Decision tree, bagging, random forest

Вспоминаем

Граф

Лист

Дерево

Классификация

Регрессия

Энтропия

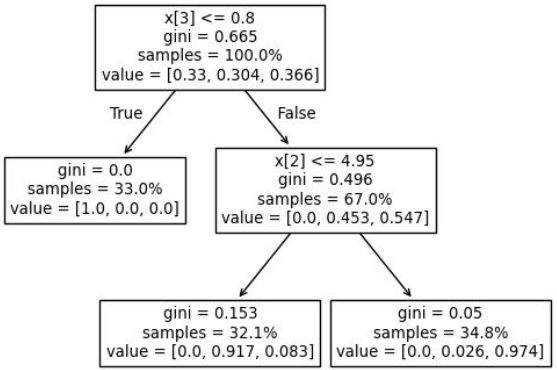
Бутстрап

параметры и гиперпараметры

Решающее дерево

Дерево состоит из узлов, которые представляют собой тесты на определённые атрибуты данных, и ветвей, которые обозначают возможные результаты тестов, ведущие к новым узлам или листьям. Листьями называются конечные узлы, которые представляют собой решения (классификационные метки или числовые значения для регрессии).

```
tree.plot_tree(clf, proportion=True)
plt.show()
```



И как мы будем решать с помощью этого алгоритма задачи?

Дерево на каждом шаге смотрит на признаки наших объектов и принимает решение, используя трешхолд

И повторяем до упора

$$B_{j,t}(x_i) = [x_{ij} \leq t]$$

Жадная оптимизация

Выбор признака и порога: На каждом узле алгоритм перебирает все доступные признаки и значения порогов, чтобы найти такое разделение, которое наилучшим образом разделяет данные. Это может быть оценено с помощью метрик, таких как прирост информации (энтропия), индекс Джини или среднеквадратичная ошибка.

Рекурсивное продолжение: После выбора оптимального признака и порога данные разделяются на две или более группы. Алгоритм повторяет процесс на каждой из подгрупп, снова выбирая наилучший признак и порог для следующего уровня дерева.

Остановка построения дерева: Жадная оптимизация продолжается до тех пор, пока не будет достигнут один из критериев остановки, например, максимальная глубина дерева, минимальное число образцов в узле или отсутствие улучшения разделения.

Преимущества жадной оптимизации

- **Простота и скорость**: Жадная оптимизация позволяет быстро строить решающие деревья, так как на каждом шаге принимается только локально оптимальное решение.
- **Интерпретируемость**: Алгоритм строит дерево последовательно, что делает его легко интерпретируемым и визуализируемым.

Недостатки жадной оптимизации

- **Локальный минимум**: Жадный подход может приводить к тому, что алгоритм застрянет в локально оптимальном решении, которое не является глобально оптимальным. Это означает, что дерево может не находить наилучшее возможное разделение данных.
- **Чрезмерная подгонка (overfitting)**: Без регуляризации или обрезки (pruning) решающее дерево может стать слишком сложным и подстроиться под шум в данных.

Мера энтропии в информации (классификация)

Критерий Джини (Gini impurity):

- Это мера чистоты узла, используемая для оценки эффективности разделения. Чем меньше значение критерия Джини, тем чище узел, то есть тем более однородны данные в нём (данные принадлежат одному классу).
- Дерево выбирает признак, который минимизирует средневзвешенное значение критерия Джини для дочерних узлов.

$$Gini = 1 - \sum_{i=1}^{n} p_i^2$$

Энтропия (Information Gain):

- Это мера неопределённости в данных, которая уменьшается при хороших разделениях. Чем меньше энтропия, тем чище узел.
- Дерево выбирает признак, который максимизирует информационный прирост, то есть разницу между энтропией до и после разделения.

$$Entropy = -\sum_{i=1}^{n} p_i \log_2 p_i$$

В задачах регрессии

Среднеквадратичная ошибка (Mean Squared Error, MSE):

- Оценивает качество разделения, вычисляя разницу между фактическими и предсказанными значениями.
- Дерево выбирает признак, который минимизирует средневзвешенное значение MSE для дочерних узлов.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2$$

Средняя абсолютная ошибка (Mean Absolute Error, MAE):

- Использует абсолютные разности между фактическими и предсказанными значениями.
- Как и в случае с MSE, дерево выбирает признак, минимизирующий средневзвешенное значение MAE.

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}|$$

Как определяем трешхолд?

