

Лекция 6

Метрики классификации. Многоклассовая классификация. Нелинейные модели классификации.

Юлия Конюшенко

тг: @ko_iulia

koniushenko.iun@phystech.edu

Лекция 1.

ML

→ обучение с учителем

обучение без учителя

- категоризация
- снижение размерн.
- визуализация

- классифр.
- регрессия
- ранжирование

Лекция 2.

1) линейная регрессия

Обучение \equiv минимизация MSE

почему именно MSE? → есть вероятностная интерпретация

2) градиентного спуск

$$a(x) = (w, x)$$

$$MSE$$

$$w^{(k)} = w^{(k-1)} - \eta \triangleright Q(w^{(k-1)})$$

стochastic \rightarrow по 1 объекту
mini-batch \rightarrow по батчу

Лекция 3. Метрики качества и функционалы

$$MSE \rightarrow RMSE \rightarrow R^2$$

ошибки

$$MAE \rightarrow MSLE \rightarrow MAPE \rightarrow SMAPE$$

квантильная регрессия

онлайн / офлайн / бизнес метрики

Признаки переобучения, регуляризация

- разница качества на train/test
- большие веса

$$\begin{aligned} \cdot L_2: & + \sum w_i^2 \\ \cdot L_1: & + \sum |w_i| \end{aligned}$$

среднее

+ оптимумов MSE и

MAE

медиана

Лекция 4.

1) Оценивание качества модели

- отложенная выборка
- кросс-валидация

K-fold

complete

leave-one-out

K = 5

K = 7

K = 10

2) Способы кодирования категориальных признаков

- one-hot encoding

- стеммы
 - сжатие
 - подсет на отложенной выборке

3) Линейные модели классификации

$$a(x, w) = \text{sign} \left(\sum_{j=1}^n w_j x_j \right)$$

$$Q(a, X) = \frac{1}{n} \sum_{i=1}^n [M_i < 0] \rightarrow \min, \text{ где } M_i = y_i \cdot (w, x_i) - \text{отступ}$$

Для оптимизации используют верхние оценки эмпирического риска
 Разные ф-ии потерь соответствуют различным типам моделей

Оптимизируются градиентным спуском

Лекция 5

1. логистическая перспектива - это линейный классификатор!

$$a(x, w) = \tilde{\sigma}(w^T x), \quad \tilde{\sigma}(z) = \frac{1}{1 + e^{-z}}$$

лог. пот.

$$Q(w) = - \sum_{i=1}^e ([y_i = +1] \log(a(x_i, w)) + [y_i = -1] \log(1 - a(x_i, w)))$$

$$L(a, x) = \sum_{i=1}^e \log(1 + e^{-y_i(a(x_i, w))})$$

! логистическая функция потерь корректно предсказывает вероятности

2. Перцептрон

$$a(x, w) = [(w, x) > 0]$$

3. Метод опорных векторов

→ линейно разделимая выборка
→ линейно неразделимая выборка
2 задачи оптимизационные

безусловное задание оптимизационное:

$$\frac{1}{2} \|w\|^2 + C \sum_{i=1}^e \max(0, 1 - y_i((w, x_i) + w_0)) \rightarrow \min_{w, w_0}$$

коопротивное между линейной разделяющей поисковой и минимизирующей суммарной ошибкой

МЕТРИКИ КАЧЕСТВА

МЕТРИКИ КАЧЕСТВА КЛАССИФИКАЦИИ

- Accuracy – доля правильных ответов:

$$\text{accuracy}(a, X) = \frac{1}{n} \sum_{i=1}^n [a(x_i) = y_i]$$

Когда работает плохо?

• Несбал. вид
99 %
100 % ↑
0,99
1 %

МЕТРИКИ КАЧЕСТВА КЛАССИФИКАЦИИ

- Accuracy – доля правильных ответов:

$$\text{accuracy}(a, X) = \frac{1}{n} \sum_{i=1}^n [a(x_i) = y_i]$$

*Недостаток: при сильно несбалансированной выборке
не отражает качество работы алгоритма*

МАТРИЦА ОШИБОК

Матрица ошибок (confusion matrix):

реальные
ответы

		Actual Value	
		positives	negatives
Predicted Value	positives	TP True Positive	FP False Positive
	negatives	FN False Negative	TN True Negative

→ мои предсказания

МЕТРИКИ КАЧЕСТВА КЛАССИФИКАЦИИ: PRECISION, RECALL

- Precision (точность):

$$Precision(a, X) = \frac{TP}{TP + FP}$$

Показывает, насколько можно доверять классификатору при $a(x) = +1$.

объекты,
 $a(x) = +1$ и
действ. +8

все объекты,
которым
классифи-
ор. +1

PRECISION: ПРИМЕР

Модель $a_1(x)$:

$$\text{precision}(a_1, X) = 0.8$$

Модель $a_2(x)$:

$$\text{precision}(a_2, X) = 0.96$$

		$y = 1$ Могут вернуть	$y = -1$ Не могут вернуть	
$a(x) = 1$ Получили кредит	80	20		
	20	80		

		$y = 1$ Могут вернуть	$y = -1$ Не могут вернуть	
$a(x) = 1$ Получили кредит	48	2		
	52	98		

МЕТРИКИ КАЧЕСТВА КЛАССИФИКАЦИИ: PRECISION, RECALL

- Precision (точность):

$$Precision(a, X) = \frac{TP}{TP + FP}$$

Показывает, насколько можно доверять классификатору при $a(x) = +1$.

- Recall (полнота):

$$Recall(a, X) = \frac{TP}{TP + FN}$$

$y = +1$ и $a(x) = +1$

все объекты, принадлежащие классу

Показывает, как много объектов положительного класса находит классификатор.

релевантен по запросу q». Благодаря большому дисбалансу, Accuracy dummy-классификатора, объявляющего все документы нерелевантными, будет близка к единице. Напомним, что $\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$, и в нашем случае высокое значение метрики будет обеспечено членом TN, в то время для пользователей более важен высокий TP.

Поэтому в случае асимметрии классов, можно использовать метрики, которые не учитывают TN и ориентируются на TP.

Если мы рассмотрим долю правильно предсказанных положительных объектов среди всех объектов, предсказанных положительным классом, то мы получим метрику, которая называется **точностью (precision)**

$$\text{Precision} = \frac{TP}{TP + FP}$$

Интуитивно метрика показывает долю релевантных документов среди всех найденных классификатором. Чем меньше ложноположительных срабатываний будет допускать модель, тем больше будет её Precision.

Если же мы рассмотрим долю правильно найденных положительных объектов среди всех объектов положительного класса, то мы получим метрику, которая называется **полнотой (recall)**

$$\text{Recall} = \frac{TP}{TP + FN}$$

Интуитивно метрика показывает долю найденных документов из всех релевантных. Чем меньше ложно отрицательных срабатываний, тем выше recall модели.

Например, в задаче предсказания злокачественности опухоли точность показывает, сколько из определённых нами как злокачественные опухолей действительно являются злокачественными, а полнота – какую долю злокачественных опухолей нам удалось выявить.

RECALL: ПРИМЕР

Модель $a_1(x)$:

$$\text{recall}(a_1, X) = 0.8$$

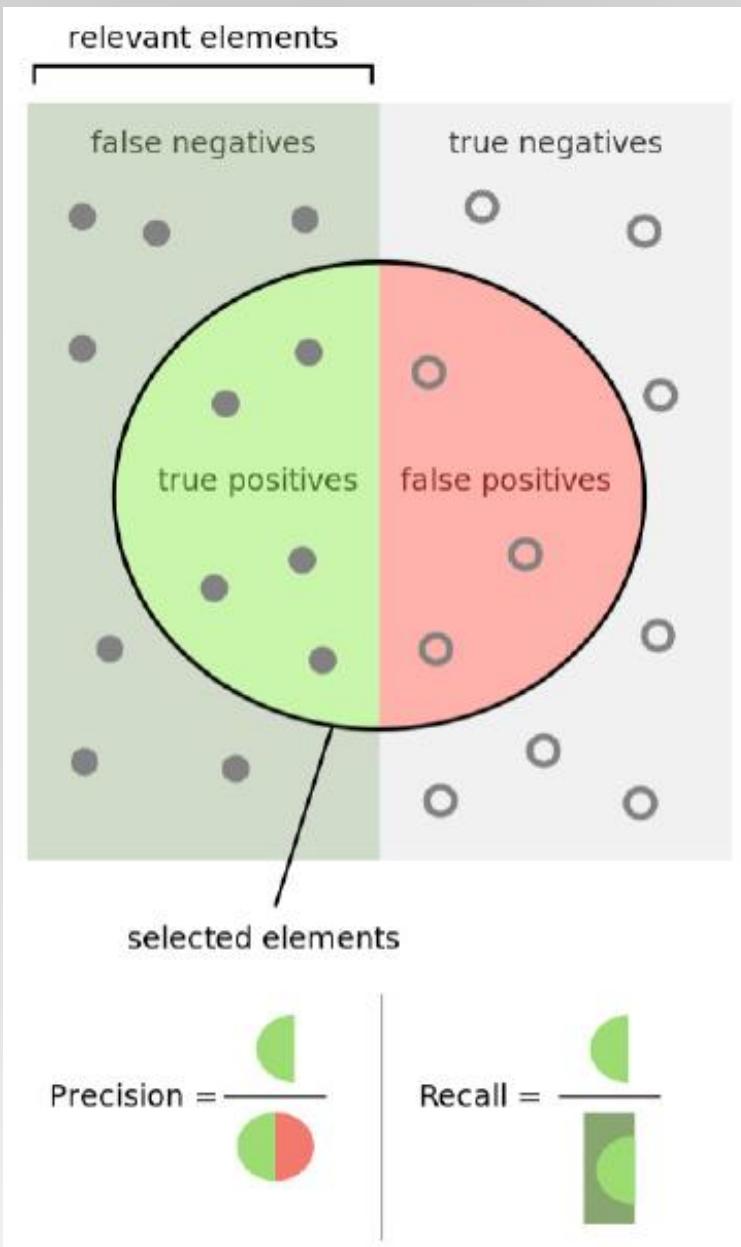
	$y = 1$ Могут вернуть	$y = -1$ Не могут вернуть
$a(x) = 1$ Получили кредит	80	20
$a(x) = -1$ Не получили кредит	20	80

Модель $a_2(x)$:

$$\text{recall}(a_2, X) = 0.48$$

	$y = 1$ Могут вернуть	$y = -1$ Не могут вернуть
$a(x) = 1$ Получили кредит	48	2
$a(x) = -1$ Не получили кредит	52	98

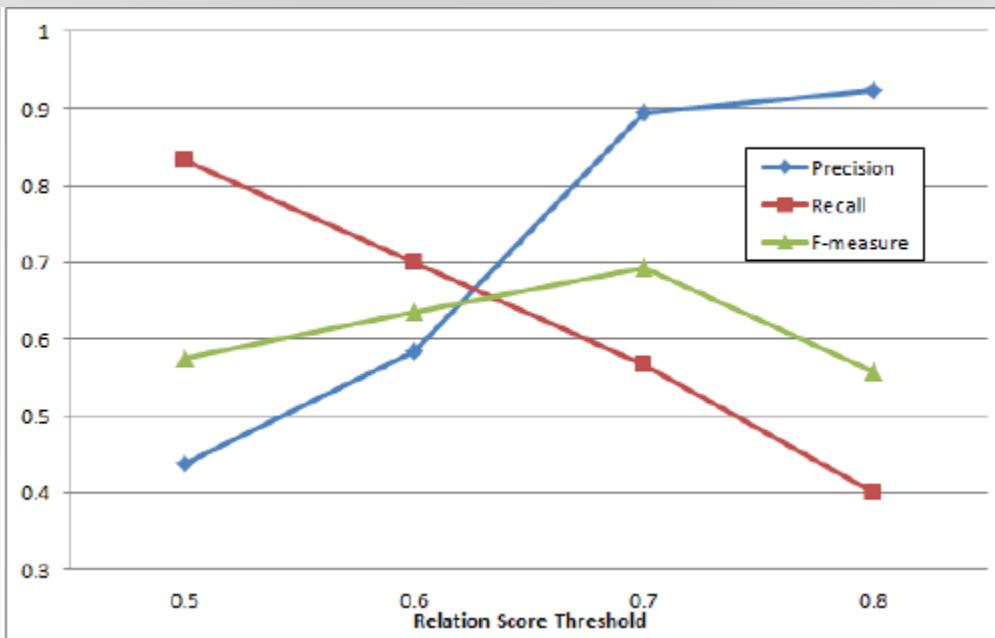
ТОЧНОСТЬ И ПОЛНОТА



F-МЕРА

F-мера – это метрика качества, учитывающая и точность, и полноту

$$F(a, X) = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$



РЕГУЛИРУЕМ ТОЧНОСТЬ И ПОЛНОТУ

Пусть $p(x)$ - уверенность классификатора в том, что объект x относится к классу +1, $p(x) \in [0; 1]$.

Обычно если $p(x) > 0.5$, то мы относим объект к положительному классу, а иначе – к отрицательному.

