Машинное обучение и анализ данных

Постановка основных задач

Дьяконов А.Г.

Московский государственный университет имени М.В. Ломоносова (Москва, Россия)

Обучение с учителем (Supervised Learning, с размеченными данными / метками)

Обучение с учителем

$$X_{\text{train}} = \{(x_1, y_1), \ldots, (x_m, y_m)\}$$

$$y : X \to Y$$

$$y(x_1) = y_1$$

$$\ldots$$

$$y(x_m) = y_m$$

y – **целевая функция (переменная)**

target/response/outputs/dependent variable

x_i – **объект (наблюдение)**

observation, example, instance, object

X – **пространство объектов (входов)**

Y – **пространство меток / значений целевого признака (выхода)**

Цели

1. Восстановление целевой зависимости

Уметь восстанавливать метки новых объектов $y(x)$

– найти зависимость целевой переменной от остальных

2. Интерпретация

Как устроена $y(x)$

3. Оценка качества полученного решения

(например, как часто ошибаемся, насколько)

Типы задач обучения с учителем

Классификация

$$|Y| = k << \infty$$

бинарная

$Y = \{0, 1\}$ **или** $Y = \{-1, +1\}$

скоринговая бинарная

$Y = [0, 1]$

на k непересекающихся классов

$Y = \{1, 2, \ldots, k\}$

на k пересекающихся классов

$Y = \{0, 1\}^k$

Регрессия

$$Y = \mathbb{R}$$

многомерная

$$Y = \mathbb{R}^n$$

Прогнозирование – задачи в которых есть признак время, алгоритм обучается на данных из прошлого, а работает на данных из будущего

Ранжирование

Y **– ЧУМ**

Пространство объектов

Практически какое угодно:

- **медицинские истории**
- **тексты**
- **сигналы / временные ряды / последовательности**
- **изображения**
- **векторы / множества / графы**
- **…**

Для удобства-простоты-теории-практики:

$$X = \mathbb{R}^n$$

n-мерное признаковое пространство

$x_i = (x_{i1}, \ldots, x_{in})$ **– объект в признаком описании**

inputs, attributes, repressors, properties, covariates, features, variables

Задача в признаковой постановке

матрица «объект-признак» (data matrix)

плохой_клиент	линии	возраст	поведение_30-59_дней	Debt_Ratio	доход	число_кредитов
0	0.111673	46	0	1.329588	800.0	8
0	0.044097	69	0	0.535122	3800.0	10
0	0.047598	77	0	0.169610	3000.0	7
0	0.761149	58	1	2217.000000	NaN	4
0	0.690684	55	0	0.432552	12416.0	7

По строкам – признаковые описания объектов

по столбцам – значения конкретных признаков

Извлечение признаков: $X \to \mathbb{R}^n$

м.б. производится автоматически

чем лучше генерация признаков, тем более простое ML нужно;)

Признаки (features)

Задачи классификации – целевой признак категориальный

Задачи регрессии – целевой признак вещественный

м.б. графом!

Замечание

Целевой признак «условен».

Часто просто дана матрица

(целевой признак приходится формировать)

mmp@cs.msu.ru → **длина = 3**

доменов = 3

«1 уровень=ru» = 1

«1 уровень=com» = 0

«1 уровень=org» = 0

Примеры

Классификация спама

X – письма

Y = {спам, норма}

признаки = длина письма, число вхождений слова, отправитель, …

Медицинская диагностика

X – пациенты

Y – диагнозы

признаки = результаты анализов, возраст, пол и т.п.

