

# Машинное обучение

Лекция 9

Отбор признаков. Уменьшение размерности.

Михаил Гущин

[mhushchyn@hse.ru](mailto:mhushchyn@hse.ru)

НИУ ВШЭ, 2023

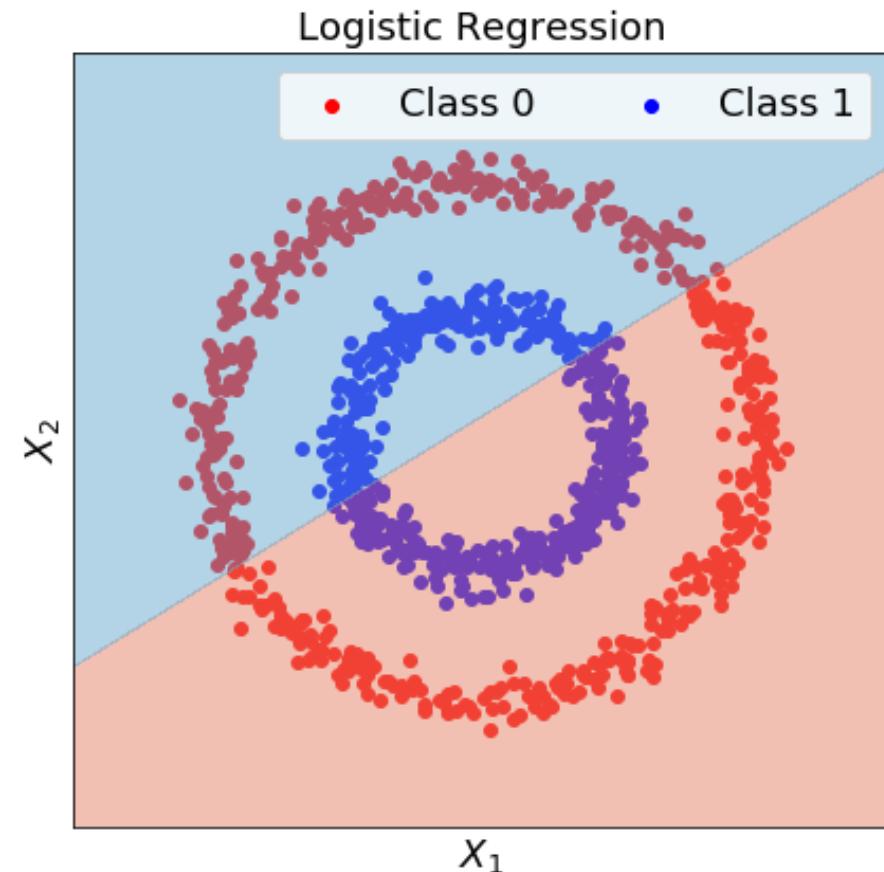


НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ  
УНИВЕРСИТЕТ

# Создание признаков (Feature engineering)

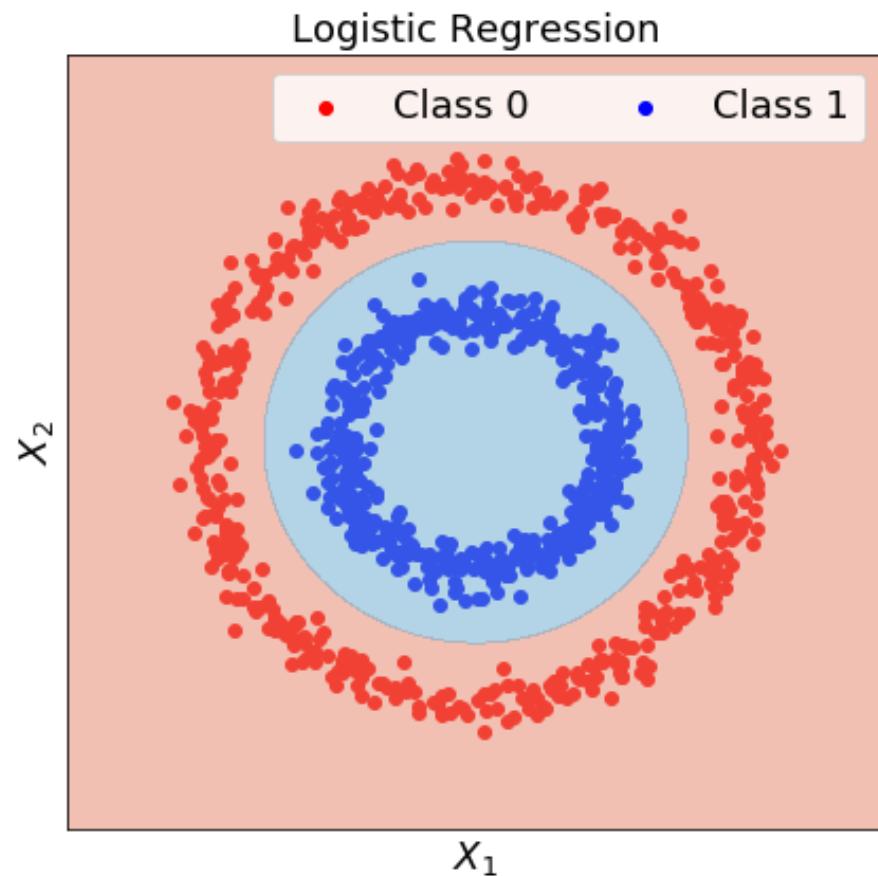
# Пример 1

- ▶ Рассмотри задачу бинарной классификации в 2D методом логистической регрессии
- ▶ Классификатор не может разделить классы используя признаки  $X_1$  и  $X_2$



# Пример 1

- ▶ Создадим новый признак  $X_3$ :  
$$X_3 = X_1^2 + X_2^2$$
- ▶ Он позволяет разделить классы прямой линией
- ▶ Теперь логистическая регрессия решает задачу идеально используя признаки  $X_1, X_2$  и  $X_3$

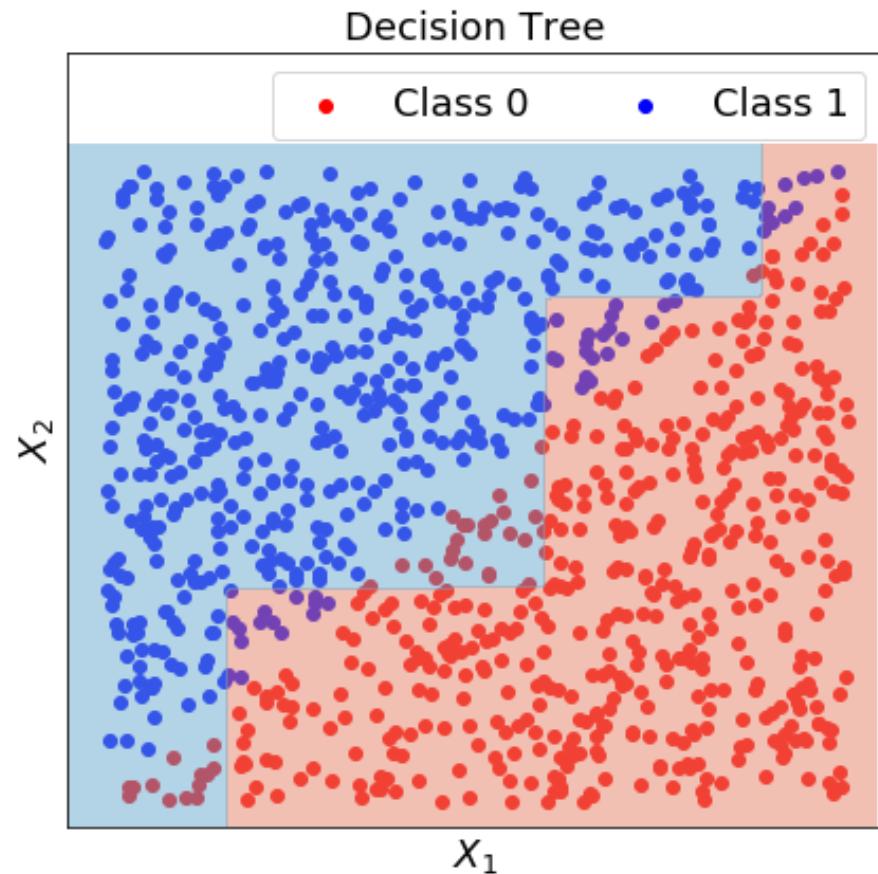


# Пример 2

- ▶ Рассмотрим пример, где классы разделяются поверхностью:

$$X_2 - X_1 = 0$$

- ▶ Такая поверхность трудная для решающих деревьев
- ▶ Требуется большая глубина дерева, чтобы разделить классы

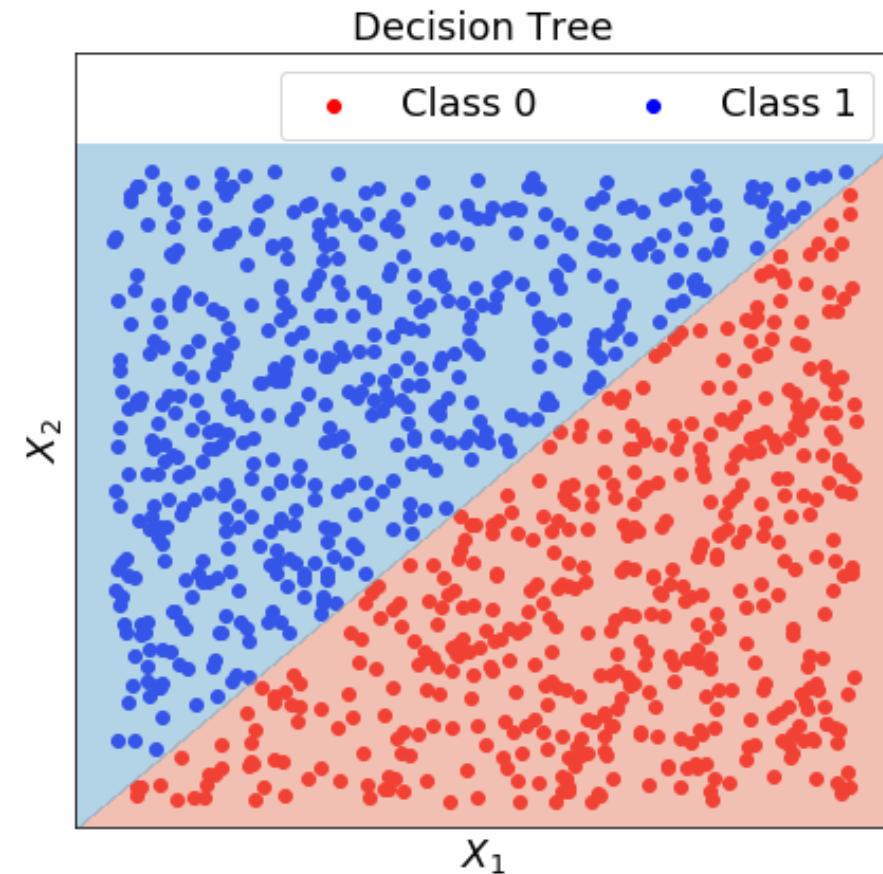


# Пример 2

- ▶ Создадим новый признак  $X_3$ , который поможет дереву:

$$X_3 = X_2 - X_1$$

- ▶ Теперь для решения задачи нужен только один предикат:  $X_3 > 0$
- ▶ Требуется дерево глубины 1, чтобы решить задачу идеально 😊



# Преимущества

Создание новых признаков позволяет:

- ▶ Улучшить качество моделей
- ▶ Снизить сложность моделей
- ▶ Ускорить обучение моделей
- ▶ Уменьшить размерность задачи, исключив менее информативные признаки ( $X_1, X_2$  в примерах)

# Принципы

Основные принципы создания новых признаков:

- ▶ Используйте любую информацию о задаче (классы в форме окружностей)
- ▶ Создавайте признаки со смыслом ( $\sqrt{X_1^2 + X_2^2}$  - радиус)
- ▶ Компенсируйте ограничения моделей (как для дерева в примере 2)

# Типичные примеры

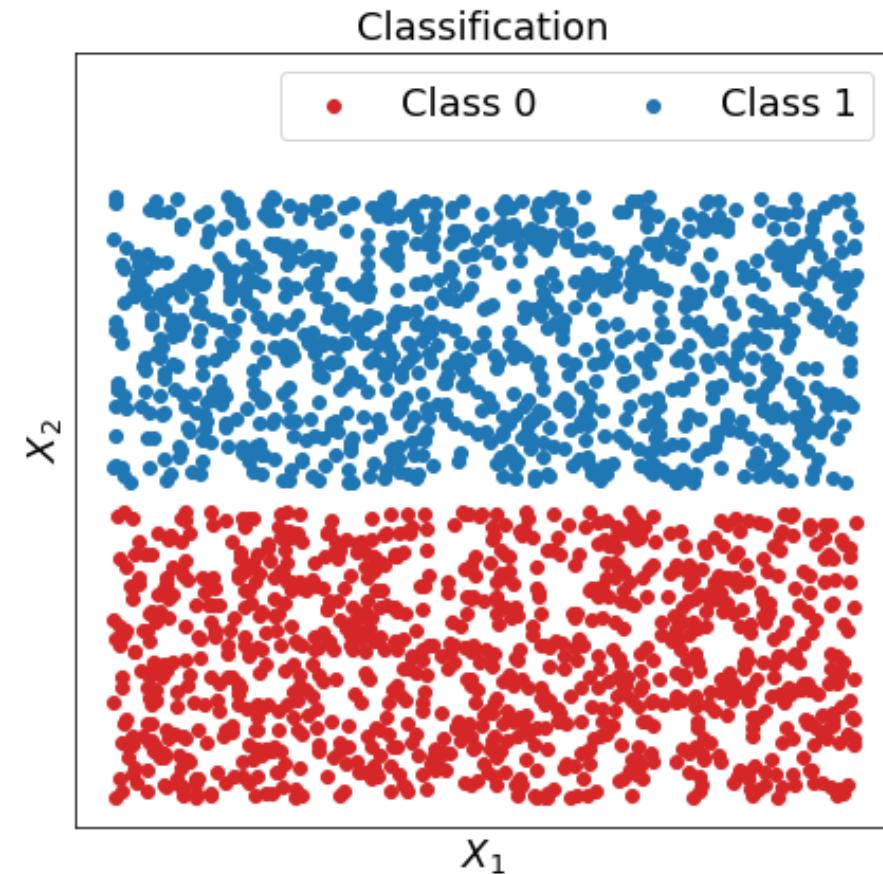
Наиболее популярные комбинации:

- ▶  $X_i^p$
- ▶  $X_1 X_2$
- ▶  $X_1^2 \pm X_2^2$
- ▶  $X_1 \pm X_2$
- ▶  $\frac{X_1 \pm X_2}{X_1 \mp X_2}$
- ▶  $\sin X_1, \cos X_1$

# Важность признаков (Feature importance)

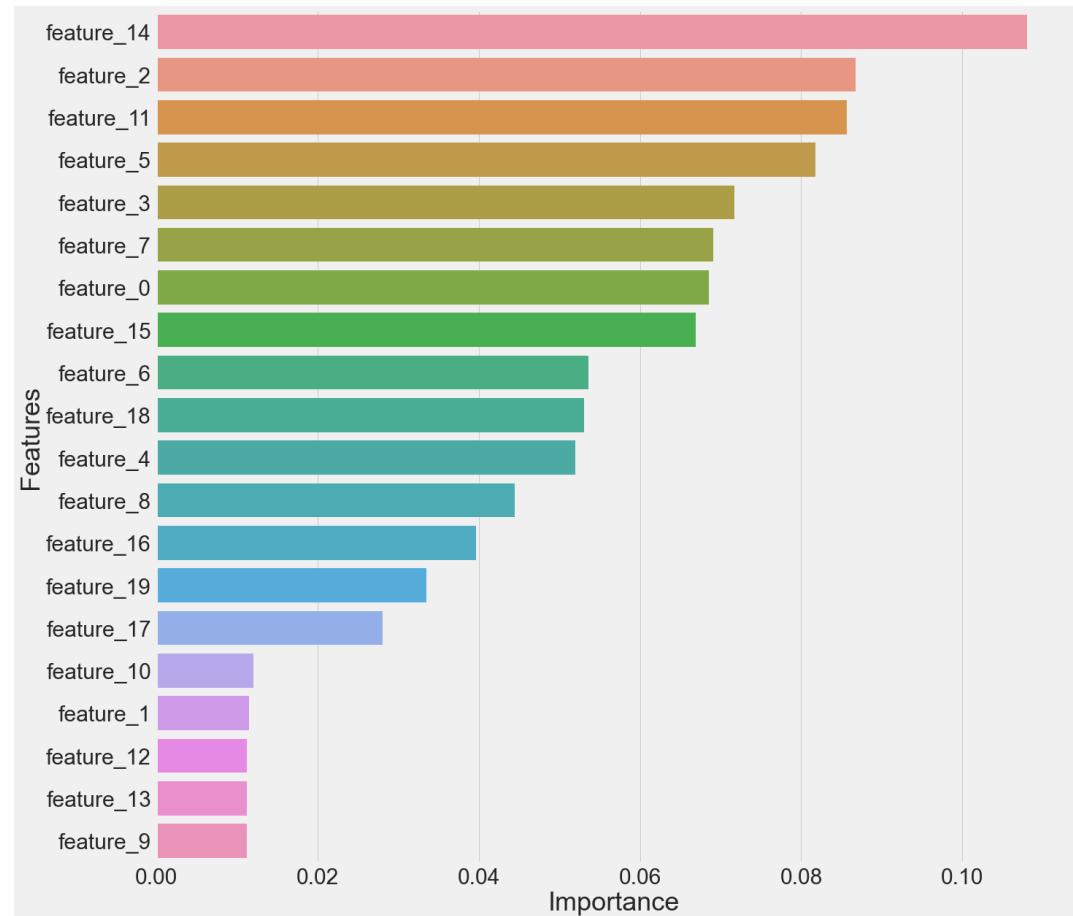
# Интуиция

- ▶ Не все признаки одинаково полезны для решения задачи
- ▶ Некоторые из них более информативны, чем другие
- ▶ Например,  $X_1$  неинформативна для классификации
- ▶ **Цель** – определить **важность каждого признака**



# Методы решения

- ▶ Подсчет корреляций
- ▶ Вероятностные критерии
- ▶ Решающие деревья
- ▶ Линейные модели
- ▶ Общий метод



# Корреляции

Для признака  $f$  вычислим его корреляцию с целевой переменной  $y$ :

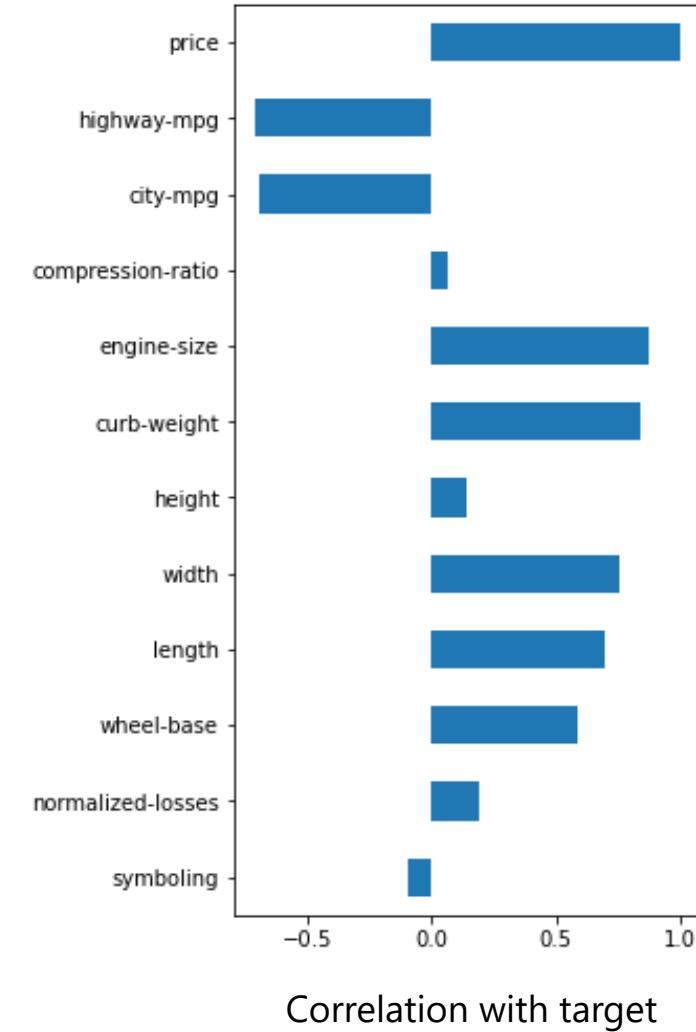
$$\rho(f, y) = \frac{\sum_i (f_i - \bar{f})(y_i - \bar{y})}{\sqrt{\sum_i (f_i - \bar{f})^2 \sum_i (y_i - \bar{y})^2}}$$

$y_i$  - метки в классификации или целевая переменная в регрессии для  $i$ -го объекта

$f_i$  - значение признака для  $i$ -го объекта

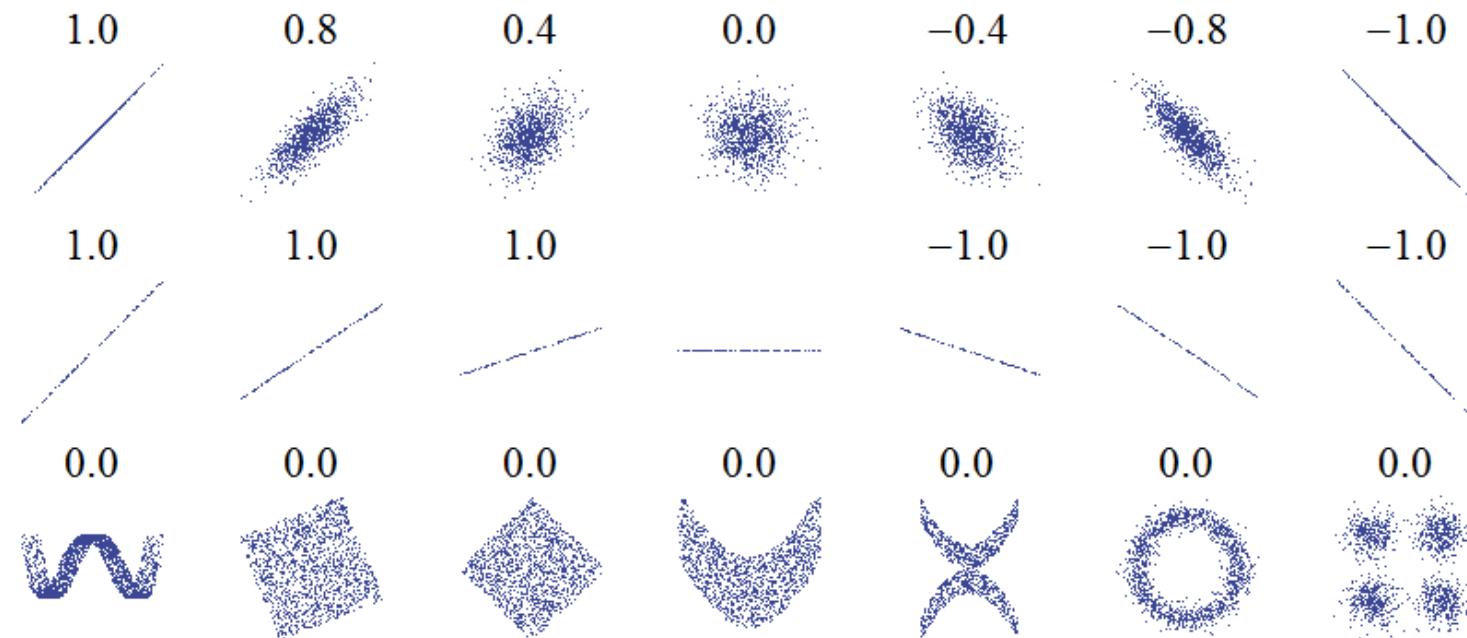
$\rho(f, y)$  - важность признака  $Imp(f)$