РЕГУЛИРУЕМ ТОЧНОСТЬ И ПОЛНОТУ

Пусть $p(x)$ - уверенность классификатора в том, что объект x относится к классу +1, $p(x) \in [0; 1]$.

Обычно если $p(x) > 0.5$, то мы относим объект к положительному классу, а иначе – к отрицательному.

Можно изменять этот порог, то есть вместо 0.5 брать другое число из отрезка $[0; 1]$.

РЕГУЛИРУЕМ ТОЧНОСТЬ И ПОЛНОТУ

Пусть $p(x)$ - уверенность классификатора в том, что объект x относится к классу +1, $p(x) \in [0; 1]$.

Обычно если $p(x) > 0.5$, то мы относим объект к положительному классу, а иначе – к отрицательному.

Можно изменять этот порог, то есть вместо 0.5 брать другое число из отрезка $[0; 1]$.

Путем изменения порога t можно регулировать точность и полноту:

➤ Чему будут равны точность и полнота при $t = 0$?

РЕГУЛИРУЕМ ТОЧНОСТЬ И ПОЛНОТУ

Пусть $p(x)$ - уверенность классификатора в том, что объект x относится к классу +1, $p(x) \in [0; 1]$.

Обычно если $p(x) > 0.5$, то мы относим объект к положительному классу, а иначе – к отрицательному.

Можно изменять этот порог, то есть вместо 0.5 брать другое число из отрезка $[0; 1]$.

Путем изменения порога t можно регулировать точность и полноту:

- при $t = 0$ мы все объекты относим к положительному классу, то есть **полнота = 1, а точность маленькая**.

РЕГУЛИРУЕМ ТОЧНОСТЬ И ПОЛНОТУ

Пусть $p(x)$ - уверенность классификатора в том, что объект x относится к классу +1, $p(x) \in [0; 1]$.

Обычно если $p(x) > 0.5$, то мы относим объект к положительному классу, а иначе – к отрицательному.

Можно изменять этот порог, то есть вместо 0.5 брать другое число из отрезка $[0; 1]$.

Путем изменения порога t можно регулировать точность и полноту:

- при $t = 0$ мы все объекты относим к положительному классу, то есть полнота = 1, а точность маленькая.
- **При увеличении t полнота уменьшается** (могут появиться объекты положительного класса, которые мы не нашли), **а точность возрастает** (появляются объекты положительного класса).

$$\text{Pree} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$t = 0 \Rightarrow \text{fee} + 1.$$

$$\frac{\text{TP}}{\text{TP} + \text{FP}} =$$

$$\text{rec} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{\text{TP}}{\text{TP}} = 1$$

$$t = 3$$

		real	
		non	omp
pred	non	TP	FP
	omp	FN	TN

[0, 1] 0, 5

res > 0.5 \Rightarrow +1

res $\leq 0.5 \Rightarrow -1$

ИНТЕГРАЛЬНАЯ МЕТРИКА: ROC-AUC

Хотим измерить качество всего семейства классификаторов независимо от выбранного порога.

Для этого будем использовать метрику AUC

AUC – *Area Under ROC Curve (площадь под ROC-кривой)*

ROC-КРИВАЯ

Для каждого значения порога t вычислим:

- **False Positive Rate** (доля неверно принятых объектов отрицательного класса):

$$FPR = \frac{FP}{FP + TN} = \frac{\sum_i [y_i = -1] [a(x_i) = +1]}{\sum_i [y_i = -1]}$$

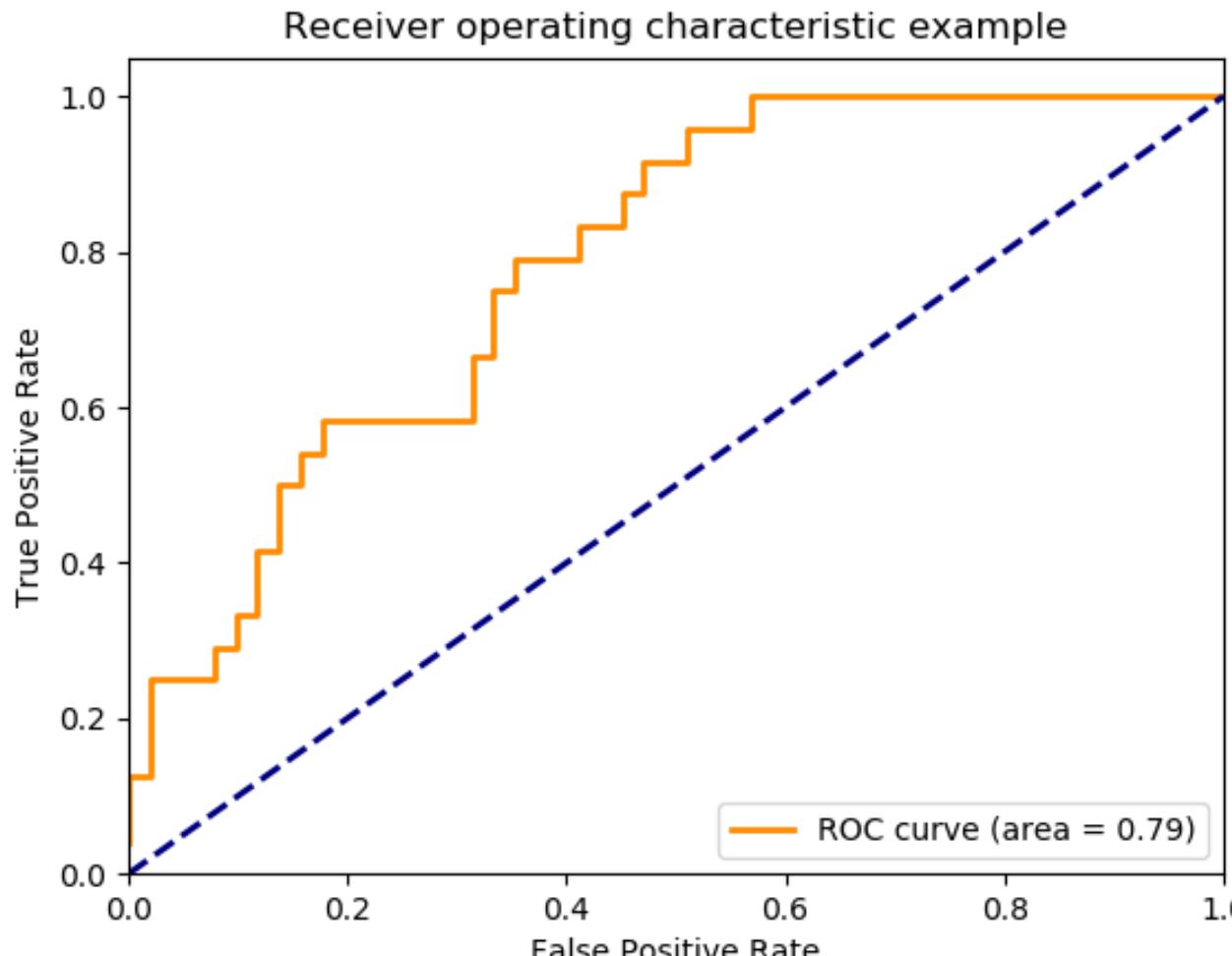
- **True Positive Rate** (доля верно принятых объектов положительного класса):

$$TPR = \frac{TP}{TP+FN} = \frac{\sum_i [y_i=+1] [a(x_i)=+1]}{\sum_i [y_i=+1]}.$$

		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

ROC-КРИВАЯ

Кривая, состоящая из точек с координатами (FPR,TPR) для всех возможных порогов – это и есть ROC-кривая.

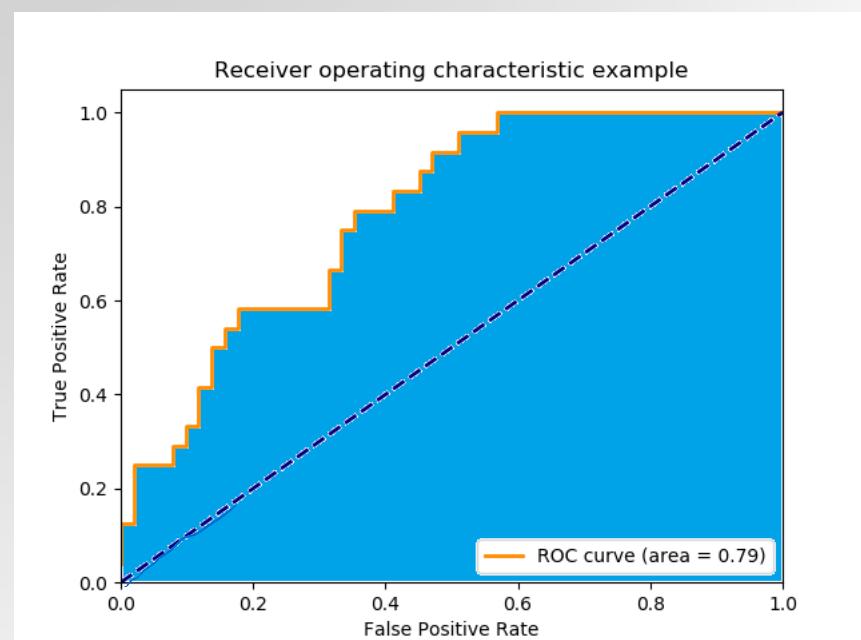


ROC-КРИВАЯ. AUC.

AUC (Area Under Curve) – площадь под ROC-кривой.

$$AUC \in [0; 1].$$

- Чему равен AUC при идеальной классификации? 1
- Чему равен AUC при случайной классификации? 0,5



ROC-КРИВАЯ. AUC.

AUC (Area Under Curve) – площадь под ROC-кривой.

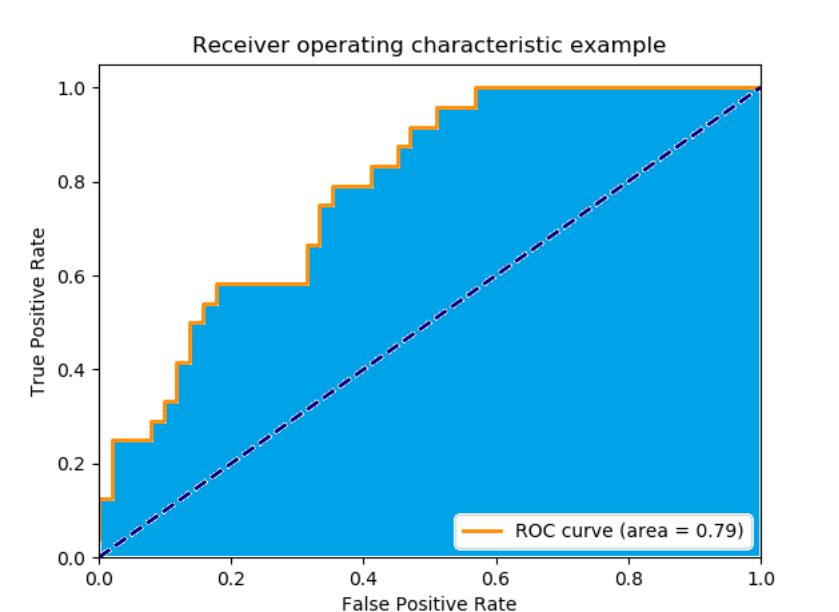
$$AUC \in [0; 1].$$

- $AUC = 1$ –

иdealная классификация

- $AUC = 0.5$ –

случайная классификация



ПРИМЕР ПОСТРОЕНИЯ ROC-КРИВОЙ

- Пусть есть выборка из 5 объектов и следующие предсказания классификатора оценки принадлежности к классу +1:

$b(x)$	0.2	0.4	0.1	0.7	0.05
y	-1	+1	-1	+1	+1

ПРИМЕР ПОСТРОЕНИЯ ROC-КРИВОЙ

- Пусть есть выборка из 5 объектов и следующие предсказания классификатора оценки принадлежности к классу +1:

$b(x)$	0.2	0.4	0.1	0.7	0.05
y	-1	+1	-1	+1	+1

- Упорядочим объекты по убыванию предсказаний: $(0.7, 0.4, 0.2, 0.1, 0.05)$

1 шаг: $t = 0.7$, то есть

$$a(x) = [b(x) > 0.7]$$

$$TPR = \frac{TP}{TP+FN}$$

$$FPR = \frac{FP}{FP+TN}$$

ПРИМЕР ПОСТРОЕНИЯ ROC-КРИВОЙ

- Пусть есть выборка из 5 объектов и следующие предсказания классификатора оценки принадлежности к классу +1:

$b(x)$	0.2	0.4	0.1	0.7	0.05
y	-1	+1	-1	+1	+1

- Упорядочим объекты по убыванию предсказаний: $(0.7, 0.4, 0.2, 0.1, 0.05)$

1 шаг: $t = 0.7$, то есть

$$a(x) = [b(x) > 0.7]$$

$$TPR = \frac{0}{0+3} = 0, \quad FPR = \frac{0}{0+2} = 0.$$

$$TPR = \frac{TP}{TP+FN}$$

$$FPR = \frac{FP}{FP+TN}$$

ПРИМЕР ПОСТРОЕНИЯ ROC-КРИВОЙ

- Пусть есть выборка из 5 объектов и следующие предсказания классификатора оценки принадлежности к классу +1:

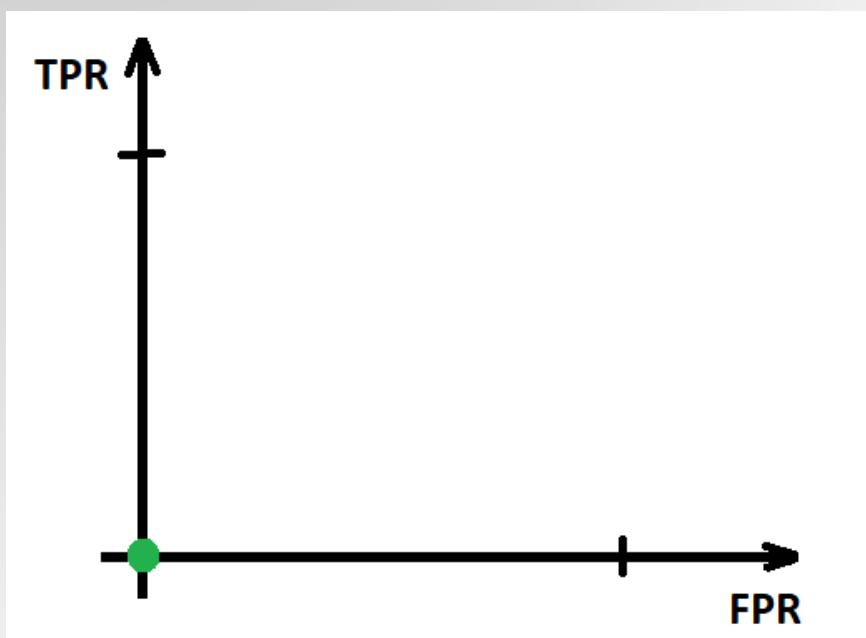
$b(x)$	0.2	0.4	0.1	0.7	0.05
y	-1	+1	-1	+1	+1

- Упорядочим объекты по убыванию предсказаний:
 $(0.7, 0.4, 0.2, 0.1, 0.05)$

1 шаг: $t = 0.7$, то есть
 $a(x) = [b(x) > 0.7]$

$$TPR = \frac{0}{0+3} = 0,$$

$$FPR = \frac{0}{0+2} = 0.$$



ПРИМЕР ПОСТРОЕНИЯ ROC-КРИВОЙ

- Оценки принадлежности к классу +1:

$b(x)$	0.2	0.4	0.1	0.7	0.05
y	-1	+1	-1	+1	+1

- БИН. Рядок
- Упорядочим объекты по убыванию предсказаний:

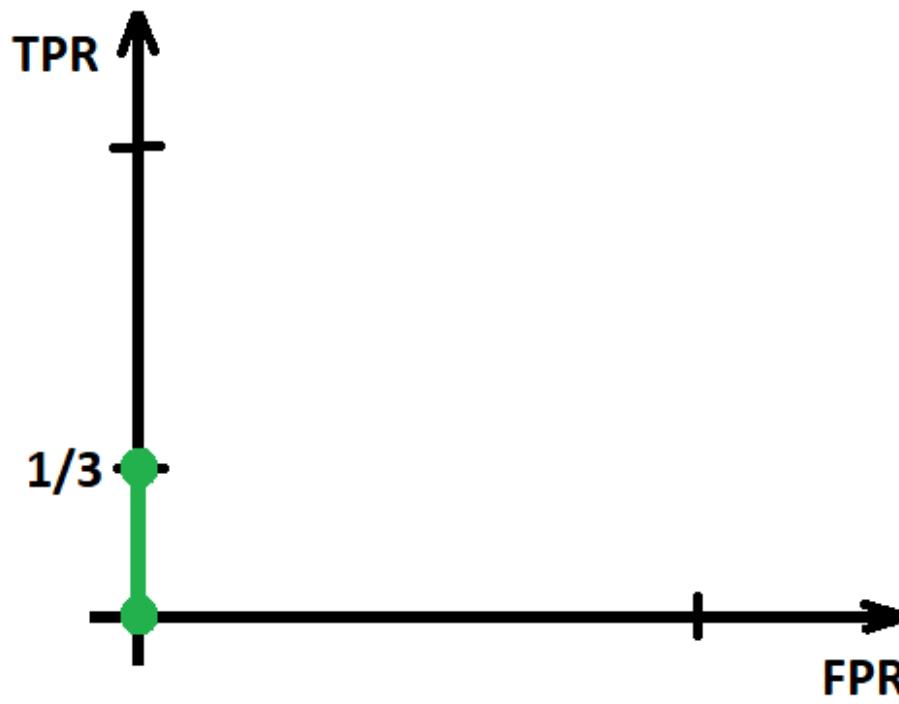
(0.7, 0.4, 0.2, 0.1, 0.05)

2 шаг: $t = 0.4$, то есть

$$a(x) = [b(x) > 0.4]$$

$$TPR = \frac{1}{1+2} = \frac{1}{3},$$

$$FPR = \frac{0}{0+2} = 0.$$



ПРИМЕР ПОСТРОЕНИЯ ROC-КРИВОЙ

- Оценки принадлежности к классу +1:

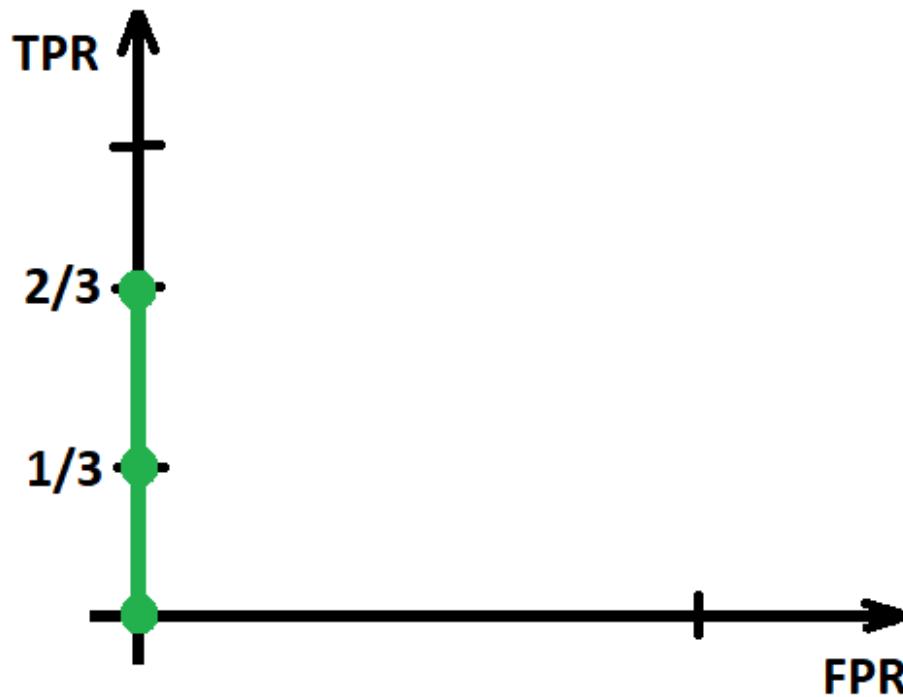
$b(x)$	0.2	0.4	0.1	0.7	0.05
y	-1	+1	-1	+1	+1

- Упорядочим объекты по убыванию предсказаний:
 $(0.7, 0.4, 0.2, 0.1, 0.05)$

3 шаг: $t = 0.2$, то есть
 $a(x) = [b(x) > 0.2]$

$$TPR = \frac{2}{2+1} = \frac{2}{3},$$

$$FPR = \frac{0}{0+2} = 0.$$



ПРИМЕР ПОСТРОЕНИЯ ROC-КРИВОЙ

- Оценки принадлежности к классу +1:

$b(x)$	0.2	0.4	0.1	0.7	0.05
y	-1	+1	-1	+1	+1

- Упорядочим объекты по убыванию предсказаний:

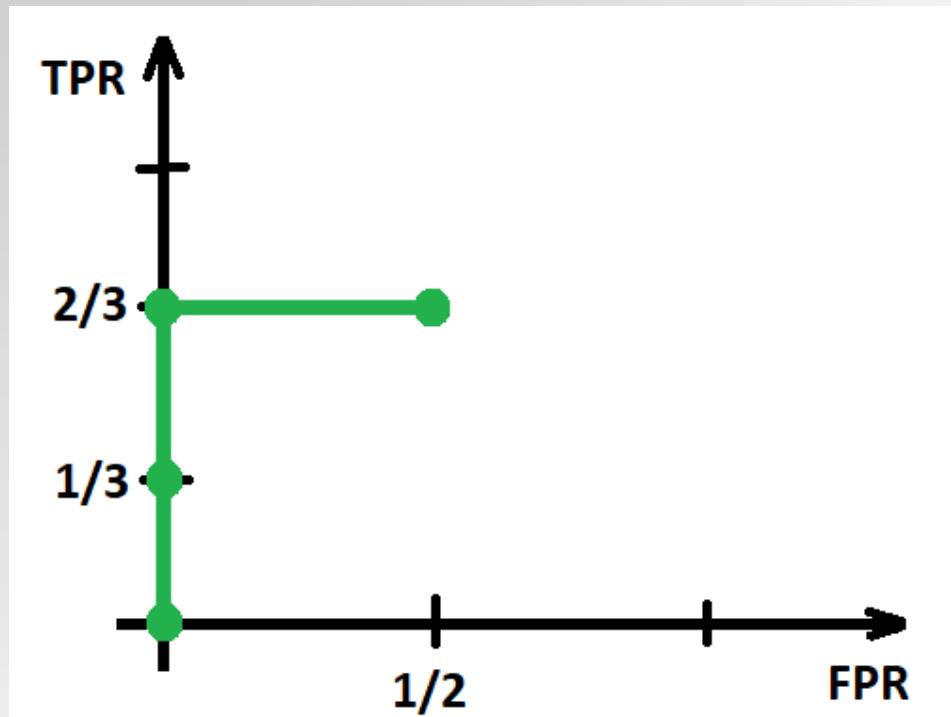
(0.7,0.4,0.2,0.1,0.05)

4 шаг: $t = 0.1$, то есть

$$a(x) = [b(x) > 0.1]$$

$$TPR = \frac{2}{2+1} = \frac{2}{3},$$

$$FPR = \frac{1}{1+1} = \frac{1}{2}.$$



ПРИМЕР ПОСТРОЕНИЯ ROC-КРИВОЙ

- Оценки принадлежности к классу +1:

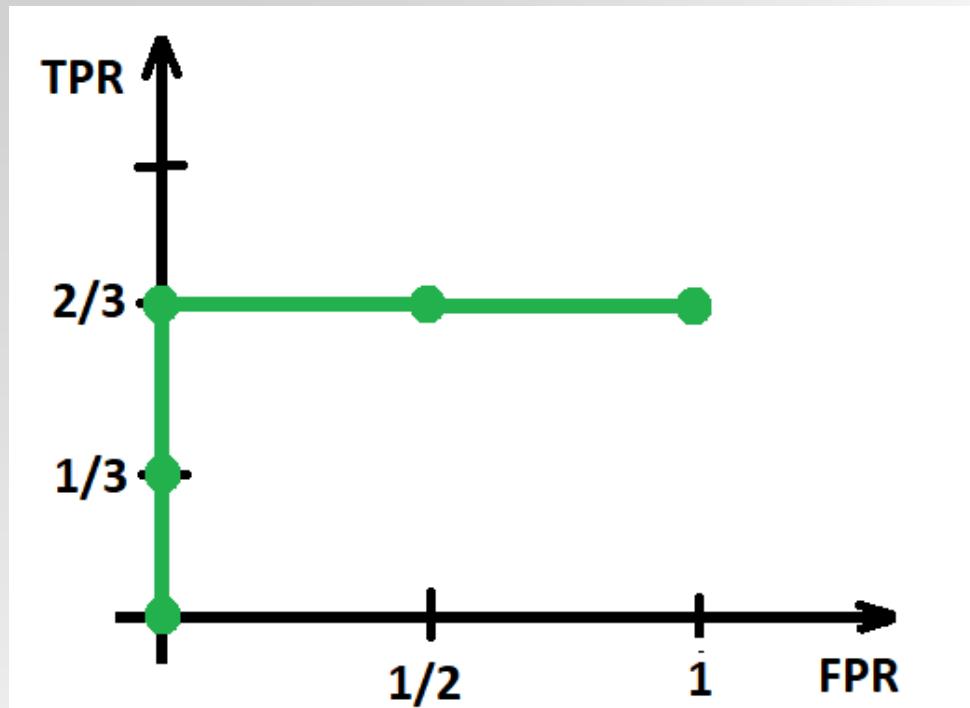
$b(x)$	0.2	0.4	0.1	0.7	0.05
y	-1	+1	-1	+1	+1