вариант постановки: предсказать вероятности болезней $Y = [0,1]^l$

Прогнозирование цен акций

X – ситуация на рынке

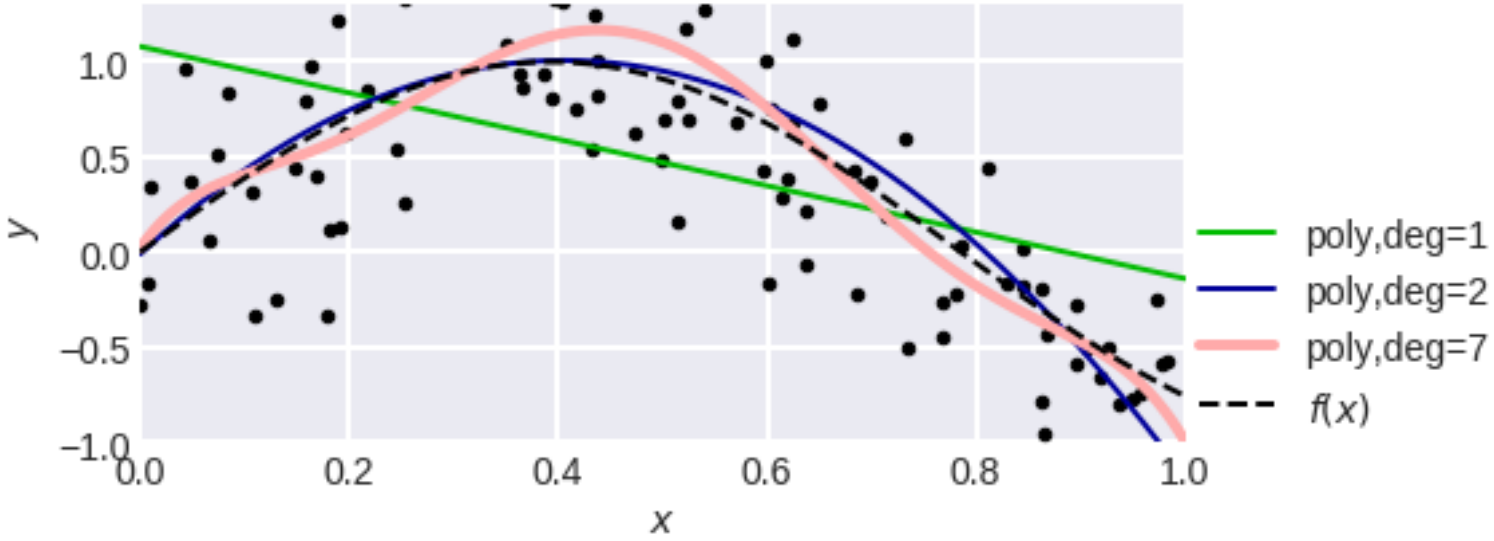
Y – цена на акцию через час

вариант постановки: множественная регрессия –

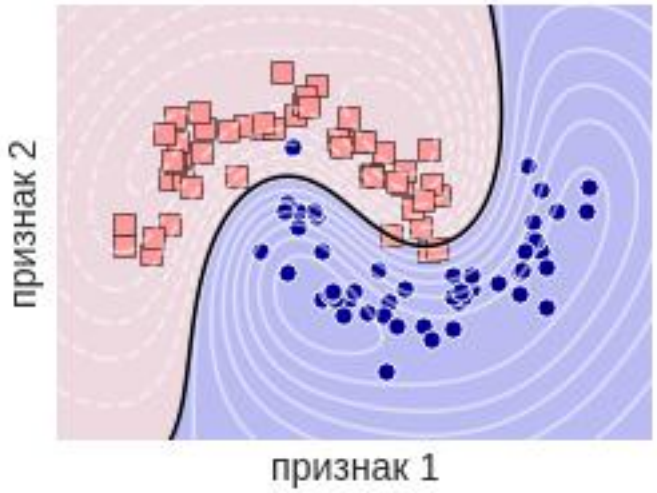
– цены нескольких акций $Y = \mathbb{R}^l$

Визуализация задач

Задача регрессии



Задача классификации



Что значит «восстановление целевой зависимости» (меток)

Строим «алгоритм» (гипотезу) $a(x)$**,**

который выдаёт предполагаемые метки

Формализация качества:

$L(y,a)$ **– функция ошибки (error / loss function)**

ошибка на объекте x

$$L(y(x),a(x))$$

$a(x)$ **– ответ нашего алгоритма** a

Примеры:

в задаче регрессии – $L(y,a)=|y-a|$

в задаче классификации – $L(y,a)=I[y=a]$

Что значит «восстановление целевой зависимости» (меток)

Если объекты имеют вероятностную природу, то

$$\int_{X\times Y} L(y(x),a(x))\partial P \to \min$$

На практике не знаем меры

можем вычислить лишь «эмпирический риск»

Обучающая выборка (обучение – не путать с процессом)

$$X_{\text{train}} = \{(x_1,y_1),\ldots,(x_m,y_m)\}$$

Ошибка на выборке (один из вариантов):

$$L(a,X_{\text{train}}) = \frac{1}{m}\sum_{i=1}^{m} L(y(x_i),a(x_i))$$

$$a^* = \arg\min L(a,X_{\text{train}})$$

На самом деле, интересна не ошибка на обучении (Training Error)!

Как минимизируется ошибка

Минимизация производится в рамках модели

Модель – параметрическое семейство алгоритмов

$$A = \{a(x;w)\}_{w \in W}$$

пример: $A = \{a(x;w) = w^{\mathrm{T}} x : \mathbb{R} \to \mathbb{R}\}_{w \in \mathbb{R}^n}$

Обучение – определение параметров алгоритма, как правило, производится с помощью оптимизации значения функции ошибки (функционала качества) или их модификаций на обучающей выборке

По сути, интеллектуальный перебор алгоритмов…

Как – дальше!

Обобщающая способность (Generalization)

Какое качество (ошибка) алгоритма на новых данных?

$$L(a, X_{\text{train}}) \vee L(a, X_{\text{test}})$$

Ошибка на тестовой выбороке (Generalization Error / Test Error)

более строго: матожидание ошибки на новых данных

обучение ≠ запоминание

потом: недообучение, переобучение, сложность…

потом: отложенная выборка, контроль и т.п.

Что такое алгоритм

Мы под этим понимаем функцию

$$a(x): X \to Y,$$

которую можно эффективно реализовать в виде программы

1. Допускает вычисление за приемлемое время
2. Использует ограниченный набор ресурсов
3. Есть специфика, связанная с вычислениями на компьютере

Требования к модели

- **Качество (Predictive Accuracy)** — см. выше
- **Эффективность (Efficiency)** — время обучения и использования
- **Робастность (Robustness)** — устойчивость к шуму/пропускам …
- **Масштабируемость (Scalability)** — использование при увеличении объёма данных
- **Интерпретируемость (Interpretability)** — объяснение результатов модели
- **Компактность (Compactness)** — затраты на хранение модели

Почему МО не оптимизация

1. Не знаем меру в

$$\int_{X\times Y} l(y(x),a(x))\partial P \to \min$$

т.е. решаем «неправильную задачу оптимизации»
и правильный выбор неправильности – особое умение
(регуляризация, проблемно-ориентированные модели и т.п.)

