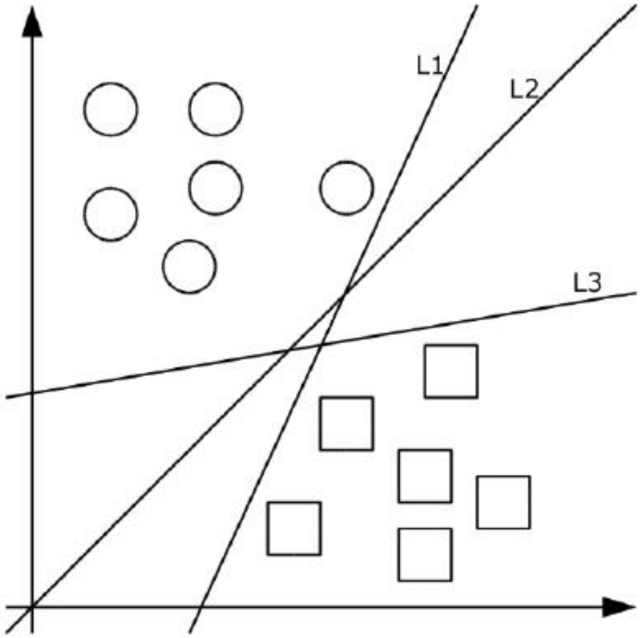
Введение в машинное обучение

Лекция 3. SVM. PCA

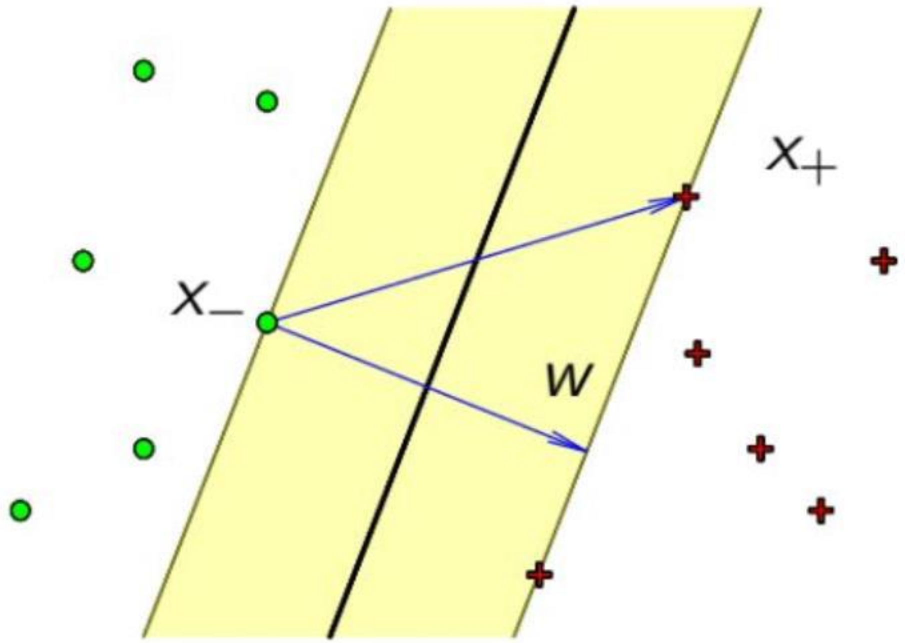
Осень 2025

Метод опорных векторов (SVM, support vector machine)

Нужно найти наилучшее решение, чтобы разделить классы



Решение классификации



Разделяющая гиперплоскость: $\mathbf{w \cdot x + w_0 = 0}$

Две вспомогательные плоскости, определяющие границу зазора:

- $\mathbf{w \cdot x + w_0 = +1}$ (для класса $y_i = +1$)
- $\mathbf{w \cdot x + w_0 = -1}$ (для класса $y_i = -1$)

