

Соревновательный анализ данных. Обзорная лекция

Гущин Александр, MADE, осень 2020

1.  Введение в соревнования: что это, как это и зачем
2.  От простых решений к сложным
3.  Организация работы по одиночке и в команде

Введение в соревнования: что это, как это и зачем

Например: является ли слово фамилией?

X (объект) у (таргет)

человек 0

бутерброд 0

иванов 1

студент ?

петров ?



InClass Prediction Competition

DMIA sport intro: Surnames classification

Определите слова, являющиеся фамилиями

115 teams · 3 years ago

Overview Data Notebooks Discussion Leaderboard Rules

Join Competition

Public Leaderboard

Private Leaderboard

The private leaderboard is calculated with approximately 75% of the test data.

⟳ Refresh

This competition has completed. This leaderboard reflects the final standings.

#	Δpub	Team Name	Notebook	Team Members	Score ⓘ	Entries	Last
1	—	Кирилл Тушин			0.98728	25	3y
2	—	Ольга Филиппова			0.97510	9	3y
3	—	Соломенник Михаил			0.97416	29	3y

Visual Doom Competition @ CIG 2018



Соревнования бывают (1)

- По типу сабмита: csv файл или докер образ
- По "игрушечности": Kaggle (баллы, медальки, призы) или Kaggle Inclass
- По набору данных: таблички, тексты, картинки, etc

Соревнования бывают (2)

- По времени: соревнование (~неделя и больше) или хакатон (обычно до недели)
- По беспристрастности оценки: четкая метрика или оценка жюри
- По степени определенности: открытый или закрытый лидерборд

Соревнования бывают (3)

- По “устойчивости” метрики: от рандома до LocalCV=PublicLB=PrivateLB

Сабмиты

Даны: x_train, y_train, (x_test)

Найти: y_test

- Сабмитим csv файл: x_test можно пощупать
- Сабмитим docker-образ: x_test не пощупать

Данные бывают очень разнообразны:

- таблицы (tsv, csv)
- тексты (txt, json)
- картинки (jpg, png)
- аудиозаписи (wav, mp3)
- графы
- медицинские данные
- временные ряды
- байт-коды
- ... и так далее

Лидерборд

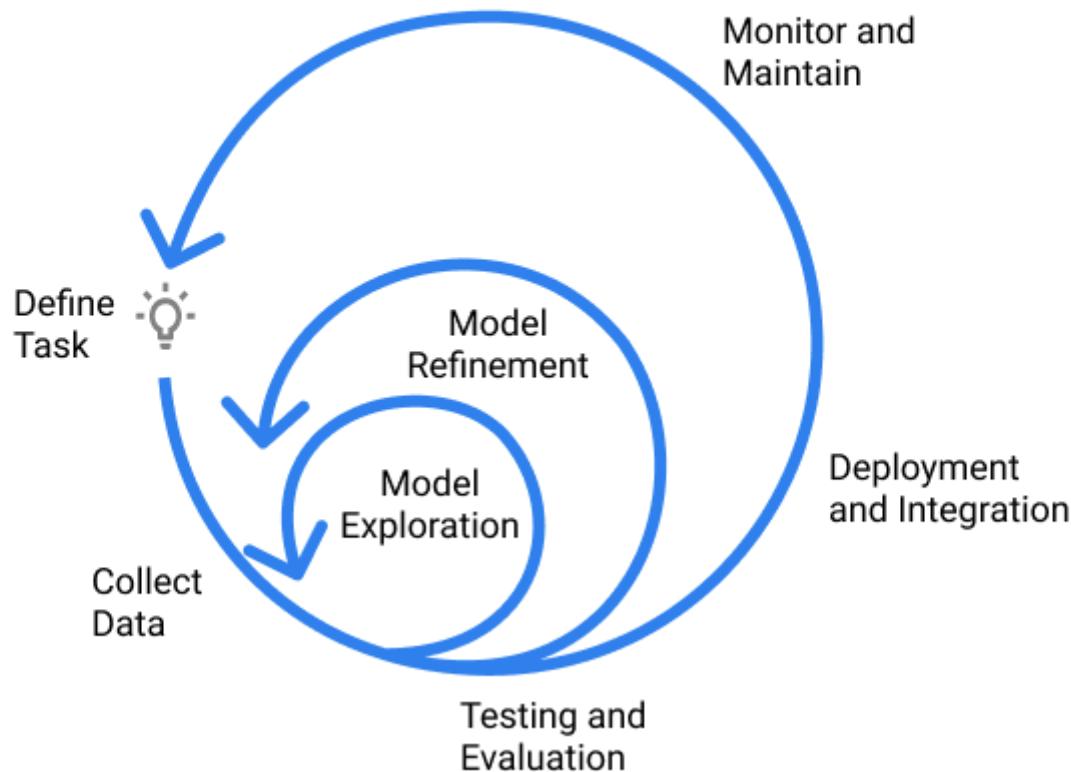
$$\text{LB score} = f(y_{\text{true}}, y_{\text{pred}})$$

Например: является ли слово фамилией?

