Автоматизированные методы машинного обучения

Алексей Зайцев
Руководитель
лаборатории
Сколтех

Skoltech

Про что мы сегодня поговорим

1. Этапы анализа данных. На каких из них может помочь AutoML

2. Существующие решения для автомл: Optuna, AutoML by Sber, AutoGluon by Amazon

3. Простой AutoML, как использовать из коробки. Grid search, Optuna

4. Правильная валидация моделей

Skoltech

Этапы анализа данных

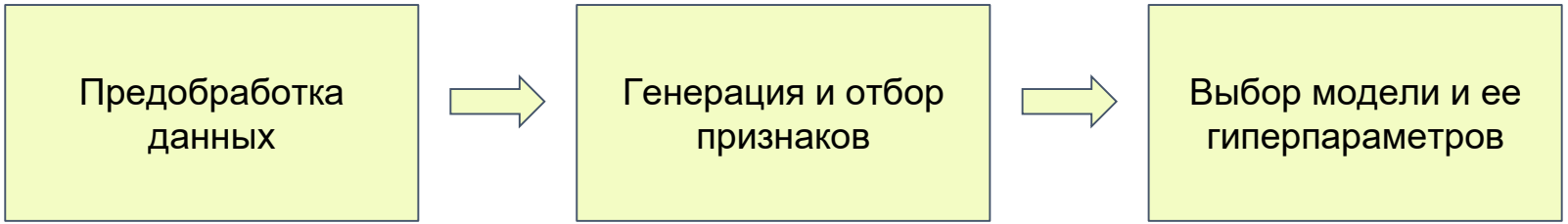
Постановка задачи

Создать как можно более качественную модель машинного обучения в рамках ограничений.

Качество: в реальной жизни

Ограничения: время работы, срок создания модели, класс моделей, доступные данные

Конфигурация решения для автоматизированного машинного обучения AutoML



- очистка данных
- детектирование выбросов
- заполнение пропусков

- генерация признаков
- представление категориальных признаков
- отбор признаков

- валидация моделей
- Байесовская оптимизация
- сравнение моделей

Skoltech

Конфигурация решения для автоматизированного машинного обучения AutoML

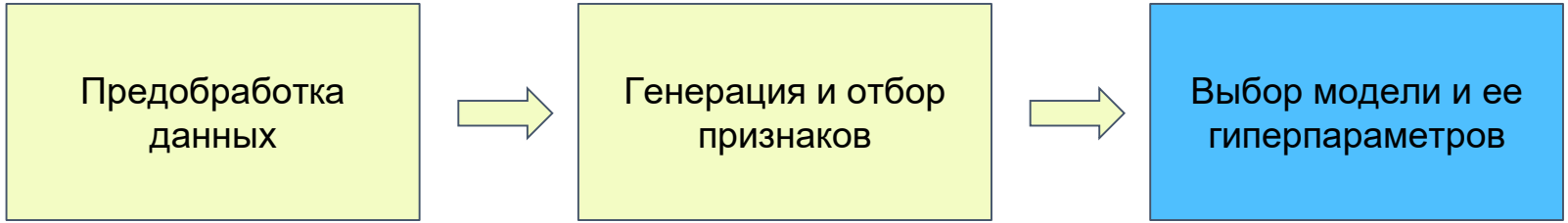
- очистка данных
- детектирование выбросов
- заполнение пропусков

- генерация признаков
- представление категориальных признаков
- отбор признаков

- валидация моделей
- Байесовская оптимизация
- сравнение моделей

На всех этапах применимы подходы автоматизированного машинного обучения: AutoML, AutoGluon

Конфигурация решения для автоматизированного машинного обучения AutoML



- очистка данных
- детектирование выбросов
- заполнение пропусков

- генерация признаков
- представление категориальных признаков
- отбор признаков

- валидация моделей
- Байесовская оптимизация
- сравнение моделей

На этапе выбора модели могут помочь подходы для оптимизации гиперпараметров: Optuna, Hyperopt

Существующие решения для AutoML

Существующие решения общего назначения

Библиотека	Разработчик	Ссылка	Функциональность	Ориентация на банковские приложения	Автоматизация всех этапов
AutoGluon	Amazon	https://github.com/awslabs/autogluon	Работа с разными данными	нет	нет
LightAutoML	Sberbank	https://github.com/sberbank-ai-lab/LightAutoML	Работа с табличными данными	да	нет
NNI	Microsoft	https://github.com/microsoft/nni	Работа с разными данными	нет	нет
H20	H20	https://github.com/h2oai/h2o-3	Все	нет	да
Auto sklearn	Auto sklearn	https://github.com/automl/auto-sklearn	Работа с табличными данными	нет	нет

Существующие решения общего назначения

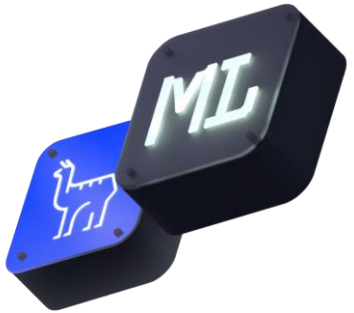
Библиотека	Разработчик	Ссылка	Функциональность
Optuna	Optuna	https://github.com/optuna/optuna	Оптимизация гиперпараметров
Hyperopt	Hyperopt	https://github.com/hyperopt/hyperopt	Оптимизация гиперпараметров
FeatureTools	Alteryx	https://github.com/alteryx/featuretools	Работа с признаками

Основные библиотеки для работы

Выберем джентельменский набор:

- Optuna для оптимизации гиперпараметров

- AutoML by Sber для оптимизации всего



- AutoGluon by Amazon для оптимизации всего

Optuna

Основные свойства:

- Используются для оптимизации гиперпараметров
- Легковесная и понятная в использовании
- Способна к параллелизации
- Доступна визуализация как dashboard
- Эффективные алгоритмы поиска, способна к ранней остановке

Способна работать со следующими модулями:

- Catboost, LightGBM, XGBoost
- Pytorch, Keras, TensorFlow
- Chainer, MXNet
- FastAI V1, V2
- AllenNLP
- Catalyst

