Основные понятия машинного обучения. Введение в NLP.

Елена Кантонистова

ВШЭ, 2024

HSE University
Faculty of Computer science

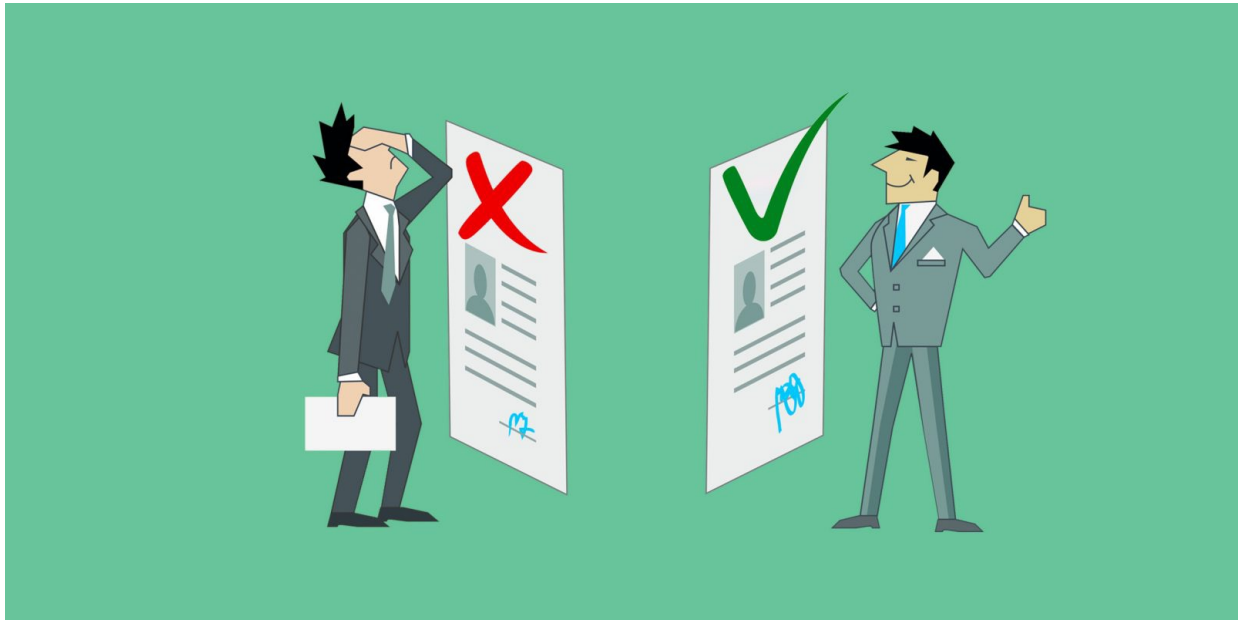
План лекции

- Основные понятия машинного обучения
- Типы задач
- Обучение модели
- Оценка качества модели
- Полный цикл проекта по анализу данных
- Введение в NLP
- Классические модели классификации и регрессии

1. Основные понятия машинного обучения

Пример: задача скоринга

- Пусть по характеристикам клиента (пол, возраст, средний доход, рейтинг кредитной истории и так далее) мы хотим предсказать, **вернёт клиент кредит или не вернёт**.

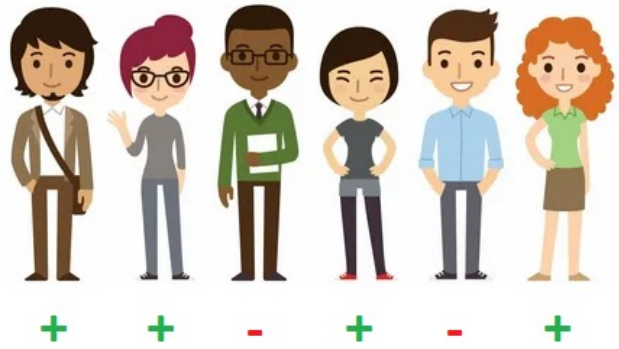


Пример: задача скоринга

- **Целевая переменная (target)**, то есть величина, которую хотим предсказать - это число (например, 1 - если человек вернет кредит, и 0 иначе).
- Характеристики клиента, а именно, его пол, возраст, доход и так далее, называются **признаками (features)**.
- Сами же клиенты - сущности, с которыми мы работаем в этой задаче - называются **объектами (objects)**.

Обучение алгоритма

- На **этапе обучения** происходит анализ большого количества данных, для которых у нас имеются правильные ответы (например, клиенты, про которых мы знаем - вернули они кредит или нет; пациенты и их анализы, где про каждого пациента мы знаем, болен он или здоров и так далее).

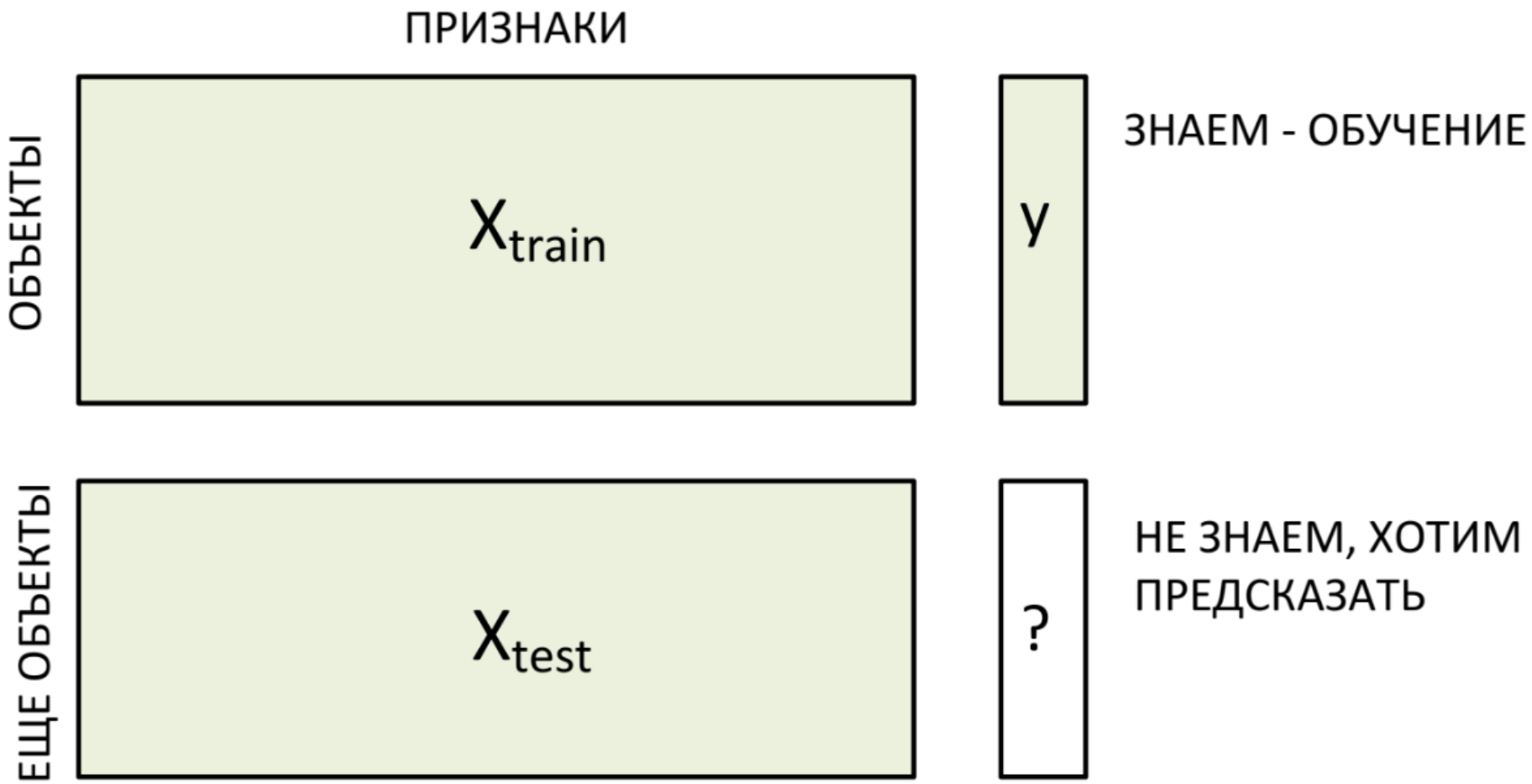


- Модель машинного обучения изучает эти данные и старается научиться делать предсказания таким образом, чтобы для каждого объекта предсказывать как можно более точный ответ. Все данные с известными ответами называются **обучающей выборкой**.

Применение алгоритма

- На **этапе применения** готовая (уже обученная) модель применяется для того, чтобы получить ответ на новых данных. Например, у нас есть подробная информация о клиентах, и мы применяем модель, чтобы она предсказала, кто из них вернет кредит, а кто нет.

Этапы машинного обучения



2. Типы задач в ML

Типы задач в ML

Что такое задача классификации?

Что такое задача регрессии?

Типы задач в ML: Классификация

- В задачах **классификации** целевая переменная - это класс объекта. То есть в задачах классификации ответ может быть одним из конечного числа классов.

Примеры:

- пол клиента (мужчина или женщина)
- уйдет клиент из компании или нет
- вернет человек кредит или нет
- болен пациент или здоров и т. д.

Примеры задач классификации

- Задачи медицинской диагностики (пациент здоров или болен)
- Задачи кредитного скоринга (выдаст банк кредит данному клиенту или нет)
- Задача предсказания оттока клиентов (уйдет клиент в следующем месяце или нет)
- Предсказание поведения пользователя (кликнет пользователь по данному баннеру или нет)
- Классификация изображений (на изображении кошка или собака)

Типы задач в ML: Регрессия

В задачах **регрессии** целевая переменная может принимать бесконечно много значений. Например, прибыль фирмы может быть любым числом (как очень большим, так и очень маленьким) - даже отрицательным или нецелым.