Перебор значений признака:

- Для каждого признака рассматриваются все его возможные значения в обучающей выборке.
- Алгоритм проверяет, какое пороговое значение лучше всего разделяет данные на два подмножества. Это делается путем сортировки значений признака и поиска подходящих точек разделения.

Определение кандидатов на трешхолд:

- Значения, которые находятся между двумя соседними различными точками значений признака, рассматриваются как кандидаты на трешхолд.
- Обычно кандидаты выбираются как среднее между двумя соседними значениями.

Оценка качества разделения:

- Для каждого кандидата на трешхолд алгоритм оценивает качество разделения данных на два подмножества — одно со значениями меньше или равными трешхолду и другое с большими значениями.
- Качество разделения измеряется с использованием выбранного критерия (например, индекса Джини или энтропии), который оценивает "нечистоту" получившихся подмножеств.

Выбор оптимального трешхолда:

- Алгоритм выбирает тот трешхолд, который минимизирует нечистоту или максимизирует "чистоту" (то есть наибольшую однородность классов) в каждом из подмножеств.
- Процесс повторяется для всех признаков, и выбирается признак с лучшим трешхолдом, который дает наибольшее улучшение качества разделения данных.

Преимущества

Простота и интерпретируемость: Деревья легко визуализировать, и они интуитивно понятны даже для людей, не знакомых с машинным обучением.

Мало требований к подготовке данных: Решающее дерево не требует масштабирования данных или их нормализации.

Работа с категориальными и числовыми данными: Поддерживает одновременно оба типа данных.

Обработка выбросов и пропущенных значений: Не сильно чувствительно к выбросам, а также может быть настроено для обработки пропущенных данных.

Недостатки

Склонность к переобучению: Деревья могут легко переобучиться на данных, особенно если не применять регуляризацию (ограничение глубины дерева, минимальное количество примеров в узле и т.д.).

Нестабильность: Небольшие изменения в данных могут привести к значительным изменениям в структуре дерева, что делает результаты непостоянными.

Жадный алгоритм построения: Решающее дерево строится, выбирая лучший признак на каждом шаге, что может привести к локально оптимальному, но не глобально лучшему решению.

Плохо работает с линейно разделимыми данными: Если зависимость в данных линейная, более простые модели (например, логистическая регрессия) могут оказаться более эффективными.

Критерии остановки

Максимальная глубина дерева

Минимальное количество образцов в узле

Минимальное количество образцов для разбиения

Минимальное улучшение качества

Одноклассовые узлы

Постоянство

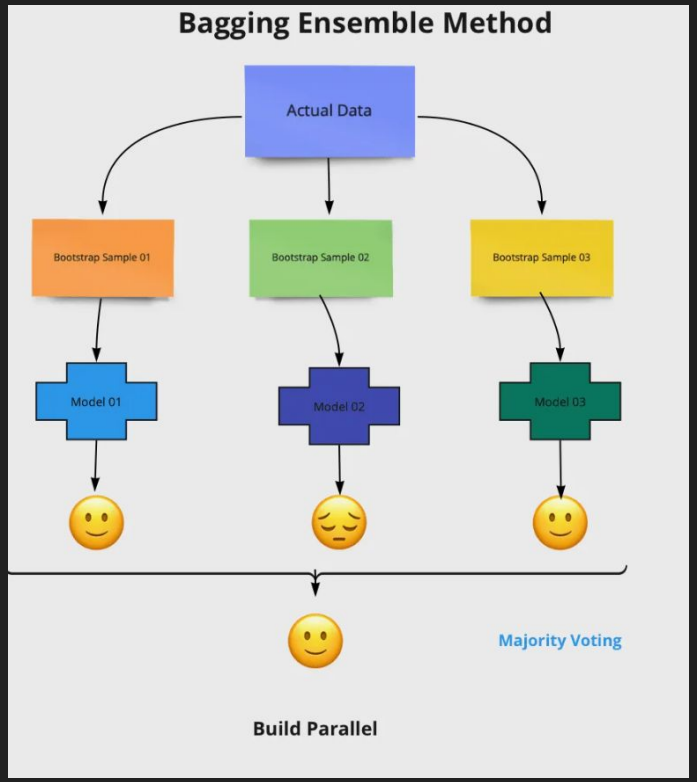
Еще особенности деревьев

1. Обработка NaN - деревьем все равно
2. Работа с категориальными признаками:
    - Кодирование
    - Сопоставление с конкретным значением
    - Catboost))

Бэггинг

Ансамбль из деревьев, обученных на подвыборках (метод бутстрепа).

