ЗАДАЧА КЛАССИФИКАЦИИ

Деревья решений,
случайный лес,
градиентный бустинг

Бинарный линейный классификатор

Дана обучающая выборка

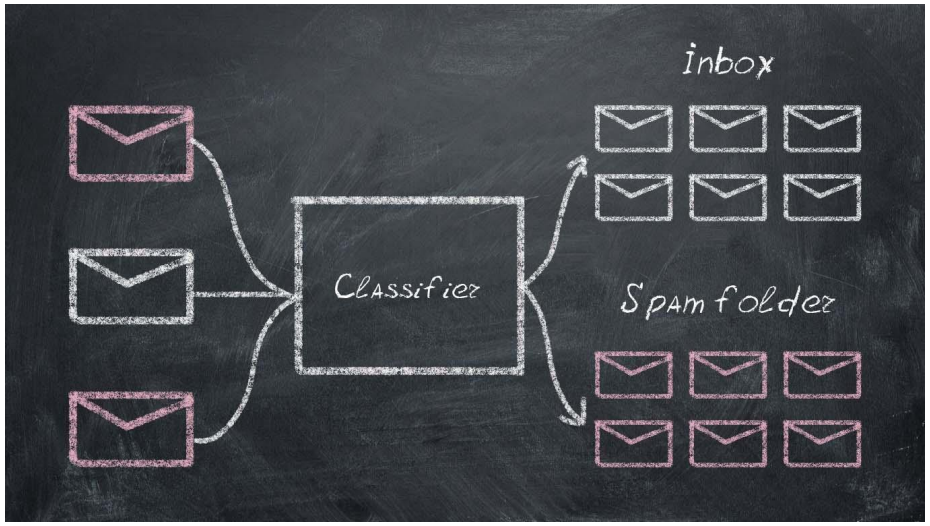
$$X_N=\{(\boldsymbol{x}_1,y_1),\ldots,(\boldsymbol{x}_N,y_N)\},\quad \boldsymbol{x}_i\in \boldsymbol{R}^P,\ y_i\in\{-1,+1\}$$

Цель: каждый новый входной вектор x отнести к одному их двух классов – положительному «+1» или отрицательному «+1»

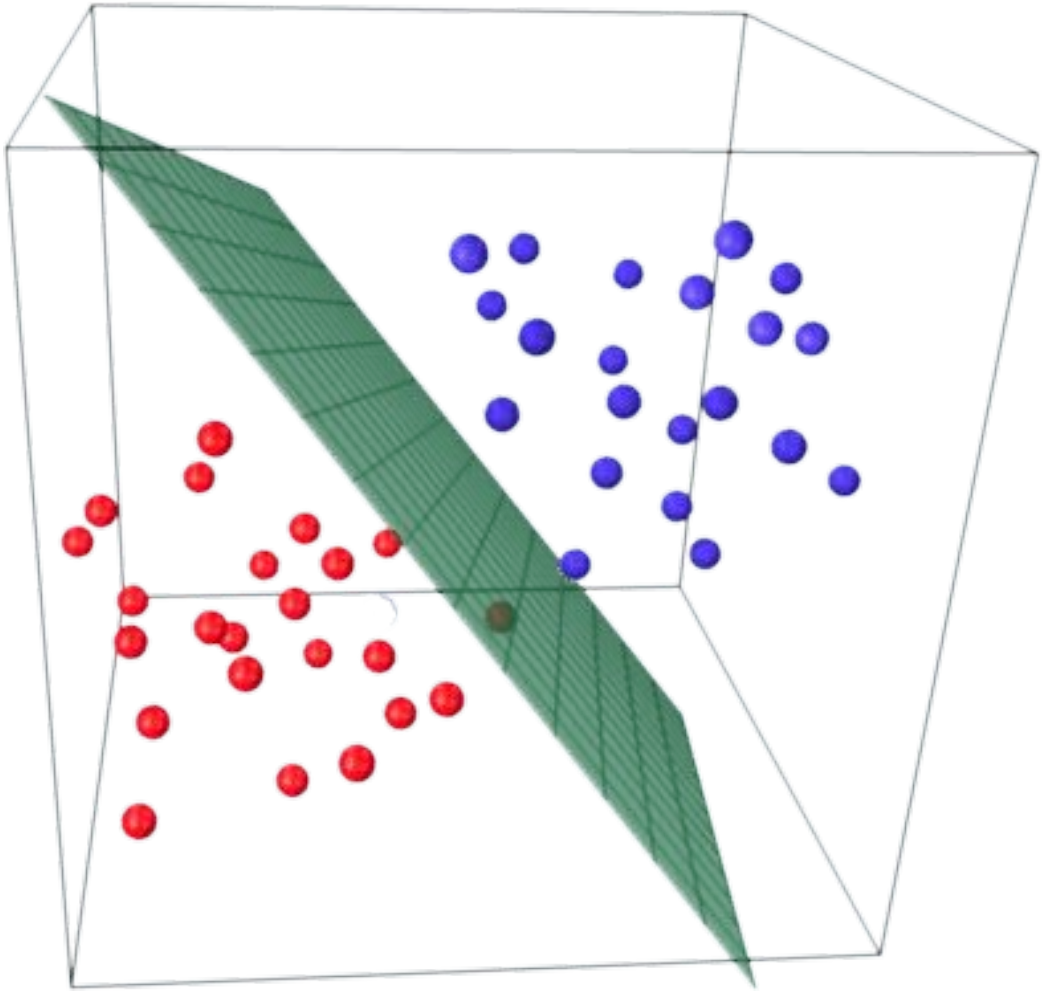
Примеры задач Data mining

- **классификация** – отнесение объекта к одной из категорий (классов) на основании его признаков
- **регрессия** – прогнозирование значения непрерывного количественного признака объекта на основании прочих его признаков
- **кластеризация** – разбиение множества объектов на группы на основании признаков этих объектов так, чтобы внутри групп объекты были похожи между собой сильнее, чем вне одной группы