X (объект)	y (таргет)	часть
человек	0	train
бутерброд	0	train
иванов	1	train
студент	?	public test
петров	?	private test

Отличие от индустриальных задач

Machine Learning Development Lifecycle



<https://www.jeremyjordan.me/ml-projects-guide>

Отличие от индустриальных задач

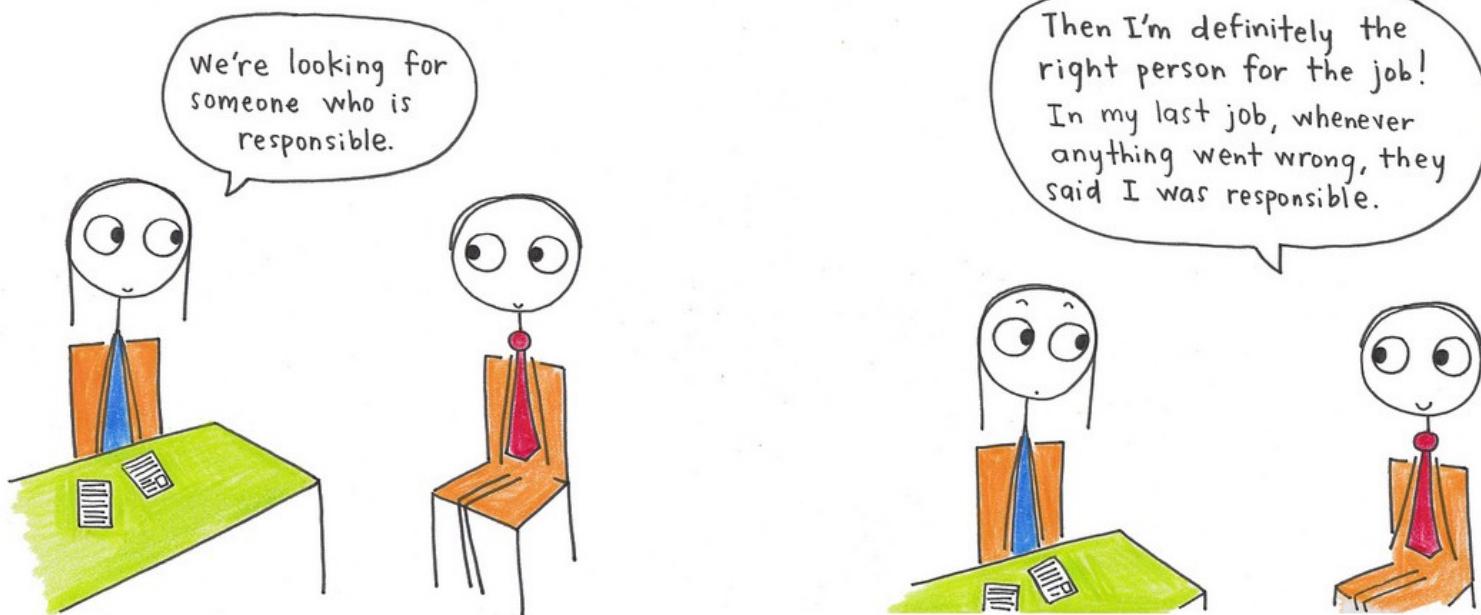
Аспект	Жизнь	Соревнование
Формализация	Y	N
Выбор метрики	Y	N
Production	Y	N
Вычислительная сложность	Y	N
Сбор данных	Y	N/Y
Сложность модели	Y	N/Y
Значение метрики	Y	Y



Единственная вещь, о которой нужно думать

Зачем нужны соревнования организаторам?

- Решение бизнес-задач
- Пиар
- Найм



Зачем нужны соревнования участникам?



- Обучение
- Знакомства
- Карьера
- Деньги (редко!)

Зачем нужны соревнования сообществу?



- Решения задач
- Объективное сравнение реализаций алгоритмов ML
- Данные
- Экосистема
- Подогревают интерес к DS

Призы

- На платформах
 - Денежные призы
 - Swag
 - баллы и медальки
- В других местах
 - Выступления на митапах
 - Рассказы в блогах

Типичные правила соревнований (1)

-  Дедлайны
 - После дедлайна заслать решение уже нельзя
 - Есть дедлайн на первый сабмит
 - Есть дедлайн на объединение в команду
-  Поделиться советом/кодом/данными
 - Можно в командах
 - Можно публично на форуме соревнования
 - Нельзя приватно

Типичные правила соревнований (2)

-  **Данные**
 - Внешние данные могут быть запрещены
 - Или частично разрешены и требуется поделиться ими на форуме заблаговременно
-  **Команды**
 - Удалить участника из команды нельзя

Соревнования проводят разные платформы

- kaggle.com
- drivendata.org
- boosters.pro
- aicrowd.com
- и так далее

Есть сборные списки текущих соревнований, например,

<https://ods.ai/competitions>

Как решать соревнования

От простого к сложному

1. Изучить задачу и метрику
2. Засабмитить константный бейзлайн
3. Написать фреймворк
4. Генерировать фичи
5. Тюнить модели
6. Объединять решения



Это - основные темы курса *How to Win a DS Competition* на курсере

Изучение задачи и метрики

1. 🕶️ Внимательно все прочитать
2. 🔍 Исследовать данные
3. 🔎 Искать необычные закономерности и лики

Удобно записывать все идеи в список



Exploratory data analysis

1. Лучше понять задачу
2. Найти закономерности
3. Найти выбросы
4. Найти лики и баги в выгрузке
5. Придумать новые гипотезы и фичи
6. Понять, какие модели предпочтительнее

Изучайте предметную область – это интересно и пригодится
при решении и других задач.

