

JeML

Модели и как их **правильно** оценивать

Варвус Артем

2024-05-06

Финансовый Университет при Правительстве РФ



1. Модель
2. Как обучать модели. Градиентный спуск
3. Ошибки

Полезные сервисы!

Полезные сервисы!



Вопросы с собеседований

Полезные сервисы!



Вопросы с собеседований

Научные работы с кодом

Полезные сервисы!



Вопросы с собеседований

Научные работы с кодом

Интересная игра



Made with  in QuickQR.Art

Модель

Вспомним что такое модель



Пусть задана выборка: (\mathbb{X}, \mathbb{Y}) , где \mathbb{X} - множество признаков, \mathbb{Y} - “ответы” к ним

Вспомним что такое модель



Пусть задана выборка: (\mathbb{X}, \mathbb{Y}) , где \mathbb{X} - множество признаков, \mathbb{Y} - “ответы” к ним

Пусть задана: $g : \mathbb{X} \times \Theta \rightarrow \mathbb{Y}$, где Θ - веса модели, которые подбираются при **обучении**



Пусть задана выборка: (\mathbb{X}, \mathbb{Y}) , где \mathbb{X} - множество признаков, \mathbb{Y} - “ответы” к ним

Пусть задана: $g : \mathbb{X} \times \Theta \rightarrow \mathbb{Y}$, где Θ - веса модели, которые подбираются при **обучении**

То модель: $A = \{g(x, \theta) | \theta \in \Theta\}$



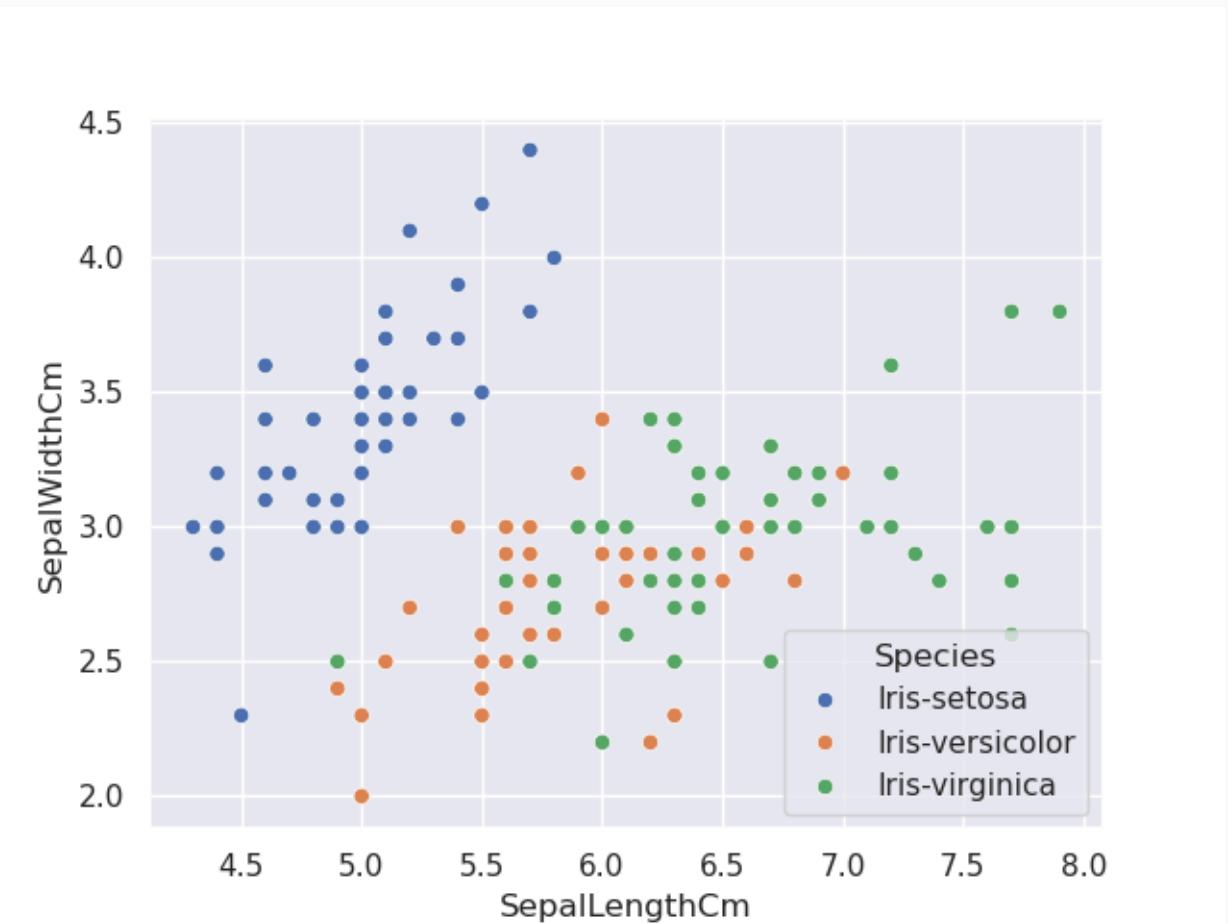
Пусть задана выборка: (\mathbb{X}, \mathbb{Y}) , где \mathbb{X} - множество признаков, \mathbb{Y} - “ответы” к ним

Пусть задана: $g : \mathbb{X} \times \Theta \rightarrow \mathbb{Y}$, где Θ - веса модели, которые подбираются при **обучении**

То модель: $A = \{g(x, \theta) | \theta \in \Theta\}$

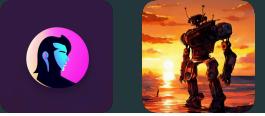
А как подобрать такую g ?

