Введение в методы машинного обучения

Задача регрессии

Ксения Балабаева

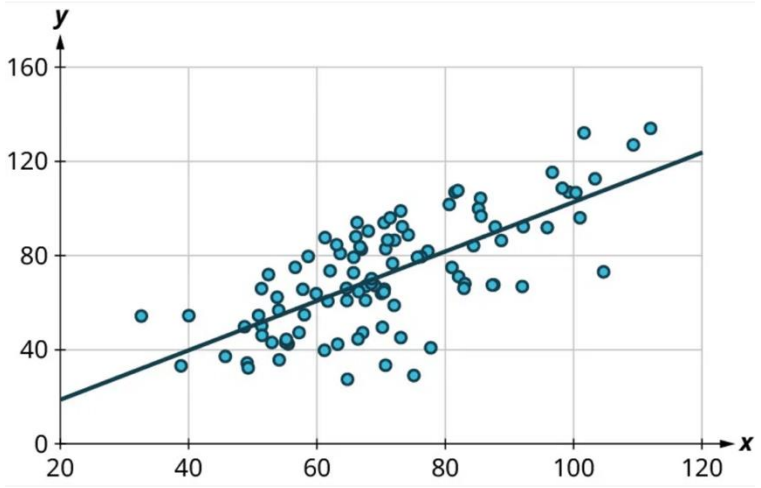
Задача регрессии в машинном обучении

Задача регрессии является одной из **фундаментальных задач машинного обучения с учителем**. В отличие от классификации, которая предсказывает дискретные метки классов, регрессия направлена на предсказание **непрерывных числовых значений**.

Модель обучается находить зависимость между входными признаками и целевой переменной, минимизируя ошибку между предсказанными и реальными значениями.

Типичные примеры задач регрессии

Прогнозирование цен на недвижимость, температуры воздуха, объема продаж, стоимости акций, потребления энергии, времени доставки, дохода населения



Ключевые характеристики задачи регрессии

01

Целевая переменная

Непрерывная числовая величина (цена, температура, доход), которую модель должна предсказать на основе входных признаков

02

Входные признаки

Набор переменных (числовых или категориальных), описывающих объект: площадь квартиры, количество комнат, район расположения

03

Функция предсказания

Модель обучается находить зависимость f(x) = y, которая минимизирует ошибку между предсказанными и реальными значениями

04

Применение

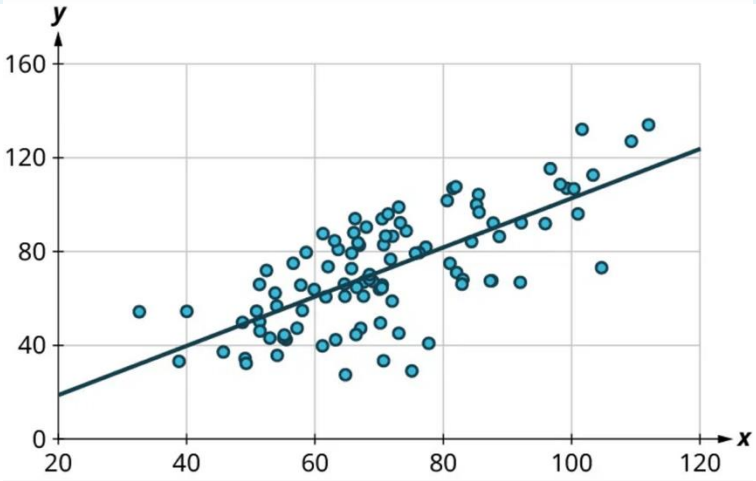
Финансовый анализ, прогнозирование спроса, оценка рисков, медицинская диагностика, инженерные расчеты

Линейная регрессия — простота и интерпретируемость

Линейная регрессия является базовым и наиболее **интерпретируемым** методом регрессионного анализа. Модель предполагает **линейную зависимость** между признаками и целевой переменной. Модель минимизирует сумму квадратов отклонений (MSE) между предсказанными и реальными значениями методом наименьших квадратов. Оптимальные веса находятся аналитически через матричные операции или итеративно через градиентный спуск.

$$y = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$$

где w — веса (коэффициенты), x — признаки, y — целевая переменная



Линейная регрессия — простота и интерпретируемость

Линейная регрессия является базовым и наиболее **интерпретируемым** методом регрессионного анализа. Модель предполагает **линейную зависимость** между признаками и целевой переменной. Модель минимизирует сумму квадратов отклонений (MSE) между предсказанными и реальными значениями методом наименьших квадратов. Оптимальные веса находятся аналитически через матричные операции или итеративно через градиентный спуск.

