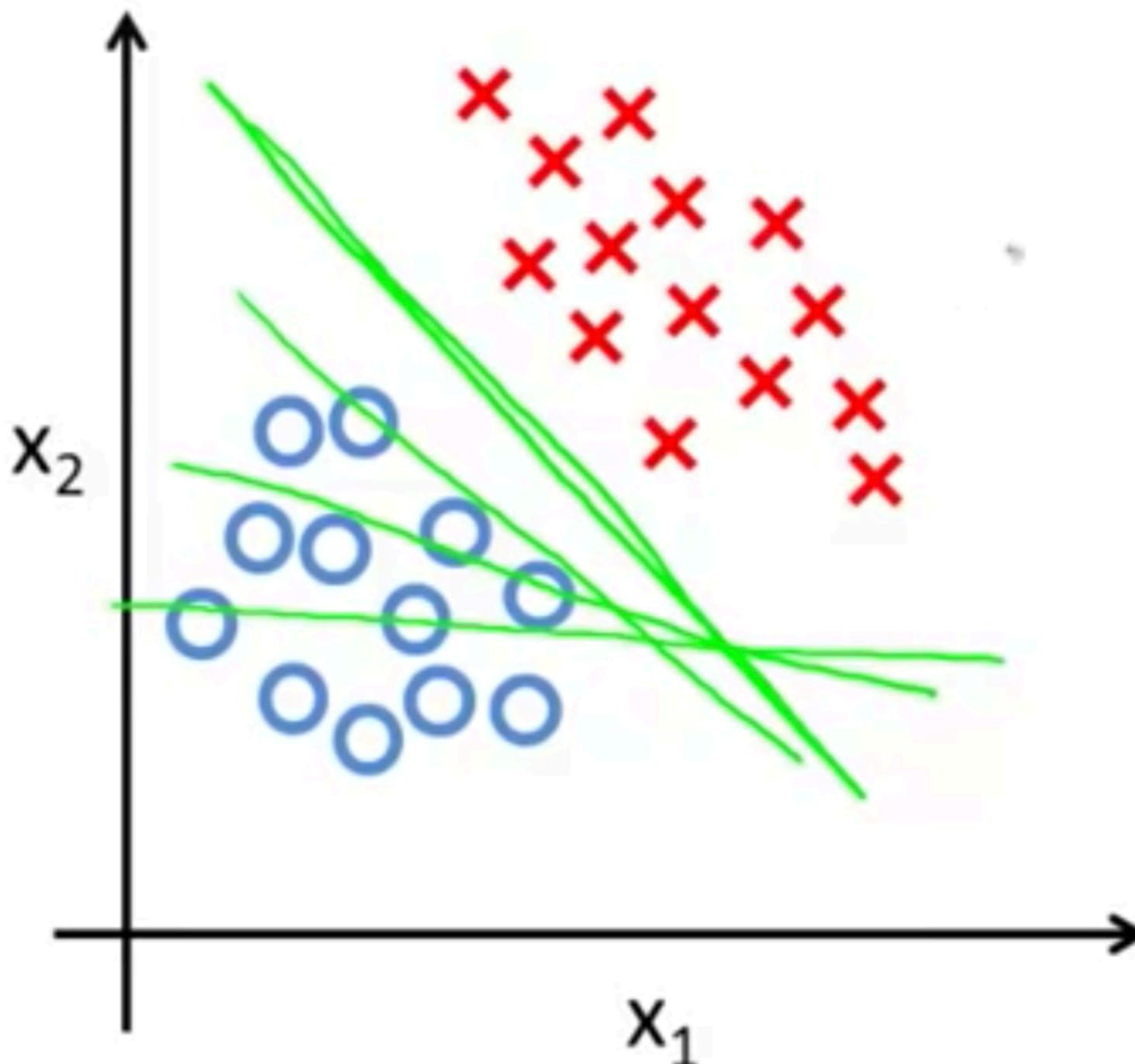
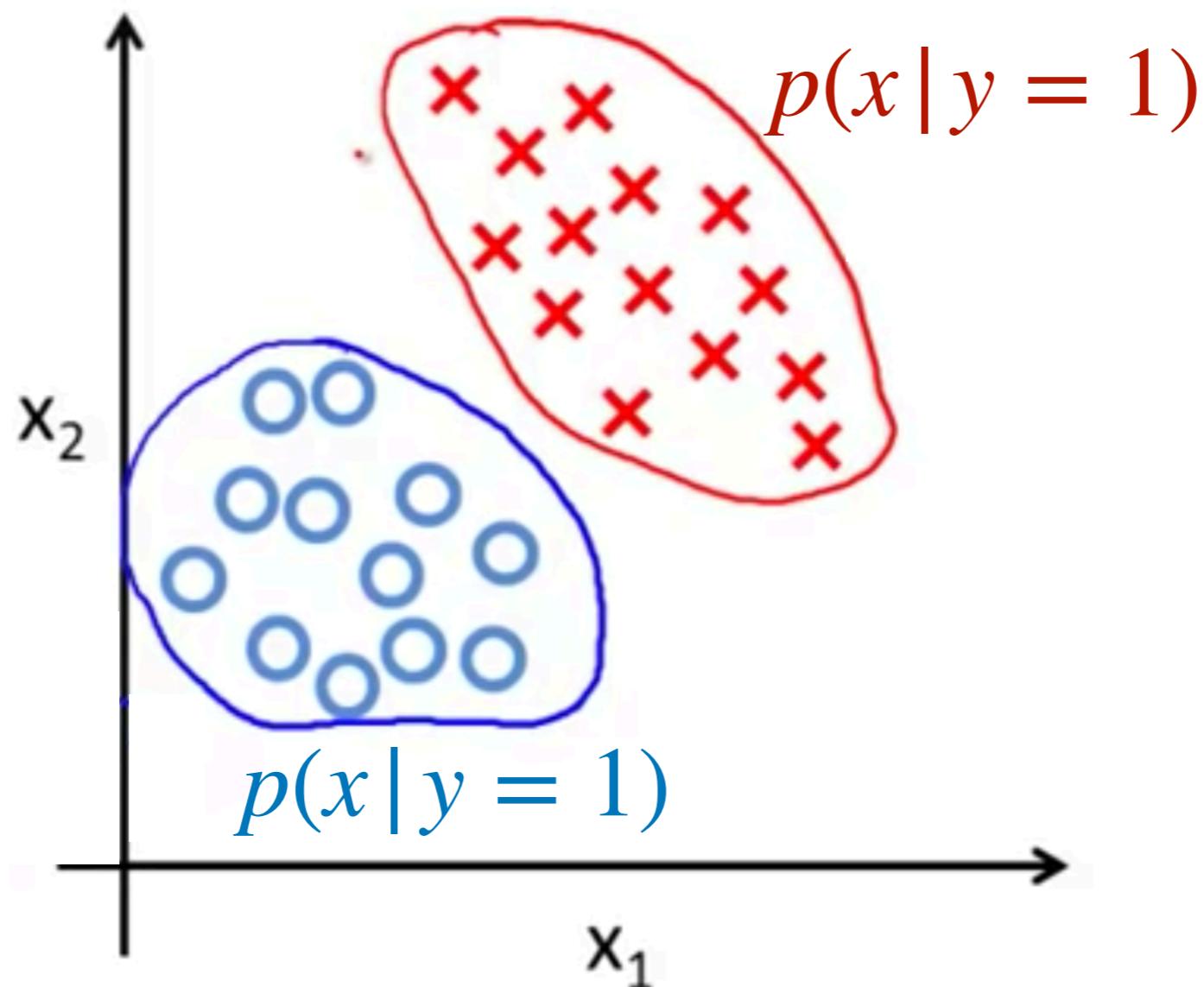


Генеративные vs дискриминативные модели



Дискриминативная модель ищет разделяющую плоскость. Задача дискриминативной модели найти $p(\text{class}|\mathbf{x})$. Если она может это делать, то дальше просто для \mathbf{x} выбираем класс с наибольшей вероятностью

Генеративная модель

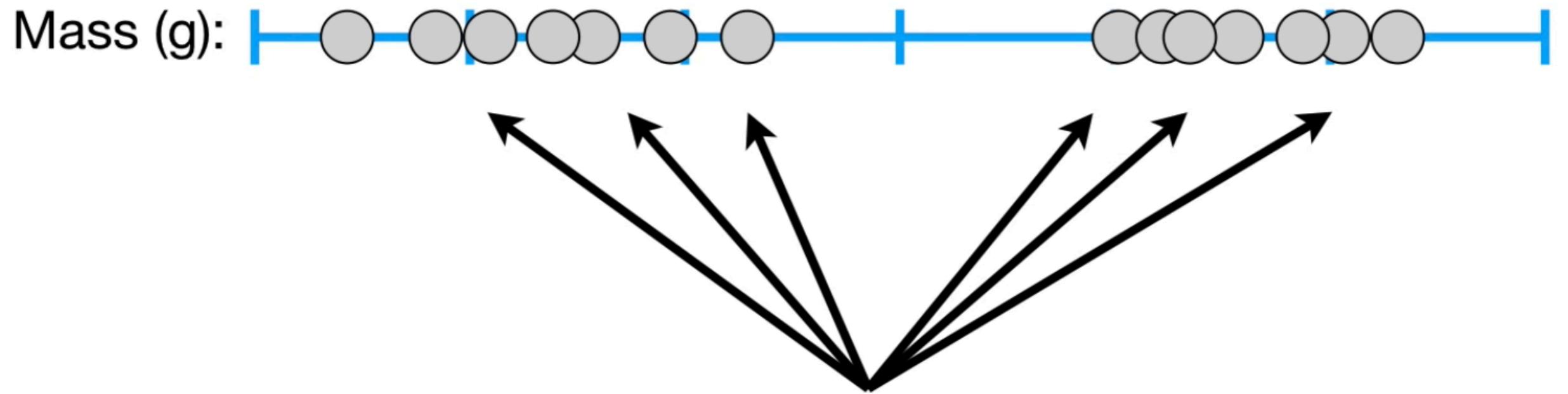


И еще знать это:

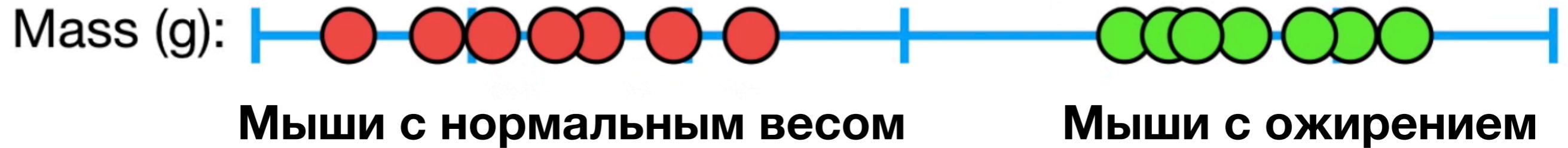
$$p(y)$$

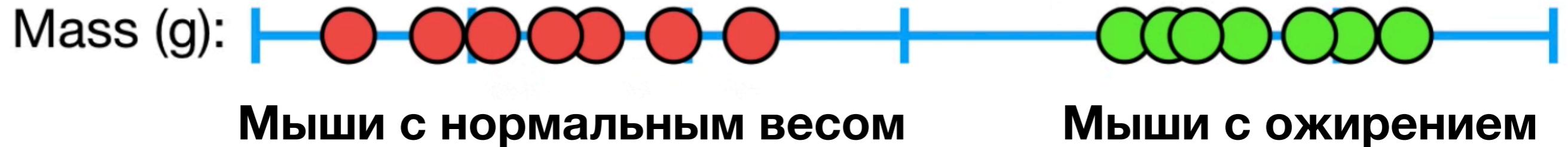
Генеративная модель пытается найти $p(y)$, $p(x, y)$ и $p(x|y)$ (второе и третье выражаются друг через друга при условии знания $p(y)$).

$$p(x, y) = p(x | y) \cdot p(y)$$

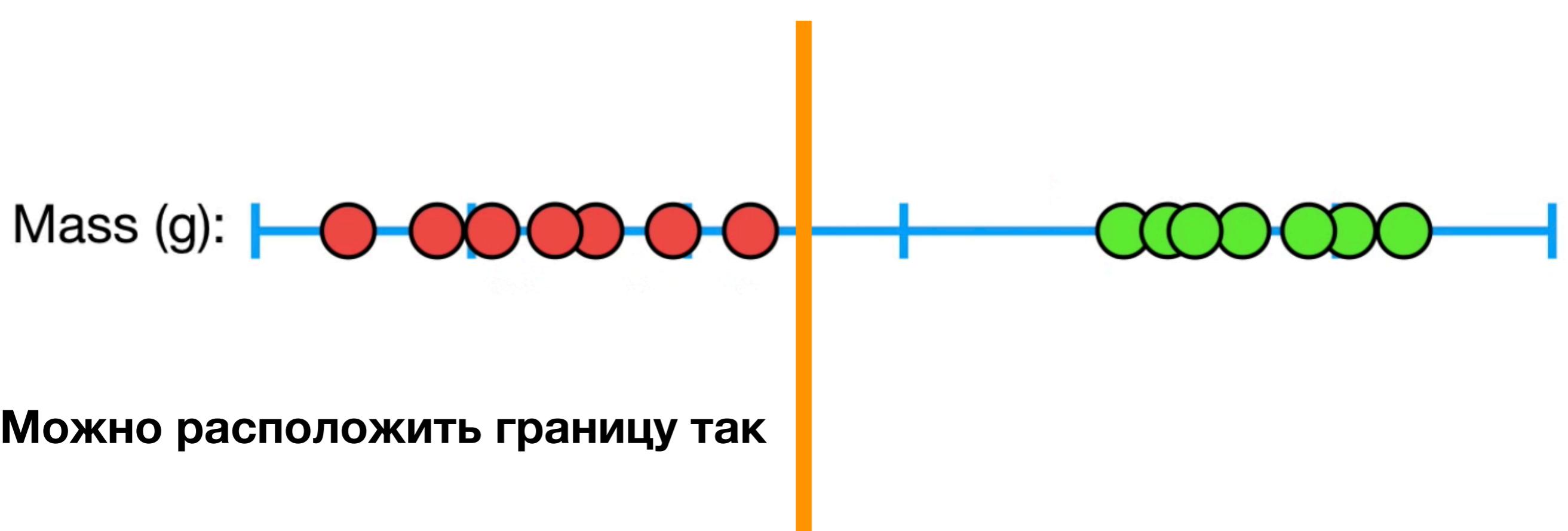


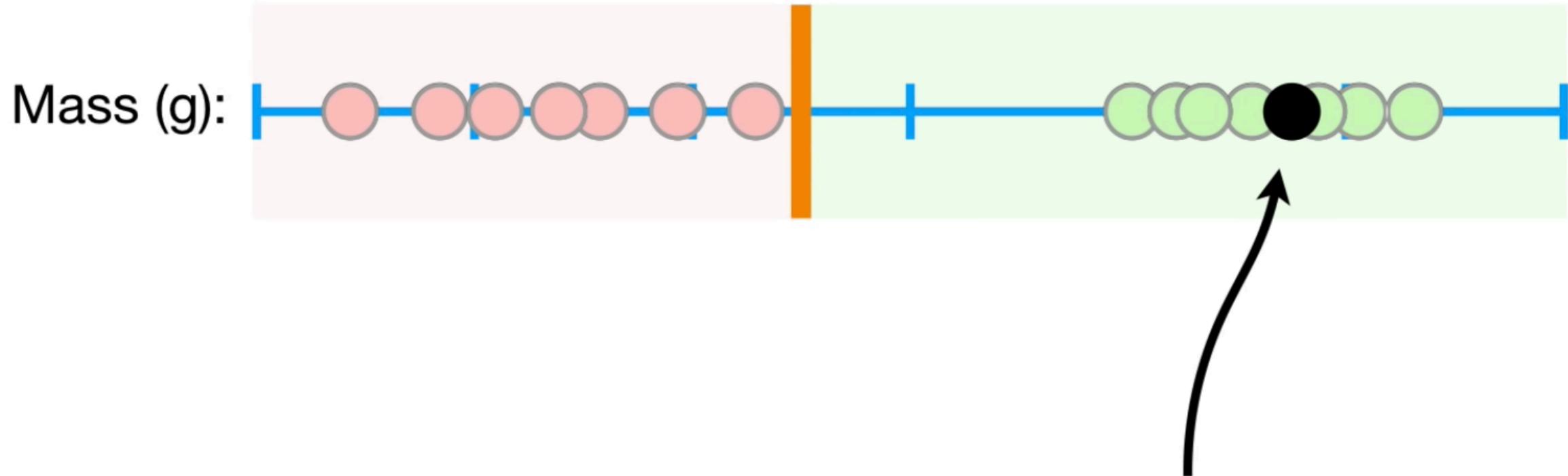
Представим себе, что мы измеряем массу мышей





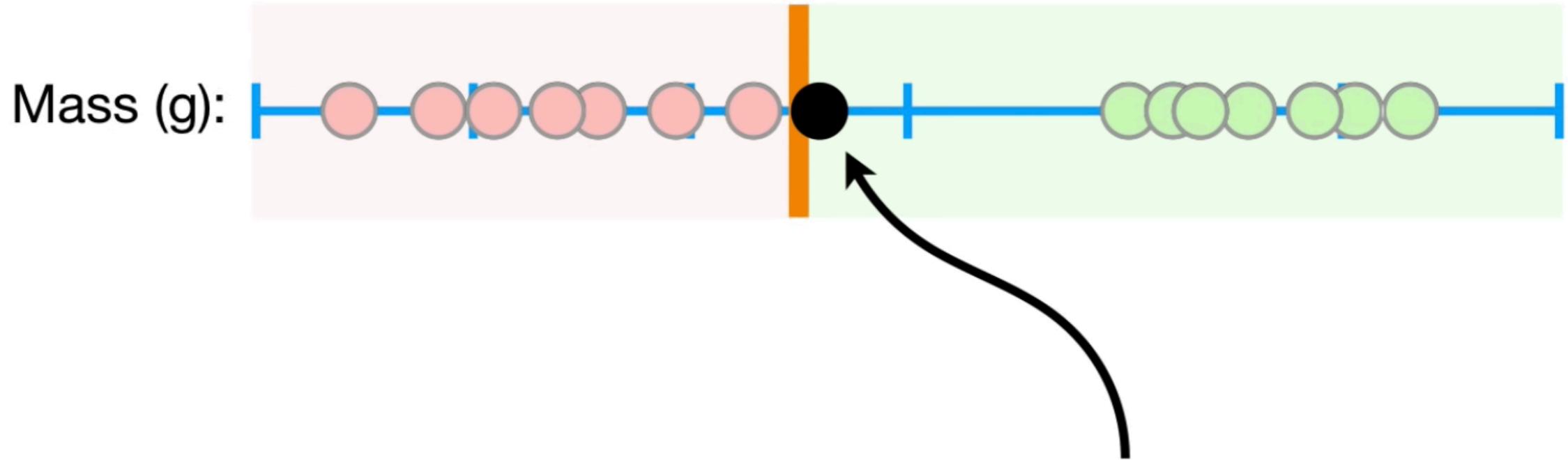
Мы хотим поставить какую-то границу, по которой отличать их



