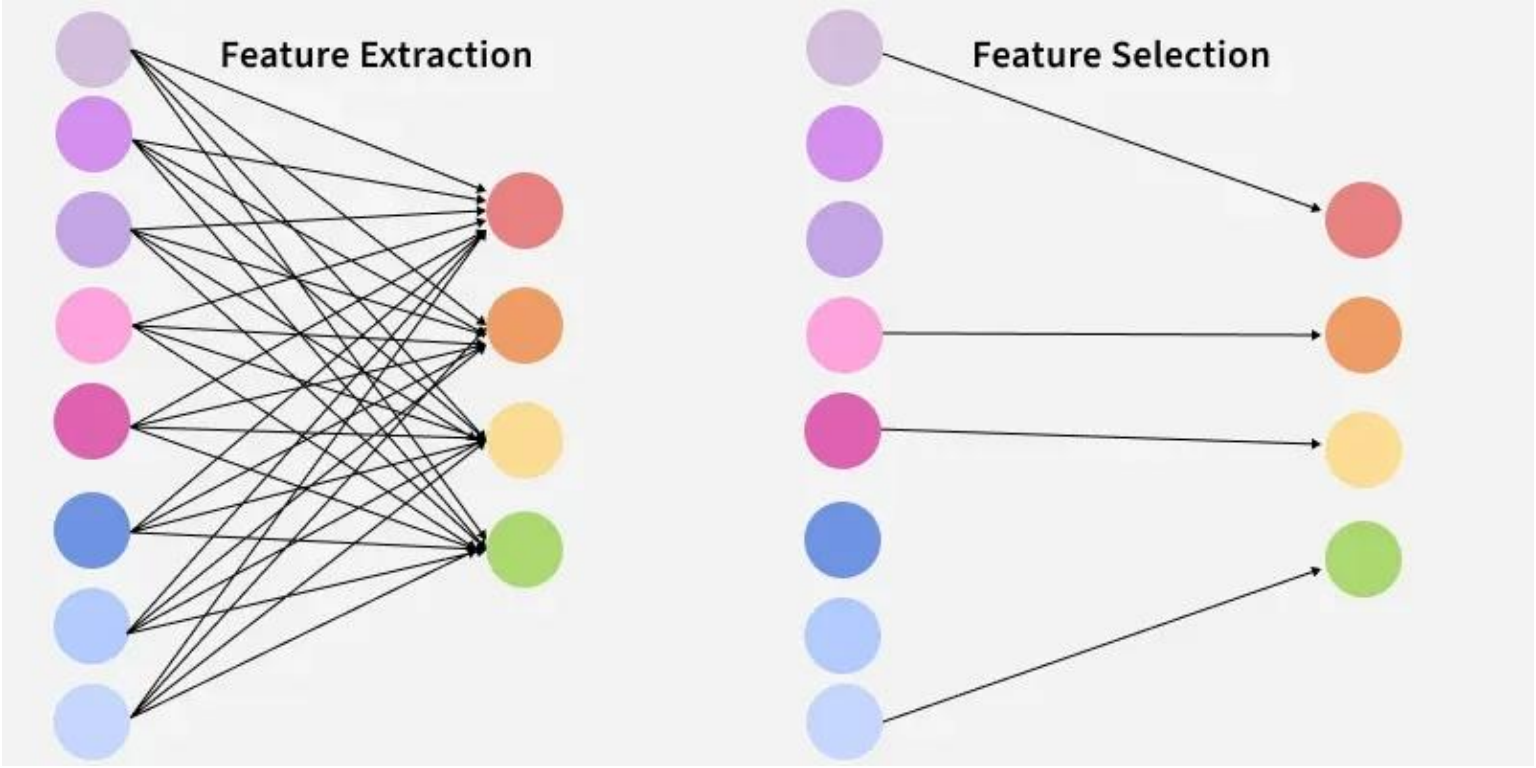


Источник: https://www.youtube.com/watch?v=a2U4F7cxFHc

Задача понижения размерности

Способы:

- Отбор признаков (feature selection)
- Извлечение признаков (feature extraction)



Источник: https://www.geeksforgeeks.org/machine-learning/feature-selection-vs-feature-extraction/

Задача понижения размерности

	Отбор признаков (Feature Selection)	**Извлечение признаков (Feature Extraction)**
Подход	Выбирает подмножество релевантных признаков из исходного набора	Преобразует исходные признаки в новый, более информативный набор
Механизм	Снижает размерность, сохраняя исходные признаки	Снижает размерность за счет преобразования данных в новое пространство
Требования	Требует предметных знаний и инженерии признаков.	Может применяться к необработанным данным без предварительной инженерии признаков.
Минусы	Может приводить к потере полезной информации, если удалены важные признаки.	Может вносить избыточность и шум, если извлеченные признаки определены плохо.
Примеры методов	Filter, Wrapper, Embedded	PCA, LDA, Kernel PCA, автоэнкодеры.

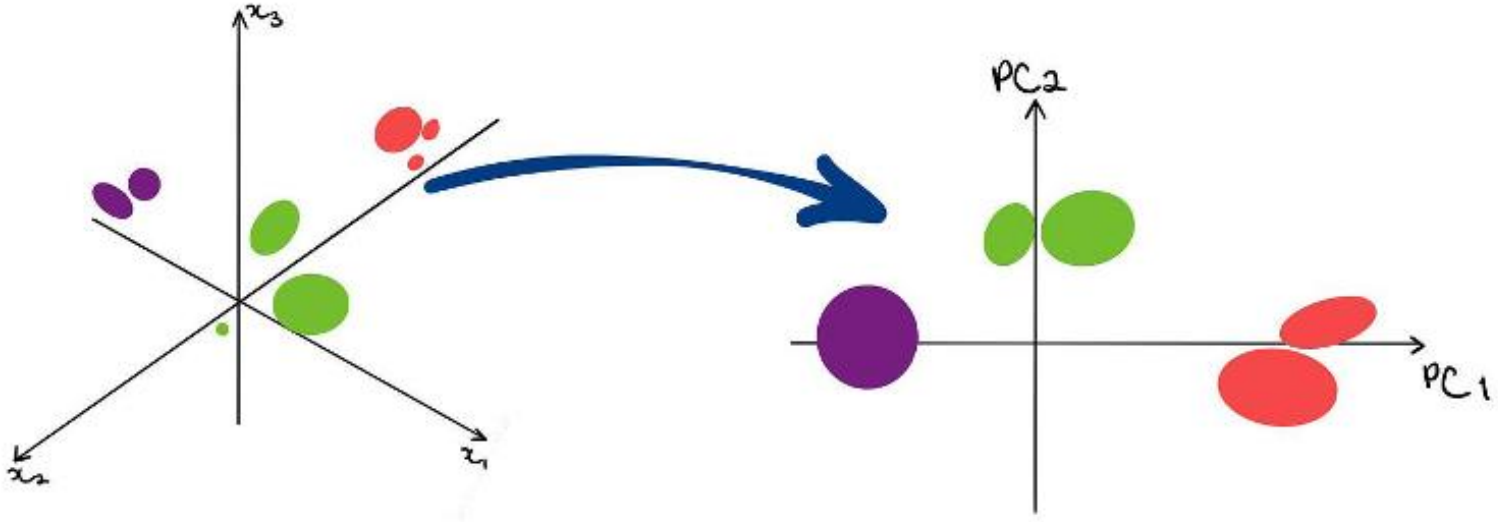
Источник: https://www.geeksforgeeks.org/machine-learning/feature-selection-vs-feature-extraction/

Отбор признаков: filter, wrapper, embedded

- Filter — фильтрация признаков на основе степени корреляции с целевой переменной
- Wrapper — выбор подмножества признаков с наилучшими результатами на обучающей выборке (похоже на перебор гиперпараметров). Подразделяются на методы включения (начинаем с пустого множества и постепенно добавляем признаки, оценивая результат) и исключения (начинаем со всего множества и постепенно убираем признаки)
- Embedded — например, регуляризация

Извлечение признаков: PCA

Метод главных компонент (Principal Component Analysis, PCA) проецирует исходные данные в пространство признаков меньшей размерности, выделяя самые важные закономерности