- Упорядочим объекты по убыванию предсказаний:
 $(0.7, 0.4, 0.2, 0.1, 0.05)$

5 шаг: $t = 0.05$, то есть
 $a(x) = [b(x) > 0.05]$

$$TPR = \frac{2}{2+1} = \frac{2}{3},$$

$$FPR = \frac{2}{2+0} = 1.$$



ПРИМЕР ПОСТРОЕНИЯ ROC-КРИВОЙ

- Оценки принадлежности к классу +1:

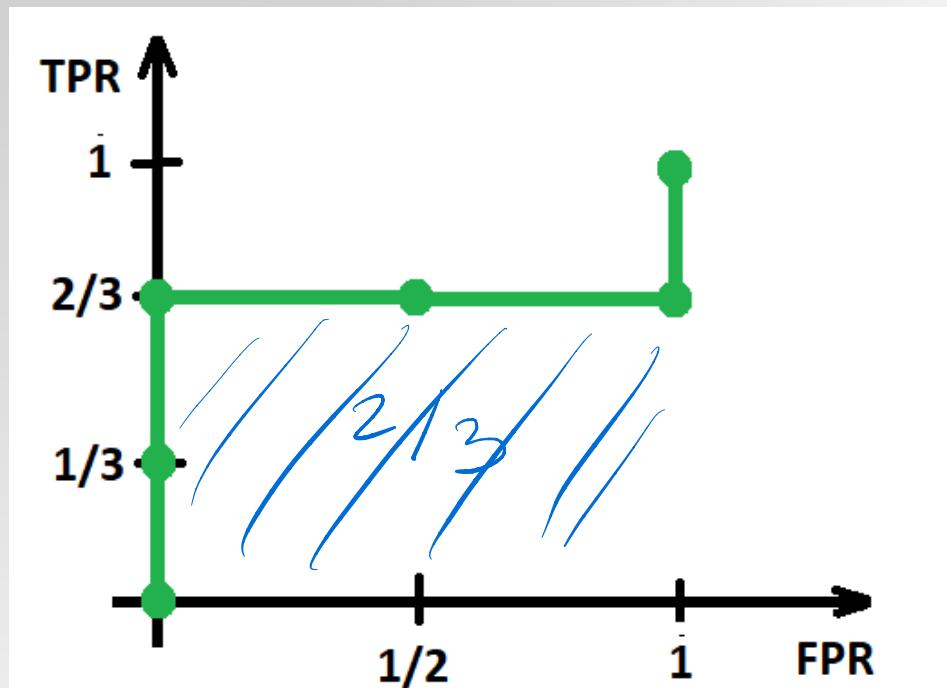
$b(x)$	0.2	0.4	0.1	0.7	0.05
y	-1	+1	-1	+1	+1

- Упорядочим объекты по убыванию предсказаний:
 $(0.7, 0.4, 0.2, 0.1, 0.05)$

5 шаг: $t = 0$, то есть
 $a(x) = [b(x) > 0]$

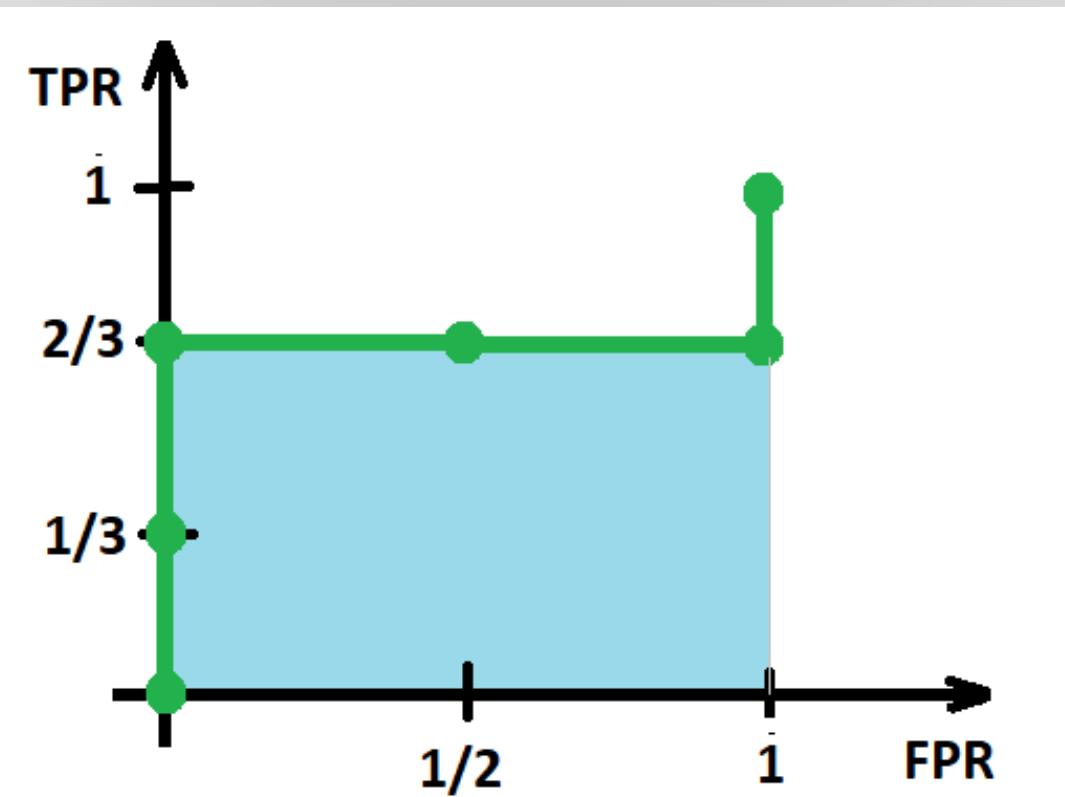
$$TPR = \frac{3}{3+0} = 1,$$

$$FPR = \frac{2}{2+0} = 1.$$



ПРИМЕР ПОСТРОЕНИЯ ROC-КРИВОЙ

$$AUC = 2/3$$

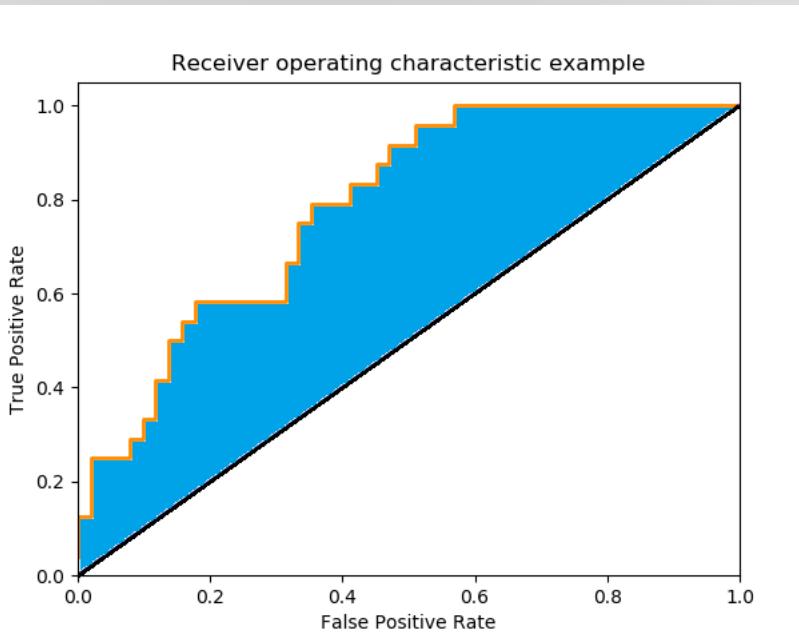


ИНДЕКС ДЖИНИ

Индекс Джини:

$$Gini = 2 \cdot AUC - 1$$

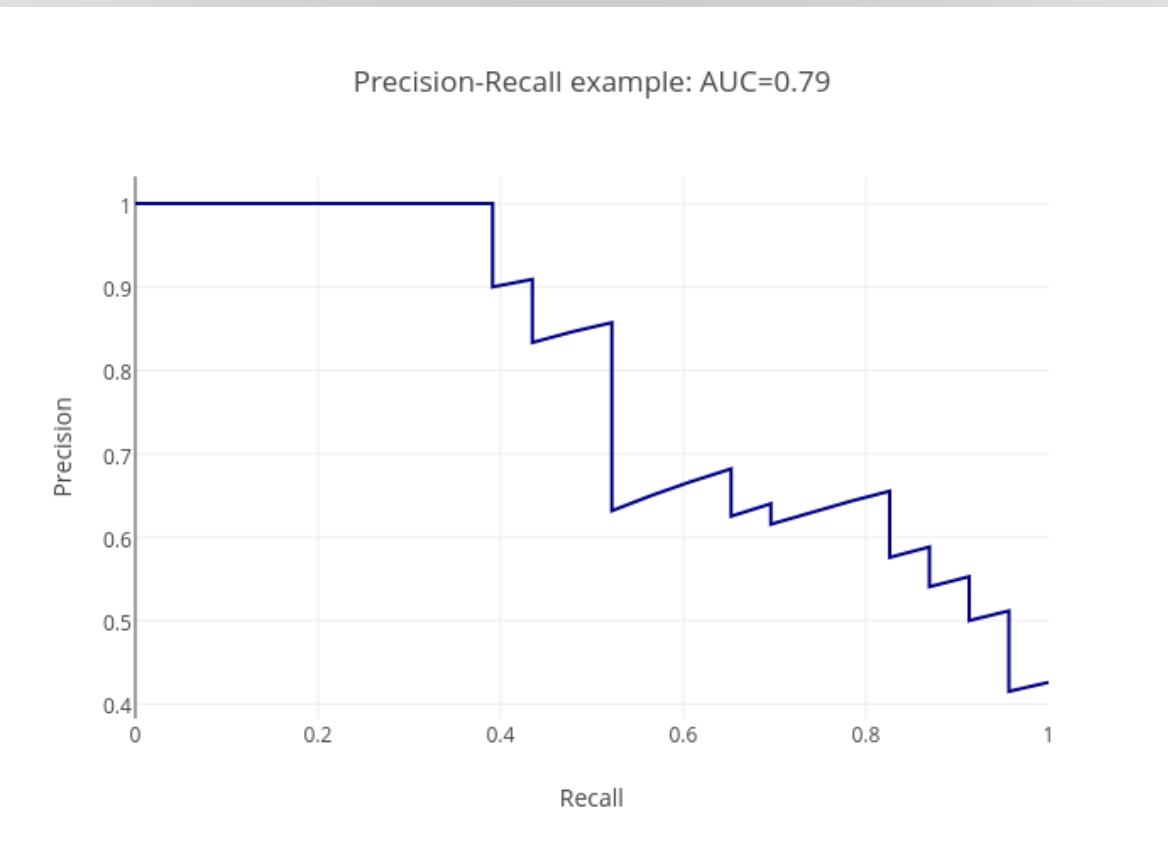
- Индекс Джини – это удвоенная площадь между главной диагональю и ROC-кривой.



PRECISION-RECALL КРИВАЯ

- В случае малой доли объектов положительного класса AUC-ROC может давать неадекватно хороший результат

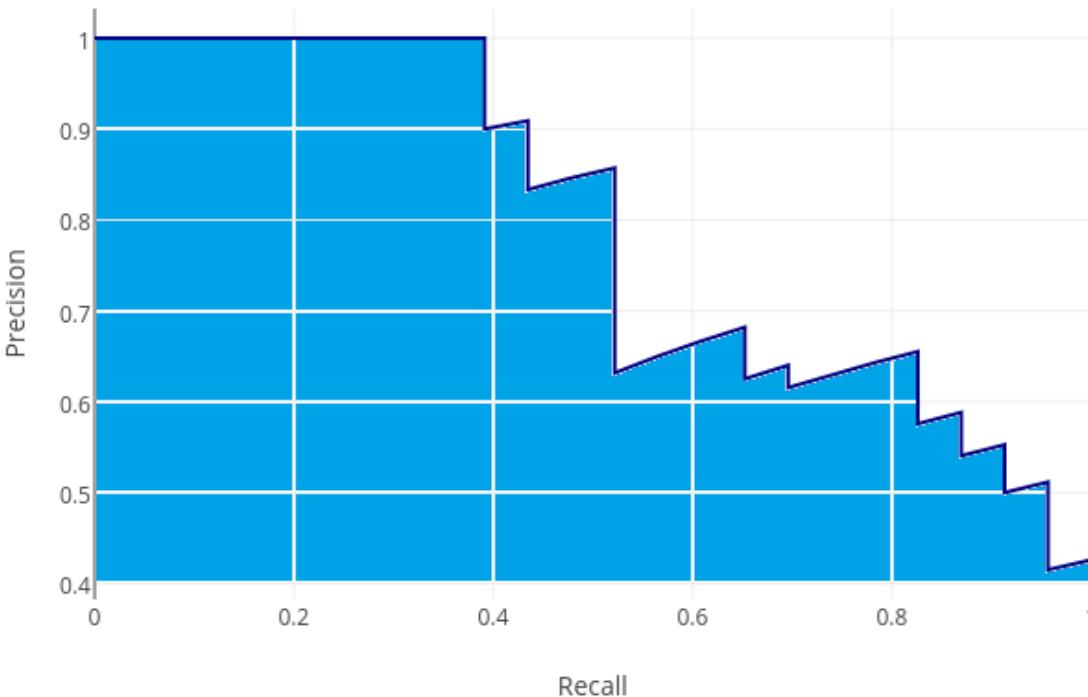
Precision-Recall кривая:



AUC-PR

AUC-PR – площадь под PR-кривой

Precision-Recall example: AUC=0.79



МНОГОКЛАССОВАЯ КЛАССИФИКАЦИЯ

ПОДХОД ONE-VS-ALL

Решаем задачу классификации на K классов.