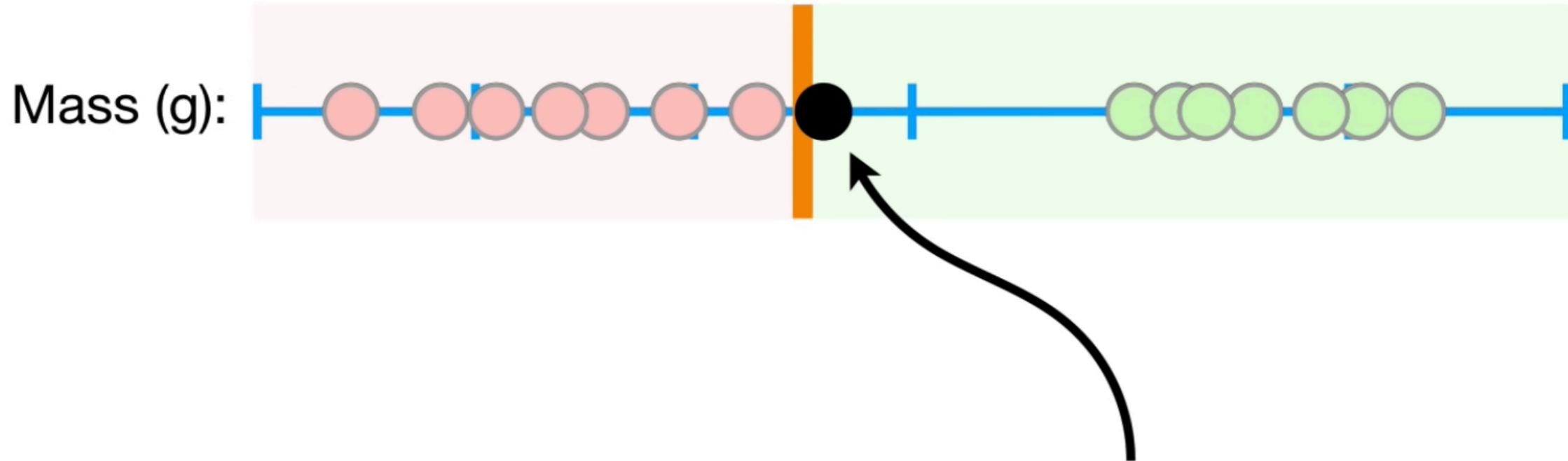


**Такую мышь
классифицируем
как мышь с избыточным весом**

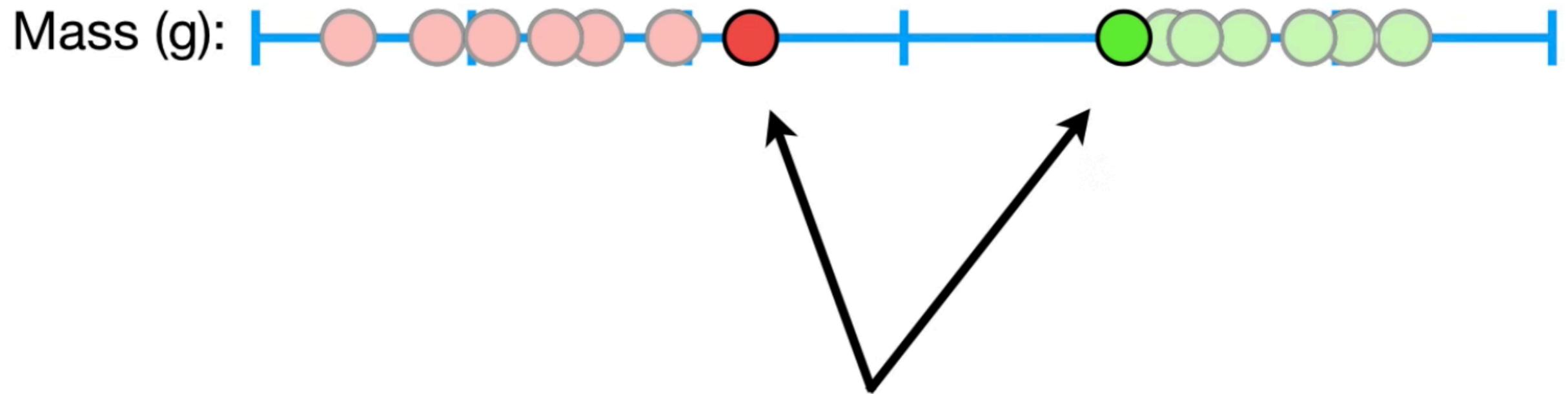
Все ли хорошо?



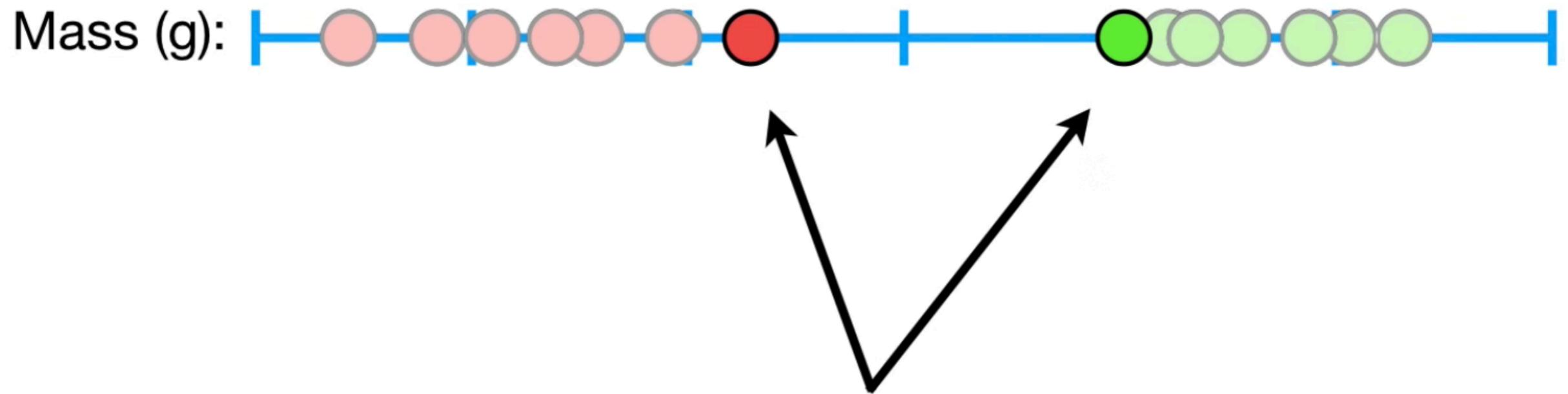
А что с этим наблюдением?



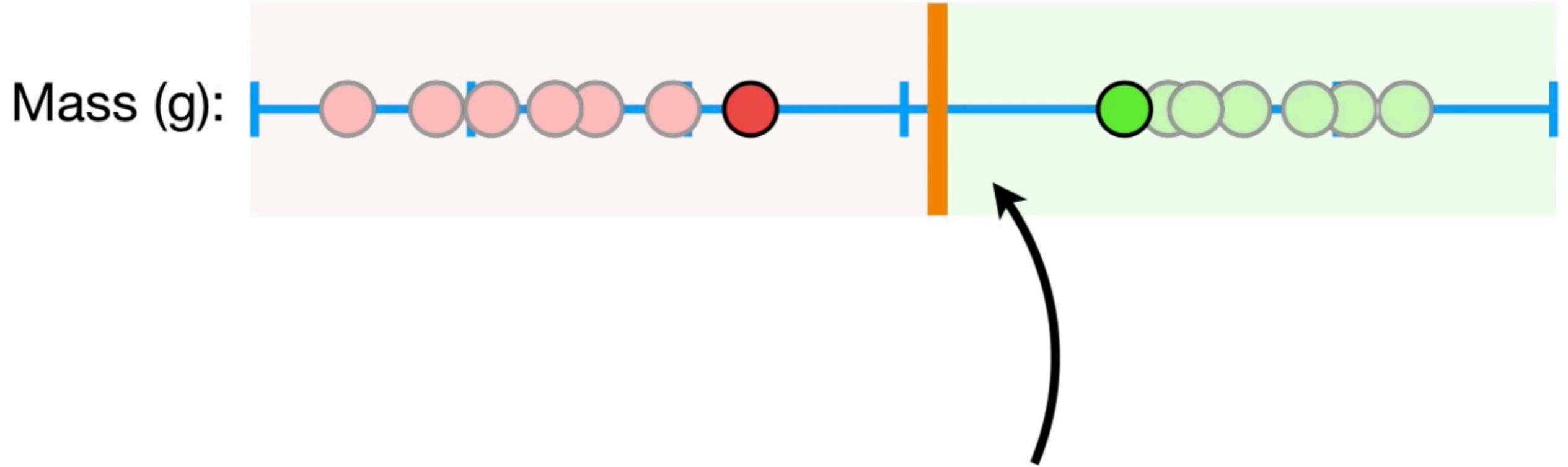
**А что с этим наблюдением?
Разумно ли его классифицировать
как мышь с ожирением?**



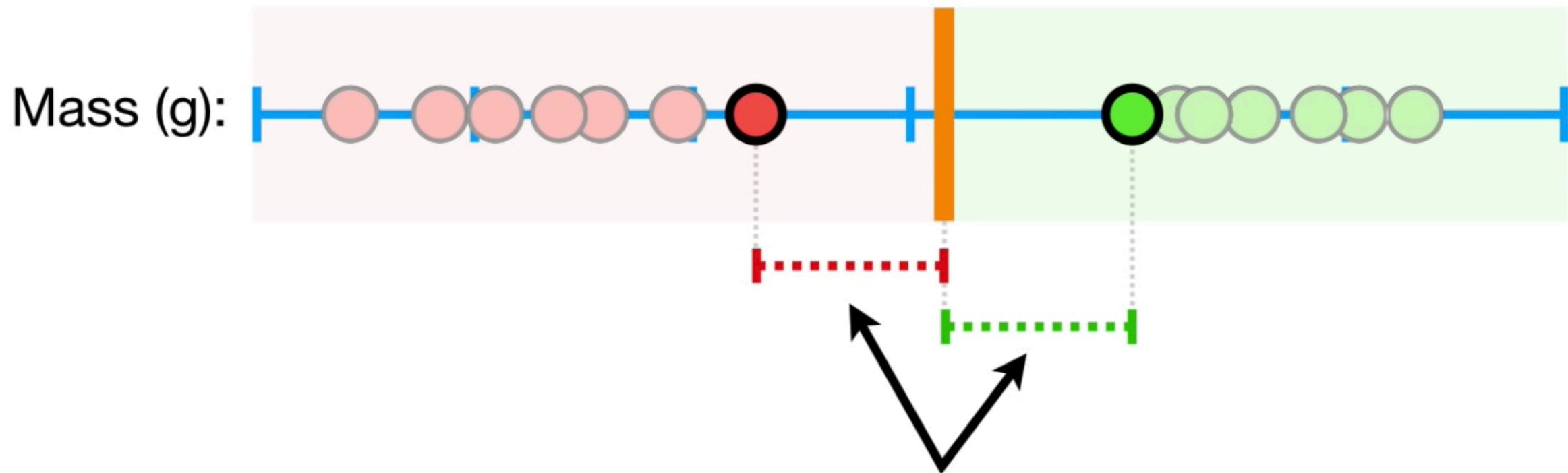
Давайте будем концентрироваться на “краях” каждой группы



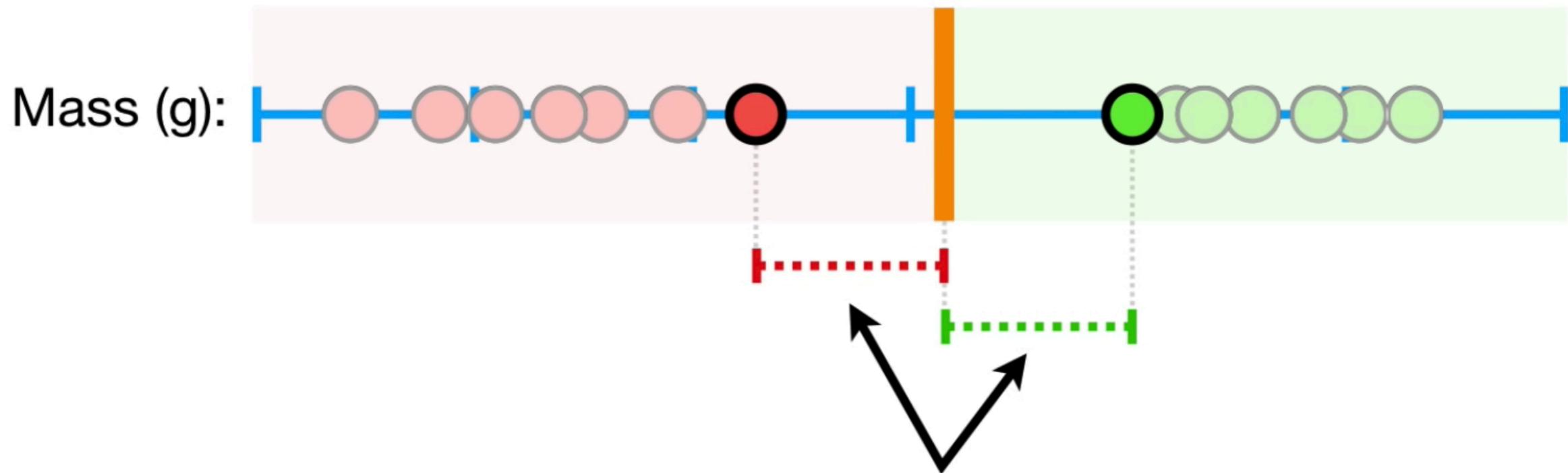
Давайте будем концентрироваться на “краях” каждой группы



Поставим нашу решающую границу между этими точками



Самая маленькое расстояние между наблюдениями и границей называется **зазором (margin)**

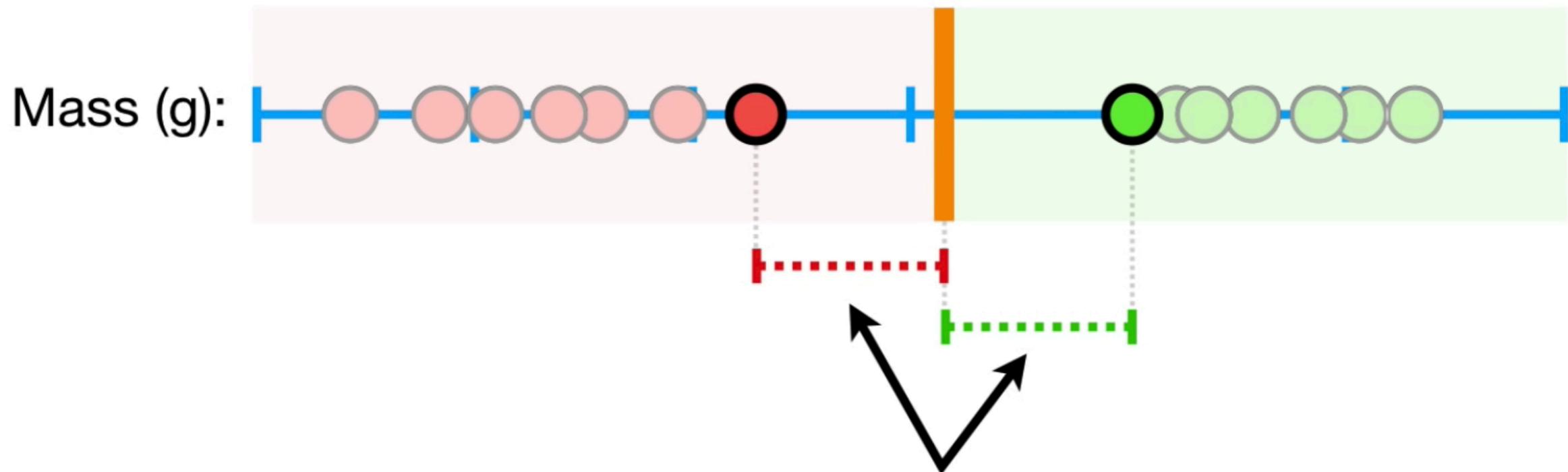


Мы ставим нашу граница так, чтобы margin был максимальен и при этом у нас не было ошибок классификации

Maximum margin classifier

$$\max_b \min_x \text{dist}(x, b)$$

b-decision **boundary** (граница решений)



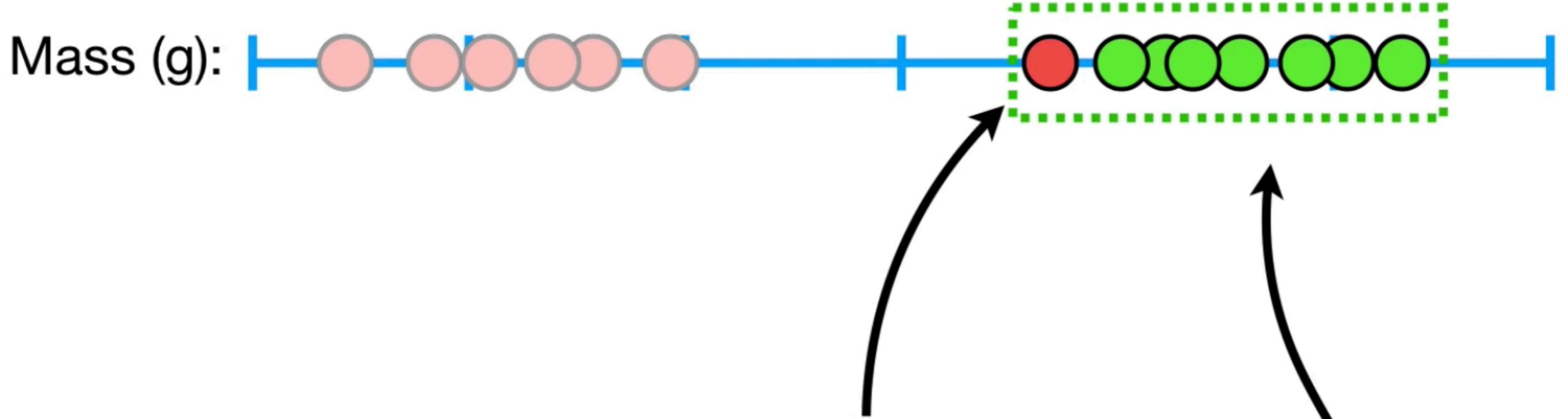
Мы ставим нашу граница так, чтобы margin был максимальен и при этом у нас не было ошибок классификации