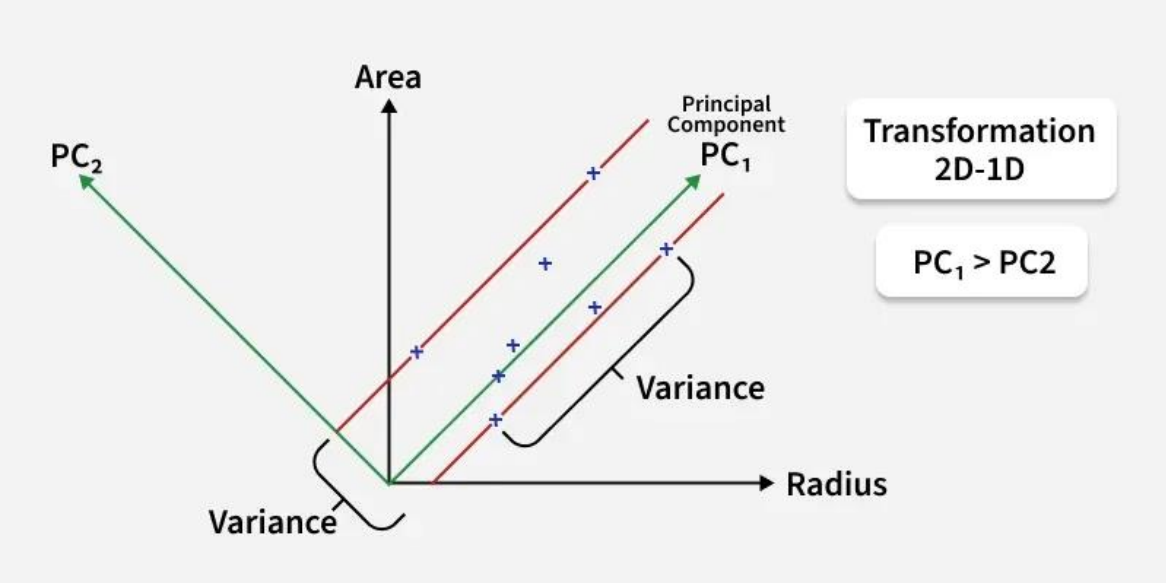
Источник: https://pub.towardsai.net/demystifying-principal-component-analysis-a-comprehensive-guide-575738912658

Извлечение признаков: PCA

Идея метода:

PCA создает новые переменные — главные компоненты, которые являются линейной комбинацией исходных признаков и содержат наиболее важную информацию

Компоненты создаются в результате поиска таких направлений в данных, вдоль которых дисперсия в признаках наивысшая



Источник: https://www.geeksforgeeks.org/data-analysis/principal-component-analysis-pca/

Извлечение признаков: PCA

Механизм:

1. Стандартизация данных

Перед применением PCA признаки нормализуются: каждому признаку придают среднее $\mu = 0$ и стандартное отклонение $\sigma = 1$, чтобы отличия в масштабе не влияли на результат

$$Z = \frac{X - \mu}{\sigma}$$

Извлечение признаков: PCA

Механизм:

2. Вычисление матрицы ковариаций

PCA вычисляет ковариацию между признаками, чтобы понять, как они связаны друг с другом — растут ли вместе, обратно ли связаны и т. д

$$cov(x_1, x_2) = \frac{\sum_{i=1}^{n}(x_{1_i} - \overline{x_1})(x_{2_i} - \overline{x_2})}{n-1}$$

где $\overline{x_1}$ и $\overline{x_2}$ — это средние значения признаков x_1 и x_2,

n — это количество наблюдений

Извлечение признаков: PCA

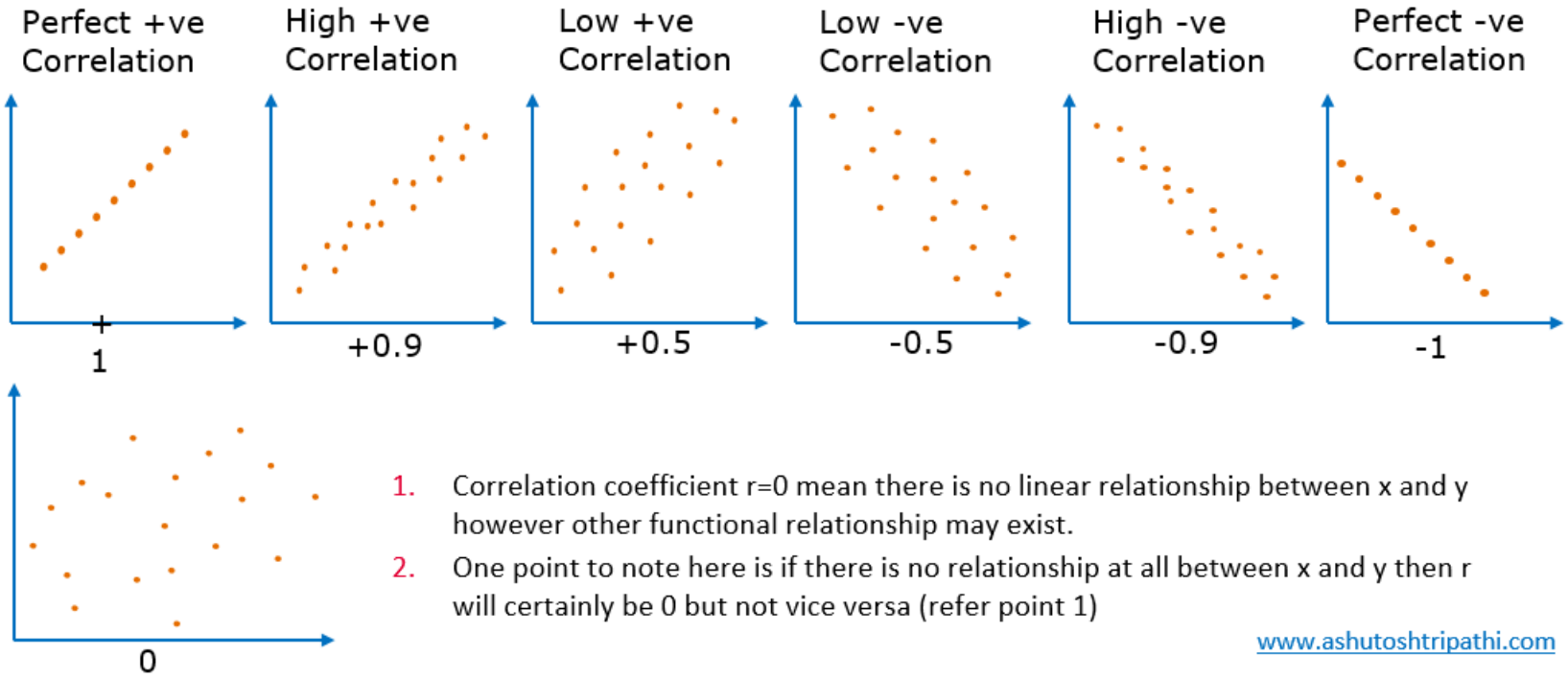
Напоминание о ковариации:

Correlation coefficient r is number between -1 to +1 and tells us how well a regression line fits the data and defined by

$$r_{xy} = \frac{s_{xy}}{s_x s_y}$$

where,

- s_{xy} is the covariance between x and y
- s_x and s_y are the standard deviations of x and y respectively.



