

## Лекция 3. Обучение и оценка качества

# Цель

Хотим, чтобы модель работала хорошо

Крошка сын

к отцу пришел,

и спросила кроха:

- Что такое

хорошо

и что такое

плохо? -

У меня

секретов нет, -

слушайте, детишки, -

папы этого

ответ

помещаю

в книжке.

# Что такое хорошо?

- ❶ Как понять «хорошесть»
  - ▶ На этапе обучения — функция потерь
  - ▶ На этапе тестирования — метрики качества
- ❷ Как добиться того, чтобы модель работала хорошо
- ❸ Как доказать, что модель работает хорошо?

# Функция потерь и метрики качества

- **Функция потерь** возникает в тот момент, когда мы сводим задачу построения модели к задаче оптимизации. Обычно требуется, чтобы она обладала хорошими свойствами (например, дифференцируемостью).
- **Метрика качества** — внешний, объективный критерий качества, обычно зависящий не от параметров модели, а только от предсказанных меток.

# На что можем влиять?

- Данные
- Признаки
- Модель
- Метод обучения

# Данные

Зафиксируем признаки:

- Достаточно ли данных?
- Как их собирать?
- Как и кто будет размечать?

# Задача

Собрали данные, определились с признаками, функцией потерь, метриками качества и выбрали модель. Надо её обучить и понять, хорошо ли модель будет работать на практике

# Способы оценки качества

## ① Black-box методы

- ▶ Online
- ▶ Offline

## ② Glass-box методы

- ▶ VC-оценки
- ▶ PAC Bayes bounds
- ▶ ...

- Наблюдение
- Эксперимент
  - + В условиях эксплуатации
  - + В положительные результаты эксперимента обычно верят
  - + Легко хвастаться результатом
    - Не всегда «боевые условия» доступны
    - Вряд ли цель эксплуатации
    - Можно навредить пользователям
    - Люди не любят быть объектом экспериментирования
    - Качество эксперимента может быть сильно хуже продакшена
    - Количество одновременных экспериментов ограничено

# Offline

- + Нельзя навредить пользователям
- + Обычно можно проводить сильно больше экспериментов
  - Обычно нужны данные (примеры)
  - Сложно «хвастаться» результатом

**Кросс-валидация** — процедура эмпирического оценивания обобщающей способности алгоритмов. С помощью кросс-валидации эмулируется наличие тестовой выборки, которая не участвует в обучении, но для которой известны правильные ответы.

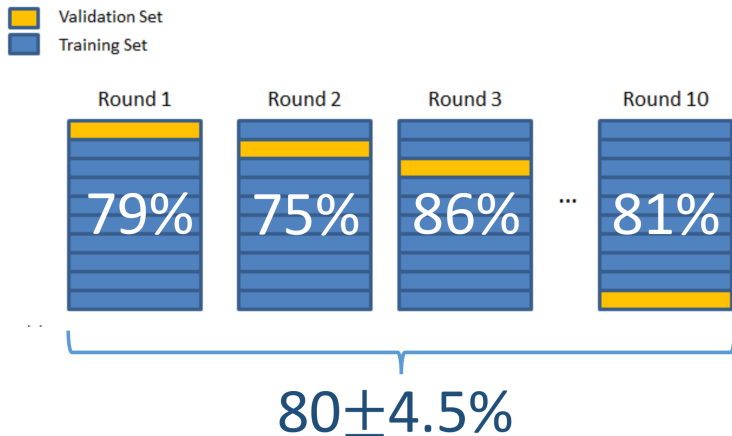
# Способы валидации

- Валидация на отложенных данных (Hold-Out)
- Полная кросс-валидация
- Кросс-валидация по отдельным объектам (Leave-One-Out)
- k-fold кросс-валидация
- со стратификацией и без
- ...

# К-фолд кросс-валидация



# Учёт разброса и распределения при кросс-валидации



# Стабильность решения

Рассматриваем, как меняются настраиваемые параметры модели (зависит от типа модели):

- Стабильные компоненты заслуживают веры
- Если все нестабильно — плохо

# Анализ важности признаков

## На одном фолде:

0.211268 Номер  
0.147105 Ширина  
0.128326 Вес  
0.0954617 Параметр 1  
0.0688576 Высота  
0.057903 Параметр 2  
0.0438185 Параметр 3  
...

## На другом:

0.285714 Номер  
0.163265 Параметр 1  
0.122449 Высота  
0.102041 Параметр 4  
0.0816327 Параметр 5  
0.0816327 Вес  
0.0612245 Параметр 2  
...

# Анализ зависимости от признаков

Если зависимость от каких-то признаков должна иметь известный вид, то можно проверить, что модель её находит правильно.

# Опасности

- Разные характеристики обучающей и рабочей выборок
- Переобучение на валидации
- Несбалансированные классы

# Сложность модели

Какая бывает информация в параметрах:

- про генеральную совокупность
- про выборку
- про random seed

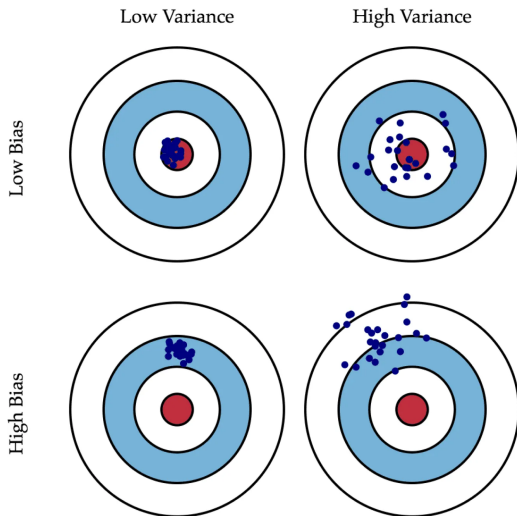
# Переобучение и недообучение

- **Переобучение**, переподгонка (overtraining, overfitting) — нежелательное явление, возникающее при решении задач обучения по прецедентам, когда вероятность ошибки обученного алгоритма на объектах тестовой выборки оказывается существенно выше, чем средняя ошибка на обучающей выборке.
- **Недообучение** (underfitting) — нежелательное явление, возникающее при решении задач обучения по прецедентам, когда алгоритм обучения не обеспечивает достаточно малой величины средней ошибки на обучающей выборке.