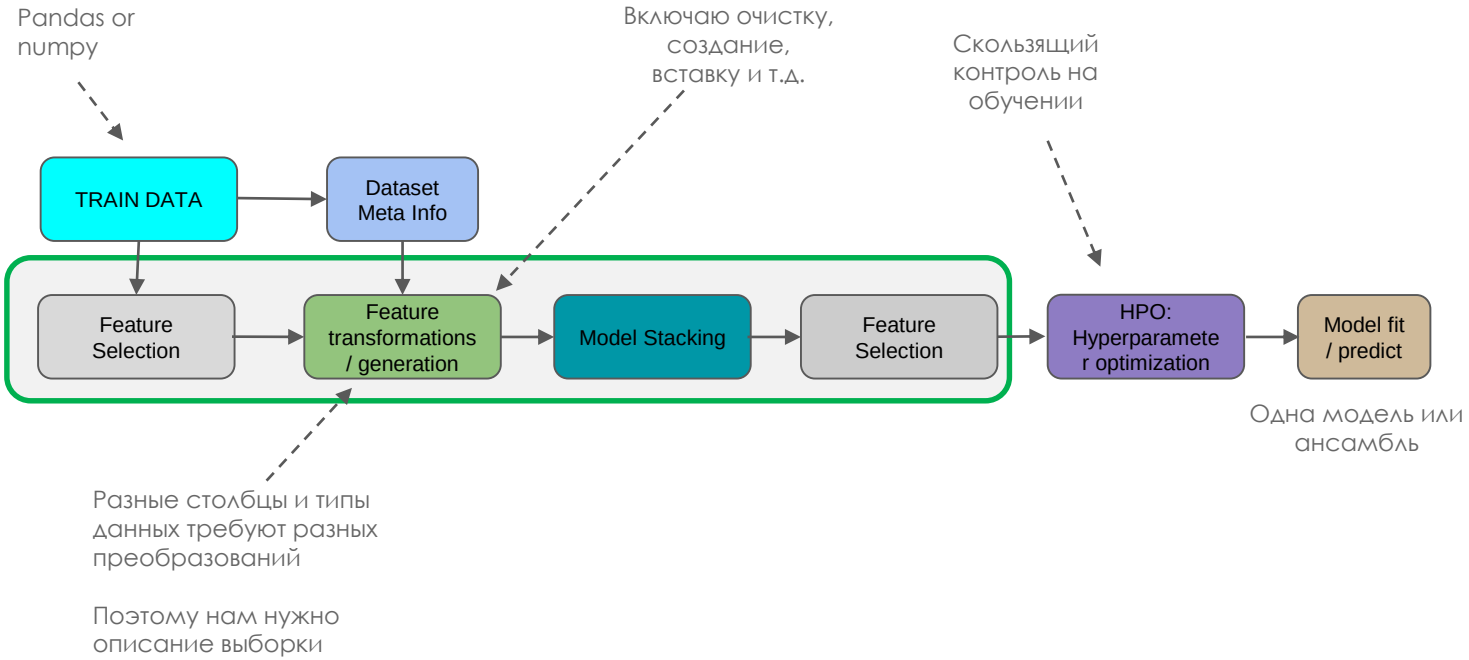
OPTUNA

Optuna - доступные методы оптимизации

- Grid Search
- Random Search
- Tree-structured Parzen Estimator algorithm
- CMA-ES based algorithm
- Algorithm to enable partial fixed parameters
- Nondominated Sorting Genetic Algorithm II
- A Quasi Monte Carlo sampling algorithm

OPTUNA

Схема пайплайна для обработки данных



Skoltech

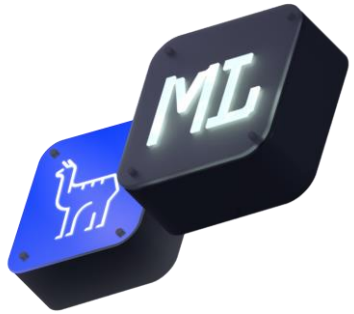
AutoGluon by Amazon

Основные свойства:

- Применима для табличных данных, изображений и текстовых данных
- Для запуска необходимо несколько строк кода
- Встроены подходы для оптимизации гиперпараметров (HPO)
- Кастомизируема

LightAutoML by Sber

Совокупность инструментов для автоматизированного решения задач построения эффективных моделей на основе данных.



AutoML позволяет быстро получать решение высокого качества, учитывает особенности входных данных.

Базовые компоненты библиотеки AutoML:

- Подбор гиперпараметров
- Кодирование признаков
- Стекинг моделей
- Аналитика входных данных

Skoltech

LightAutoML by Sber - реализованные блоки

Model adapters a wrapper for various models with unified: • fit • predict • save • load • get_params • set_params	Data transformation a wrapper for data processing methods with unified: • fit • transform • k-fold split during training	Data Meta data transformations history logging, utils for data slicing and keeping feature types labeled	Main Module Includes right now • fit • predict with internal data processing and feature generation • cross validation	Feature selection • permutation selection • shap tree selector based on LGBM
Stacking adapter Stacking as a part of feature generation • one model stack • multi model stacker • hyperparameters for stackers • different stacking strategies	Cross validation with stratification by a number of groups for cross validation Splits data into k-folds and keeps distribution in each fold for specified columns roughly the same. Not deterministic but works pretty well.	HPO - hyperparameters optimization • HPO for individual models • logging • HPO history for one model	SAVING • saving separate models	

Skoltech

Примеры использования из коробки

GridSearch

Идея: использовать полный перебор возможных параметров, выбрать параметры лучшего результата (например, лучшая валидация)

Плюсы и минусы подхода:

+ Перебор по всем возможным параметрам, решение будет гарантировано лучшее на рассматриваемом множестве
+ Простой в реализации подход
- Время работы растет экспоненциально с числом новых параметров
- В каждом случае нужно ждать конца обучения модели

```
>>> from sklearn import svm, datasets
>>> from sklearn.model_selection import GridSearchCV
>>> iris = datasets.load_iris()
>>> parameters = {'kernel':('linear', 'rbf'), 'C':[1, 10]}
>>> svc = svm.SVC()
>>> clf = GridSearchCV(svc, parameters)
>>> clf.fit(iris.data, iris.target)
GridSearchCV(estimator=SVC(),
             param_grid={'C': [1, 10], 'kernel': ('linear', 'rbf')})
>>> sorted(clf.cv_results_.keys())
['mean_fit_time', 'mean_score_time', 'mean_test_score',...
 'param_C', 'param_kernel', 'params',...
 'rank_test_score', 'split0_test_score',...
 'split2_test_score', ...
 'std_fit_time', 'std_score_time', 'std_test_score']
```

Optuna

Идея: по умолчанию используется Tree-structured Parzen Estimator algorithm - подход, основанный на гауссовских процессах.