1. Correlation coefficient r=0 mean there is no linear relationship between x and y however other functional relationship may exist.
2. One point to note here is if there is no relationship at all between x and y then r will certainly be 0 but not vice versa (refer point 1)

www.ashutoshtripathi.com

Источник: https://ashutoshtripathi.com/wp-content/uploads/2019/02/correlation-1.png

Извлечение признаков: PCA

Механизм:

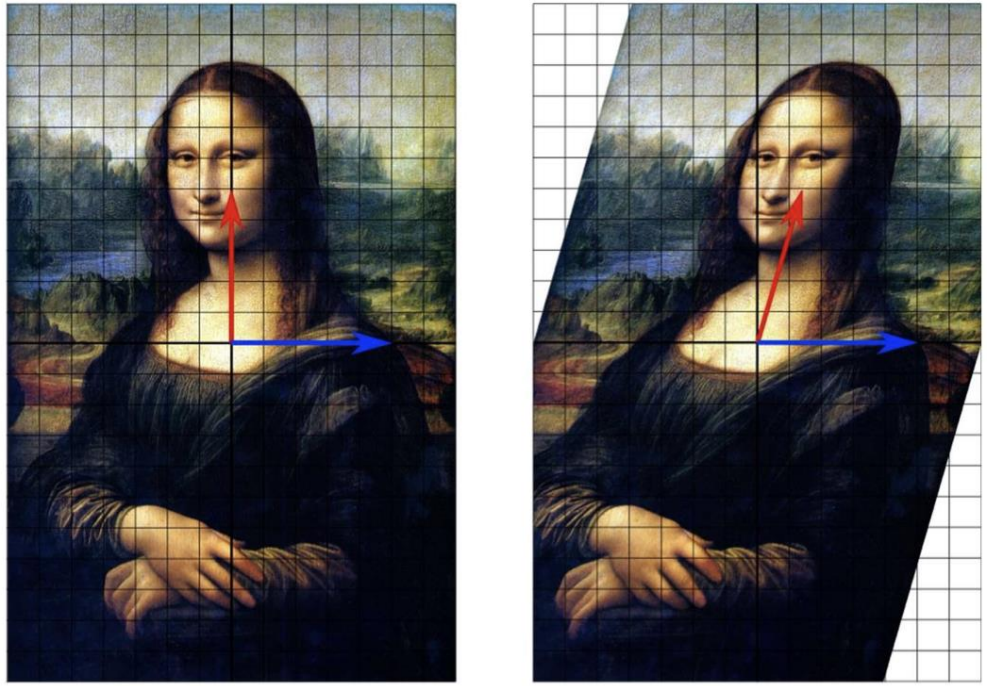
3. Нахождение главных компонент

Метод находит направления, вдоль которых данные «распылены» больше всего (максимальная дисперсия). Для этого вычисляются собственные вектора и собственные значения ковариационной матрицы.

- Первая компонента (PC1) — направление с наибольшей дисперсией.
- Вторая компонента (PC2) — перпендикулярная к первой, с наибольшей оставшейся дисперсией, и т.д.

Извлечение признаков: PCA

Напоминание о собственном векторе: вектор v, который под действием матрицы A не меняет своего расположения, называется собственным вектором матрицы A: $Av = \lambda v$



Источник: https://en.wikipedia.org/wiki/File:Mona_Lisa_eigenvector_grid.png

Извлечение признаков: PCA

Механизм:

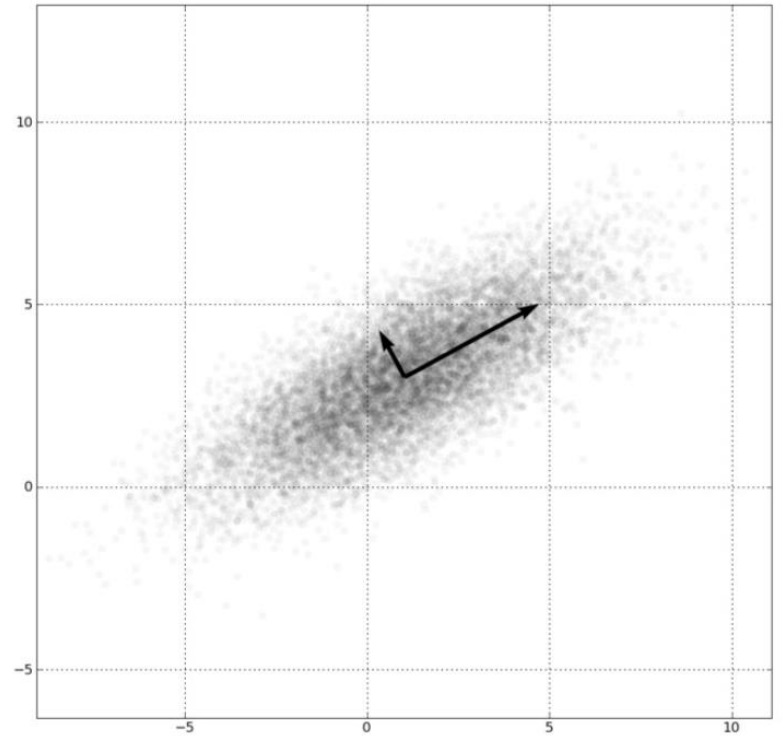
4. Выбор топ-компонент и проекция данных

После ранжирования компонент по «важности» выбирается несколько первых (например, так, чтобы они покрывали $\sim 95\ \%$ дисперсии). Затем исходные данные проецируются на пространство, образованное этими компонентами — это и есть пониженная размерность.

Извлечение признаков: PCA

Механизм:

Таким образом, чтобы найти главные компоненты, достаточно найти все собственные векторы матрицы ковариации, или — что то же самое — перейти в базис, где она диагональна

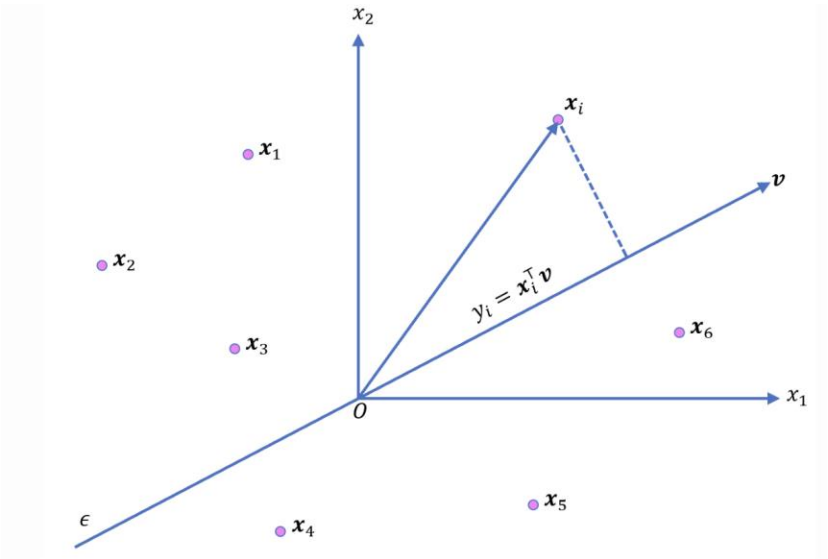


Источник: https://commons.wikimedia.org/wiki/Category:Principal_component_analysis

Извлечение признаков: PCA

Математический вывод

- Предположим, что есть $x_1, \ldots, x_N$ отцентрированных точек: $(x_1 + \ldots + x_N)/\, N = 0$
- Пусть есть некоторый вектор $\boldsymbol{v}$, на который проецируются $x_1, \ldots, x_N$
- Тогда длина такой проекции $y_i = x_i^T v$



Извлечение признаков: PCA

Математический вывод

Тогда дисперсия вдоль данного направления может быть вычислена как

$$V = \frac{1}{N}\sum_{i=1}^{N} y^2 = \frac{1}{N}\sum_{i=1}^{N} (x_i^T v)^2 = \frac{1}{N}\sum_{i=1}^{N} v^T x_i \cdot x_i^T v = v^T \left[\frac{1}{N}\sum_{i=1}^{N} x_i^T x_i\right] v = v^T C v$$

Где матрица C — это матрица ковариации

$$C = \frac{1}{N}\sum_{i1}^{N} (x_i - \mu)(x_i - \mu)^T$$

Извлечение признаков: PCA

Математический вывод

Выпишем оптимизационную задачу с ограничениями на длину вектора |v| = 1

$$\begin{cases} v^T C v \to max_v \\ v^T v = 1 \end{cases}$$

Найдем Лагранжиан для данной задачи:

$$L(v, \lambda) = v^T C v - \lambda(v^T v - 1)$$

Извлечение признаков: PCA

Математический вывод

Найдем стационарные точки

$$\frac{\partial \mathcal{L}}{\partial v} = 2v^T C - 2\lambda v^T = 0$$

$$\frac{\partial \mathcal{L}}{\partial v} = v^T v - 1 = 0$$

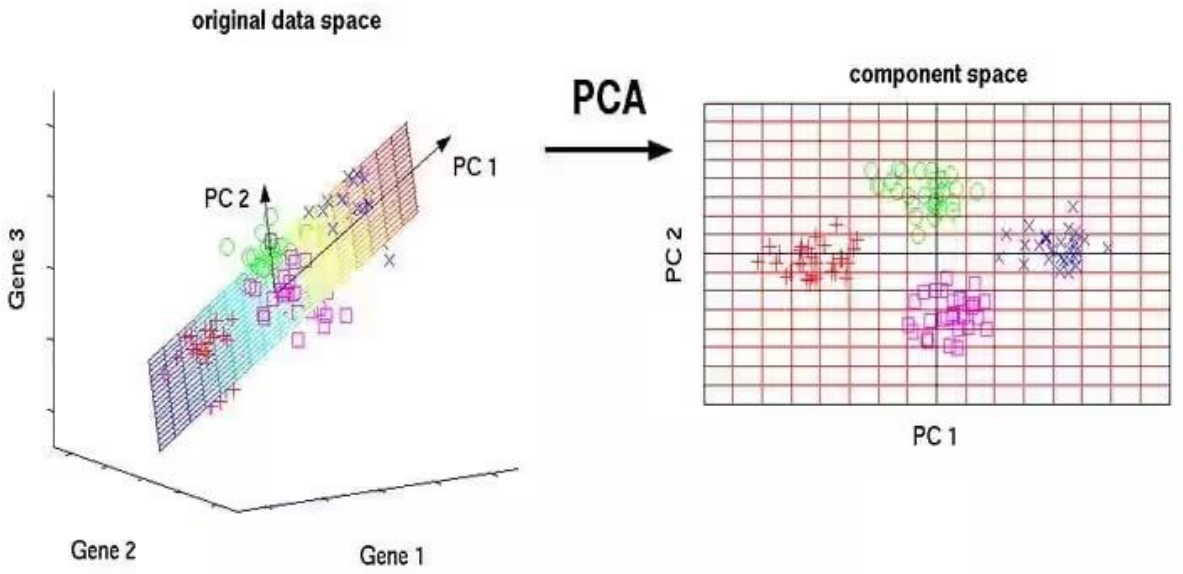
Откуда следует, что вектор v есть собственный вектор матрицы C

$$Cv = \lambda v$$

Извлечение признаков: PCA

Другая интерпретация:

Поиск главных компонент — это поиск подпространства меньшей размерности, где квадрат отклонений проекций до данного подпространства минимален



Источник: http://www.nlpca.org/pca-principal-component-analysis-matlab.html?v=1.0.18

Извлечение признаков: PCA

Пример работы:

Датасет Faces

Источник:
https://medium.com/@sebastiannorena/pca-principal-components-analysis-applied-to-images-of-faces-d2fc2c083371

Извлечение признаков: PCA

Пример работы:

Главные компоненты

(берем компоненты,

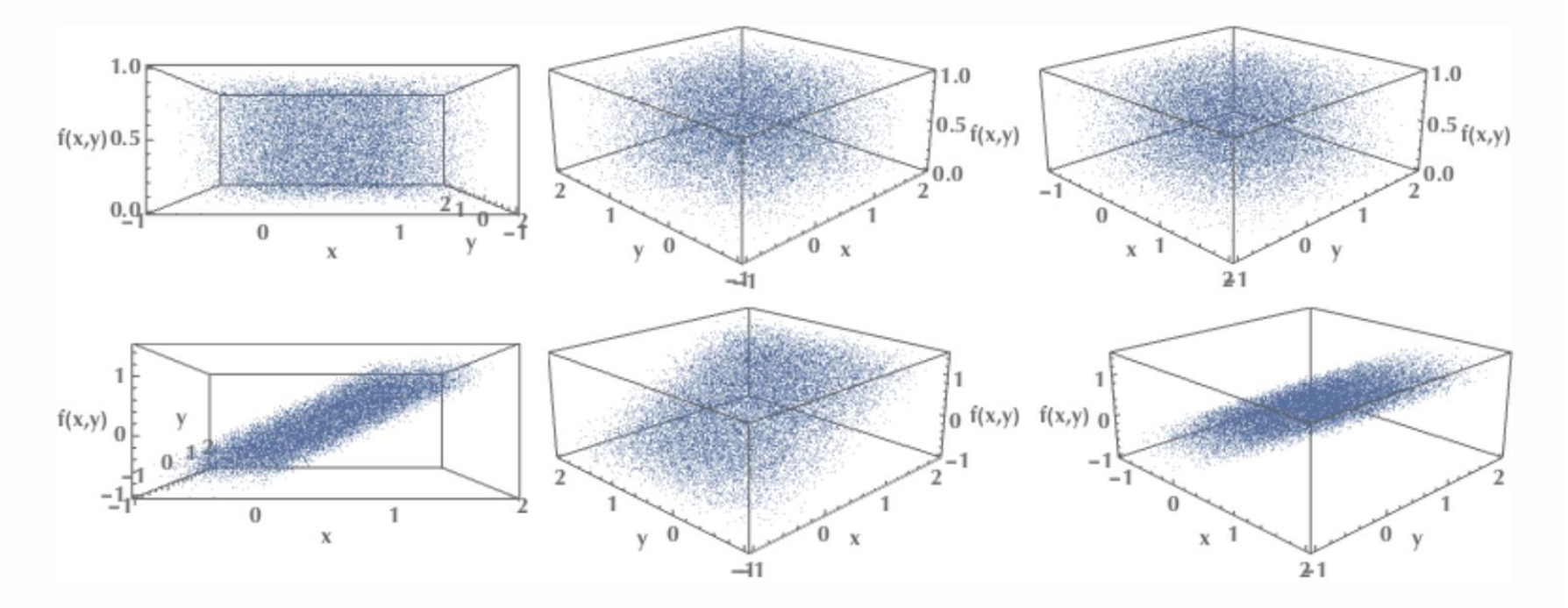
объясняющие 80

процентов дисперсии)

Источник:
https://medium.com/@sebastiannorena/pca-principal-components-analysis-applied-to-images-of-faces-d2fc2c083371

Извлечение признаков: PCA

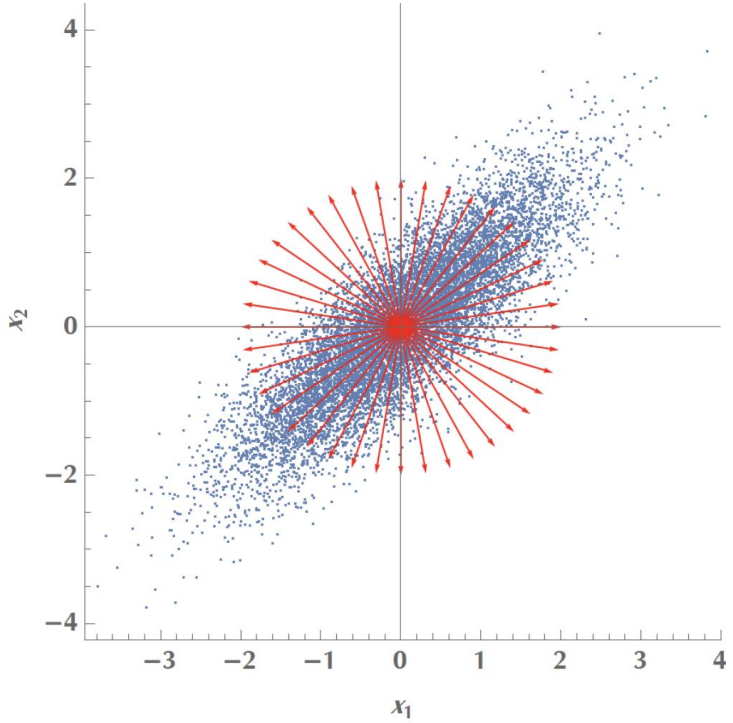
Имеет смысл использовать в тех случаях, когда в данных есть четко выраженная корреляция между признаками