расстояние от произвольной точки =
$|w \cdot x_0 + b| / ||w|| = 1/ ||w||$

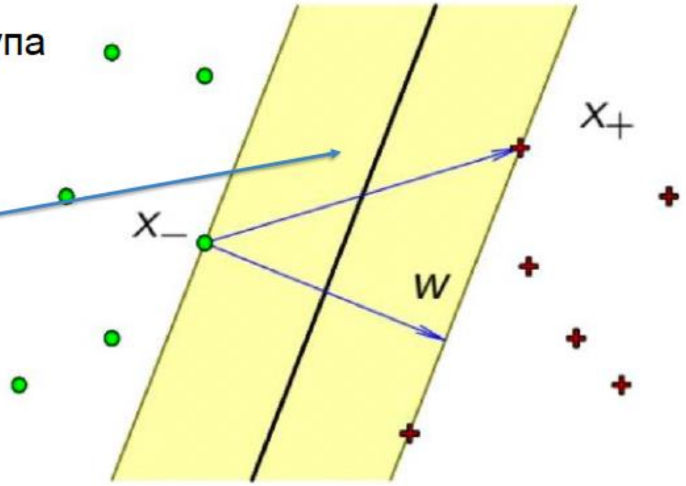
Зазор (Margin) - Расстояние между двумя опорными плоскостями = $2/ ||w||$ -> MAX

Решение классификации (идеальный случай)

Максимизация зазора -> минимизация нормы вектора

Поставим задачу максимизации такого отступа

$$\begin{cases} \frac{1}{2}\|w\|^2 \to \min\limits_{w,w_0}; \\ M_i(w, w_0) \geqslant 1, \quad i = 1, \dots, \ell. \end{cases}$$



Soft margin

Данные не являются идеально линейно разделимыми? Добавляем ослабляющую переменную.

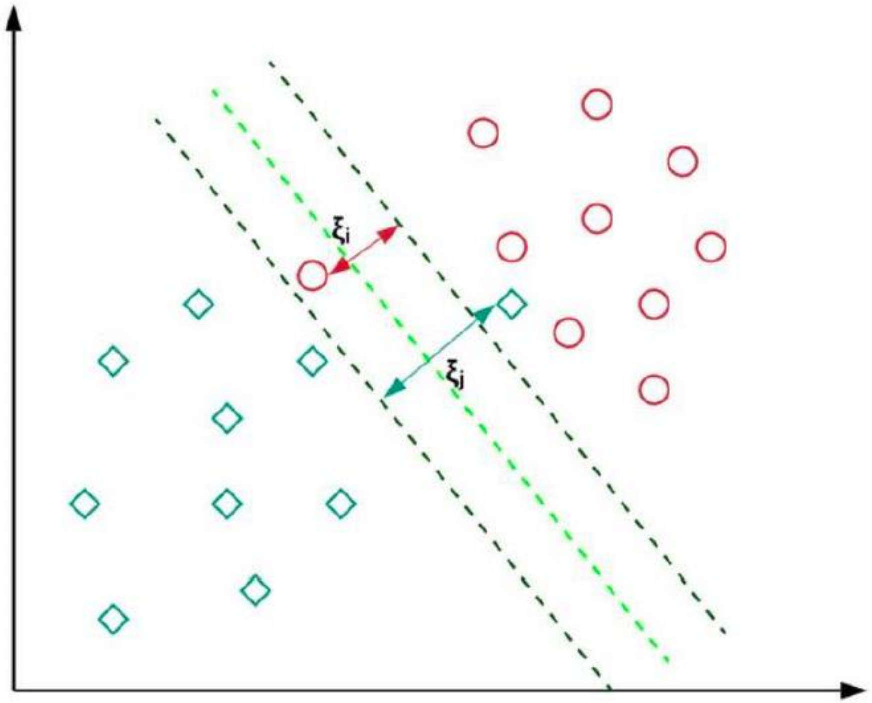
Поставим задачу максимизации такого отступа

$$\begin{cases} \frac{1}{2}\|w\|^2 \to \min\limits_{w,w_0}; \\ M_i(w, w_0) \geqslant 1, \quad i = 1, \dots, \ell. \end{cases}$$