Примерчик





Логистическая регрессия - линейная регрессия + функция на конце



Логистическая регрессия - линейная регрессия + функция на конце

$$z = \left(\sum_{i=1}^n \omega_i x_i \right) + b, \omega_i - \text{веса}, b - \text{смещение}$$

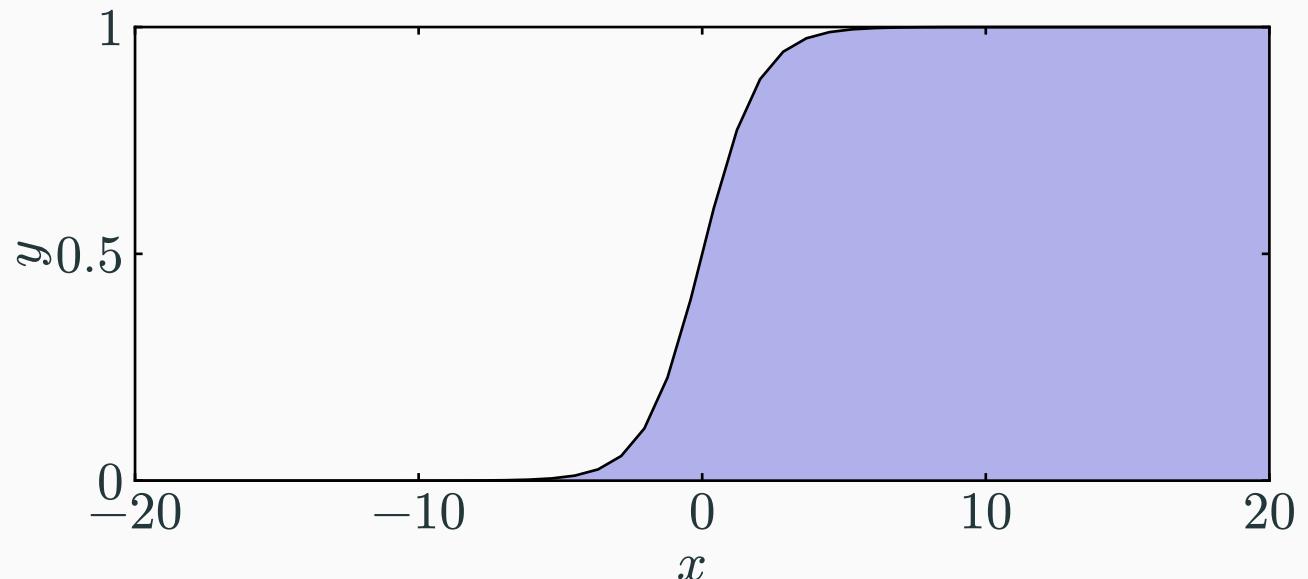


Логистическая регрессия - линейная регрессия + функция на конце

$$z = \left(\sum_{i=1}^n \omega_i x_i \right) + b, \omega_i - \text{веса}, b - \text{смещение}$$

$$g(z) = \frac{1}{1 + e^{-z}} = \frac{1}{1 + \exp(-z)}$$

Почему такая странная функция?

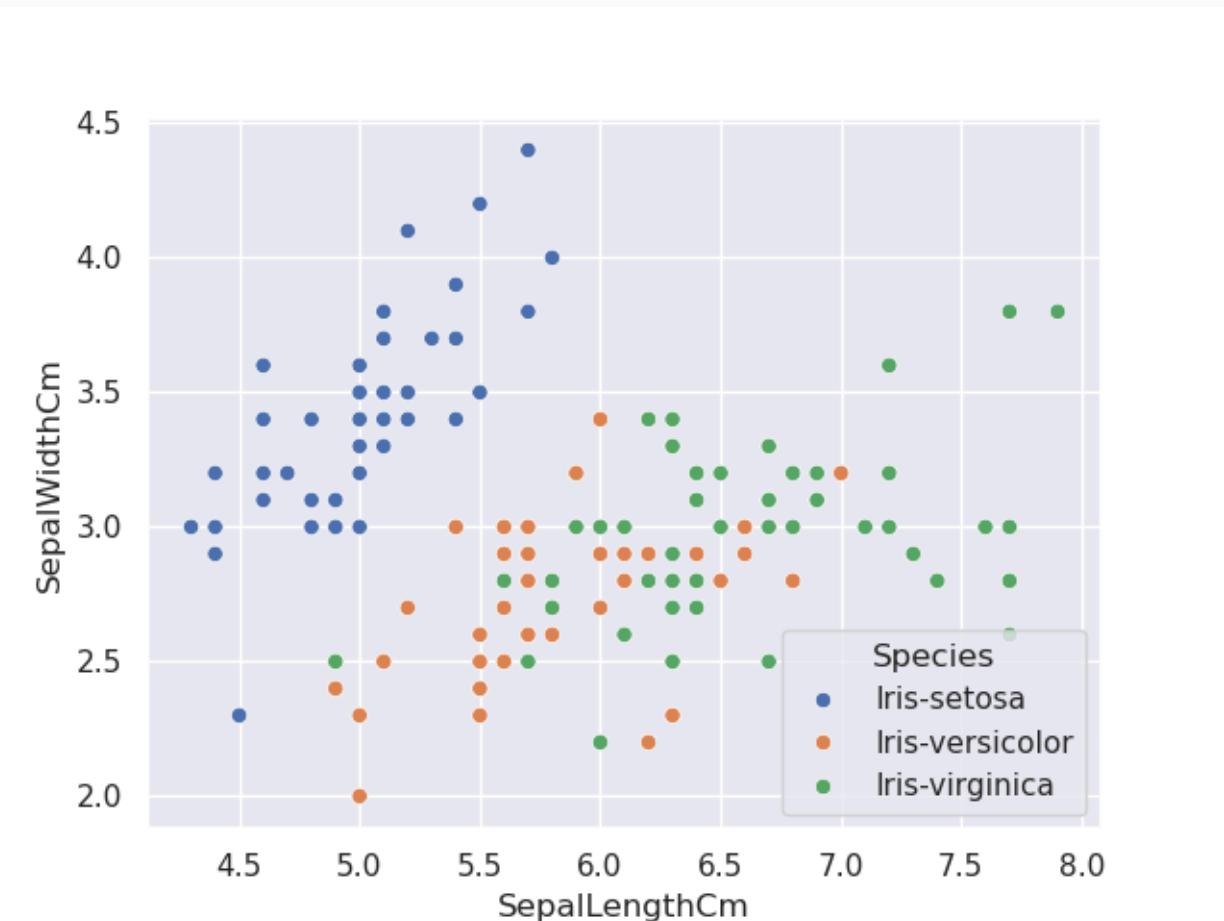


- При $x \rightarrow \infty, g(x) \rightarrow 1$, т.е. принадлежность классу равна 1
- Дифференцируема на \mathbb{R}