# Пример

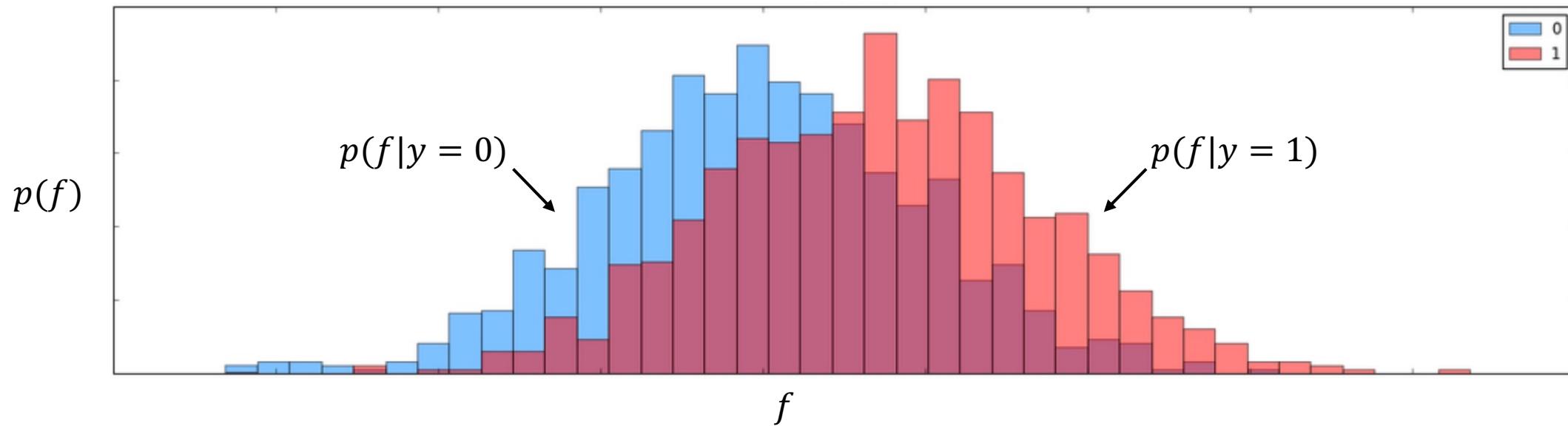


# Корреляция

- ▶ Легко посчитать
- ▶ Но отражает только линейные зависимости



# Вероятностные критерии (расстояния)

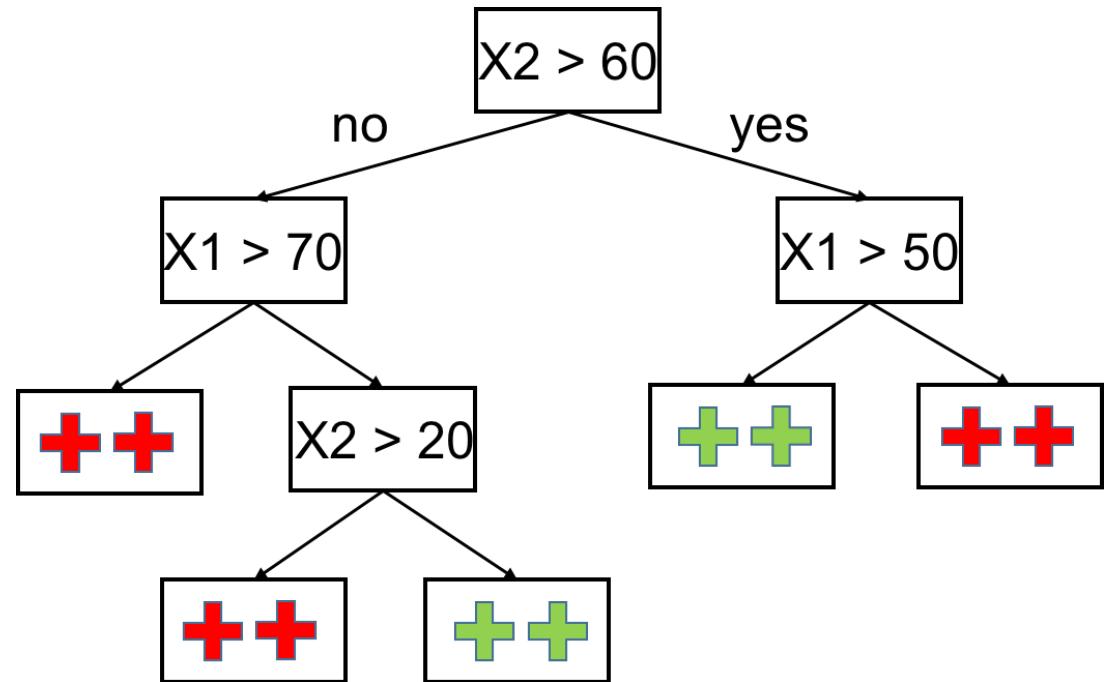


Важность признака  $Imp(f)$  как расстояние полной вариации между двумя распределениями:

$$Imp(f) = \int |p(f|y=1) - p(f|y=0)| df$$

# Решающие деревья

- ▶ Каждая вершина  $t$  имеет двух потомков
- ▶  $n_t$  - число объектов в вершине
- ▶  $I(t)$  – значения критерия информативности (gini, cross-entropy, MSE) для вершины



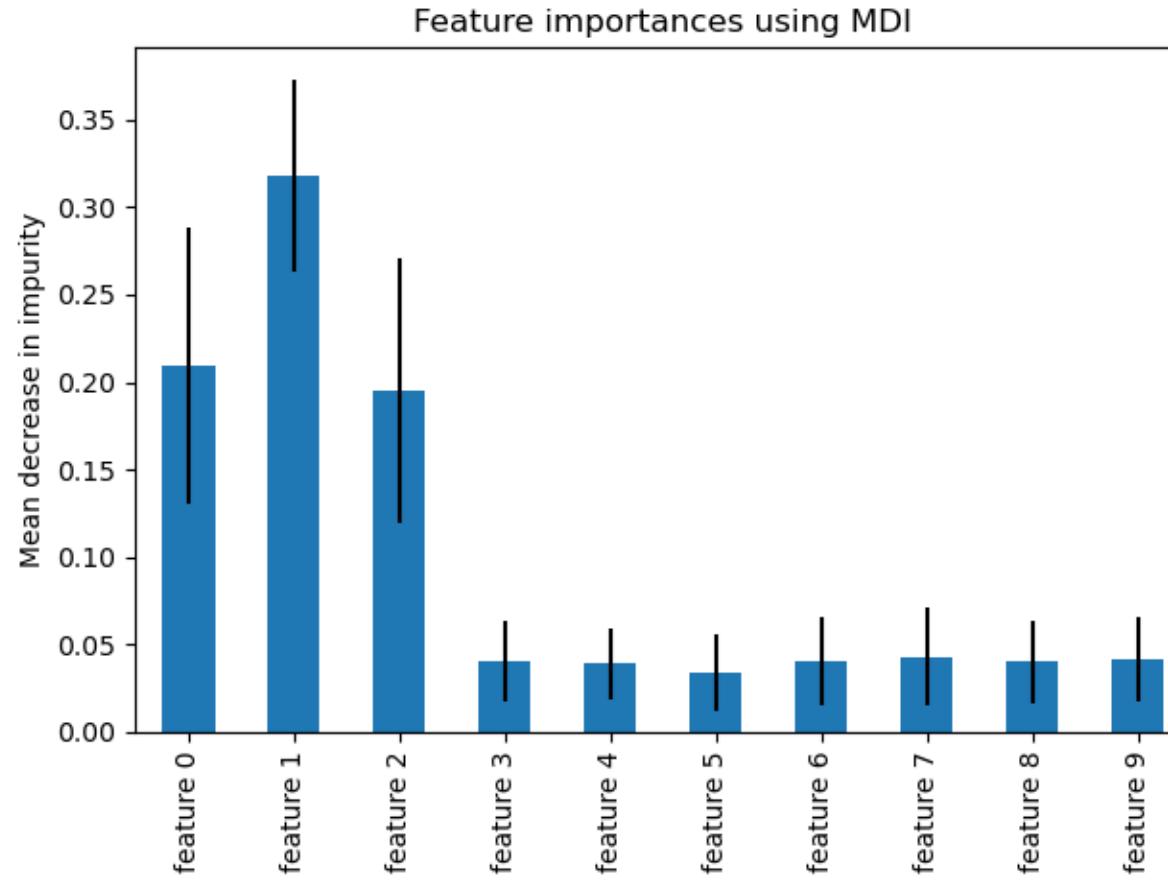
# Решающие деревья

Пусть  $T(f)$  – набор всех вершин дерева, где использовался признак  $f$  в предикате. Тогда важность признака  $f$ :

$$Imp(f) = \sum_{t \in T(f)} n_t \Delta I(t)$$

$$\Delta I(t) = I(t) - \sum_{c \in children(t)} \frac{n_c}{n_t} I(c)$$

# Пример



[https://scikit-learn.org/stable/auto\\_examples/ensemble/plot\\_forest\\_importances.html](https://scikit-learn.org/stable/auto_examples/ensemble/plot_forest_importances.html)

# Линейные модели

Рассмотрим линейную модель с регуляризацией ( $L_1$  или  $L_2$ ):

$$\hat{y} = w_0 + w_1 f_1 + w_2 f_2 + \cdots + w_k f_k$$

Если признаки нормированы (значения одного масштаба), то  
важность признака  $f_i$  равна:

$$Imp(f_i) = |w_i|$$

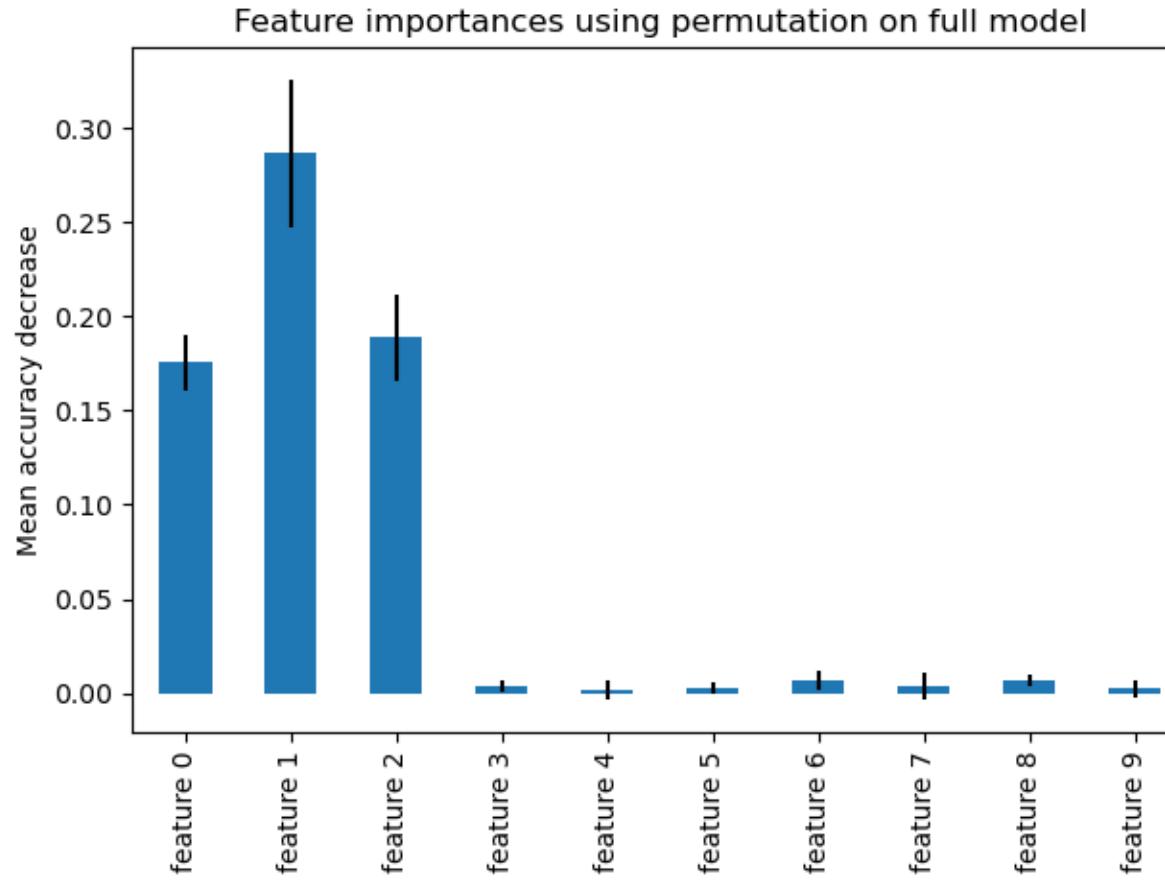
# Общий метод

- ▶ Обучаем модель
- ▶ Считаем метрику качества  $Q_0$  на тестовой выборке
- ▶ Для каждого признака  $f$ :
  - Случайно перемешиваем значения признака
  - Считаем новое значение метрики  $Q_f$  на тестовой выборке
  - Определяем важность признака:

$$Imp(f) = Q_0 - Q_f$$

	RD Spend	Administration	Marketing Spend	Profit	state_California
1	165349.2	136897.8	471784.1	192261.83	0
2	162597.7	151377.59	443898.53	191792.06	1
3	153441.51	101145.55	407934.54	191050.39	1
...	...	...	...	...	...
48	0	135426.92	0	42559.73	1
49	542.05	51743.15	0	35673.41	0
50	0	116983.8	45173.06	14681.4	1

# Пример

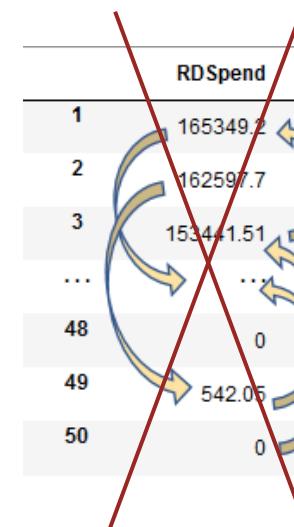


[https://scikit-learn.org/stable/auto\\_examples/ensemble/plot\\_forest\\_importances.html](https://scikit-learn.org/stable/auto_examples/ensemble/plot_forest_importances.html)

# Общий метод (модификация)

- ▶ Обучаем модель **на всем наборе признаков**
- ▶ Считаем метрику качества  $Q_0$  на тестовой выборке
- ▶ Для каждого признака  $f$ :
  - Обучаем модель без этого признака
  - Считаем новое значение метрики  $Q_f$  на тестовой выборке
  - Определяем важность признака:

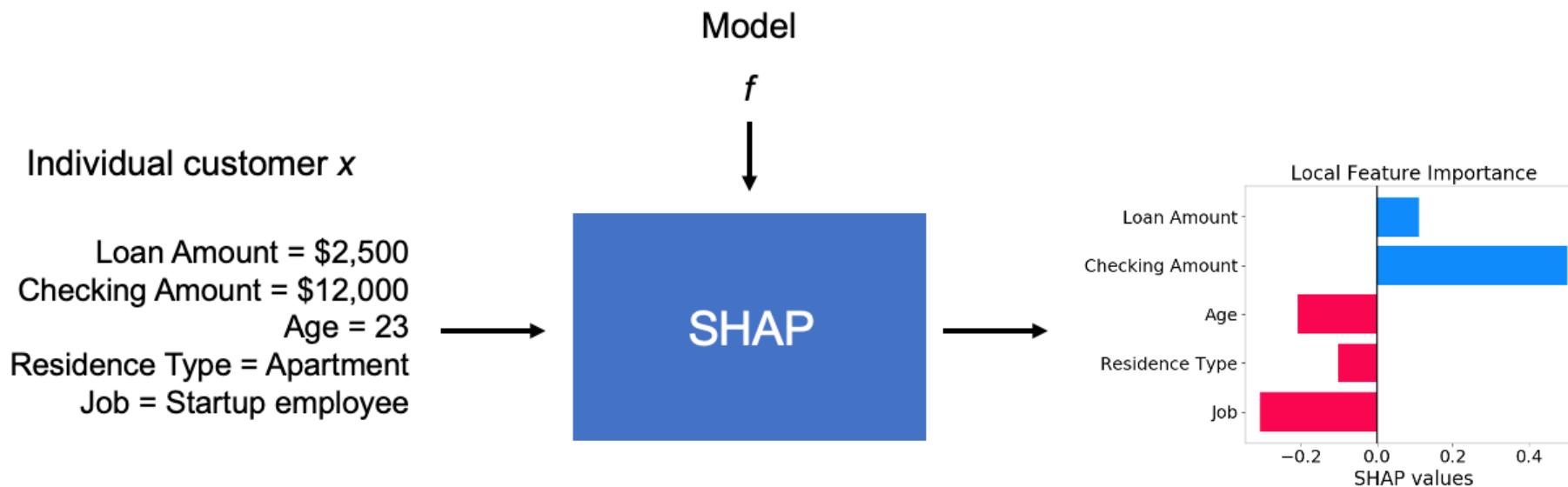
$$Imp(f) = Q_0 - Q_f$$



	RDSpend	Administration	Marketing Spend	Profit	state_California
1	165349.2	136897.8	471784.1	192261.83	0
2	162597.7	151377.59	443898.53	191792.06	1
3	153441.51	101145.55	407934.54	191050.39	1
...	...	...	...	...	...
48	0	135426.92	0	42559.73	1
49	542.05	51743.15	0	35673.41	0
50	0	116983.8	45173.06	14681.4	1