$$y = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$$

где w — веса (коэффициенты), x — признаки, y — целевая переменная

Преимущества

- Высокая скорость обучения и предсказания
- Простая интерпретация коэффициентов — каждый показывает вклад признака
- Отсутствие гиперпараметров для настройки в базовой версии
- Аналитическое решение через матричные операции
- Хорошо работает при линейных зависимостях в данных

Ограничения

- Предполагает линейную зависимость между признаками и целевой переменной
- Чувствительна к выбросам из-за квадратичной функции потерь
- Не улавливает сложные нелинейные паттерны в данных
- Требует нормализации признаков для корректной работы
- Может страдать от мультиколлинеарности признаков

Гиперпараметры — настройка поведения алгоритма

Гиперпараметры — это параметры модели, которые задаются до начала обучения и не обновляются в процессе оптимизации. В отличие от обучаемых параметров (весов), гиперпараметры контролируют структуру модели, процесс обучения и механизмы регуляризации.

Отличие от обучаемых параметров: Веса модели (например, коэффициенты линейной регрессии) обновляются автоматически в процессе обучения через оптимизацию функции потерь. Гиперпараметры же устанавливаются вручную или через автоматизированный поиск до начала обучения и определяют, как именно будет происходить этот процесс.

01

Структурные

Определяют архитектуру модели: глубина дерева, количество слоев нейросети, степень полинома. Влияют на выразительную способность модели и её сложность.

02

Регуляризационные

Контролируют сложность и предотвращают переобучение: коэффициенты L1/L2 регуляризации, dropout, ранняя остановка. Помогают модели лучше обобщать на новые данные.

03

Оптимизационные

Управляют процессом обучения: learning rate, размер батча, количество эпох, momentum. Влияют на скорость и стабильность сходимости алгоритма оптимизации.

04

Алгоритмические

Специфичны для конкретных алгоритмов: количество деревьев в бустинге, количество соседей в KNN, тип ядра в SVM. Определяют особенности работы конкретного метода.

Гиперпараметры линейной регрессии

Хотя базовая линейная регрессия не имеет гиперпараметров, на практике используются регуляризованные версии, которые добавляют штрафной член к функции потерь для контроля сложности модели и предотвращения переобучения.

Ridge регрессия (L2-регуляризация)

Добавляет штраф пропорциональный квадрату весов. Уменьшает веса, но не обнуляет их полностью, что позволяет сохранить все признаки в модели.

Штраф: $\alpha \cdot \sum w^2$

Параметр α: обычно от 0.01 до 100

Lasso регрессия (L1-регуляризация)

Добавляет штраф пропорциональный модулю весов. Может обнулять веса менее важных признаков, выполняя автоматический отбор признаков.

Штраф: $\alpha \cdot \sum |w|$

Параметр α: обычно от 0.01 до 100

ElasticNet

Комбинирует L1 и L2 регуляризацию, объединяя преимущества обоих методов. Параметр ρ контролирует баланс между Ridge и Lasso.

Штраф: $\alpha \cdot (\rho \cdot \sum |w| + (1-\rho) \cdot \sum w^2)$

Параметр ρ: от 0 до 1 (баланс L1/L2)

Нормализация признаков

Стандартизация (mean=0, std=1) или min-max масштабирование критически важны для корректной работы регуляризации, так как штрафы зависят от масштаба весов.

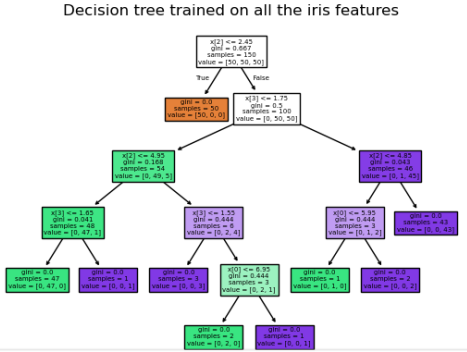
Стандартизация: $(x - \mu) / \sigma$

Применяется перед обучением модели

Деревья решений — иерархическое разбиение пространства признаков

Дерево решений строит иерархическую структуру из узлов-условий и листьев-предсказаний. В каждом узле алгоритм выбирает признак и порог, которые наилучшим образом разделяют данные на две группы с минимальной дисперсией внутри каждой группы.