- Обучим K бинарных классификаторов $b_1(x), \dots, b_K(x)$, каждый из которых решает задачу: *принадлежит объект x к классу k_i или не принадлежит?*

Например, линейные классификаторы будут иметь вид

$$b_k(x) = \text{sign}((w_k, x) + w_{0k})$$

Как получаем итоговое предсказание?

ПОДХОД ONE-VS-ALL

Решаем задачу классификации на K классов.

- Обучим K бинарных классификаторов $b_1(x), \dots, b_K(x)$, каждый из которых решает задачу: *принадлежит объект x к классу k_i или не принадлежит?*

Например, линейные классификаторы будут иметь вид

$$b_k(x) = \text{sign}((w_k, x) + w_{0k})$$

- Тогда в качестве итогового предсказания будем выдавать класс самого уверенного классификатора:

$$a(x) = \underset{k \in \{1, \dots, K\}}{\text{argmax}} ((w_k, x) + w_{0k})$$

ПОДХОД ONE-VS-ALL

Решаем задачу классификации на K классов.

- Обучим K бинарных классификаторов $b_1(x), \dots, b_k(x)$, каждый из которых решает задачу: *принадлежит объект x к классу k_i или не принадлежит?*

Например, линейные классификаторы будут иметь вид

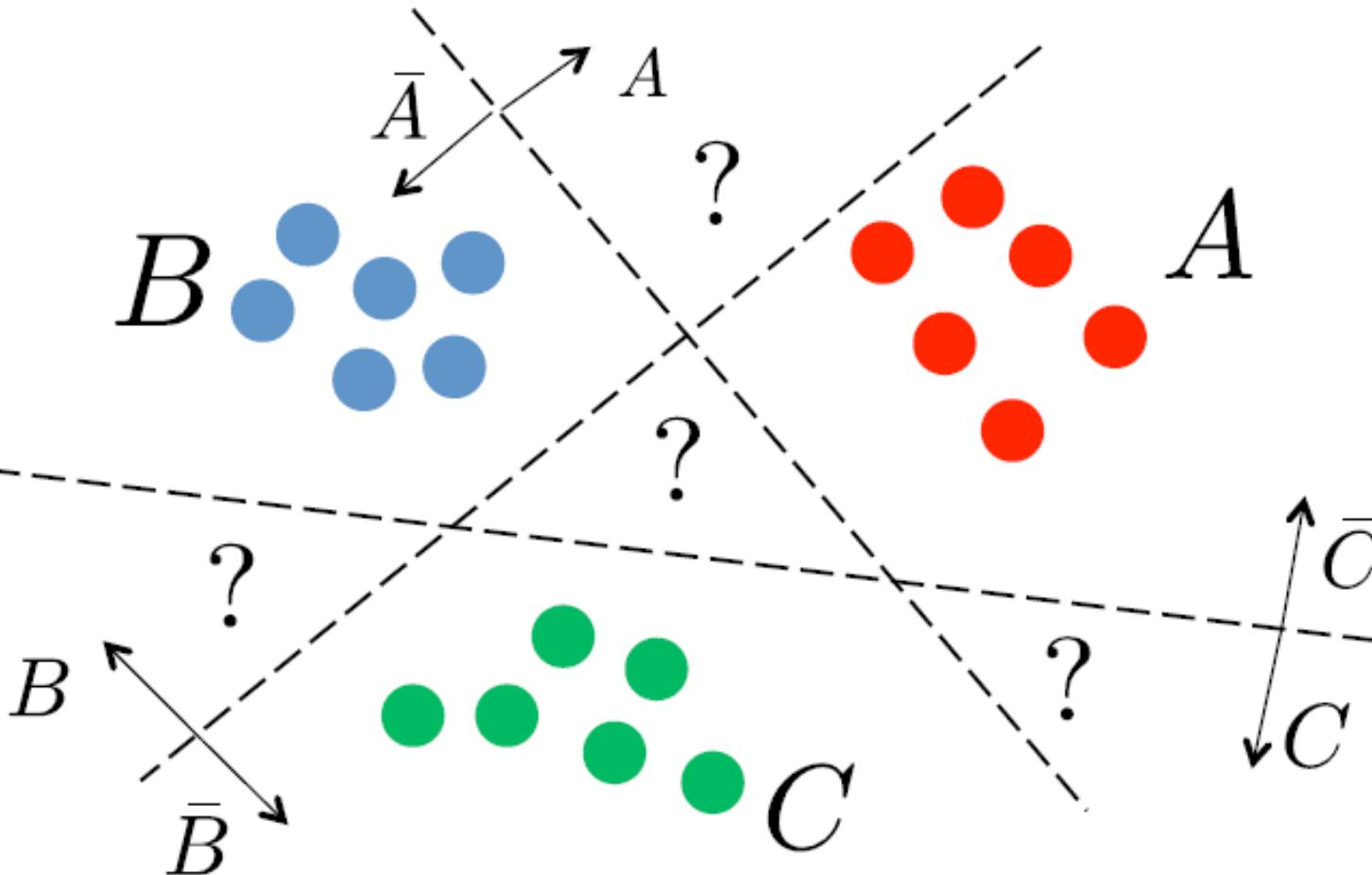
$$b_k(x) = \text{sign}((w_k, x) + w_{0k})$$

- Тогда в качестве итогового предсказания будем выдавать класс самого уверенного классификатора:

$$a(x) = \underset{k \in \{1, \dots, K\}}{\text{argmax}} ((w_k, x) + w_{0k})$$

- Предсказания классификаторов могут иметь разные масштабы, поэтому сравнивать их некорректно.

ПОДХОД ONE-VS-ALL



ПОДХОД ALL-VS-ALL

- Для каждой пары классов i и j обучим бинарный классификатор $a_{ij}(x)$, который будет предсказывать класс i или j

(если всего K классов, то получим C_K^2 классификаторов).

Каждый такой классификатор будем обучать только на объектах классов i и j .

Как получаем итоговое предсказание?

ПОДХОД ALL-VS-ALL

- Для каждой пары классов i и j обучим бинарный классификатор $a_{ij}(x)$, который будет предсказывать класс i или j

(если всего K классов, то получим C_K^2 классификаторов).

Каждый такой классификатор будем обучать только на объектах классов i и j .

- В качестве итогового предсказания выдадим класс, который предсказало наибольшее число алгоритмов:

$$a(x) = \operatorname{argmax}_{k \in \{1, \dots, K\}} \sum_{i=1}^K \sum_{j \neq i} [a_{ij}(x) = k]$$

ПОДХОД ALL-VS-ALL

- Для каждой пары классов i и j обучим бинарный классификатор $a_{ij}(x)$, который будет предсказывать класс i или j

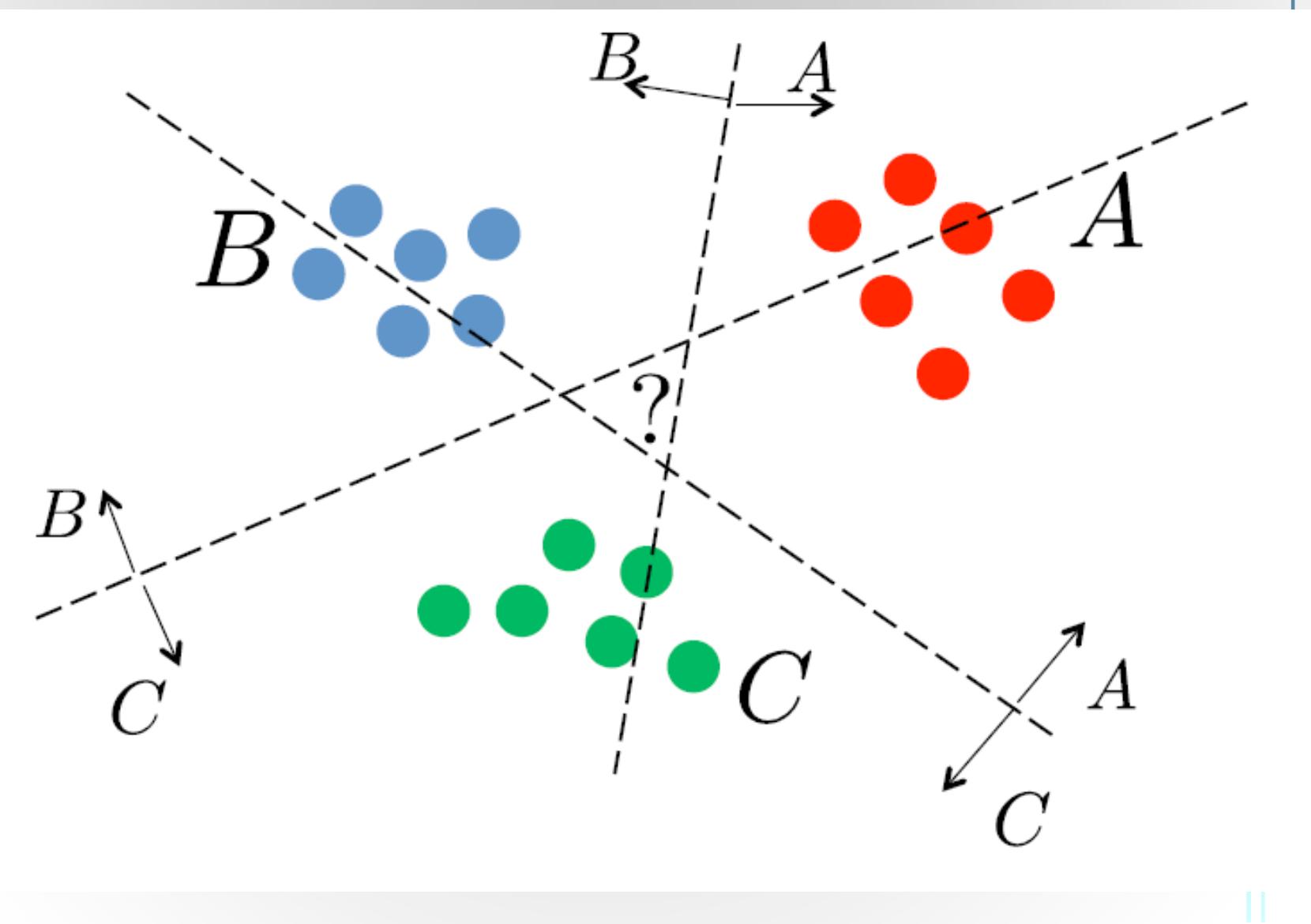
(если всего K классов, то получим C_K^2 классификаторов).

Каждый такой классификатор будем обучать только на объектах классов i и j .

- В качестве итогового предсказания выдадим класс, который предсказало наибольшее число алгоритмов:

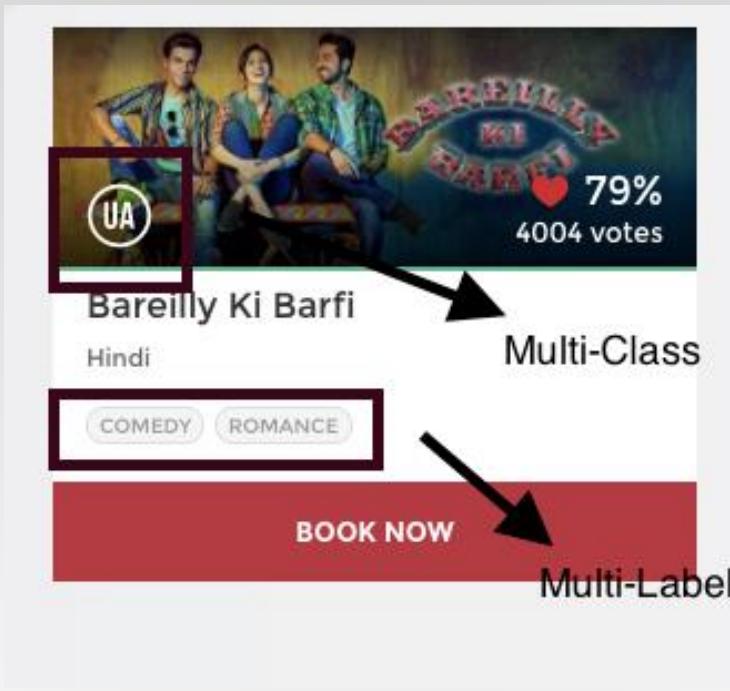
$$a(x) = \operatorname{argmax}_{k \in \{1, \dots, K\}} \sum_{i=1}^K \sum_{j \neq i} [a_{ij}(x) = k]$$

ПОДХОД ALL-VS-ALL



MULTICLASS AND MULTI-LABEL CLASSIFICATION

- Если каждый объект может принадлежать только одному классу, то решаем задачу multiclass классификации
- Если каждый объект может принадлежать нескольким классам (задача классификации с пересекающимися классами), то решаем задачу multi-label классификации.