# SHAP

- ▶ Существует множество других методов
- ▶ Один из наиболее интересный методов основан на векторах Шепли (Shapley values): <https://github.com/slundberg/shap>



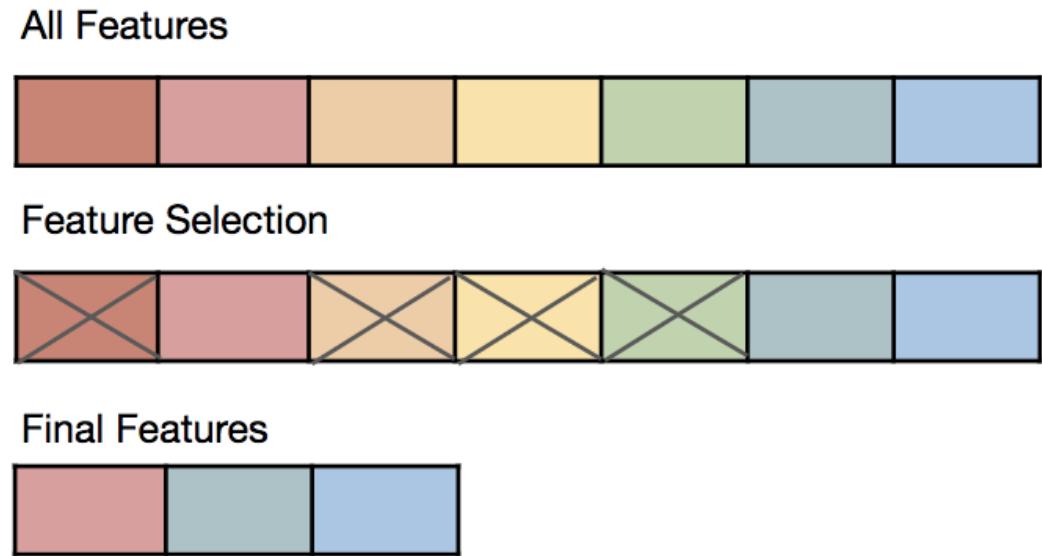
# Отбор признаков (Feature Selection)

# Отбор признаков

**Цель** – уменьшить число признаков с минимальными потерями качества модели.

Примеры:

- ▶ Оставить  $K$  из  $D$  наилучших признаков
- ▶ Удалить как можно больше признаков так, чтобы качество модели  $Q \geq Q_{min}$



# Методы

- ▶ Метод фильтрации
- ▶ Встроенные методы
- ▶ Рекурсивное удаление признаков (recursive feature elimination)

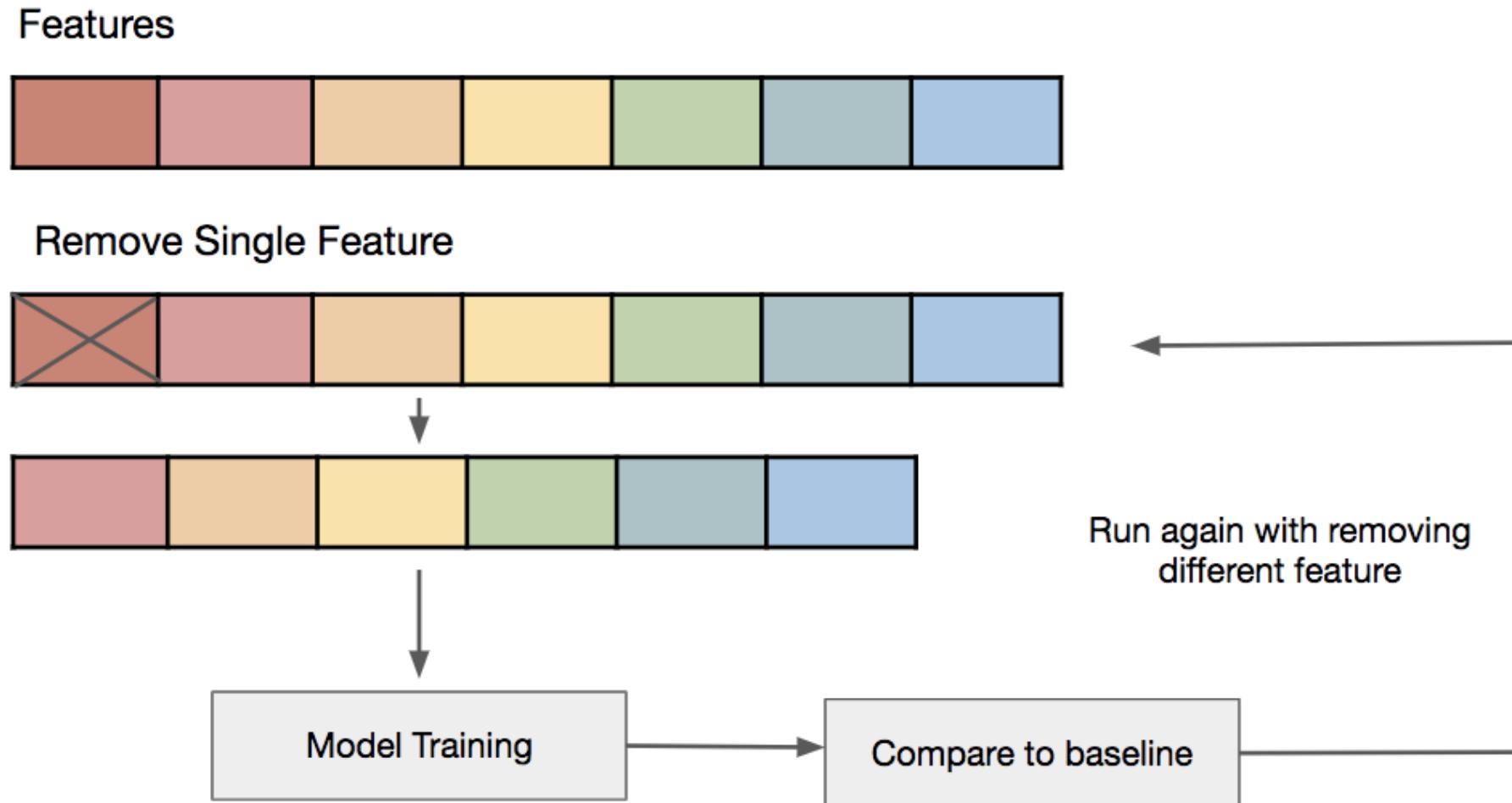
# Метод фильтрации

- ▶ Определяем важность отдельных признаков:  
 $Imp(f_1), Imp(f_2), \dots, Imp(f_D)$
  - ▶ Выбираем нужное число признаков с наибольшими значениями важности
- 
- ▶ Простой и быстрый метод
  - ▶ Плохо работает для скоррелированных признаков

# Встроенные методы

- ▶ Определяем важность отдельных признаков с помощью решающих деревьев или линейных моделей:  
 $Imp(f_1), Imp(f_2), \dots, Imp(f_D)$
- ▶ Выбираем нужное число признаков с наибольшими значениями важности
- ▶ Популярный метод
- ▶ Учитывает корреляции между признаками

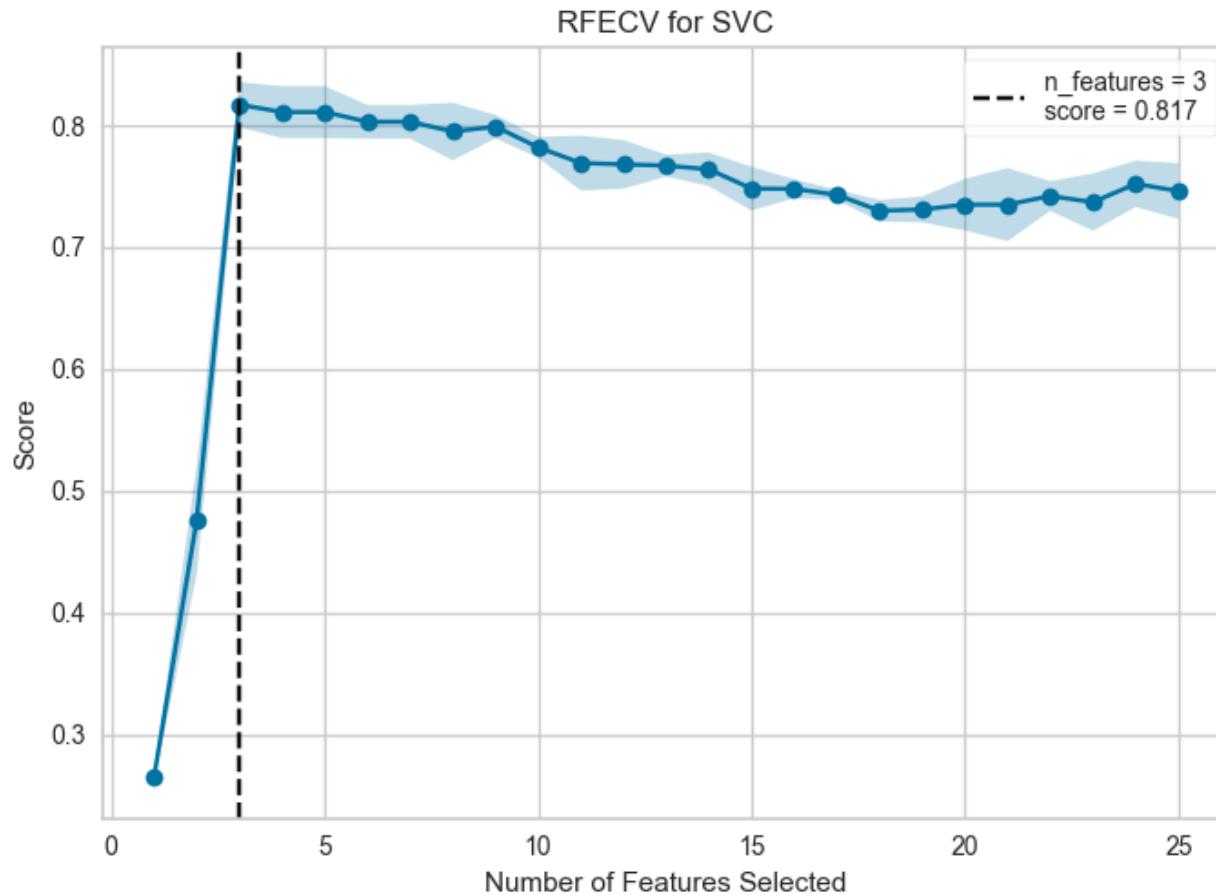
# Рекурсивное удаление признаков



# Рекурсивное удаление признаков

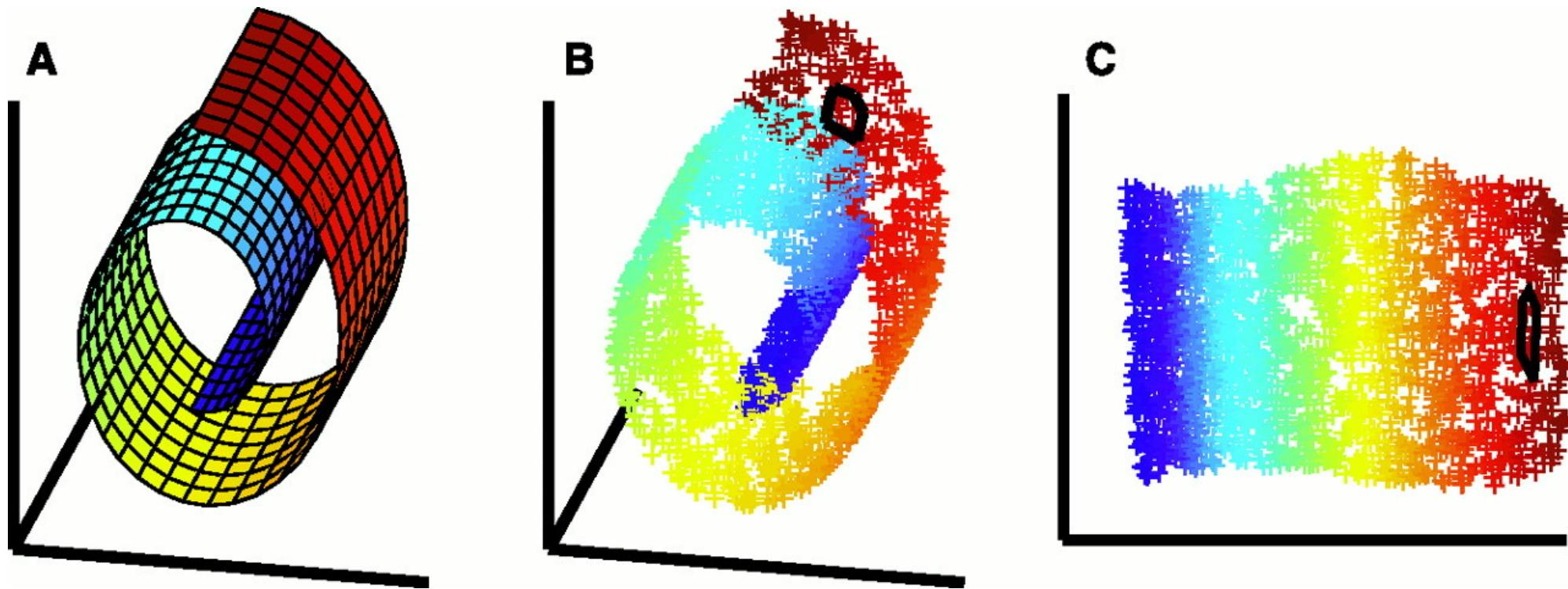
- ▶ Обучаем модель на всем наборе признаков
- ▶ Определите важность признаков использую эту модель
- ▶ Удаляем наименее важный признак
- ▶ Повторяем процедуру