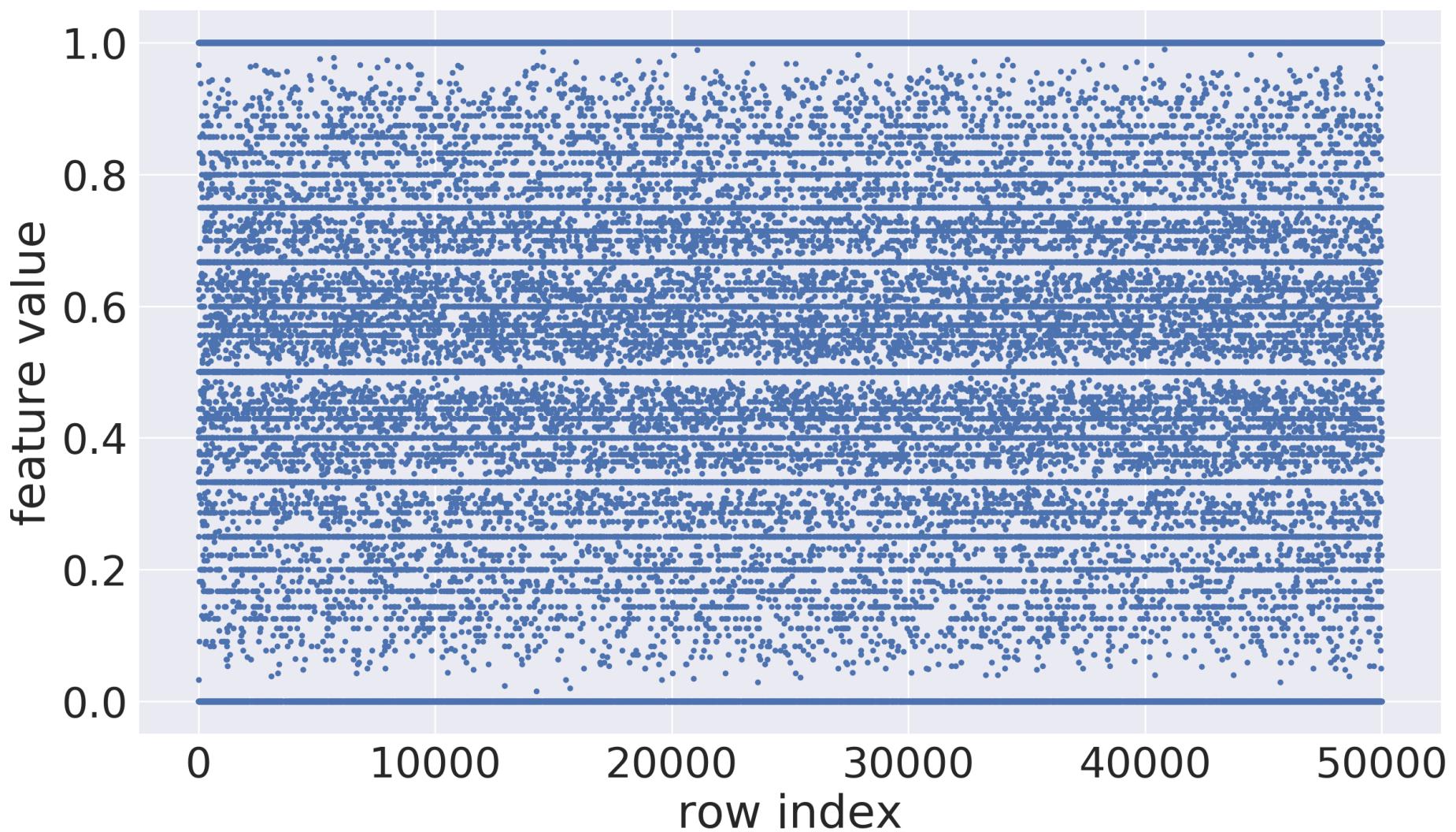
Способы смотреть на данные

1. Признаки
 1. Отдельные признаки
 2. Группы признаков
2. Объекты
 1. Отдельные объекты
 2. Группы объектов
3. Файлы
 1. Содержимое файлов
 2. Метаданные файлов