МЕТРИКИ КАЧЕСТВА

Идея: сводим подсчет метрик к бинарному случаю

Подход 1 ([микроусреднение, micro average](#)):

- Вычислим для каждого двухклассового классификатора $a^k(x) = [a(x) = k]$ метрики TP_k, FP_k, FN_k, TN_k
- Усредним каждую характеристику по всем классам, например, $TP = \frac{1}{K} \sum_{k=1}^K TP_k$.

Тогда точность в многоклассовом случае:

$$precision(a, X) = \frac{TP}{TP + FP}$$

МЕТРИКИ КАЧЕСТВА

Идея: сводим подсчет метрик к бинарному случаю

Подход 2 (макроусреднение, macro average):

- Вычислим для каждого двухклассового классификатора $a^k(x) = [a(x) = k]$ метрики TP_k, FP_k, FN_k, TN_k
- Вычислим итоговую метрику для каждого класса в

отдельности: $precision_k(a, X) = \frac{TP_k}{TP_k + FP_k}$

Тогда точность в многоклассовом случае:

$$precision(a, X) = \frac{1}{K} \sum_{k=1}^K precision_k(a, X)$$

макроуровень

1. считаем κ знающих TP_k, FP_k, TN_k, FN_k
2. усредняем TP, FP, TN, FN
3. по упр. TP, FP, TN, FN 算出 $pr, recall$

макроуровень

1. считаем κ знающих TP_k, FP_k, TN_k, FN_k
2. считаем κ знающих pr_k, rec_k
3. усредняем pr, rec .

МЕТРИКИ КАЧЕСТВА (ПРИМЕР)

Результаты некоторого классификатора:

		True/Actual		
		Cat (img alt="Cat icon" data-bbox="445 375 485 425})	Fish (img alt="Fish icon" data-bbox="645 375 685 425})	Hen (img alt="Hen icon" data-bbox="845 375 885 425})
Predicted	Cat (img alt="Cat icon" data-bbox="115 475 245 525})	4	6	3
	Fish (img alt="Fish icon" data-bbox="115 575 245 625})	1	2	0
	Hen (img alt="Hen icon" data-bbox="115 695 245 745})	1	2	6

МЕТРИКИ КАЧЕСТВА (ПРИМЕР)

		True/Actual		
		Cat (img alt="Cat icon" data-bbox="425 215 475 275})	Fish (img alt="Fish icon" data-bbox="615 215 665 275})	Hen (img alt="Hen icon" data-bbox="805 215 855 275})
Predicted	Cat (img alt="Cat icon" data-bbox="135 305 235 365")	4	6	3
	Fish (img alt="Fish icon" data-bbox="135 425 235 485")	1	2	0
	Hen (img alt="Hen icon" data-bbox="135 545 235 605")	1	2	6

	precision	recall	f1-score	support
Cat	0.308	0.667	0.421	6
Fish	0.667	0.200	0.308	10
Hen	0.667	0.667	0.667	9
micro avg	0.480	0.480	0.480	25
macro avg	0.547	0.511	0.465	25
weighted avg	0.581	0.480	0.464	25

МНОГОКЛАССОВАЯ ЛОГИСТИЧЕСКАЯ РЕГРЕССИЯ

- Бинарная лог.регрессия предсказывает вероятность класса 1:

$$(w, x) \rightarrow a(x) = \frac{1}{1 + e^{-(w, x)}} = \frac{e^{(w, x)}}{1 + e^{(w, x)}}$$

- Предположим, у нас есть K линейных моделей, каждая из которых дает оценку принадлежности выбранному классу: $b_k(x) = (w_k, x)$.
- Преобразуем вектор предсказаний в вектор вероятностей (softmax-преобразование):

$$\text{softmax}(\mathbf{b}_1, \dots, \mathbf{b}_K) = \left(\frac{\exp(b_1)}{\sum_{i=1}^K \exp(b_i)}, \frac{\exp(b_2)}{\sum_{i=1}^K \exp(b_i)}, \dots, \frac{\exp(b_K)}{\sum_{i=1}^K \exp(b_i)} \right)$$

Тогда вероятность класса k :

$$P(y = k | x, w) = \frac{\exp((w_k, x))}{\sum_{i=1}^K \exp((w_i, x))}$$

ОБУЧЕНИЕ ВЕСОВ МОДЕЛИ

$$a_j(x) = P(y = j|x, w) = \frac{\exp(b_j(x))}{\sum_{i=1}^K \exp(b_i(x))}$$

*Обучение – по методу максимального правдоподобия
(аналогично бинарной классификации):*

$$\Pi = \prod_{i=1}^n a_1(x_i)^{[y_i=1]} \cdot a_2(x_i)^{[y_i=2]} \cdot \dots a_K(x_i)^{[y_i=K]} =$$

$$= \prod_{i=1}^n \prod_{j=1}^K a_j(x_i)^{[y_i=j]} \rightarrow \max_{w_1, \dots, w_K}$$

$$-\sum_{i=1}^n \sum_{j=1}^K [y_i = j] \log P(y = j|x_i, w) \rightarrow \min_{w_1, \dots, w_K}$$

ОБУЧЕНИЕ ВЕСОВ МОДЕЛИ

$$a_j(x) = P(y = j|x, w) = \frac{\exp(b_j(x))}{\sum_{i=1}^K \exp(b_i(x))}$$

*Обучение – по методу максимального правдоподобия
(аналогично бинарной классификации):*

$$\Pi = \prod_{i=1}^n a_1(x_i)^{[y_i=1]} \cdot a_2(x_i)^{[y_i=2]} \cdot \dots a_K(x_i)^{[y_i=K]} =$$

$$= \prod_{i=1}^n \prod_{j=1}^K a_j(x_i)^{[y_i=j]} \rightarrow \max_{w_1, \dots, w_K}$$

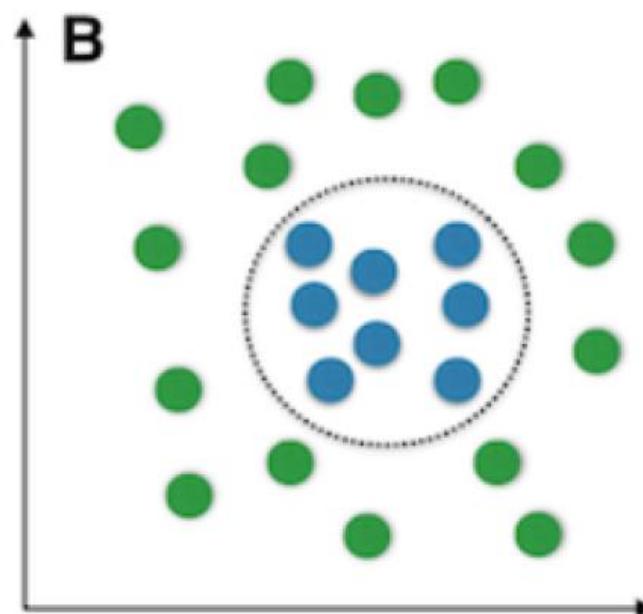
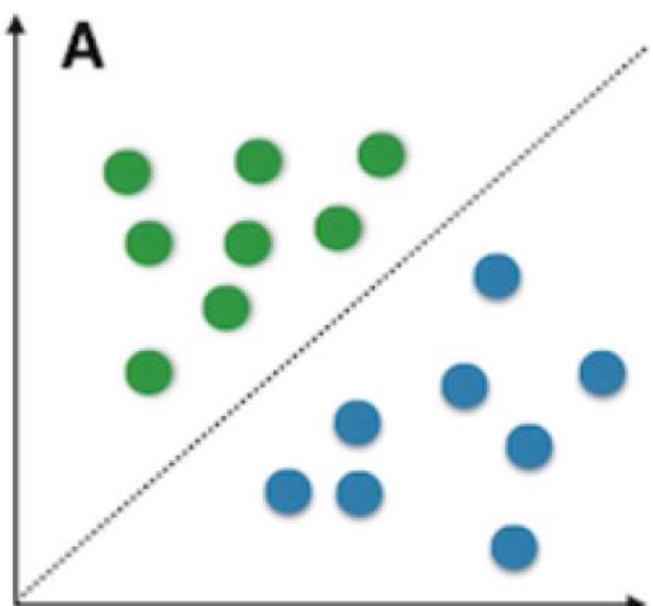
То есть в итоге обучаем одну модель (а не K моделей)

$$-\sum_{i=1}^n \sum_{j=1}^K [y_i = j] \log P(y = j|x_i, w) \rightarrow \min_{w_1, \dots, w_K}$$

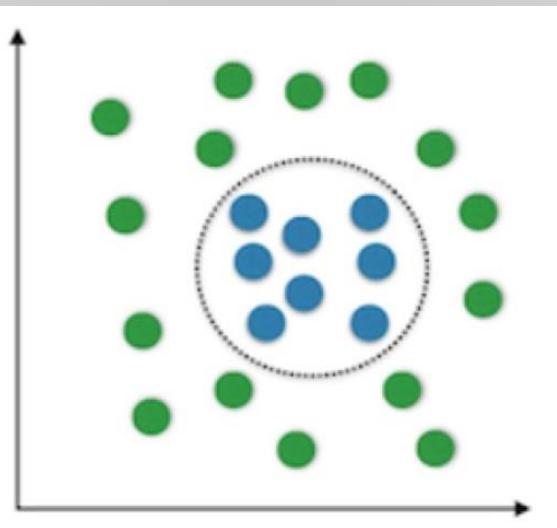
ЯДРОВЫЕ МЕТОДЫ

НЕЛИНЕЙНЫЕ ЗАДАЧИ КЛАССИФИКАЦИИ

Linear vs. nonlinear problems

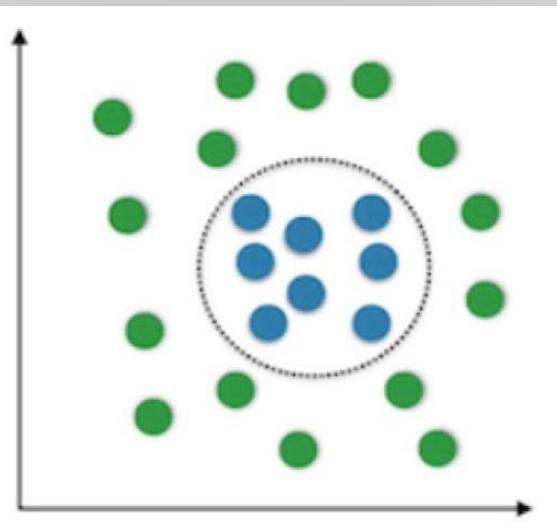


НЕЛИНЕЙНЫЕ ЗАДАЧИ КЛАССИФИКАЦИИ



$$(x_1, x_2) \rightarrow (x_1, x_2, x_3 = x_1^2 + x_2^2)$$

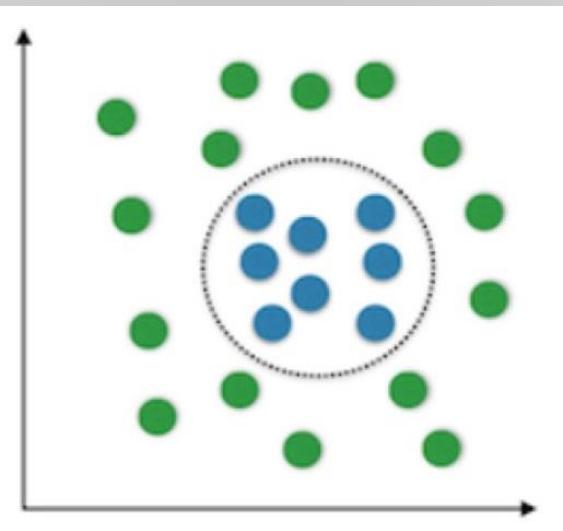
НЕЛИНЕЙНЫЕ ЗАДАЧИ КЛАССИФИКАЦИИ



$$(x_1, x_2) \rightarrow (x_1, x_2, x_3 = x_1^2 + x_2^2)$$

- В новом пространстве признаков выборка идеально разделяется гиперплоскостью.