Maximum margin classifier

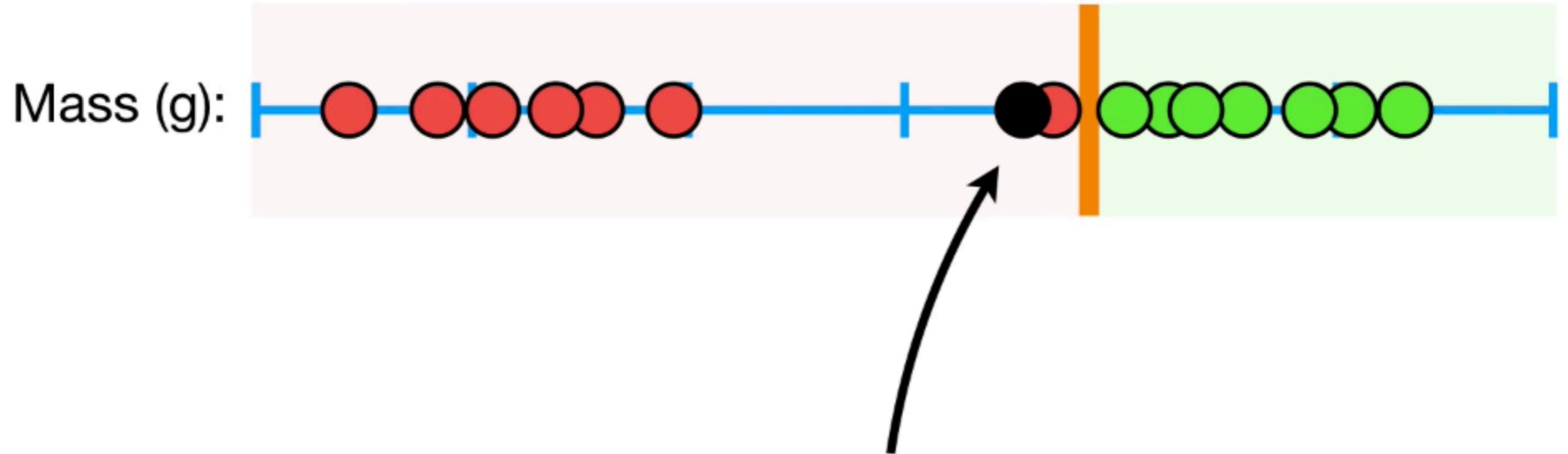
$$\max_b \min_x dist(x, b)$$

b-decision **boundary** (граница решений)

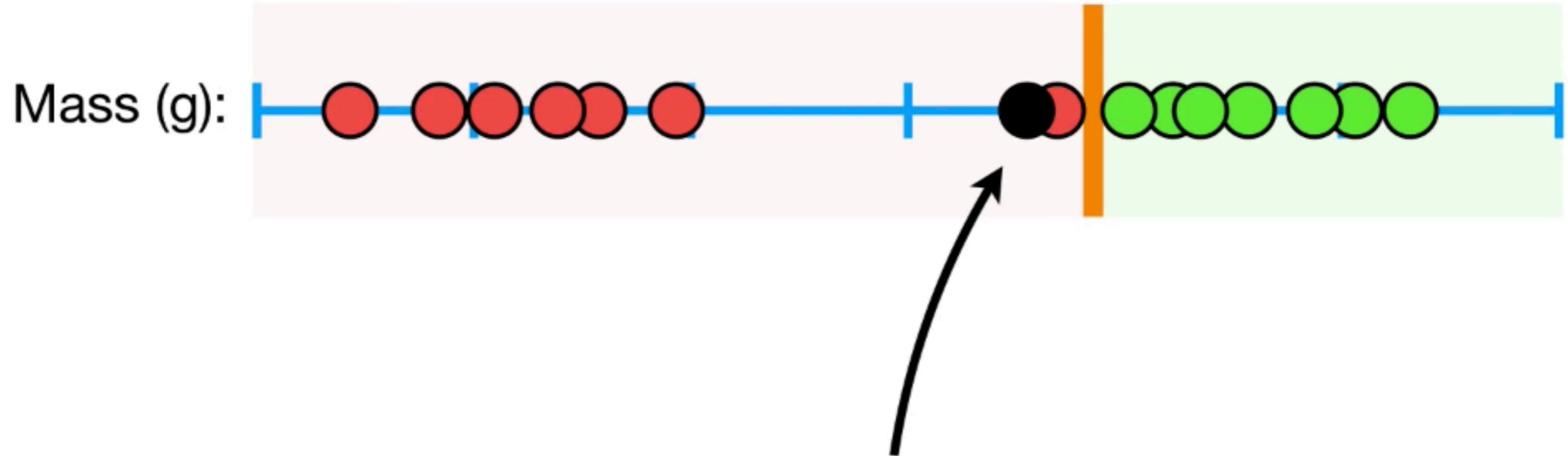
Все ли хорошо?



**Допустим у нас есть мышь,
которая ошибочно отмечена как мышь
без ожирения**

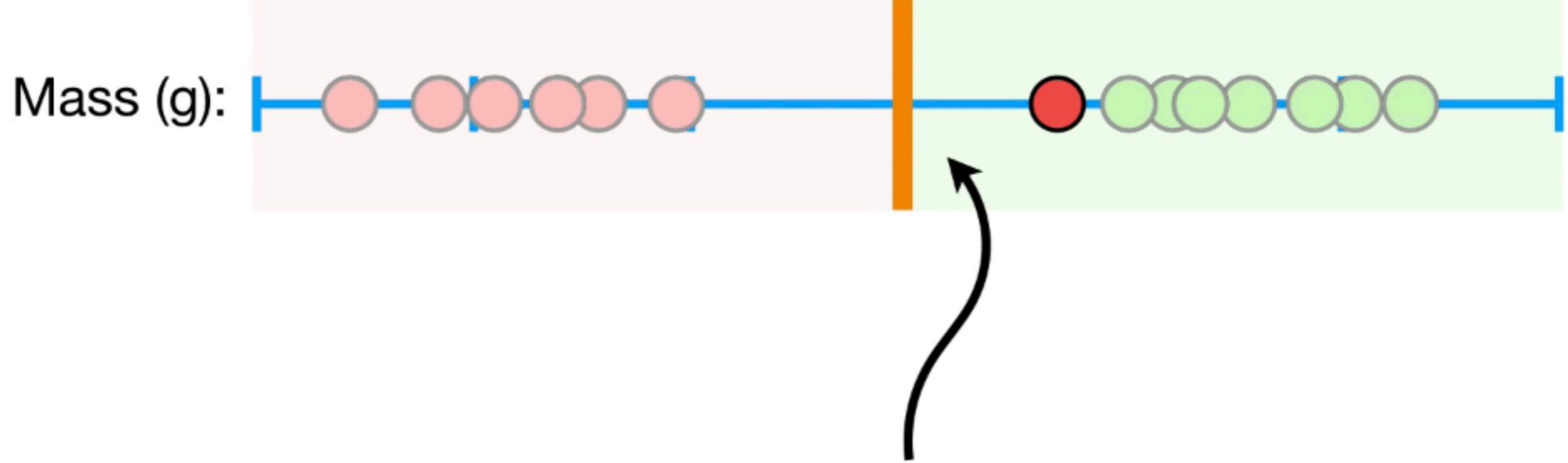


Мы классифицируем эту точку как мышь без ожирения, хотя она очевидно намного ближе к мышам с ним

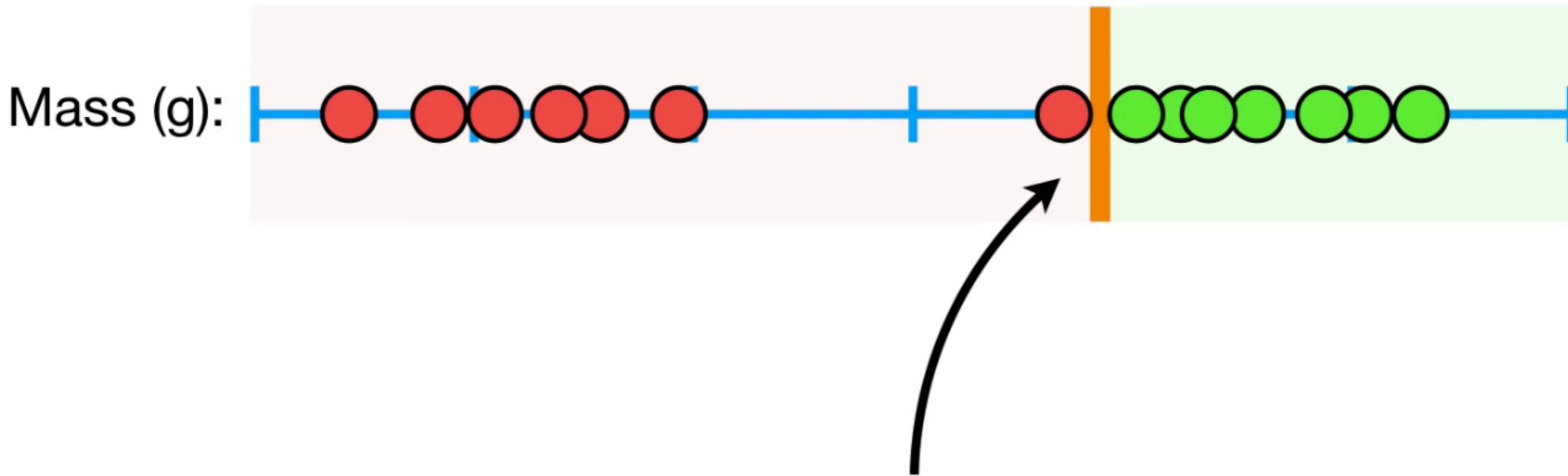


Мы классифицируем эту точку как мышь без ожирения, хотя она очевидно намного ближе к мышам с ним

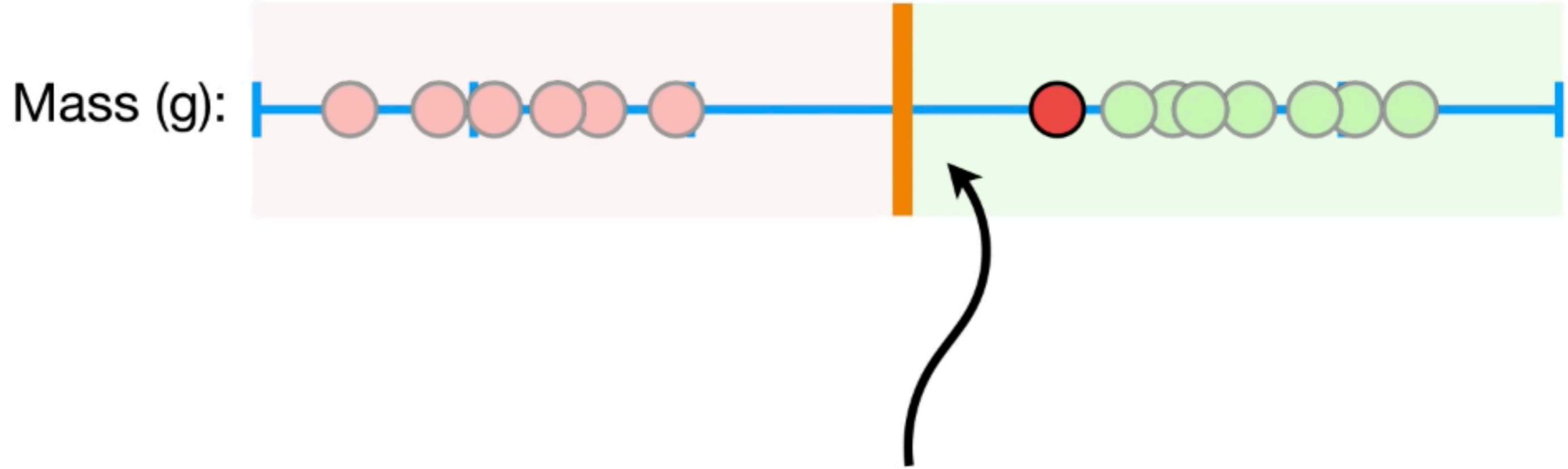
Наш классификатор крайне неустойчив к выбросам:(



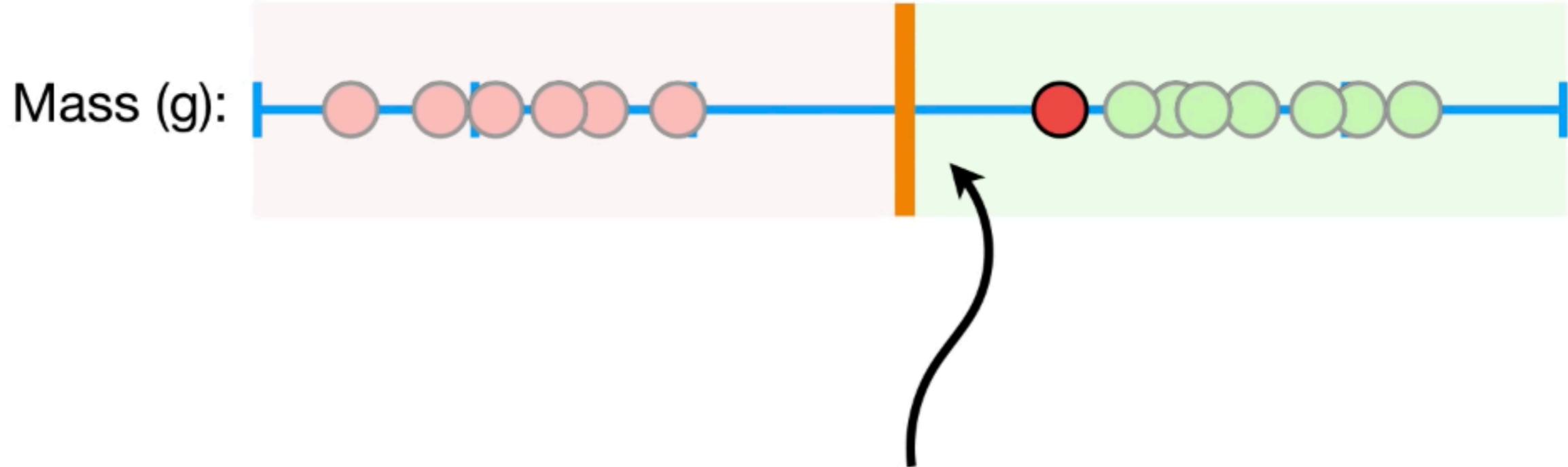
**Теперь мы выбираем порог, позволяя нам
изредка ошибаться на обучающем
множестве**



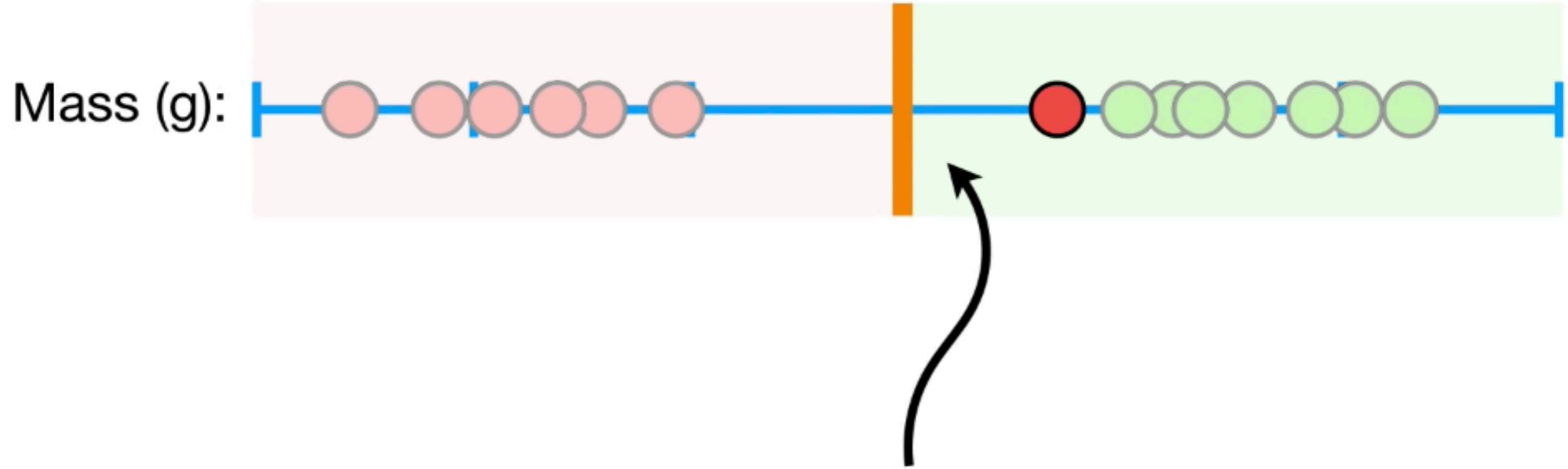
**Наша модель в данном случае пыталась запомнить все из
обучающей выборки. У нее был низкий bias.**



Теперь наша модель менее чувствительна к тренировочным данным. Bias стал больше, но сильно уменьшился variance.