Примеры задач регрессии

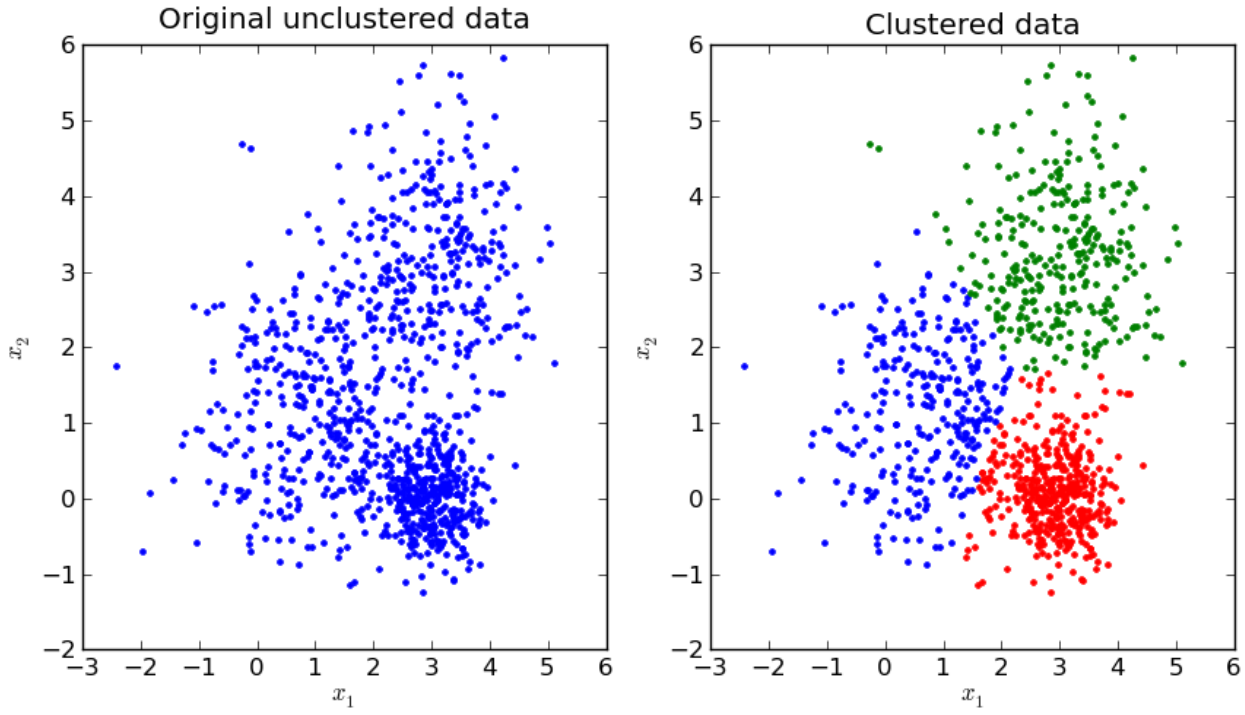
- Предсказание стоимости недвижимости (стоимость квартиры в Москве)
- Предсказание прибыли ресторана
- Предсказание поведения временного ряда в будущем (стоимость акций)
- Предсказание зарплаты выпускника вуза по его оценкам

Типы задач в ML

Какие еще типы задач в ML вы знаете?

Типы задач в ML: кластеризация

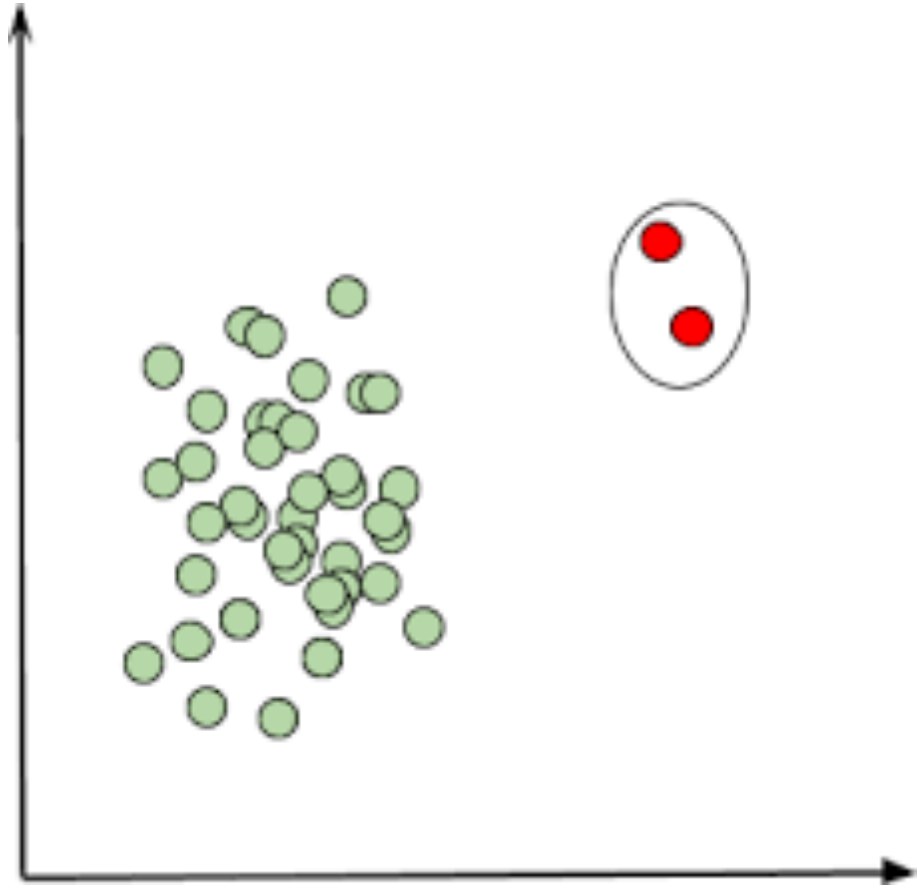
Кластеризация – задача разделения объектов на группы, при этом где целевые переменные для объектов неизвестны (или не существуют). Разделение происходит только на основе признаковых описаний объектов.



Другие типы задач в ML

- Ранжирование
- Снижение размерности
- Поиск аномалий
- Генерация
- Визуализация

И другие.



Типы задач машинного обучения

- Если нам известны значения целевой переменной, то есть алгоритм обучается так, чтобы правильно предсказывать целевую переменную – это **обучение с учителем**. Сюда относят классификацию, регрессию и ранжирование.
- Если нам неизвестны значения целевой переменной или целевая переменная вообще отсутствует, то есть алгоритм обучается только по признакам объектов, то это **обучение без учителя**. Примерами обучения с учителем являются кластеризация, понижение размерности и др.

3. Обучение модели

Обучение алгоритма

Предположим, что мы хотим предсказать *стоимость дома* y по его *площади* (x_1) и *количеству комнат* (x_2).

Обучение алгоритма

Предположим, что мы хотим предсказать *стоимость дома* y по его *площади* (x_1) и *количеству комнат* (x_2).

Как правило, алгоритм $a(x)$ выбирают из некоторого семейства алгоритмов A.

Обучение алгоритма

Предположим, что мы хотим предсказать *стоимость дома* y по его *площади* (x_1) и *количеству комнат* (x_2).

Как правило, алгоритм $a(x)$ выбирают из некоторого семейства алгоритмов A.

Используем линейную модель для предсказания стоимости. Она будет выглядеть так:
$$\boldsymbol{a(x) = w_0 + w_1 x_1 + w_2 x_2},$$

где w_0, w_1, w_2 -

параметры модели (*веса*).

Обучение алгоритма

Предположим, что мы хотим предсказать *стоимость дома* y по его *площади* (x_1) и *количеству комнат* (x_2).

Как правило, алгоритм $a(x)$ выбирают из некоторого семейства алгоритмов A.

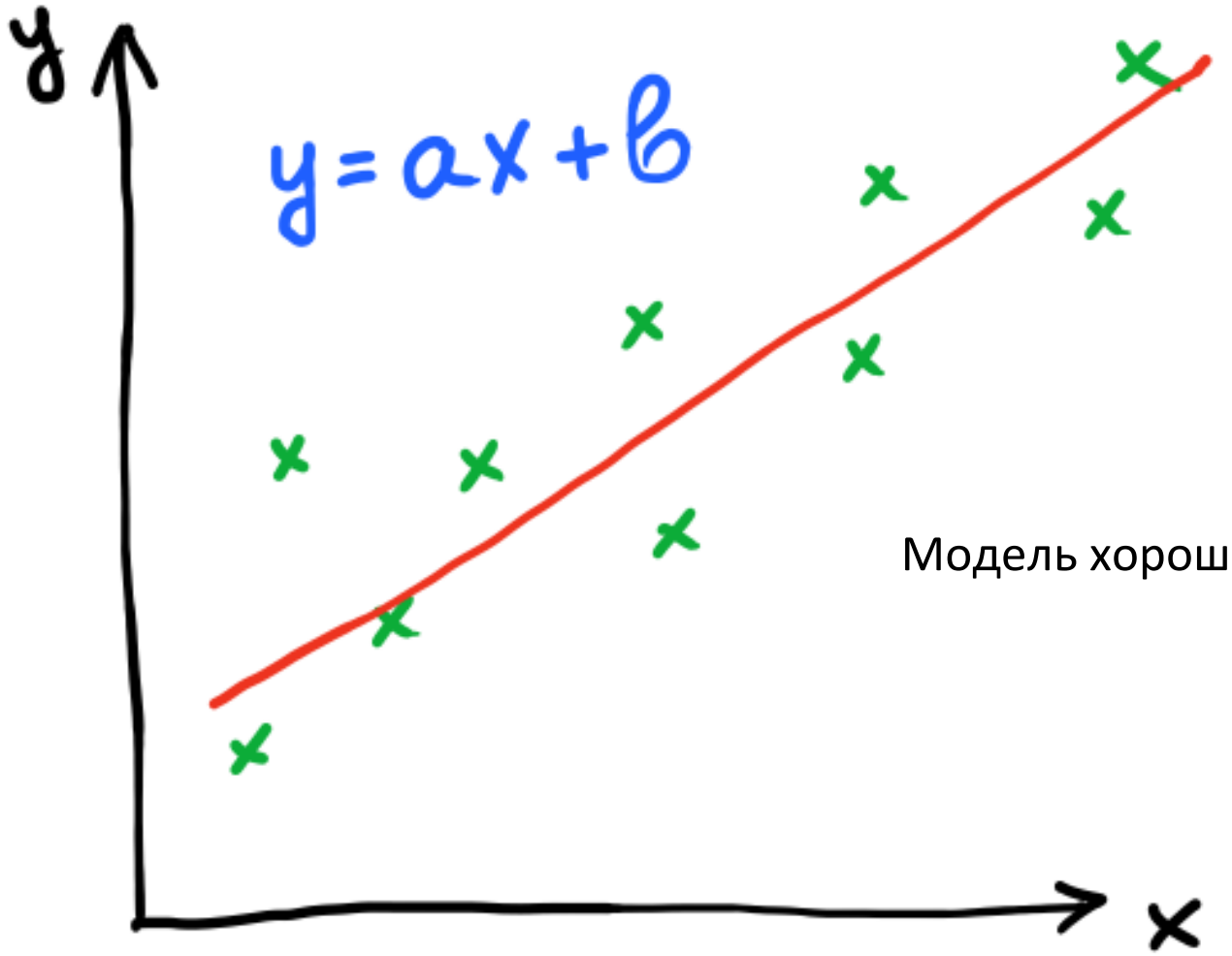
Используем линейную модель для предсказания стоимости. Она будет выглядеть так:
$$\boldsymbol{a(x) = w_0 + w_1 x_1 + w_2 x_2},$$

где w_0, w_1, w_2 -

параметры модели (*веса*).