Введем эмпирику

$$M_i(w, w_0) \geqslant 1 \boxed{- \xi_i,} \quad i = 1, \dots, \ell;$$

Допускаем, что классификатор ошибается



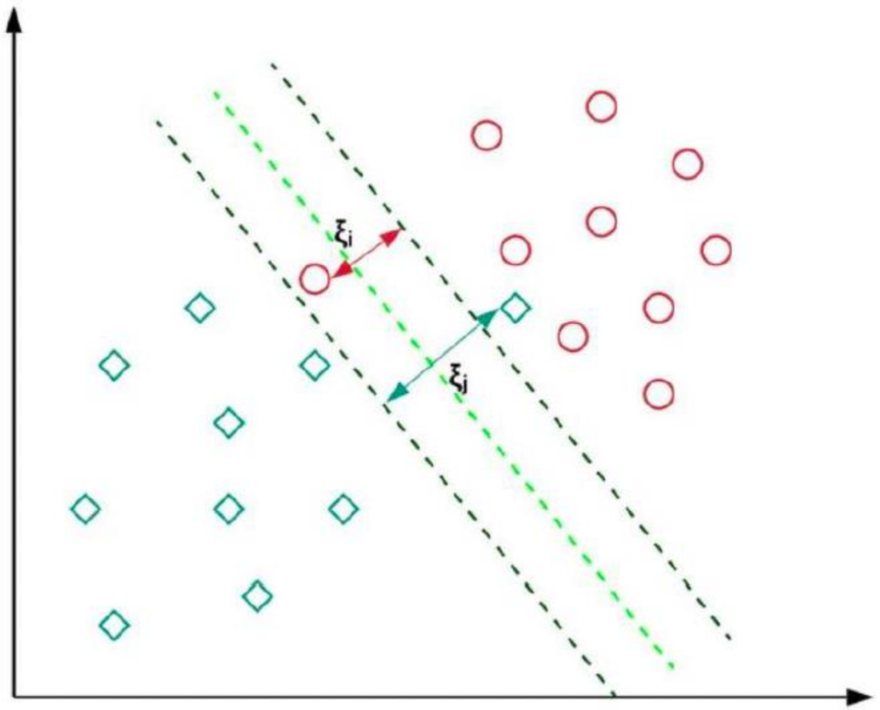
$$\begin{cases} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{\ell}\xi_i \to \min\limits_{w,w_0,\xi}; \\ M_i(w,w_0) \geqslant 1 - \xi_i, \quad i = 1,\dots,\ell; \\ \xi_i \geqslant 0, \quad i = 1,\dots,\ell. \end{cases}$$

Эквивалентная задача безусловной оптимизации

$$\boxed{C\sum_{i=1}^{\ell}\left(1 - M_i(w,w_0)\right)_+} + \frac{1}{2}\|w\|^2 \to \min_{w,w_0}.$$

Кусочно-линейная

C – скалярная величина, сила регуляризации (гиперпараметр)



Геометрическая постановка

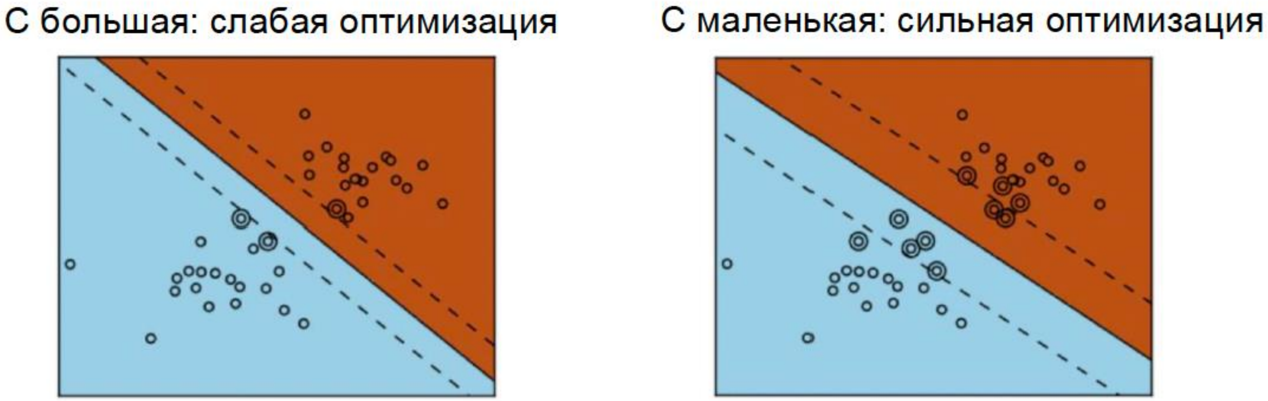
[Машинное обучение. Метод опорных векторов. К.В. Воронцов, Школа анализа данных, Яндекс]