Плюсы и минусы подхода:

+ Для получения оптимальных параметров не надо перебирать всевозможные значения
+ За малое количество шагов оптимизации ищет решение, близкое к оптимальному
- Нужно полностью ждать обучения модели (однако с помощью инструмента pruner можно это нивелировать)
- При сложном пространстве параметров не гарантирует нахождение оптимальных

Optuna - разберем пример

Здесь присутствуют три основных сущности подхода:

- objective - целевая функция, которую максимизируем (минимизируем)
- trial - объект, содержит в себе информацию об оптимизруемых гиперпараметрах, также обозначает шаг оптимизатора
- study - объект обучения

```
import optuna

def objective(trial):
    x = trial.suggest_float('x', -10, 10)
    return (x - 2) ** 2

study = optuna.create_study()
study.optimize(objective, n_trials=100)

study.best_params  # E.g. {'x': 2.002108042}
```

Skoltech

AutoGluon

Пример работы AutoGluon на табличных данных

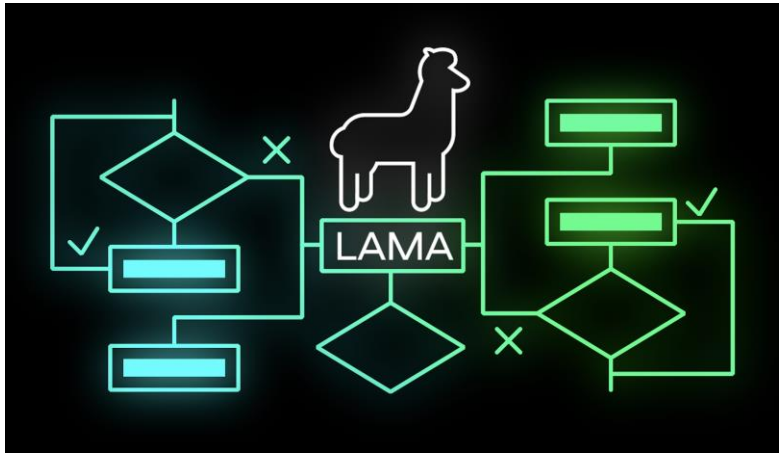
```
from autogluon.tabular import TabularDataset, TabularPredictor

data_root = 'https://autogluon.s3.amazonaws.com/datasets/Inc/'
train_data = TabularDataset(data_root + 'train.csv')
test_data = TabularDataset(data_root + 'test.csv')

predictor = TabularPredictor(label='class').fit(train_data=train_data)
predictions = predictor.predict(test_data)
```

AutoML

- Новые блоки легко добавлять с помощью общих классов адаптеров и файла конфигурации
- Порядок блоков и выполнения операций задается пользователем и под каждую задачу можно быстро собрать свой уникальный пайплайн.

Пример работы AutoML

```
aml_light = AMLib(
        features_processing=['MEstimateEncoder','LeaveOneOutEncoder','FeSelectorPer'],

        models_names=['LGBM'],
        problem_type="classification",
        default_fe_processing_sample = 1,
        hpo_n_trials=25,
        cv_groups = ['X5','X6'],
        metric="roc_auc",
        user_params = {'crossval': {'n_splits': 5,},

                       'FeSelectorPer':{'num_buckets':7,'choose_k_first':3},

                       "cat_encoders":{"n_splits_train": 3, "n_splits_bagging": 30},},
        dense_transform="RankGauss",)

aml_light.fit(train,
              target_col='TARGET',
              column_types=column_types)

preds = aml_light.predict(val)
print('LGBM',aml_light.metric(val['TARGET'], preds['LGBM'][:,1]))
```



Skoltech

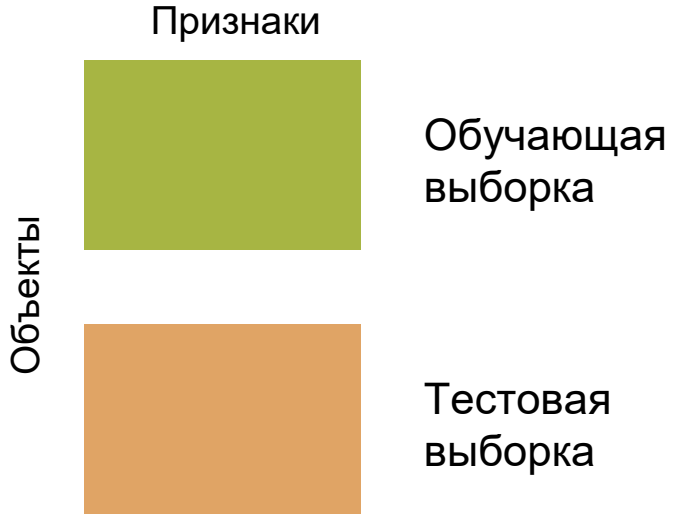
Как валидировать модели?