Алгоритм начинает с корневого узла, содержащего все данные, перебирает все признаки и возможные пороги разбиения, выбирая разбиение, максимально снижающее суммарную дисперсию в дочерних узлах. Процесс рекурсивно повторяется до достижения критериев остановки.

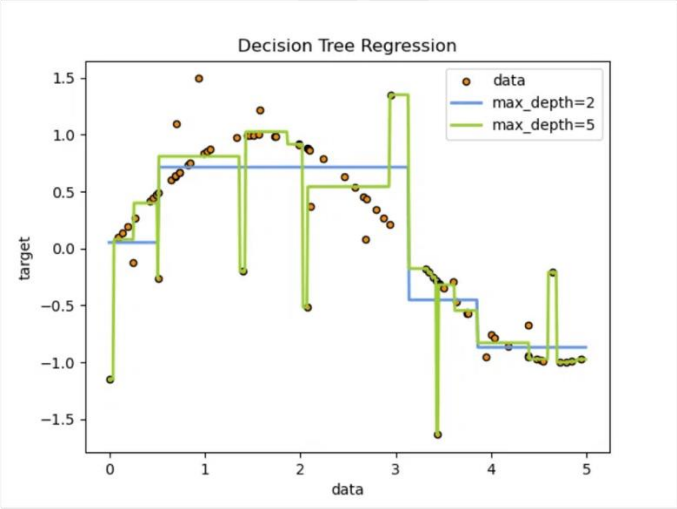


Преимущества

Не требует нормализации признаков, автоматически учитывает нелинейности и взаимодействия, работает с категориальными признаками, легко визуализируется и интерпретируется

Недостатки

Склонность к переобучению на зашумленных данных, высокая вариативность (малое изменение данных сильно меняет структуру), ступенчатые предсказания



Ключевые гиперпараметры деревьев решений

Контроль сложности дерева критичен для баланса между недо- и переобучением. Правильная настройка этих параметров определяет способность модели обобщать знания на новые данные.

01 max_depth

Ограничивает максимальное количество уровней в дереве. Малые значения предотвращают переобучение, но могут привести к недообучению.

Типичные значения: 3-10. Для простых задач 3-5, для сложных 7-10

02 min_samples_split

Минимальное количество объектов в узле, необходимое для его разбиения. Большие значения создают более консервативные деревья.

Типичные значения: 2-20. По умолчанию 2, для больших данных 10-20

03 min_samples_leaf

Минимальное количество объектов в листовом узле. Предотвращает создание листьев с единичными наблюдениями, снижая переобучение.

Типичные значения: 1-10. Обычно 1-5, для зашумленных данных 5-10

04 max_features

Количество признаков для рассмотрения при поиске лучшего разбиения. Добавляет случайность и снижает корреляцию между деревьями в ансамблях.

Типичные значения: 'sqrt', 'log2', или число/доля признаков

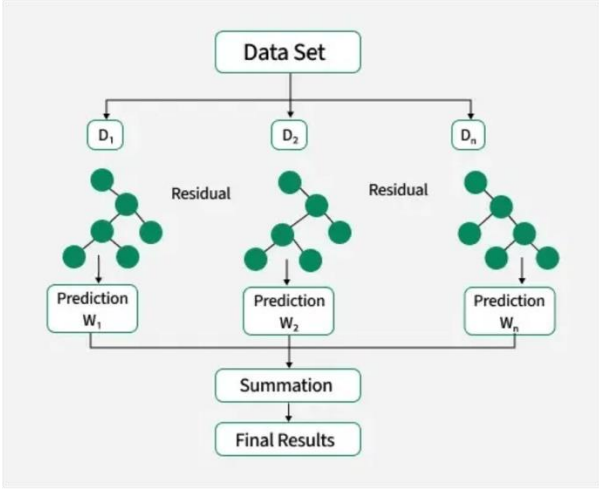
XGBoost — градиентный бустинг для максимальной точности

XGBoost (Extreme Gradient Boosting) является одним из наиболее эффективных алгоритмов машинного обучения, основанным на технике градиентного бустинга.

Метод последовательно строит **ансамбль слабых моделей** (обычно неглубоких деревьев), где каждое новое дерево обучается **предсказывать остаточные ошибки** всех предыдущих деревьев.