$$\begin{cases} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{\ell}\xi_i \to \min\limits_{w,w_0,\xi}; \\ M_i(w,w_0) \geqslant 1 - \xi_i, \quad i = 1,\dots,\ell; \\ \xi_i \geqslant 0, \quad i = 1,\dots,\ell. \end{cases}$$

Параметр C — гиперпараметр регуляризации.

- Малый C: Больший зазор, но большее количество ошибок на обучении (сильная регуляризация, модель проще).

- Большой C: Меньший зазор, но меньшее количество ошибок на обучении (слабая регуляризация, модель может переобучиться).
C контролирует компромисс между шириной зазора и числом ошибок классификации.



Задача квадратичного программирования. Теорема Карyша-Куна-Таккера

Функция Лагранжа: $\mathscr{L}(w, w_0, \xi; \lambda, \eta) =$

$$= \frac{1}{2}\|w\|^2 - \sum_{i=1}^{\ell} \lambda_i \big(M_i(w, w_0) - 1\big) - \sum_{i=1}^{\ell} \xi_i (\lambda_i + \eta_i - C),$$

λ_i — переменные, двойственные к ограничениям $M_i \geqslant 1 - \xi_i$;
η_i — переменные, двойственные к ограничениям $\xi_i \geqslant 0$.

$$\begin{cases} \frac{\partial \mathscr{L}}{\partial w} = 0, \quad \frac{\partial \mathscr{L}}{\partial w_0} = 0, \quad \frac{\partial \mathscr{L}}{\partial \xi} = 0; \\ \xi_i \geqslant 0, \quad \lambda_i \geqslant 0, \quad \eta_i \geqslant 0, \quad i = 1, \dots, \ell; \\ \lambda_i = 0 \ \text{либо} \ M_i(w, w_0) = 1 - \xi_i, \quad i = 1, \dots, \ell; \\ \eta_i = 0 \ \text{либо} \ \xi_i = 0, \quad i = 1, \dots, \ell; \end{cases}$$

Машинное обучение. Метод опорных векторов. К.В. Воронцов, Школа анализа данных, Яндекс

Типизация объектов:

1. $\lambda_i = 0;\ \ \eta_i = C;\ \ \xi_i = 0;\ \ M_i \geqslant 1.$
— периферийные (неинформативные) объекты.

2. $0 < \lambda_i < C;\ \ 0 < \eta_i < C;\ \ \xi_i = 0;\ \ M_i = 1.$
— опорные граничные объекты.

3. $\lambda_i = C;\ \ \eta_i = 0;\ \ \xi_i > 0;\ \ M_i < 1.$
— опорные-нарушители.

λ = 0 => выполняется Mi = yi(w·x_i + b) > 1 – точка далеко ЗА границей

[Машинное обучение. Метод опорных векторов. К.В. Воронцов, Школа анализа данных, Яндекс]

Теорема Каруша-Куна-Таккера

- это набор условий, которые гарантируют, что решение оптимально

4 правила ККТ:

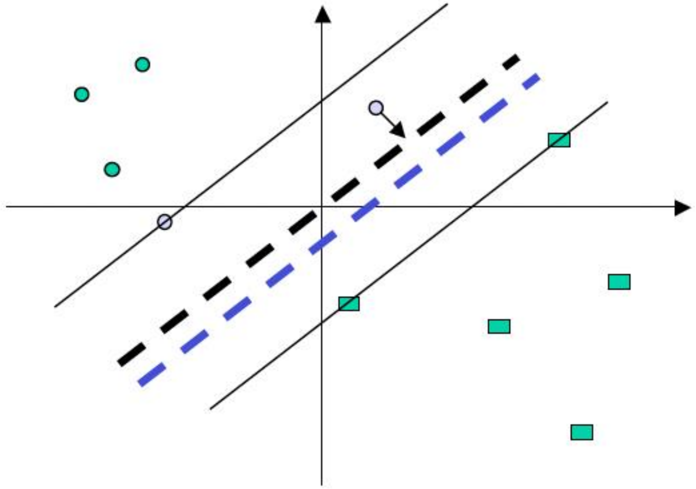
1) стационарность – производные должны быть нулевые
2) допустимость – ограничения должны выполняться