Стоит также помнить о таргете

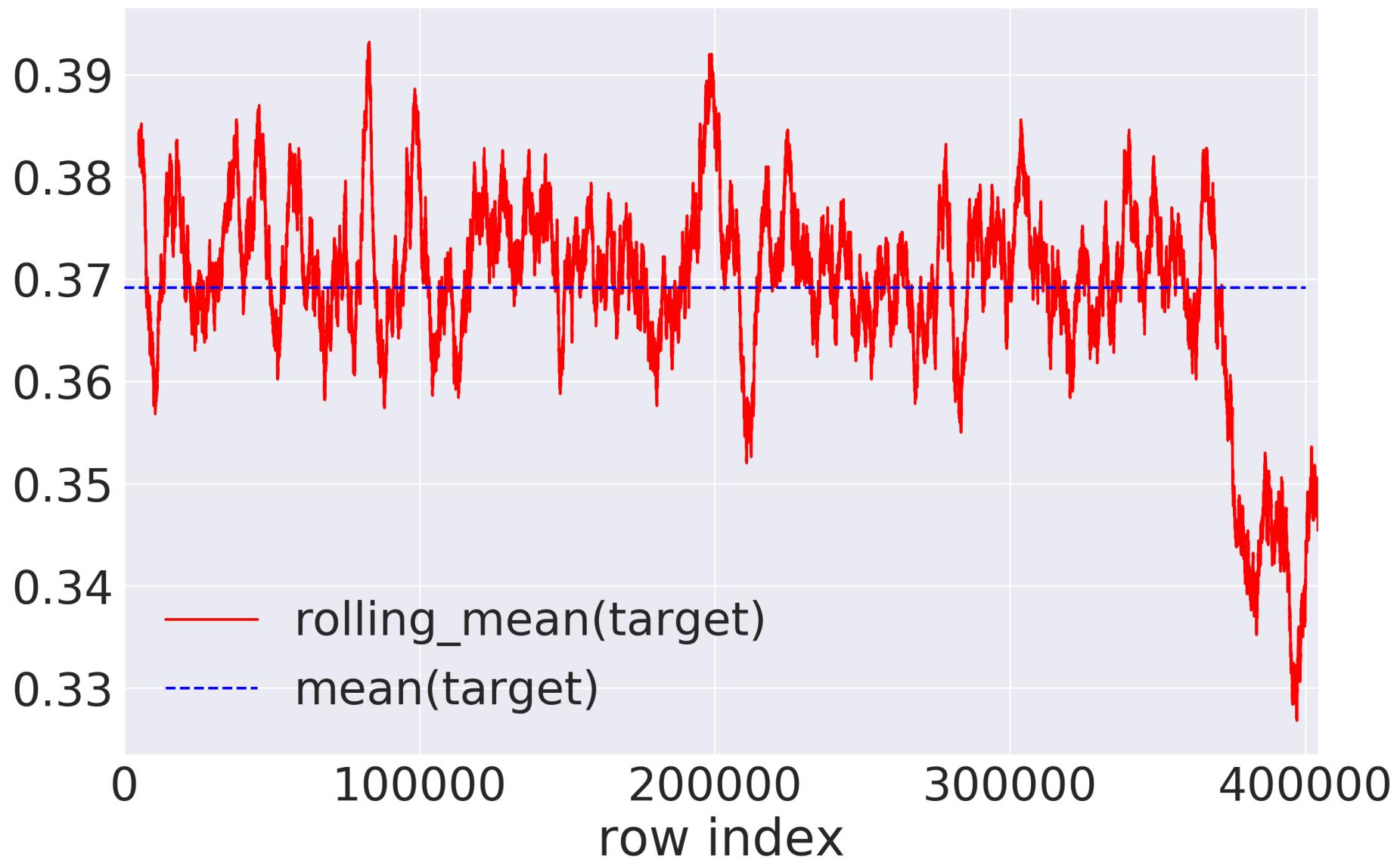
Исследовать один признак.
Зачем?

- обратить анонимизацию, масштабирование признака
- добавить дополнительные внешние данные
- проверить значения признака на потенциальные проблемы



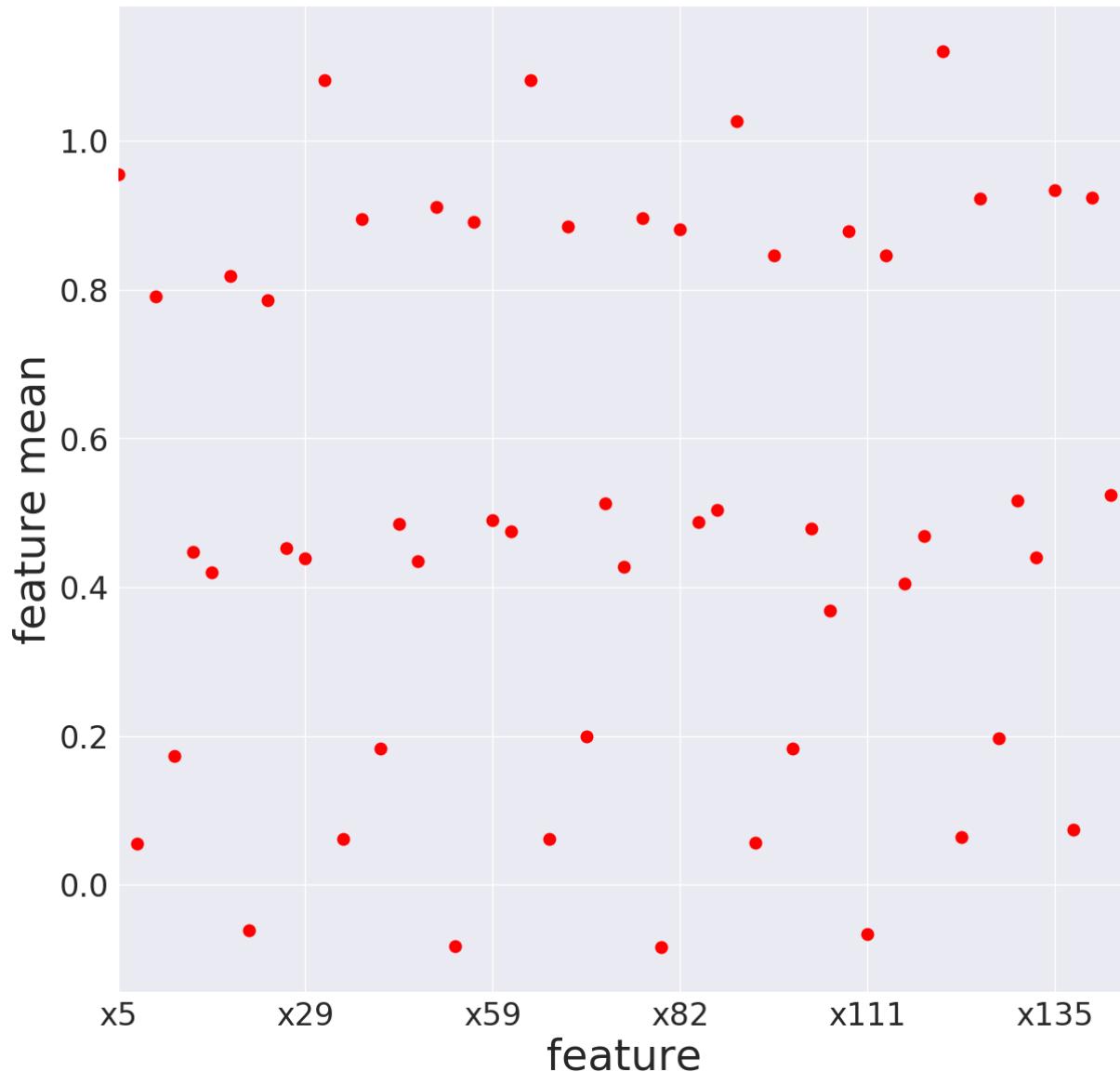
Проверить значения признака на потенциальные проблемы:

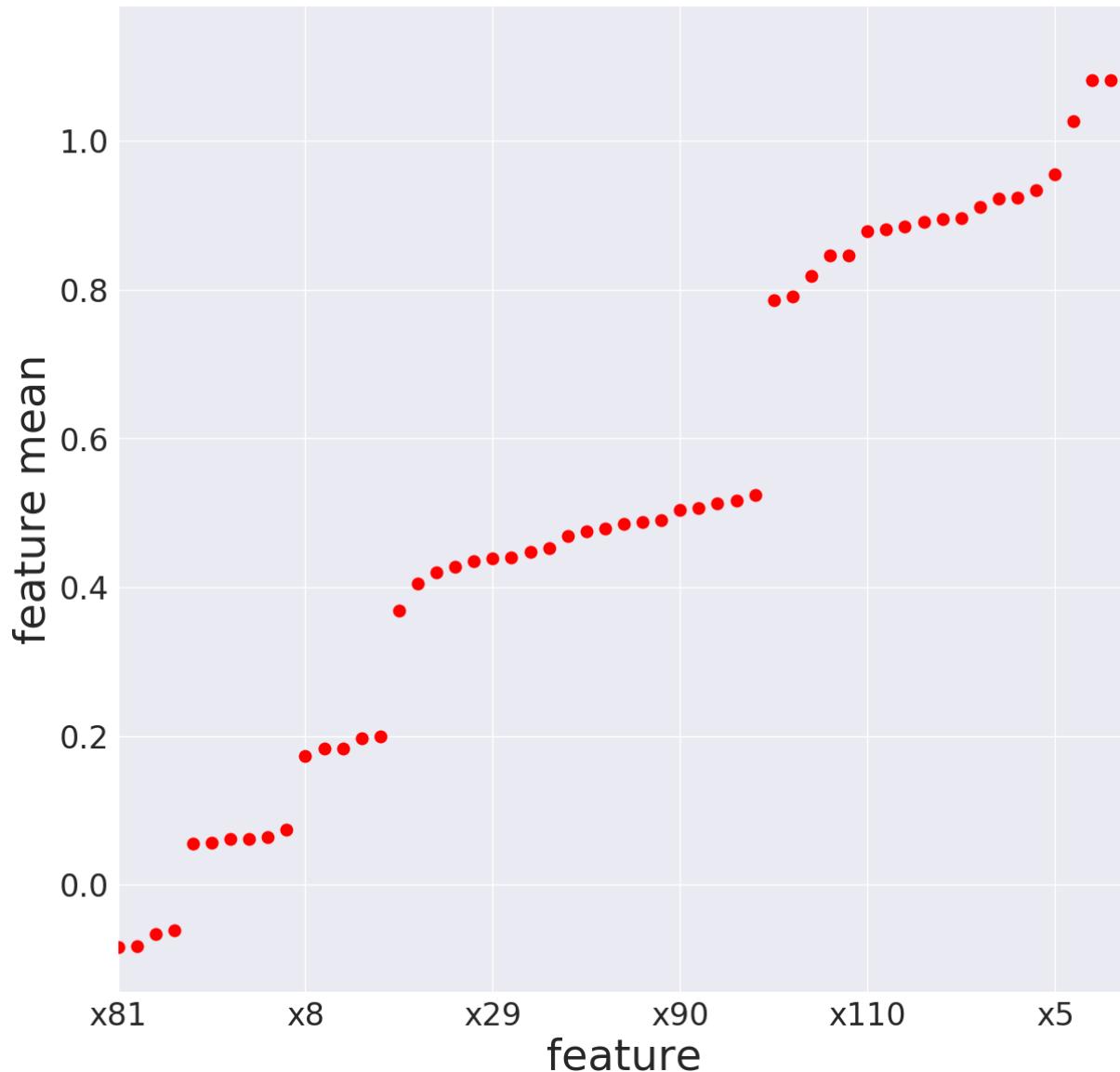
- нереалистичные значения признака
- изучить, меняет ли признак масштаб/распределение со временем