2. Оптимизация не в классе функций, а в классе алгоритмов
дополнительные требования на решение

Схема решения задачи

1. Уточнение и постановка задачи

понимание бизнес-задачи

понимание исходных данных

подготовка данных для модели

2. Выбор

- **Алгоритма**
  - модели
  - способа обучения (метапараметры, методы оптимизации)
- **Контроля**
  - функции ошибки
  - способа контроля (разбиение train/test/valid)
- **Признаков**
  - генерация
  - селекция

3. Обучение

4. Предсказание → Проверка качества

5. Deploy / Release

(алгоритм переобучается на всех данных)

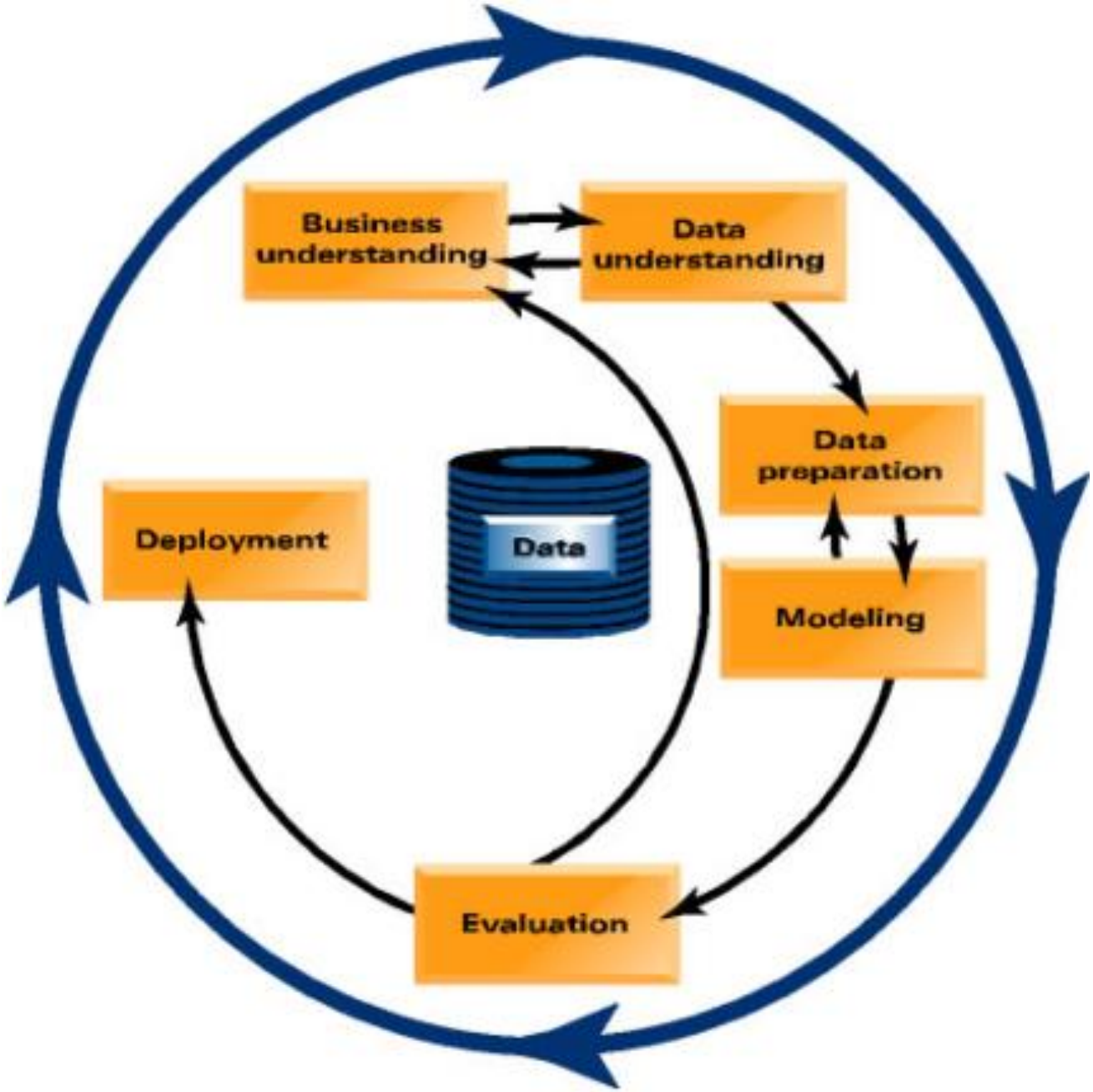
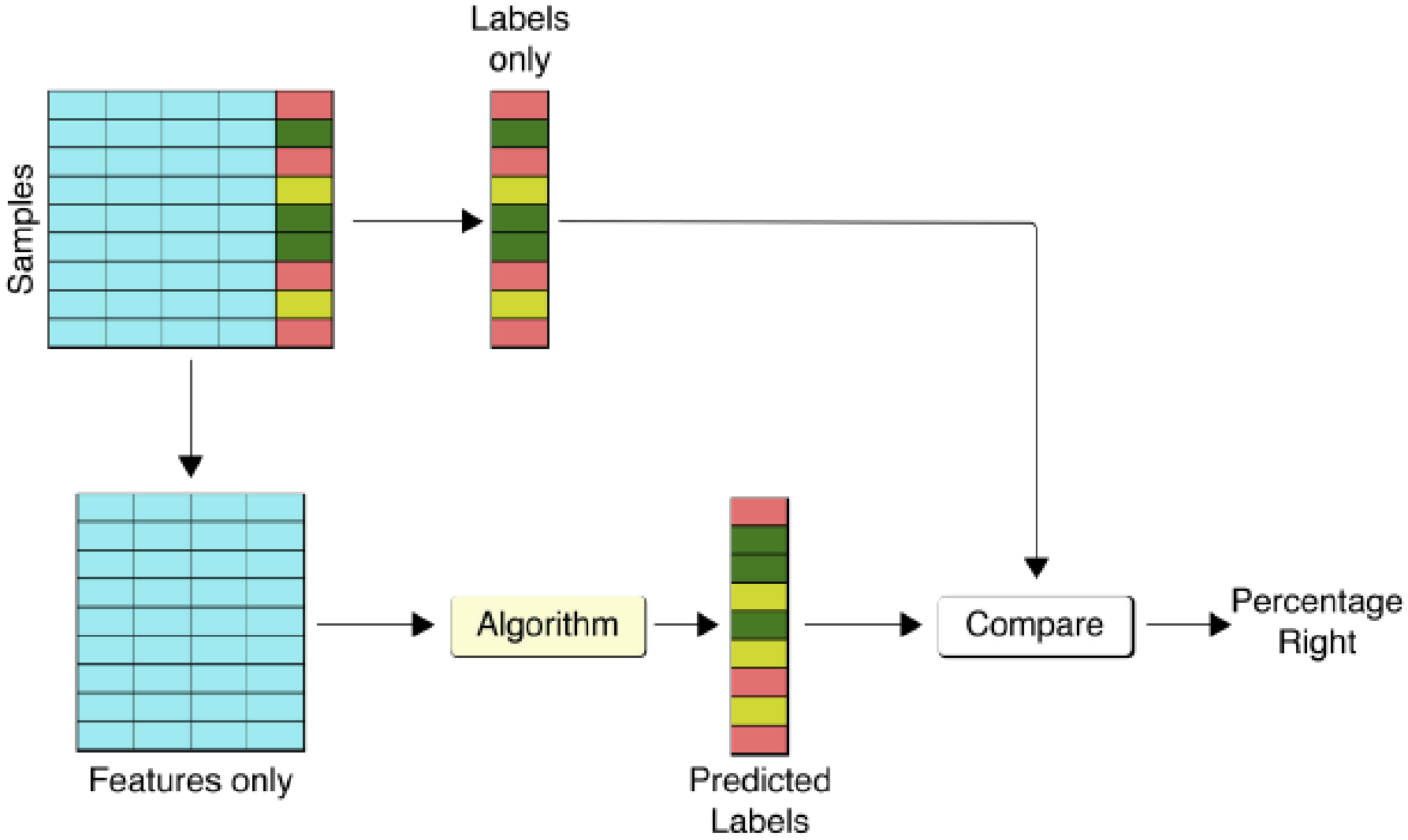