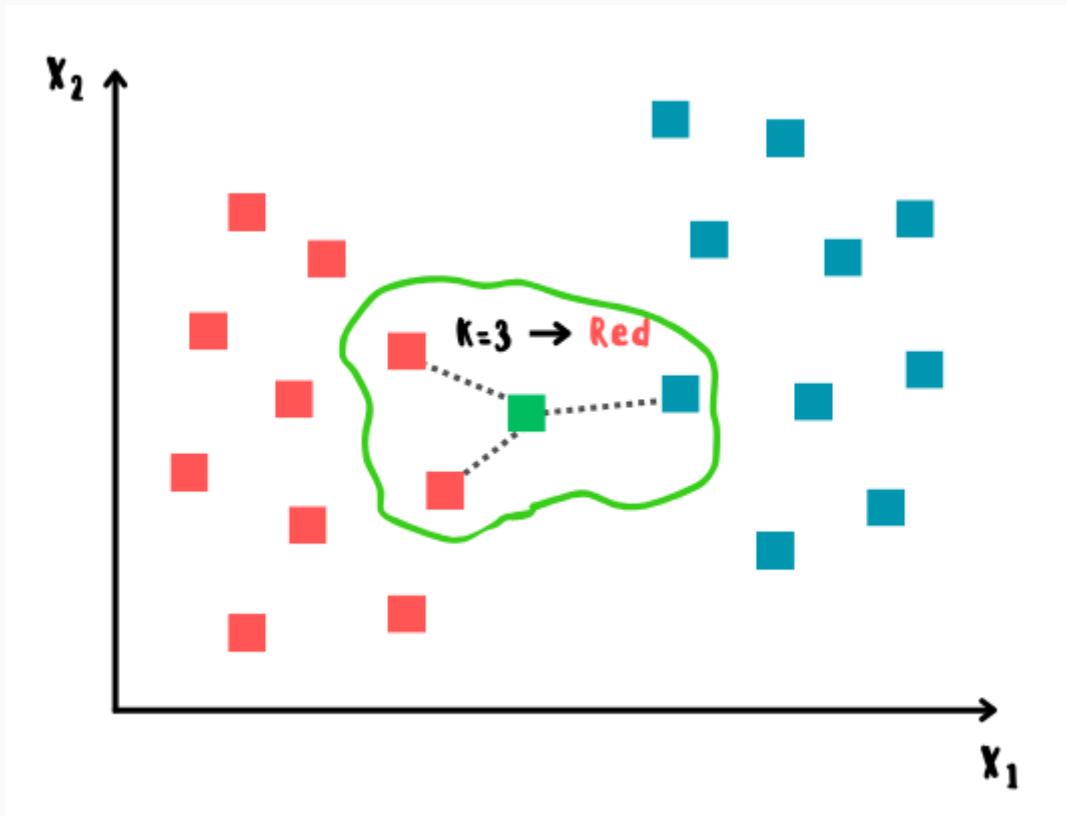


KNN - метод *классификации и регрессии*, основанный на гипотезе компактности, которая предполагает, что расположенные близко друг к другу объекты в пространстве признаков имеют схожие значения целевой переменной или принадлежат к одному классу.

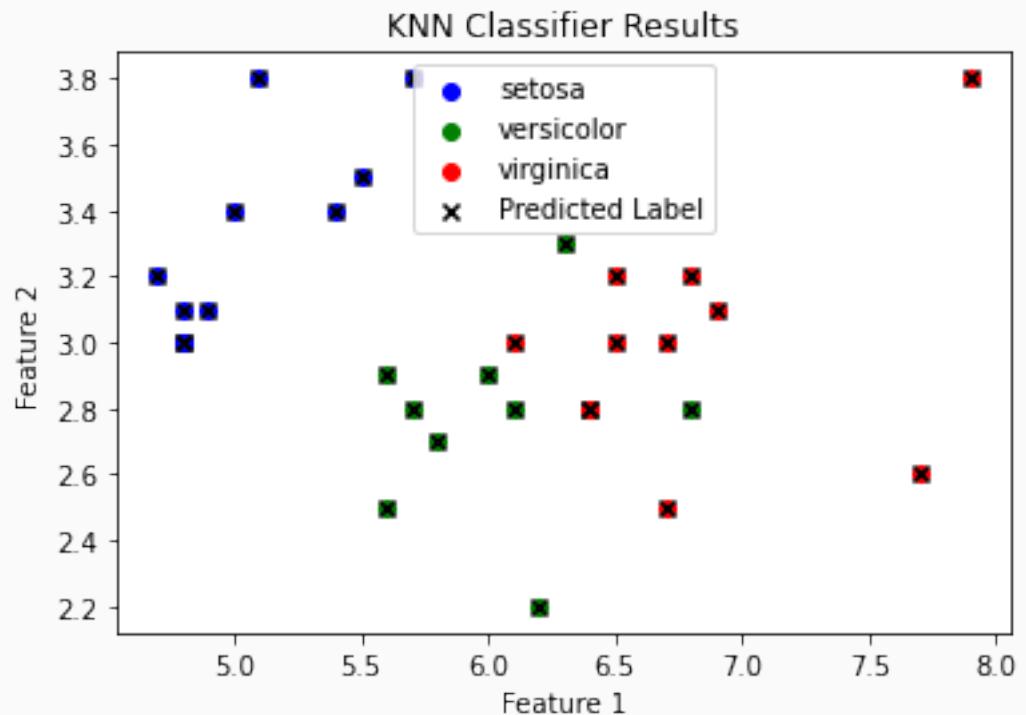
Примерчик



Примерчик



Примерчик



А как определить расстояние между точками?