Решение принимается по большинству голосов за конкретный класс (классификация) или усредняя предсказания (регрессия)

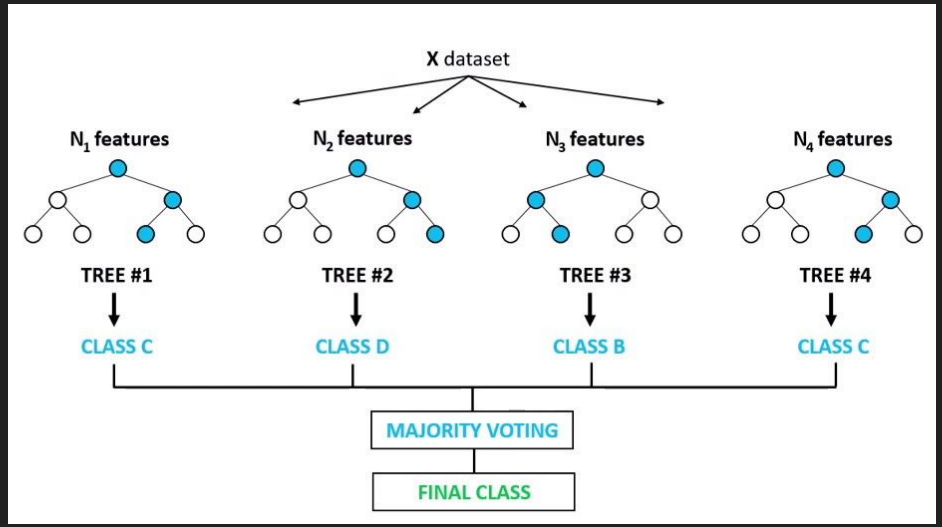


Random forest

Ансамбль из деревьев, обученных на подвыборках (метод бутстрепа).

Решение принимается по большинству голосов за конкретный класс (классификация) или усредняя предсказания (регрессия)

Добавляется ограничение на выбор признаков



Преимущества

- **Устойчивость к переобучению**: За счет ансамблевого подхода и случайности Random Forest менее подвержен переобучению по сравнению с отдельными деревьями решений.
- **Обработка больших объемов данных**: Может работать с большим количеством признаков и образцов, легко справляется с пропущенными значениями.
- **Оценка важности признаков**: Random Forest предоставляет методы для оценки значимости различных признаков, что помогает в интерпретации модели.
- **Скорость**: Обучение и предсказание могут быть выполнены параллельно, что увеличивает скорость работы.

Недостатки

- **Сложность модели**: Из-за большого количества деревьев интерпретировать результаты Random Forest может быть сложнее, чем для одиночного дерева решений.
- **Время и память**: Хотя Random Forest может быть быстрее, чем некоторые другие алгоритмы, он все же требует значительных вычислительных ресурсов, особенно для очень больших наборов данных.

Out of bag

Принципы работы OOB:

- **Бутстреп-выборки - ни одна из моделей не видела датасет целиком**
- **Оценка модели**:
  - После того как ансамбль моделей (например, случайный лес) обучен, можно использовать OOB-примеры для оценки производительности. Для каждого экземпляра данных, который не использовался для обучения, модель предсказывает класс (или значение) на основе других моделей, которые были обучены на других данных.
  - Оценка точности производится путем сравнения предсказанных значений с истинными значениями для OOB-примеров.
-

Преимущества

Нет необходимости в отдельном тестовом наборе: OOB-оценка позволяет использовать все данные для обучения, сохраняя при этом возможность оценить качество модели, что особенно полезно в случае ограниченного объема данных.

Быстрая и эффективная оценка: OOB-оценка позволяет быстро получить представление о производительности модели, так как не требует дополнительного времени на разделение данных на обучающую и тестовую выборки.

Предоставляет надежные оценки: Оценка на OOB-примерах дает стабильные и надежные результаты, так как она основана на данных, которые не были использованы для обучения.

Недостатки

Ограничения по объему данных: Для небольших наборов данных количество OOB-примеров может быть недостаточно для получения статистически значимых оценок.

Не всегда может быть доступен: В некоторых случаях, если размер выборки очень мал, может быть трудно использовать OOB-оценку, так как много наблюдений может не иметь OOB-примера.