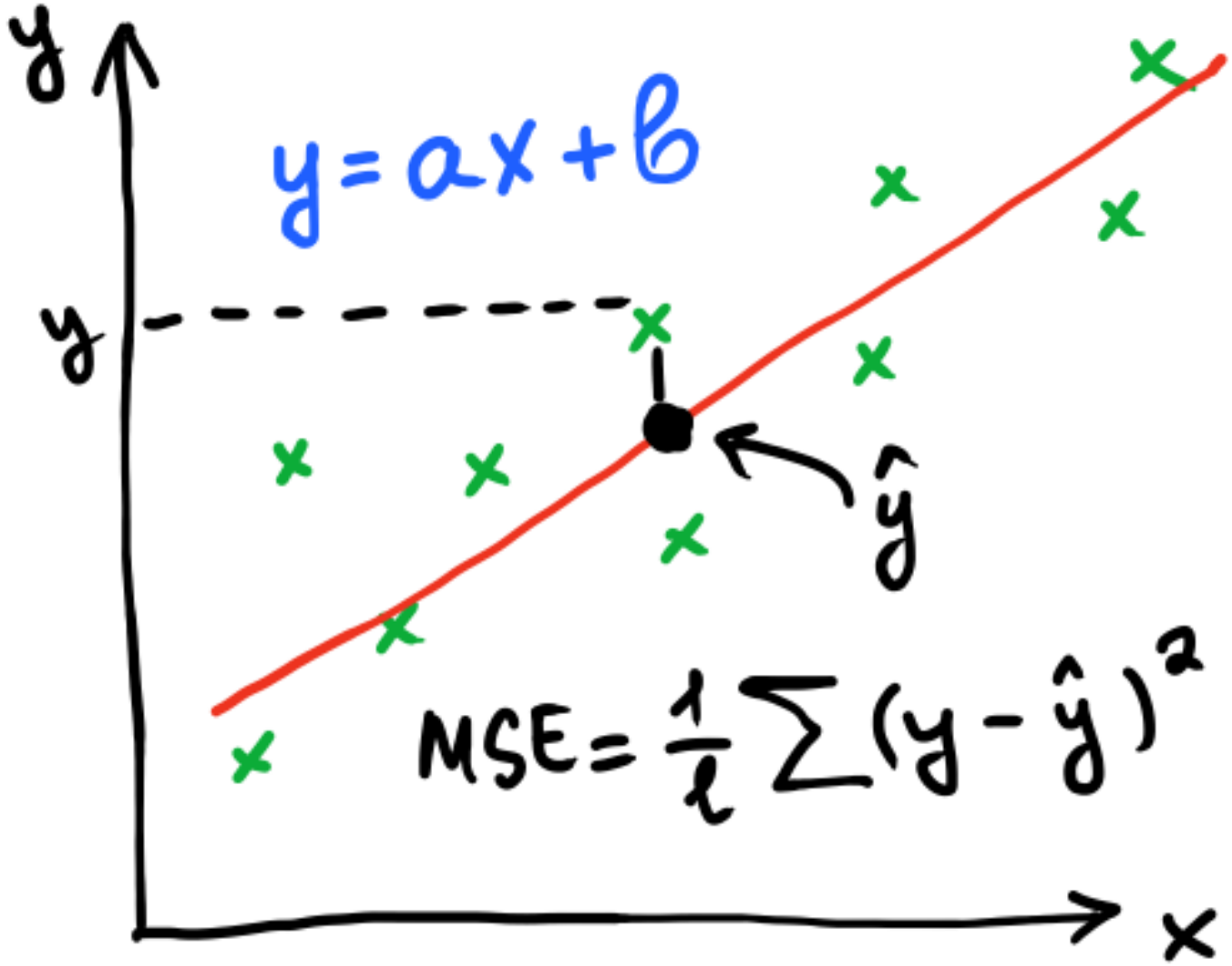
Общий вид линейных моделей:
$$\boldsymbol{A = \{a(x) = w_0 + w_1 x_1 + \cdots + w_d x_d | w_0, w_1, \ldots, w_d \in \mathbb{R}\}}$$

Обучение алгоритма



Модель хорошая?

Обучение алгоритма



Функционал ошибки

Как измерить ошибку алгоритма на всех объектах выборки?

Функционал ошибки – функционал, измеряющий качество работы алгоритма.

Пример (среднеквадратичная ошибка, MSE):

$$Q(a, X) = \frac{1}{l}\sum_{i=1}^{l}(a(x_i) - y_i)^2$$

X – объекты, l – количество объектов

a – алгоритм, $a(x_i)$ – ответ алгоритма на объекте x_i

y_i - истинные ответы

Функционал ошибки

Функционал ошибки – функционал, измеряющий качество работы алгоритма.

Пример (среднеквадратичная ошибка, MSE):

$$Q(a, X) = \frac{1}{l}\sum_{i=1}^{l}(a(x_i) - y_i)^2 \to \boldsymbol{min}$$

X – объекты, l – количество объектов

a – алгоритм, $a(x_i)$ – ответ алгоритма на объекте x_i

y_i - истинные ответы

При обучении алгоритма мы минимизируем функционал ошибки.

Обучение алгоритма

Пример (семейство линейных моделей):

$$A = \{a(x) = w_0 + w_1 x_1 + \cdots + w_d x_d | w_0, w_1, \ldots, w_d \in \mathbb{R}\}$$

Функционал ошибки:

$$Q(a, X) = \frac{1}{l} \sum_{i=1}^{l} (a(x_i) - y_i)^2$$

Функционал ошибки для линейной модели стоимости дома:

$$Q(a, X) = \frac{1}{l} \sum_{i=1}^{l} (w_0 + w_1 x_1 + w_2 x_2 - y_i)^2$$

Обучение алгоритма

Параметры w_0, w_1, w_2 подбираются так, чтобы на них достигался минимум функции потерь (на обучающей выборке):

Функционал ошибки для линейной модели стоимости дома:

$$Q(a, X) = \frac{1}{l} \sum_{i=1}^{l} (w_0 + w_1 x_1 + w_2 x_2 - y_i)^2 \to \min_{w_0, w_1, w_2}$$

Обучение алгоритма

Процесс поиска оптимального алгоритма (оптимального набора параметров или *весов*) называется **обучением**.

4. Оценка качества модели

4. Оценка качества модели

Чем отличается функция потерь от метрики качества?

Оценка качества модели

В задачах машинного обучения для оценки качества моделей и сравнения различных алгоритмов используются ***метрики качества***.

Метрики качества

В задачах машинного обучения для оценки качества моделей и сравнения различных алгоритмов используются ***метрики качества***.

Примеры:

- Корень из среднеквадратичной ошибки – для регрессии

$$RMSE(a, X) = \sqrt{\frac{1}{l}\sum_{i=1}^{l}(a(x_i) - y_i)^2}$$

Метрики качества

В задачах машинного обучения для оценки качества моделей и сравнения различных алгоритмов используются ***метрики качества***.

Примеры:

- Корень из среднеквадратичной ошибки – для регрессии
- Доля правильных ответов – для классификации