- Евклидово расстояние:

$$\rho(a, b) = \sqrt{\sum_{i=1}^n (a_i - b_i)^2}$$



- Евклидово расстояние:

$$\rho(a, b) = \sqrt{\sum_{i=1}^n (a_i - b_i)^2}$$

- Манхэттенское расстояние:

$$\rho(a, b) = \sum_{i=1}^n |a_i - b_i|$$



- Евклидово расстояние:

$$\rho(a, b) = \sqrt{\sum_{i=1}^n (a_i - b_i)^2}$$

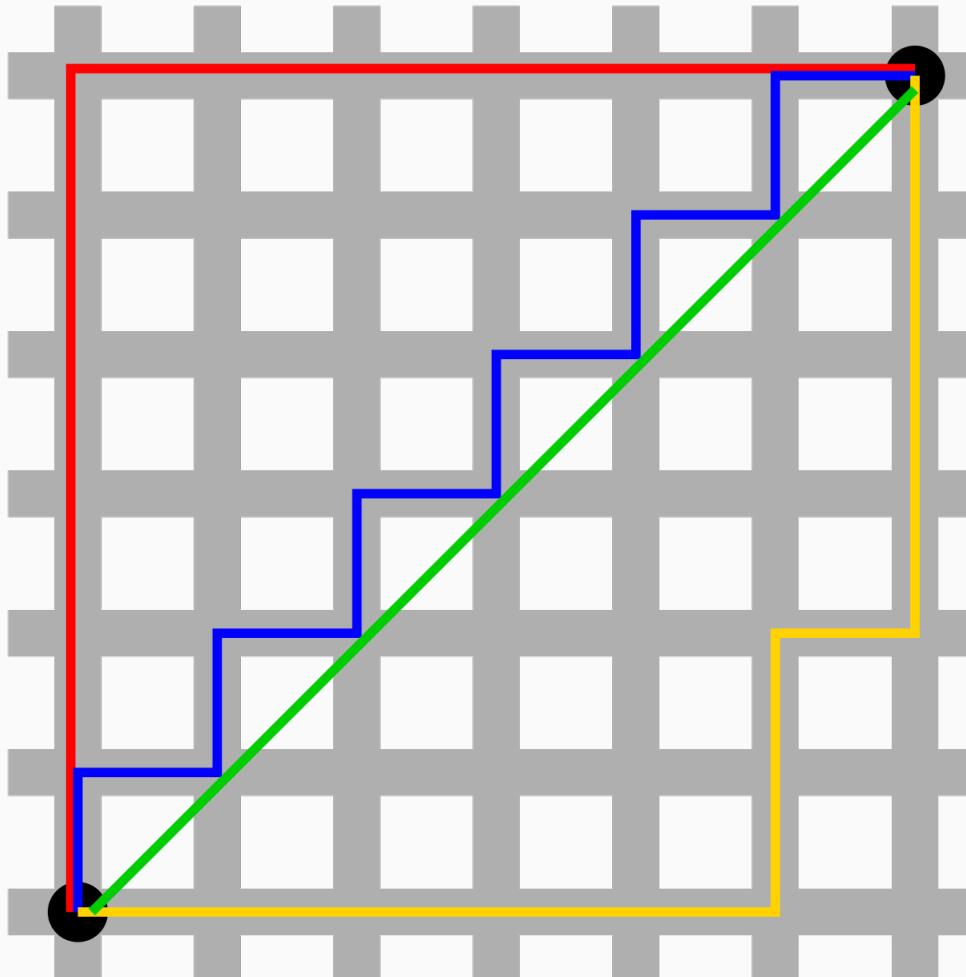
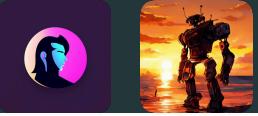
- Манхэттенское расстояние:

$$\rho(a, b) = \sum_{i=1}^n |a_i - b_i|$$

- Косинусное расстояние:

$$\rho(a, b) = -\frac{\sum_{i=1}^n a_i b_i}{\sqrt{\sum_{i=1}^n a_i^2} \sqrt{\sum_{i=1}^n b_i^2}}$$

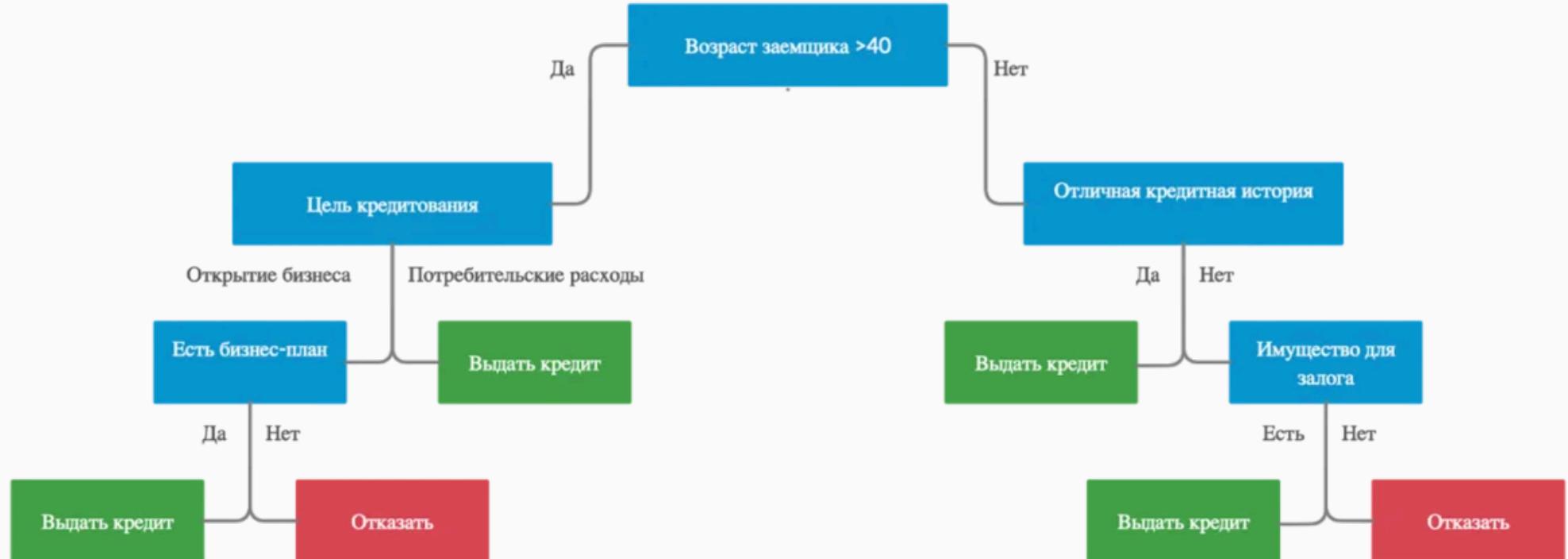
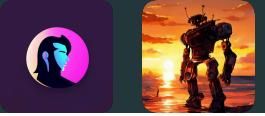
Примерчик

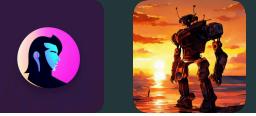




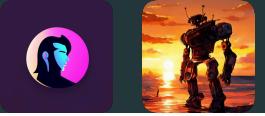
Дерево решений - средство поддержки принятия решений, использующееся в машинном обучении, анализе данных и статистике. Структура дерева представляет собой «листья» и «ветки». На рёбрах («ветках») дерева решения записаны признаки, от которых зависит целевая функция, в «листьях» записаны значения целевой функции, а в остальных узлах — признаки, по которым различаются случаи. Чтобы классифицировать новый случай, надо спуститься по дереву до листа и выдать соответствующее значение.