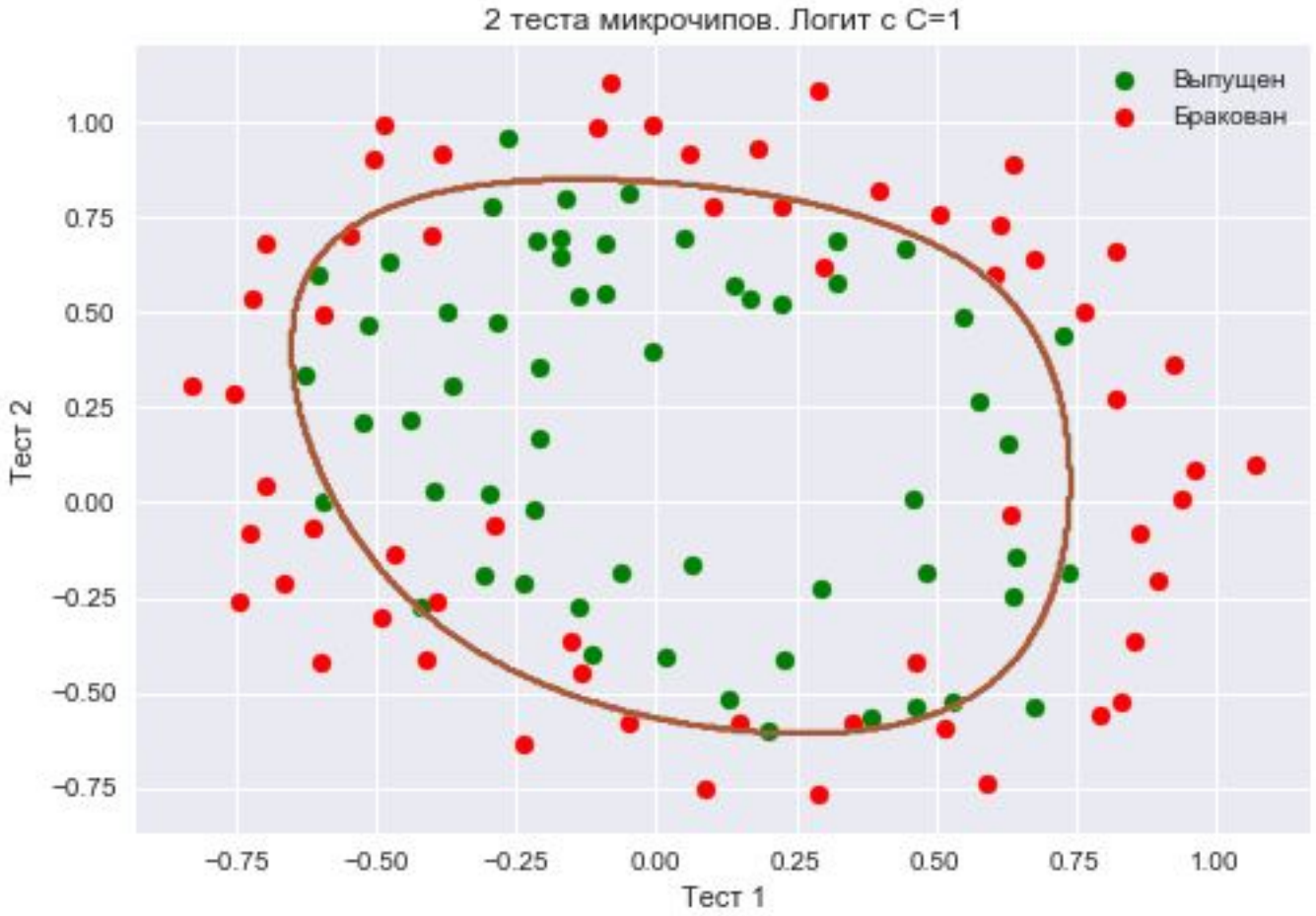
Задачи классификации и регрессии – это задачи обучения с учителем.



Линейная модель классификации



Пример нелинейного разделения классов



Confusion matrix (матрица ошибок классификации)

	$y = 1$	$y = -1$
$\hat{y} = 1$	True Positive (TP)	False Positive (FP)
$\hat{y} = -1$	False Negative (FN)	True Negative (TN)

Здесь $\hat{y}$ – это ответ алгоритма на объекте, а y – истинная метка класса на этом объекте.

Таким образом, ошибки классификации бывают двух видов: False Negative (FN) и False Positive (FP).

Метрики качества классификации

- Доля правильных ответов: $accuracy = \frac{TP + TN}{TP + TN + FP + FN}$

Малоинформативна в задачах с неравными классами.

Пример. Допустим, мы хотим оценить работу спам-фильтра почты. У нас есть 100 не-спам писем, 90 из которых наш классификатор определил верно, и 10 спам-писем, 5 из которых классификатор также определил верно. Предположим, класс1- спам, а класс -1 -не спам

$$accuracy = \frac{5 + 90}{5 + 90 + 10 + 5} = 0.864$$

Если мы просто будем предсказывать все письма как не-спам, то получим более высокую accuracy

$$accuracy = \frac{0 + 100}{0 + 100 + 0 + 10} = 0.909$$

	Спам	Не спам
Предсказан спам	5	10
Предсказан не спам	5	90

Метрики качества классификации

- precision (точность) и recall (полнота).