Business understanding
Data understanding
Data preparation
Modeling
Evaluation
Deployment
Data

Схема проверки алгоритма



[Glassner]

Как решаются задачи

Пусть $y = f(X_1,\ldots,X_n)+\varepsilon$

y **– продажи,**

X_1 **– затраты на рекламу по TV,**

X_2 **– затраты на рекламу в Интернете,**

X_3 **– затраты на рекламу на радио, и т.д.**

Надеемся

$$a(X_1,\ldots,X_n) \approx f(X_1,\ldots,X_n),$$

a **~ алгоритм (алгоритмически реализуемая функция)**

Ищем в параметризованном семействе $A \in \{A\}$ **(модели)**

ε **– неустранимая ошибка (irreducible error)**

Подход основанный на близости

$$a(x) = \text{mean}(y_i \mid x_i = x)$$

но если в обучающей выборке нет именно таких объектов

$$a(x) = \text{mean}(y_i \mid x_i \in N(x))$$

$N(x)$ **– окрестность (neighborhood) объекта** x

(похожие на него объекты)

Параметризация может определять размер окрестности

Но что такое окрестность при больших размерностях…

curse of dimensionality (след. лекция)

Параметрические модели

Линейная модель

$$a(X_1,\ldots,X_n) = w_0 + w_1X_1 + \ldots + w_nX_n$$

Параметры оцениваются с помощью подгонки на данных обучения (fitting the model to training data)

$$a(x_i) = y_i,\ i = 1,2,\ldots,m$$

w_i **– веса (weights) / параметры (parameters) модели**

здесь w_0 **– смещение (bias)**

Линейная модель – простая,

можно усложнить – полиномиальная модель.

$$a(X_1,\ldots,X_n) = w_0 + \sum_t w_tX_t + \ldots + \sum_{i,j} w_{ij}X_iX_j$$

Переобучение

Чем сложнее модель, тем проще настроиться на данные, но возникает проблема – переобучение (overfitting) – качество на контроле существенно ниже чем на обучении