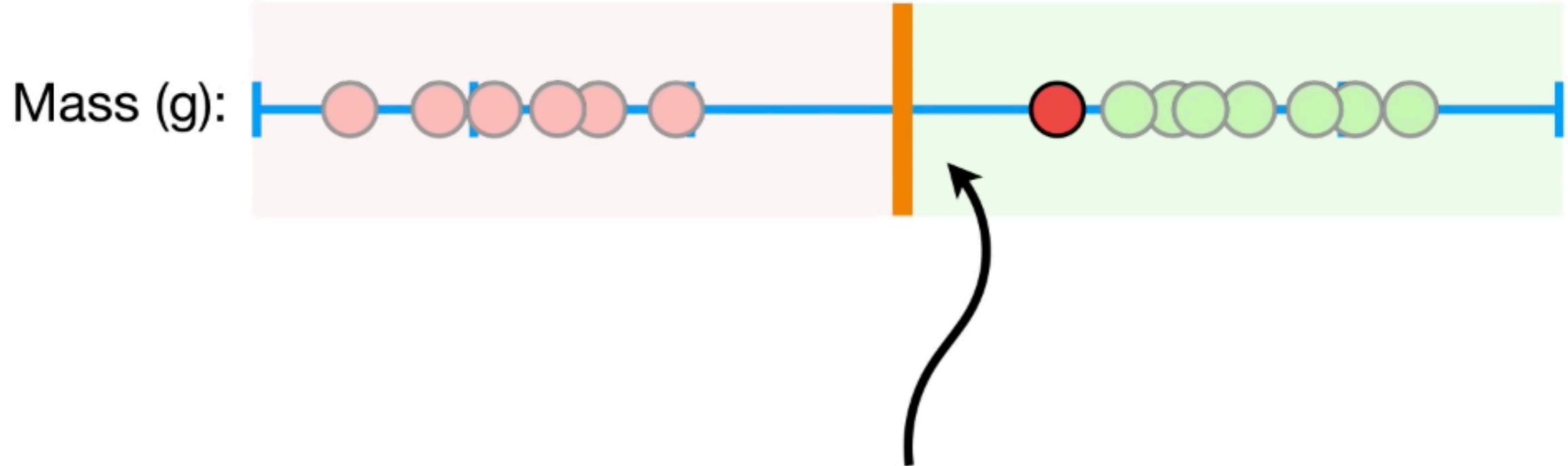


То есть в данном типе моделей за **bias-variance tradeoff** в числе прочего отвечает то, сколько мы позволяем неверных классификациям на тренировочной выборке



То есть в данном типе моделей за **bias-variance tradeoff** в числе прочего отвечает то, сколько мы позволяем неверных классификация на тренировочной выборке

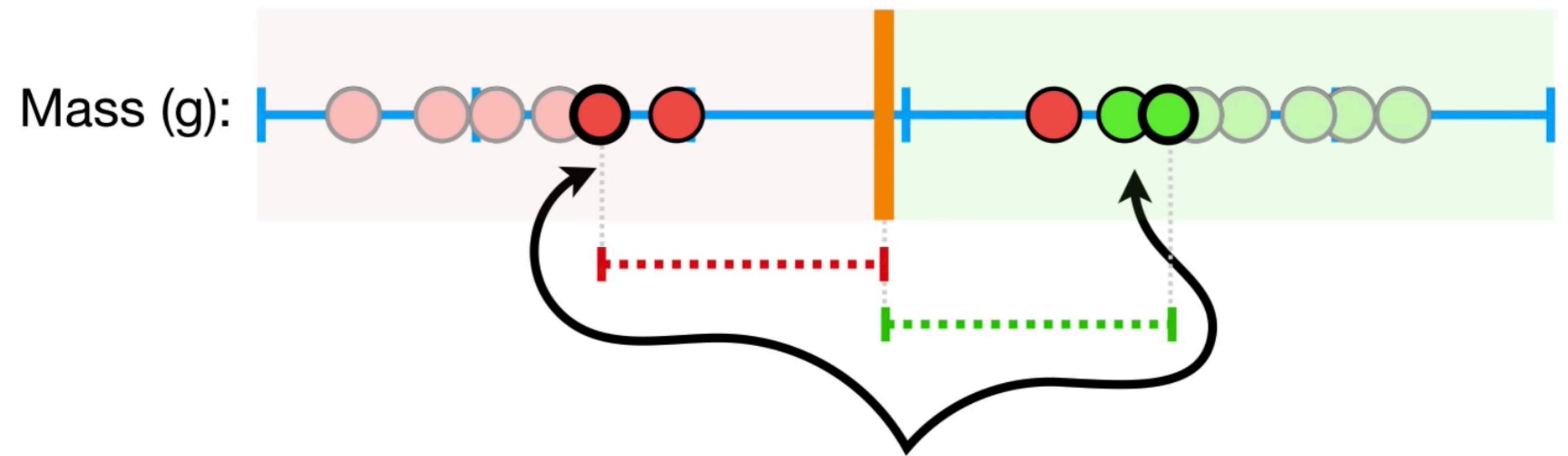
Как мы определяем насколько неточным мы можем позволить быть классификатору?



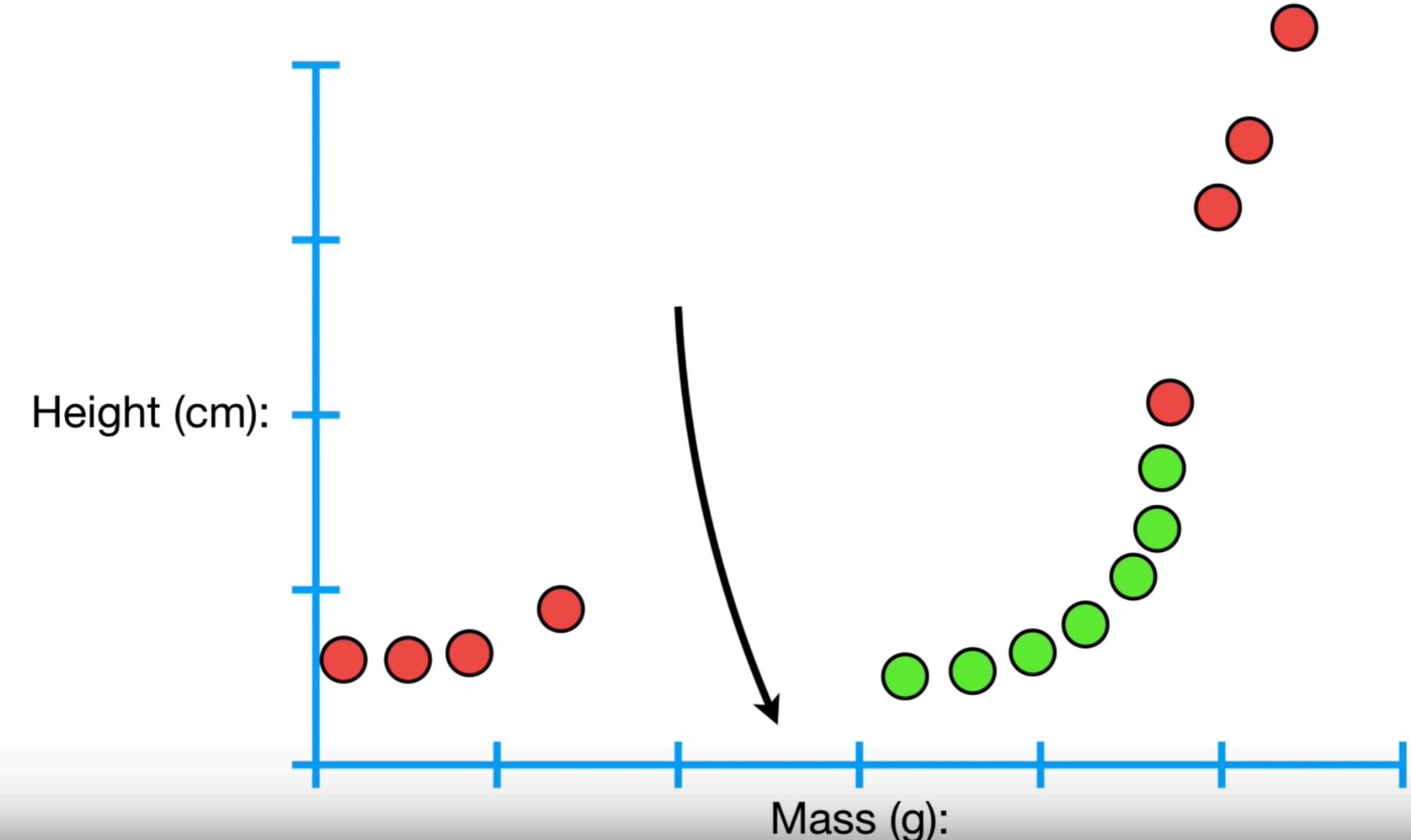
То есть в данном типе моделей за **bias-variance tradeoff** в числе прочего отвечает то, сколько мы позволяем неверных классификация на тренировочной выборке

Как мы определяем насколько неточным мы можем позволить быть классификатору?

С помощью валидации

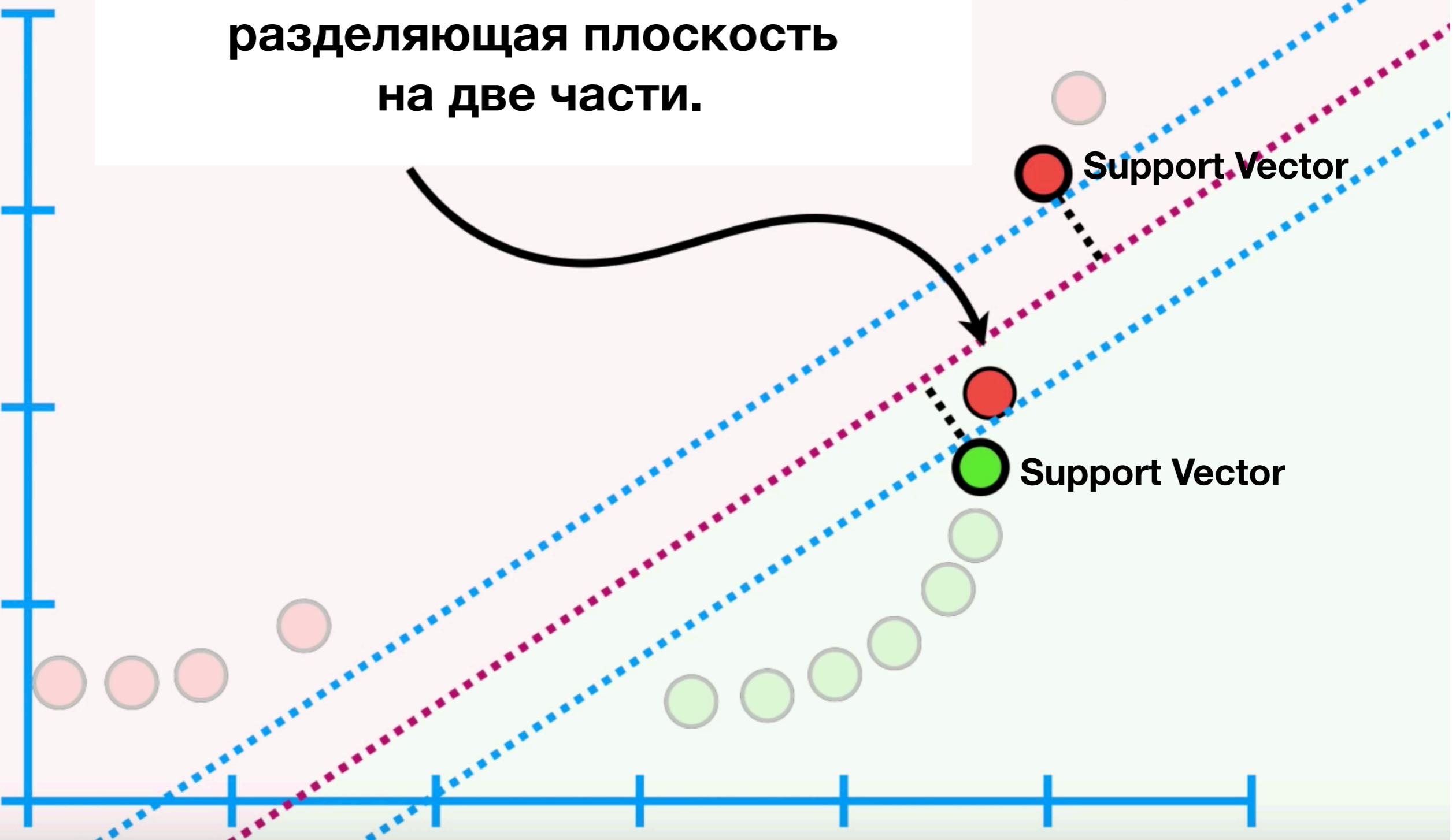


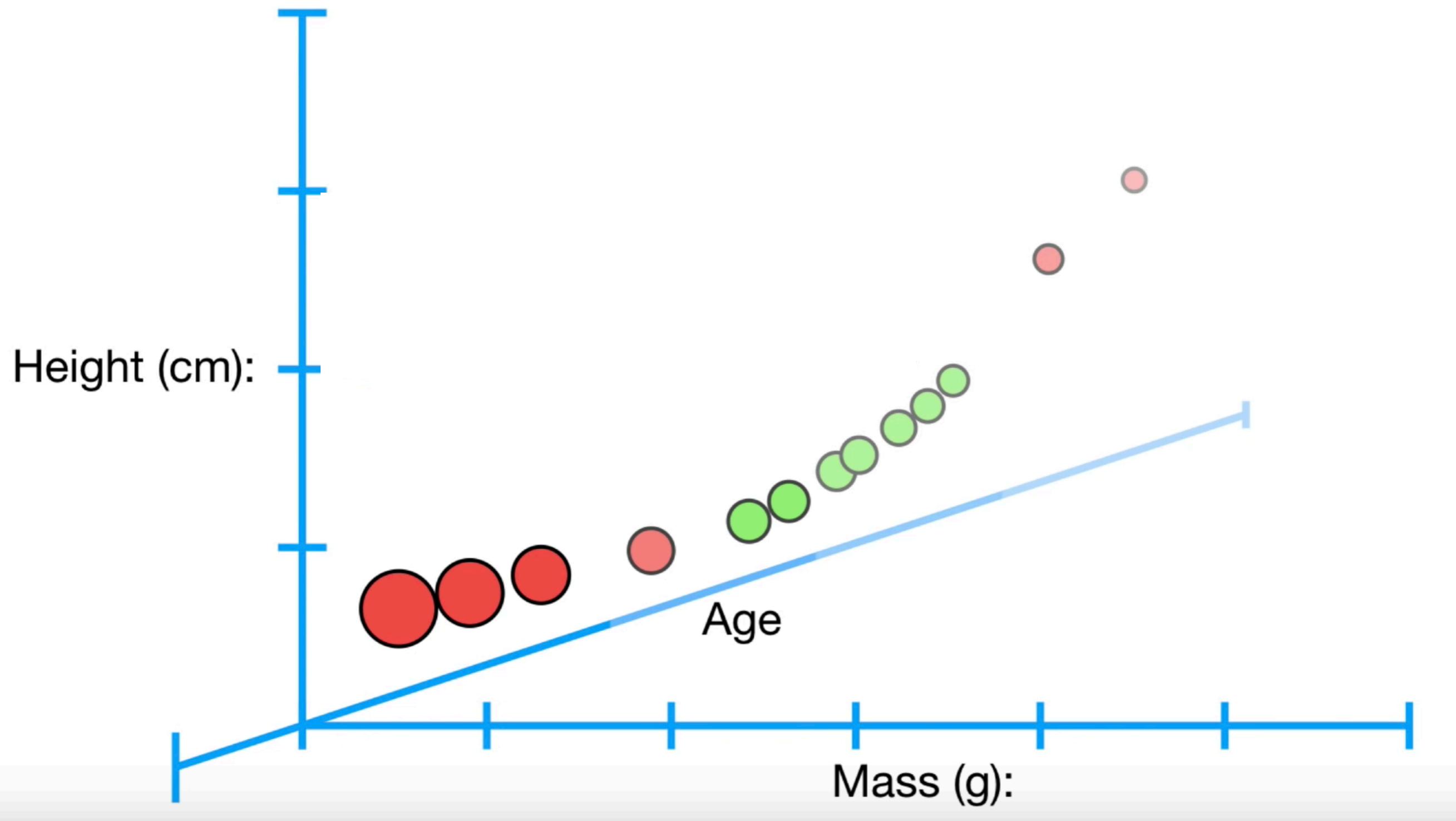
**Margin у такого классификатора называется
Soft Margin**
**Сам он называется Soft Margin Classifier =
Support VectorClassifier**



Как выглядит граница в этом случае?

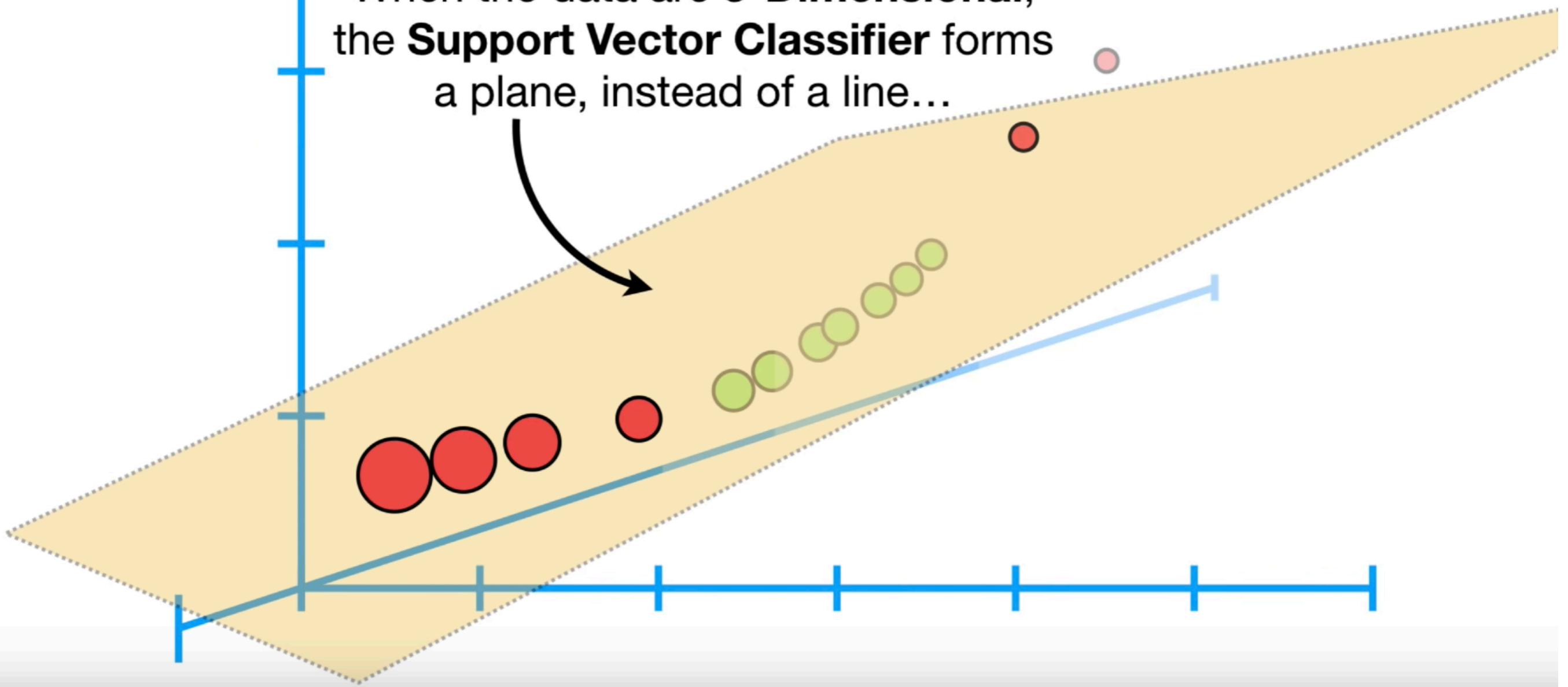
Как прямая, разделяющая плоскость на две части.





