Дизайн систем машинного обучения

5. Выбор и обучение модели

План курса

1) ~~Практическое применение машинного обучения~~
2) ~~Основы проектирования ML-систем~~
3) ~~Обучающие данные~~
4) ~~Подготовка и отбор признаков~~
5) **Выбор модели, разработка и обучение модели — Вы находитесь здесь**
6) Оценка качества модели
7) Развертывание систем
8) Диагностика ошибок и отказов ML-систем
9) Мониторинг и обучение на потоковых данных
10) Жизненный цикл модели
11) Отслеживание экспериментов и версионирование моделей
12) Сложные модели: временные ряды, модели над графами
13) Непредвзятость, безопасность, управление моделями
14) ML инфраструктура и платформы
15) Интеграция ML-систем в бизнес-процессы

Найдите тупое решение

- Найдите несколько крутых статей →
- Обычно они сравнивают свое хорошее сложное решение с каким-нибудь тупым и простым
- Найдите это тупое решение и реализуйте его

(c) Не помню, но точно не я придумал

Квантовые алгоритмы vs SVM + CRF

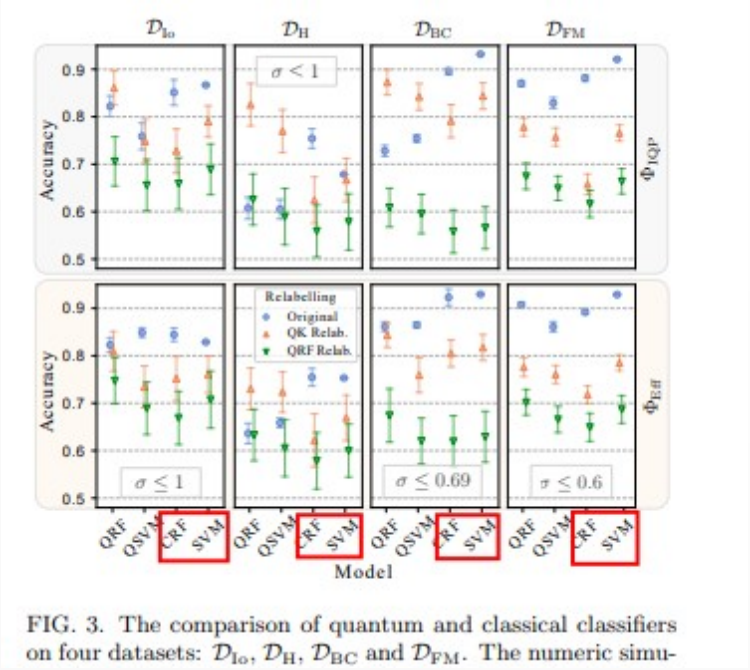


FIG. 3. The comparison of quantum and classical classifiers on four datasets: $\mathcal{D}_{\text{lo}}$, $\mathcal{D}_{\text{H}}$, $\mathcal{D}_{\text{BC}}$ and $\mathcal{D}_{\text{FM}}$. The numeric simu-

https://arxiv.org/abs/2210.02355

Осторожнее с SOTA

- Может быть медленнее
  - Больше памяти
  - Дольше считается
- Может не быть готовых библиотек
- Может не работать на больших наборах данных
- Может не работать на маленьких наборах данных
- Может вообще не работать