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

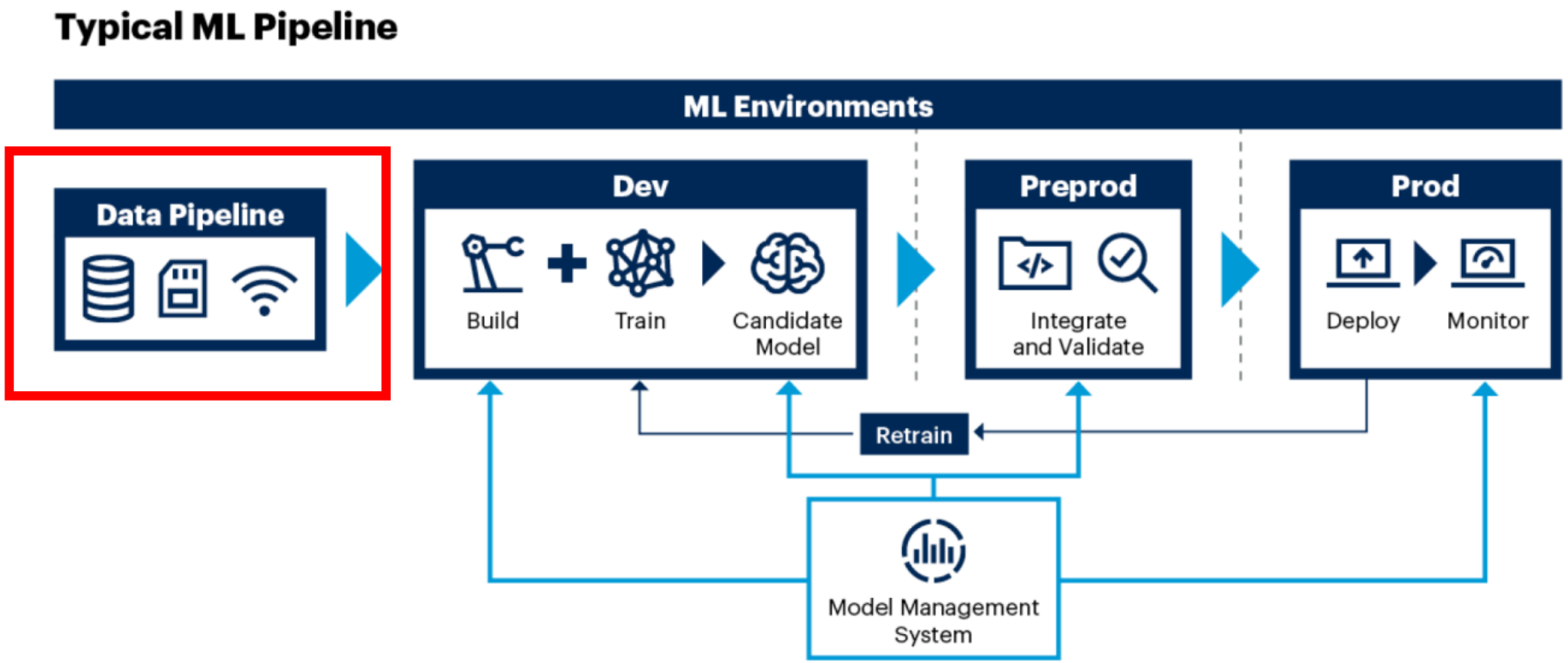
Метрики качества

Какие еще метрики качества знаете?

- *в задачах классификации*
- *в задачах регрессии*
- *в задачах кластеризации*

5. Цикл проекта по машинному обучению

Анализ данных



Source: Gartner

718951_C

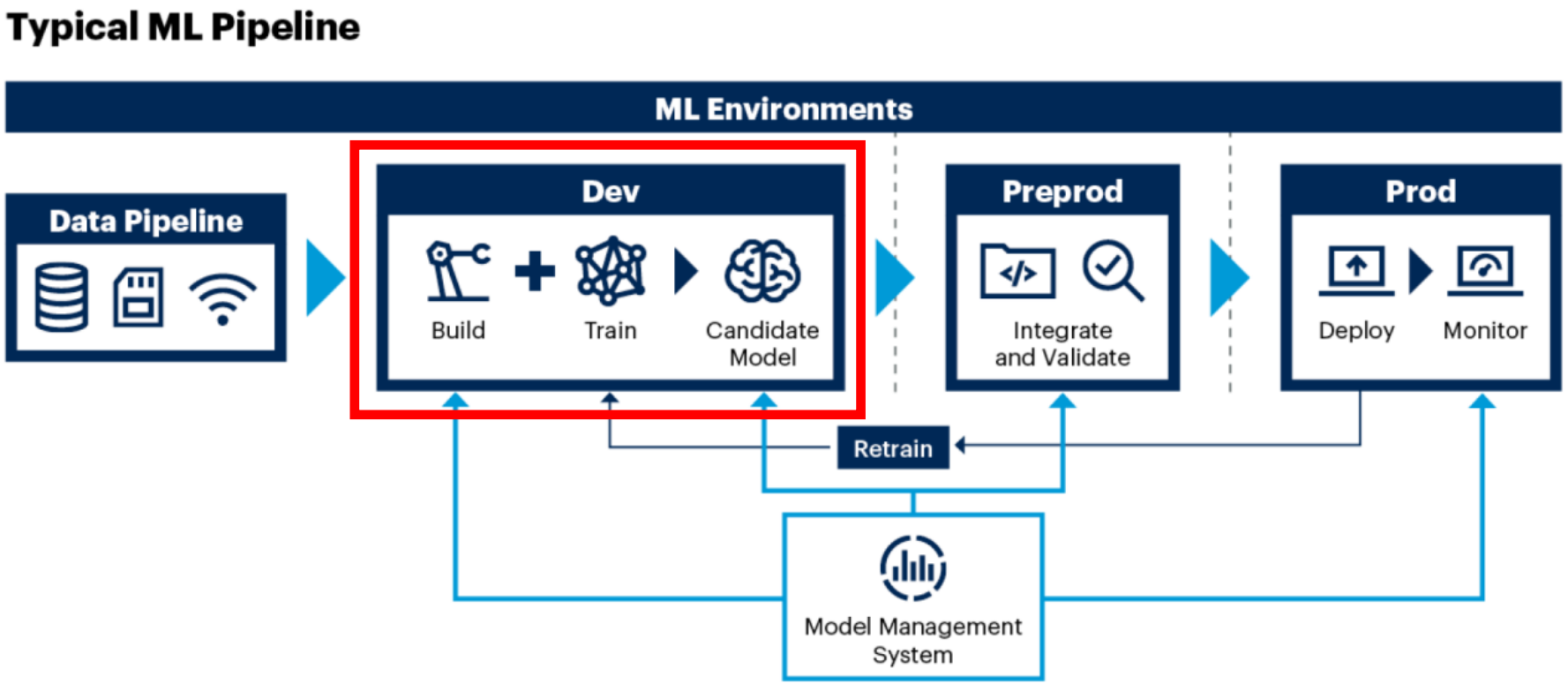
Анализ данных

1. *Сбор данных*: в каких источниках хранятся данные? Есть ли к ним доступы?
2. *Обработка данных*:

- Проверка качества данных
- Очистка данных
- Feature engineering
- Агрегация данных

3. *Загрузка данных в хранилище*
4. *Автоматизация процесса сбора, обработки и загрузки данных*

Обучение и валидация модели



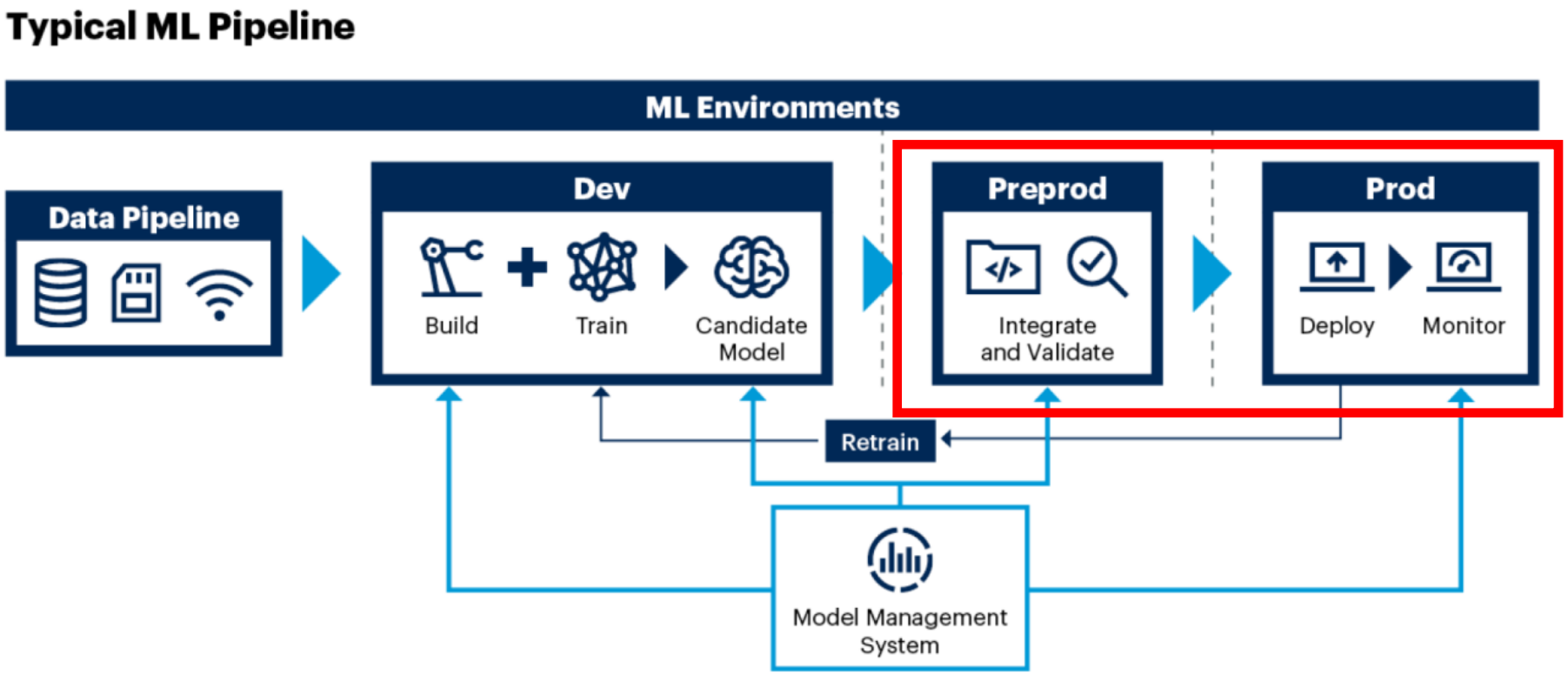
Source: Gartner

718951_C

Обучение и валидация модели

1. *Выбор модели* (линейные модели, деревья, бустинги, нейронные сети)
2. *Обучение модели*
3. *Валидация модели* (оценка качества модели на тестовых данных)
4. *Подбор гиперпараметров модели*
5. *Выбор наилучшей модели*