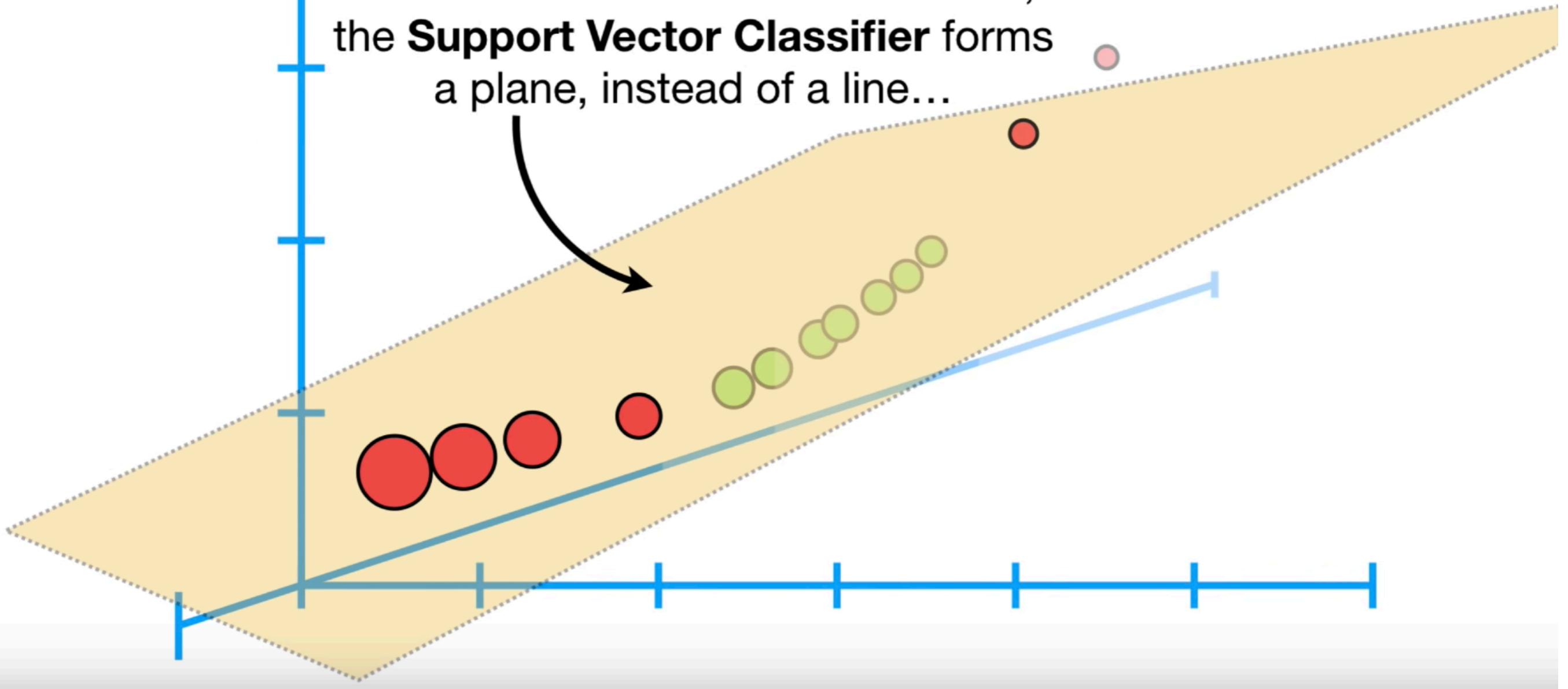
Как выглядит граница решений в этом случае?

When the data are **3-Dimensional**,
the **Support Vector Classifier** forms
a plane, instead of a line...



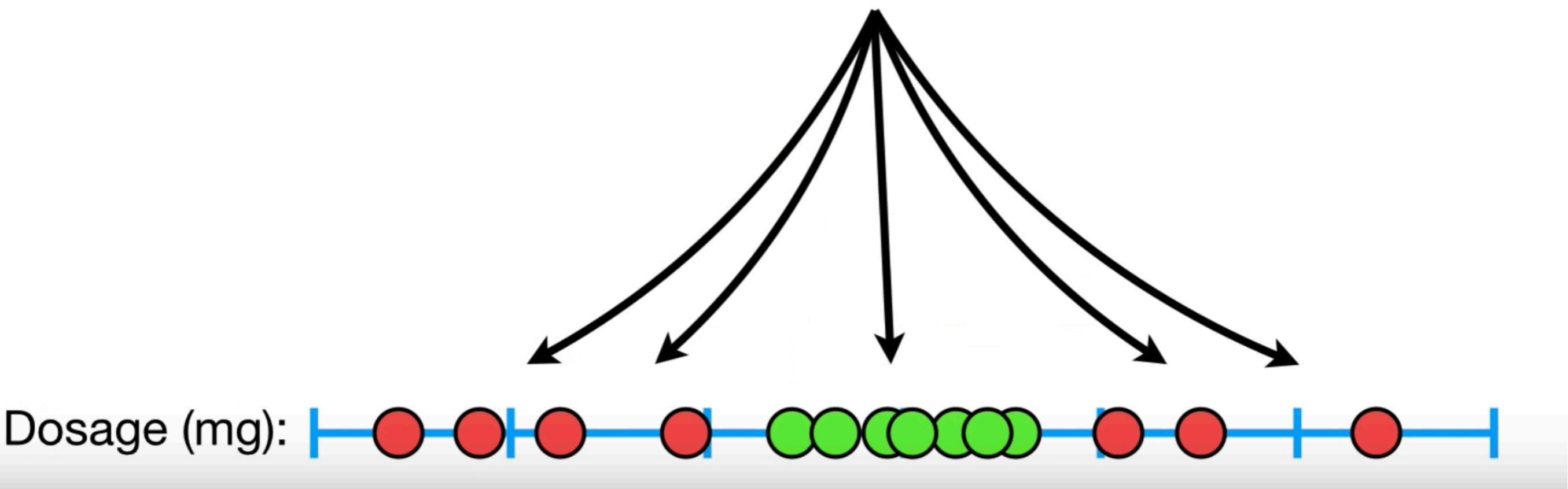
Плоскость

When the data are **3-Dimensional**,
the **Support Vector Classifier** forms
a plane, instead of a line...

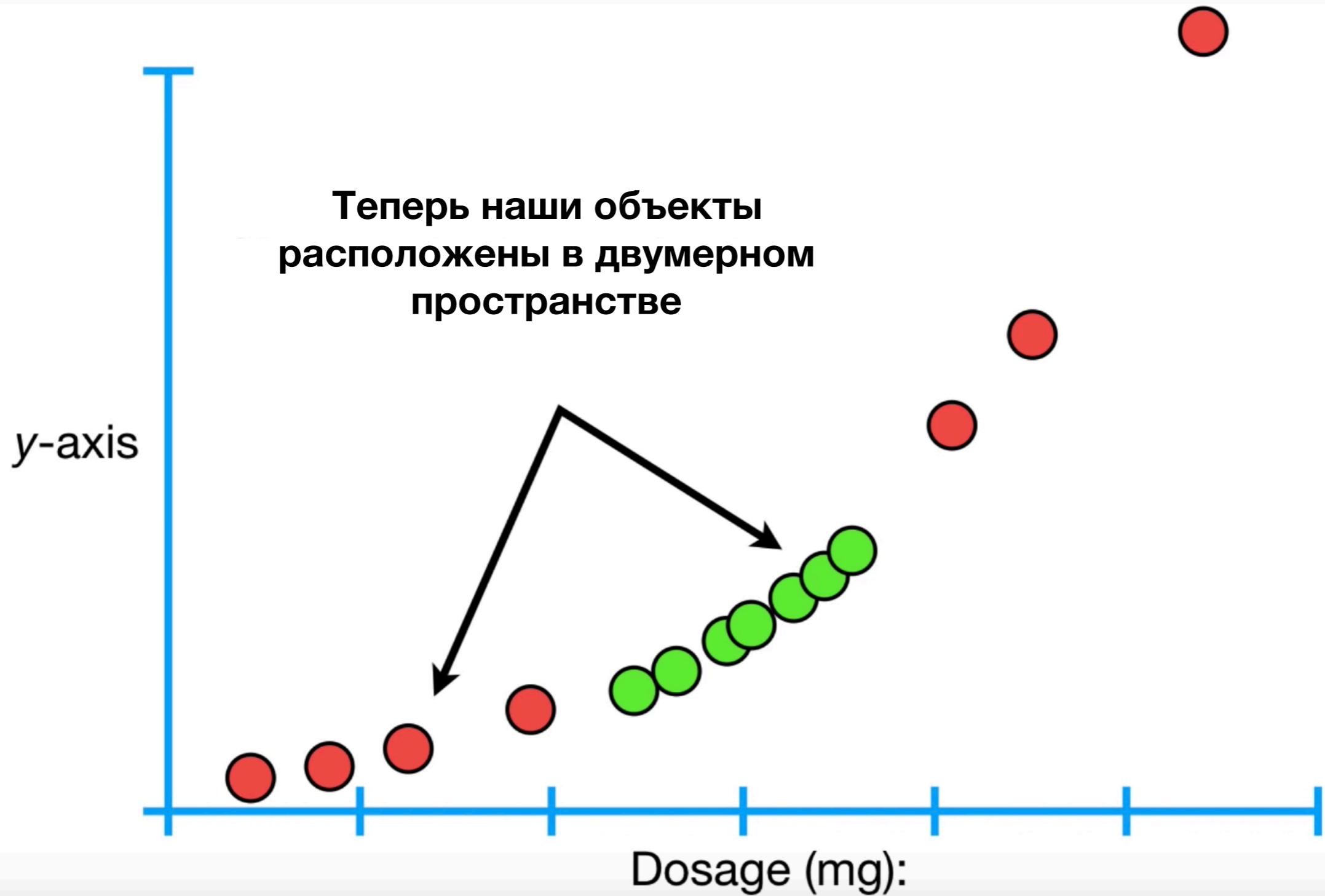


В общем случае в N -мерном пространстве разделяющей границей SVM будет $(N-1)$ -мерная гиперплоскость

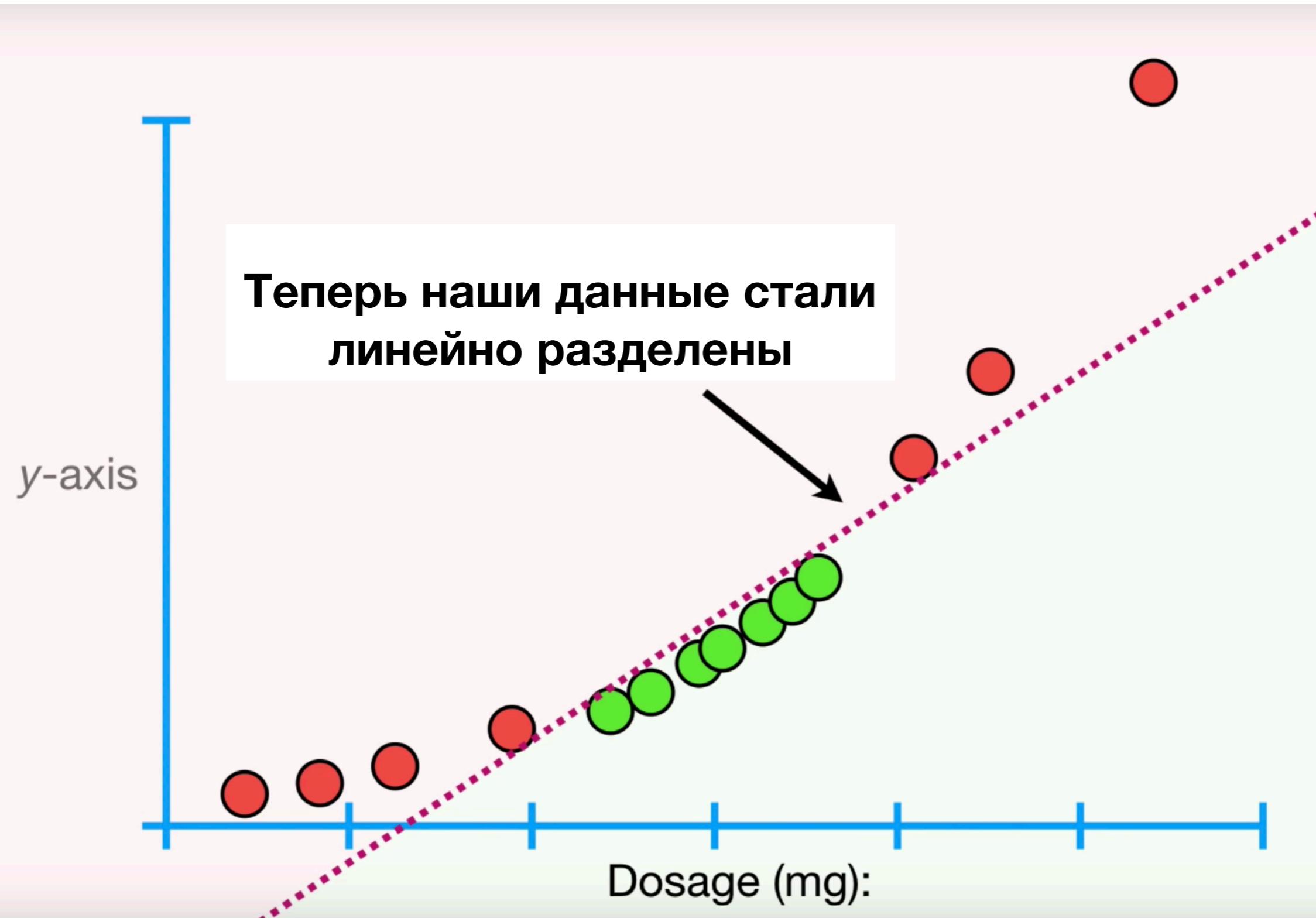
Что делать, если наши данные линейно неразделимы?



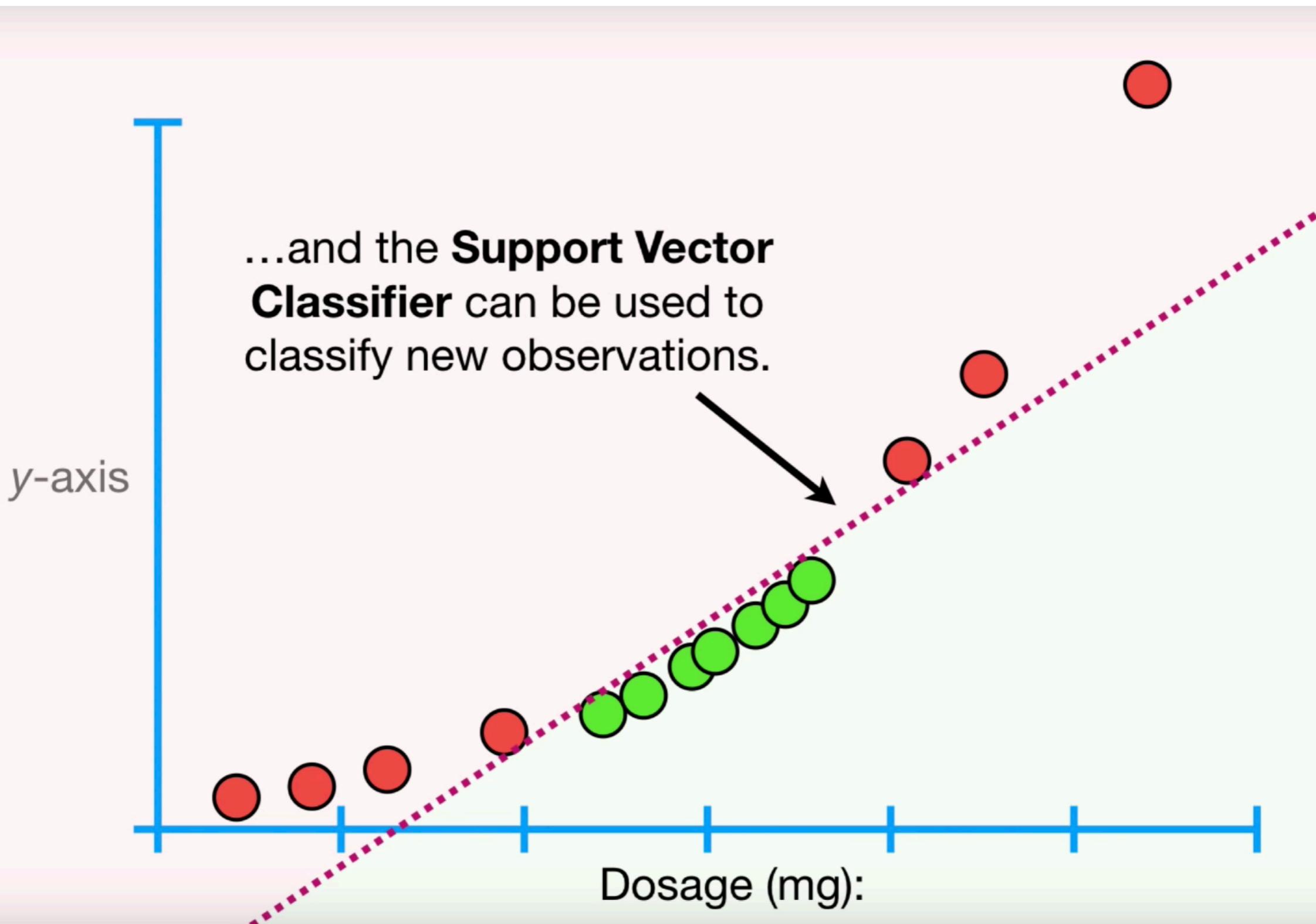
**Доза лекарства, нужен оптимум:
мало не поможет, много - убьет**