$$precision = \frac{TP}{TP + FP} \qquad recall = \frac{TP}{TP + FN}$$

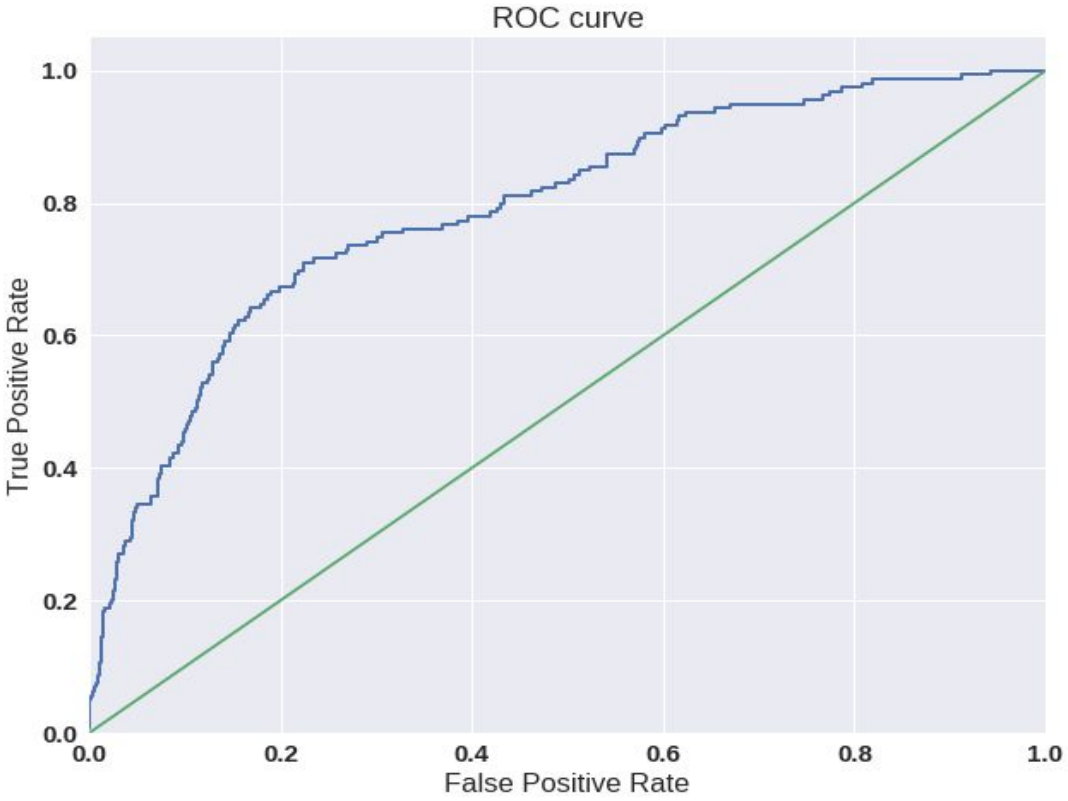
Precision показывает долю объектов, названных классификатором положительными и при этом действительно являющимися положительными, а recall показывает, какую долю объектов положительного класса из всех объектов положительного класса нашел алгоритм.

Precision не позволяет записывать все объекты в один класс, так как в этом случае растет значение FP. Recall демонстрирует способность алгоритма обнаруживать данный класс вообще, а precision — способность отличать этот класс от других классов.

AUC-ROC –площадь под кривой ошибок

$$TPR = \frac{TP}{TP + FN} \qquad FPR = \frac{FP}{FP + TN}$$

TPR - это полнота, а FPR показывает, какую долю из объектов отрицательного класса алгоритм предсказал неверно.



Кривая ошибок или **ROC-кривая** – графическая характеристика качества бинарного классификатора, зависимость доли верных положительных классификаций от доли ложных положительных классификаций при варьировании порога решающего правила.

AUC-ROC –площадь под кривой ошибок

В идеальном случае, когда классификатор не делает ошибок (FPR = 0, TPR = 1), площадь под кривой, равна 1; в противном случае, когда классификатор случайно выдает вероятности классов, AUC-ROC = 0.5. Каждая точка на графике соответствует выбору некоторого порога вероятности, разделяющего положительный и отрицательный класс.

Площадь под кривой в данном случае показывает качество алгоритма (больше — лучше), кроме этого, важной является крутизна самой кривой — мы хотим максимизировать TPR, минимизируя FPR, а значит, наша кривая в идеале должна стремиться к точке (0,1).

Критерий AUC-ROC устойчив к несбалансированным классам и может быть интерпретирован как вероятность того, что случайно выбранный положительный объект будет иметь более высокую вероятность быть положительно определенным данным классификатором, чем случайно выбранный отрицательный объект.