Линейная модель хорошо интерпретируемая

- легко объяснить, как работает
- легко объяснить, почему получен такой ответ

⇐ простая, небольшое число переменных

простая ⇒ надёжная

(оценка ошибки, как правило, соответствует действительности)

Пример задачи машинного обучения

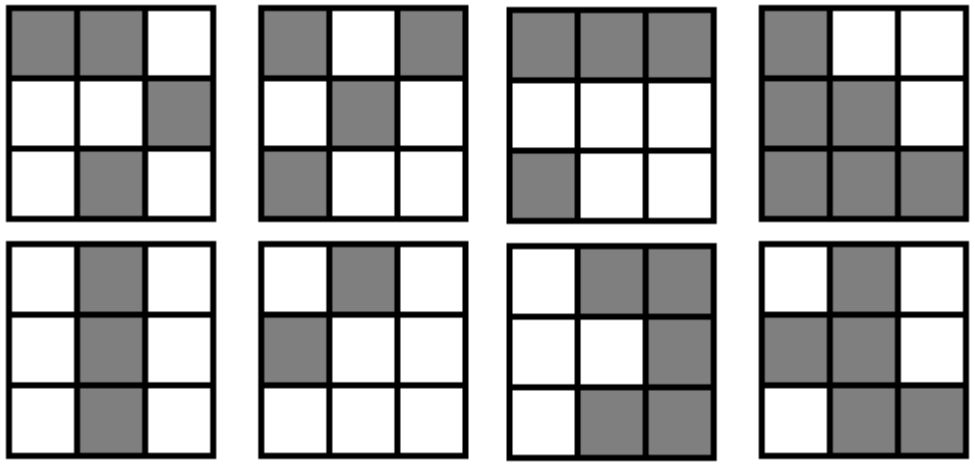
	x_0	x_1	x_2	x_3	x_4	y
0	1.5	7.4	2.6	5.3	0.1	3.8
1	9.2	9.0	0.3	9.6	1.4	6.2
2	2.8	6.1	9.4	8.5	0.0	6.1
3	5.2	5.5	4.9	7.7	1.6	5.2
4	7.6	0.2	1.4	1.2	3.1	3.1
5	6.7	4.7	8.2	2.9	7.3	6.5
6	7.0	3.3	3.3	9.8	6.2	4.5
7	9.5	7.7	8.3	4.1	4.5	8.5
8	4.0	10.0	1.8	9.6	4.2	5.3
9	4.2	4.6	3.7	4.7	0.4	4.2

Как зависит целевая переменная от остальных?

Пример задачи машинного обучения

класс 1

класс 2



Как определяется класс?

Другие виды обучения

Обучение без учителя (unsupervised Learning)

с неразмеченными данными, без меток

Обучение без учителя

$$X_{\text{train}} = \{x_1, \ldots, x_m\} \subseteq X$$

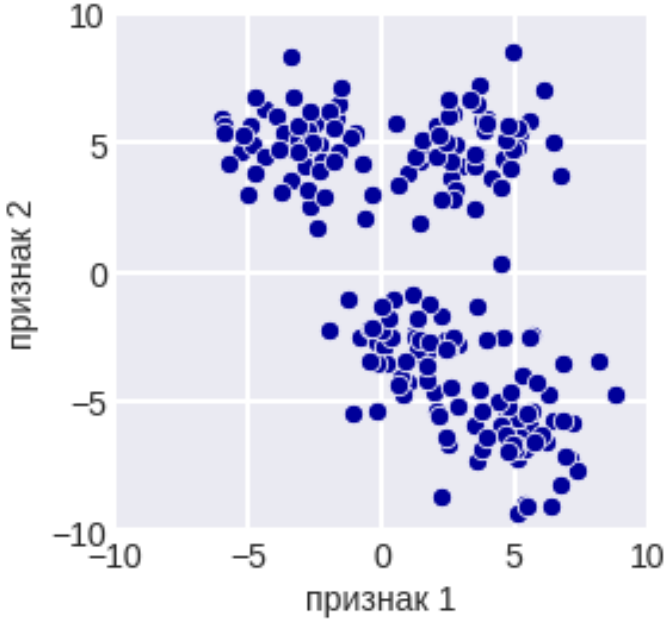
Понять «структуру» пространства X

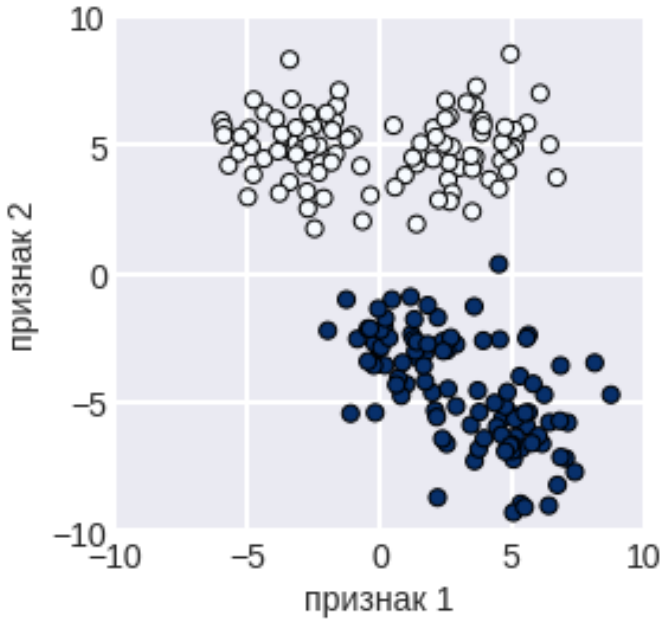
Как на нём распределены объекты?