Добавим ось Y и отложим на этой оси
квадрат дозы лекарства

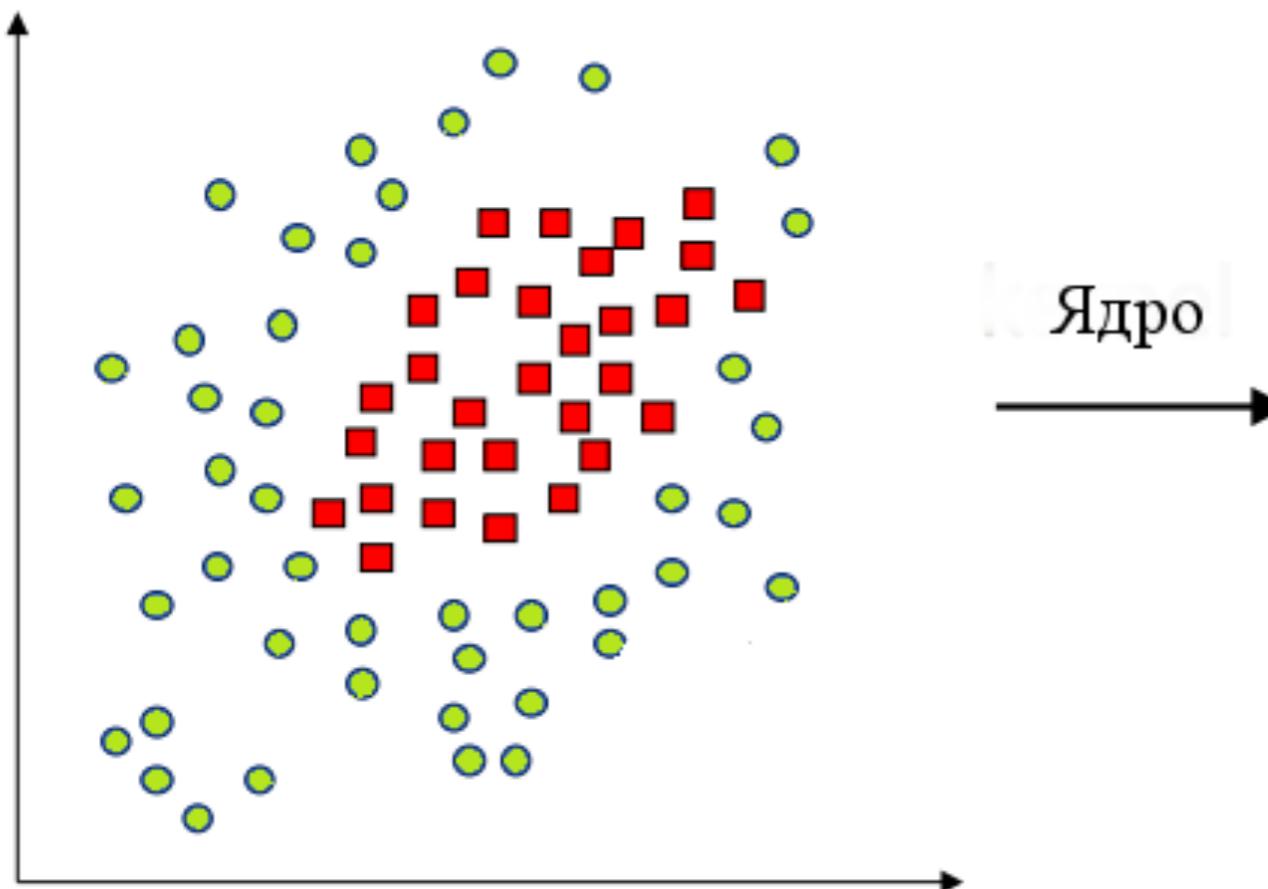


Получаем SVM - support vector machine

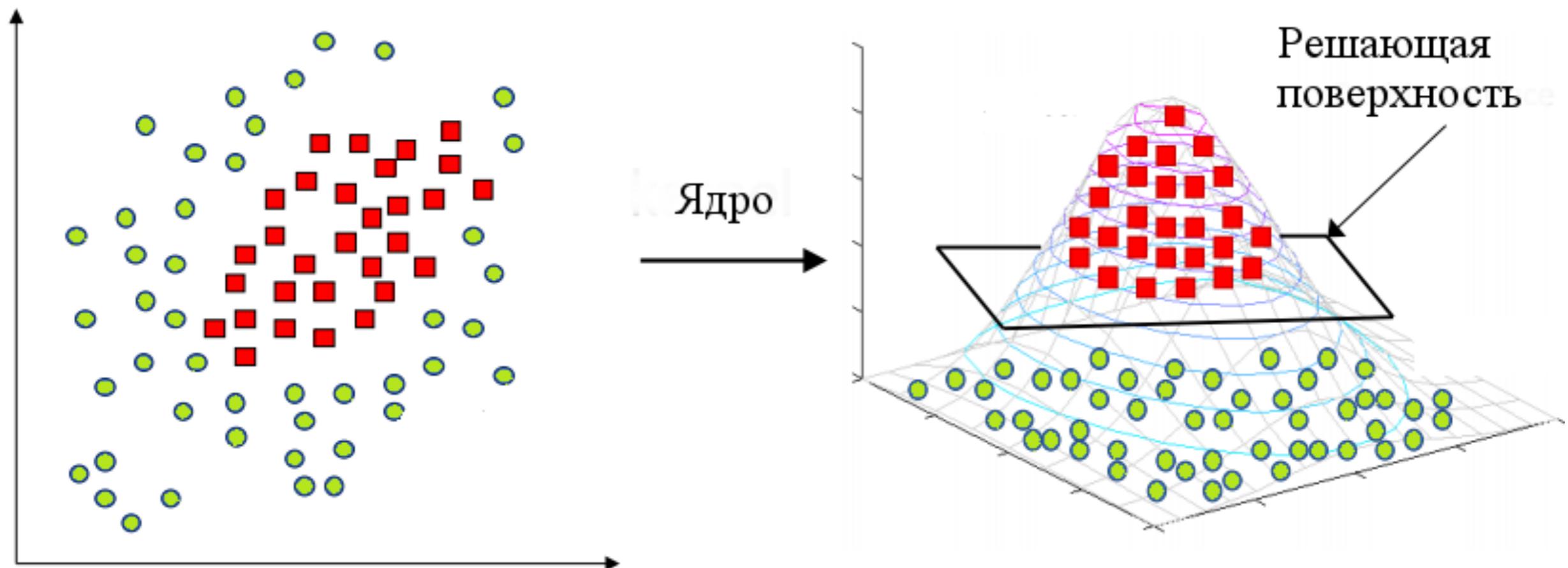


Таким образом, чтобы разделить данные, которые нельзя/ неудобно разделить в оригинальном пространстве , SVM, в общем случае, переводит их в пространство большей размерности, где они легко разделим

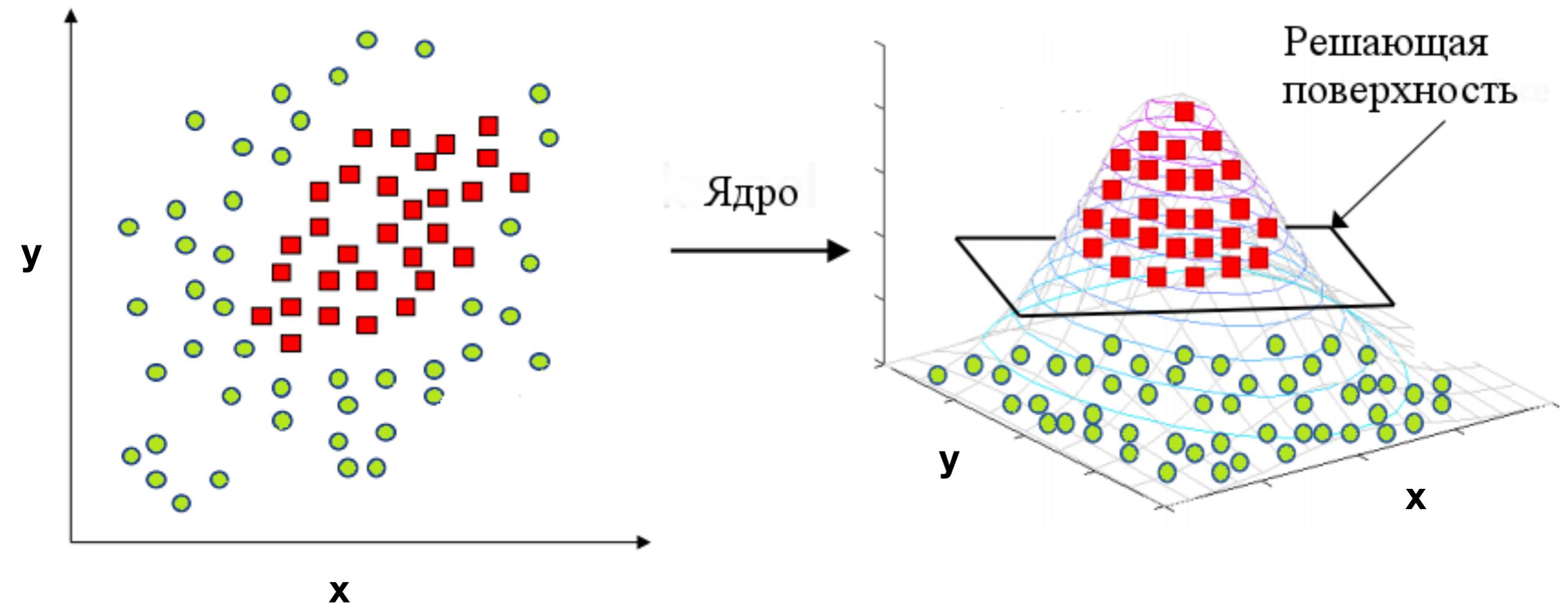
Как подбирать пространство, в которое нужно перевести наши точки?



Как подбирать пространство, в которое нужно перевести наши точки?



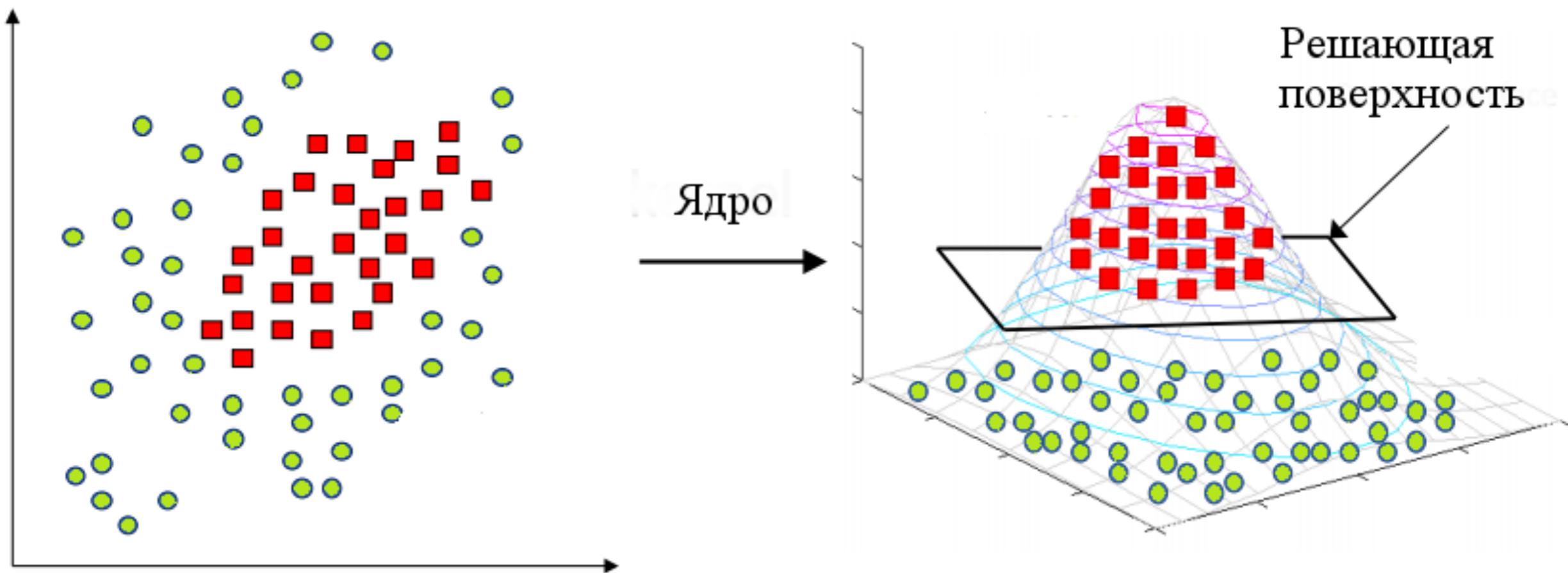
Как подбирать пространство, в которое нужно перевести наши точки?



Есть известные пространства, в которых можно искать

Kernel trick

Довольно сложно искать среди всех возможных пространств, особенно если у нас много признаков. Например, для двух признаков x и y полиномиальное пространство 2й степени: $\{x, y, x^2, y^2, x * y\}$



Число признаков сильно увеличивается при таких переходах

Kernel trick

На самом деле при обучении SVM нам нужны не ПРИЗНАКИ объектов, а их схожесть, которая считается через скалярное произведение признаков

$$\vec{a} \{1; 7; 9\}$$

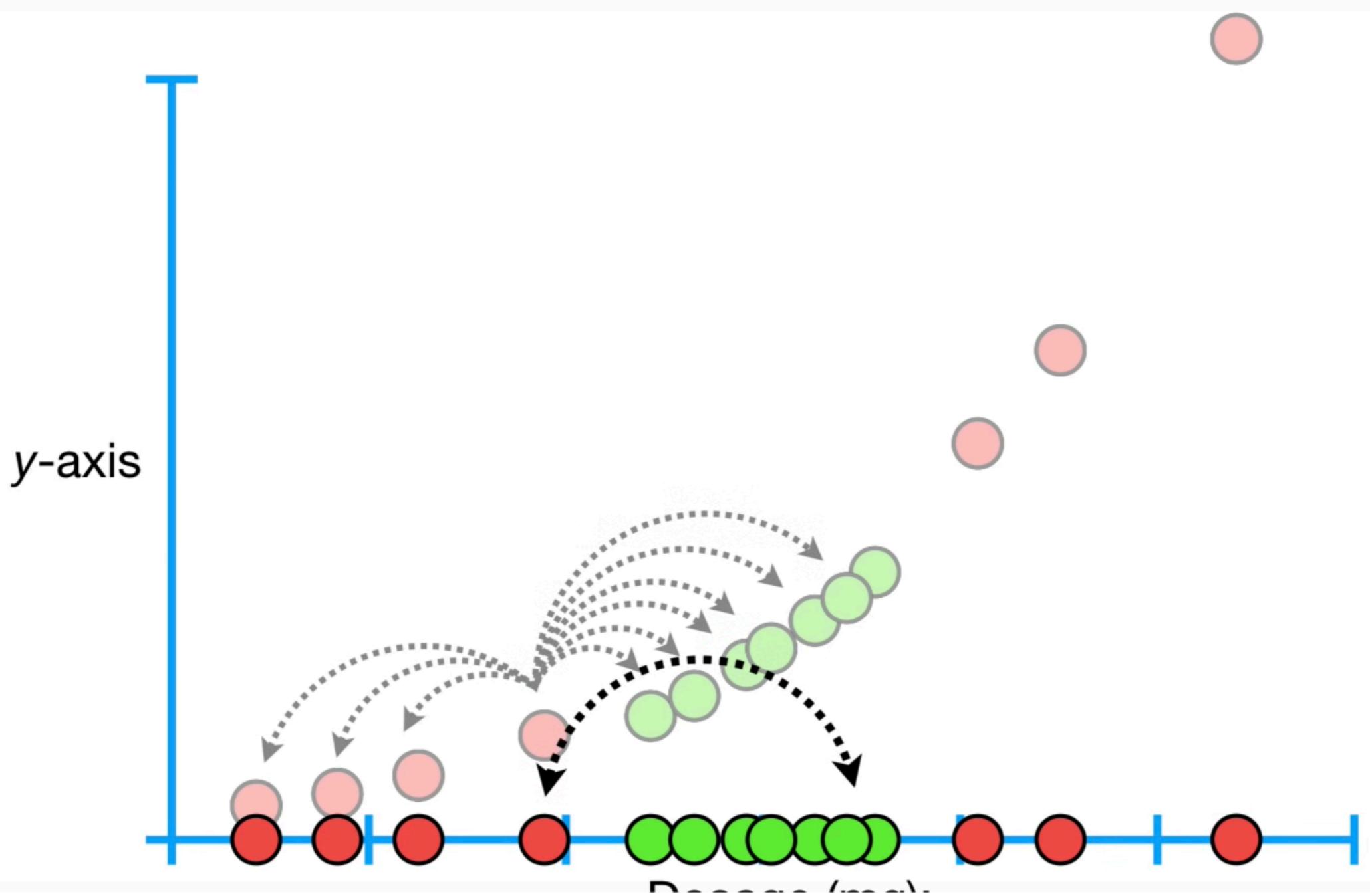
$$\vec{b} \{-2; 4; 0\}$$

$$\vec{a} \cdot \vec{b} = x_1x_2 + y_1y_2 + z_1z_2$$

$$\vec{a} \cdot \vec{b} = 1 \cdot (-2) + 7 \cdot 4 + 9 \cdot 0 = 26$$

Kernel trick

На самом деле при обучении SVM нам нужны не ПРИЗНАКИ объектов, а их схожесть, которая считается через скалярное произведение признаков



Kernel trick

На самом деле при обучении SVM нам нужны не ПРИЗНАКИ объектов, а их схожесть, которая считается через скалярное произведение признаков

$$\vec{a} \{1; 7; 9\}$$

$$\vec{b} \{-2; 4; 0\}$$

$$\vec{a} \cdot \vec{b} = x_1x_2 + y_1y_2 + z_1z_2$$

$$\vec{a} \cdot \vec{b} = 1 \cdot (-2) + 7 \cdot 4 + 9 \cdot 0 = 26$$

И есть функции, которые умеют считать скалярное произведение признаков объектов в новом пространстве, не переводя объекты в новое пространство (и не вычисляя признаков в нем)

Support vector machines

- Common kernel functions for SVM

- linear

$$k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1 \cdot \mathbf{x}_2$$

- polynomial

$$k(\mathbf{x}_1, \mathbf{x}_2) = (\gamma \mathbf{x}_1 \cdot \mathbf{x}_2 + c)^d$$