Дерево решений





- Обязано быть **двоичным**. Только две ветки выходят из узла
- В самых нижних листах обязан быть ответ
- Конечная глубина



Чтобы выбрать правило разбиения в вершине:

$$H = - \sum_{i=1}^n \frac{N_i}{N} \log\left(\frac{N_i}{N}\right)$$

Теоретико-информационный критерий или же энтропия (мера хаоса)



Чтобы выбрать правило разбиения в вершине:

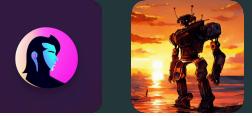
$$H = - \sum_{i=1}^n \frac{N_i}{N} \log\left(\frac{N_i}{N}\right)$$

Теоретико-информационный критерий или же энтропия (мера хаоса)

$$\text{Gain}(A) = \text{Info}(S) - \text{Info}(S_A)$$

$\text{Info}(S)$ - информация, связанная с подмножеством S до разбиения, $\text{Info}(S_A)$ - после разбиения

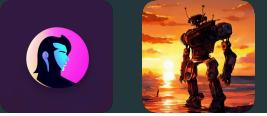
Как обучать модели. Градиентный спуск



Да кто такой этот
ваш градиентный
спуск!?

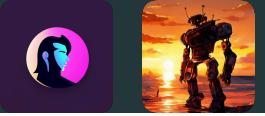


Как обучаются модели?



Наша задача подобрать Θ - веса модели. Но как это делать?

Как обучаются модели?



Наша задача подобрать Θ - веса модели. Но как это делать? Возьмем для примера Логистическую регрессию ([логрег](#)):



Наша задача подобрать Θ - веса модели. Но как это делать? Возьмем для примера Логистическую регрессию ([логрег](#)):

$$z = \left(\sum_{i=1}^n \omega_i x_i \right) + b$$

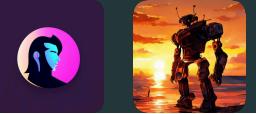
Нам надо найти все ω_i



В зависимости от модели мы можем это сделать аналитически. В случае логрега мы можем это сделать!

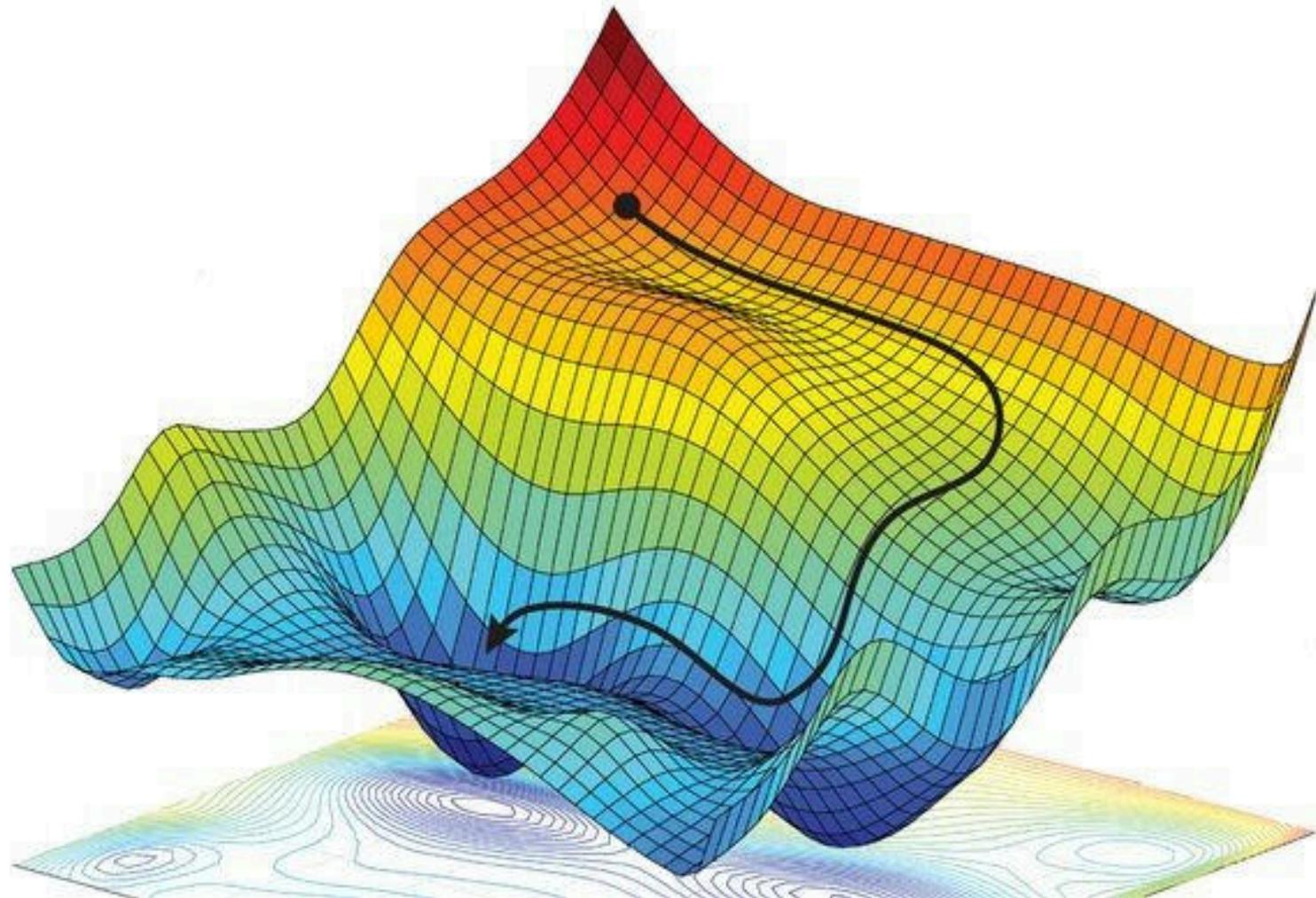
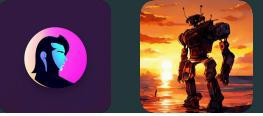
$$\hat{\vec{\omega}} = \arg \min_{\vec{\omega}} J(X, \vec{y}, \vec{\omega}) = \arg \min_{\vec{\omega}} \left(C \sum_{i=1}^l \log(1 + \exp(-y_i \vec{\omega}^T \vec{x}_i)) + |\vec{\omega}|^2 \right)$$

Что если аналитикой не получается?