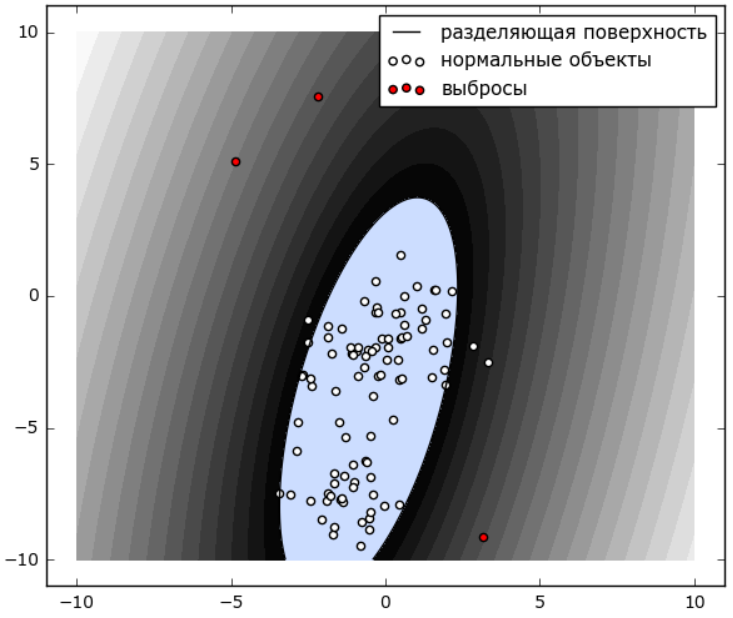
Можно ли его разделить на подпространства похожих объектов?

Можно ли эффективно описать объекты/пространство?

Часто нет понимания, насколько хорошо решается задача





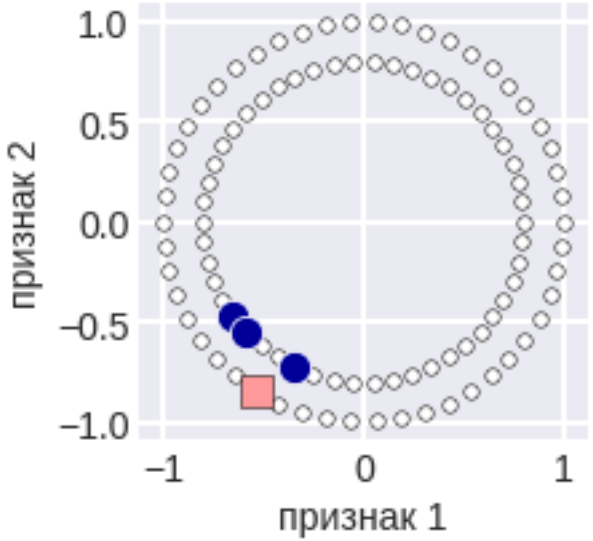


Обучение с частично размеченными данными (Semi-Supervised Learning)

$$X_{\text{train}} = \{(x_1, y_1), \ldots, (x_k, y_k), x_k, \ldots, x_m\}$$

Если заранее известна контрольная выборка $x'_1, \ldots, x'_q$,

то это трансдуктивное обучение (transductive learning)



Другие виды обучения

Обучение с подкреплением (Reinforcement Learning)

обучение агента, который взаимодействует со средой и получает награду за взаимодействие

Структурный вывод (Structured output)

на выходе набор значений со связями между ними, примеры:

- **Грамматический разбор (parsing):** текст $\rightarrow$ дерево
- **Аннотирование изображений (Image Captioning):** изображение $\rightarrow$ текст
- **Транскрипция (Transcription):** X $\rightarrow$ текст
- **Машинный перевод (Machine translation):** текст $\rightarrow$ текст

Синтез: выборка $\rightarrow$ выборка

Другие виды обучения

Активное обучение (Active Learning)

влияем на формирование обучающей выборки

Онлайн-обучение (Online Learning)

в каждый момент времени нам доступна небольшая группа объектов (м.б. один объект)

Transfer/Multitask Learning

решение новых задач с помощью решения старых

Обучение (выучивание) признаков (Feature Learning)

автоматическое получение хороших признаков из сырых данных

Другие виды обучения

Заполнение пропусков (Imputation of missing values)

выборка $\rightarrow$ выборка

Устранение шума (Denoising)

объект $\rightarrow$ объект

Сложности в ML

- **переобучение – основная теоретическая проблема**

- **проблема формализации**

надо переформулировать бизнес-задачу в математическую задачу выявления зависимости, выбор адекватного функционала качества

- **размеры данных**

много объектов (низкого уровня – транзакций, высокого – клиентов)

много признаков (обработка текстов)