Тестирование моделей машинного обучения

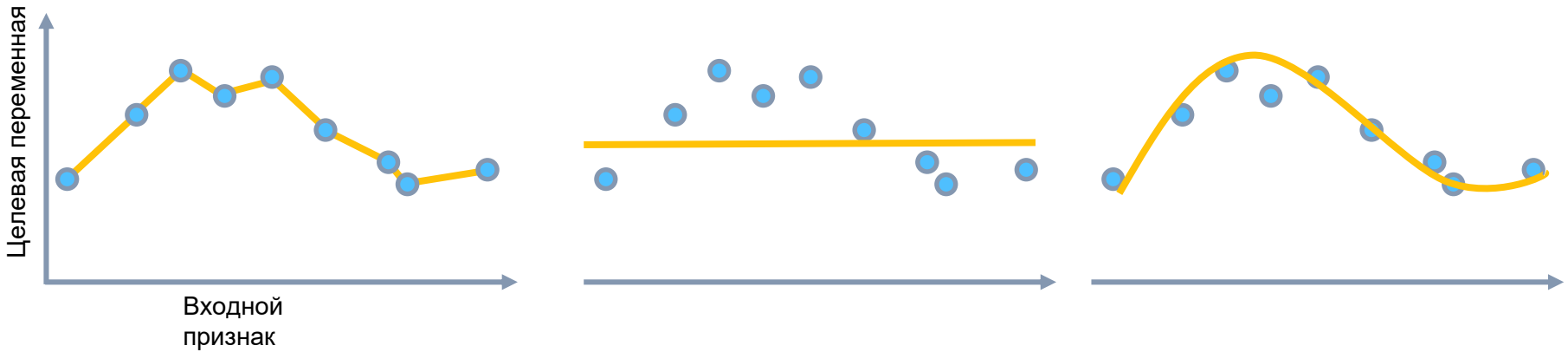
Идеальные случай: есть независимая тестовая выборка, которая похожа на то, как все будет работать в действительности

Решение: обучаем модель на обучающей выборке, тестируем на тестовой (или другое название – валидационной)

Пример задачи: обучение моделей распознавания изображений на ImageNet



Какие типы поведения бывают?



Переобучение

Модель подстроилась под обучающие данные, но будет плохо работать на тестовой выборке

Недообучение

Модель недостаточно подстроилась под обучающие данные, на тестовой выборке тоже все будет плохо

Нормальная модель

Модель ошибается, но не слишком сильно

- точки обучающей выборки

- построенная модель

Оценка качества моделей машинного обучения

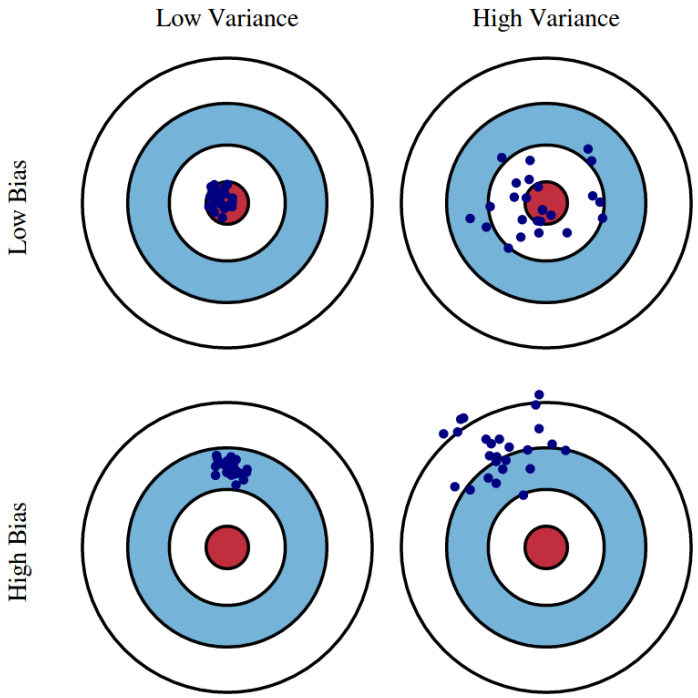
Ошибка на обучающей выборке	Ошибка на тестовой выборке	Анализ
Низкая	Низкая	Все ОК
Низкая	Высокая	Переобучение
Высокая	Высокая	Недообучение
Высокая	Низкая	???

Skoltech

Выбор между смещением и разбросом

Смещение. Если возьмем слишком мало данных для обучения – то качество модели, обученной на подвыборке будет меньше, чем качество модели, обученной на всех доступных данных

Разброс. Мы усредняем ошибку по тестовым данным. Если их мало – то оценка ошибки будет не очень точной.

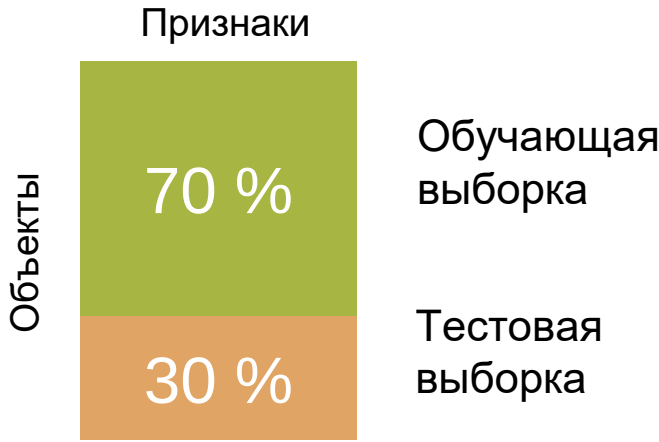


Выделение тестовой выборки

Обычный случай: есть одна выборка данных, по которой нужно оценивать качество модели

Решение: положить в тестовую выборку то, что больше всего похоже на использование модели в реальности

Пример: есть данные за 2017-2020, для обучения используем 2017-2019, для теста - 2020