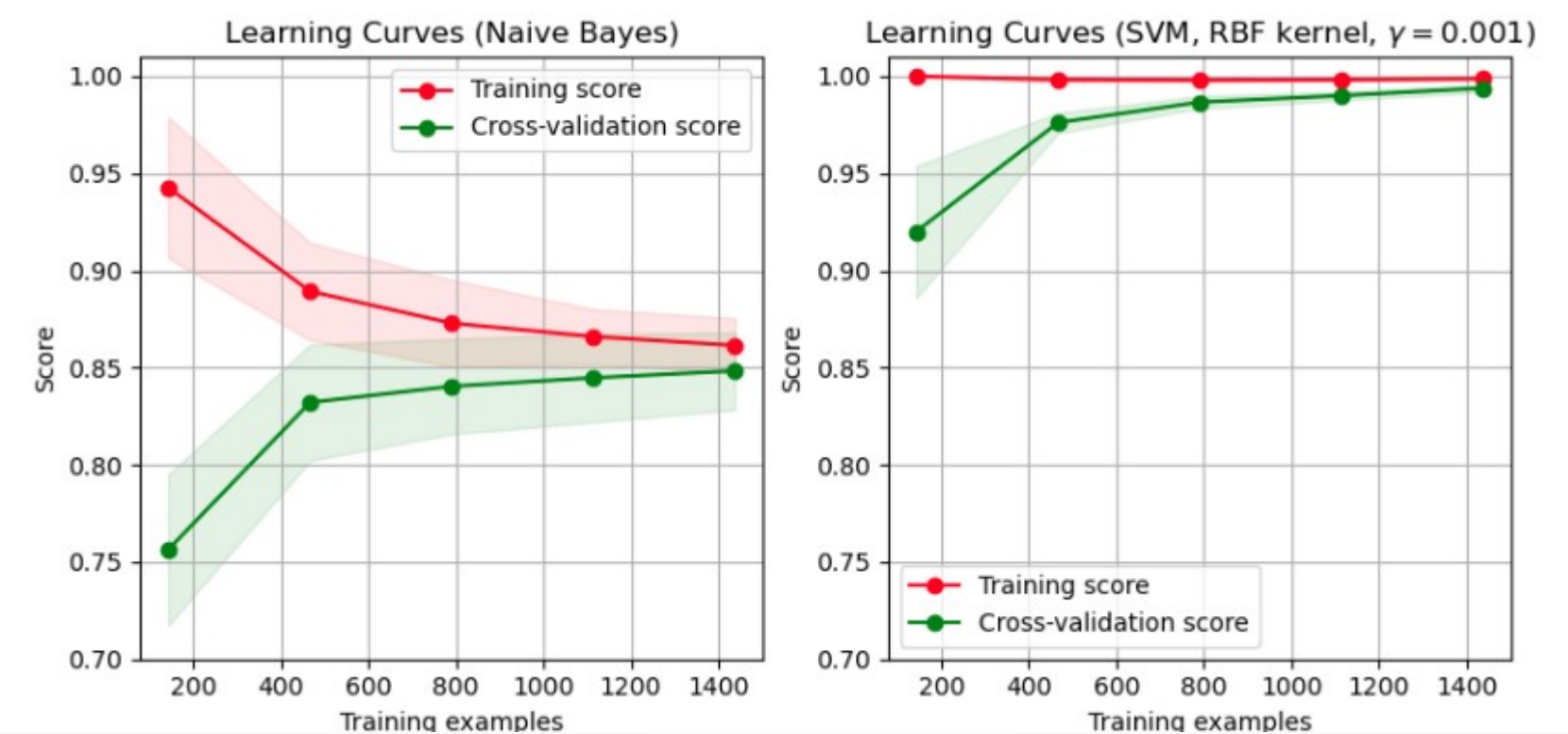
Начинайте с простой модели

- Простая в алгоритмическом смысле, не в смысле усилий
- Проще в развертывании
  - делает проще всю систему
- Проще в отладке
  - позволяет найти ошибки на входе и выходе
- Все равно нужен baseline
  - чтобы было с чем сравнивать сложную модель

Ошибки в сравнении моделей

- Модели трудно сравнивать
- Если какая-то модель плохо работает,
  может быть, мы просто не умеем ее готовить
- Экспериментируйте с гиперпараметрами
- Отслеживайте эксперименты
- Сомневайтесь в результатах сравнения

Смотрим вперед — Learning Curve



https://scikit-learn.org/stable/auto_examples/model_selection/plot_learning_curve.html

Компромиссы

- Precision vs Recall
- Качество vs сложность развертывания
- Качество vs вычислительные ресурсы
- Качество vs задержка инференса
- Качество vs эксплуатационные издержки
- Качество vs устойчивость к атакам
- Качество vs устойчивость к шуму