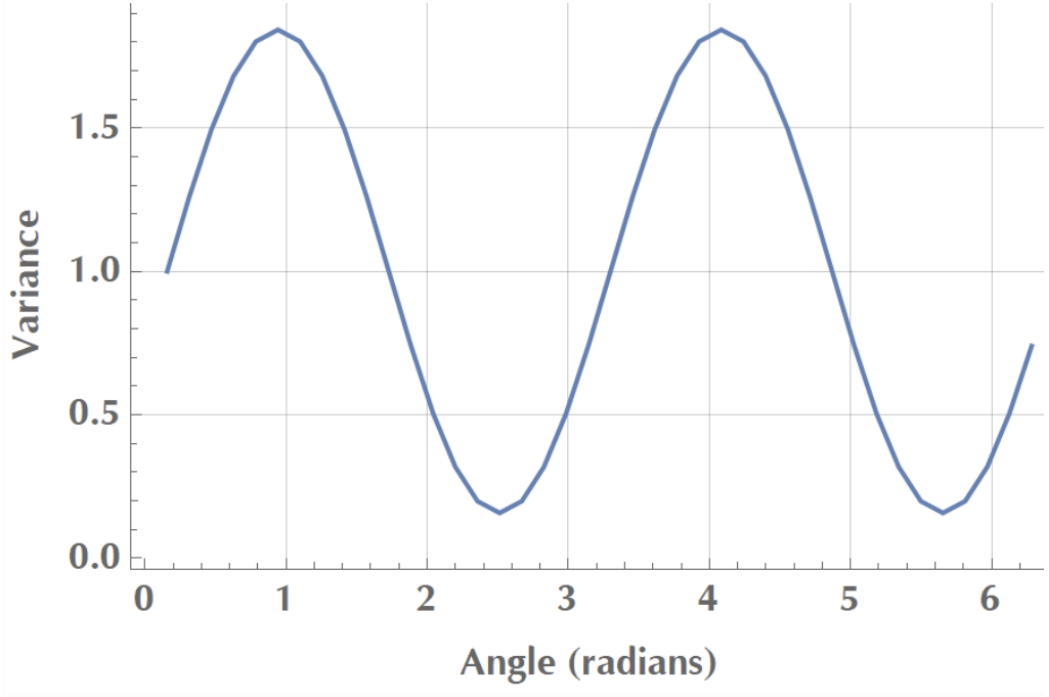


Источник: https://ekamperi.github.io/mathematics/2020/11/01/principal-component-analysis-lagrange-multiplier.html

Извлечение признаков: PCA

Доля сохраненной дисперсии. Сильная корреляция

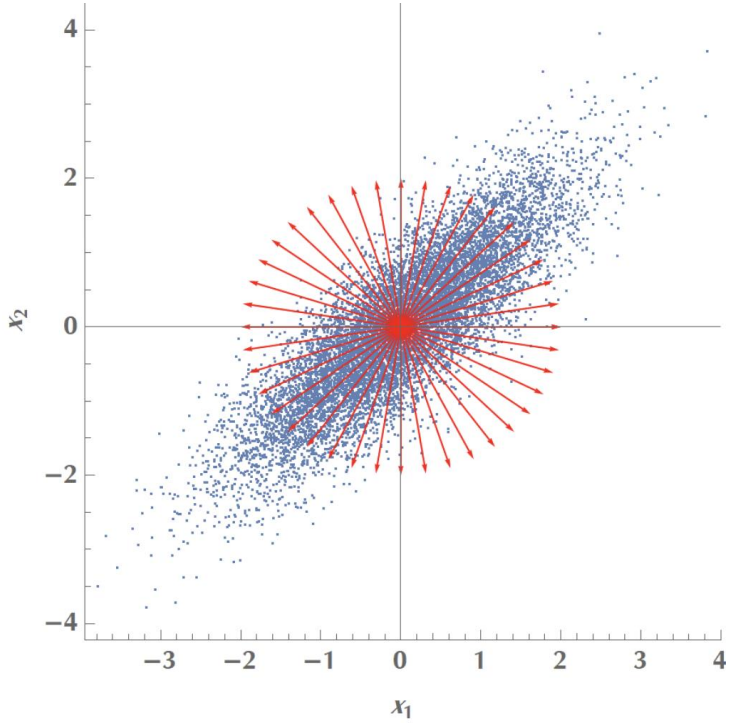


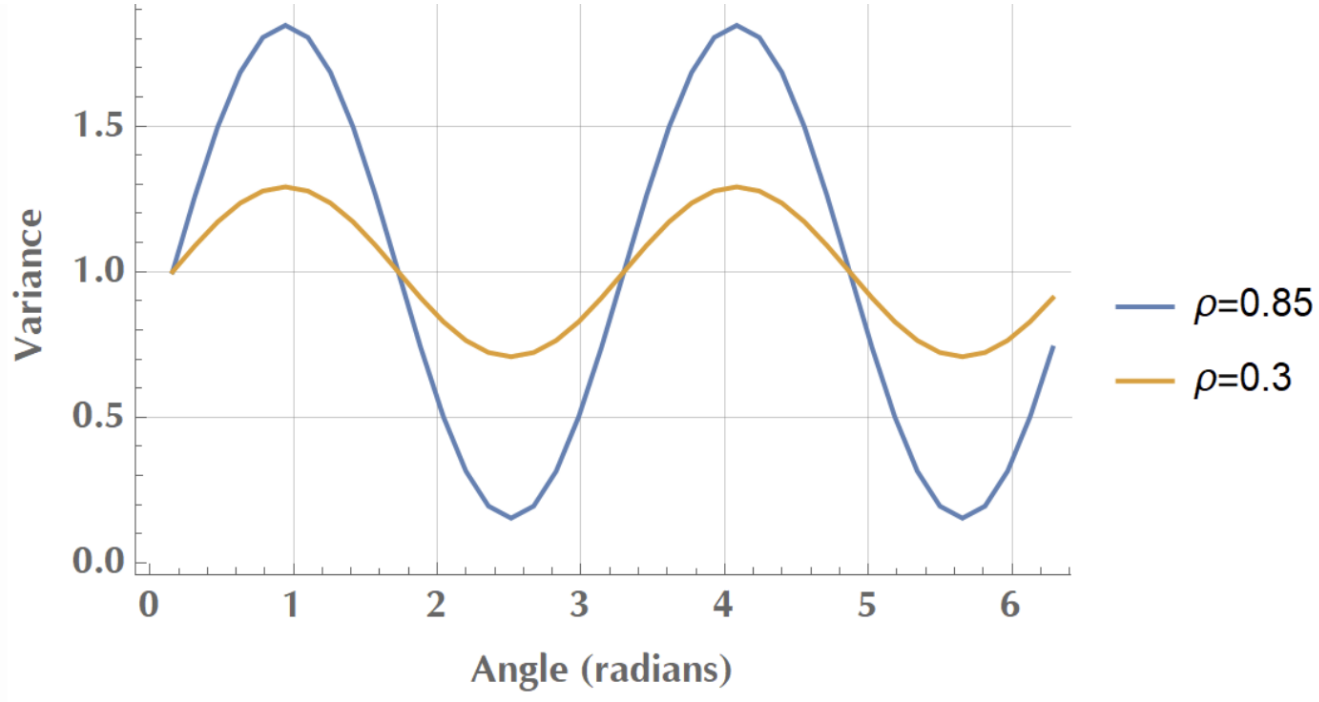


Источник: https://ekamperi.github.io/mathematics/2020/11/01/principal-component-analysis-lagrange-multiplier.html

Извлечение признаков: PCA

Доля сохраненной дисперсии. Слабая корреляция





Источник: https://ekamperi.github.io/mathematics/2020/11/01/principal-component-analysis-lagrange-multiplier.html

Извлечение признаков: PCA

Ограничения / недостатки:

- Новые компоненты — это линейные комбинации исходных признаков, поэтому интерпретировать их бывает сложно.
- PCA чувствителен к масштабу признаков: нужно обязательно стандартизовать данные. При слишком сильном уменьшении размеров могут потеряться важные детали, возможна потеря информации.
- Предполагается, что важная информация — в линейных комбинациях признаков; если связи между признаками нелинейные, PCA может быть неэффективен.