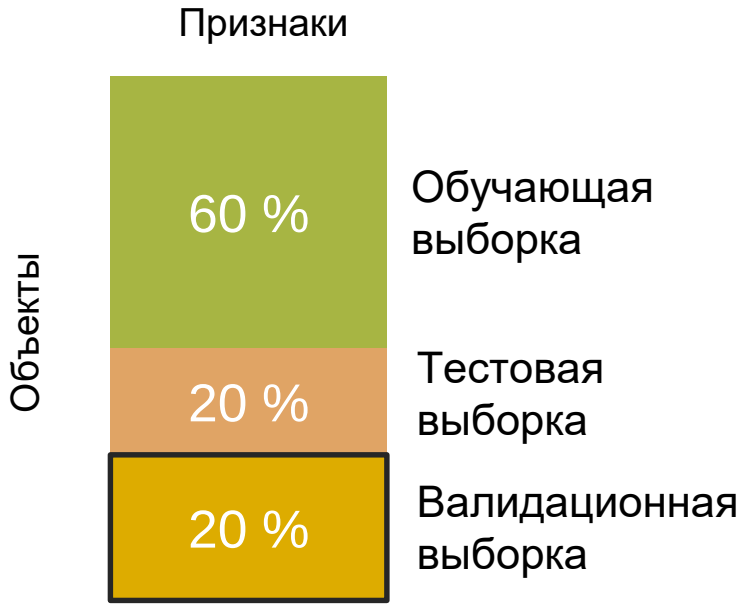


Выделение *тестовой* и *валидационной* выборки

Обычный случай: есть одна выборка данных, по которой нужно оценивать качество модели

Решение: положить в тестовую выборку то, что больше всего похоже на использование модели в реальности, выделить дополнительную валидационную выборку для итогового тестирования

Пример: есть данные за 2017-2020, для обучения используем 2017-2018, для теста – первые два квартала 2020, для валидации – последние два квартала.

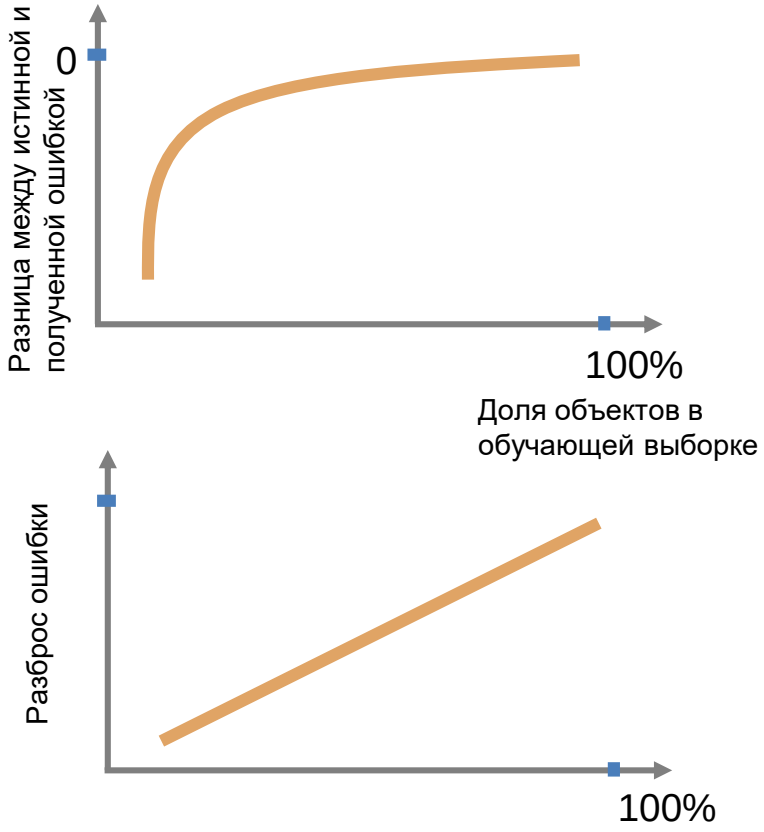


Выбор размера обучающей, тестовой и валидационной выборки

Смещение. Если возьмем слишком мало данных для обучения – то качество модели, обученной на подвыборке будет меньше, чем качество модели, обученной на всех доступных данных

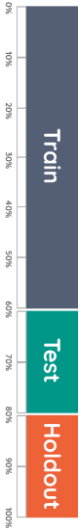
Разброс. Мы усредняем ошибку по тестовым данным. Если их мало – то оценка ошибки будет не очень точной.

Получается, что в таком подходе мы используем не все данные для обучения и не все для тестирования. Можно ли делать иначе?

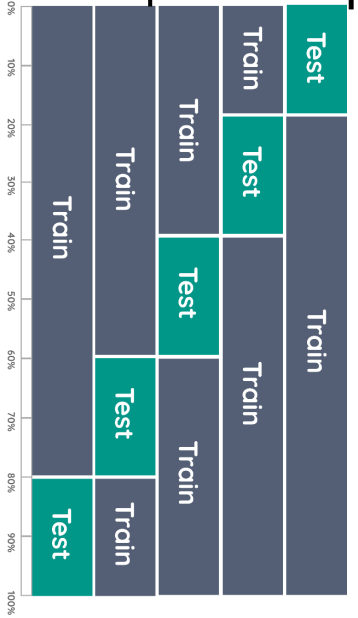


Скользящий контроль / Cross-validation

Обычное разбиение



Скользящий контроль



Skoltech

Преимущества и недостатки скользящего контроля

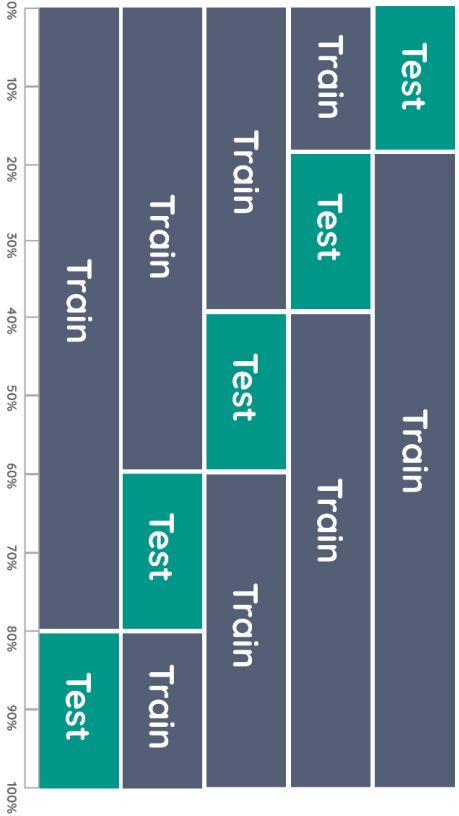
Обычно в процессе скользящего контроля делают 5 разбиений

Можно оценить устойчивость модели: если ошибки сильно отличаются, то модель неустойчива.