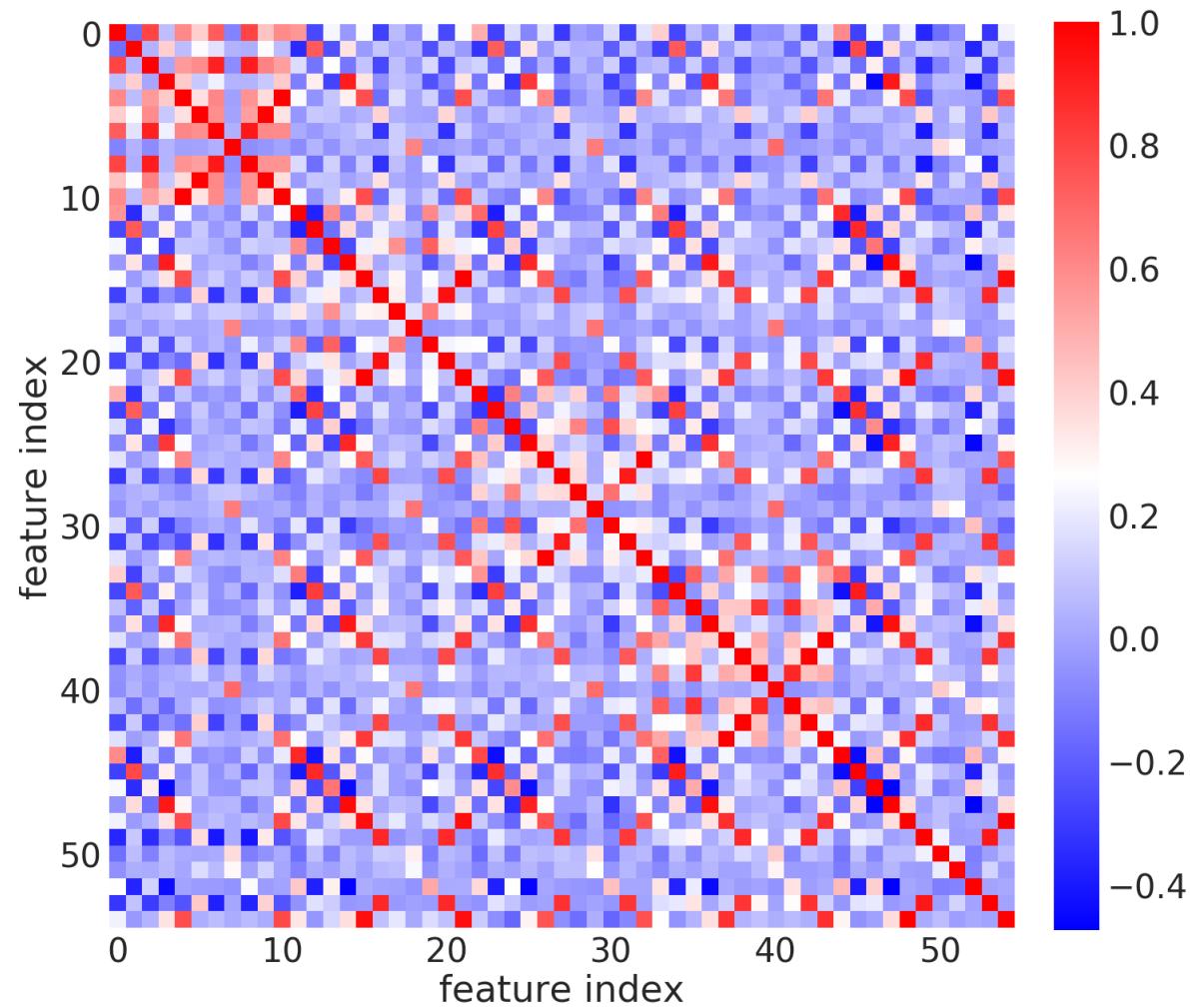
Исследовать признаки вместе.
Зачем?

- Сгруппировать признаки
- Изучить взаимодействия признаков





Матрица корреляций между признаками



Изучение метрики / функции потерь

-  Популярные метрики - знать
 1. Оптимальные константы для метрик
 2. Какие алгоритмы оптимизируют эту метрику?
 3. Особенности метрик
-  Странные метрики - пробовать
 1. Ответить на вопросы выше теоретически и экспериментально
 2. Свести к популярной метрике или приблизить ей

Классификация

- Accuracy
- Logloss
- ROC AUC
- F-score

Как найти оптимальную константу?

Регрессия

- MSE / RMSE
- MAE
- MAPE

Какие алгоритмы оптимизируют эти функции потерь?

Как заставить алгоритм оптимизировать MAPE?

Странные метрики, например,

- коэффициент корреляции Пирсона
- MAE при целых y_{true}
- Cohen's kappa
- ...

Константный бейзлайн

Где взять стартовый скор

1. Лидерборд
2. Обучить модель и засабмитить
3. Посчитать константу и засабмитить

Зачем?

1. Оценивать улучшение
2. Проверить Train/Public на смещение
3. Оценивать сколько людей на ЛБ лучше чем константа
4. Есть соревнования, где константа попадает в топ...

Написание фреймворка

- Фреймворк - ноутбук / скрипт, который позволяет сделать сабмит
- Его можно и нужно дорабатывать в течении всего соревнования
- Его цель - облегчить и ускорить этот процесс, и желательно - сделать воспроизводимым

Data Mining In Action

How to do ML?

1. Import libraries

```
In [1]: from sklearn import datasets, model_selection, neighbors, metrics
```

2. Load data

```
In [2]: data = datasets.load_iris()
```

3. Split data into train and test

```
In [3]: X_train, X_test, y_train, y_test = model_selection.train_test_split(  
                           data.data, data.target, test_size = 0.3, random_state = 0)
```

4. Fit model

```
In [4]: model = neighbors.KNeighborsClassifier()
```

```
In [5]: model.fit(X_train, y_train)
```

```
Out[5]: KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',  
                           metric_params=None, n_jobs=1, n_neighbors=5, p=2,  
                           weights='uniform')
```

5. Make predictions

```
In [6]: predictions = model.predict(X_test)
```

6. Estimate quality

```
In [7]: metrics.accuracy_score(predictions, y_test)
```

```
Out[7]: 0.9777777777777777
```

That's it!