НЕЛИНЕЙНЫЕ ЗАДАЧИ КЛАССИФИКАЦИИ

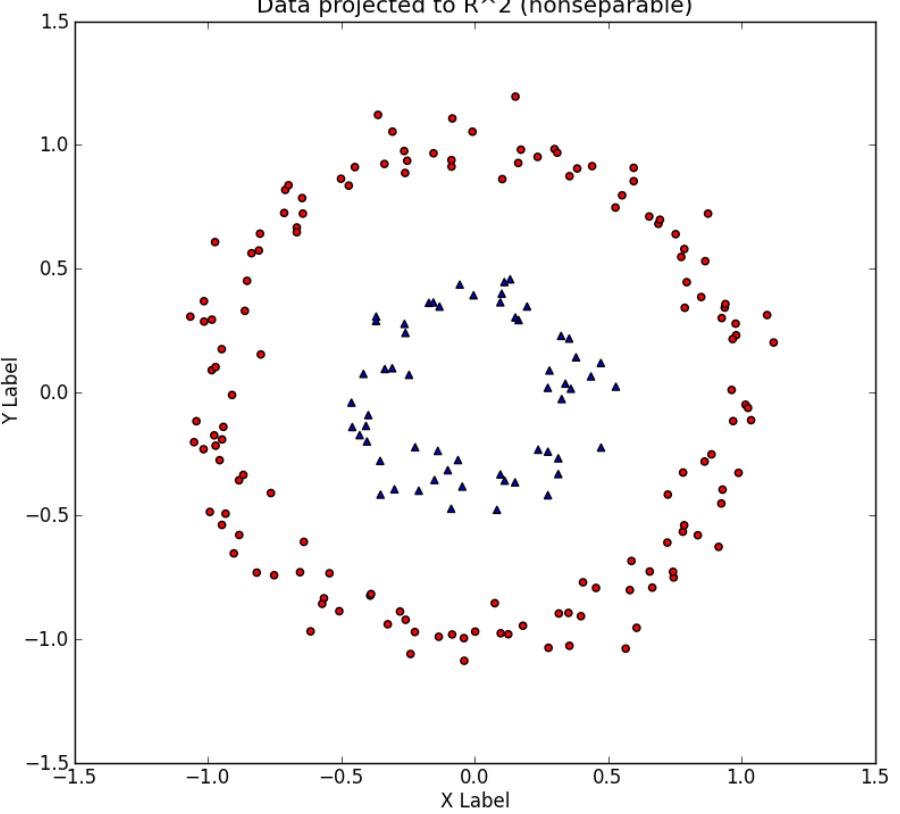


$$(x_1, x_2) \rightarrow (x_1, x_2, x_3 = x_1^2 + x_2^2)$$

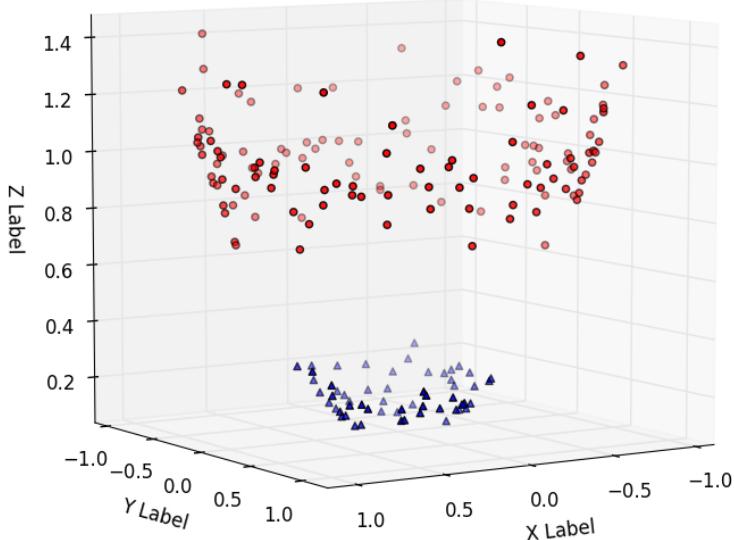
- В новом пространстве признаков выборка идеально разделяется гиперплоскостью.
- ***В новом признаковом пространстве*** классификатор имеет вид $a(x) = sign(w, x) = sign(w_0 + w_1x_1 + w_2x_2 + w_3x_3)$, то есть ***разделяющая поверхность*** $w_0 + w_1x_1 + w_2x_2 + w_3x_3 = 0$ – линейная.

ПЕРЕХОД К НОВЫМ ПРИЗНАКАМ

Data projected to \mathbb{R}^2 (nonseparable)



Data in \mathbb{R}^3 (separable)



ПЕРЕХОД К НОВЫМ ПРИЗНАКАМ

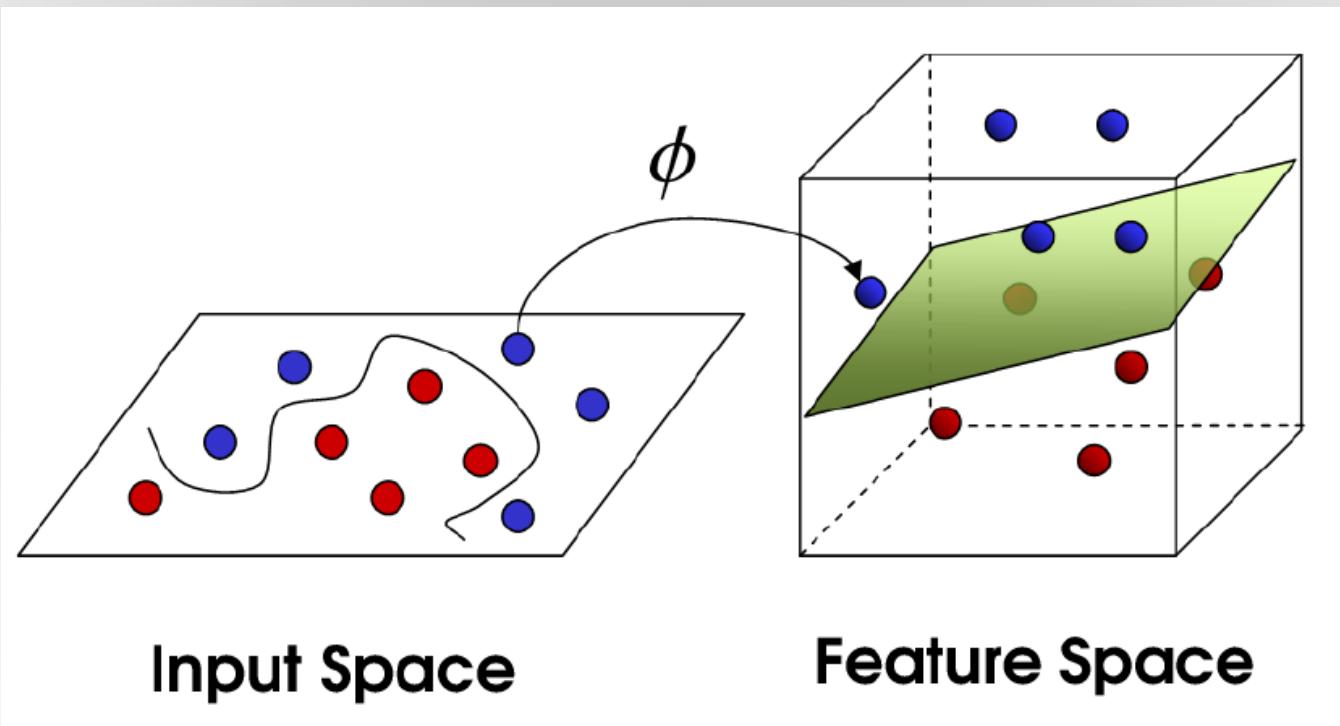
Переход к новым признакам

$$x = (x_1, x_2, \dots, x_d) \rightarrow \varphi_1(x), \varphi_2(x), \dots, \varphi_m(x)$$

Позволяет построить классификатор:

- являющийся *нелинейным в исходном пространстве признаков x_1, x_2, \dots, x_d* (и строящий нелинейную разделяющую поверхность)
- являющийся *линейным в новом пространстве признаков $\varphi_1(x), \varphi_2(x), \dots, \varphi_m(x)$* – спрямляющем пространстве

ПЕРЕХОД К НОВЫМ ПРИЗНАКАМ



ПЕРЕХОД К НОВЫМ ПРИЗНАКАМ

Линейный классификатор:

$$a(x) = \text{sign}((w, \mathbf{x}) + w_0)$$

Классификатор в новом признаковом пространстве

$$a(x) = \text{sign}((w, \varphi(\mathbf{x})) + w_0)$$

ПРОБЛЕМА

При переходе к новым признакам

$$x = (x_1, \dots, x_n) \rightarrow \varphi_1(x), \varphi_2(x), \dots \varphi_N(x)$$

новых признаков может потребоваться довольно много, чтобы линейно разделить выборку в новом пространстве.

Поэтому **сильно возрастает вычислительная сложность алгоритма, а также объем памяти, нужный для хранения всех данных.**

ЯДРОВОЙ ТРЮК (KERNEL TRICK)

При переходе к новым признакам

$$x = (x_1, \dots, x_n) \rightarrow \varphi_1(x), \varphi_2(x), \dots \varphi_N(x)$$

новых признаков может потребоваться довольно много, чтобы линейно разделить выборку в новом пространстве.

Поэтому **сильно возрастает вычислительная сложность алгоритма, а также объем памяти, нужный для хранения всех данных.**

- Существует подход к решению этой проблемы под названием ***kernel trick (ядровой трюк)***. Ядровой трюк позволяет перейти в спрямляющее пространство без увеличения вычислительной сложности и требуемой памяти.

ЯДРО

Ядро – это функция $K(x, z)$, представимая в виде скалярного произведения $\mathbf{K}(x, z) = (\varphi(x), \varphi(z))$, где $\varphi: X \rightarrow H$ – отображение из исходного признакового пространства X в некоторое спрямляющее пространство H .

ЯДРО

Ядро – это функция $K(x, z)$, представимая в виде скалярного произведения $K(x, z) = (\varphi(x), \varphi(z))$, где $\varphi: X \rightarrow H$ – отображение из исходного признакового пространства X в некоторое спрямляющее пространство H .

Теорема (Мерсер). Функция $K(x, z)$ является ядром тогда и только тогда, когда:

1) $K(x, z) = K(z, x)$

2) Для любой конечной выборки (x_1, \dots, x_l) матрица $K =$

$$\left(K(x_i, x_j) \right)_{i,j=1}^l \text{ неотрицательно определена}$$

ЯДРО

Ядро – это функция $K(x, z)$, представимая в виде скалярного произведения $K(x, z) = (\varphi(x), \varphi(z))$, где $\varphi: X \rightarrow H$ – отображение из исходного признакового пространства X в некоторое спрямляющее пространство H .

Теорема (Мерсер). Функция $K(x, z)$ является ядром тогда и только тогда, когда:

1) $K(x, z) = K(z, x)$

2) Для любой конечной выборки (x_1, \dots, x_l) матрица $K =$

$$\left(K(x_i, x_j) \right)_{i,j=1}^l \text{ неотрицательно определена}$$

Из теоремы Мерсера следует, что ядро $K(x, z)$ задаёт скалярное произведение объектов x и z .

KERNEL TRICK

- Идея ядрового трюка состоит в том, что некоторые модели машинного обучения (в частности, линейную регрессию и SVM) можно записать в таком виде, чтобы и модель, и функционал ошибки зависели *только от скалярных произведений объектов (а не от самих объектов)*.

KERNEL TRICK

Идея ядрового трюка состоит в том, что некоторые модели машинного обучения (в частности, линейную регрессию и SVM) можно записать в таком виде, чтобы и модель, и функционал ошибки зависели **только от скалярных произведений объектов (а не от самих объектов)**.

Пример – SVM:

- Исходная модель: $a(x, w) = \text{sign}((w, x) + w_0)$
- Модель можно записать в виде (двойственная запись):

$$a(x, \lambda) = \text{sign}\left(\sum_{i=1}^l \lambda_i y_i (\mathbf{x}_i, \mathbf{x}) - w_0\right)$$

KERNEL TRICK

Идея ядрового трюка состоит в том, что некоторые модели машинного обучения (в частности, линейную регрессию и SVM) можно записать в таком виде, чтобы и модель, и функционал ошибки зависели **только от скалярных произведений объектов (а не от самих объектов)**.

Пример – SVM: *переходим к новым признакам $\varphi(x)$*

- Исходная модель: $a(x, w) = \text{sign}((w, \varphi(x)) + w_0)$
- Модель можно записать в виде (двойственная запись):

$$a(x, \lambda) = \text{sign}\left(\sum_{i=1}^l \lambda_i y_i K(x_i, x) - w_0\right)$$

KERNEL TRICK

Идея ядрового трюка состоит в том, что некоторые модели машинного обучения (в частности, линейную регрессию и SVM) можно записать в таком виде, чтобы и модель, и функционал ошибки зависели **только от скалярных произведений объектов (а не от самих объектов)**.

Пример – SVM: *переходим к новым признакам $\varphi(x)$*

- Исходная модель: $a(x, w) = \text{sign}((w, \varphi(x)) + w_0)$
- Модель можно записать в виде (двойственная запись):

$$a(x, \lambda) = \text{sign}\left(\sum_{i=1}^l \lambda_i y_i K(x_i, x) - w_0\right)$$

Также можно записать функционал ошибки только через скалярное произведение объектов

KERNEL TRICK

Идея ядрового трюка состоит в том, что некоторые модели машинного обучения (в частности, линейную регрессию и SVM) можно записать в таком виде, чтобы и модель, и функционал ошибки зависели **только от скалярных произведений объектов (а не от самих объектов)**.