Нужно учить много моделей!

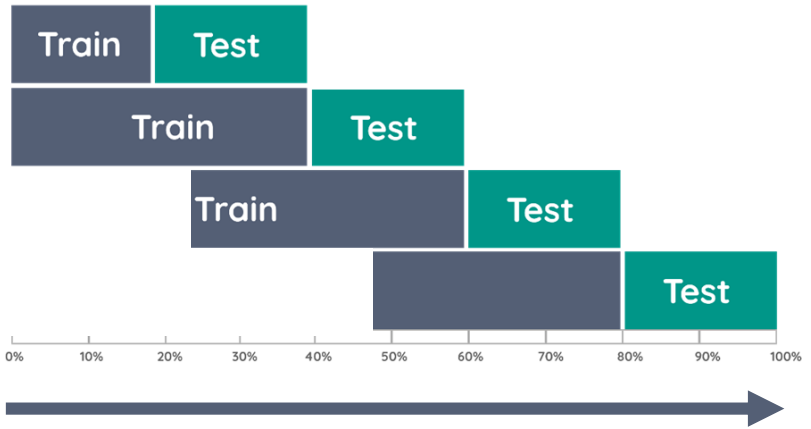
Leave-one-out cross-validation:

Размер тестовой выборки в процессе скользящего контроля – 1

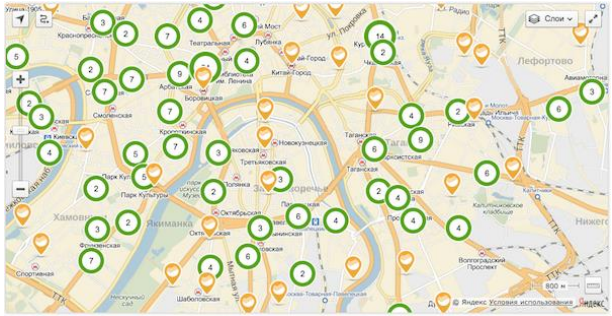


Специальные случаи оценки качества моделей

Данные удобно *разбивать по времени*: это похоже на реальный сценарий использования модели, где нам доступны только данные из прошлого, а предсказывать мы должны будущее.

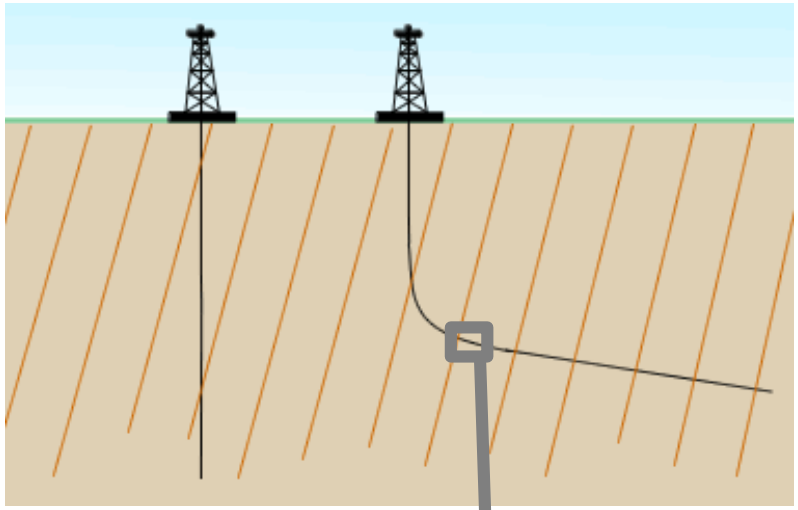


Данные удобно *разбивать по географии*: если данные для разных регионов – это тоже нужно учесть при разбиении выборки на обучение и контроль!



Skoltech

Кейс, прогноз типа породы



Мы обучали модель на одних скважинах, а тестировали на других

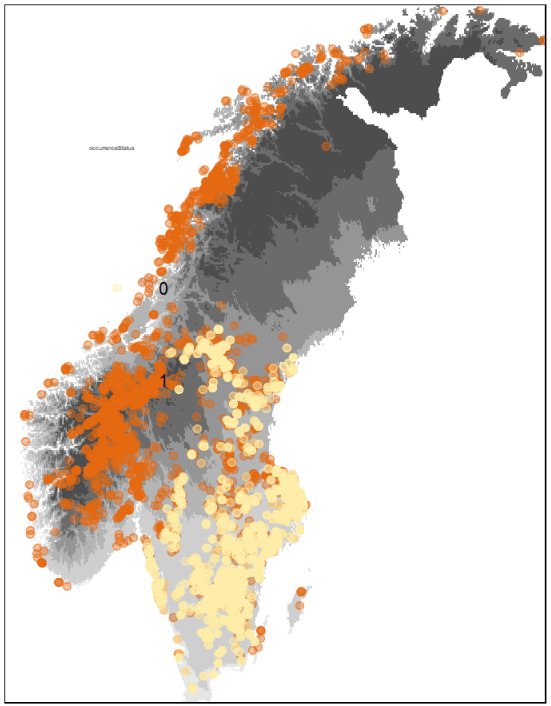
Если разбивать данные без учета принадлежности к скважинам – получается оптимистичная оценка качества модели.