Типичная схема

№ Шаг



, чтобы

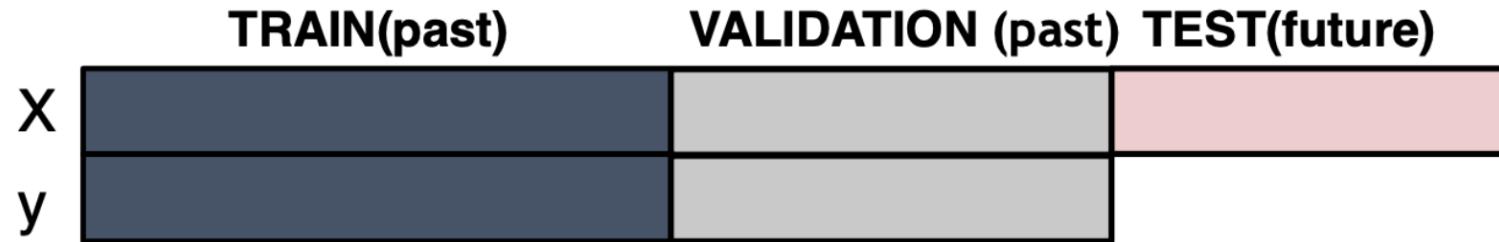
-
- | | | |
|---|---------------------------------|--------------------|
| 1 | Загрузка данных и препроцессинг | не ждать |
| 2 | Разбиение на трейн/тест | воспроизвести |
| 3 | Генерация фичей | не ждать |
| 4 | Обучение модели | смешивать прогнозы |
| 5 | Смешивание моделей | отправить самбит |
-

Разбиение на трейн/тест

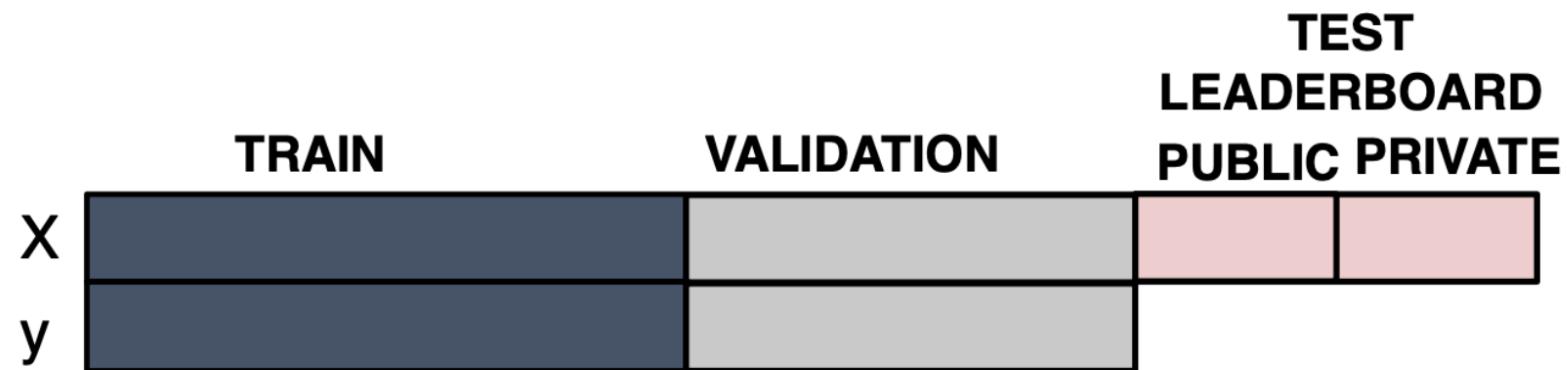
1. Holdout / KFold - по скорости вычислений, устойчивости метрик
2. Воспроизведение разбиения на трейн/тест для выделения контроля

В чем отличие от DS на работе?

В реальной жизни



В соревнованиях



Частые разбиения

- Случайное - i.i.d
- По времени - есть временная зависимость
- По id - для id есть несколько объектов в выборке
- Комбинации, например, по времени и id

Примеры разбиений по id

- id клиента
- id города
- id рекламной компании
- id поездки на автомобиле

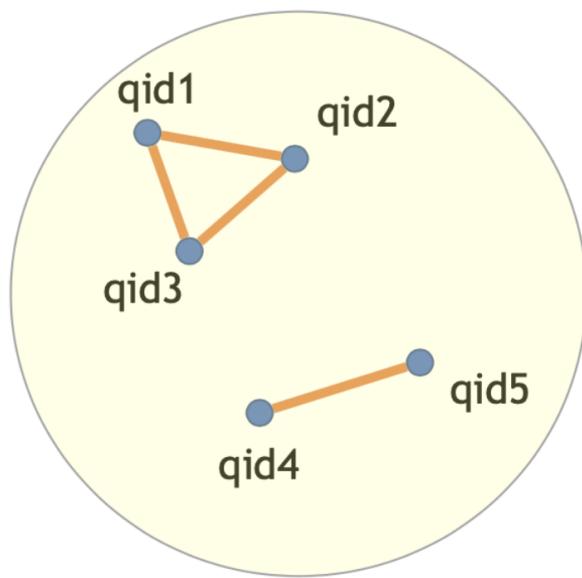
Пример сложной валидации в Quora Question Pairs