# Пример



# Уменьшение размерности (Dimensionality reduction)

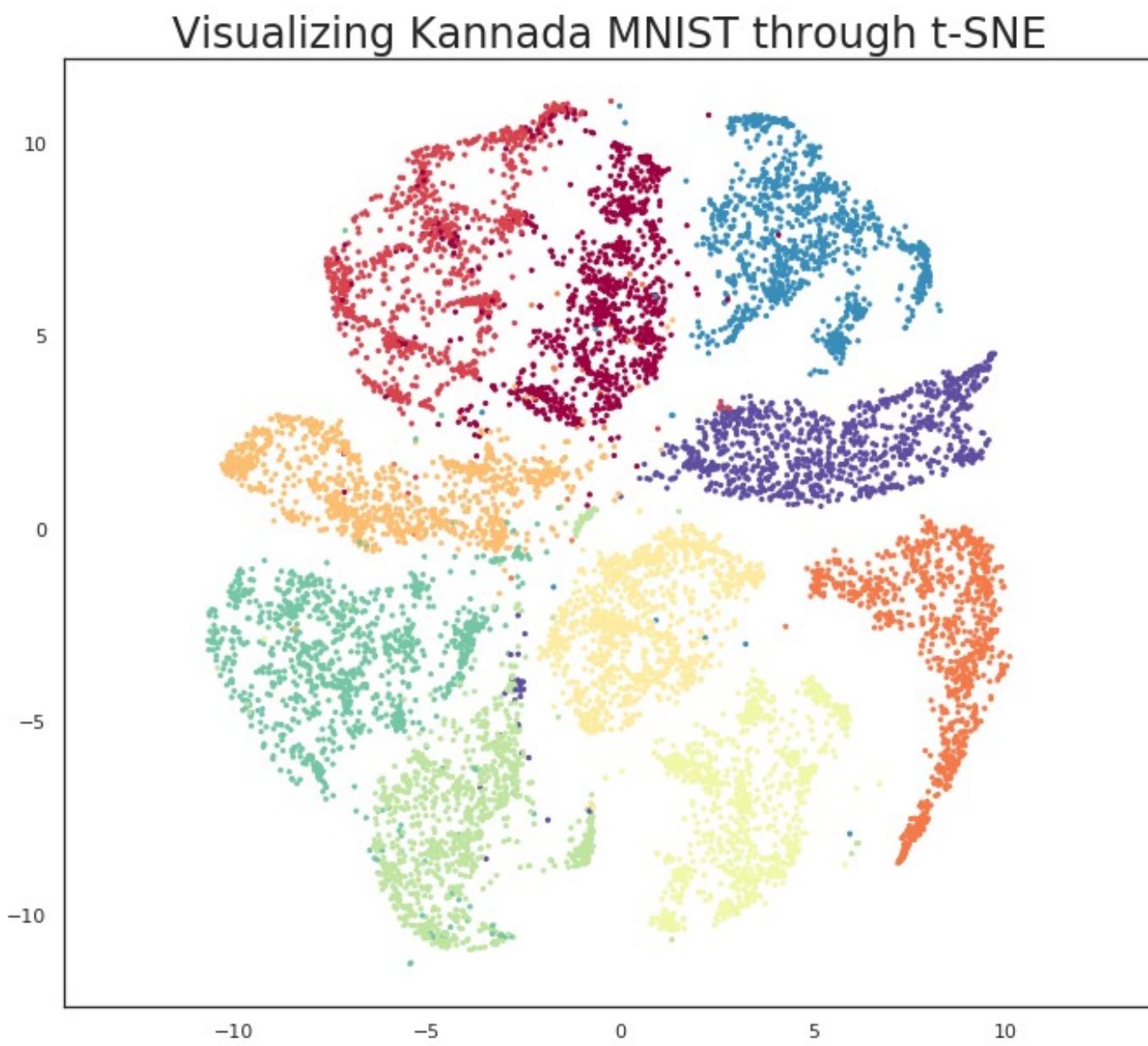
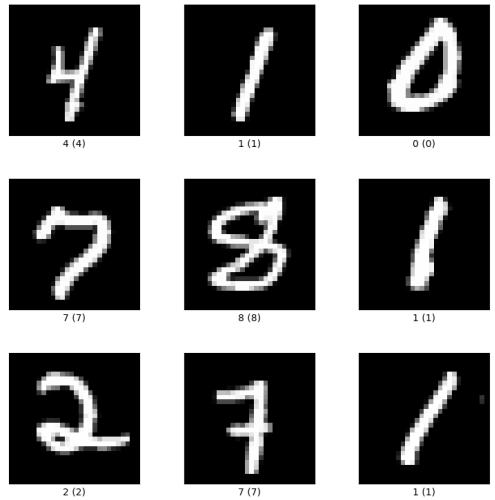
# Постановка задачи



- ▶ Уменьшить размерность пространства входных признаков
- ▶ Минимизировать потери полезной информации
- ▶ Новые признаки – преобразование старых признаков

# Пример

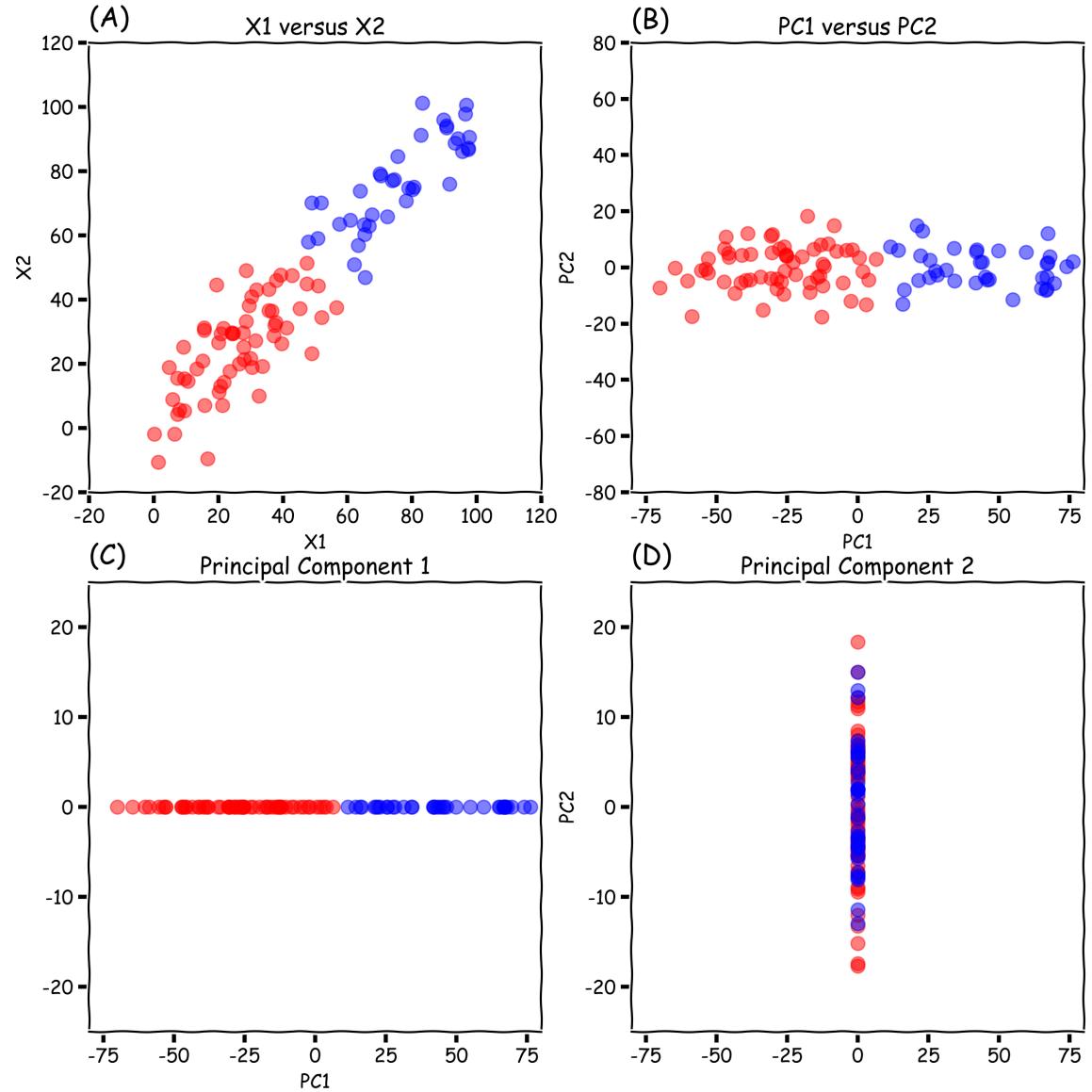
Уменьшение размерности:  
28x28 → 2



# Метод главных компонент (Principal Component Analysis) (PCA)

# Интуиция

- ▶ Даны 2 класса в 2D пространстве ( $X_1$ ,  $X_2$ )
- ▶ Данные расположены вдоль диагонали
- ▶ Повернем системы координат так, чтобы данные располагались вдоль одной из осей
- ▶ Получили новые признаки ( $PC_1$ ,  $PC_2$ )
- ▶ Классы разделяются по  $PC_1$

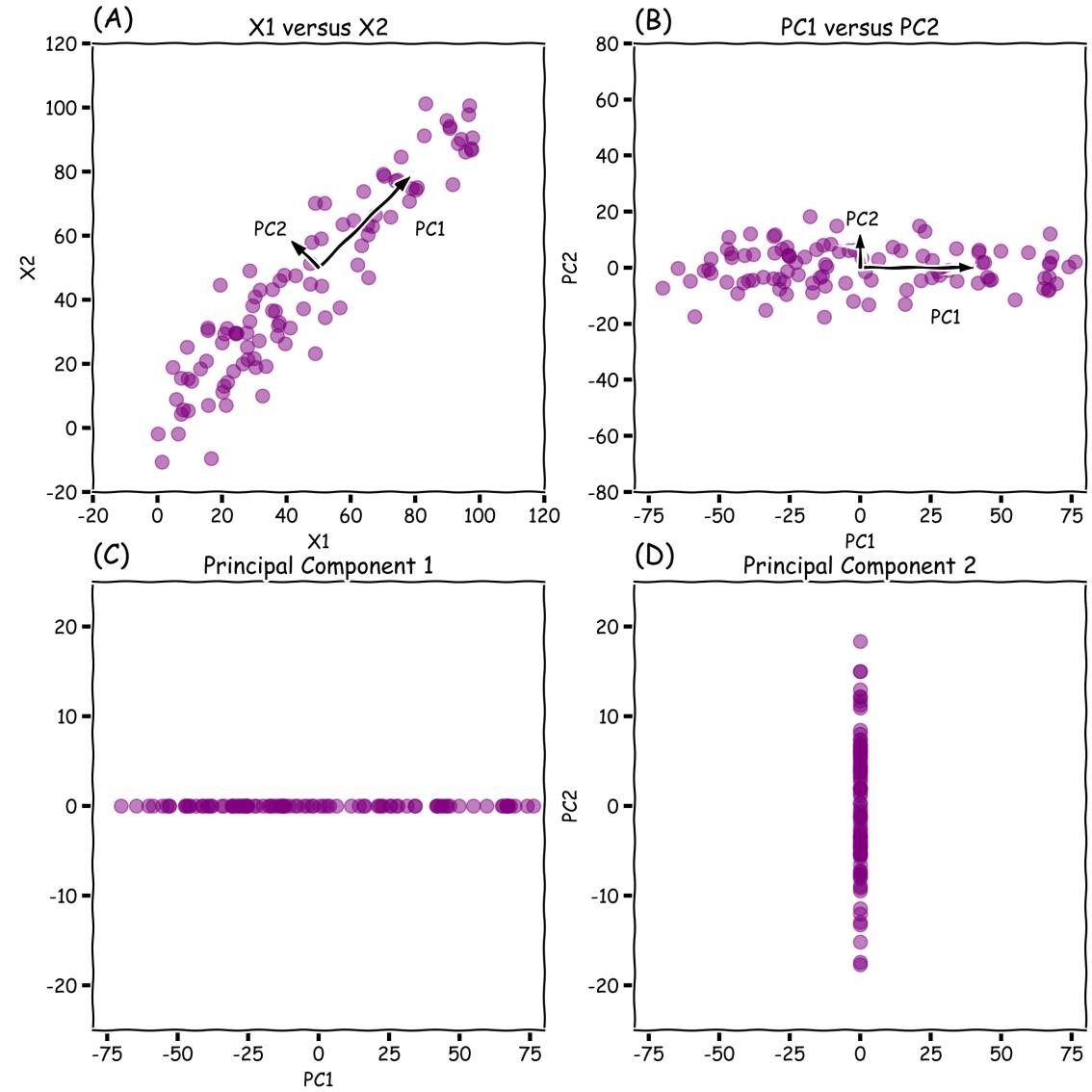


# Задача

- ▶ Даны наблюдения с признаками ( $X_1, X_2, X_3, \dots$ )
- ▶ Нужно найти главные компоненты ( $PC_1, PC_2, PC_3, \dots$ )
- ▶ Мы будем обозначать главные компоненты как  $a_1, a_2, a_3, \dots$
- ▶ При этом, будем требовать их ортогональность:

$$a_i^T a_i = 1$$

$$a_i^T a_j = 0, \quad i \neq j$$



# Поиск первой компоненты

- ▶ Пусть признаки в матрице наблюдений  $X$  нормированы (применили Standard Scaler)
- ▶ Пусть дан вектор  $a$
- ▶ Тогда проекция вектора  $x_i$  на  $a$  задается как  $z_i = a^T x_i$
- ▶ Посчитаем дисперсию  $z_i$ :

$$\sigma_a^2 = \frac{1}{n} \sum_{i=1}^n (z_i - \mu_z)^2$$

- ▶ Будем искать такой вектор  $a$ , чтобы

$$\sigma_a^2 \rightarrow \max_a$$

# Поиск первой компоненты

- ▶ Распишем дисперсию  $z_i$ :

$$\sigma_a^2 = \frac{1}{n} \sum_{i=1}^n (z_i - \mu_z)^2 = \frac{1}{n} \sum_{i=1}^n (a^T x_i)^2 =$$

$$= \frac{1}{n} \sum_{i=1}^n a^T (x_i x_i^T) a = a^T \left( \frac{1}{n} \sum_{i=1}^n x_i x_i^T \right) a =$$

$$= a^T X^T X a = a^T C a \rightarrow \max_a$$

# Поиск первой компоненты

- ▶ Тогда первую главную компоненту  $a_1$  находим так:

$$\begin{cases} a_1 X^T X a_1 \rightarrow \max_{a_1} \\ a_1^T a_1 = 1 \end{cases}$$

# Поиск первой компоненты

- ▶ Тогда первую главную компоненту  $a_1$  находим так:

$$\begin{cases} a_1 X^T X a_1 \rightarrow \max_{a_1} \\ a_1^T a_1 = 1 \end{cases}$$

- ▶ Решаем максимизацией лагранжиана:

$$L = a_1 X^T X a_1 - \nu(a_1^T a_1 - 1) \rightarrow \max_{\nu, a_1}$$

$$\frac{\partial L}{\partial a_1} = 2X^T X a_1 - 2\nu a_1 = 0$$

$$X^T X a_1 = \nu a_1$$

# Поиск первой компоненты

- ▶ Видим, что вектор первой главной компоненты – собственный вектор матрицы  $X^T X$
- ▶ Но какой собственный вектор?
- ▶ Уже знаем, что

$$a_1 X^T X a_1 \rightarrow \max_{a_1}$$

- И знаем, что

$$X^T X a_1 = \nu a_1$$

- Тогда подставим в первое выражение:

$$\nu a_1^T a_1 = \nu \rightarrow \max$$

- ▶ Получили, что  $a_1$  - собственный вектор с максимальным собственным числом.

# Поиск второй компоненты

- ▶ Вторую главную компоненту  $a_2$  находим так:

$$\begin{cases} a_2 X^T X a_2 \rightarrow \max_{a_2} \\ a_2^T a_2 = 1 \\ a_2^T a_1 = 0 \end{cases}$$

# Алгоритм РСА

- ▶ Нормируем матрицу наблюдений  $X$
- ▶ Находим матрицу  $C = X^T X$
- ▶ Находим первые  $k$  собственных векторов  $a_i$  и собственных значений  $\lambda_i$ :

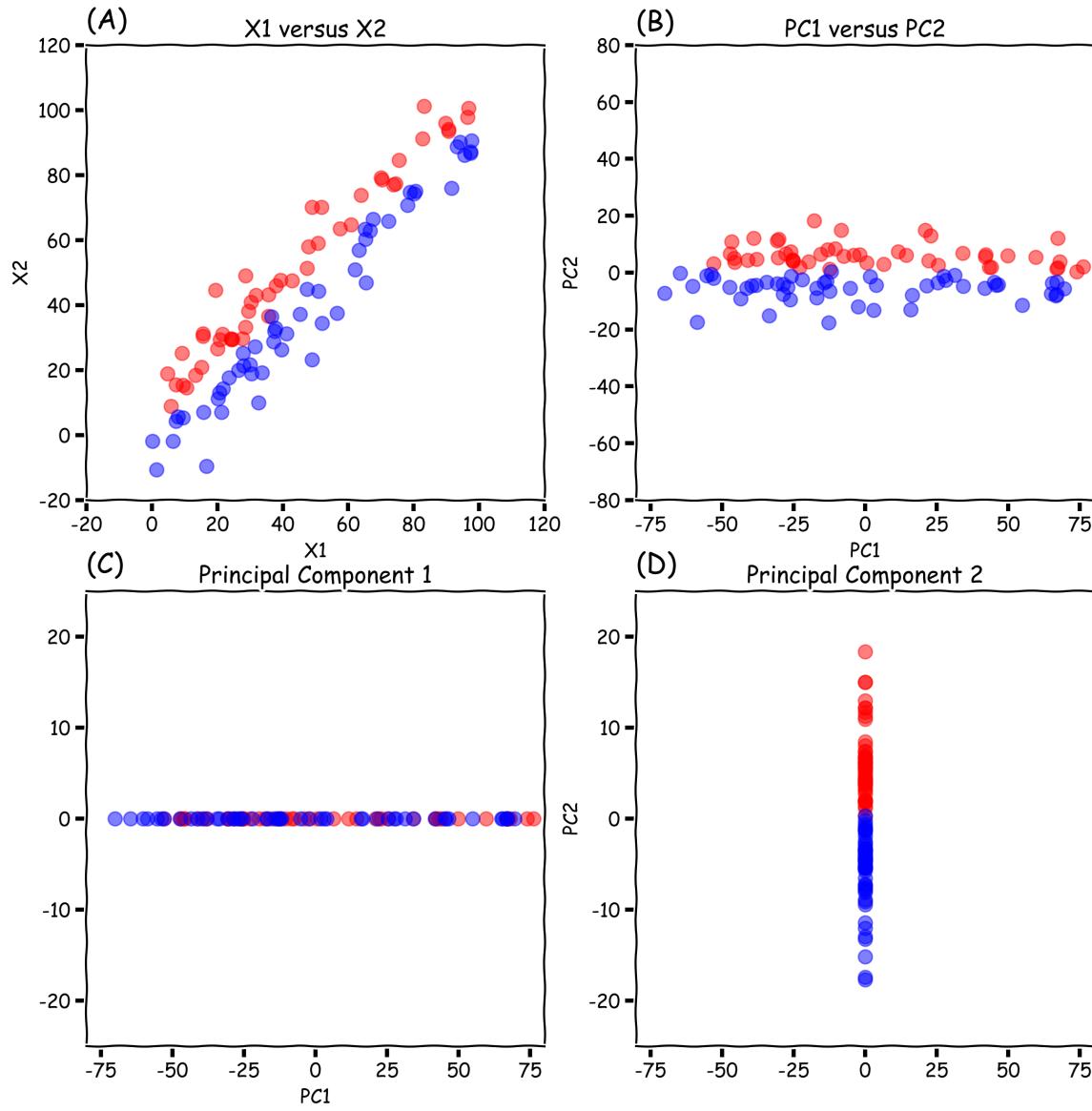
$$A = \begin{bmatrix} | & | & & | \\ a_1 & a_2 & \dots & a_k \\ | & | & & | \end{bmatrix}$$

- ▶ Делаем проекцию на собственные вектора:

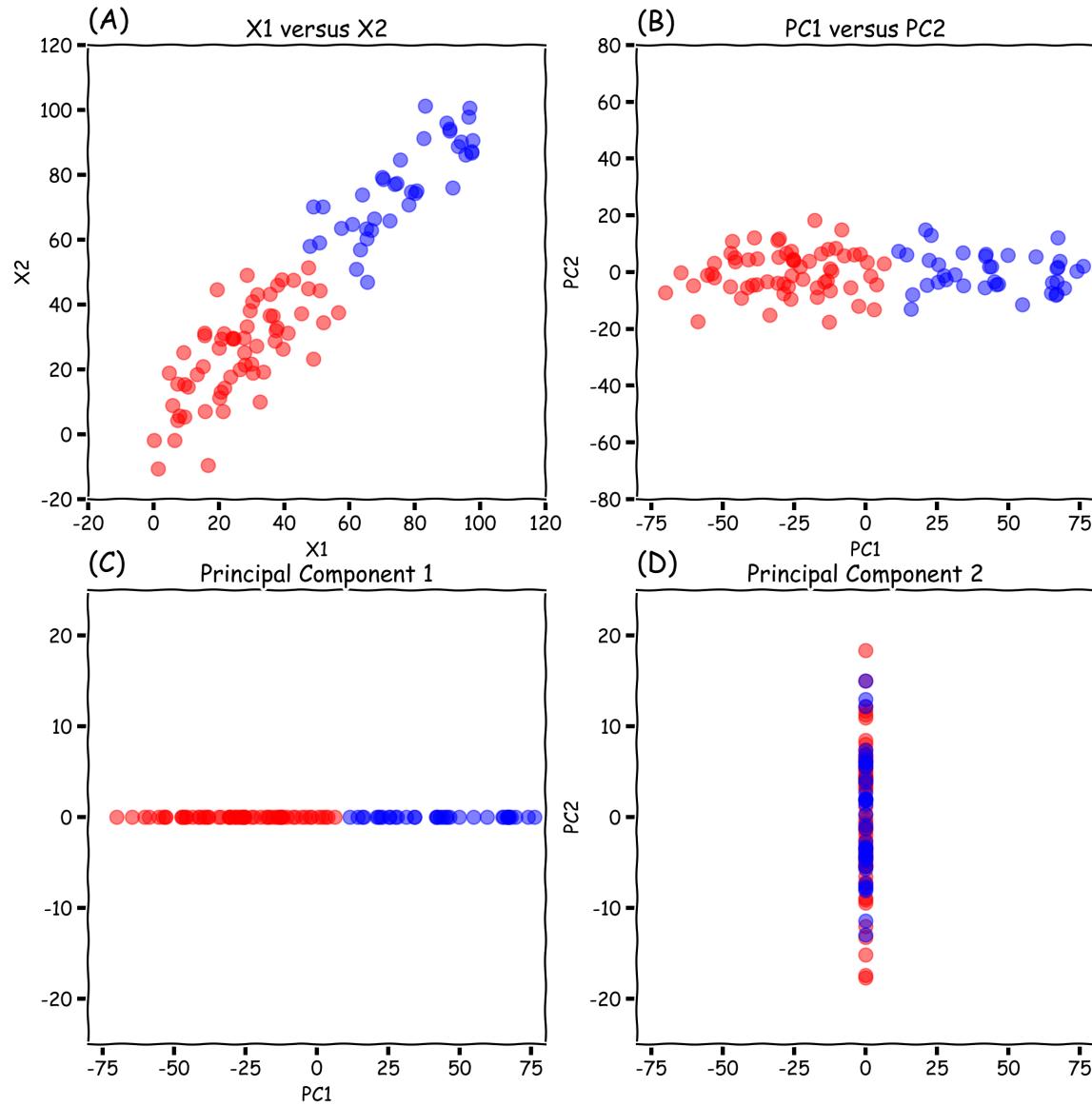
$$Z = XA$$

- ▶ Матрица  $Z$  – новая матрица наблюдений

# Пример



# Пример



# Объясненная дисперсия (Explained variance)

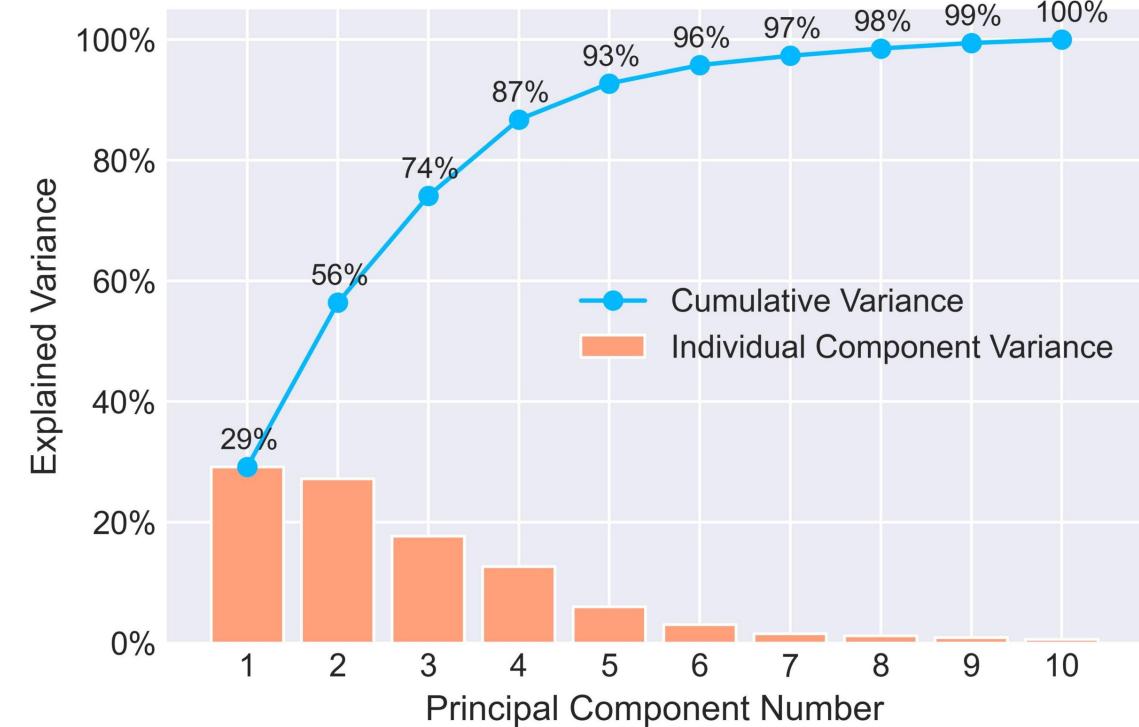
- ▶ Зачем нужны собственные значения  $\lambda_i$ ?
- ▶ Они позволяют оценить дисперсию наблюдений вдоль собственных векторов
- ▶ Чем меньше дисперсия, тем больше наблюдения похожи друг на друга  $\rightarrow$  содержат меньше информации
- ▶ Собственные значения позволяют оценить потери информации, если мы выбросим какие-то собственные вектора