Внедрение модели в production



Source: Gartner

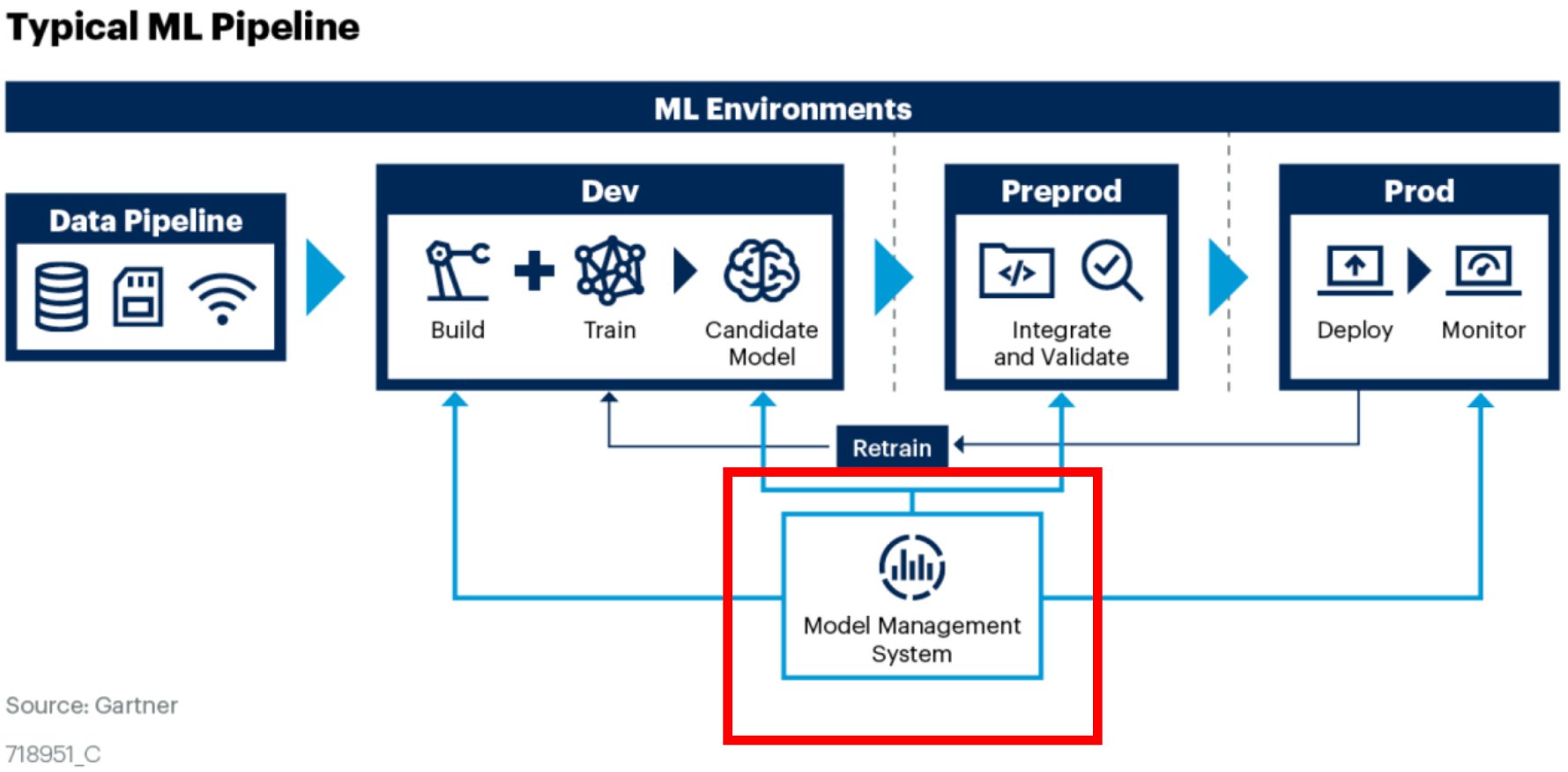
718951_C

Внедрение модели в production

Варианты внедрения:

- *Сервис (Streamlit, FastApi и другие)*
- *Telegram-бот*
- *Внедрение модели как компонента большого бизнес-процесса*

Оркестрация пайплайна и мониторинг



Оркестрация пайплайна и мониторинг

Для оркестрации (управления) пайплайна сбора, обработки данных, построения и валидации моделей используют:

- *AirFlow*
- *MLFlow*

6. Основы Natural Language Processing

1. *Задачи NLP*
2. *Простые способы векторизации текстов*
3. *Практика!*

Задачи NLP

- **Машинный перевод**: Автоматическое переведение текстов с одного языка на другой
- **Анализ тональности**: Определение эмоциональной окраски текста (положительная, отрицательная, нейтральная)
- **Извлечение информации**: Извлечение структурированной информации из текста (например, извлечение имен, дат, событий из новостных статей)
- **Классификация текста**: Определение категории или метки для текста (например, классификация сообщений на спам и не спам)
- **Генерация текста**: Создание текстовых данных, будь то генерация новостных статей, продолжение предложений и т. д.

Задачи NLP

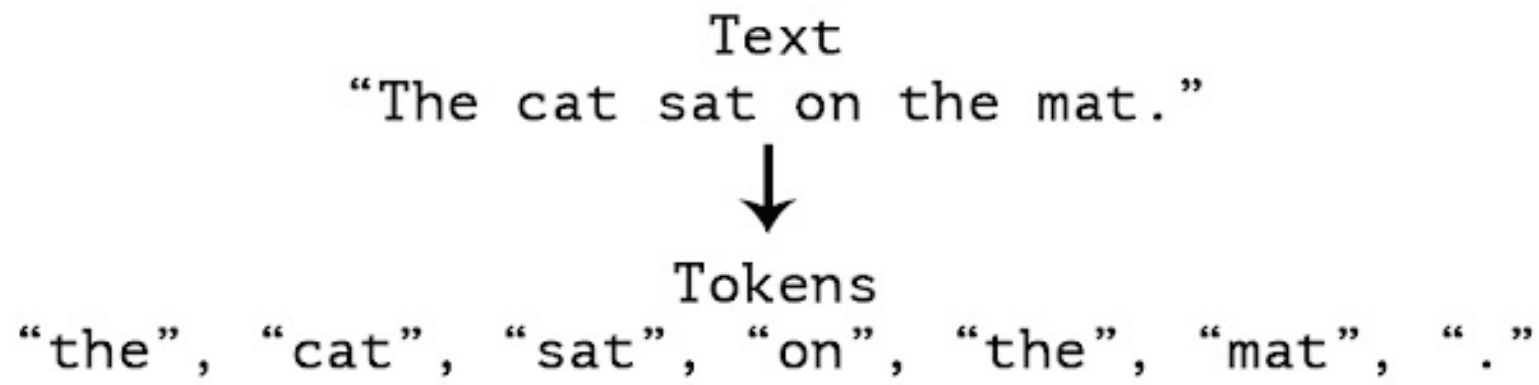
- **Автоматическое реферирование**: Создание кратких рефератов текста
- **Распознавание именованных сущностей**: Идентификация и классификация именованных сущностей в тексте (например, имена людей, организации, местоположения)
- **Синтаксический анализ**: Анализ грамматической структуры предложения
- **Генерация ответов на вопросы и речи**: Создание ответов на вопросы на основе текстовой информации
- **Моделирование диалогов**: Создание систем, способных вести разговор с пользователем

Терминология

- документ = текст
- корпус – набор документов
- токен – формальное определение “слова”; токен может не иметь смыслового значения (например, “12fdh” или “авыдшл”), но обычно отделен от остальных токенов пробелами или знаками препинания

Токенизация текста

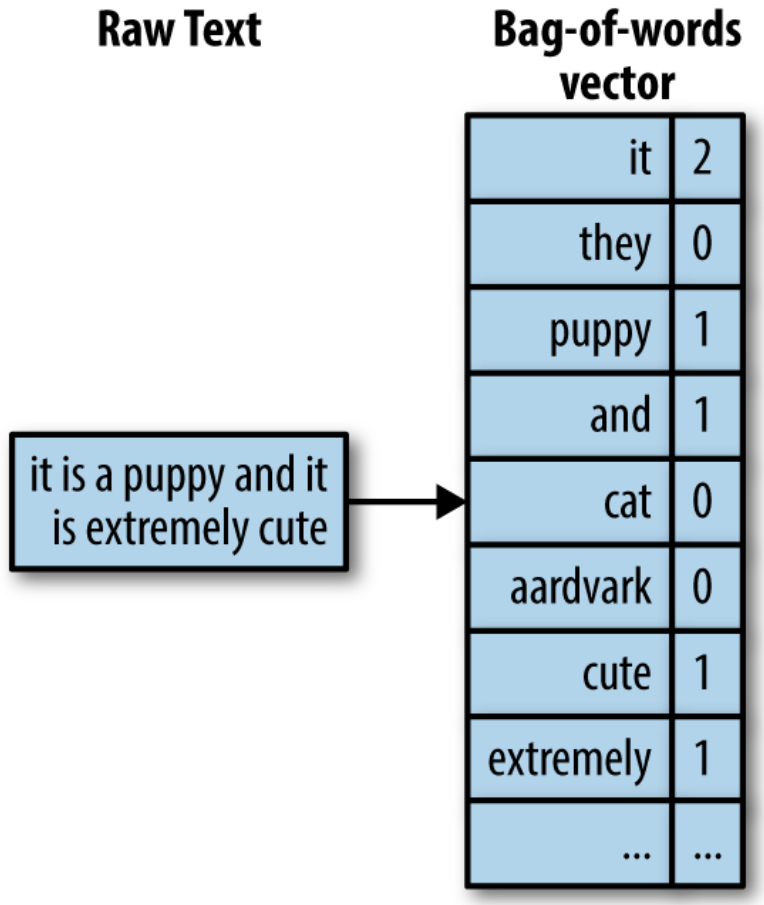
Чтобы работать с текстом, необходимо разбить его на токены. В простейшем случае токены – это слова (а также наборы букв, знаки препинания и т.д.).



Методы кодирования текстовых данных

Bag of words (мешок слов)

- По корпусу создадим словарь из всех встречающихся в нем слов (можно убрать общеупотребительные часто встречающиеся слова и очень редкие слова).
- Каждое слово закодируем вектором, в котором стоит единица на месте, соответствующем месту этого слова в словаре, все остальные компоненты вектора – 0.
- Для кодирования документа сложим коды всех его слов.



Bag of words (пример)

Пусть корпус состоит из следующих документов:

- D1 - “I am feeling very happy today”
- D2 - “I am not well today”
- D3 - “I wish I could go to play”

Кодировка этих документов будет такой:

	I	am	feeling	very	happy	today	not	well	wish	could	go	to	play
D1	1	1	1	1	1	1	0	0	0	0	0	0	0
D2	1	1	0	0	0	1	1	1	0	0	0	0	0
D3	2	0	0	0	0	0	0	0	1	1	1	1	1

Bag of words

Используя bag of words (BOW), мы теряем информацию о порядке слов в документе.