Предположения модели

- Предположения линейной регрессии →
- Сверточные сети — локальность, разделяемые параметры →
- Марковские модели — процесс без памяти
- Кластеризация — предположения о метрике
- Алгоритмы на деревьях — делают меньше всего предположений
- У каждой модели есть предположения и ограничения →

Итого, выбор модели:

- Избегайте ловушки SOTA
- Начинайте с простейшей модели
- Делайте поправку на человеческие ошибки
- Как качество модели изменится в будущем
- Компромисс
- Изучите ограничения и предположения модели

Ансамбли

- Стекинг Stacking →
- Терминологическая путаница →
- Готовые ансамбли — градиентный бустинг и случайный лес
- StackingClassifier StackingRegressor →
- Почти всегда:
  - не имеет смысла ансамблировать слабые алгоритмы
  - не имеет смысла ансамблировать сильно скоррелированные алгоритмы
  - исключение — weak-supervision: Snorkel at al.

Например, независимые

- Алгоритмы A, B, C — вероятность ошибки 0,3
- Ошибки независимы, $Perr(A|B) = Perr(A)$ и т.д.
- Ансамблируем большинством VotingClassifier
- Вероятность одновременной ошибки 3-х = $0,3^3 = 0,027$
- Вероятность ошибки 2-х = $0,3^2 * 0,7 * 3 = 0,189$
- Итого вероятность ошибки 0,189 + 0,027 = 0,216
- Есть выигрыш от ансамблирования

Например, зависимые

- Алгоритмы A, B, C — вероятность ошибки 0,3
- Ошибки полностью синхронны
- Ансамблируем большинством VotingClassifier
- Вероятность ошибки = 0,3
- Нет выигрыша от ансамблирования

Например, независимые слабые

- Алгоритмы A, B, C — вероятность ошибки 0,3 0,4 0,45
- Ошибки независимы, Perr(A|B) = Perr(A) и т.д.
- Ансамблируем большинством VotingClassifier
- Итого вероятность ошибки 0.327
- Хуже чем алгоритм A в одиночку

Ансамбль из слабаков

- Ансамбли на деревьях, например «решающие пни»
  - Боремся с корреляцией ошибки (подвыборки строк, столбцов)
  - Большой ансамбль
- Weak Supervision
  - Три класса — да/нет/не знаю, в да/нет области сильный
- Сильный со слабым — с разной структурой ошибки
- Например, градиентый бустинг после регрессии

Отслеживание экспериментов

- Нужно проверить много гипотез
- Меняются модели
- Меняются признаки
- Меняются данные
- При переобучении у моделей получается разное качество

Инструменты

- W & B →
- MLFLow + Optuna + Hydra →
- DVC →
- ClearML →
- TensorBoard →
- CometML →

см https://mymlops.com/

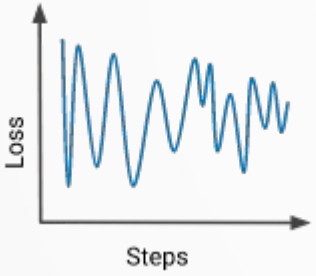
Распределенное обучение

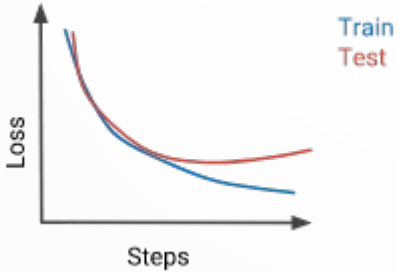
- Accelerate →
  - Работа с большими моделями
- DeepSpeed →
  - Распределенное обучение моделей
- Ray →
- Dask-ML →
- LZY →

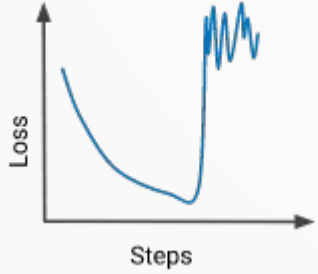
Подбор гиперпараметров и AutoML

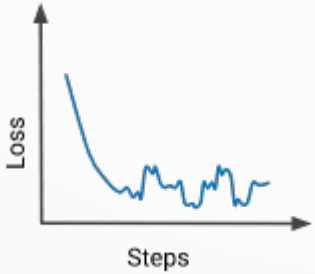
- Optuna →
- Hyperopt →
- TPOT ->
- Auto-PyTorch →
- Auto-sklearn →
- AutoML Book →