SVD-разложение

Теорема о сингулярном разложении матрицы (singular value decomposition, SVD)

Матрицу A ∈ $R^{m\times n}$ можно представить в виде

$$A = U\Sigma V^T$$

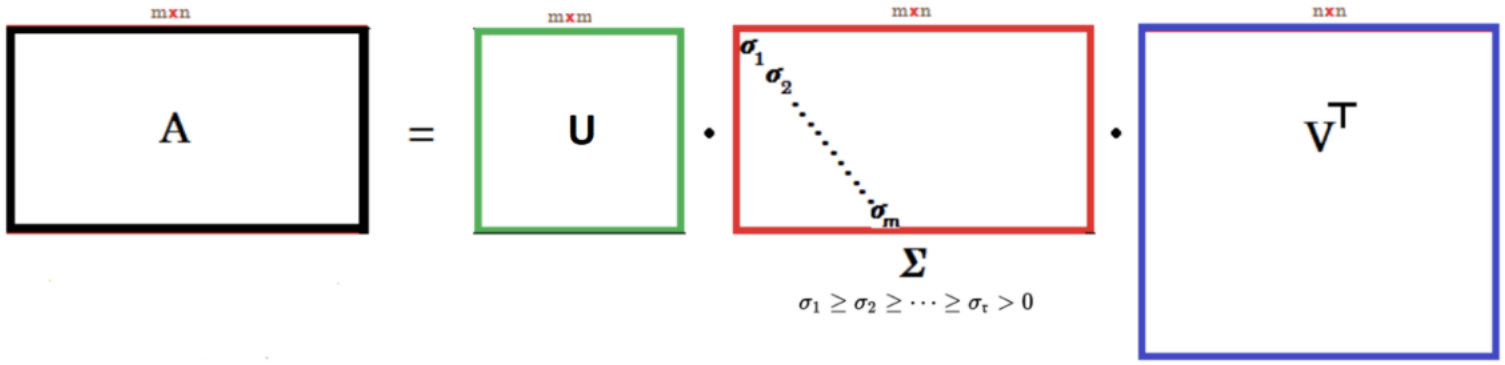
- $U \in R^{m\times m}$, V ∈ $R^{n\times n}$ – ортогональные матрицы
- Σ ∈ $R^{m\times n}$– диагональная матрица с ненулевыми элементами на диагонали $\sigma_i\sqrt{\lambda_i}$, где λ_i – собственные значения матрицы $A^T\ A$

При этом

- Столбцы матрицы U – собственные вектора AA^T
- Столбцы матрицы V – собственные вектора матрицы A^TA

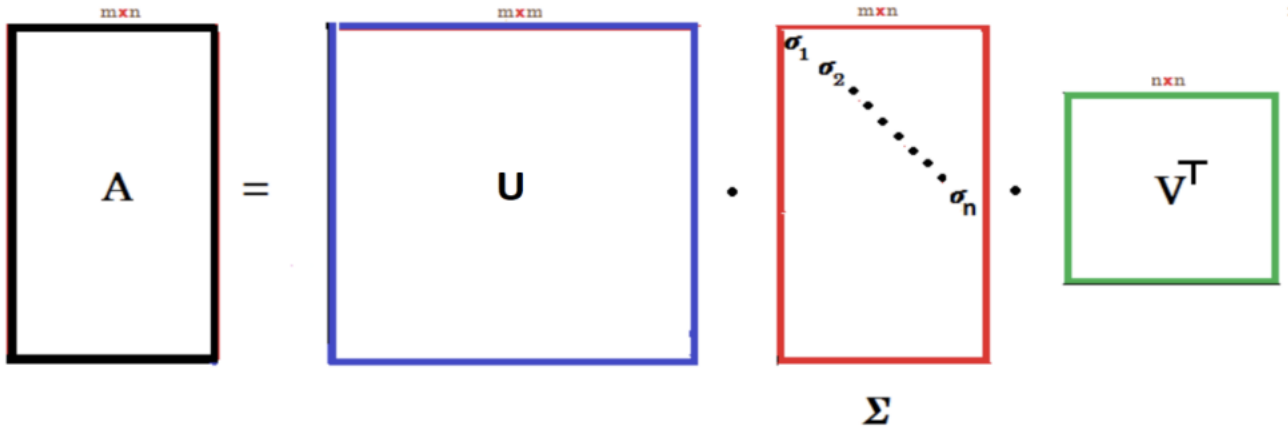
SVD-разложение

- При $m \leq n$:



$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$

- При $m > n$:



Источник: https://vmath.ru/vf4/algebra2/svd

PCA и SVD

Пусть X — матрица объект-признак, для которой мы хотим снизить размерность, и $\mathrm{X} = U\Sigma V^T$ — это ее SVD-разложение. Тогда:

Столбцы матрицы V — это собственные векторы матрицы $X^T X$, то есть векторы $v_1, \dots v_n$ — это главные компоненты.

Столбцы матрицы $U\Sigma$ — это новые признаки, то есть проекции исходных признаков на главные компоненты $Z = Xv$

$(X = U\Sigma V^T \Leftrightarrow U\Sigma = XV)$

Сингулярные числа матрицы Σ — это корни из собственных чисел матрицы $X^T X$

PCA и SVD

Для снижения размерности берем первые k столбцов матрицы U и верхний $k \times k$-квадрат матрицы Σ.

Тогда матрица $U_k \Sigma_k$ содержит k новых признаков, соответствующих первым k главным компонентам.

PCA и SVD

Вычислительно эффективнее при прочих равных использовать SVD, поскольку:

- Существуют вычислительные трудности с нахождением собственных значений, в этом недостаток PCA
- Существует итерационный алгоритм без нахождения собственных значений для нахождения SVD

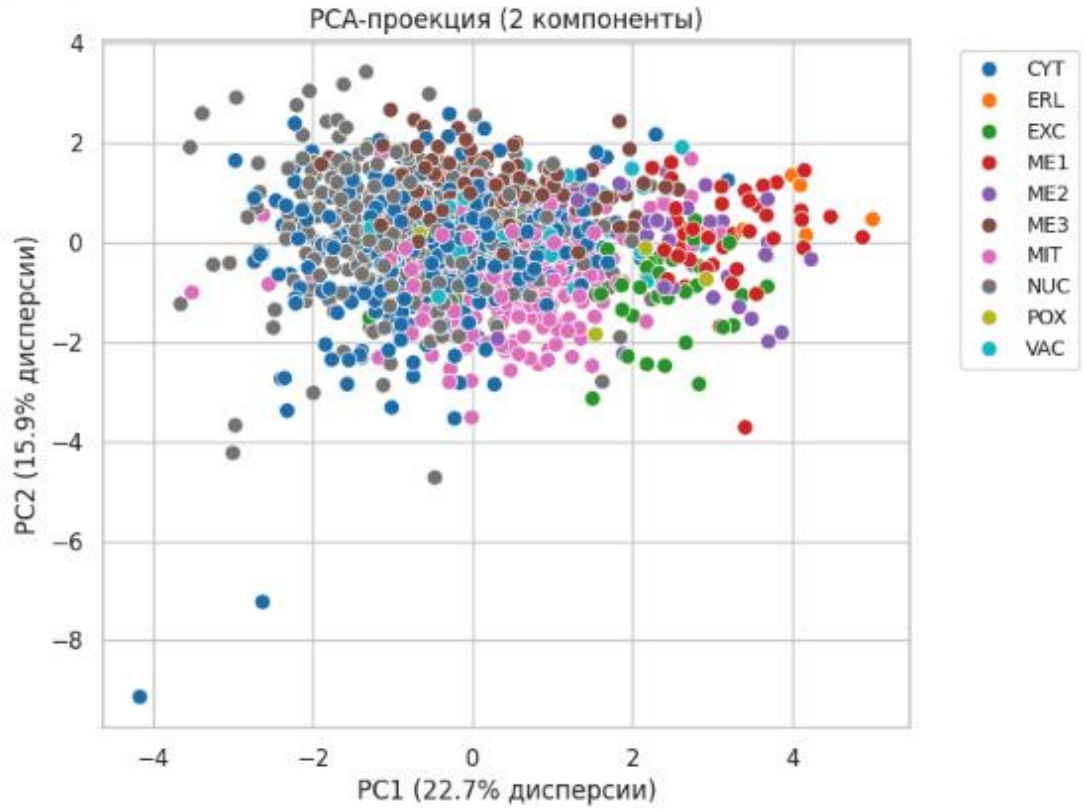
PCA для визуализации

Многомерные данные сводятся к двум или трем измерениям, что делает возможным их графическое преставление. Это помогает:

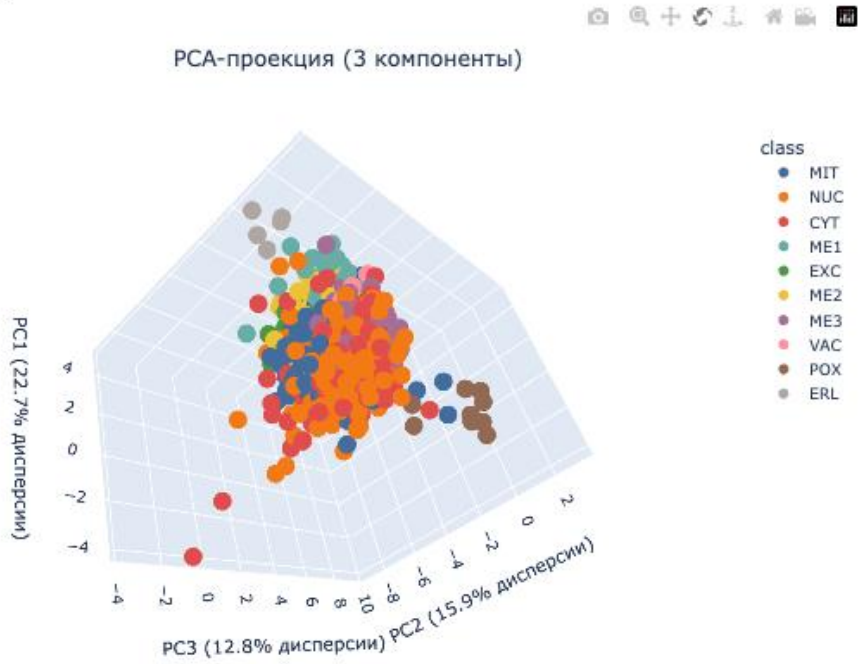
- упрощать анализ данных
- визуально выявлять кластера
- обнаруживать выбросы

PCA для визуализации

```
Объясненная дисперсия по компонентам: PC1 = 22.68%, PC2 = 15.88%
Суммарно: 38.56%
```

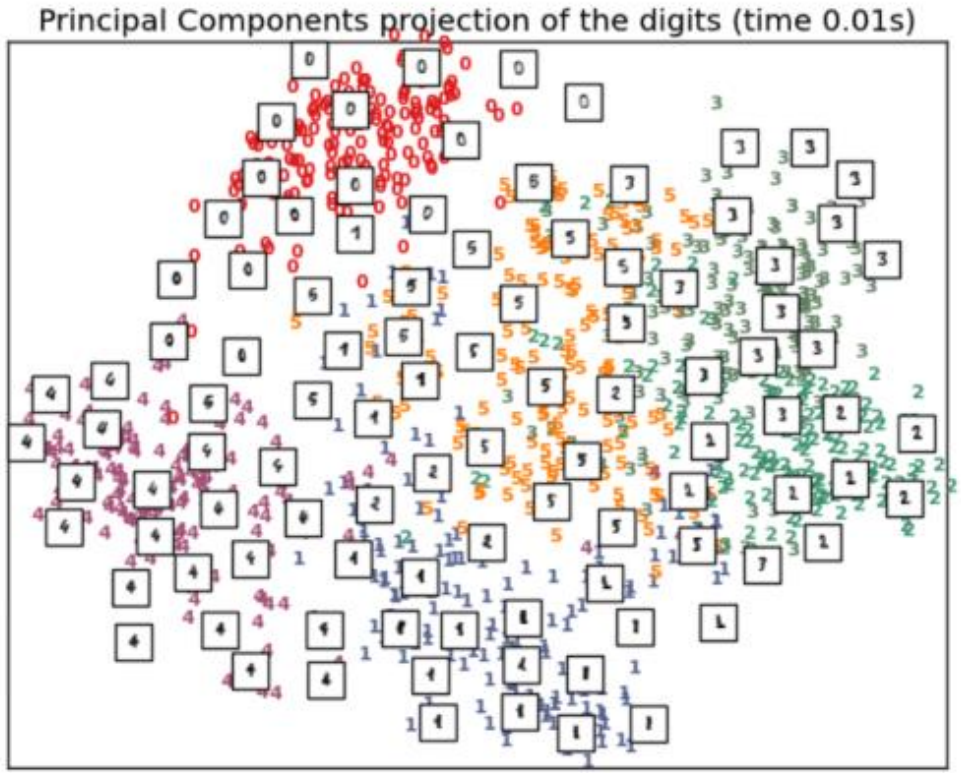



```
Объясненная дисперсия по компонентам: [22.68 15.88 12.77]%
Суммарно: 51.32%
```



Визуализация проекции датасета Yeast на 2 и 3 компоненты

PCA для визуализации

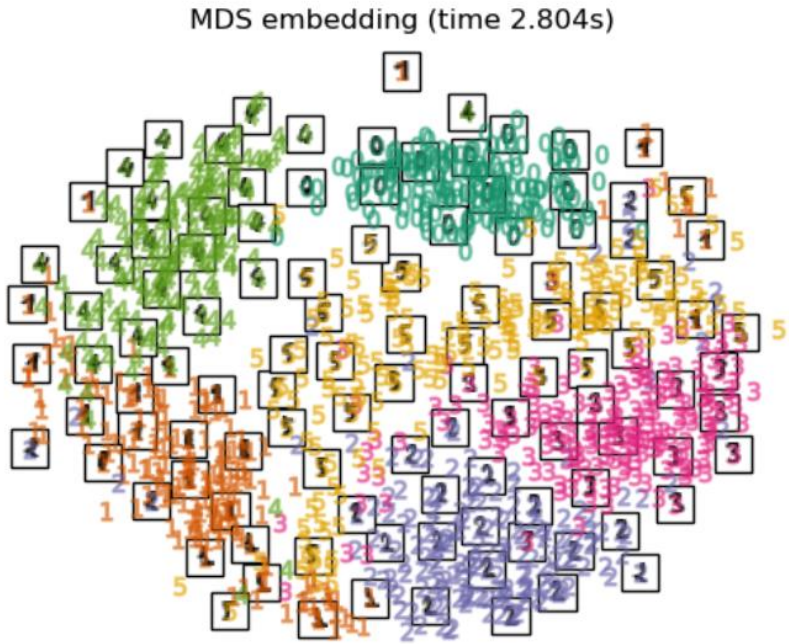


Визуализация датасета Digits. Источник: https://scikit-learn.org

Multidimensional scaling (MDS)

Идея метода многомерного шкалирования (MDS) — минимизация квадратов отклонений между исходными и новыми попарными расстояниями

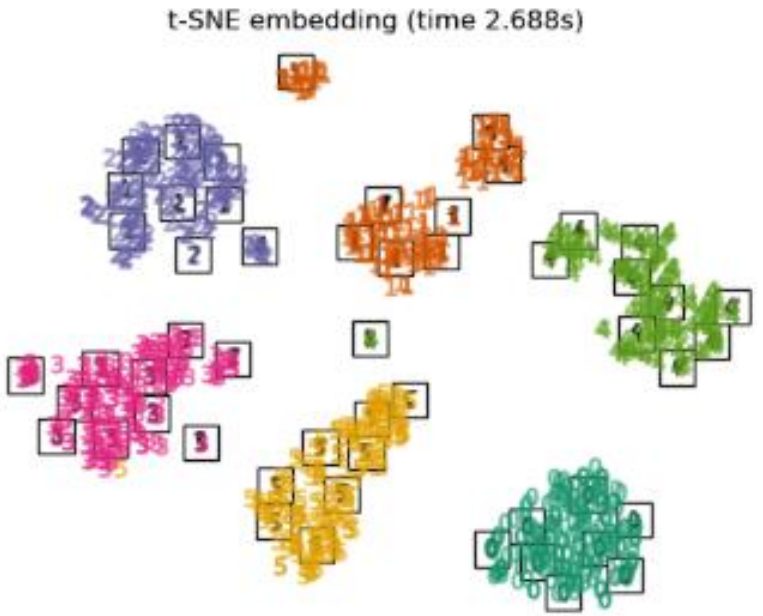
Целевая функция MDS — raw stress — определяется как $\sum_{i<j}(\widehat{d_{ij}} - d_{ij}(Z))^2$, где $d_{ij}(Z)$ — это попарные расстояния между координатами Z встроенных точек.



Визуализация того же датасета Digits с помощью MDS.
Источник: https://scikit-learn.org/stable/auto_examples/manifold/plot_lle_digits.html

T-SNE

T-SNE (t-distributed stochastic neighbor embedding) — метод, в котором важно сохранение не абсолютных расстояний между объектами, а пропорций между ними, за счет чего имеет меньшую склонность сжимать точки в центр, чем другие алгоритмы.



