Data science

Лекция 3. Методы классификации

2025/2026 учебный год

Доцент кафедры МО&МО, Махно В.В.

Кодовое слово для способа МАГИСТРЫ 2026 = **12022026**

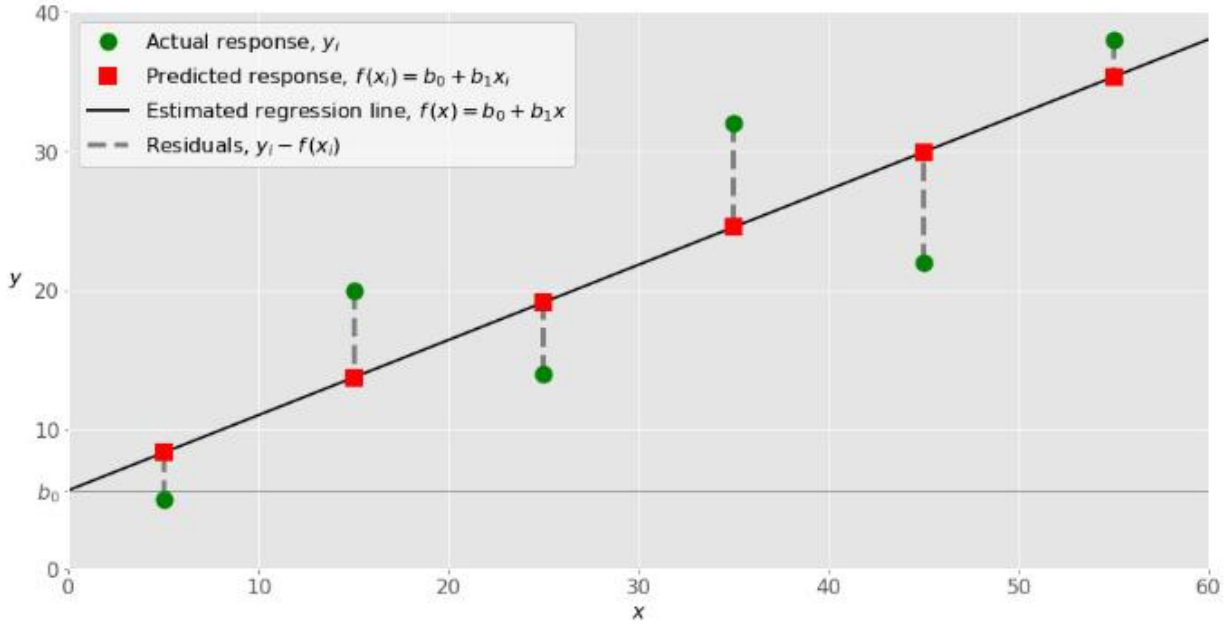
Data Science и инструменты Анализа Данных

ИНСТИТУТ
МАТЕМАТИКИ
МЕХАНИКИ
КОМПЬЮТЕРНЫХ
НАУК
имени И.И. Воровича

Линейная регрессия

Линейная регрессия — это модель, которая **ищет прямую зависимость между признаками и целевой переменной.**

То есть: "Как изменяется y, если изменить x?«



Как зависит **цена квартиры** от её **площади**

Как влияет **опыт работы** на **зарплату**

Формула Линейной Регресии

Классическая линейная модель:

$y=w \cdot x+b$

где:

- x — признак (например, площадь)
- y — результат (например, цена)
- w — **вес (коэффициент)** признака
- b — **сдвиг (intercept, свободный член)**

Площадь (м²)	Цена (₽ тыс)
30	2500
50	4000
70	5200

Модель "учится" подбирать w и b, чтобы как можно точнее предсказывать y.

Модель подбирает такие коэффициенты w и b, чтобы **суммарная ошибка между предсказанными и реальными значениями была минимальной**.

Ошибка модели (MSE):

$$\text{MSE} = \frac{1}{n} \sum (y_{\text{реальное}} - y_{\text{предсказанное}})^2$$

Пример на Python

```
import pandas as pd from sklearn.linear_model
import LinearRegression
X = data[['area']] # признаки
y = data['price'] # целевая переменная
model = LinearRegression()
model.fit(X, y)
```

Минусы Линейной Регресии

- Модель **предполагает линейную зависимость** — если зависимость не линейная, результат может быть неточным.
- **Чувствительна к выбросам** — экстремальные значения могут сильно сместить прямую.
- Не умеет учитывать **сложные связи между признаками**.

Как измерять качество регрессии?

🔹 **MAE (Mean Absolute Error) — средняя абсолютная ошибка**

$$MAE = \frac{1}{n} \sum |y_{\text{истинное}} - y_{\text{предсказанное}}|$$

🔸 **Интерпретация:**

На сколько в среднем ошибается модель (в тех же единицах, что и ответ).