Градиентный спуск – метод нахождения минимального значения функции потерь. Минимизация любой функции означает поиск самой глубокой впадины в этой функции

Градиентный спуск





1. Инициализировать параметры модели случайными значениями
- 2.
- 3.
- 4.
- 5.



1. Инициализировать параметры модели случайными значениями
2. Подать входные данные в модель и получить предсказания
- 3.
- 4.
- 5.



1. Инициализировать параметры модели случайными значениями
2. Подать входные данные в модель и получить предсказания
3. Вычислить значение функции ошибки, сравнив предсказания с фактическими значениями (Считаем метрику)
- 4.
- 5.



1. Инициализировать параметры модели случайными значениями
2. Подать входные данные в модель и получить предсказания
3. Вычислить значение функции ошибки, сравнив предсказания с фактическими значениями (Считаем метрику)
4. Определить градиент функции ошибки по каждому параметру модели
- 5.



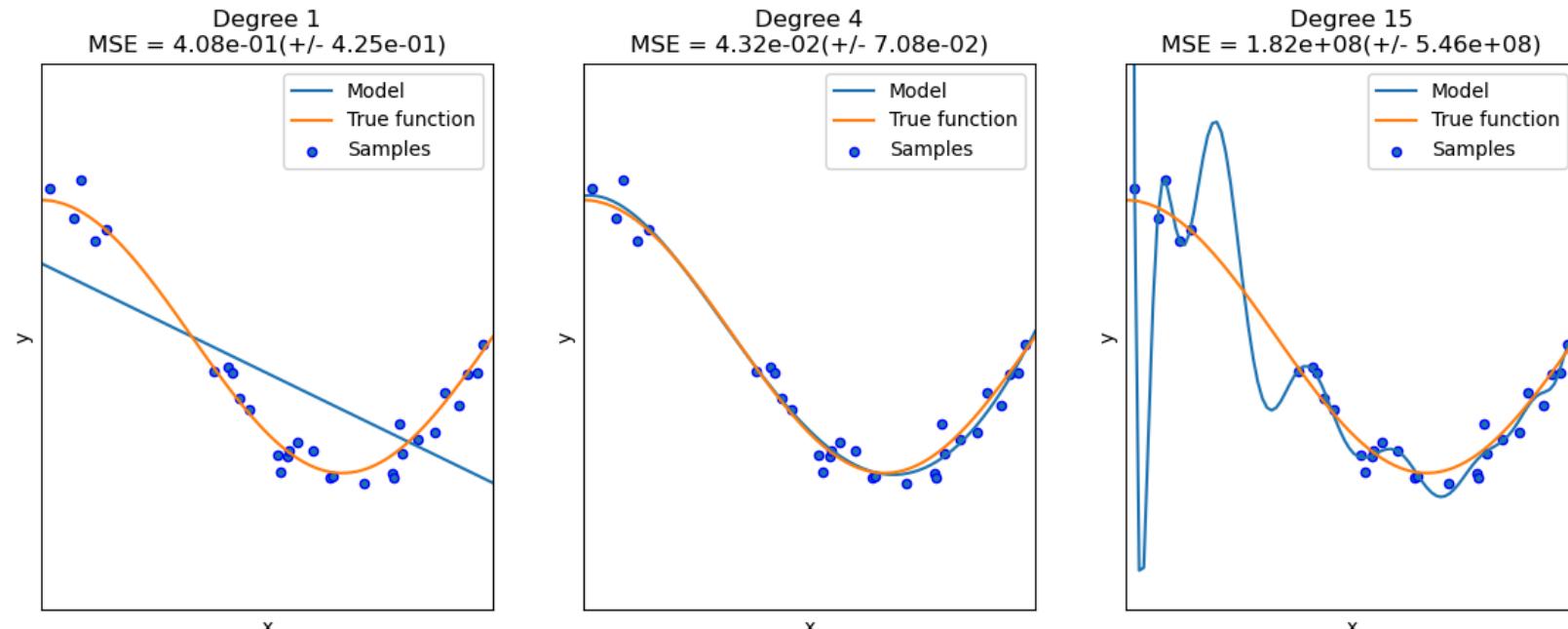
1. Инициализировать параметры модели случайными значениями
2. Подать входные данные в модель и получить предсказания
3. Вычислить значение функции ошибки, сравнив предсказания с фактическими значениями (Считаем метрику)
4. Определить градиент функции ошибки по каждому параметру модели
5. Обновляем параметры по схеме: $\omega^* = \omega - \alpha \cdot \nabla$

ω^* - новые веса, ω - прежние веса, ∇ - градиент, α - learning rate

Ошибки

Почему предыдущее решение неправильное...

Переобучение и Недообучение - это кто...



Как это исправить?