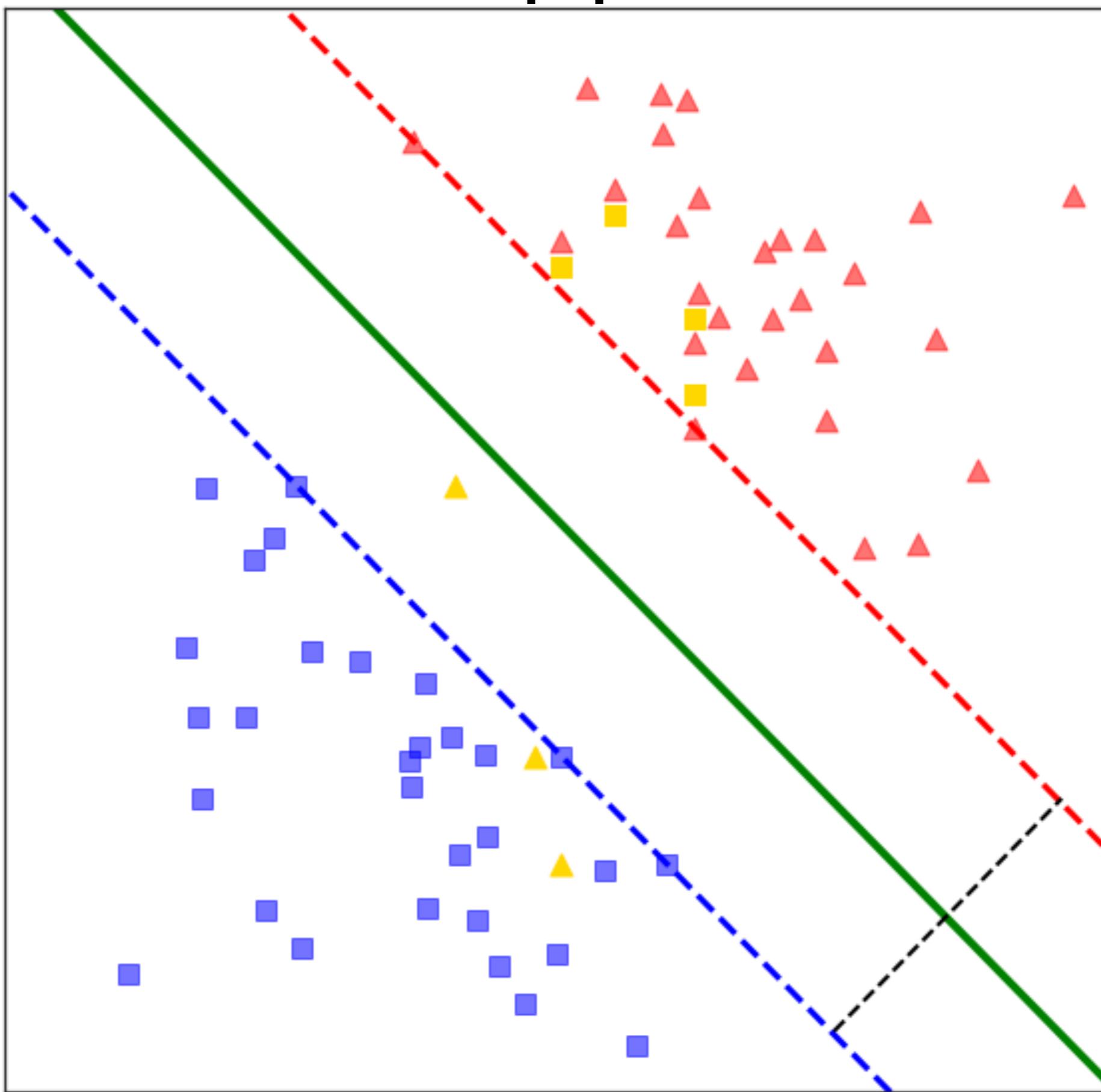
- Gaussian or radial basis

$$k(\mathbf{x}_1, \mathbf{x}_2) = \exp(-\gamma \|\mathbf{x}_1 - \mathbf{x}_2\|^2)$$

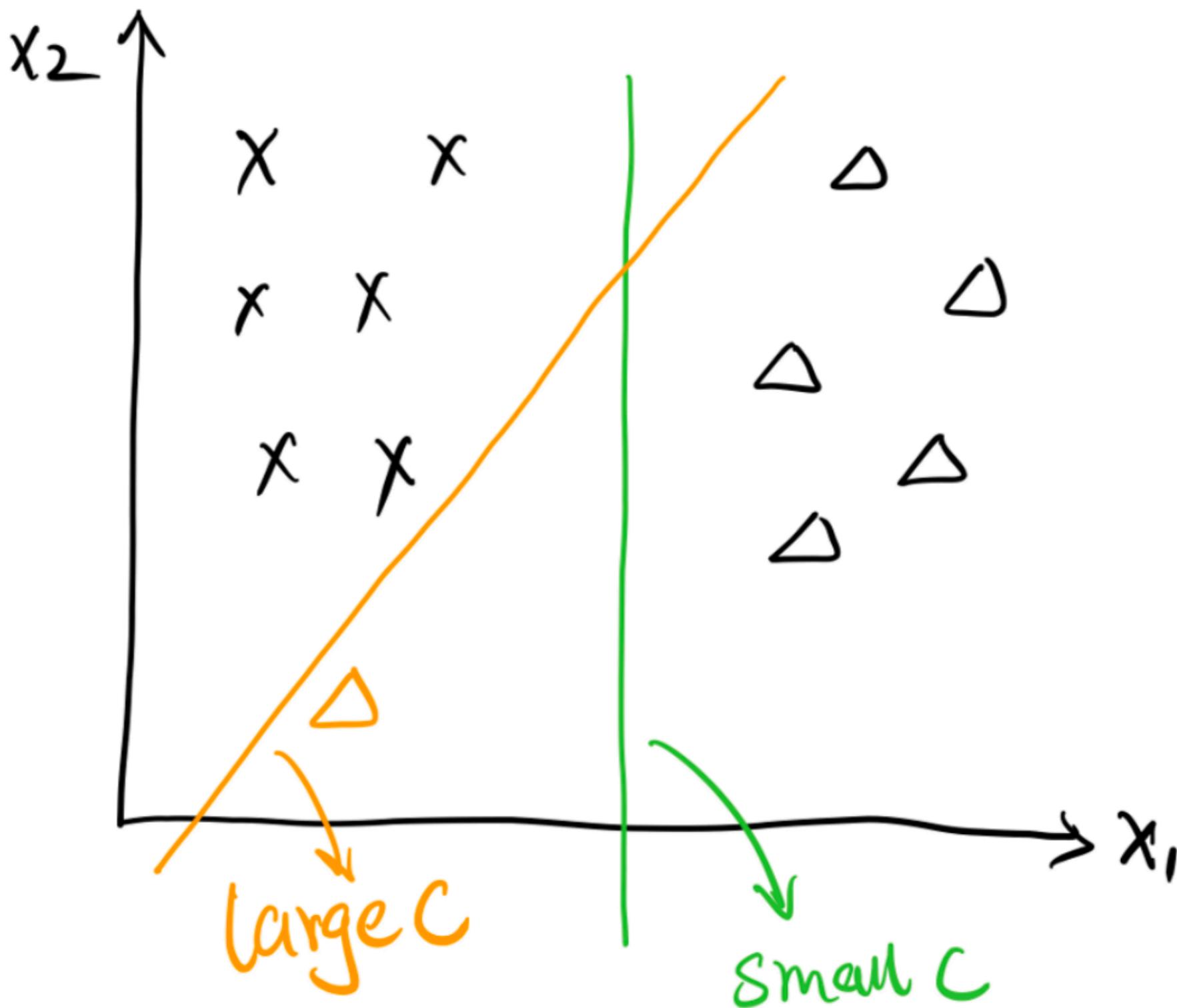
- sigmoid

$$k(\mathbf{x}_1, \mathbf{x}_2) = \tanh(\gamma \mathbf{x}_1 \cdot \mathbf{x}_2 + c)$$

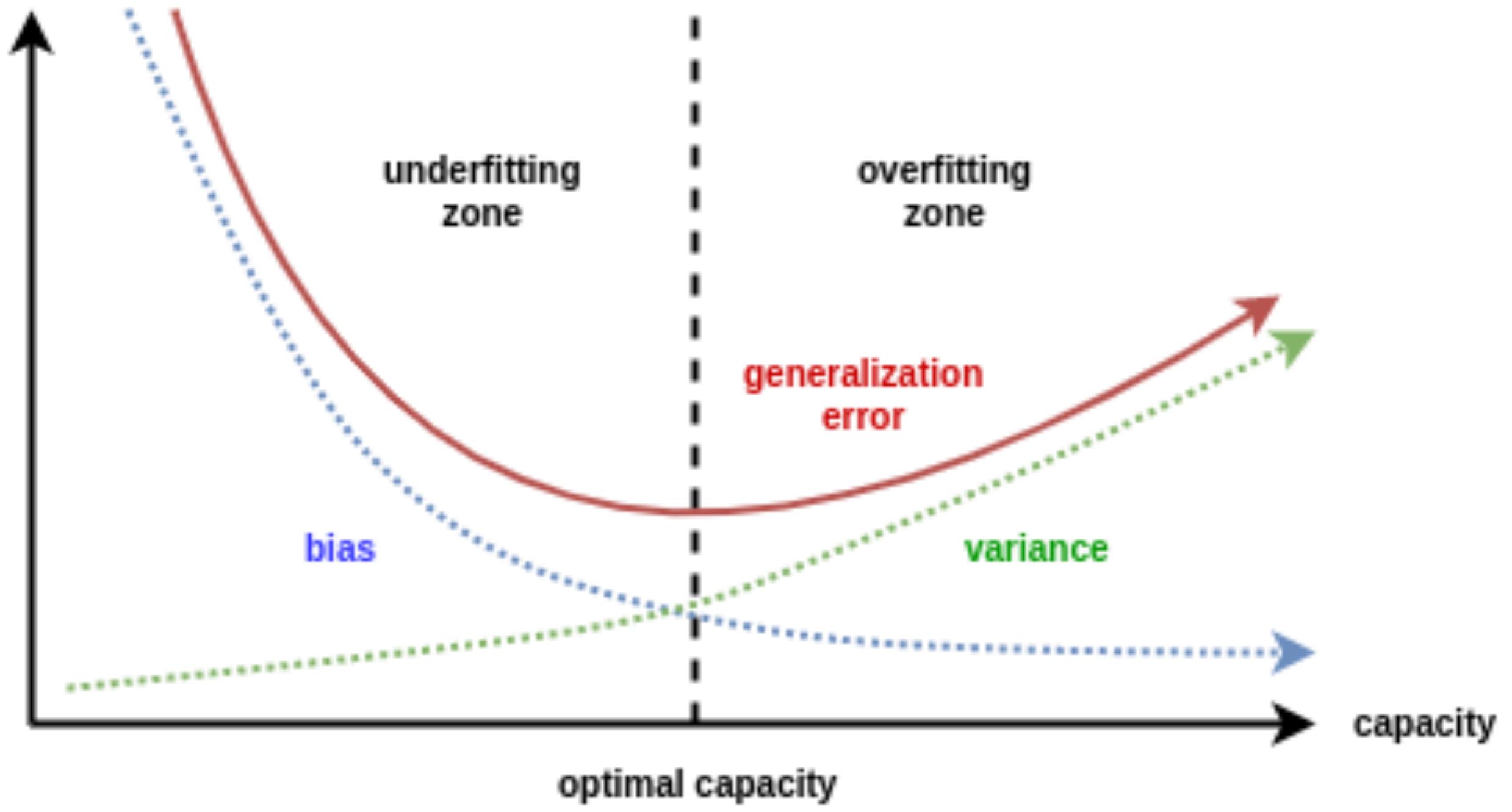
Как влиять на то, сколько объектов мы позволяем себе игнорировать?



Введем параметр С, штрафующий нас за неверно классифицированные объекты

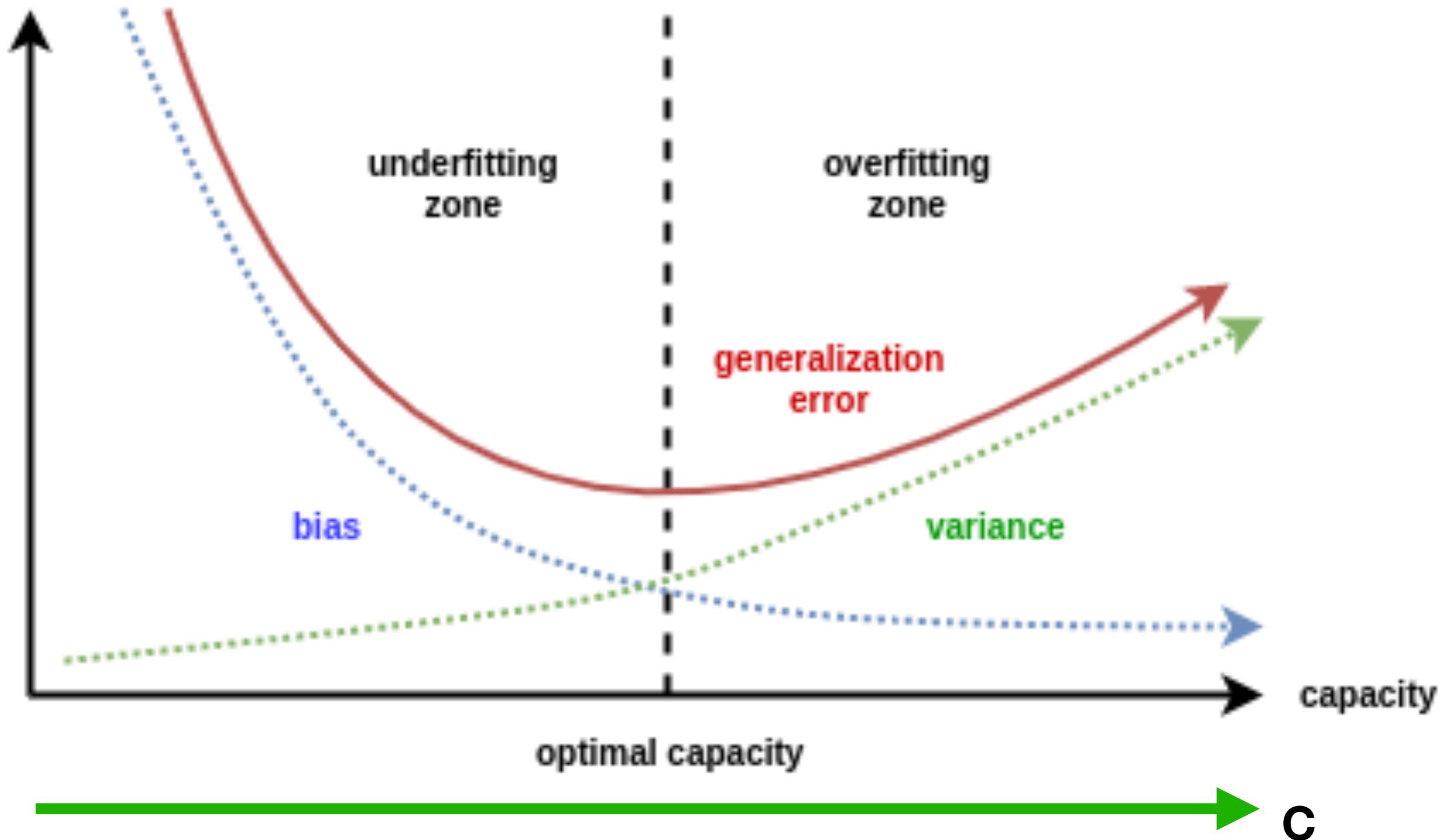


Bias-variance tradeoff



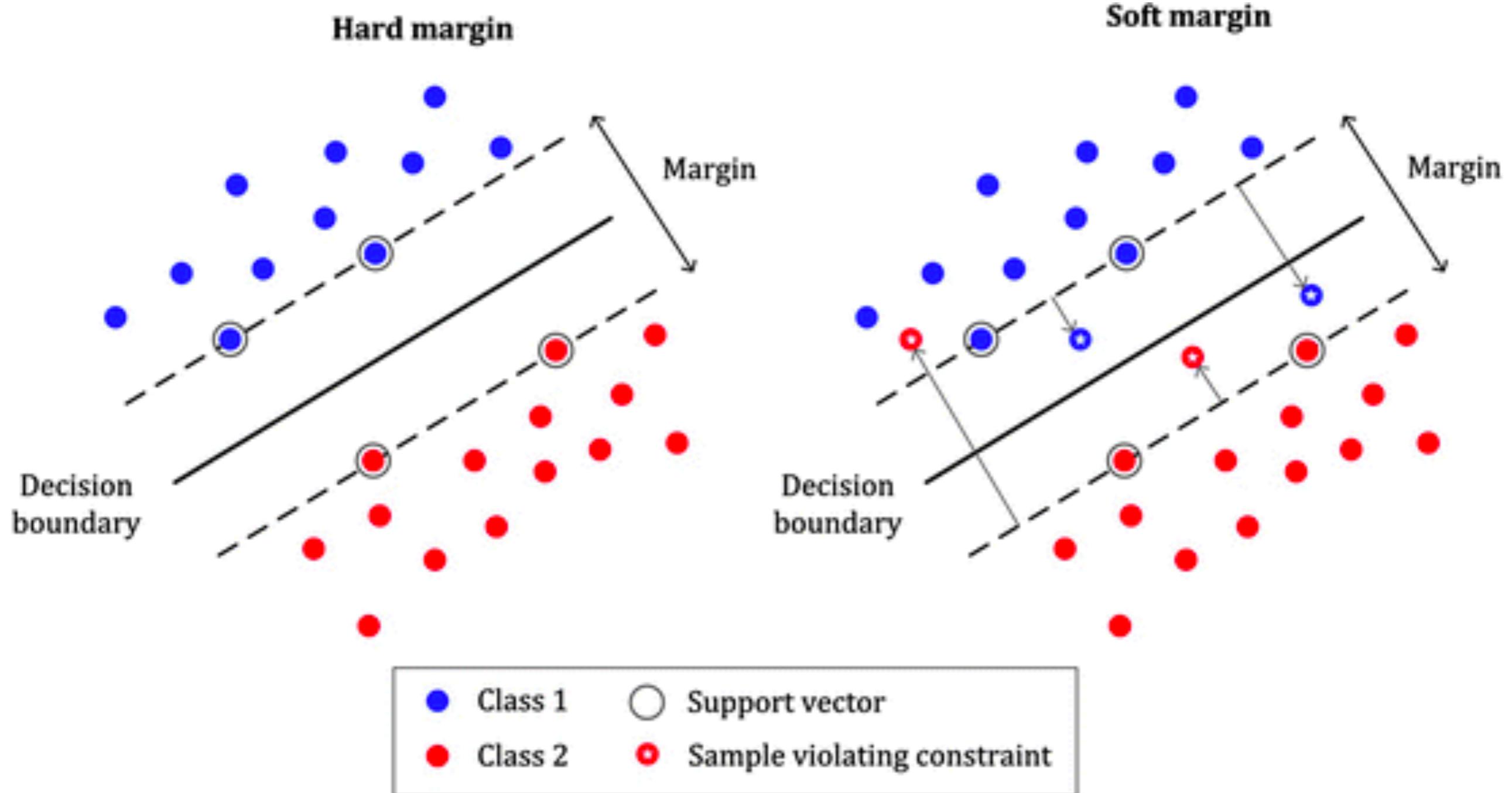
Как влияет С на bias и variance?

Bias-variance tradeoff



Как влияет С на bias и variance?

Почему support vectors?



Как предсказывать вероятность класса для объектов?

Decision function

$$\sum_{i \in SV} y_i \alpha_i K(x_i, x) + b,$$

x_i - точки из обучающей выборки, которые используются для построение разделяющей плоскости (support vectors)

x - точка, для которой предсказываем

Если больше 0, то класс 1, иначе класс 0

Как предсказывать вероятность класса для объектов?

- Никак
- Есть decision function, на основании значения которой решается, к какому классу принадлежит объект. Отражает то, насколько объект близок к разделяющей плоскости. Не вероятность
- Можно嘗試 approximirovatiy вероятность

Platt scaling

$$P(y = 1 \mid x) = \frac{1}{1 + \exp(Af(x) + B)}$$

Учим логистическую регрессию предсказывать вероятность класса на основе предсказания функции классификатора

Platt scaling

$$P(y = 1 \mid x) = \frac{1}{1 + \exp(Af(x) + B)}$$

Учим логистическую регрессию предсказывать вероятность класса на основе предсказания функции классификатора

Какие проблемы?

Platt scaling

$$P(y = 1 \mid x) = \frac{1}{1 + \exp(Af(x) + B)}$$

Учим логистическую регрессию предсказывать вероятность класса на основе предсказания функции классификатора

Какие проблемы?