Для проекта про нефтяные месторождения объект – часть интервала бурения

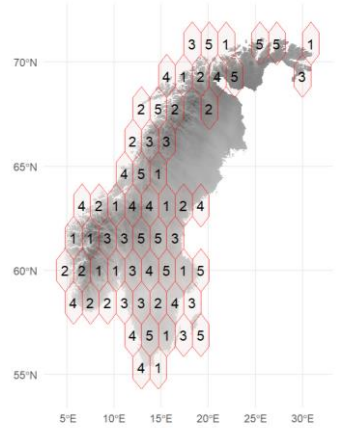
Целевая переменная – тип породы в интервале

Skoltech

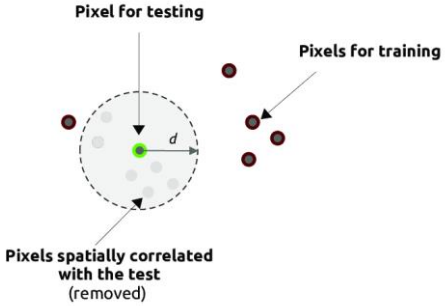
Кейс, прогноз наличия вида



- наличие
- отсутствие

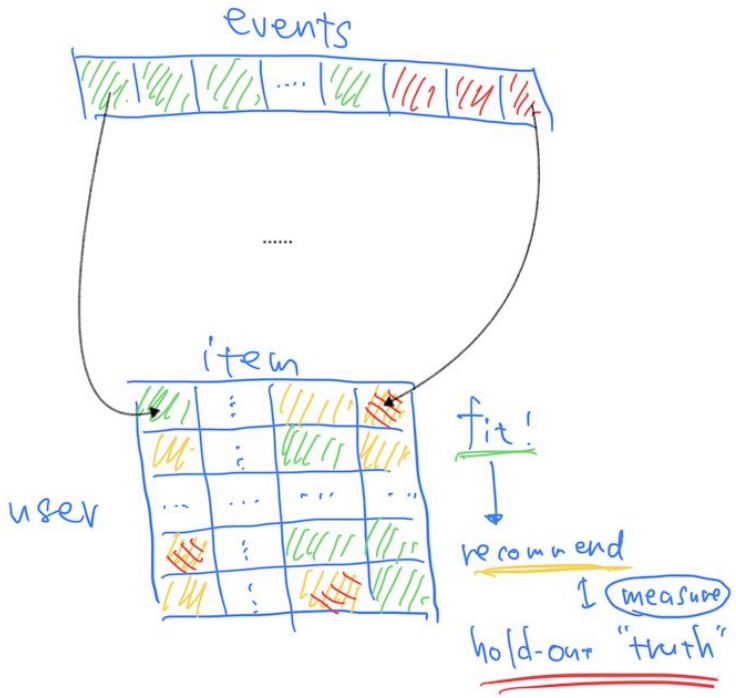


Разбиение на фолды



Skoltech

Валидация для рекомендательных систем

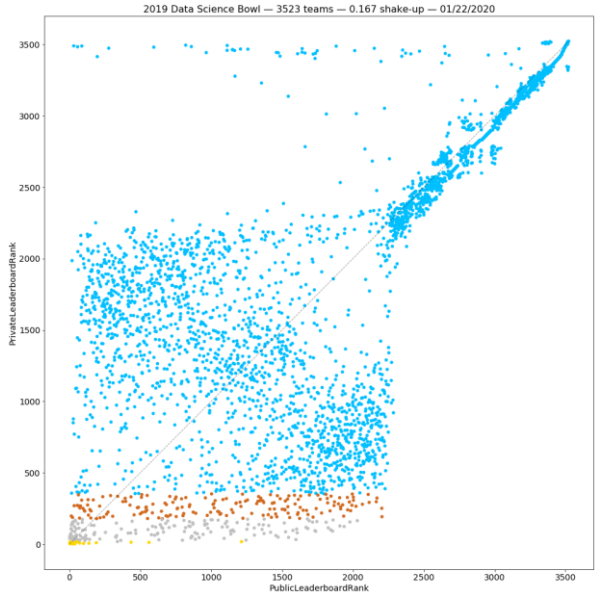


Shake-up на kaggle

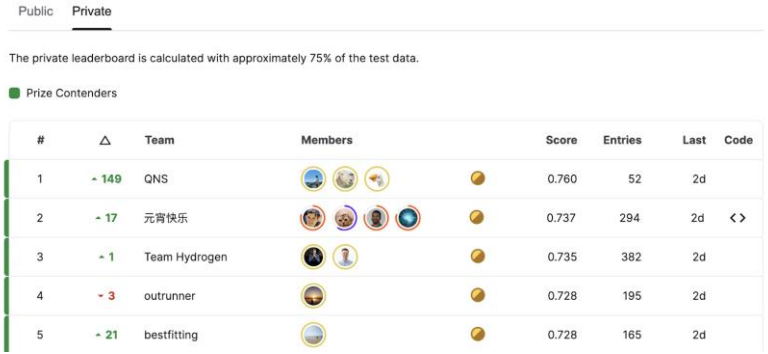
Есть три выборки:
- Train
- Public test
- Private test

- Метки участникам доступны только на Train.
- Они с помощью модели ставят метки на Test,
- По части Public test считается качество модели, которое видно участникам соревнования
- По части Private test считается качество модели, которое видно участникам соревнования

Всегда происходит shake-up



	0-99 Teams	100-249 Teams	250-999 Teams	1000+ Teams
Bronze	Top 40%	Top 40%	Top 100	Top 10%
Silver	Top 20%	Top 20%	Top 50	Top 5%
Gold	Top 10%	Top 10	Top 10 + 0.2%*	Top 10 + 0.2%*



Данные по соревнованиям: https://www.kaggle.com/datasets/daniboy370/competitions-shakeup

Skoltech

Как делать скользящий контроль правильно?