🔧 Пример:

Предсказываем зарплату → MAE = 5000 → в среднем ошибка ±5 тыс

🔹 **MSE (Mean Squared Error) — средняя квадратичная ошибка**

$$MSE = \frac{1}{n} \sum (y_{\text{истинное}} - y_{\text{предсказанное}})^2$$

🔸 **Чувствительна к выбросам** (сильно штрафует большие ошибки)

🔹 **RMSE — корень из MSE**

$$RMSE = \sqrt{MSE}$$

🔸 То же, что MSE, но в тех же единицах, что и ответ

🔹 **R² (коэффициент детерминации)**

$$R^2 = 1 - \frac{MSE}{\text{вариация данных}}$$

🔸 Показывает, **какая доля дисперсии объясняется моделью**

📈 Значения от 0 до 1 (чем ближе к 1 — тем лучше)

Метрика	Показывает	Комментарий
MAE	Среднюю ошибку	Просто и понятно
MSE	Квадратичную ошибку	Штрафует сильнее
RMSE	Корень из MSE	Более интерпретируемо
R^2	"Процент объяснённости"	Часто используется в отчётах

Классификация —выбор категорий

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

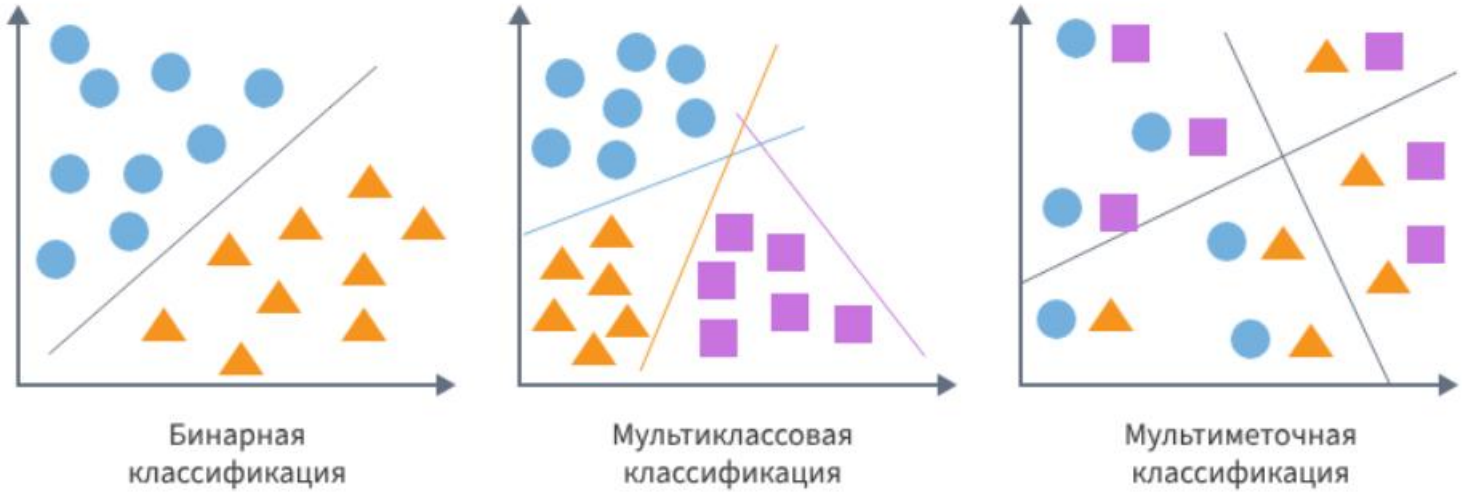
- Не «Сколько?», а **«Что именно?»**

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**.

Пример

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.
- Может плохо работать, если признаки сильно коррелируют друг с другом.

Применяется в базовых задачах классификации: отбор кандидатов, определение спама, кредитный скоринг.

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).
- Плохо работает при большом количестве признаков (проклятие размерности).

Применение

- Системы рекомендаций, распознавание образов, поиск похожих объектов.

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).
- Неустойчив — малое изменение данных может сильно изменить дерево.

Применение

- В задачах, где важна интерпретируемость: медицина, банковское дело, скоринг.

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).

Применение

- Почти везде, где нужна высокая точность: от финансов до биоинформатики.

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).
- Долго обучается на больших данных.

Применение

- Классификация текстов, распознавание лиц, биоинформатика.

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.

Применение

- Классификация текстов (спам-фильтры, тональность отзывов), рекомендательные системы.

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

Задание на практику 1

Подключение библиотек:

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.datasets import make_classification, make_blobs, load_iris
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
import seaborn as sns

# Для визуализации границ решений
def plot_decision_boundary(X, y, model, title):
    x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
    y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
    xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.02),
                         np.arange(y_min, y_max, 0.02))

    Z = model.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)

    plt.contourf(xx, yy, Z, alpha=0.3, cmap='coolwarm')
    plt.scatter(X[:, 0], X[:, 1], c=y, edgecolors='k', cmap='coolwarm')
    plt.title(title)
    plt.xlabel('Признак 1')
    plt.ylabel('Признак 2')
    plt.show()
```

Логистическая регрессия (Logistic Regression)

1.Измените значение C (параметр регуляризации) и посмотрите, как меняется граница.

2.Попробуйте обучить модель на данных с 5 признаками и выведите веса каждого признака. Какой признак влияет сильнее всего?

```
from sklearn.linear_model import LogisticRegression

# Генерируем данные
X, y = make_classification(n_samples=200, n_features=2, n_redundant=0,
                           n_clusters_per_class=1, random_state=42)

# Разделяем на train/test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Создаем и обучаем модель
model = LogisticRegression()
model.fit(X_train, y_train)

# Предсказание
y_pred = model.predict(X_test)

# Оценка
print(f"Accuracy: {accuracy_score(y_test, y_pred):.3f}")
print(f"Коэффициенты модели: {model.coef_[0]}")
print(f"Свободный член: {model.intercept_[0]}")

# Визуализация
plot_decision_boundary(X_test, y_test, model, "Логистическая регрессия")
```

Кодовое слово для способа МАГИСТРЫ 2026 = **12022026**