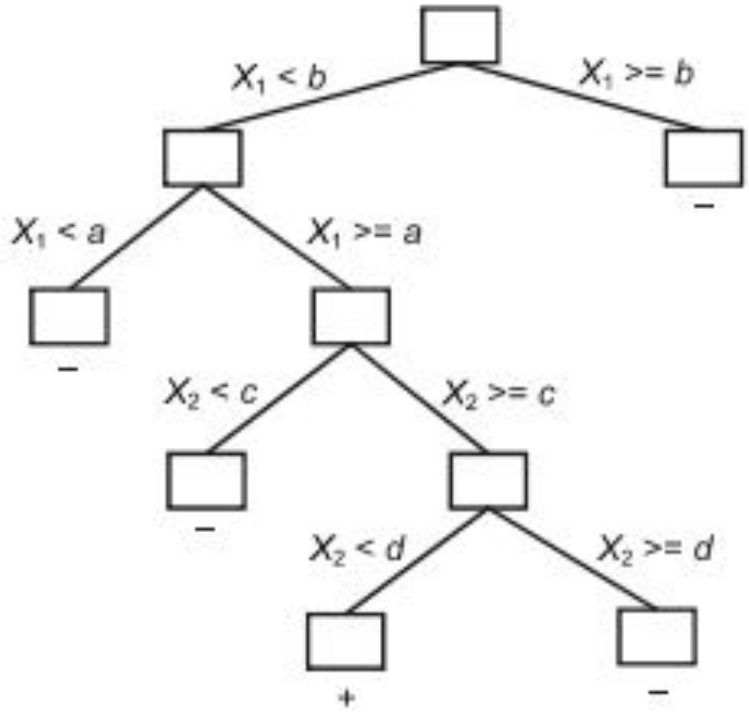
Чувствительность и специфичность

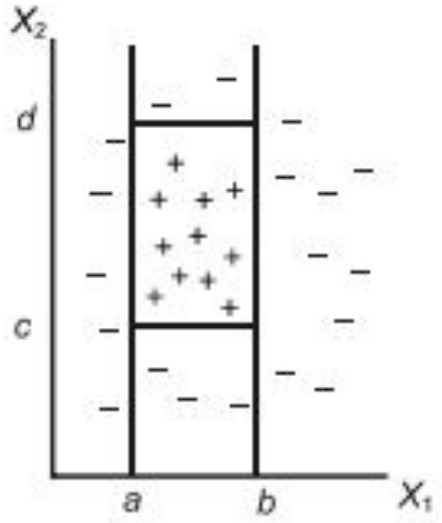
- Наряду с FPR и TPR при оценке качества классификации используют также понятия *чувствительности* и *специфичности*, которые изменяются в интервале [0,1]:
- *чувствительность* алгоритма совпадает с TPR (долей положительных объектов, правильно классифицированных алгоритмом);
- *специфичность* алгоритма определяется как 1-FPR (это доля отрицательных объектов, правильно классифицированных алгоритмом).
- Модель с высокой чувствительностью чаще дает истинный результат при наличии положительного исхода (хорошо обнаруживает положительные примеры). Наоборот, модель с высокой специфичностью чаще дает истинный результат при наличии отрицательного исхода (хорошо обнаруживает отрицательные примеры).

Дерево решений

Деревья решений - это метод, позволяющий предсказывать значения зависимой переменной в зависимости от соответствующих значений одной или нескольких предикторных (независимых) переменных. Применяется в задачах классификации и (реже) регрессии.

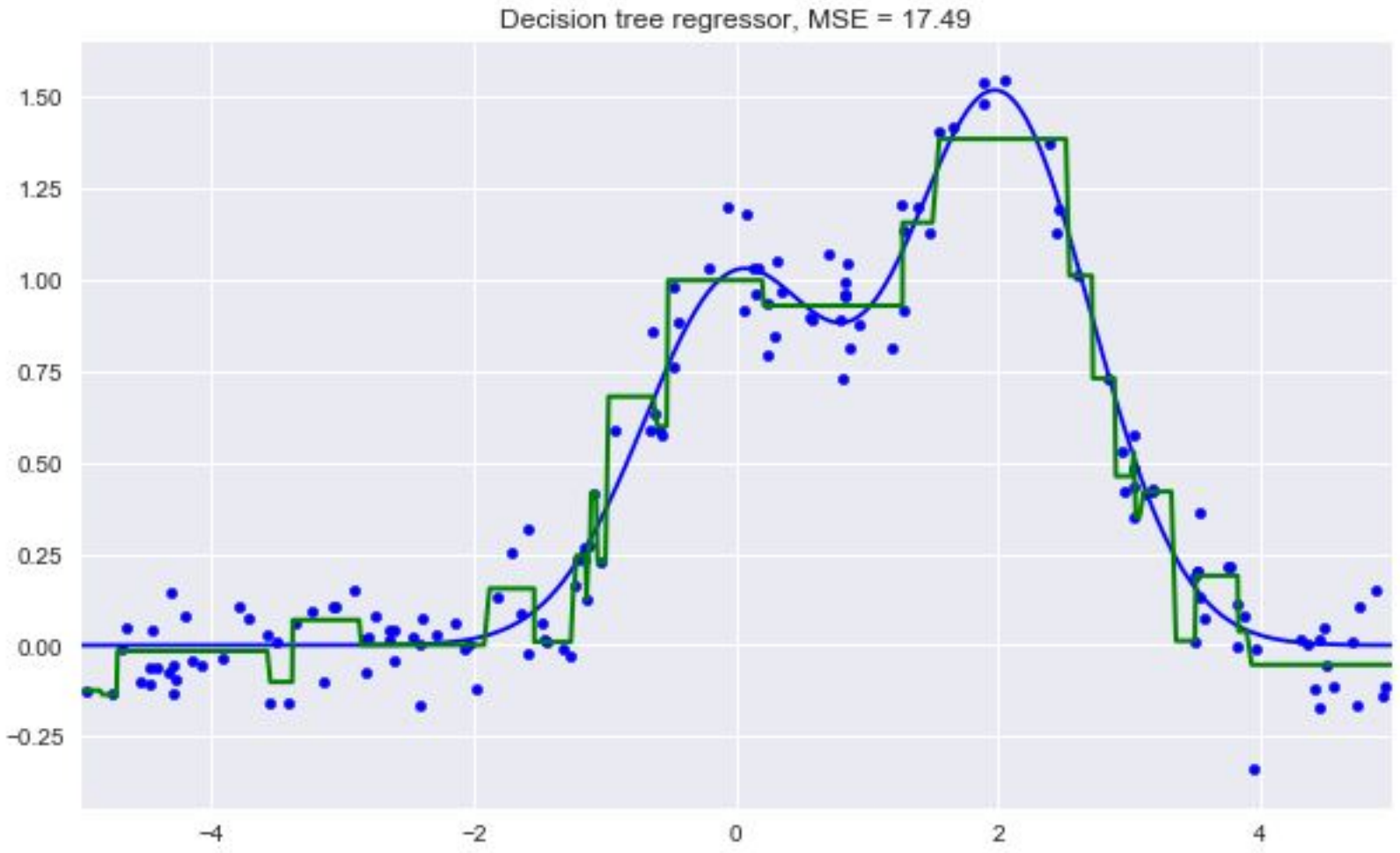
Графическая иллюстрация нелинейного разделения классов





На рисунки приведен пример классификации объектов по двум непрерывным признакам. Объекты, относящиеся к разным классам, отмечены знаками "+" и "–".

