Тема 7. Машинное обучение на больших данных

Лекция 7.1. Масштабирование машинного обучения на кластер

Стандартные алгоритмы машинного обучения предполагают, что вся обучающая выборка помещается в оперативную память одного компьютера. При объёме данных в терабайты и выше это невозможно. Требуются либо алгоритмы, работающие с данными порциями (online/incremental learning), либо распределённые реализации, способные параллельно обрабатывать данные на кластере.

— Data Parallelism: обучающие данные разбиваются на части, каждый узел обучает модель на своём фрагменте, градиенты агрегируются (Allreduce, параметрический сервер). Применяется в TensorFlow Distributed, PyTorch DDP, Horovod.

— Model Parallelism: сама модель разбивается по узлам. Применяется для нейронных сетей с сотнями миллиардов параметров, которые не помещаются в память одного GPU.

— Mini-batch SGD: обновление весов на малом случайном подмножестве (батче) — не нужно загружать весь датасет в память. Является основой большинства современных алгоритмов обучения нейросетей.

— Approximate computing: sketch-алгоритмы вычисляют точные или приближённые ответы (с доказанными гарантиями) на основе компактного резюме данных, а не полного набора.

Лекция 7.2. Apache Spark MLlib

Spark MLlib — встроенная библиотека машинного обучения Apache Spark. Работает поверх DataFrame API и интегрируется с Pipeline. Алгоритмы автоматически масштабируются на кластер: разработчик описывает логику (какой алгоритм применить и с какими параметрами), Spark берёт на себя распределение вычислений.

Алгоритмы обучения с учителем

— Линейная регрессия: регуляризация L1 (Lasso, обнуляет незначимые признаки), L2 (Ridge, сглаживает веса), ElasticNet (комбинация). Оптимизация: L-BFGS или SGD.

— Логистическая регрессия: бинарная и мультиклассовая классификация (One-vs-Rest или Multinomial). Быстрая, интерпретируемая, устойчивая — хороший базовый уровень для любой задачи.

— Decision Tree: разбивает пространство признаков бинарными условиями. Полностью интерпретируема. Склонна к переобучению без ограничения глубины.

— Random Forest: ансамбль деревьев, каждое обучается на случайной подвыборке данных и признаков (bagging + feature subsampling). Устойчив к переобучению,

хорошо работает без тюнинга.

— Gradient Boosted Trees (GBT): деревья добавляются последовательно, каждое исправляет ошибки предыдущих. Высокая точность, но требует тюнинга. Spark реализует классический GBT; для production часто используют XGBoost или LightGBM поверх Spark.

— Linear SVM (LinearSVC): линейный классификатор с максимизацией разделяющего зазора. Эффективен для высокоразмерных разрежённых данных (тексты).

— Naive Bayes: вероятностный классификатор на основе теоремы Байеса с предположением о независимости признаков. Быстрый, отлично работает на текстах.

Алгоритмы обучения без учителя

— K-Means и Bisecting K-Means: кластеризация.

— Gaussian Mixture Model: вероятностная мягкая кластеризация.

— LDA (Latent Dirichlet Allocation): тематическое моделирование коллекций текстов.

— PCA (Principal Component Analysis): линейное снижение размерности.

— FPGrowth: поиск ассоциативных правил.

— ALS (Alternating Least Squares): матричная факторизация для рекомендательных систем. Реализован в Spark MLlib, масштабируется на миллионы пользователей и товаров.

Лекция 7.3. Оценка качества моделей

Метрики классификации

Точность (Accuracy) — доля верных предсказаний — не информативна при дисбалансе классов. Если 99% транзакций нормальные, классификатор «всё нормально» даст точность 99%, но не обнаружит ни одного мошенничества.

— Precision: из всех объектов, помеченных моделью как положительный класс, доля действительно положительных. Важна, когда ложноположительные дорогостоящи (ложная тревога).

— Recall (Sensitivity): из всех реальных положительных, доля обнаруженных моделью. Важна, когда пропуск положительных дорогостоящ (пропущенная болезнь).

— F1-score: гармоническое среднее Precision и Recall. Баланс между ними.

— AUC-ROC: площадь под кривой TPR от FPR при изменении порога классификации. Не зависит от порога, пригодна для сравнения моделей. 1,0 — идеальная модель, 0,5 — случайное угадывание.

— AUC-PR (Precision-Recall AUC): более информативна при сильном дисбалансе классов.

Метрики регрессии

— MAE (Mean Absolute Error): среднее абсолютных отклонений. Интерпретируема в единицах целевой переменной. Устойчива к выбросам.

— RMSE (Root Mean Squared Error): квадратный корень из MSE. Штрафует большие ошибки сильнее MAE. Чувствительна к выбросам.