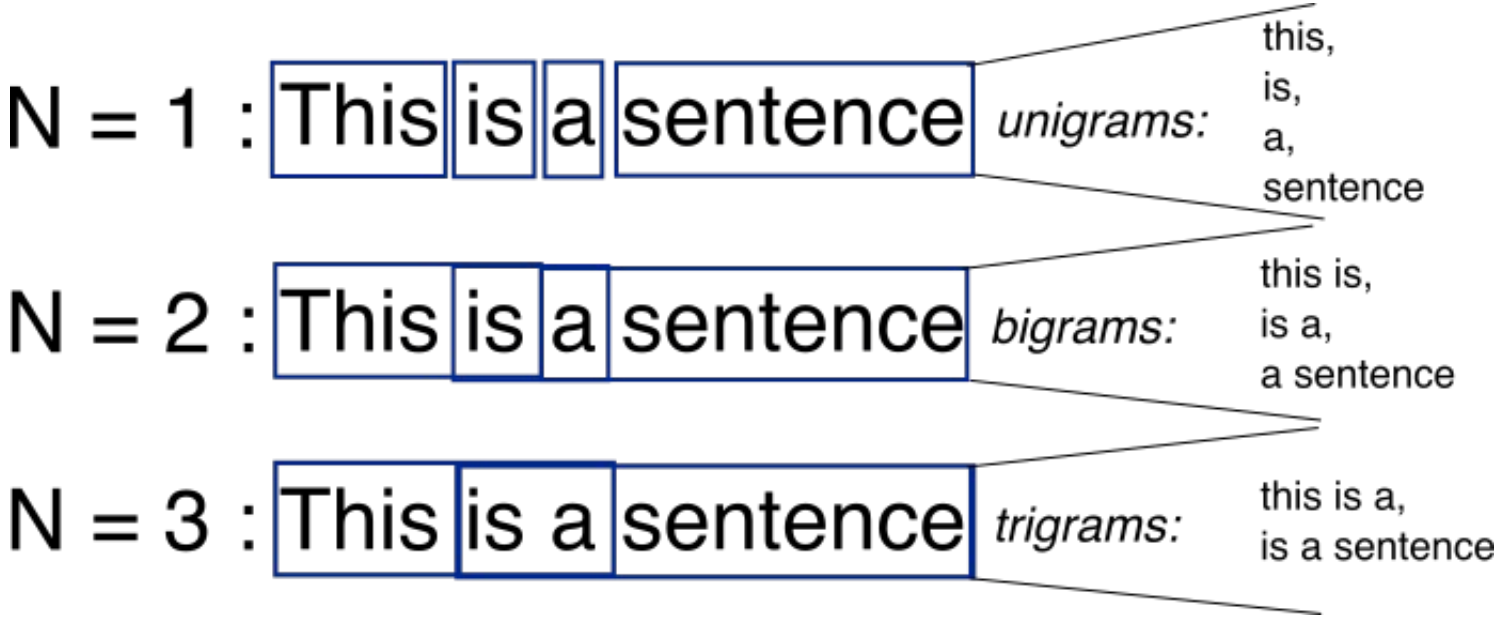
Пример: векторы документов “I have no cats” и “No, I have cats” будут идентичны.

N-gram bag of words

В качестве слов в словаре можно использовать:

- N-граммы из букв (наборы букв длины N в слове)
- N-граммы из слов (наборы фраз длины N в документе)

Такой подход поможет учесть сходственные слова и опечатки.



Tf-Idf

- Слова, которые редко встречаются в корпусе, но присутствуют в документе, могут оказаться важными для характеристики документа
- Слова, которые встречаются во всех документах, наоборот, не важны

Tf-Idf

Tf-Idf (term frequency – inverse document frequency):

- $tf(t, d)$ - *частота вхождения слова* t *в документ* d:

$$\boldsymbol{tf(t, d)} = \frac{\boldsymbol{n_t}}{\sum_{\boldsymbol{k}} \boldsymbol{n_k}} = \frac{\text{число вхождений слова } \boldsymbol{t} \text{ в документ}}{\text{общее число слов в документе}}$$

$tf(t, d)$ показывает важность слова t в документе d.

Tf-Idf

- $tf(t, d)$ - *частота вхождения слова* t *в документ* d:

$$\boldsymbol{tf(t, d)} = \frac{\boldsymbol{n_t}}{\sum_{\boldsymbol{k}} \boldsymbol{n_k}} = \frac{\text{число вхождений слова } \boldsymbol{t} \text{ в документ}}{\text{общее число слов в документе}}$$

$tf(t, d)$ *показывает важность слова* t *в документе* d.

- $idf(t, D)$ - *величина, обратная частоте, с которой слово* t *встречается в документах корпуса* D.

$$\boldsymbol{idf(t, D)} = \mathbf{log} \frac{|\boldsymbol{D}|}{|\{\boldsymbol{d_i} \in \boldsymbol{D} \mid \boldsymbol{t} \in \boldsymbol{d_i}\}|},$$

$|D|$ — число документов *в корпусе,*

$|\{d_i \in D \mid t \in d_i\}|$ - *число документов, в которых встречается слово* t

Учёт idf *уменьшает вес часто используемых в корпусе слов.*

Tf-Idf

Tf-idf слова $\boldsymbol{t}$ в документе $\boldsymbol{d}$ из корпуса $\boldsymbol{D}$:

$$\boldsymbol{tfidf(t,d,D) = tf(t,d) \times idf(t,D)},$$

Пример:

Дана коллекция D из $10000000 = 10^7$ документов, в 1000 из них встречается слово “заяц”. В данном документе d из коллекции 100 слов, и слово “заяц” встречается 3 раза.

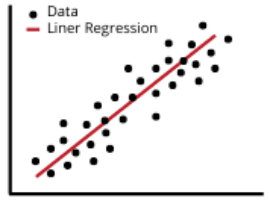
$$tf(\text{заяц}, d) = \frac{3}{100} = 0{,}03$$

$$idf(\text{заяц}, D) = \log\left(\frac{10^7}{10^3}\right) = 4$$

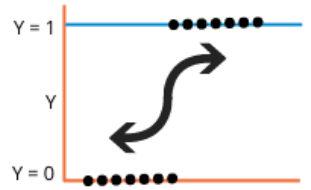
Поэтому $tfidf(\text{заяц}, d, D) = 0{,}03 \cdot 4 = 0{,}12$.