# Объясненная дисперсия (Explained variance)

- ▶ Посчитаем объясненную дисперсию для собственного вектора  $a_i$ :

$$e_i = \frac{\lambda_i}{\sum_{d=1}^D \lambda_d}$$

- где  $D$  – число собственных векторов матрицы  $C$

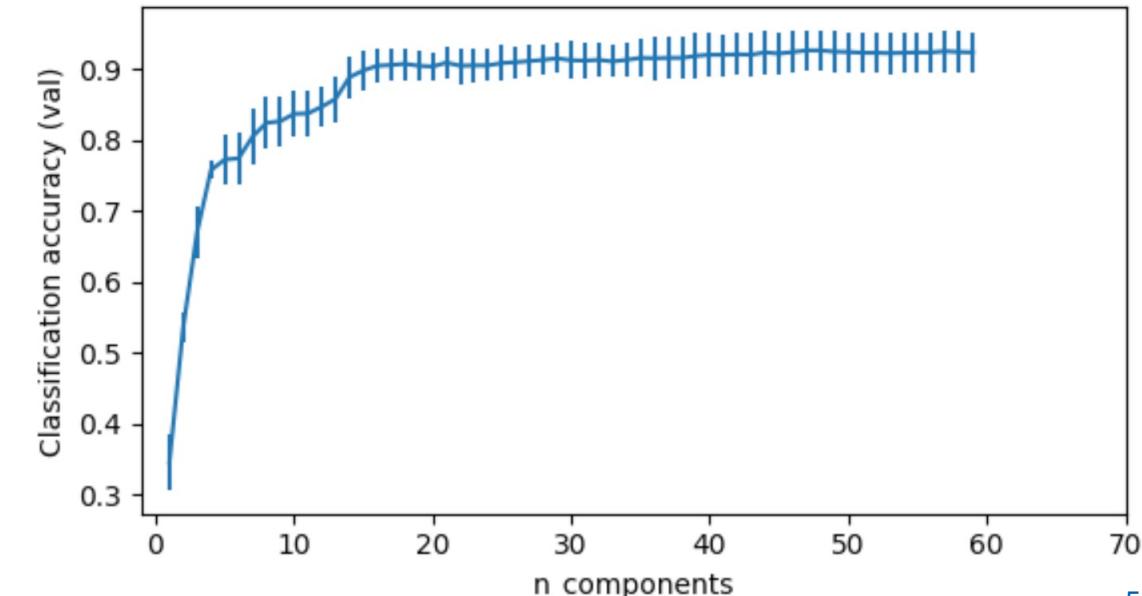
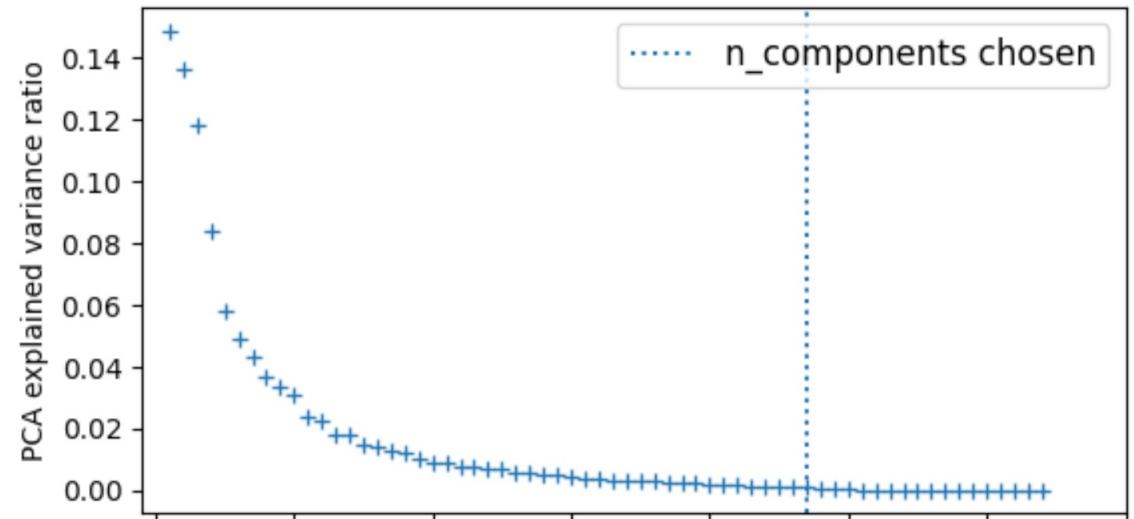


# Пример

Зависимость объясненной дисперсии и качества классификации от числа главных компонент PCA

- ▶ После 20 компонент качество классификации почти перестает расти
- ▶ Оптимальное число компонент - 47

Best parameter (CV score=0.927):  
{'logistic\_\_C': 0.046415888336127774, 'pca\_\_n\_components': 47}



# Другие методы уменьшения размерности

## Dimensionality reduction methods

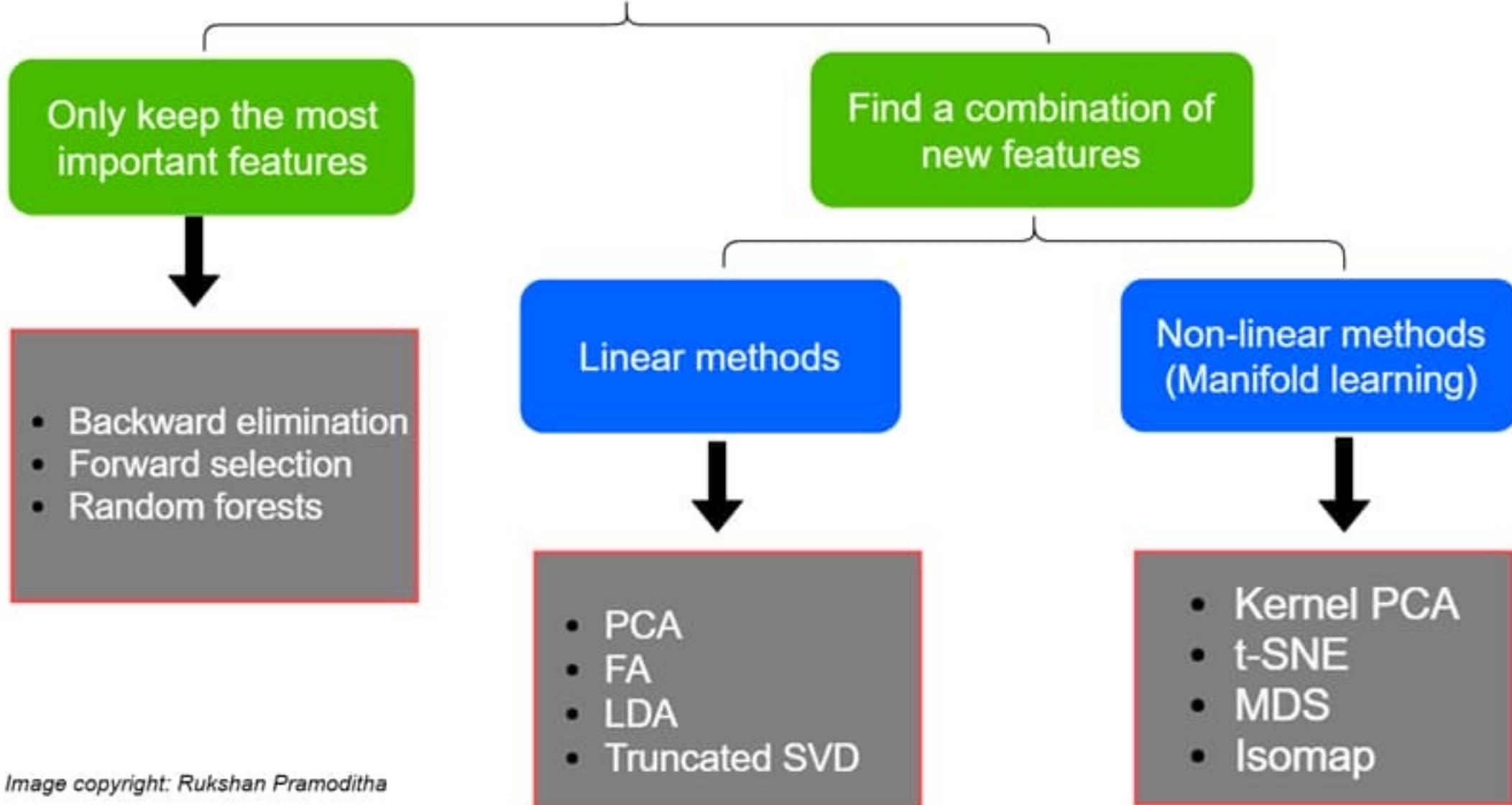
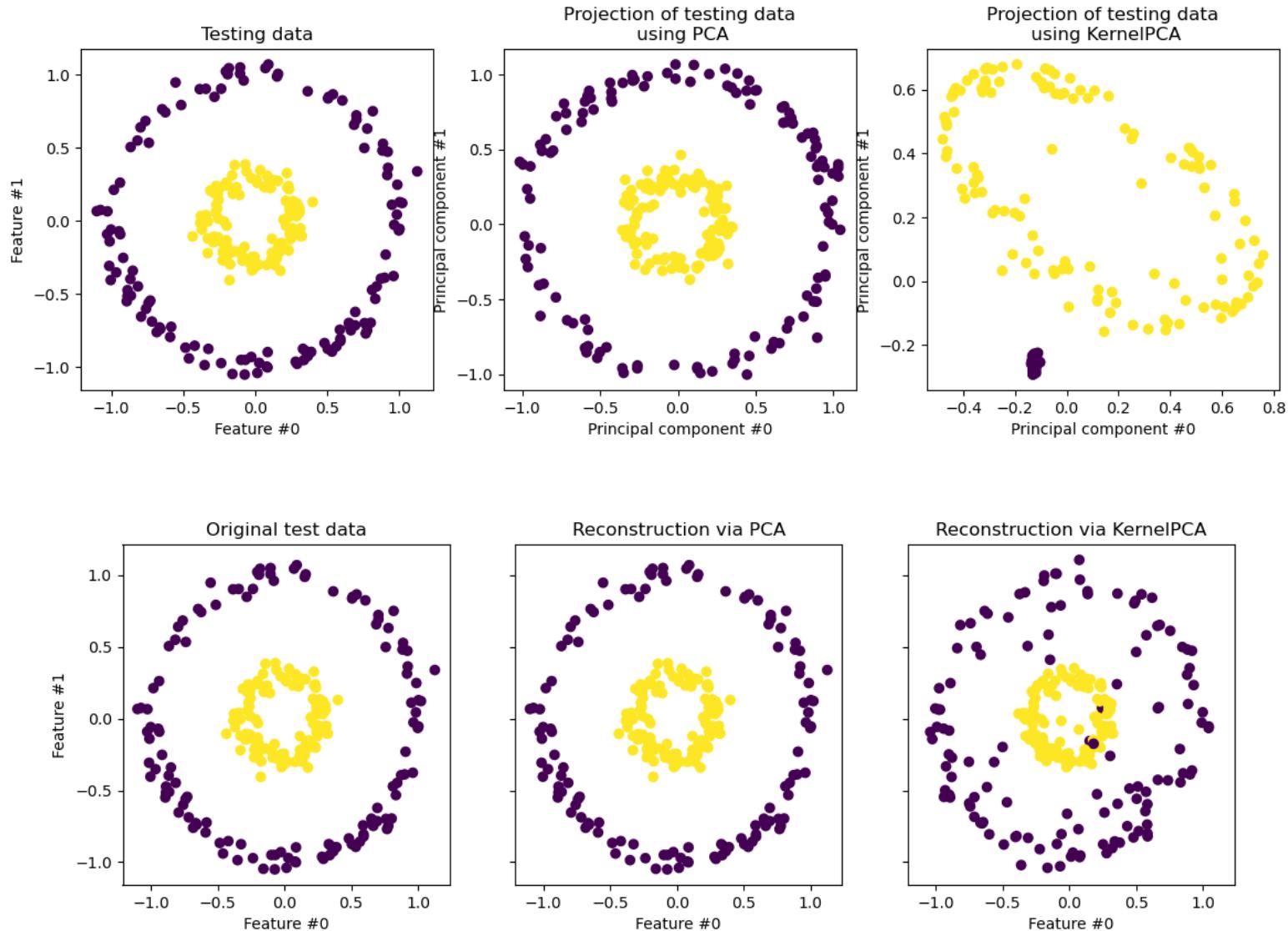