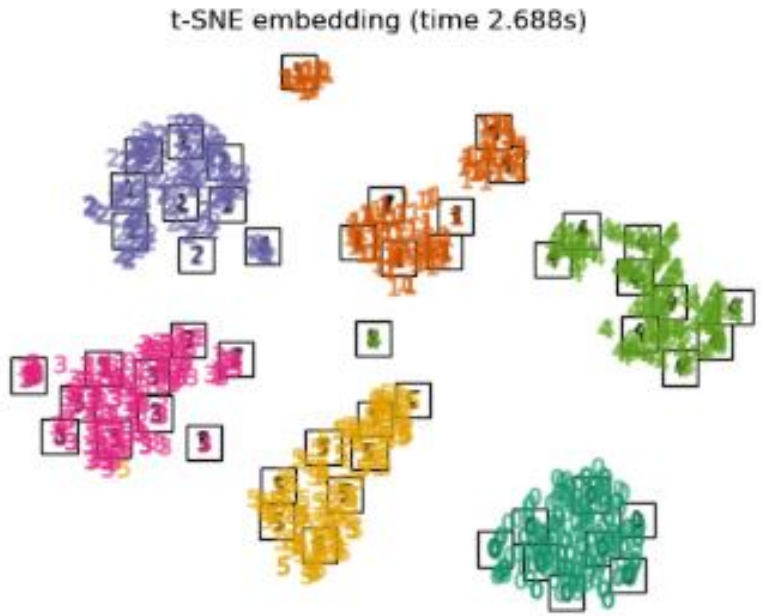
Визуализация того же датасета Digits с помощью MDS.
Источник: https://scikit-learn.org/stable/auto_examples/manifold/plot_lle_digits.html

T-SNE

Метод минимизирует дивергенцию Кульбака–Лейблера (KL-divergence) между совместными вероятностями в исходном и встроенном пространствах с помощью градиентного спуска.

Важно отметить, что по своим свойствам при разных начальных условиях алгоритм может прийти к разным локальным минимумам.

Поэтому полезно запускать алгоритм с разными начальными значениями и выбирать вариант с наименьшей KL-дивергенцией.



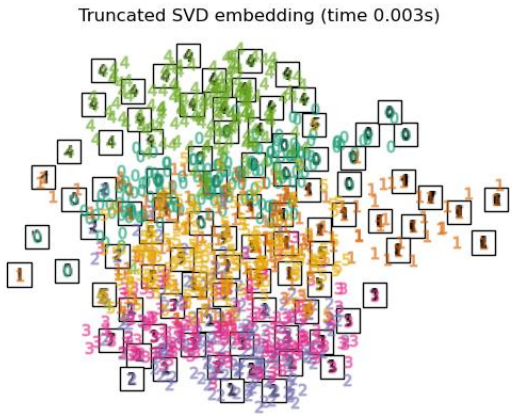
Визуализация того же датасета Digits с помощью MDS.
Источник: https://scikit-learn.org/stable/auto_examples/manifold/plot_lle_digits.html

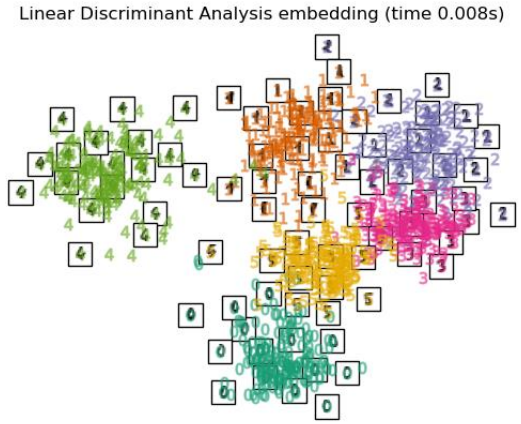
T-SNE

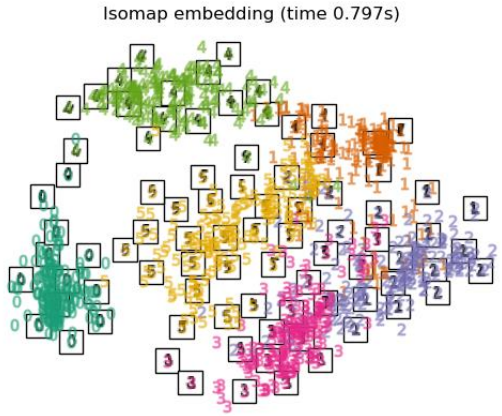
Недостатки:

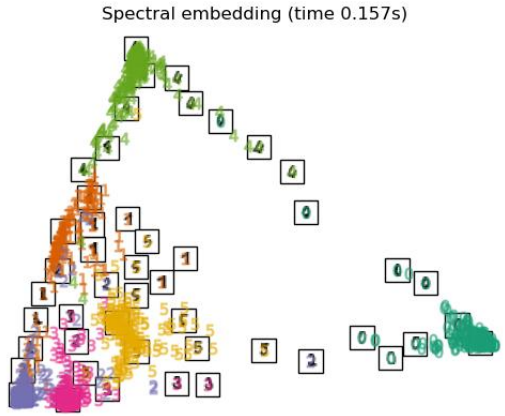
- Метод вычислительно дорог: на наборах данных размером в миллионы объектов вычисление может занимать часы, тогда как PCA завершится за секунды или минуты.
- Алгоритм стохастический: разные запуски с разными начальными значениями могут приводить к разным результатам.
- Глобальная структура данных явно не сохраняется. Этот недостаток частично устраняется инициализацией точек методом PCA (init='pca').

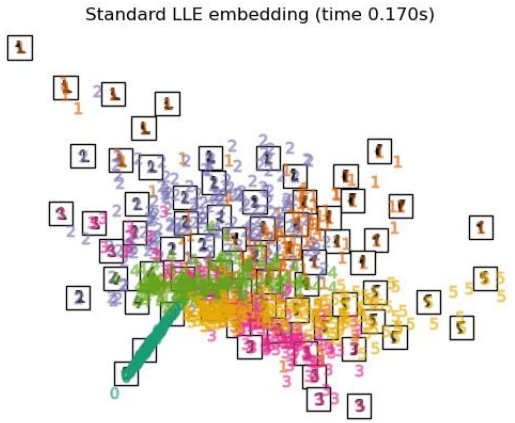
Другие методы

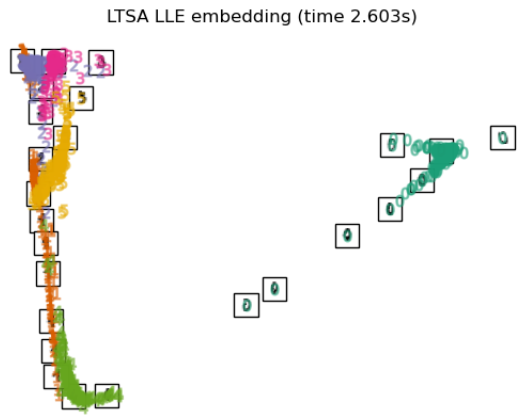


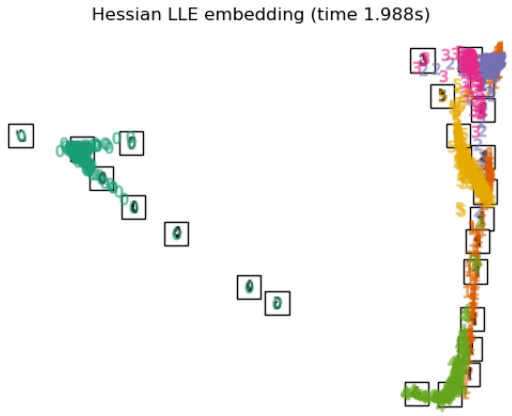


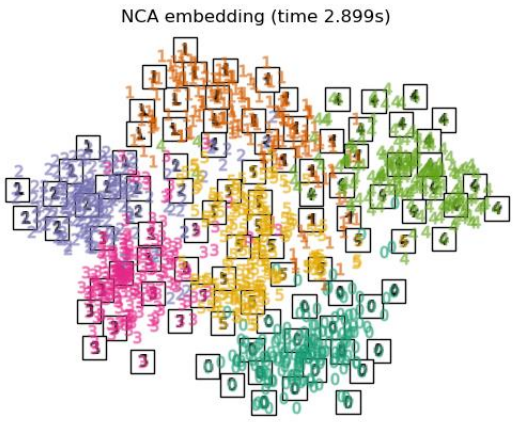












Визуализация датасета Digits. Источник: https://scikit-learn.org/stable/auto_examples/manifold/plot_lle_digits.html