Использование деревьев решений в задачах регрессии



Этапы построения дерева решений

1. Выбор критерия точности прогноза
2. Выбор типа ветвления
3. Определение момента прекращения ветвлений
4. Определение "подходящих" размеров дерева

Выбор критерия точности прогноза

Accuracy, precision, recall – в задачах классификации

MSE,MAE– в задачах регрессии

Выбор типа ветвления (criterion)

- Есть различные способы выбирать очередной признак для текущего ветвления:
- Алгоритм ID3, где выбор атрибута происходит на основании прироста информации (*Gain*).
- Алгоритм C4.5 (улучшенная версия ID3), где выбор атрибута происходит на основании нормализованного прироста информации (*Gain Ratio*).
- Алгоритм CART где выбор атрибута происходит на основании индекса Джини.

Энтропия

Энтропия Шеннона для системы с s возможными состояниями:

- $H = -\sum_{i=1}^{s} p_i log_2 p_i$

p_i −вероятности нахождения системы в i − м состоянии
В нашем случае:
Предположим, что имеется множество A, состоящее из n элементов, обладающих свойством S, которое может принимать s различных значений, m_i - количество объектов множества A, имеющих i-е значение свойства S. Тогда

$$p_i = \frac{m_i}{n},$$

$$H(A, S) = -\sum_{i=1}^{s} \frac{m_i}{n} \log \frac{m_i}{n}.$$

Прирост информации (ID3)

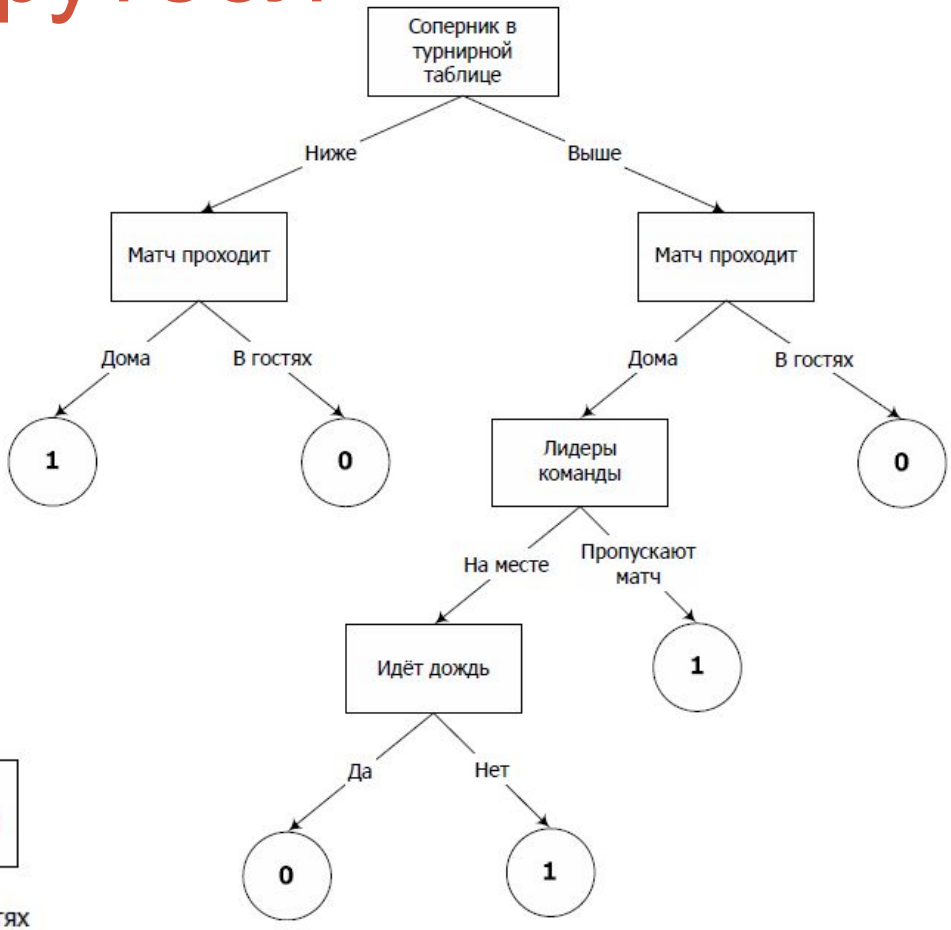
Предположим, что множество A элементов, характеризующихся свойством S, классифицировано посредством атрибута Q, имеющего q возможных значений. Тогда прирост информации (information gain) определяется как

$$\mathrm{Gain}(A, Q) = H(A, S) - \sum_{i=1}^{q} \frac{|A_i|}{|A|} H(A_i, S),$$

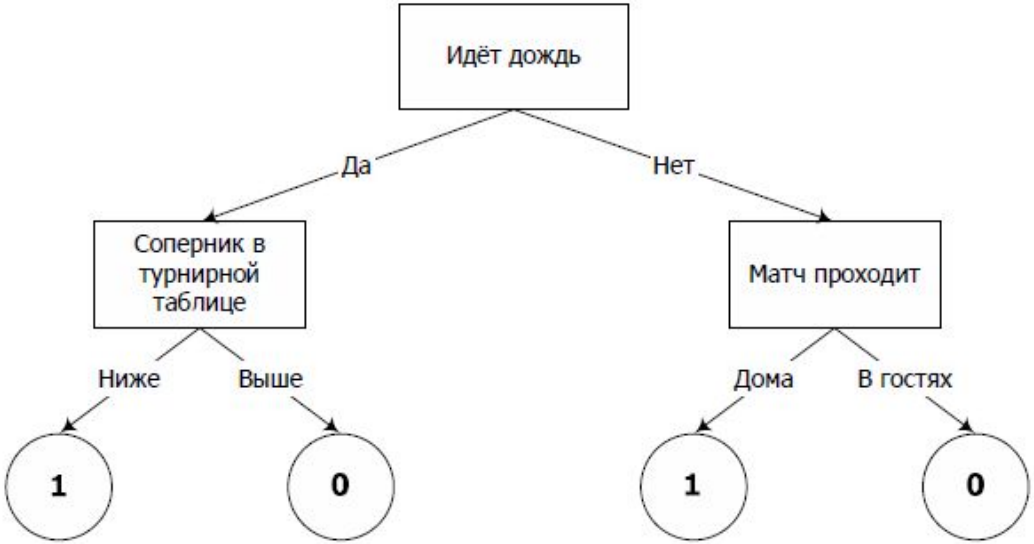
где A_i — множество элементов A, на которых атрибут Q имеет значение i.

