Машинное обучение

2025/2026 учебный год

Доцент кафедры МО&МО, Махно В.В.

Запись на курс

Курс: Машинное обучение (ПО)

кодовое слово **16032026**

Лекция 2. Задача классификации —выбор категорий

Классификация — это задача, в которой модель **предсказывает, к какому классу относится объект.**

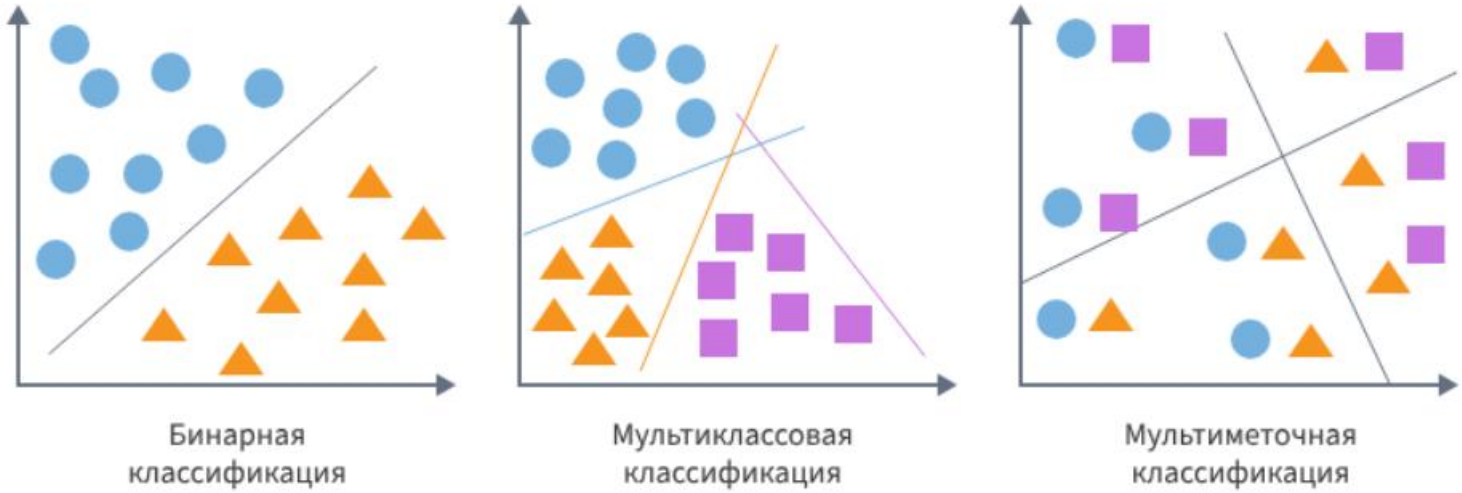
- Не «Сколько?», а **«Что именно?»**

Примеры задач:

- Спам или не спам?
- Мужчина или женщина?
- Солнечно, облачно, дождь?
- ❤ Купит клиент товар или нет?

Типы классификации

Тип задачи	Пример	Классы
Бинарная	Спам или не спам	0 / 1
Мультиклассовая	Вид животного	Кошка / Собака / Лиса
Многоклассовая + мульти-ответ	Темы статьи	[Технологии, Наука]



Классификация —выбор категорий

Цель модели

- Обучить алгоритм, который по **набору признаков** будет **относить объект к правильному классу**.

Допустим, у нас есть данные о покупателях:
Мы хотим научить модель **предсказывать**, купит ли следующий клиент товар, основываясь на возрасте и доходе.

Возраст	Доход	Купил (target)
22	30	Нет (0)
45	90	Да (1)
35	70	Да (1)

Алгоритмы классификации

Модель	Особенности
LogisticRegression	Простая, быстрая, интерпретируемая
KNN	Смотрит на "похожих" соседей
DecisionTree	Принимает решения по признакам, как вопросы
RandomForest	Несколько деревьев голосуют
SVM	Строит границу между классами
NaiveBayes	Часто используется в текстах (спам-фильтры)

Применение классификации

Область	Пример
Маркетинг	Классификация клиентов: уйдёт / останется
Финансы	Одобрит ли банк кредит
Медицина	Заболел / не заболел
HR	Подходит ли кандидат
E-commerce	Рекомендации и персонализация

Logistic Regression (Логистическая регрессия)

Модель оценивает вероятность принадлежности объекта к определённому классу. Она применяет сигмоиду (логистическую функцию) к линейной комбинации признаков, чтобы «сжать» результат в диапазон от 0 до 1.