Image copyright: Rukshan Pramoditha

# Методы уменьшения размерности

- ▶ PCA (<https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html>)
- ▶ Kernel PCA (<https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.KernelPCA.html>)
- ▶ t-SNE (<https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html>)
- ▶ Isomap (<https://scikit-learn.org/stable/modules/generated/sklearn.manifold.Isomap.html>)

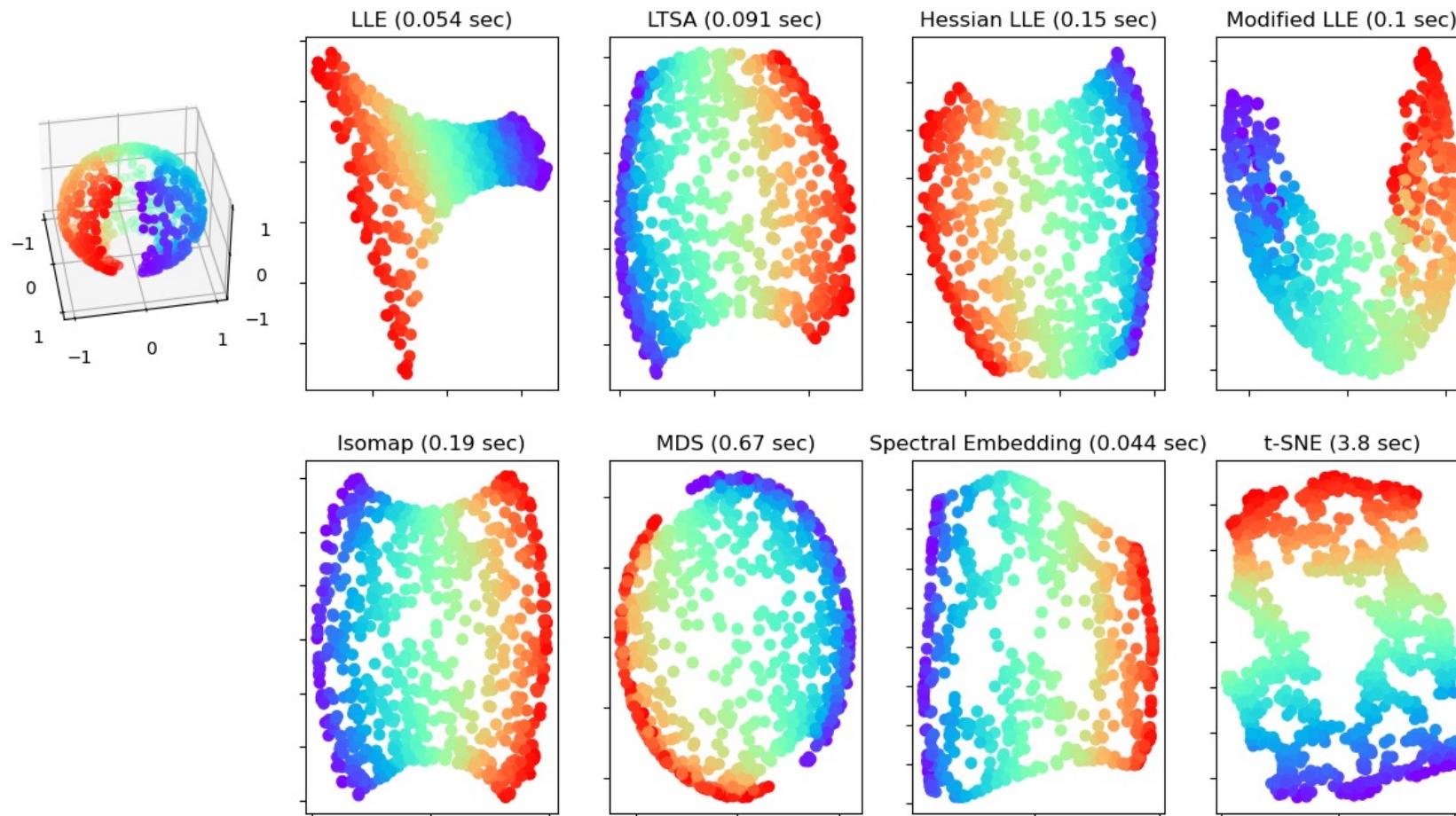
# Пример



Источник: [https://scikit-learn.org/stable/auto\\_examples/decomposition/plot\\_kernel\\_pca.html#sphx-glr-auto-examples-decomposition-plot-kernel-pca-py](https://scikit-learn.org/stable/auto_examples/decomposition/plot_kernel_pca.html#sphx-glr-auto-examples-decomposition-plot-kernel-pca-py)

# Пример

Manifold Learning with 1000 points, 10 neighbors

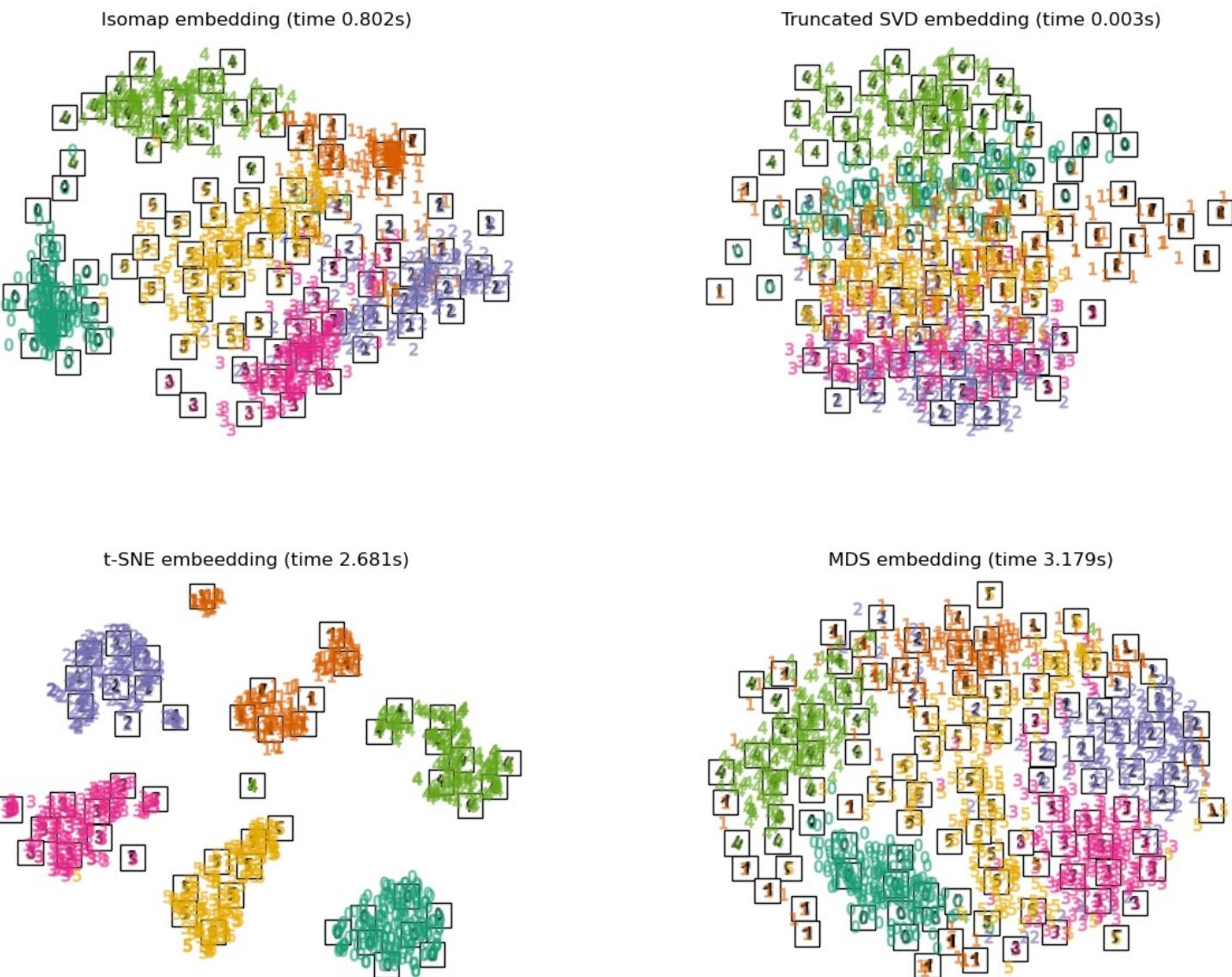


Источник: [https://scikit-learn.org/stable/auto\\_examples/manifold/plot\\_manifold\\_sphere.html#sphx-glr-auto-examples-manifold-plot-manifold-sphere-py](https://scikit-learn.org/stable/auto_examples/manifold/plot_manifold_sphere.html#sphx-glr-auto-examples-manifold-plot-manifold-sphere-py)

# Пример

A selection from the 64-dimensional digits dataset

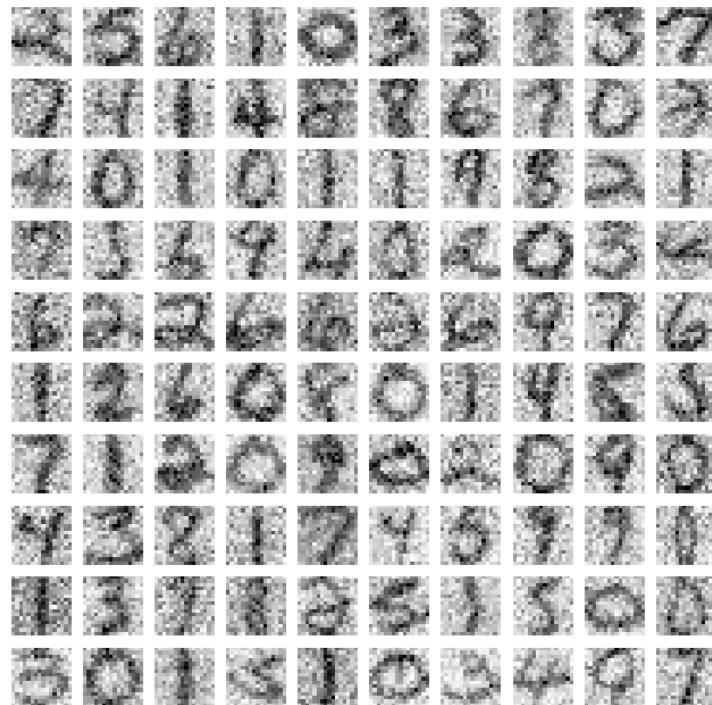
0	1	2	3	4	5	0	4	1	3
4	5	0	1	2	3	4	5	0	5
5	5	0	4	1	3	5	1	0	0
2	2	2	0	1	2	3	3	3	3
4	4	1	5	0	5	2	4	0	0
1	3	2	1	4	3	4	3	1	4
3	1	4	0	5	7	1	5	4	4
2	2	2	5	5	4	4	0	0	1
2	3	4	5	0	1	2	3	4	5
0	1	2	3	4	5	0	5	5	5



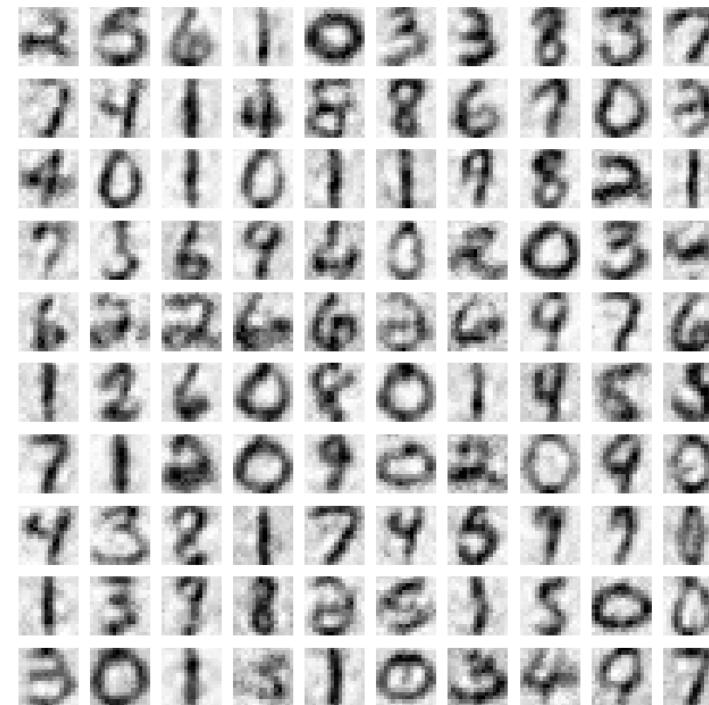
Источник: [https://scikit-learn.org/stable/auto\\_examples/manifold/plot\\_lle\\_digits.html#sphx-glr-auto-examples-manifold-plot-lle-digits-py](https://scikit-learn.org/stable/auto_examples/manifold/plot_lle_digits.html#sphx-glr-auto-examples-manifold-plot-lle-digits-py)

# Денойзинг с использованием PCA

Noisy test images  
MSE: 0.06



PCA reconstruction  
MSE: 0.01



Uncorrupted test images