Принцип работы алгоритма

Алгоритм начинает с простого предсказания (например, среднего значения целевой переменной). Затем итеративно добавляет новые деревья, каждое из которых обучается на градиентах функции потерь относительно текущих предсказаний. Финальное предсказание получается суммированием предсказаний всех деревьев с весовыми коэффициентами.



Последовательное обучение

Каждое новое дерево фокусируется на исправлении ошибок предыдущих моделей, постепенно улучшая качество предсказаний. XGBoost использует регуляризацию, обработку пропущенных значений и параллельные вычисления для повышения производительности.

Гиперпараметры XGBoost: параметры бустинга и деревьев

XGBoost имеет обширный набор гиперпараметров, разделенных на несколько категорий: параметры бустинга, параметры деревьев и параметры регуляризации. Правильная настройка критична для достижения оптимальной производительности.

01

n_estimators и learning_rate

Количество деревьев в ансамбле и скорость обучения. Малый learning_rate требует больше деревьев, но улучшает обобщающую способность модели за счет более плавного обучения.

Типичные значения: n_estimators: 50-1000, learning_rate: 0.01-0.3

02

max_depth и min_child_weight

Максимальная глубина отдельных деревьев и минимальная сумма весов в листовом узле. Контролируют сложность отдельных деревьев и предотвращают переобучение на уровне базовых моделей.

Типичные значения: max_depth: 3-10, min_child_weight: 1-10

Гиперпараметры XGBoost: сэмплирование и регуляризация

03

subsample и colsample_bytree

Доля объектов и признаков, используемых для обучения каждого дерева. Добавляют случайность в процесс обучения, снижают переобучение и ускоряют вычисления за счет работы с подвыборками данных.

Типичные значения: subsample: 0.5-1.0, colsample_bytree: 0.5-1.0

04

gamma, alpha (L1), lambda (L2)

Параметры регуляризации: gamma — минимальное снижение функции потерь для разбиения узла, alpha — L1-регуляризация весов, lambda — L2-регуляризация весов. Обеспечивают дополнительный контроль сложности модели.

Типичные значения: gamma: 0-5, alpha: 0-1, lambda: 0-10

Функции потерь — математическая основа обучения

Функция потерь является **центральным элементом процесса обучения** модели машинного обучения. Она определяет численную меру качества предсказаний модели, которую алгоритм оптимизации стремится минимизировать.
Выбор функции потерь зависит от **специфики задачи, распределения данных и требований к модели**.

MSE (Mean Squared Error)

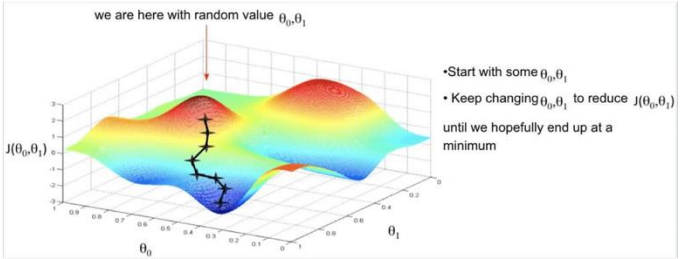
$$L = (1/n) \cdot \sum(y - \hat{y})^2$$

Наиболее популярная функция, сильно штрафует большие ошибки из-за квадрата. Чувствительна к выбросам. Предсказывает среднее значение целевой переменной.

MAE (Mean Absolute Error)

$$L = (1/n) \cdot \sum|y - \hat{y}|$$

Более робастна к выбросам, так как использует модуль вместо квадрата. Предсказывает медиану вместо среднего. Все ошибки взвешены одинаково.



Дополнительные функции потерь для регрессии

Huber Loss

L = MSE для малых ошибок, MAE для больших

Комбинирует MSE для малых ошибок и MAE для больших. Параметр δ контролирует порог переключения. Баланс между чувствительностью и робастностью к выбросам.

Log-Cosh Loss

L = ∑log(cosh(y - ŷ))

Гладкая аппроксимация MAE, дважды дифференцируемая. Менее чувствительна к выбросам, чем MSE. Хорошо работает с градиентным спуском.

Методы поиска оптимальных гиперпараметров: Grid и Random Search

Систематический подбор гиперпараметров повышает качество модели на 10-30%. Различные методы отличаются балансом между вычислительной эффективностью и качеством найденного решения.