Прогноз игры в футбол

Соперник	Играем	Лидеры	Дождь	Победа
Выше	Дома	На месте	Да	Нет
Выше	Дома	На месте	Нет	Да
Выше	Дома	Пропускают	Нет	Да
Ниже	Дома	Пропускают	Нет	Да
Ниже	В гостях	Пропускают	Нет	Нет
Ниже	Дома	Пропускают	Да	Да
Выше	В гостях	На месте	Да	Нет
Ниже	В гостях	На месте	Нет	???



Первый вариант дерева



Второй вариант дерева

Вычисление энтропии и прироста информации

$$H(A, \text{Победа}) = -\frac{4}{7}\log_2\frac{4}{7} - \frac{3}{7}\log_2\frac{3}{7} \approx 0.9852.$$

$$\text{Gain}(A, \text{Соперник}) = H(A, \text{Победа}) - \frac{4}{7}H(A_{\text{выше}}, \text{Победа}) - \frac{3}{7}H(A_{\text{ниже}}, \text{Победа}) \approx$$
$$\approx 0.9852 - \frac{4}{7}\left(-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2}\right) - \frac{3}{7}\left(-\frac{2}{3}\log_2\frac{2}{3} - \frac{1}{3}\log_2\frac{1}{3}\right) \approx 0.0202.$$

$$\text{Gain}(A, \text{Играем}) = H(A, \text{Победа}) - \frac{5}{7}H(A_{\text{дома}}, \text{Победа}) - \frac{2}{7}H(A_{\text{в гостях}}, \text{Победа}) \approx 0.4696.$$

$$\text{Gain}(A, \text{Лидеры}) = H(A, \text{Победа}) - \frac{3}{7}H(A_{\text{на месте}}, \text{Победа}) - \frac{4}{7}H(A_{\text{пропускают}}, \text{Победа}) \approx 0.1281.$$

$$\text{Gain}(A, \text{Дождь}) = H(A, \text{Победа}) - \frac{3}{7}H(A_{\text{да}}, \text{Победа}) - \frac{4}{7}H(A_{\text{нет}}, \text{Победа}) \approx 0.1281.$$

Нормализованный прирост информации (C4.5)

Проблема: прирост информации выбирает атрибуты, у которых

Gain Ratio учитывает не только количество информации, требуемое для записи результата, но и количество информации, требуемое для разделения по текущему атрибуту. Поправка:

$$\mathrm{SplitInfo}(A, Q) = -\sum_{i=1}^{q} \frac{|A_q|}{|A|} \log_2 \frac{|A_q|}{|A|},$$

Сам критерий — максимизация величины

$$\mathrm{GainRatio}(A, Q) = \frac{\mathrm{Gain}(A, Q)}{\mathrm{SplitInfo}(A, Q)}.$$

Индекс Gini (CART)

Для набора тестов A и свойства S, имеющего s значений, этот индекс вычисляется как

$$\mathrm{Gini}(A, S) = 1 - \sum_{i=1}^{s} \left(\frac{|A_i|}{|A|} \right)^2 .$$

Соответственно, для набора тестов A, атрибута Q, имеющего q значений, и целевого свойства S, имеющего s значений, индекс вычисляется следующим образом:

$$\mathrm{Gini}(A, Q, S) = \mathrm{Gini}(A, S) - \sum_{j=1}^{q} \frac{|A_j|}{|A|} \mathrm{Gini}(A_j, S).$$

Правила разбиения (CART)

- 1) Вектор, подаваемый на вход дерева может содержать как порядковые так и категориальные переменные.
- 2) В каждом узле разбиение идет только по *одной переменной*.

2.1) Если переменная числового типа, то в узле формируется правило вида $x_i <= c$. Где c – некоторый порог, который чаще всего выбирается как среднее арифметическое двух соседних *упорядоченных* значений переменной x_i обучающей выборки.

2.2) Если переменная категориального типа, то в узле формируется правило $x_i \in V(x_i)$, где $V(x_i)$ – некоторое непустое подмножество множества значений переменной x_i в обучающей выборке.

Следовательно, для n значений числового атрибута алгоритм сравнивает $n-1$ разбиений, а для категориального $(2^{n-1} – 1)$.