Если учить на той же выборке, на которой учили SVM - получим проблемы

Platt scaling

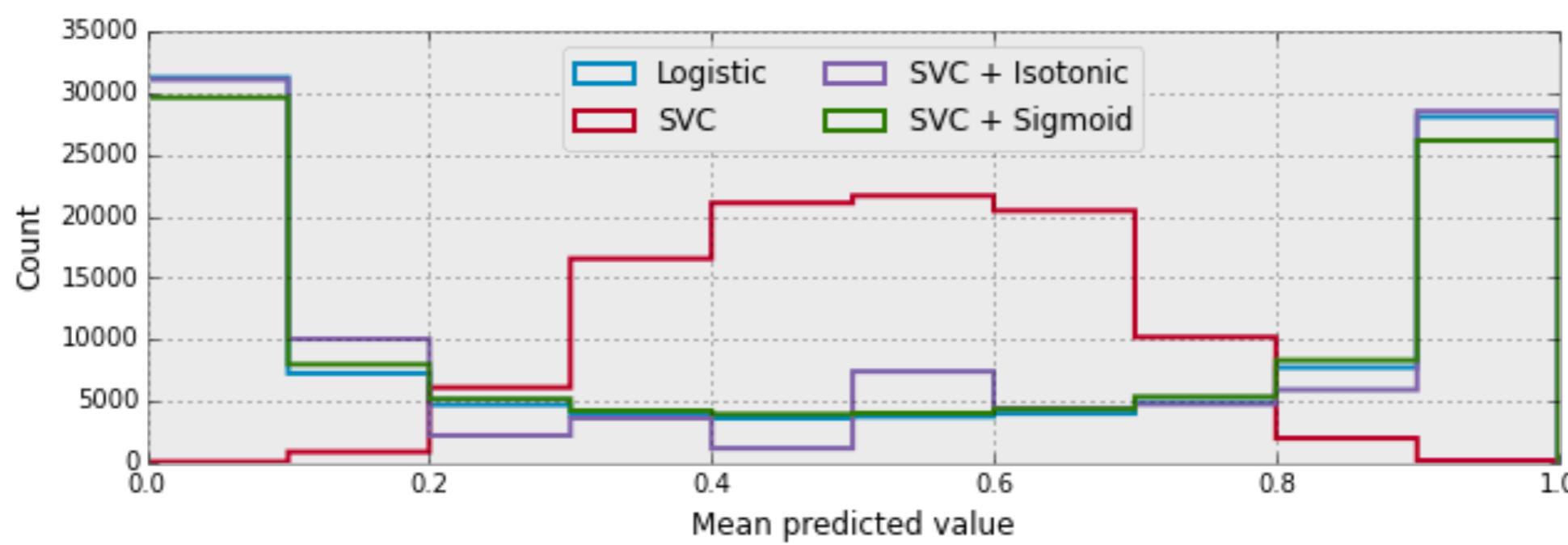
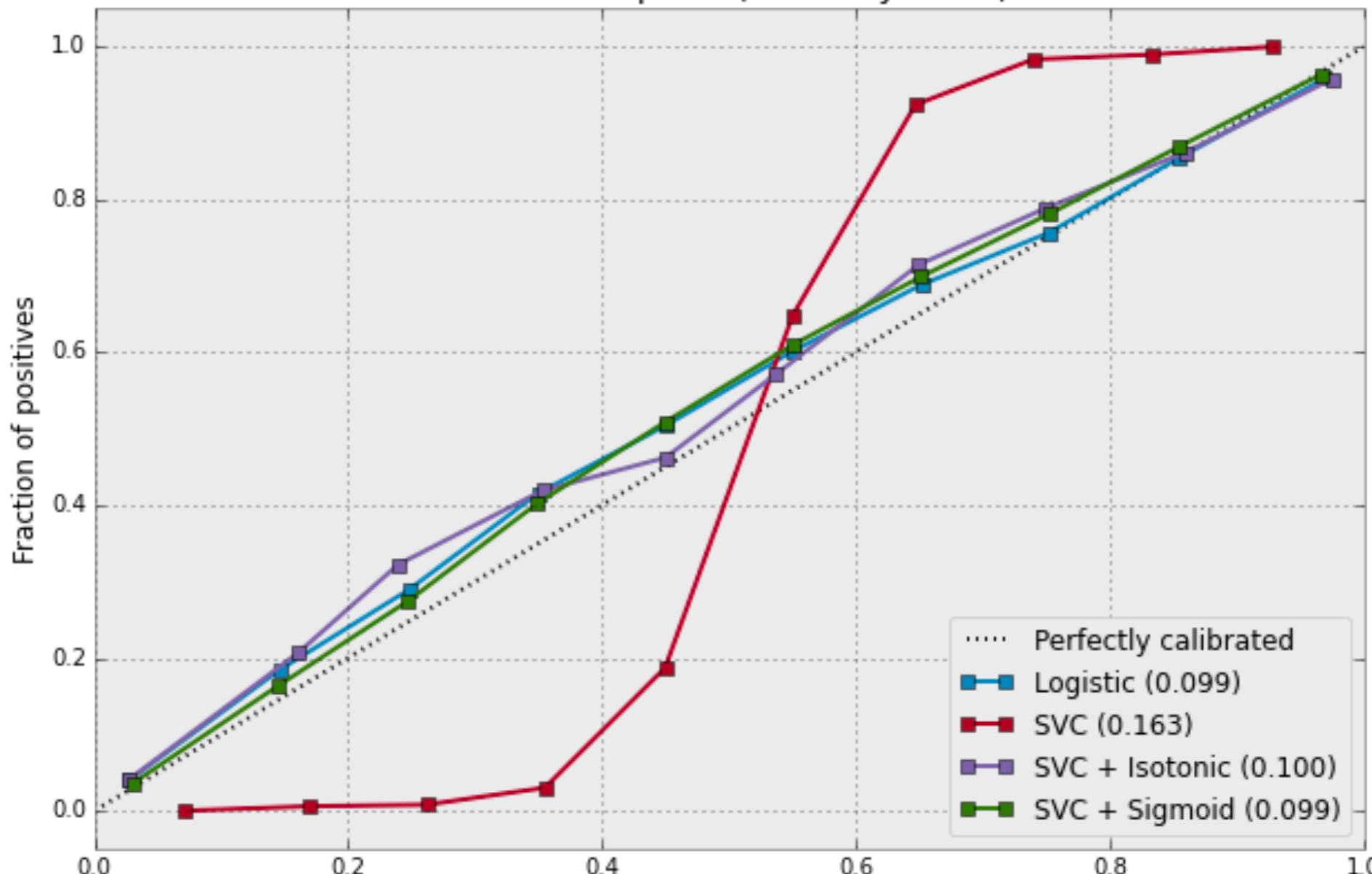
$$P(y = 1 \mid x) = \frac{1}{1 + \exp(Af(x) + B)}$$

Учим логистическую регрессию предсказывать вероятность класса на основе предсказания функции классификатора

Какие проблемы?

Разбиваем выборку на много частей. Учим SVM на одних, учим регрессию на других. И так много раз. Потом вероятности усредняем

Calibration plots (reliability curve)



Как растет качество SVM с ростом размера выборки?

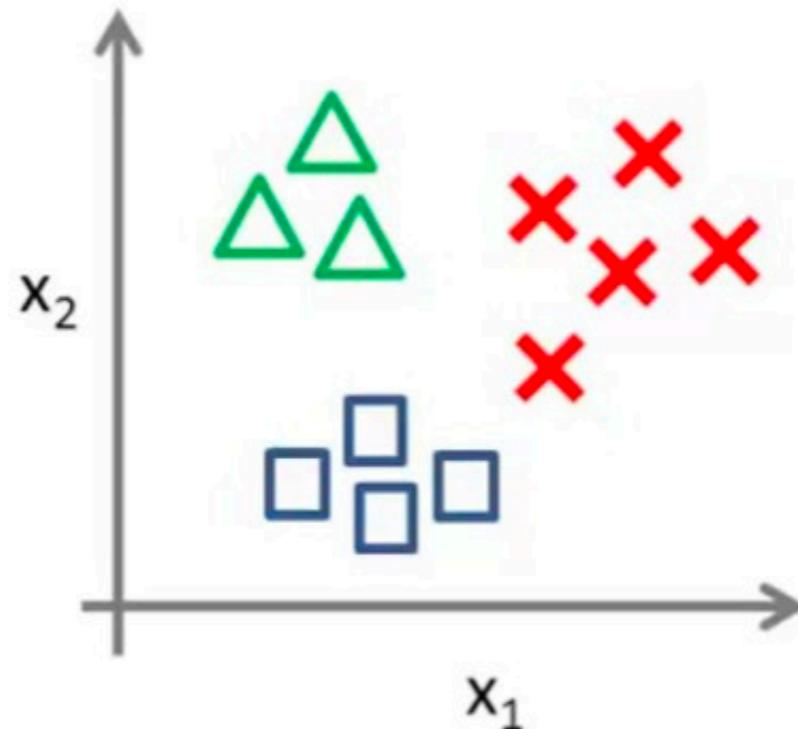
- Неправильно сказать, что чем линейно или около того
- SVM учитывает только опорные вектора, потому вполне возможна ситуация, когда даже сильное увеличение размера выборки на качестве никак не оказывается

Как предсказывать для нескольких классов?

- One vs rest (тренируем по классификатору на задачу “отличи класс К от остальных”) - далее выбираем наибольшую вероятность
- One vs one (тренируем по классификатору на задачу “отличи класс К от класса М”) - выбираем класс, для которого больше всего голосов

One vs rest

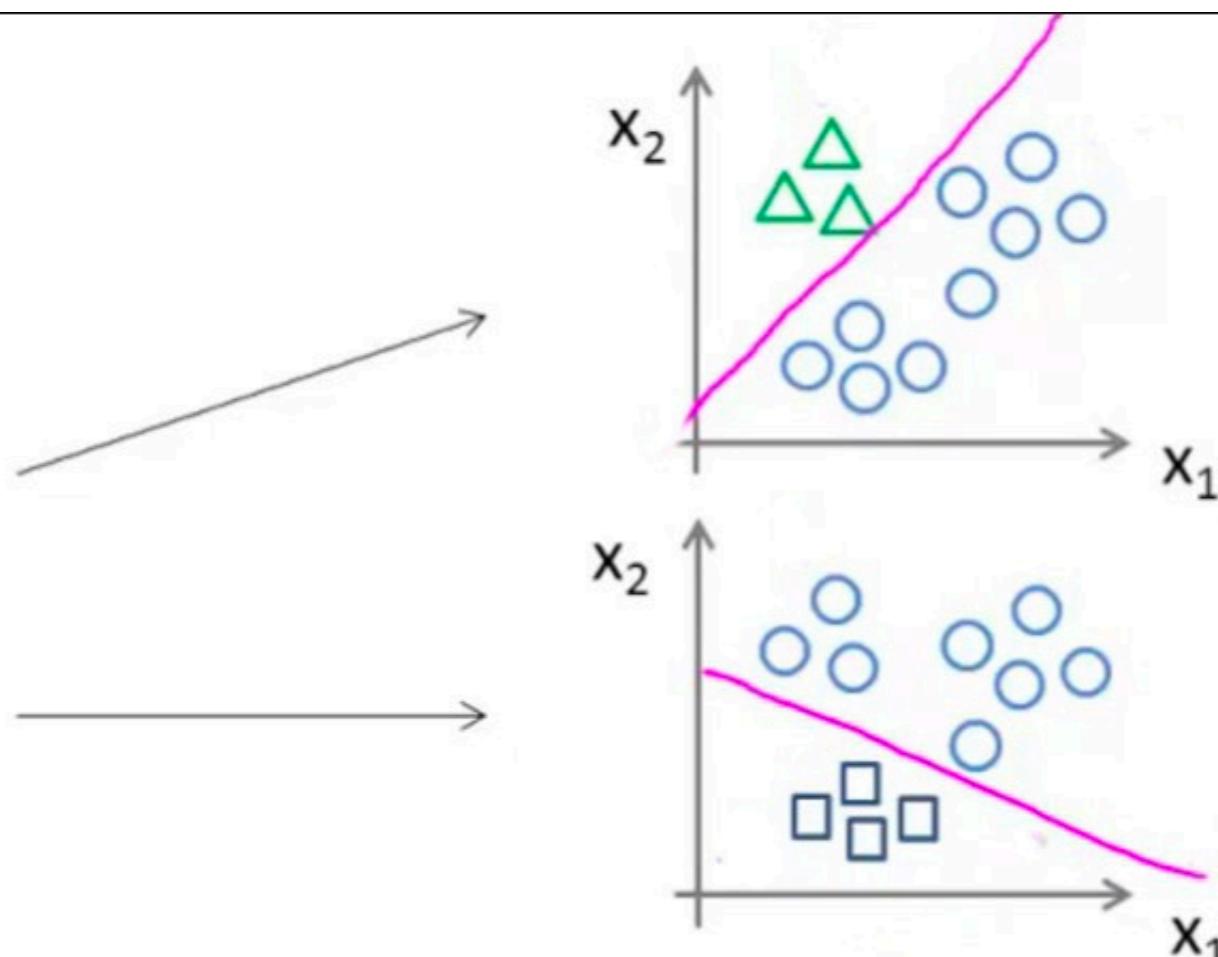
One-vs-all (one-vs-rest):



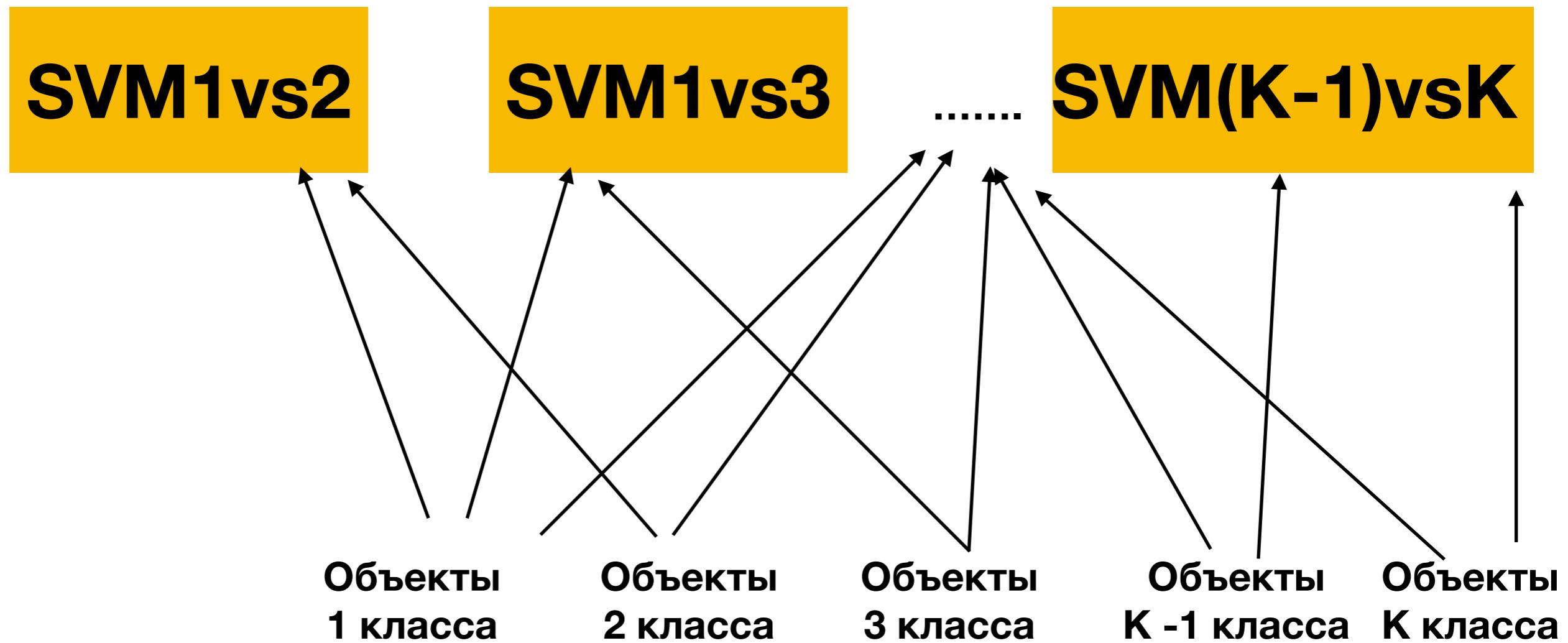
Class 1:

Class 2:

Class 3:



One vs one

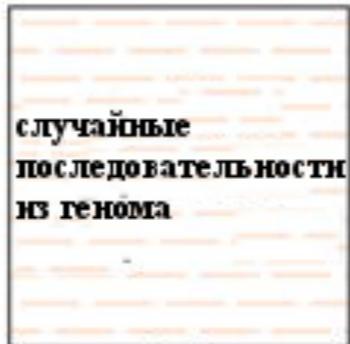


deltaSVM

Положительные примеры



Отрицательные примеры



Изучаемая однонуклеотидная замена

Энхансер

Ген

Дикий тип TCTCTGCAA**C**AAAGACAGA...
Замена ...TCTCTGCAA**A**AAAGACAGA...

1) обучение gkm-SVM

Словарь регуляторных последовательностей

Все уникальные 10-меры	веса SVM	
ATGACTCATC	3.275	положительная
ATGAGTCATC	3.147	влияние на
ATCATGTGAC	3.091	экспрессию
GTCACATGAC	2.992	
⋮	⋮	
ACGAGAAACA	0.0002	нейтральное
ACTATAACCA	0.0001	влияние на
ATTGCTAACGC	-0.0001	экспрессию
GGATAAAATA	-0.0001	
⋮	⋮	
CAGGTGTGAG	-1.171	отрицательное
ATCACACCTG	-1.183	влияние на
ACACACCTGT	-1.253	экспрессию
AATCCAGGTG	-1.282	

вычисление deltaSVM

10-меры дикого типа	Вес	10-меры варианта	Вес	Разница
TCTCTGCAA C	0.012	TCTCTGCAA A	0.298	0.286
CTCTGCAA CA	0.082	CTCTGCAA AA	0.104	0.021
TCTGCAA CAA	0.280	TCTGCAA AAA	0.114	-0.166
CTGCAA AAAA	0.330	CTGCAA AAAAA	-0.029	-0.359
TGCAA AAAAG	0.441	TGCAA AAAAG	-0.025	-0.466
GCAA CAAAGA	0.784	GCAA AAAAGA	0.008	-0.776
CAA CAAAGAC	1.031	CAA AAAAGAC	0.109	-0.922
AAC AAAGACA	0.545	A AAAAGACA	-0.453	-0.998
A CAAAGACAG	0.671	A AAAAGACAG	-0.516	-1.187
CAAAGACAGA	-0.036	AAAAGACAGA	-0.478	-0.442

deltaSVM = -5.007