01

Grid Search (Поиск по сетке)

Перебирает все комбинации из заданного набора значений для каждого гиперпараметра. Гарантирует нахождение лучшей комбинации в пределах заданной сетки, но вычислительно затратен при большом количестве параметров.

Особенности: Полный перебор, детерминированный результат, экспоненциальный рост времени с увеличением параметров

02

Random Search (Случайный поиск)

Случайно сэмплирует комбинации гиперпараметров из заданных распределений. Часто эффективнее Grid Search при ограниченном бюджете вычислений, так как исследует более широкое пространство параметров.

Особенности: Стохастический подход, лучше для высокоразмерных пространств, легко распараллеливается

Методы валидации — оценка обобщающей способности

Валидация является критически важным этапом разработки модели машинного обучения. Она позволяет оценить, насколько хорошо модель будет работать на новых, ранее не виденных данных, и предотвратить переобучение — ситуацию, когда модель отлично работает на обучающей выборке, но плохо обобщает на реальные данные.

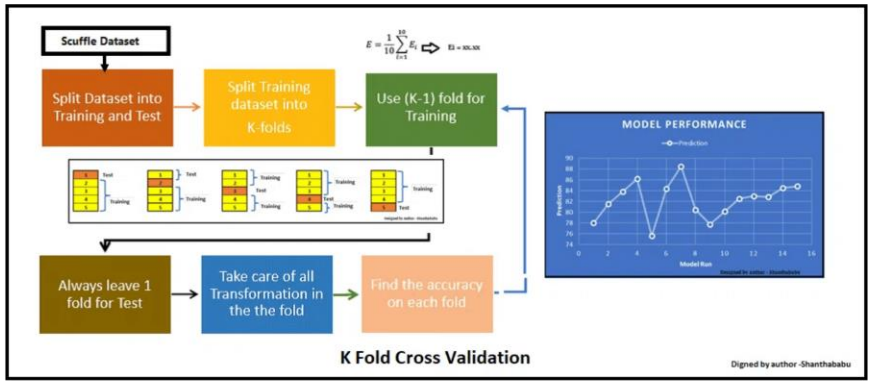
Train/Test Split

Простейший метод: данные разделяются на обучающую (обычно 70-80%) и тестовую выборки. Модель обучается на train, оценивается на test. Быстро, но дает только одну оценку качества.

K-Fold Cross-Validation

Данные делятся на K частей (обычно 5-10). Модель обучается K раз, каждый раз используя K-1 частей для обучения и 1 часть для валидации.

Итоговая оценка — среднее по всем K итерациям. Более надежная оценка качества.



Специализированные методы валидации

Stratified K-Fold

Модификация K-Fold, которая сохраняет пропорции классов (для классификации) или распределение целевой переменной (для регрессии) в каждом фолде. Особенно важна при несбалансированных данных.

Time Series Rolling Window Split

Специальный метод для временных рядов: обучающая выборка всегда предшествует валидационной по времени. Предотвращает утечку информации из будущего в прошлое. Критичен для корректной оценки прогнозных моделей.

Метрики регрессии — количественная оценка качества

MAE

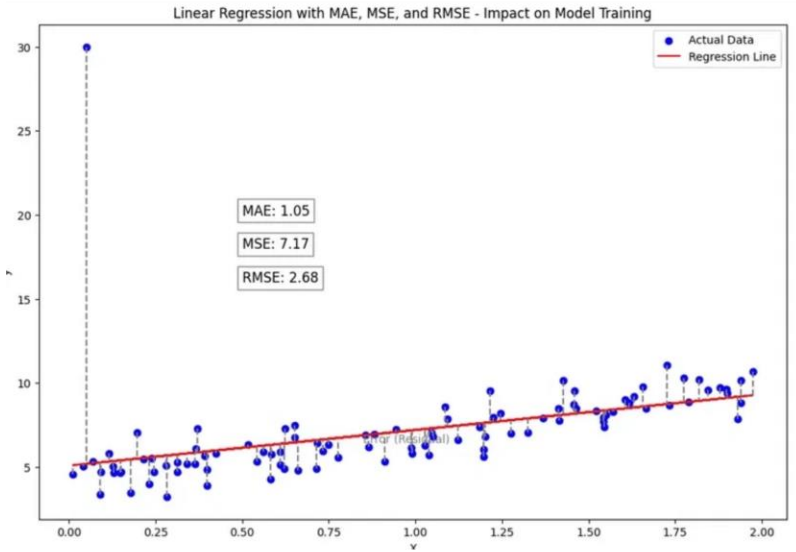
$$MAE = (1/n) \cdot \sum |y - \hat{y}|$$

Средняя абсолютная ошибка. Легко интерпретируется в единицах целевой переменной. Робастна к выбросам. Все ошибки имеют равный вес.