Если $P(y=1|x) > 0.5$ — объект относится к классу 1, иначе — к классу 0.

Плюсы

- Простая и быстрая.
- Хорошо интерпретируема: мы можем посмотреть на веса признаков и понять, что влияет на результат.
- Не требует много данных.

Минусы

- Предполагает линейную связь между признаками и логарифмом шансов.

K-Nearest Neighbors (KNN / Метод k-ближайших соседей)

Модель, не обучает, а просто запоминает данные.

Когда приходит новый объект, алгоритм ищет k ближайших к нему объектов из обучающей выборки (по расстоянию — евклидову, манхэттенскому и т.д.) и «голосованием» определяет класс.

Плюсы

- Очень прост для понимания.
- Не делает предположений о распределении данных.
- Хорошо работает, если данных много, а границы классов сложные.

Минусы

- Очень медленный на больших данных (нужно считать расстояние до всех точек).
- Чувствителен к масштабу признаков (обязательно нужно нормировать данные).

Decision Tree (Дерево решений)

Имитирует процесс принятия решений человеком: вопросы «да/нет» и ветвление.

- Алгоритм рекурсивно разбивает данные на подмножества по определённому признаку так, чтобы в каждом узле росла «чистота» классов (обычно через критерии *Gini impurity* или *энтропию*).

Плюсы

- Прозрачная и интерпретируемая (можно визуализировать).
- Не требует масштабирования признаков.
- Может работать как с числами, так и с категориями.

Минусы

- Склонен к переобучению (если не ограничивать глубину).

Random Forest (Случайный лес)

Ансамбль деревьев: «Один ум хорошо, а тысяча — лучше».

- Строится множество деревьев, каждое на своей случайной подвыборке данных и случайном наборе признаков (метод *бэггинга*). Итоговый класс определяется голосованием большинства деревьев.

Плюсы

- Очень высокая точность.
- Устойчив к переобучению (за счёт усреднения).
- Может оценивать важность признаков.
- Работает с пропусками и выбросами.

Минусы

- Медленнее и требует больше памяти, чем одно дерево.
- Менее интерпретируем (сотни деревьев не объяснить человеку).

Support Vector Machine (SVM / Метод опорных векторов)

Ищет не просто разделяющую линию, а самую «жирную» разделяющую полосу.

- Алгоритм строит гиперплоскость, которая максимально далеко отстоит от ближайших точек разных классов (эти точки и называются *опорными векторами*).
- С помощью *ядер* (kernel trick) SVM может работать даже с нелинейными данными, проецируя их в пространство более высокой размерности.

Плюсы

- Отлично работает в многомерных пространствах.
- Эффективен, когда число признаков больше числа объектов.
- Гибкость за счёт выбора ядра.

Минусы

- Чувствителен к масштабу данных.
- Плохо интерпретируем.
- Требует подбора параметров (C, gamma).

Naive Bayes (Наивный Байес)

Основан на теории вероятностей и «наивном» предположении, что все признаки независимы.

- Использует теорему Байеса для вычисления вероятности принадлежности объекта к классу при условии его признаков. «Наивность» в том, что мы считаем все признаки независимыми (хотя в жизни это редко так).

Плюсы

- Очень быстрый и простой.
- Хорошо работает с текстами.
- Требует мало данных для обучения.
- Не чувствителен к нерелевантным признакам.

Минусы

- Наивное предположение о независимости — его главный недостаток.
- Плохо работает, если признаки сильно коррелируют.