7. Классические модели: recap

Linear Regression

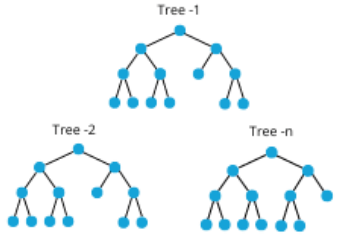


Logistic Regression

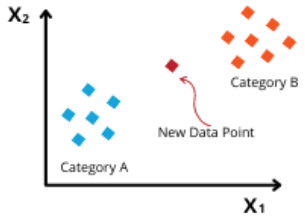


Decision Trees

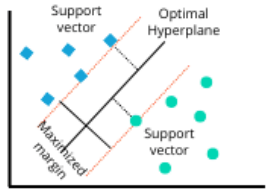
Random Forest



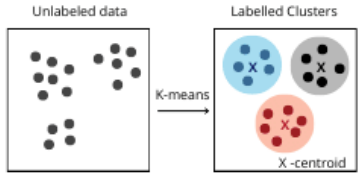
K-Nearest Neighbor



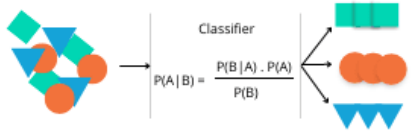
Support Vector Machine



K-Means Clustering

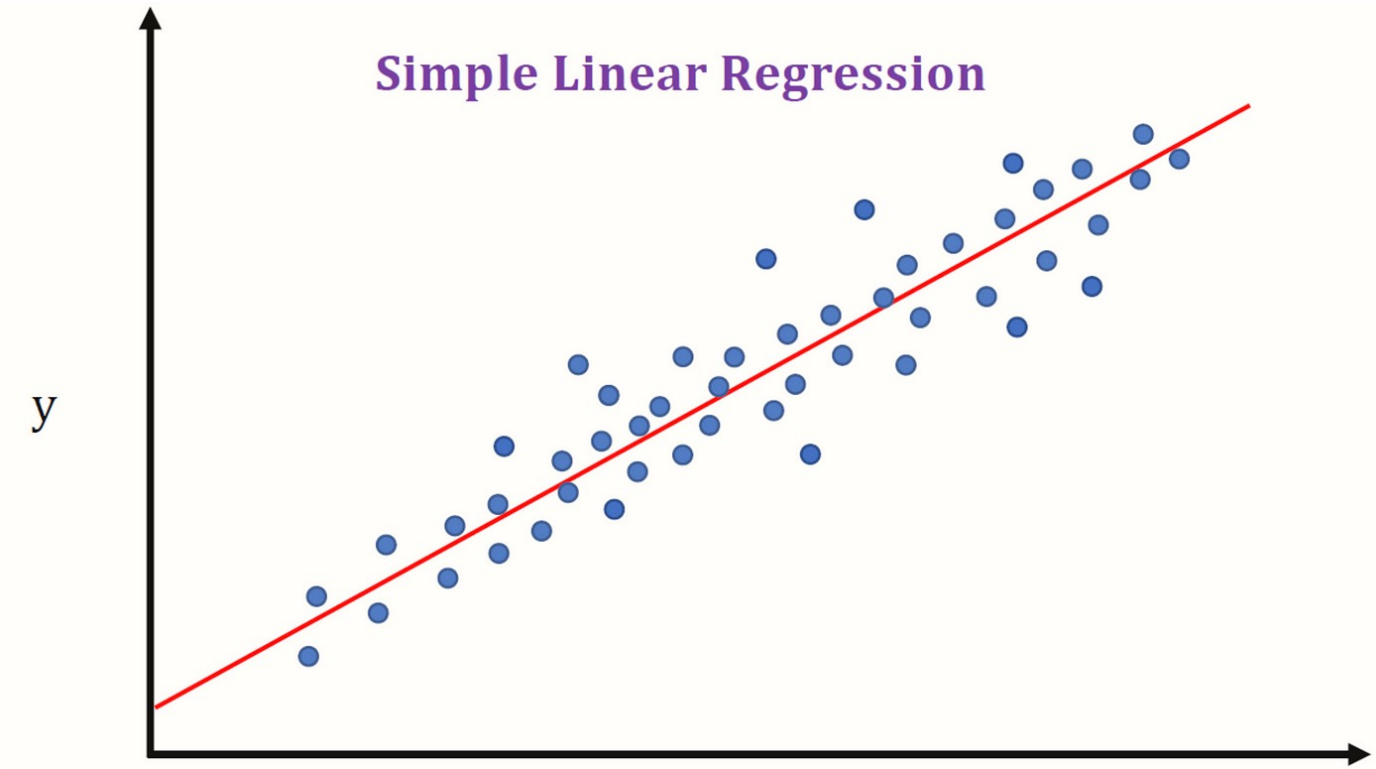


Naïve Bayes



Регрессия

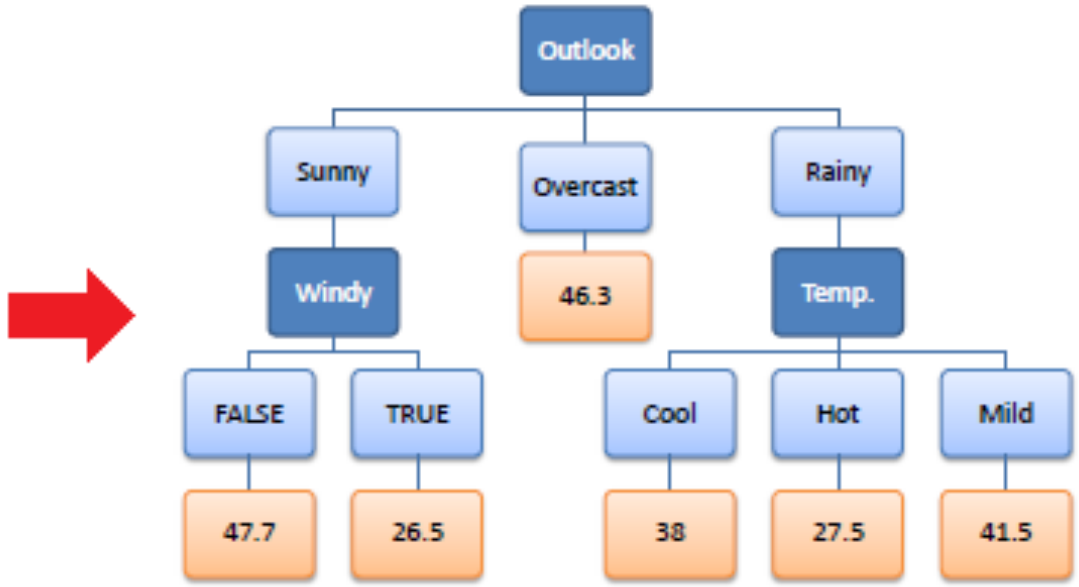
- Линейная регрессия (LinearRegression, Ridge, Lasso, ElasticNet)
- Метод k ближайших соседей
- Решающее дерево
- Случайный лес
- Градиентный бустинг



Регрессия

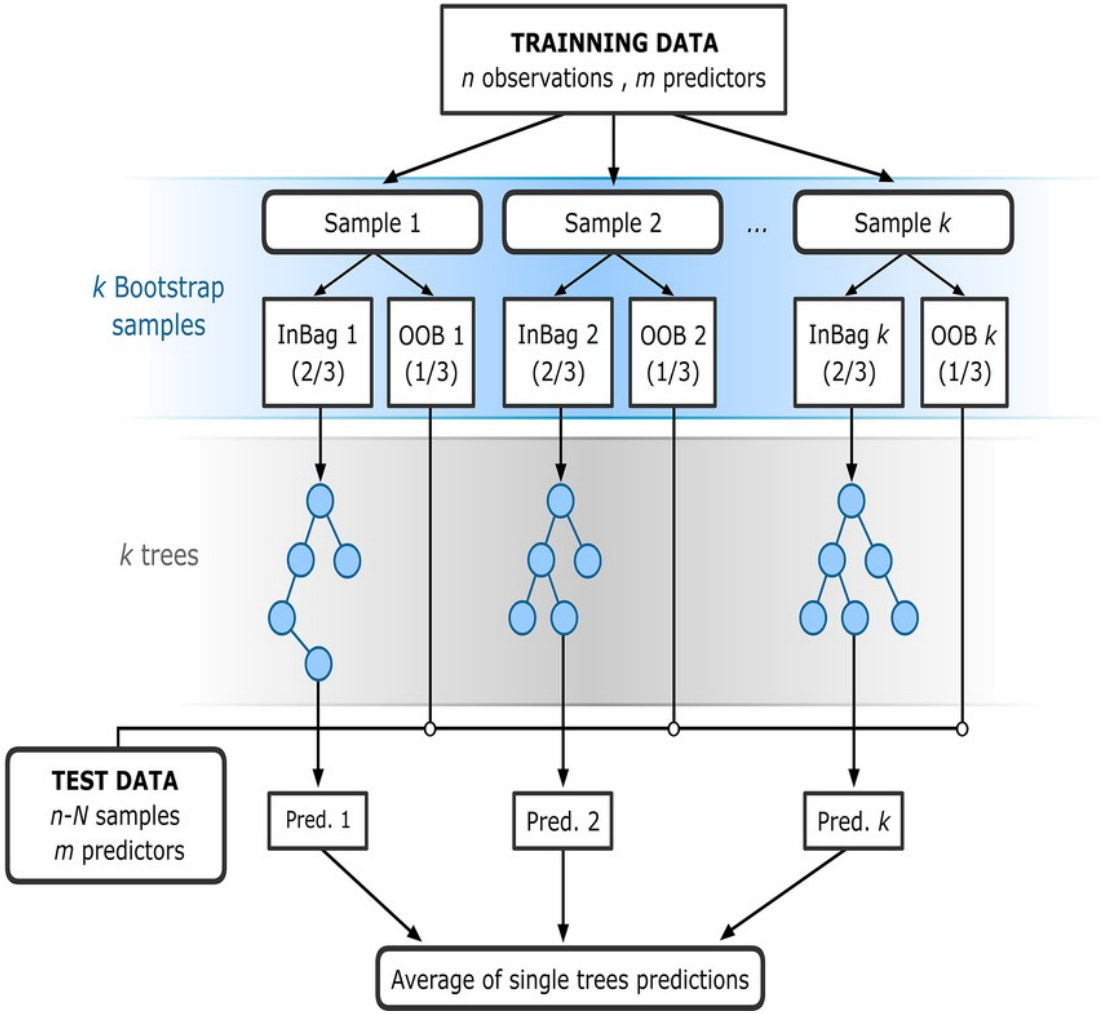
Решающее дерево

Predictors				Target
Outlook	Temp.	Humidity	Windy	Hours Played
Rainy	Hot	High	False	25
Rainy	Hot	High	True	30
Overcast	Hot	High	False	48
Sunny	Mild	High	False	45
Sunny	Cool	Normal	False	52
Sunny	Cool	Normal	True	23
Overcast	Cool	Normal	True	43
Rainy	Mild	High	False	35
Rainy	Cool	Normal	False	38
Sunny	Mild	Normal	False	46
Rainy	Mild	Normal	True	48
Overcast	Mild	High	True	52
Overcast	Hot	Normal	False	44
Sunny	Mild	High	True	30



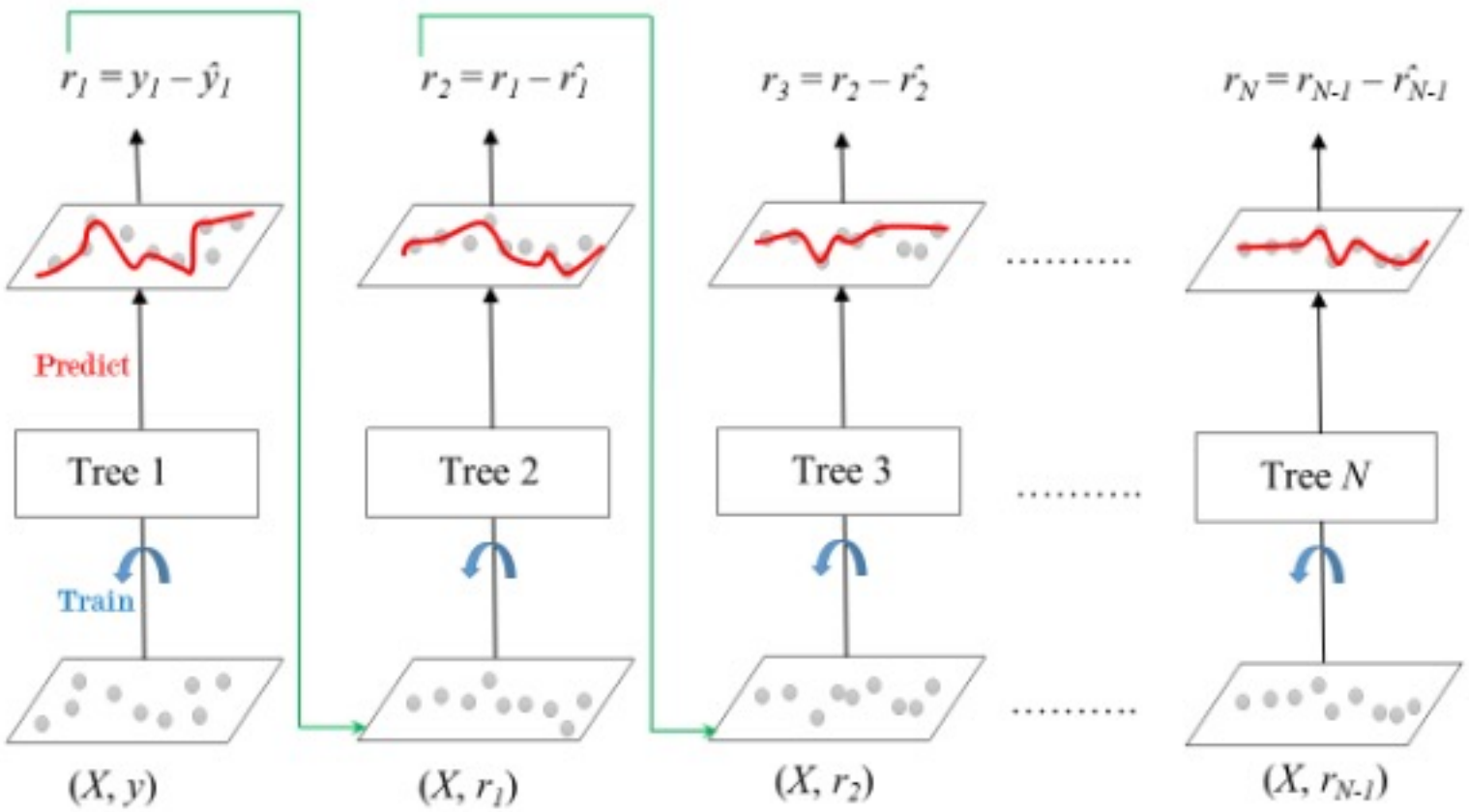
Регрессия

Случайный лес (Random Forest)