MSE

$$MSE = (1/n) \cdot \sum (y - \hat{y})^2$$

Средняя квадратичная ошибка. Сильно штрафует большие ошибки. Чувствительна к выбросам. Дифференцируема везде, удобна для оптимизации.



RMSE

$$RMSE = \sqrt{MSE}$$

Корень из средней квадратичной ошибки. Возвращает метрику в исходные единицы измерения. Сохраняет чувствительность к выбросам как MSE.

R^2 (коэффициент детерминации)

$$R^2 = 1 - SS_{res}/SS_{tot}$$

Доля дисперсии, объясненная моделью. Значения от 0 до 1 (может быть отрицательным для плохих моделей). Легко интерпретируется как процент объясненной вариации.

Дополнительные метрики и их применение

Специализированные метрики регрессии предназначены для конкретных типов задач и распределений данных. Они учитывают особенности предметной области и требования к интерпретируемости результатов.

01 MAPE (Mean Absolute Percentage Error)

$$MAPE = (100\%/n) \cdot \sum|((y - \hat{y})/y)|$$

Процентная ошибка, легко интерпретируемая для бизнеса. Применяется в прогнозировании продаж и спроса. Проблемы: не определена при y=0, **асимметрична** (переоценка штрафуется сильнее недооценки).

02 SMAPE (Symmetric MAPE)

$$SMAPE = (100\%/n) \cdot \sum(|y - \hat{y}|/(|y| + |\hat{y}|))$$

Симметричная версия MAPE, одинаково штрафует переоценку и недооценку. Ограничена диапазоном 0-100%. Используется в прогнозировании временных рядов, когда важна симметричность ошибок.

03 RMSLE (Root Mean Squared Log Error)

$$RMSLE = \sqrt{((1/n) \cdot \sum(\log(y+1) - \log(\hat{y}+1))^2)}$$

Логарифмическая метрика, штрафует относительные ошибки. Менее чувствительна к выбросам и большим значениям. Применяется **при экспоненциальном росте данных**, прогнозировании цен и популяций.

Заключение

Успешное решение задачи регрессии требует комплексного подхода, сочетающего глубокое понимание данных, правильный выбор алгоритмов и метрик, а также систематическую валидацию результатов.

01

Понимание данных

Исследуйте распределение признаков и целевой переменной, выявляйте выбросы и пропуски, анализируйте корреляции. Качественная предобработка данных часто важнее выбора сложного алгоритма.

02

Итеративный подход

Начинайте с простых моделей (линейная регрессия), постепенно переходя к более сложным (деревья, бустинг). Каждая итерация дает инсайты о данных и направлениях улучшения.

03

Правильная валидация

Используйте кросс-валидацию для надежной оценки качества. Для временных рядов применяйте Time Series Split. Отделяйте тестовую выборку и не используйте её до финальной оценки.

04

Настройка гиперпараметров

Систематически подбирайте гиперпараметры через Grid Search, Random Search или другие. Правильная настройка может повысить качество модели на 10-30%.

05

Выбор метрики

Выбирайте метрику в соответствии с бизнес-задачей: MAE для интерпретируемости, MSE для штрафа больших ошибок, MAPE для процентных оценок. Метрика должна отражать реальную цель проекта.

Продвинутые методы поиска гиперпараметров

03

Bayesian Optimization (Байесовская оптимизация)

Использует вероятностную модель для предсказания перспективных областей пространства гиперпараметров. Эффективен для дорогих функций оценки, так как требует меньше итераций для нахождения оптимума.

Особенности: Интеллектуальный поиск, баланс exploration/exploitation, требует меньше вычислений

04

Hyperband / ASHA (Адаптивные методы)

Адаптивные методы, которые рано останавливают обучение неперспективных конфигураций гиперпараметров. Значительно ускоряют поиск при большом количестве итераций обучения за счет динамического распределения ресурсов.

Особенности: Ранняя остановка, динамическое выделение ресурсов, оптимален для глубоких моделей