# Как можно переобучиться

- Линейные модели — степень полинома
- Деревья решений — глубина дерева
- Нейронные сети — ширина и глубина
- SVM — kernel trick
- ...

# Bias-variance tradeoff

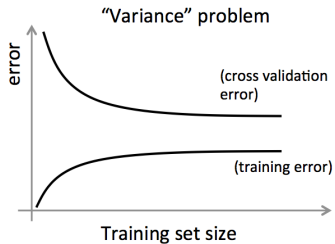
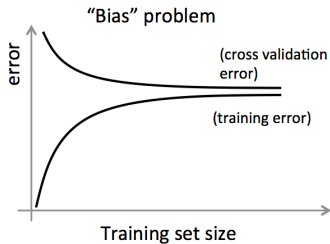


# Bias-variance tradeoff

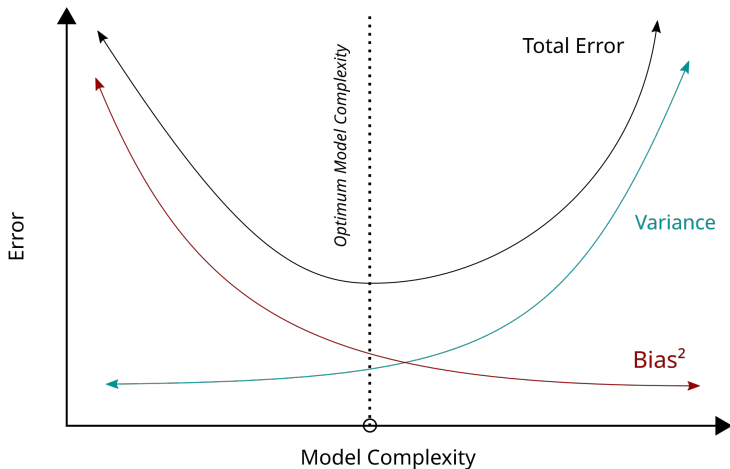
$$\mathbb{E}_X \left[ (y(x) - \hat{f}(x))^2 \right] = \underbrace{\sigma_\varepsilon^2}_{\text{Неустранимый шум}} + \underbrace{\left( \text{Bias}[\hat{f}(x)] \right)^2}_{\text{Смещение}} + \underbrace{\text{Var}[\hat{f}(x)]}_{\text{Разброс}}$$

- $\text{Bias} = y(x) - \mathbb{E}_X[\hat{f}(x)]$  — ошибка из-за упрощающих предположений модели.
- $\text{Var} = \mathbb{E}_X \left[ \left( \hat{f}(x) - \mathbb{E}_X[\hat{f}(x)] \right)^2 \right]$  — ошибка чувствительности к малым отклонениям в выборке.
- $\sigma_\varepsilon^2$  — неустранимый шум в данных.

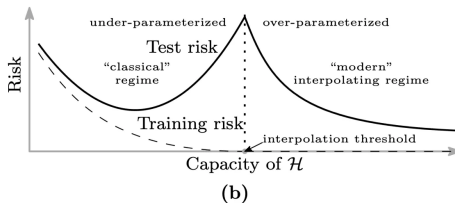
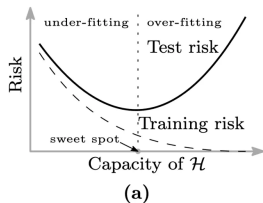
# Как понять, где находимся?



# Зависимость переобучения от сложности модели



# Overparametrization



# Кто виноват и что делать?

- Увеличение числа примеров для обучения исправляет high variance
- Меньшее число факторов исправляет high variance
- Уменьшение сложности модели исправляет high variance
- Увеличение числа факторов исправляет high bias
- Увеличение сложности модели исправляет high bias

# Немного математики

Хотим

$$R_D[f] = \mathbb{E}_{(x,y) \sim D} r(y, f(x)) \rightarrow \min_{f \in \mathcal{F}}. \quad (1)$$

Можем

$$\hat{R}_S[f] = \frac{1}{N} \sum_{k=1}^N r(y_k, f(x_k)) \rightarrow \min_{f \in \mathcal{F}}, \quad (2)$$

где

- $f$  — модель;
- $\mathcal{F}$  — класс моделей;
- $D$  — распределение данных ( $x$  — вход,  $y$  — метка);
- $R$  — функция риска;
- $S$  — обучающая выборка.

## Немного математики

Пусть  $\hat{f}_S$  — решение задачи (2). Что мы можем сказать о  $R_D[\hat{f}_S]$ ?  
Посчитаем  $\hat{R}_{S'}[\hat{f}_S]$  на тестовой выборке  $S'$ . Тогда

$$R_D[\hat{f}_S] \leq \hat{R}_{S'}[\hat{f}_S] + \sqrt{\frac{\ln \frac{1}{\delta}}{2N'}}$$