Регрессия

Градиентный бустинг (Gradient Boosting)

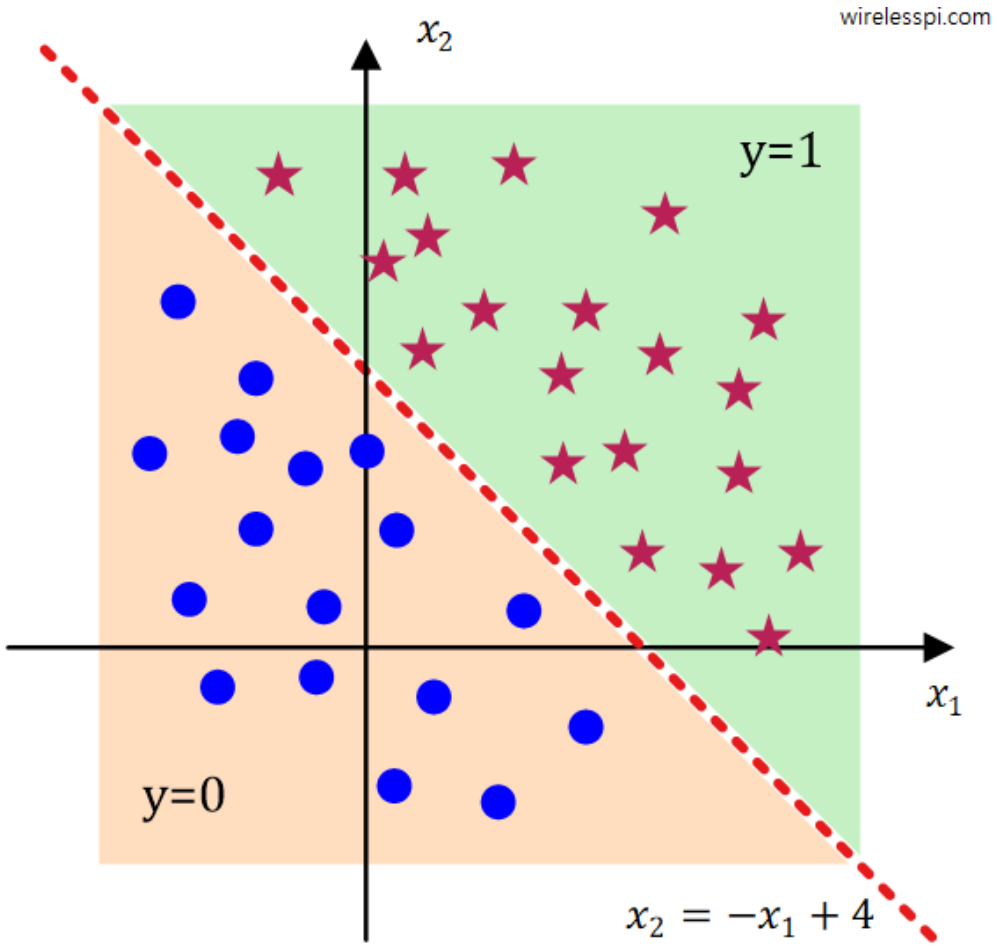


Классификация

- Логистическая регрессия
- Метод k ближайших соседей
- Метод опорных векторов
- Наивный байесовский классификатор
- Решающее дерево
- Случайный лес
- Градиентный бустинг

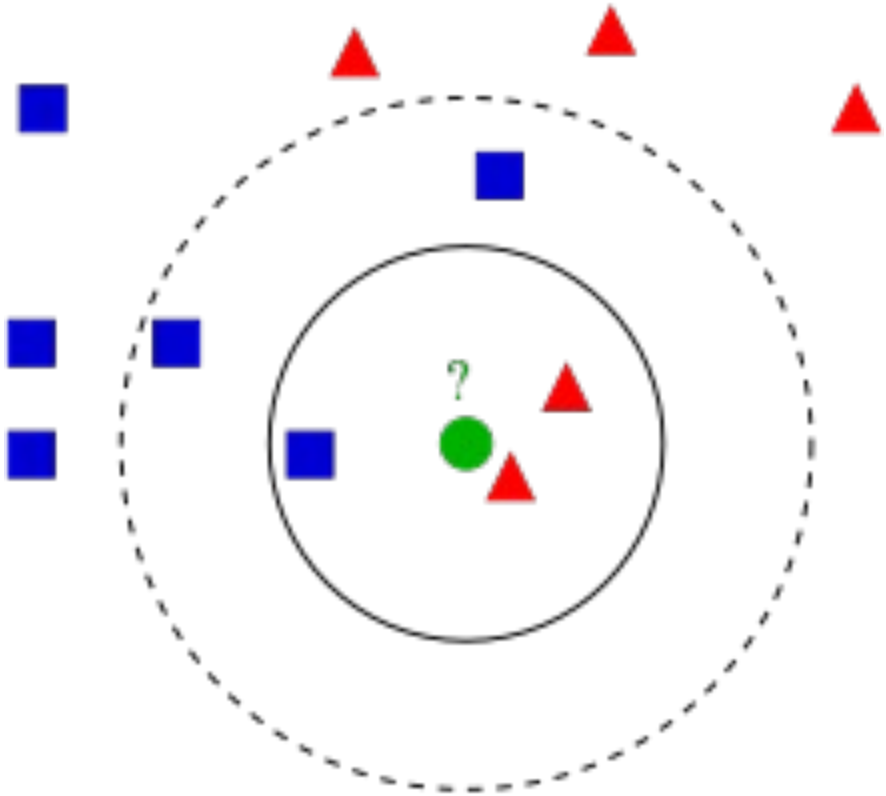
Классификация (линейная)

Логистическая регрессия (Logistic Regression), Метод опорных векторов (SVM)



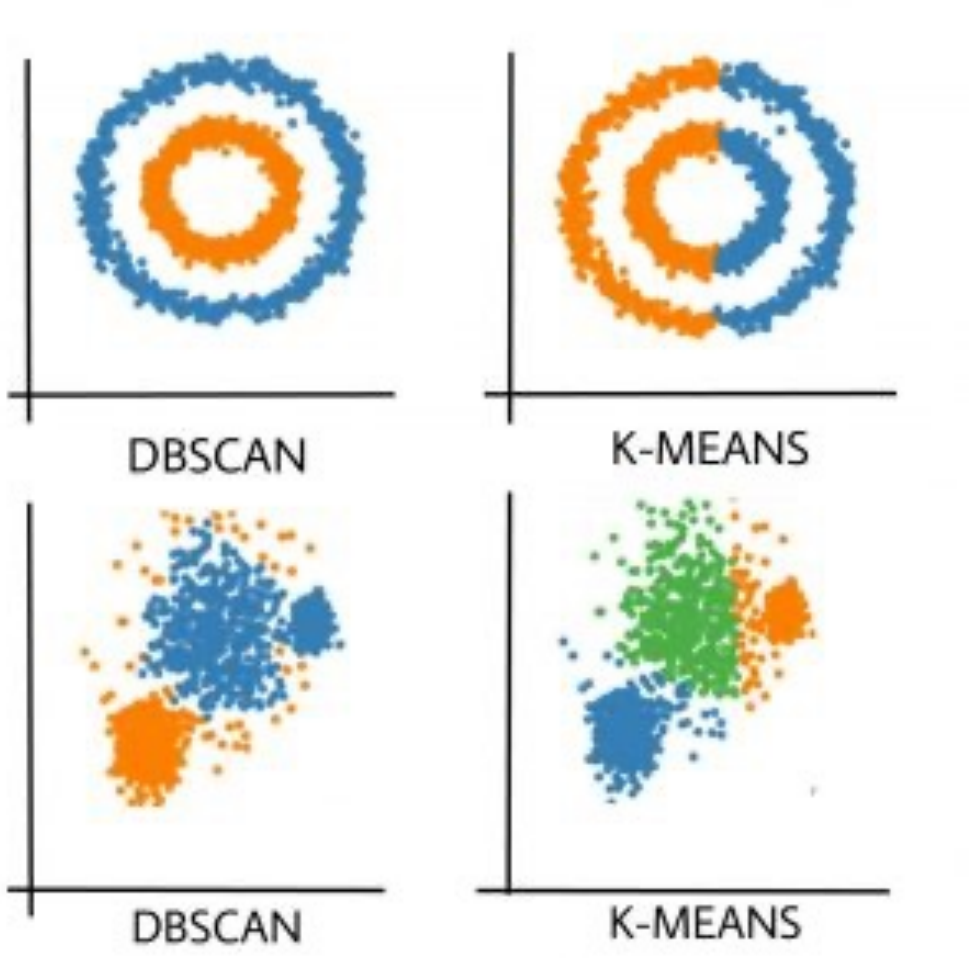
Классификация

Метод k ближайших соседей



Кластеризация

- Метод k средних (KMeans)
- DBSCAN, HDBSCAN
- Иерархическая кластеризация



Источники для изучения

- Лекции и семинары с курса Евгения Соколова
- Онлайн-курс “Практический Machine Learning” (промокод STUDCAMP)

Полезные материалы

- Инструкция для верификации аккаунта на Kaggle