Сравнение моделей классификации

Модель	Тип	Интерпретируемость	Скорость	Точность	Когда использовать
Logistic Regression	Линейная	Высокая	Высокая	Средняя	Базовый уровень, интерпретация
KNN	Ленивая	Средняя	Низкая	Средняя	Мало данных, сложные границы
Decision Tree	Древесная	**Очень высокая**	Высокая	Средняя	Нужно объяснить решения
Random Forest	Ансамбль	Низкая	Средняя	**Высокая**	Максимальная точность
SVM	Ядерная	Низкая	Средняя	Высокая	Много признаков, текст
Naive Bayes	Вероятностная	Высокая	**Очень высокая**	Средняя	Тексты, быстрые решения

Табличные данные

Заработная плата	Возраст	Должность	Уровень образования	Город проживания	Стаж работы (годы)	Вернет ли клиент кредит
100000	26	Риэлтор	Высшее	Санкт-Петербург	5	Да
50000	20	Продавец-консультант	Высшее	Москва	1	Нет
35000	39	Автомеханик	Среднее специальное	Воронеж	8	Нет
25000	23	Программист	Высшее	Самара	2	Да
75000	41	Юрист	Среднее	Москва	14	Да

Задача классификации

Пример с вакансиями.

Нужно построить алгоритм, который позволит системе определить, есть ли в резюме кандидата необходимые параметры. Если есть – отправить в папку «Собеседование», если нет – в папку «Отказать».

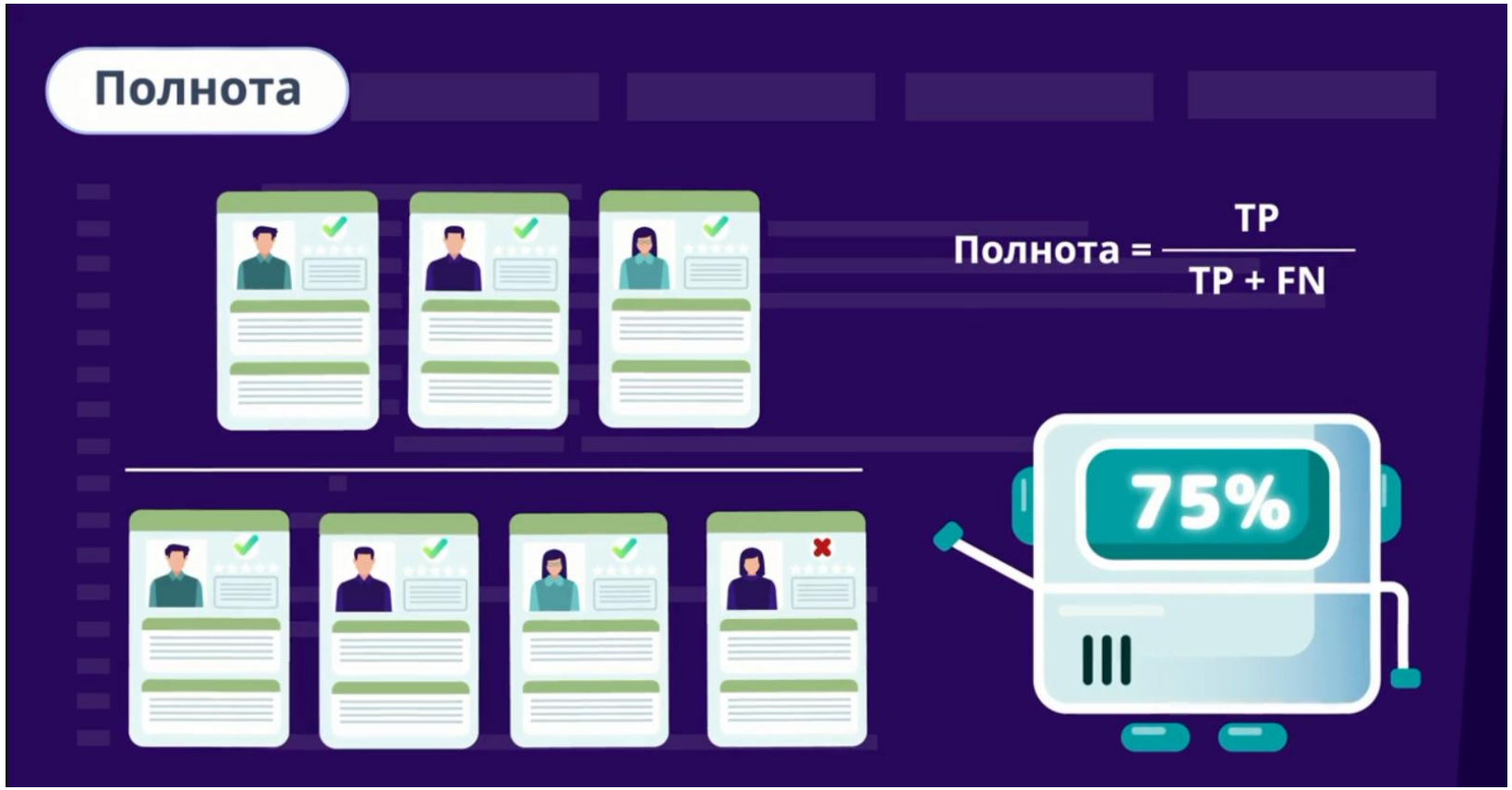
- TP — true positive, алгоритм верно пометил резюме как подходящее
- TN — true negative, алгоритм верно отнес резюме к неподходящим
- FP — false positive, алгоритм ошибочно считает подходящим резюме, в котором нет нужных качеств
- FN — false negative, алгоритм ошибочно отбраковал подходящее резюме