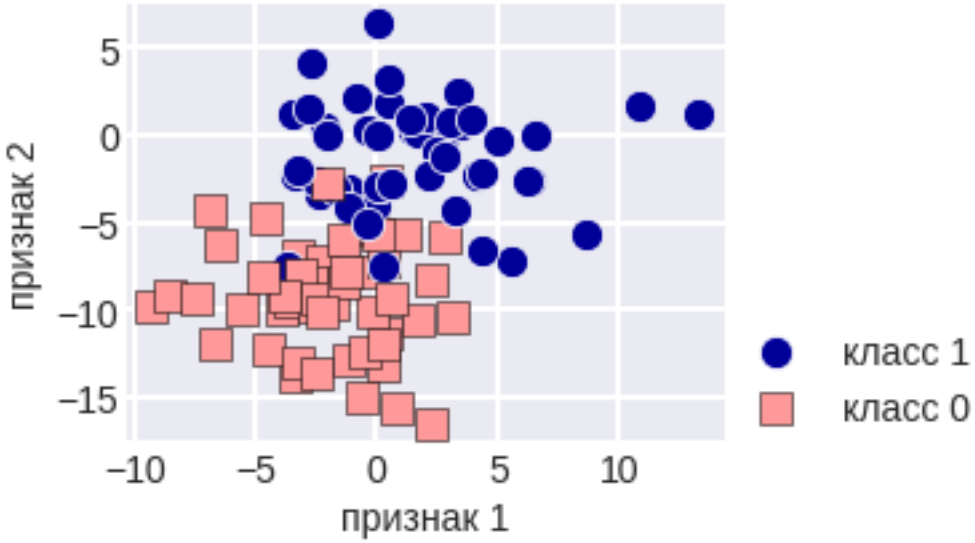
- **качество данных**

невыполнение всех свойств (полнота, корректность, правдивость, ясность и т.п.)

- **несоответствие обучения и контроля**

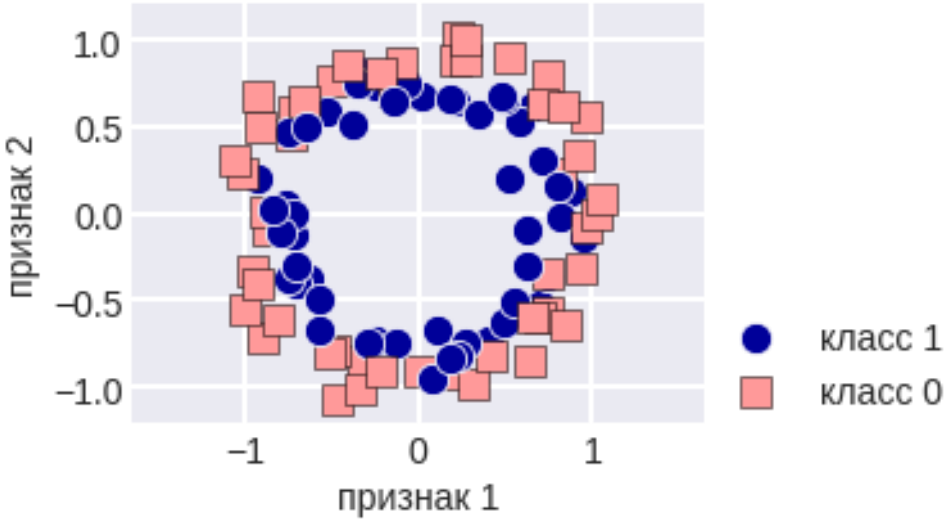
это больше, чем проблема репрезентативности выборки – это проблема прогноза / адаптации (распознавание голоса, спама)

Примеры модельных задач



«Кучки»

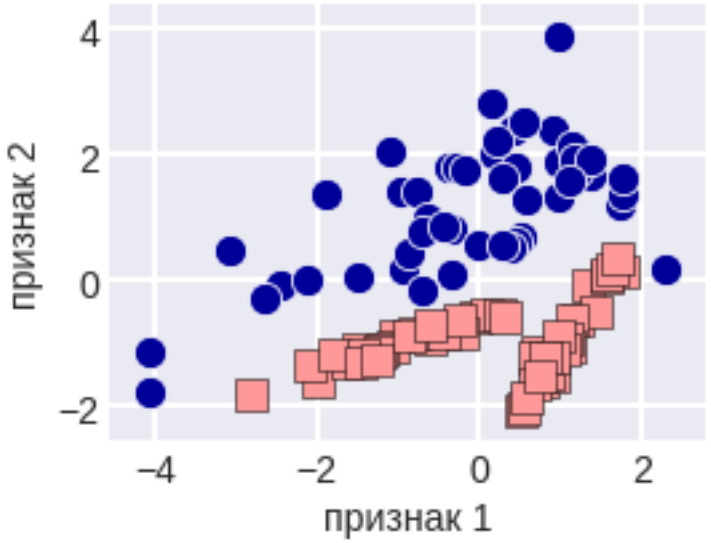
```
from sklearn.datasets import
make_blobs

X, y = make_blobs(centers=2,
                  random_state=2)

plt.scatter(X[:, 0], X[:, 1],
            c=y, s=75)
```



«Кольца»