$$(y_i(w \cdot x_i + b) \geq 1)$$

3) дополняющая нежесткость

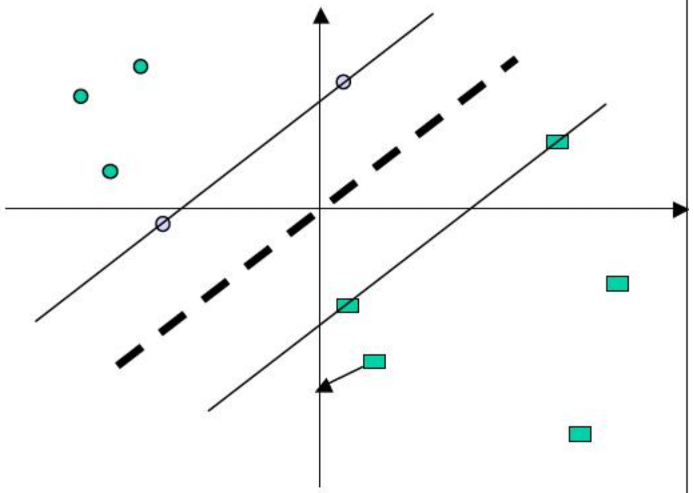
$$\alpha_i \times [y_i(w \cdot x_i + b) - 1] = 0$$

4) неотрицательность $\alpha_i \geq 0$



Moving a support vector moves the decision boundary

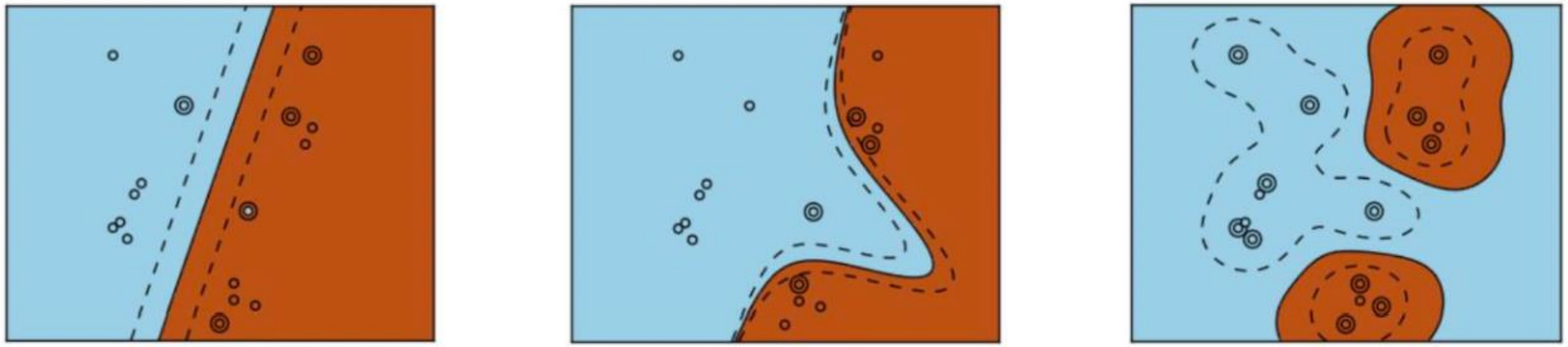
Moving the other vectors has no effect



$$a(x) = \mathrm{sign}\Big(\sum_{i=1}^{\ell} \lambda_i y_i \langle x, x_i \rangle - w_0 \Big).$$

$K(x, x') =< x, x' >^2$ - quadratic
$K(x, x') =< x, x' >^d$ - polynomial with degree d
$K(x, x') = (< x, x' > +1)^d$ - polynomial with degree $\leq d$
$K(x, x') = exp(-\gamma ||x - x'||^2)$ - Radial Basis Functions (RBF) kernel

$\langle x, x' \rangle$ $\quad$ $\big(\langle x, x' \rangle + 1\big)^d, \; d{=}3$ $\quad$ $\exp\big(-\gamma \|x - x'\|^2\big)$



Когда выбрать данный метод?