Метрики

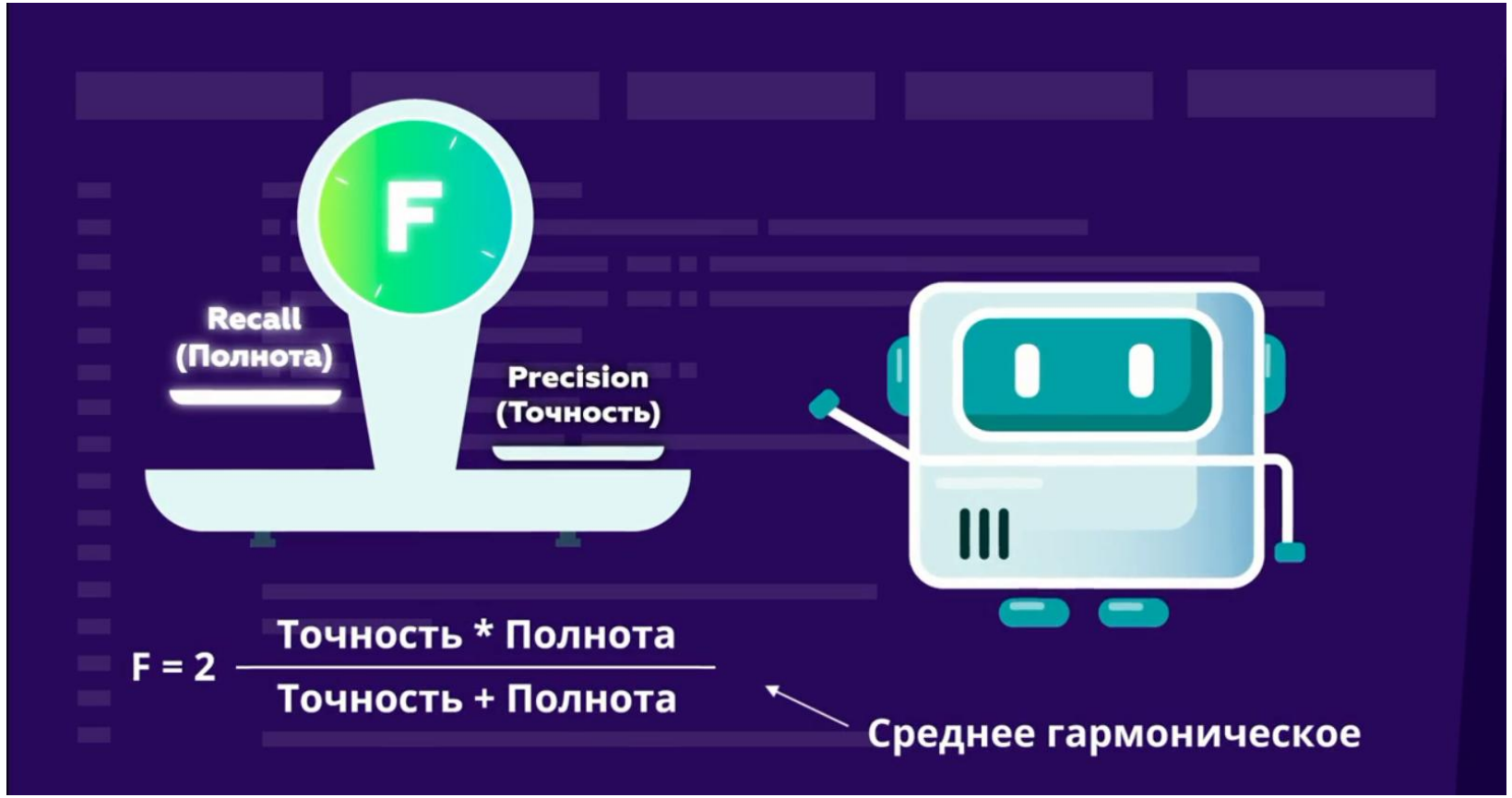
- Самая простая метрика – доля правильных предсказаний: сколько раз прогноз машины и разметка программиста совпали между собой.
- Другая метрика – точность. Она показывает отношение количества верно угаданных подходящих резюме к количеству тех, кого машина вообще отнесла к группе «собеседование».
- Кроме точности есть еще метрика полноты. Она показывает отношение количества верно угаданных подходящих резюме, к другому значению: количеству кандидатов, которых следовало пригласить по мнению программиста.