1. Отдельная валидационная выборка, не трогаем ее до конца проекта

2. Скользящий контроль с большим количеством разбиений, чем больше – тем лучше

3. Out-of-time валидация, имитируем реальные условия

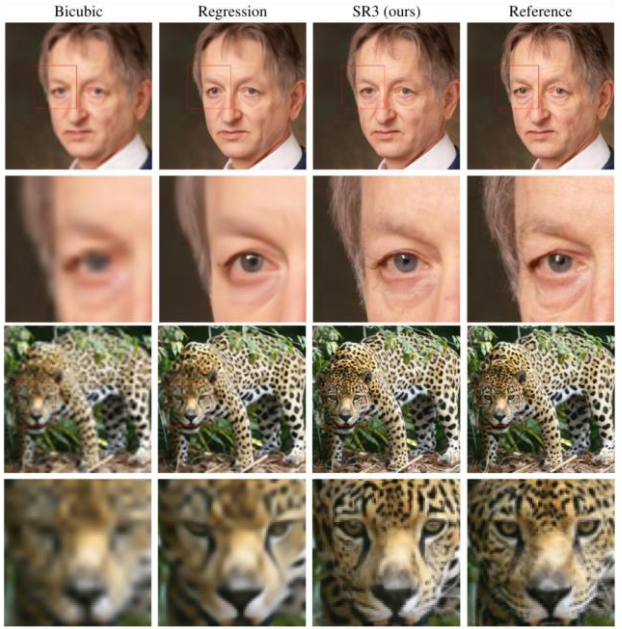
4. Принимаем во внимание специфику задачи, когда строит процедуру скользящего контроля – **думаем во время построения процедуры валидации**

Еще про метрики

- Выбор метрики: качество бизнес-процесса не всегда коррелирует с выбранной метрикой качества.
  Пример: Netflix prize

- Может не быть единой метрики
  Пример: NDCG@5, NDCG@20, NDCG@50, surprise

- Нормальных метрик может не существовать
  Пример: PSNR для super-resolution, генеративные модели



Skoltech

Анализ внутренних данных по проведенным валидациям в крупном банке

Skoltech

Внутренние данные по проведенным валидациям

В исходных данных 1014 пар валидаций моделей: первичная и периодическая

Целевая переменная: перешла ли модель в красную зону при периодической валидации

Признаков всего 26:

- дней с прошлой валидации, результат прошлой валидации
- блок-заказчик модели,
- целевая метрика (GINI, R2)
- тип алгоритма и др.

Целевая переменная для анализа

Результат периодической валидации

Skoltech

Прогноз исхода валидации моделей

1. Верхнеуровневый прогноз перехода моделей в красную зону в разрезе определенных классов моделей с точностью 75-85% и полнотой 60-70%
2. Качество прогноза может быть повышено при использовании низкоуровневых данных по динамике качества каждой модели и изменению факторов

Выборка валидаций: 662 объекта, из них для 154, переход в красную зону

Результаты скользящего контроля в процентах, усреднение по десяти повторениям пяти разбиений

	CatBoost	XGBoost	LightGBM	Logistic Regression	Decision Tree
Precision	86%	**81%**	80%	75%	77%
Recall	62%	**71%**	70%	57%	58%
Accuracy	89%	**90%**	89%	86%	86%

Skoltech

Анализ эффектов

Наиболее важные признаки:

- Число дней с последней валидации
- Результат последней валидации — Желтый
- Значение метрики при последней валидации
- Длительность последней валидации

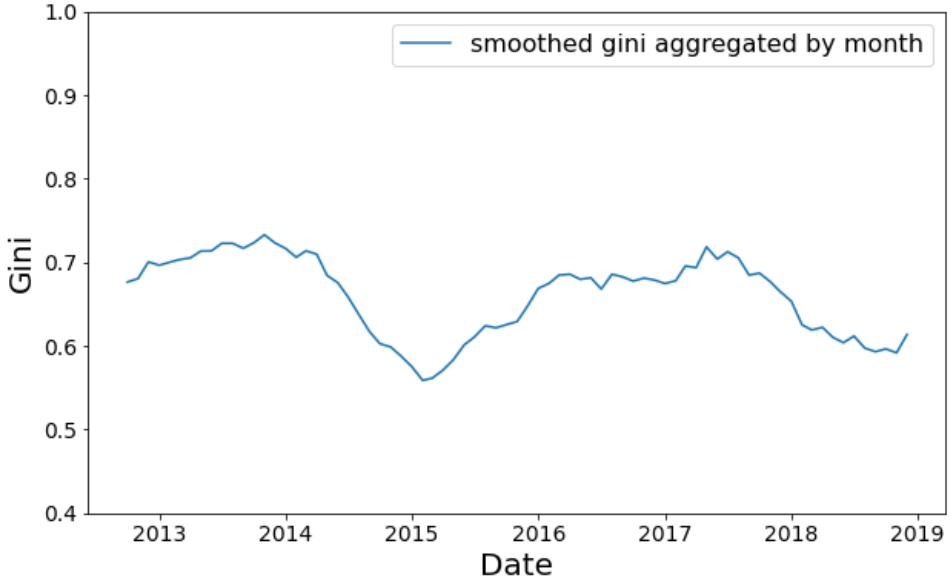
	Ранг значимости фактора *(усредненный по оценкам алгоритмов)*	Направление связи с вероятностью перехода модели в красную зону *(при росте фактора)*
Результат последней валидации — Желтый	1	
Число дней с последней валидации	2	
Задача класса PD	3	
При последней валидации значение метрики улучшилось	4	
Длительность последней валидации	5	
Задача класса отклика для розничных клиентов	6	
Модель для розничных клиентов (независимо от типа задачи)	7	
Тип задачи — прогноз денежных потоков	8	
Значение метрики при последней валидации	9	
Длительность последней валидации относительно среднего	10	

Skoltech

Динамика метрики качества GINI

Данные для рейтинговой модели, 01/2012-12/2018

Агрегация качества модели GINI по месяцам, скользящее среднее



- *У качества модели выраженная волатильность*
- *На качество модели влияют кризисы*

Skoltech

Предсказание метрик качества работы модели

3 статистически значимых фактора:

1. Среднее по кварталу для признака *qual_factor_7* – качественная оценка возможности оттока капитала в компании

2. Дисперсия по кварталу для признака *quant_factor_4* – соотношение операционной прибыли к выручке

3. Цена нефти *oil_price*

Skoltech

Выводы

Выводы

- Подходы автоматизированного машинного обучения помогают подбирать гиперпараметры на всем этапе решения задачи

- Для более простого случая, выбора модели, достаточно использовать подходы с оптимизацией гиперпараметров

- Существует множество подходов для эффективной и быстрой оптимизации гиперпараметров. Не стоит использовать наивный поиск по сетке

- Оценка ошибки по выборке – нетривиальная задача