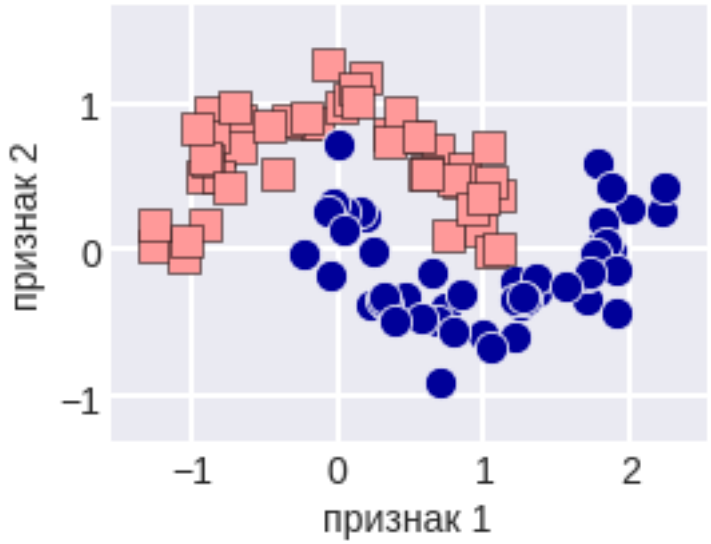
```
from sklearn.datasets import
make_circles

X, y = make_circles(noise=0.1,
                    random_state=1)
```

Примеры модельных задач



«Классификация»

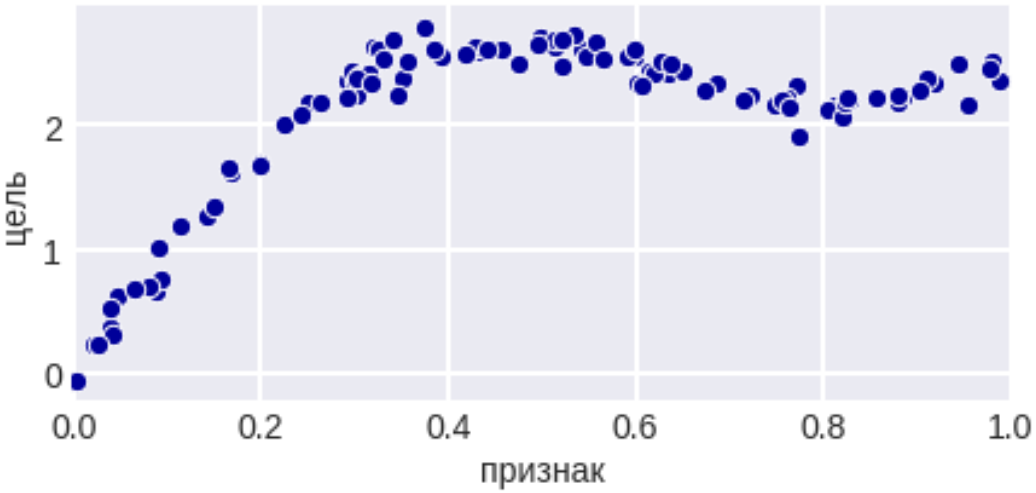
```
make_classification(n_samples=100,
                    n_features=2,
                    n_informative=2,
                    n_redundant=0,
                    n_repeated=0,
                  n_clusters_per_class=2)
```



Два месяца

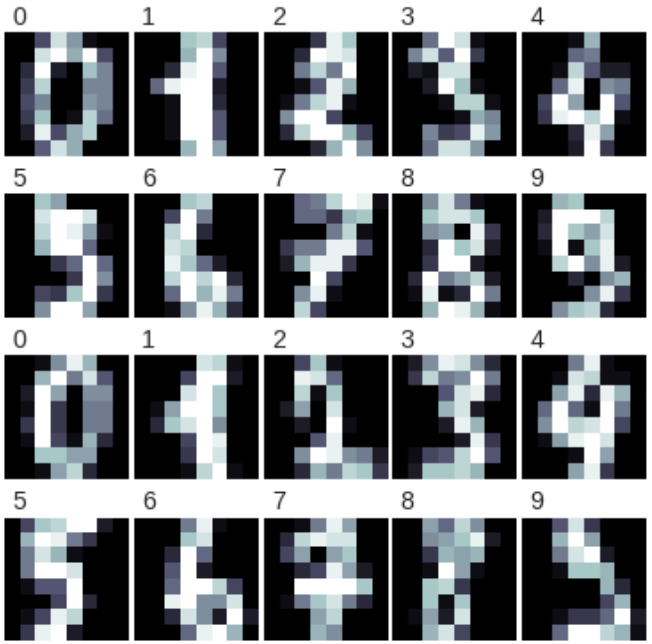
```
make_moons(n_samples=100,
           noise=0.15,
           random_state=1)
```

Ручная генерация данных




```
n_samples = 100
np.random.seed(10)
X = np.random.rand(n_samples)
Y = np.sin(5 * X) + 5 * np.log1p(X)
 + 0.1 * np.random.randn(n_samples)
```

Классические датасеты




```
from sklearn.datasets import load_digits
digits = load_digits()
X_digits, y_digits = digits.data, digits.target
```

Ссылки

Andrew Glassner Deep Learning, Vol. 1-2: From Basics to Practice // http://www.glassner.com/portfolio/deep-learning-from-basics-to-practice/

использована

- **лекция «Библиотека языка Питон Scikit-Learn»**
  https://github.com/Dyakonov/IML/blob/master/IML2018_06_scikitlearn_10.pdf