- Train/Test split
-
-
-



- Train/Test split
- Кросс валидация
-
-

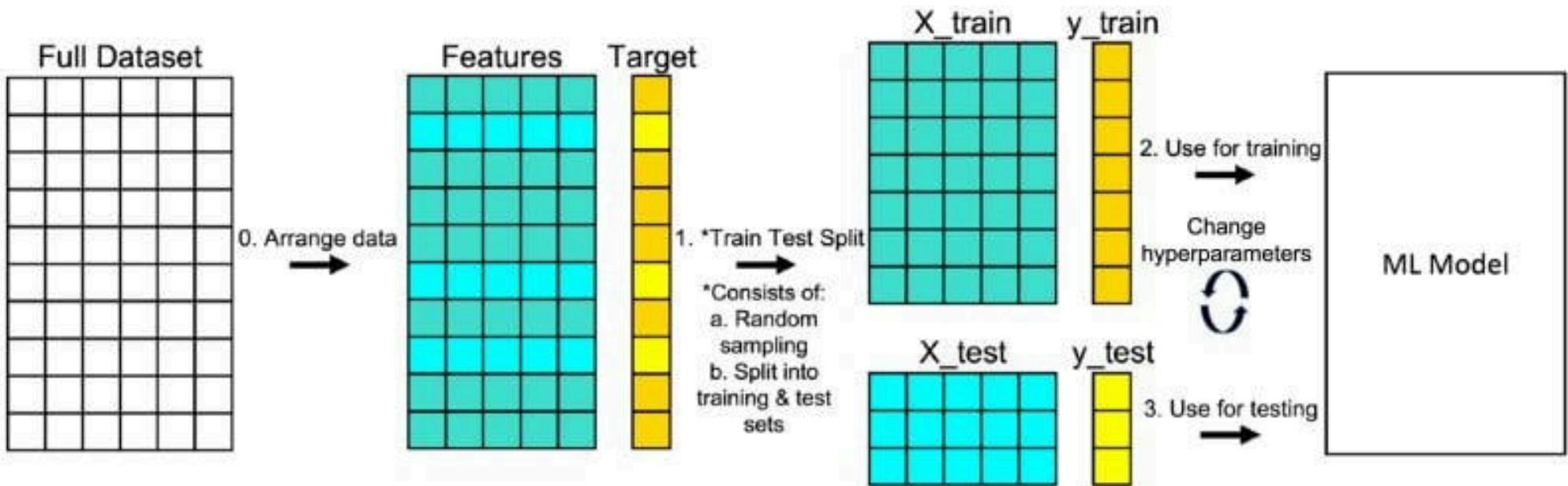


- Train/Test split
- Кросс валидация
- Исправление **гиперпараметров** модели
-



- Train/Test split
- Кросс валидация
- Исправление **гиперпараметров** модели
- Early stopping

Train/Test split

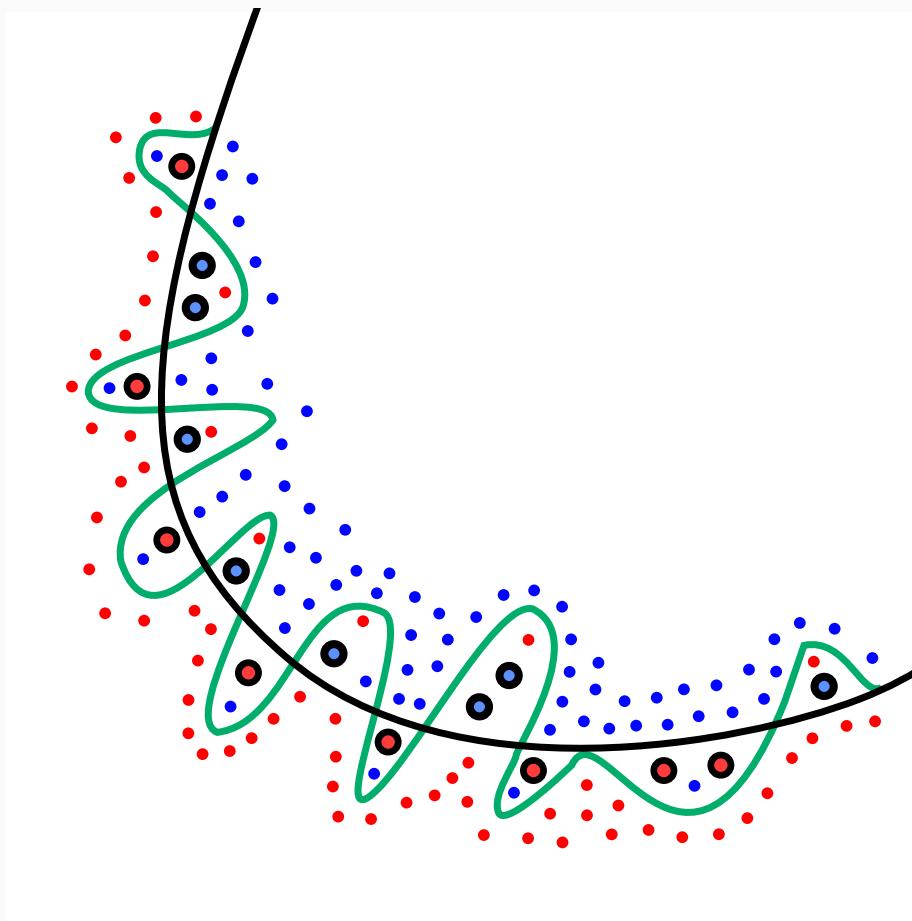




Ну разбили мы и разбили, а почему стало работать?



Ну разбили мы и разбили, а почему стало работать?