Loss Curves







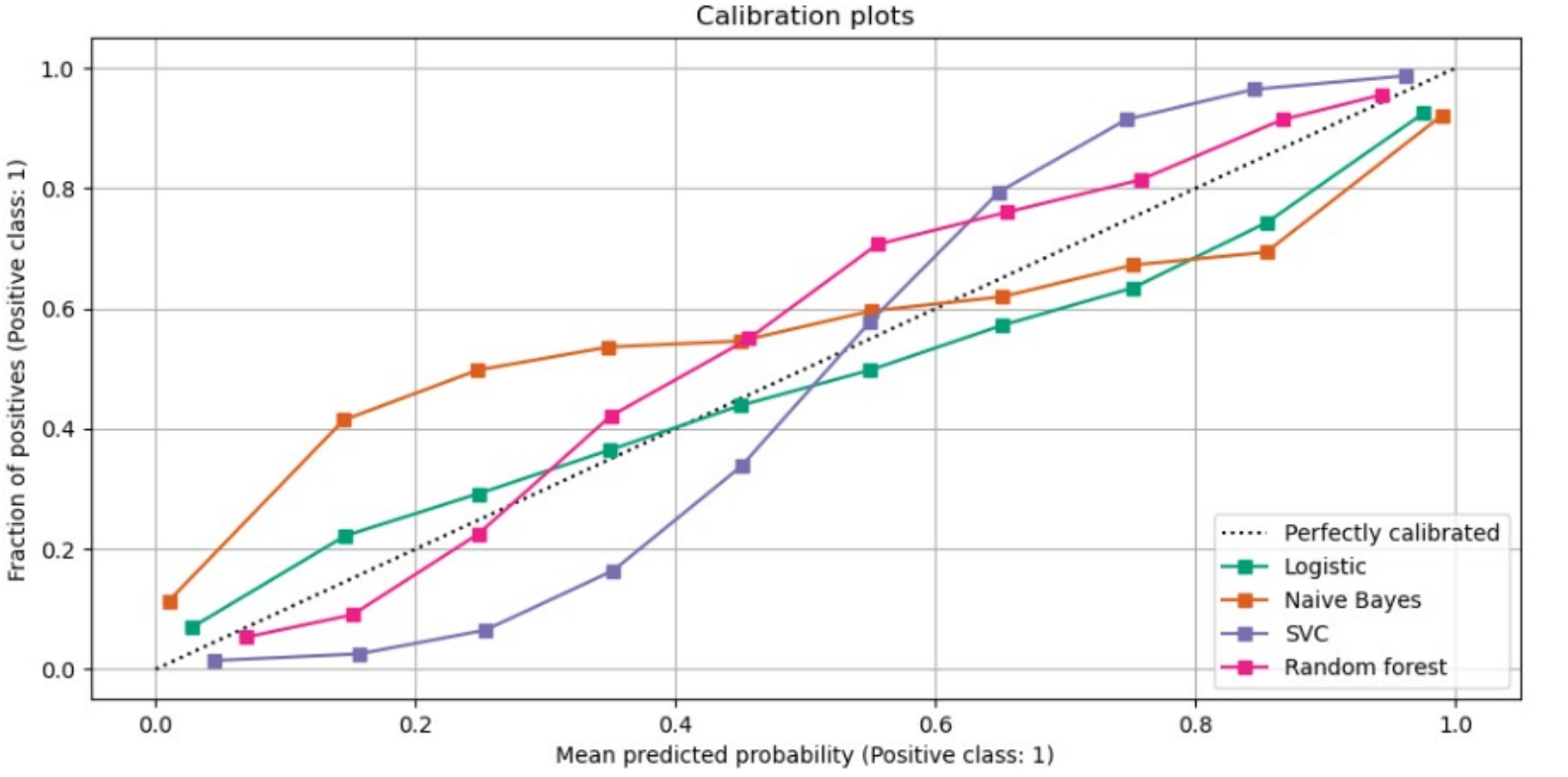


https://developers.google.com/machine-learning/testing-debugging/metrics/interpretic