— MAPE (Mean Absolute Percentage Error): относительная ошибка в процентах. Позволяет сравнивать модели для рядов с разным масштабом. Не определена при нулевых фактических значениях.

— R-squared: доля объяснённой дисперсии. 1,0 — идеальная модель, 0 — модель не лучше константного предсказания (среднего), <0 — хуже среднего.

Лекция 7.4. Переобучение, регуляризация, подбор гиперпараметров

Переобучение и недообучение

Переобучение (Overfitting): модель выучила шум в обучающих данных. Отличный результат на train, плохой на test. Признак: большой разрыв между метриками на train и validation. Недообучение (Underfitting): модель слишком проста, не улавливает закономерности. Плохой результат на обоих наборах. Решение: более сложная модель или лучшие признаки.

Методы борьбы с переобучением

— Регуляризация L1 (Lasso): добавляет к функции потерь сумму |w|. Обнуляет коэффициенты незначимых признаков — встроенный отбор признаков.

— Регуляризация L2 (Ridge): добавляет сумму w^2. Плавно уменьшает все веса, не обнуляя их.

— ElasticNet: взвешенная комбинация L1 и L2. Работает лучше при многих коллинеарных признаках.

— Ранняя остановка (Early Stopping): прекращать обучение, когда метрика на валидационном наборе перестаёт улучшаться n эпох подряд.

— Dropout: для нейросетей — случайное обнуление активаций нейронов в слое с вероятностью p при обучении.

Подбор гиперпараметров в Spark

— ParamGridBuilder: определяет сетку значений гиперпараметров для перебора.

— CrossValidator: k-fold кросс-валидация для каждой комбинации из сетки. Возвращает модель с лучшими гиперпараметрами.

— TrainValidationSplit: однократное разделение на train/validation (быстрее CrossValidator, менее надёжно).

— Random Search эффективнее Grid Search при большом пространстве гиперпараметров: случайные комбинации покрывают пространство лучше при том же числе экспериментов.

Лекция 7.5. Рекомендательные системы

Рекомендательные системы — одно из наиболее экономически значимых приложений машинного обучения. Netflix: 75% просмотров идёт через рекомендации, система приносит около 1 миллиарда долларов в год за счёт снижения оттока. Amazon: около 35% выручки через «Customers Also Bought». Spotify: ежедневный Discover Weekly playlist использует коллаборативную фильтрацию.

— Content-Based Filtering: рекомендует объекты, похожие на те, что понравились пользователю. Использует признаки объектов (жанр, актёры, теги). Проблема «информационного пузыря»: пользователь видит только то, что уже знает.

— Collaborative Filtering (пользовательский): «пользователи, похожие на вас, также смотрели…». User-User CF: находит похожих пользователей и рекомендует их предпочтения. Item-Item CF: находит похожие объекты (Amazon использует именно этот подход).

— Matrix Factorization (ALS): матрица «пользователь–товар» факторизуется на две матрицы скрытых факторов (latent factors). Каждый пользователь и каждый объект представлены плотным вектором в скрытом пространстве. ALS (Alternating Least Squares) оптимизирует матрицы поочерёдно — хорошо масштабируется на Spark.

— Гибридные системы: комбинируют контентный и коллаборативный подходы, добавляют контекст (время, геолокация, устройство). Используются на Netflix, Spotify, YouTube.

Лекция 7.6. MLOps: производственная эксплуатация ML-моделей

Большинство ML-проектов не доходит до production. MLOps — набор практик для надёжного, воспроизводимого и масштабируемого развёртывания моделей. Аналог DevOps, но для машинного обучения.

Полный жизненный цикл ML-модели

— Постановка задачи: определить целевую метрику, критерий успеха, ограничения.

— Сбор и маркировка данных: источники, объём, качество разметки.

— Exploratory Data Analysis (EDA): понять распределения, связи, аномалии.

— Конструирование признаков (Feature Engineering): создание информативных признаков из сырых данных.

— Обучение и валидация: эксперименты с алгоритмами, тюнинг гиперпараметров.

— Deployment: упаковка в Docker, REST API (FastAPI, Flask), batch scoring или real-time inference.

— Мониторинг и переобучение: отслеживание дрейфа данных и концепции, регулярное обновление модели.

Инструменты MLOps

— MLflow: трекинг экспериментов (метрики, параметры, артефакты), реестр моделей, развёртывание. Интегрирован с Spark и Databricks.

— Feature Store (Feast, Tecton, Databricks): централизованное хранилище вычисленных признаков. Гарантирует, что один и тот же признак вычисляется одинаково при обучении и inference.

— Evidently AI, Arize, WhyLabs: мониторинг дрейфа данных и деградации модели в production.

— Kubeflow Pipelines: ML-пайплайны на Kubernetes для воспроизводимого запуска экспериментов.