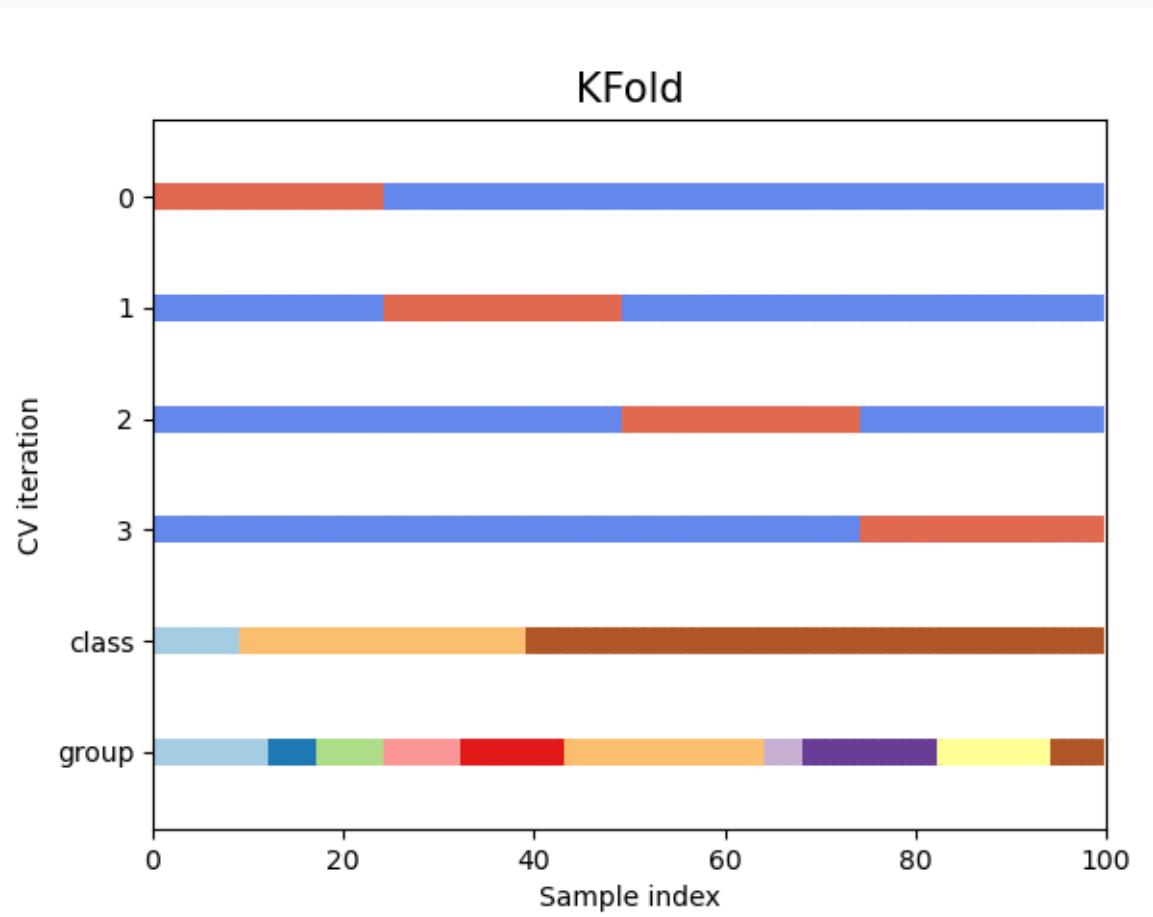


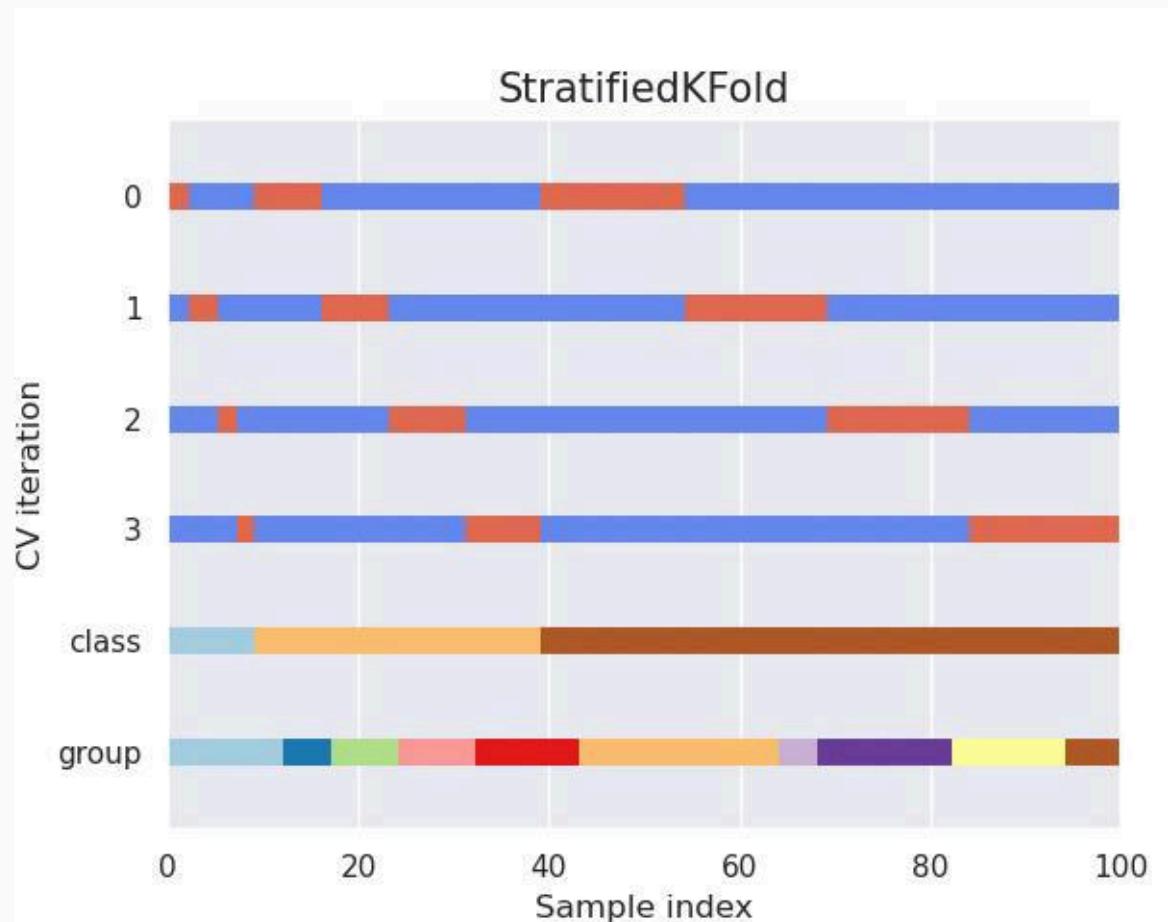
Обучаясь на `train` части, вы оставляете `test`, который модель вообще не видела, значит не могла на нем переобучиться. И вы сможете увидеть, что на `test` возникают проблемы. После этого применяете меры по устранению



- KFold
- GroupKFold
- StratifiedKFold
- TimeSeriesSplit

и др.





Как пользоваться кроссвалидацией



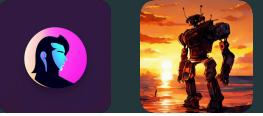
```
scores = []
kf = KFold(n_splits=n_splits, shuffle=True, random_state=2024)
for num, (train_index, test_index) in enumerate(kf.split(X)):
    X_train, X_test = X.iloc[train_index], X.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]

model = ...
model.fit(X_train, y_train)

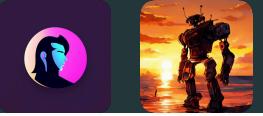
y_pred = model.predict(X_test)

score = metric(y_test, y_pred)
scores.append(score)
```





Автоматический поиск гиперпараметров рассмотрим позже



Автоматический поиск гиперпараметров рассмотрим позже [Optuna](#)

Отдых!



Переходим к практике



Спасибо за внимание!