Если не работает

- Переобучиться на микродатасете (нулевой Loss)
- Проверить порядок размерностей PyTorch vs TF
- Визуализировать градиенты →
- Уменьшить/увеличить на порядок скорость обучения
- Обучиться на небольшой части данных (~5%)
- Дать модели поработать подольше

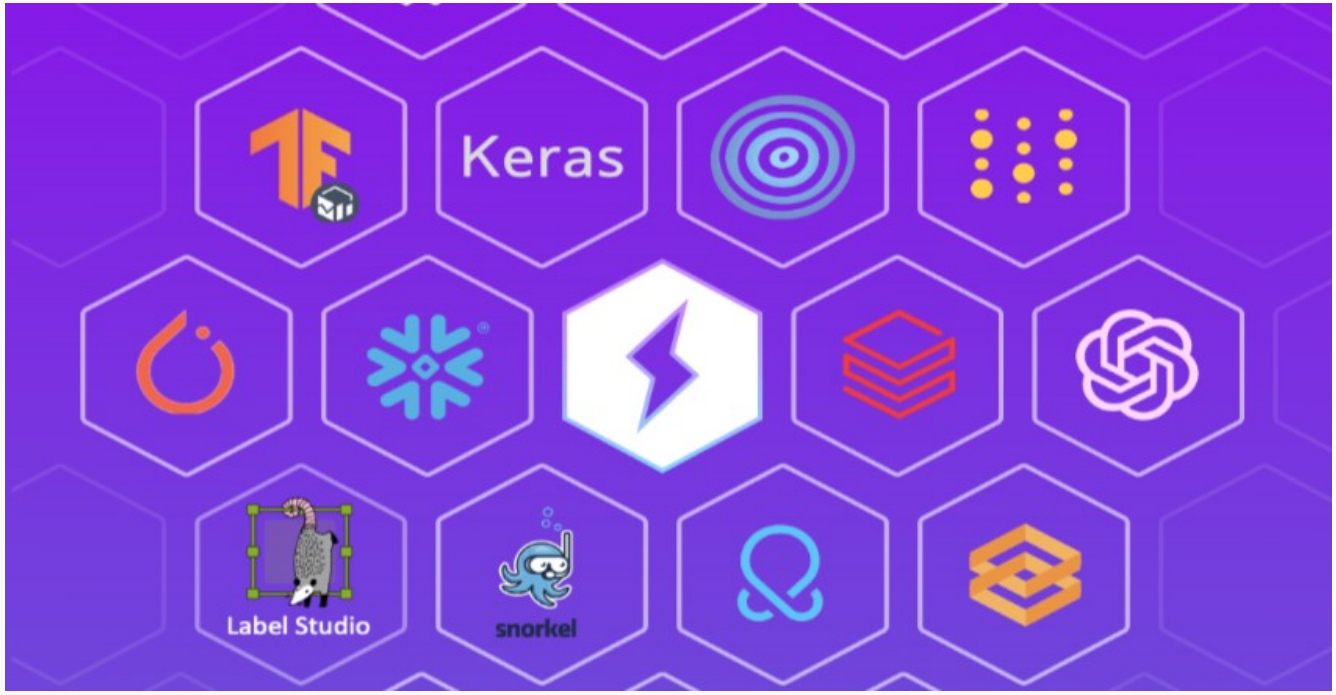
Калибровка модели



Калибровка

- Scikit-learn
  - CalibratedClassifierCV
- Модели на деревьях
  - Calibration Trees
- Нейронные сети
  - Softmax с температурой

А еще попробуйте Lightning



https://lightning.ai/

Дополнительные материалы

- A Recipe for Training Neural Networks by Andrej Karpathy →
- Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning by Sebastian Raschka →

Все будет в телеграм-канале