Метрика точность

Метрика полнота



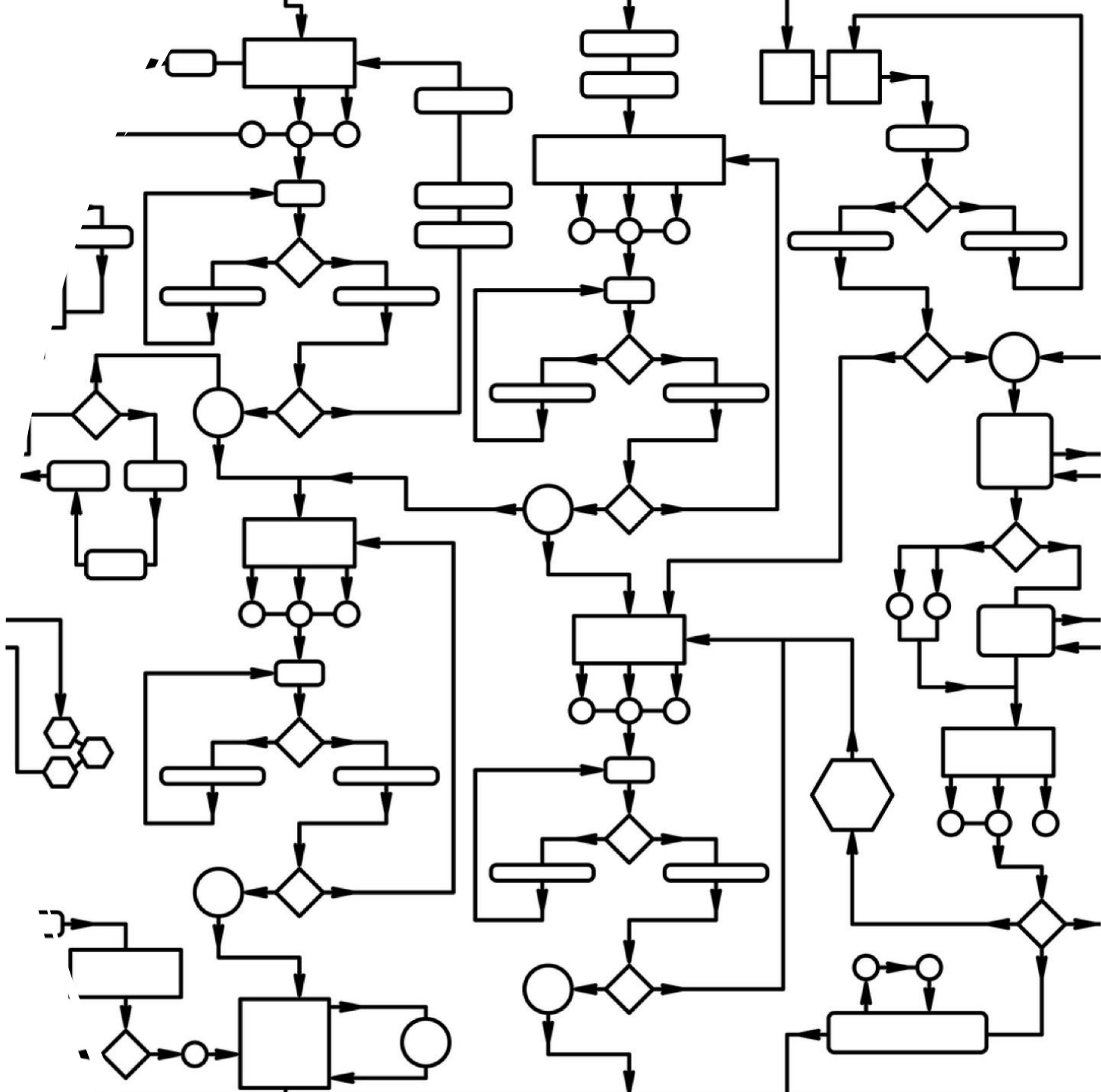
F-метрика


F
Recall
(Полнота)
Precision
(Точность)
F = 2
Точность * Полнота
Точность + Полнота
Среднее гармоническое

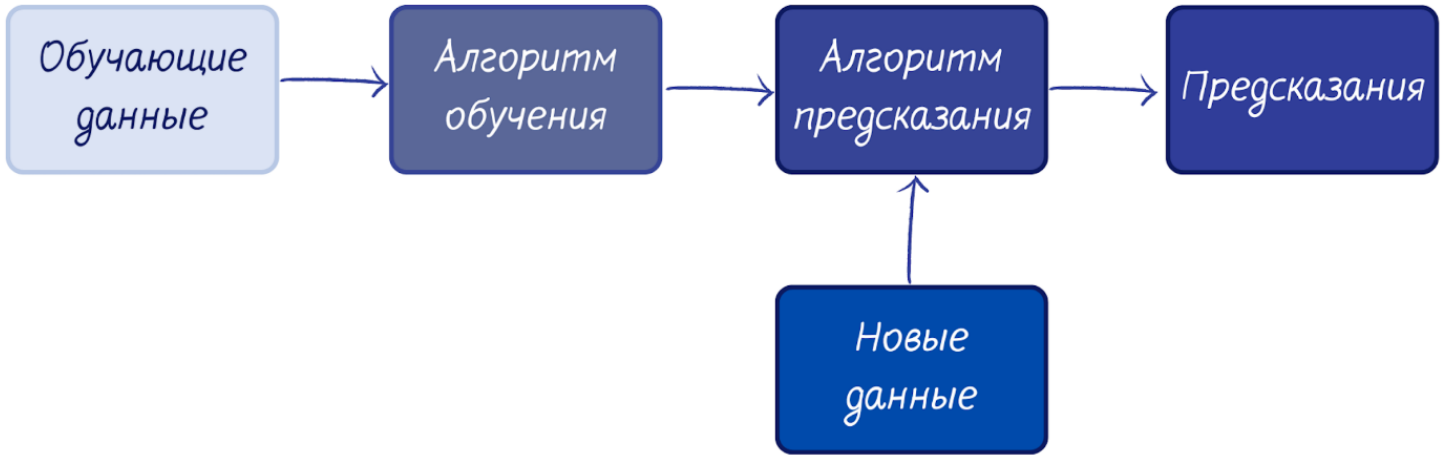
Создание алгоритма классификации

В этой задаче каждому объекту (строке в таблице данных) соответствует класс — значение из заданного набора классов.

Задача классификации состоит в том, чтобы разработать алгоритм, который по признакам объекта будет предсказывать класс



Машинное обучение



Бесплатный курс от Сбера по генеративному искусству

https://courses.sberuniversity.ru/generative_art?utm_source=tg&utm_medium=organic&utm_campaign=courses&utm_content=gen_i&utm_term=01_09_2023