Когда принимаем во внимание только »самые проблемные» точки, ближе к границе

Если важны все точки – NN, линейная регрессия

Пример

Задача: Классификация рукописных цифр (например, из набора данных MNIST).

1. **Данные:** Изображения 28x28 пикселей (784 признака) с метками от 0 до 9.

2. **Применение SVM:** Мы можем использовать SVM

3. **Процесс:**
    - Обучаем 10 бинарных классификаторов (каждый "один против всех"): один классификатор отличает цифру '0' от всех остальных, другой — '1' от всех остальных и т.д.
    - Для нового изображения запускаем все 10 классификаторов.
    - Цифра присваивается тому классу, классификатор которого выдал наибольшее значение решающей функции (наибольшую "уверенность").

4. **Результат:** SVM является одним из классических и эффективных методов для этой задачи, показывая высокую точность.

Фильтрация данных

Дисперсия признаков (для +/- константных признаков, не учитываются ответы)

$$R_j = \frac{1}{l} \sum_{i=1}^{l} \left(x_{ij} - \bar{x}_j \right)^2$$

Корреляция (только линейная связь, для классификации лучше T-score, многоклассовой F-score)

$$R_j = \frac{\sum_{i=1}^{l} \left(x_{ij} - \bar{x}_j \right) \left(y_i - \bar{y} \right)}{\sqrt{\sum_{i=1}^{l} \left(x_{ij} - \bar{x}_j \right)^2 \sum_{i=1}^{l} \left(y_i - \bar{y} \right)^2}}$$

«Одномерные» критерии

Эти критерии отображают влияние одного признака, но не их совместное влияние

Перебор

Убираем попеременно признаки и строим модели. Наблюдаем влияние

Метод главных компонент (principal component analysis)

- метод снижения размерности данных
- статистический инструмент для выделения наиболее значимой информации
- линейное преобразование, находящее новые ортогональные оси координат

Зачем это нужно?
- визуализация многомерных данных
- ускорение работы алгоритмов машинного обучения
- удаление шума и избыточности
- выявление скрытых закономерностей

PCA

генерирует новые признаки на основе линейной комбинации существующих

- исходные признаки x_{ij}, D

-> новые z_{ij}, d

$$z_{ij} = \sum_{k=1}^{D} w_{ij} x_{ij}$$

Как работает PCA

- стандартизует данные. Вычитает среднее значение для признака и делит на стандартное отклонение
- строится ковариационная матрица. Она показывает, насколько признаки или переменные расходятся друг с другом (растянуты).

$$\begin{bmatrix} cov(x_1, x_1) & \dots & cov(x_1, x_n) \\ \dots & & \\ cov(x_n, x_1) & \dots & cov(x_n, x_n) \end{bmatrix}$$

- вычисляются векторы, направленные в сторону наибольшей дисперсии – это собственные векторы.

Поиск собственных векторов

$$X^T X \cdot e = \lambda \cdot e$$

e – ортогональный базис; $X^T X$ – ковариационная матрица; λ – собственные значения матрицы

Найдя все собственные значения выполняется сортировка их по убыванию. Максимальная λ – первый PCA и так далее