Пример – SVM: *переходим к новым признакам $\varphi(x)$*

- Исходная модель: $a(x, w) = \text{sign}((w, \varphi(x)) + w_0)$
- Модель можно записать в виде (двойственная запись):

$$a(x, \lambda) = \text{sign}\left(\sum_{i=1}^l \lambda_i y_i K(x_i, x) - w_0\right)$$

➤ То есть для вычисления предсказания не нужно вычислять значения новых признаков $\varphi(x)$, а достаточно уметь вычислять ядро $K(x_i, x)$.

KERNEL TRICK

Идея ядрового трюка состоит в том, что некоторые модели машинного обучения (в частности, линейную регрессию и SVM) можно записать в таком виде, чтобы и модель, и функционал ошибки зависели **только от скалярных произведений объектов (а не от самих объектов)**.

Пример – SVM: *переходим к новым признакам $\varphi(x)$*

- Исходная модель: $a(x, w) = \text{sign}((w, \varphi(x)) + w_0)$
- Модель можно записать в виде (двойственная запись):

$$a(x, \lambda) = \text{sign}\left(\sum_{i=1}^l \lambda_i y_i K(x_i, x) - w_0\right)$$

- То есть для вычисления предсказания не нужно вычислять значения новых признаков $\varphi(x)$, а достаточно уметь вычислять ядро $K(x_i, x)$.
- Таким образом, мы можем изначально задать только ядро и не задавать (даже не знать!) явный вид преобразования $\varphi(x)$ и тем самым решить проблему размерности.

МЕТОДЫ ПОСТРОЕНИЯ ЯДЕР

Теорема 1. Пусть $K_1(x, z)$ и $K_2(x, z)$ - ядра, заданные на множестве X . Тогда следующие функции являются ядрами:

$$1) K(x, z) = K_1(x, z) + K_2(x, z)$$

$$2) K(x, z) = \alpha K_1(x, z), \alpha > 0$$

$$3) K(x, z) = K_1(x, z)K_2(x, z)$$

$$4) K(x, z) = f(x)f(z), f(x) - вещественная функция на X$$

$$5) K(x, z) = K_3(\varphi(x), \varphi(z)), \varphi: X \rightarrow \mathbb{R}^n -$$

векторная функция на X , K_3 – ядро, заданное на \mathbb{R}^n .

МЕТОДЫ ПОСТРОЕНИЯ ЯДЕР

Теорема 2. Пусть $K_1(x, z), K_2(x, z), \dots$ - последовательность ядер, причем предел

$$K(x, z) = \lim_{n \rightarrow \infty} K_n(x, z)$$

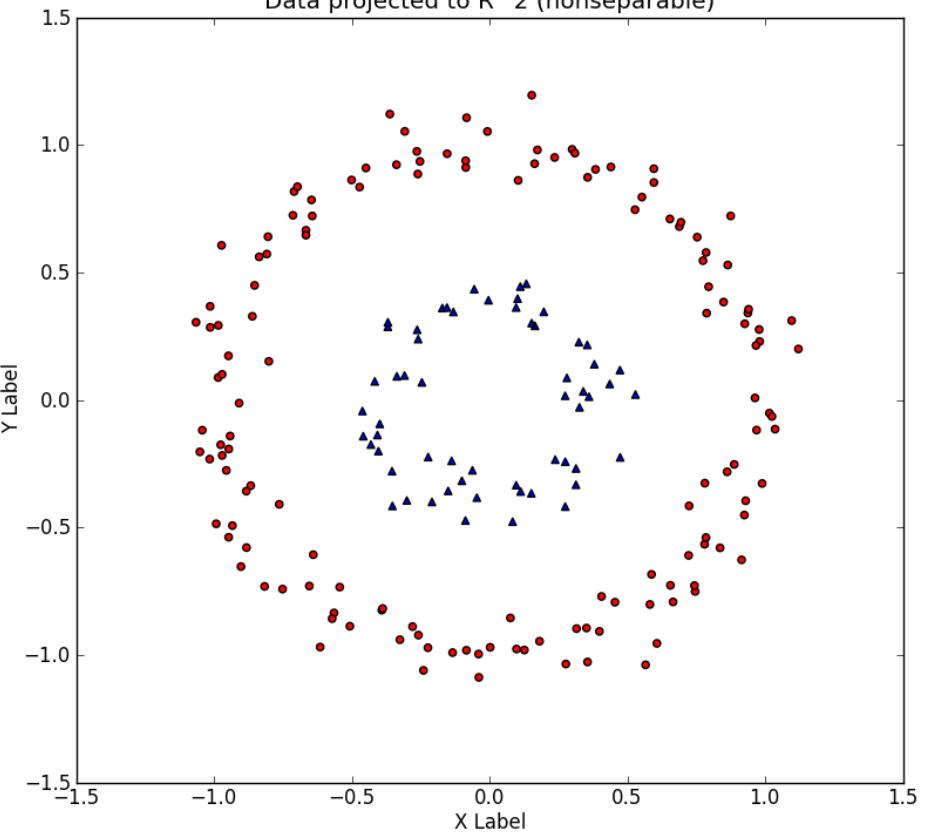
Существует для всех x и z . Тогда $K(x, z)$ ядро.

ПРИМЕРЫ ЯДЕР

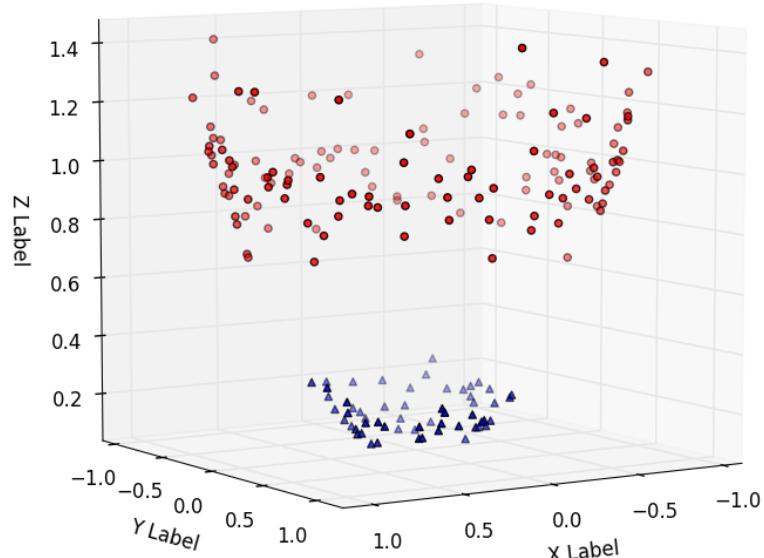
- $K(x, z) = 1$
- $K(x, z) = (x, z)$ – скалярное произведение
- $K(x, z) = (x, z)^2$, где $x = (x_1, x_2), z = (z_1, z_2)$
- $K(x, z) = \exp(-\gamma \|x - y\|^2)$ – гауссовское или радиальное ядро (RBF-ядро).
- $K(x, z) = p((x, z))$, где p – многочлен с положительными коэффициентами
- $K(x, z) = ((x, z) + R)^d, R > 0$ – полиномиальное ядро

РАДИАЛЬНОЕ ЯДРО

Data projected to R^2 (nonseparable)

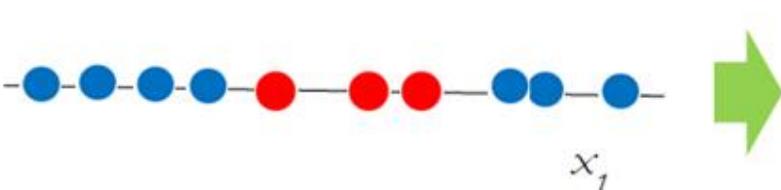


Data in R^3 (separable)

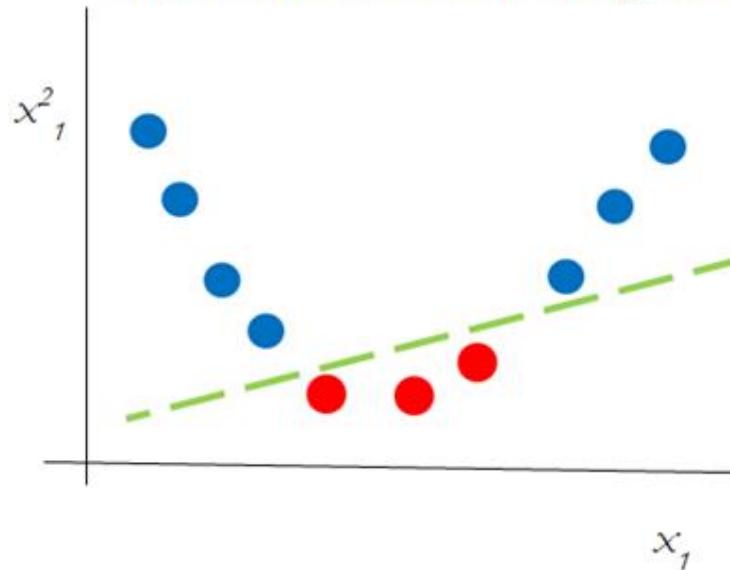


ПОЛИНОМИАЛЬНОЕ ЯДРО

*1-Dimensional Linearly
Inseparable Classes*

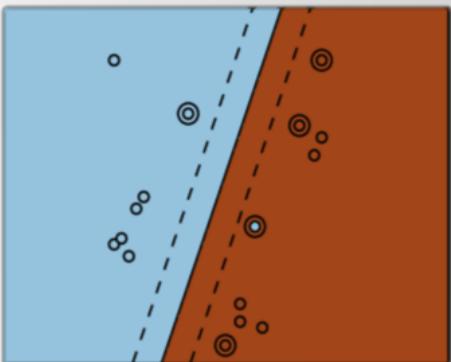


*1-Dimensional Linearly
Inseparable Classes transformed with
Polynomial Kernel of Degree 2*



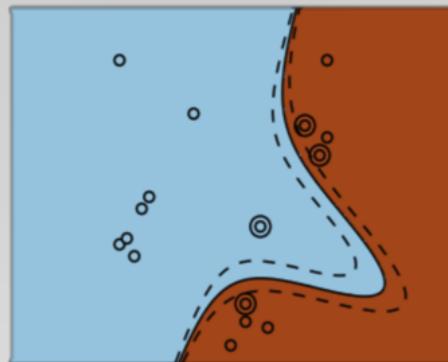
ПРИМЕР: SVM С РАЗЛИЧНЫМИ ЯДРАМИ (ДВА КЛАССА)

Linear Kernel



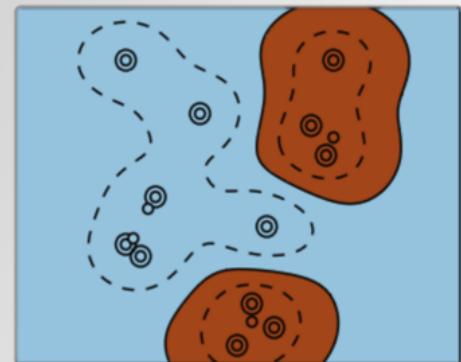
C hyperparameter

Polynomial Kernel



C plus gamma, degree and coefficient hyperparameters

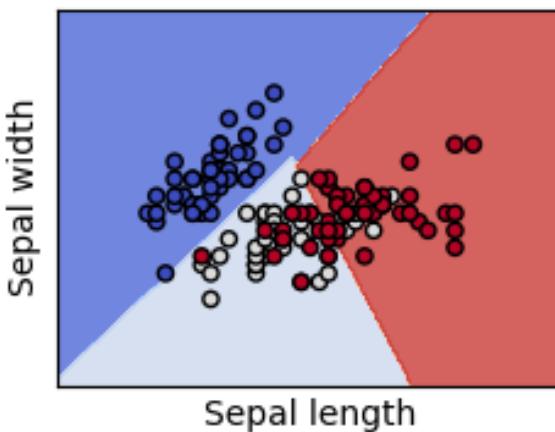
RBF Kernel



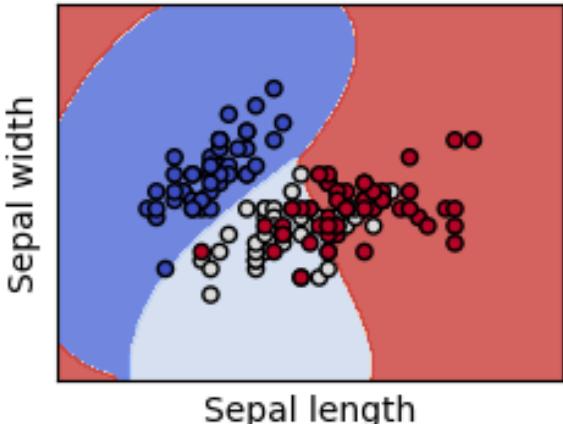
C plus gamma hyperparameter

ПРИМЕР: SVM С РАЗЛИЧНЫМИ ЯДРАМИ (ТРИ КЛАССА)

SVC with linear kernel



SVC with RBF kernel



SVC with polynomial (degree 3) kernel