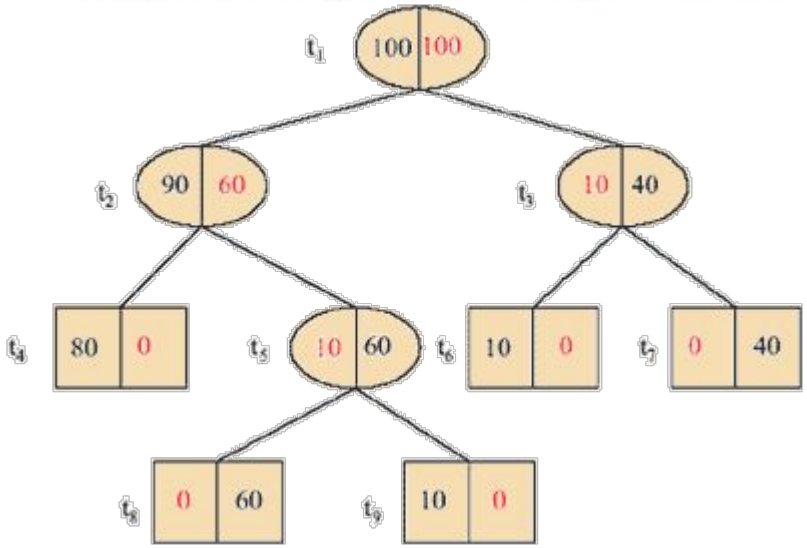
Правила остановки

- **Минимальное число объектов, при котором выполняется расщепление (min_samples_split).** В этом варианте ветвление прекращается, когда все терминальные вершины, содержащие более одного класса, содержат не более чем заданное число объектов (наблюдений).
- **Минимальное число объектов в листьях (min_samples_leaf)**
- **Доля неклассифицированных.** В этом варианте ветвление прекращается, когда все терминальные вершины, содержащие более одного класса, содержат не более чем заданную долю неправильно классифицированных объектов (наблюдений).
- **Максимальная глубина деревьев (max_depth)**

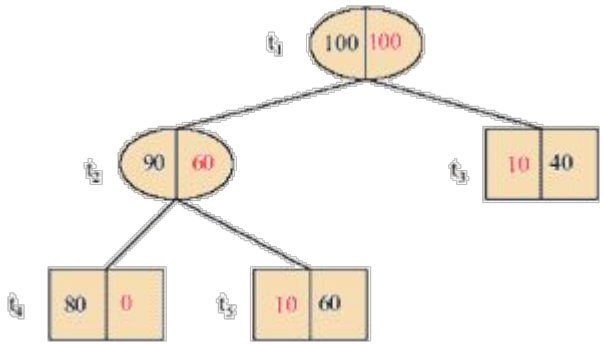
Механизм отсечения дерева (CART)

Обозначим $|T|$ – число листов дерева, $R(T)$ – ошибка классификации дерева, равная отношению числа неправильно классифицированных примеров к числу примеров в обучающей выборке. Определим $C_\alpha(T)$ – полную стоимость (оценку/показатель затраты-сложность) дерева T как:

$C_\alpha(T) = R(T) + \alpha * |T|$, где $|T|$ – число листов (терминальных узлов) дерева, – некоторый параметр, изменяющийся от 0 до + ∞. Полная стоимость дерева состоит из двух компонент – ошибки классификации дерева и штрафа за его сложность.