Все компоненты - ортогональны

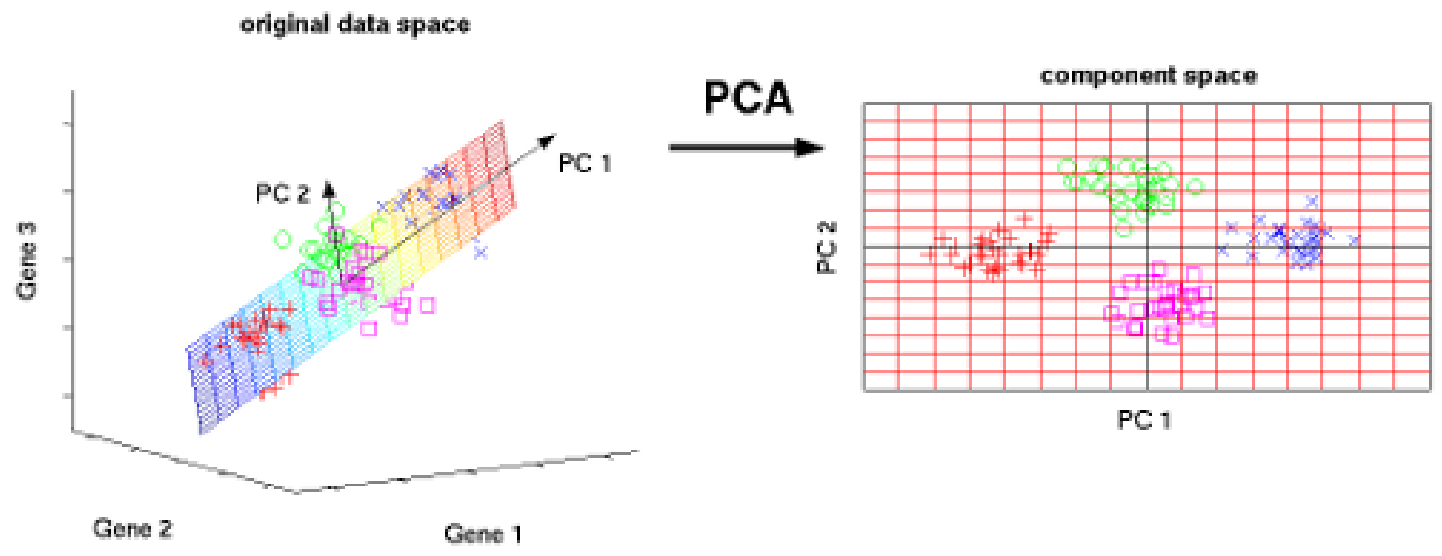
Пример

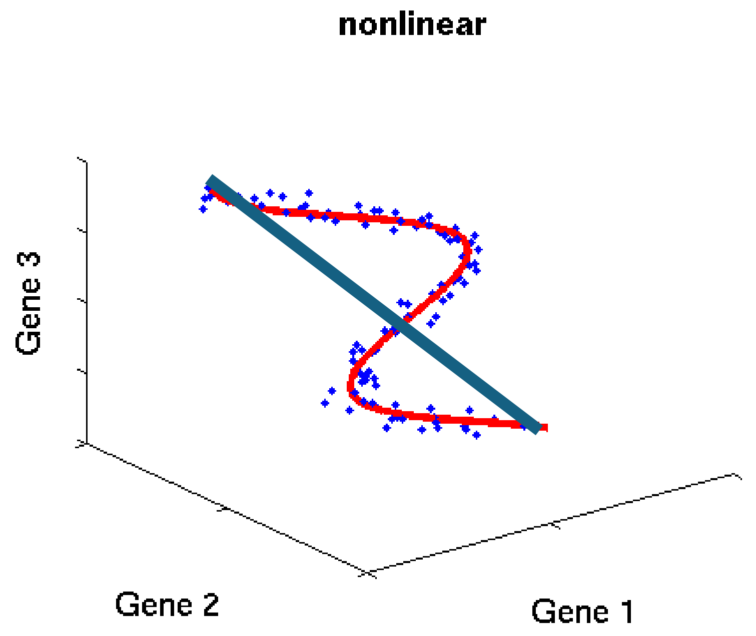
Собственные значения: [4.5, 2.1, 0.8, 0.4, 0.2]

Дисперсия:

- PC1: 4.5/8.0 = 56.25%
- PC1+PC2: (4.5+2.1)/8.0 = 82.5% ✓
- PC1+PC2+PC3: 91.25%

Количество выбирается исходя из задачи, требований к производительности, качеству, стандартов области





http://phdthesis-bioinformatics-maxplanckinstitute-molecularplantphys.matthias-scholz.de/

- Часто данные содержат помехи и неинформативны
- Удалим компоненты с низкой дисперсией в PCA

Links

- https://www.kaggle.com/learn/feature-engineering