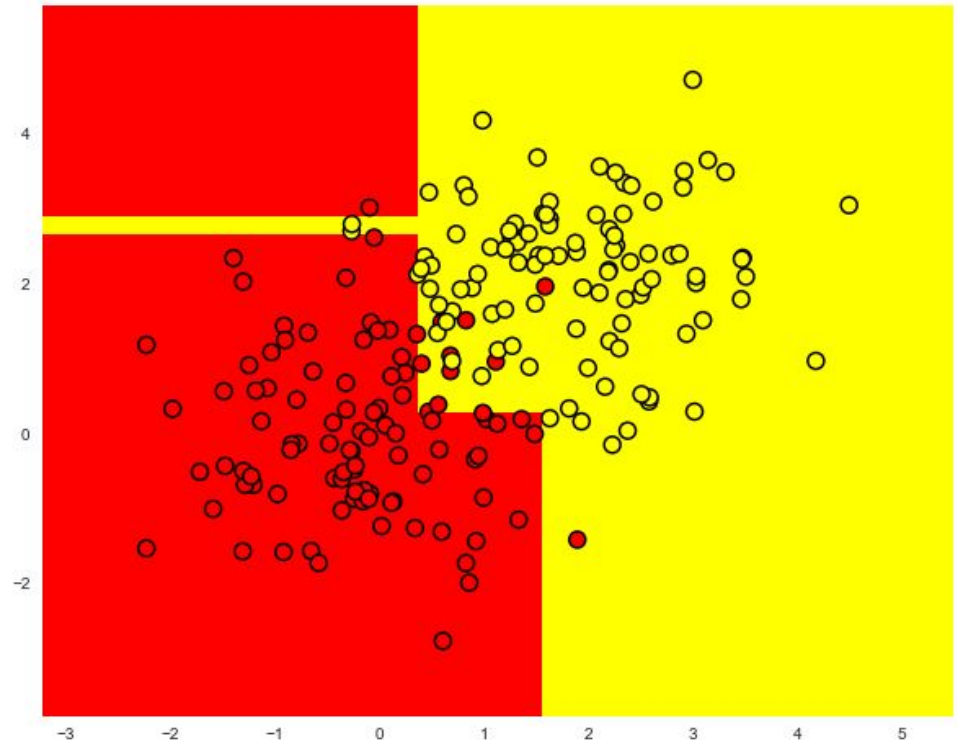


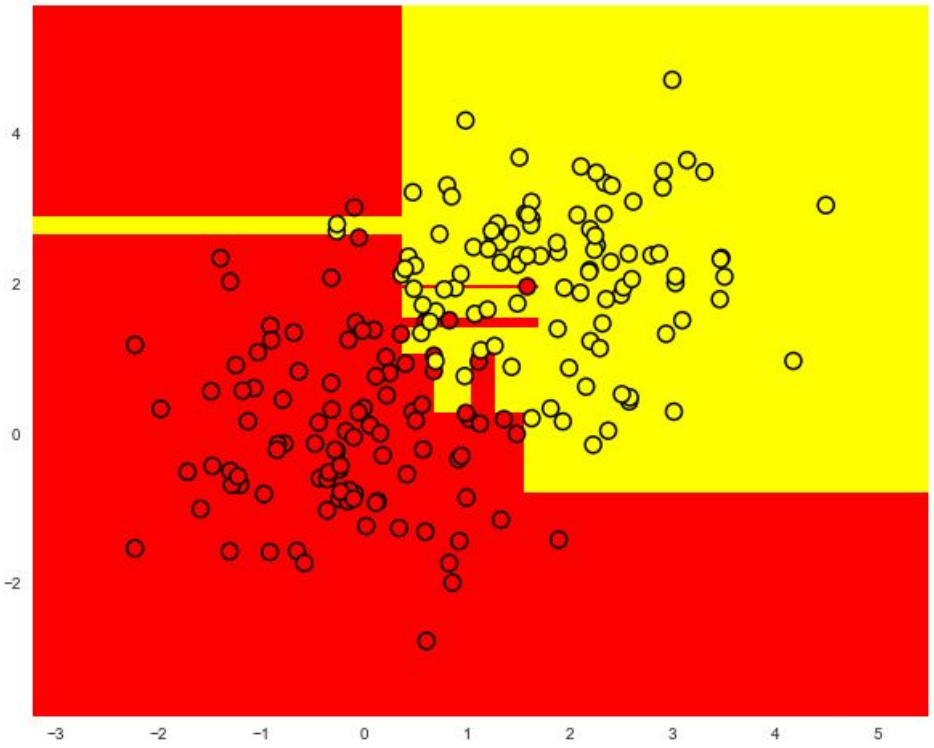
$\alpha_1 = 0$,



$\alpha_2 = 1/20$,

Иллюстрация переобучения

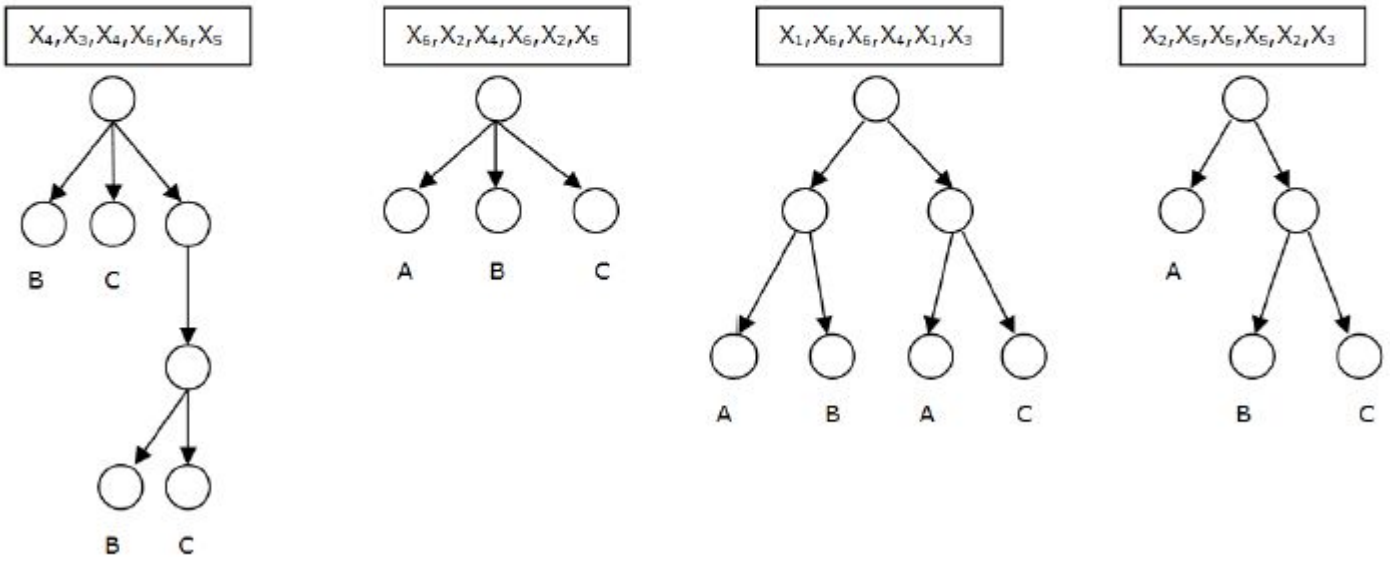




Случайный лес (Random forest)

- Случайный лес — алгоритм машинного обучения, заключающийся в использовании комитета (ансамбля) деревьев решений.

Тренировочный набор:
$\{(X_1, A), (X_2, A), (X_3, B), (X_4, B), (X_5, C), (X_6, C)\}$



Обучение случайного леса

- Пусть обучающая выборка состоит из *N* примеров, размерность пространства признаков равна *M*, и задан параметр *m* (в задачах классификации обычно m ≈ $\sqrt{M}$.
- Все деревья комитета строятся независимо друг от друга по следующей процедуре:
- Сгенерируем случайную подвыборку **с повторением** размером *N* из обучающей выборки. (Таким образом, некоторые примеры попадут в неё несколько раз, а в среднем $N\left(1-\frac{1}{N}\right)^N$, т.е. примерно *N*/e примеров не войдут в неё вообще)
- Построим дерево, классифицирующее примеры данной подвыборки, причём в ходе создания очередного узла дерева будем выбирать признак, на основе которого производится разбиение, не из всех *M* признаков, а лишь из *m* случайно выбранных.
- Дерево строится до полного исчерпания подвыборки и не подвергается процедуре отсечения.
- Классификация объектов проводится путём голосования: каждое дерево комитета относит классифицируемый объект к одному из классов, и побеждает класс, за который проголосовало наибольшее число деревьев.
- Оптимальное число деревьев (**n_estimators)** подбирается таким образом, чтобы минимизировать ошибку классификатора на валидационной выборке.

Достоинства и недостатки

- Достоинства:
- Способность эффективно обрабатывать данные с большим числом признаков и классов.
- Нечувствительность к масштабированию значений признаков.
- Одинаково хорошо обрабатываются как непрерывные, так и дискретные признаки. Существуют методы построения деревьев по данным с пропущенными значениями признаков.
- Существуют методы оценивания значимости отдельных признаков в модели.
- Высокая параллелизуемость и масштабируемость.

Недостатки:

Большой размер получающихся моделей.