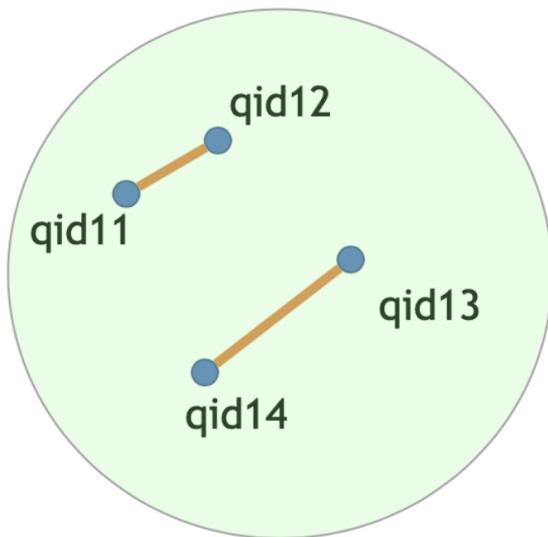
	id	qid1	qid2	question1	question2	is_duplicate
0	0	1	2	What is the step by step guide to invest in share market in india?	What is the step by step guide to invest in share market?	0
1	1	3	4	What is the story of Kohinoor (Koh-i-Noor) Diamond?	What would happen if the Indian government stole the Kohinoor (Koh-i-Noor) diamond back?	0
2	2	5	6	How can I increase the speed of my internet connection while using a VPN?	How can Internet speed be increased by hacking through DNS?	0
3	3	7	8	Why am I mentally very lonely? How can I solve it?	Find the remainder when 23^{24} is divided by 24,23?	0
4	4	9	10	Which one dissolve in water quickly sugar, salt, methane and carbon di oxide?	Which fish would survive in salt water?	0
5	5	11	12	Astrology: I am a Capricorn Sun Cap moon and cap rising...what does that say about me?	I'm a triple Capricorn (Sun, Moon and ascendant in Capricorn) What does this say about me?	1

Quora

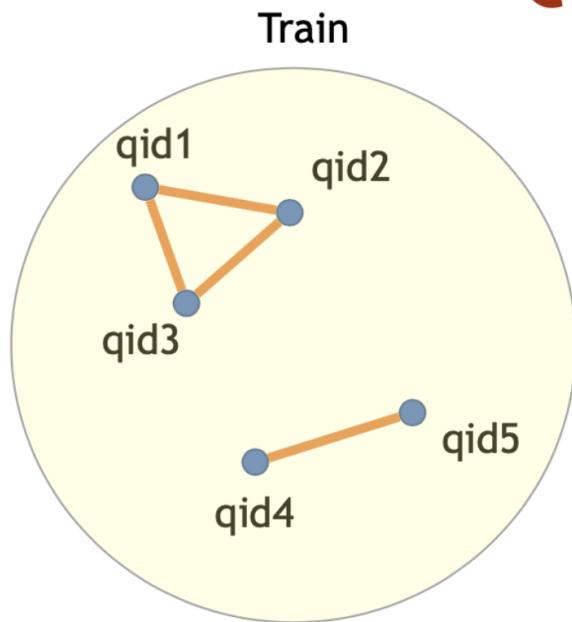
Train



Test



Quora



Разделение 1

Fold1:

qid1 - qid2

qid1 - qid3

...

Fold2:

qid2 - qid3

qid4 - qid5

...

Разделение 2

Fold1:

qid1 - qid2

qid1 - qid3

qid2 - qid3

...

Fold2:

qid4 - qid5

...

Что лучше?

Причины расхождения валидации и лидерборда



- Разбиение трейна на обучение/валидацию отличается разбиения на трейн/тест
- Распределения данных в валидации и teste отличаются
- Недостаток данных в (Public) LB

Leaderboard shuffle

		■ In the money	■ Gold	■ Silver	■ Bronze			
#	△pub	Team Name	Kernel	Team Members	Score ⓘ	Entries	Last	
1	▲ 3	Shize & Nir			0.82907	298	3y	
2	▲ 848	kg_joi			0.82906	35	3y	
3	▲ 6	#1 Leustagos			0.82853	227	3y	+8
4	▲ 1	Why so noise?			0.82846	195	3y	
5	▲ 2024	Michael Hartman			0.82834	22	3y	
6	▲ 229	Noah Xiao @ Accenture			0.82834	44	3y	
7	▲ 4	Rolling Stones (Can't Get No [...]			0.82829	198	3y	
8	▲ 718	Matt Motoki			0.82824	30	3y	
9	▲ 4	no one			0.82823	54	3y	
10	▲ 1240	Bang Nguyen			0.82818	35	3y	

1600	▼ 928	caiomsouza			0.82547	33	3y	
1601	▼ 917	Booster			0.82547	14	3y	
1602	▼ 917	back22010			0.82547	11	3y	
1603	▼ 916	Anorexic Hippo			0.82547	25	3y	
1604	▼ 1361	جانے کب ہوں گے کم، بینک والوں کے غم			0.82547	9	3y	
1605	▼ 967	AnonSci			0.82547	25	3y	
1606	▼ 1155	vettipaiyan			0.82547	15	3y	FREE BUGS
1607	▼ 917	paulperry			0.82547	11	3y	
1608	▼ 1390	fallLeaf			0.82547	62	3y	
1609	▼ 916	ArjoonnSharma			0.82547	34	3y	
1610	▼ 915	__=_=			0.82547	21	3y	
1611	▼ 915	Nilesh Kadam			0.82547	9	3y	
1612	▼ 1151	ShankerDS			0.82547	15	3y	

Leaderboard shuffle



- Данных мало в Public или Private
- Данные в Public и Private отличаются
- Прогноз не слишком отличается от случайного
- Метрика неустойчива к выбросам

Генерация признаков

Целевая переменная

Числовая переменная

Переменные-счетчики

Категориальные
переменные

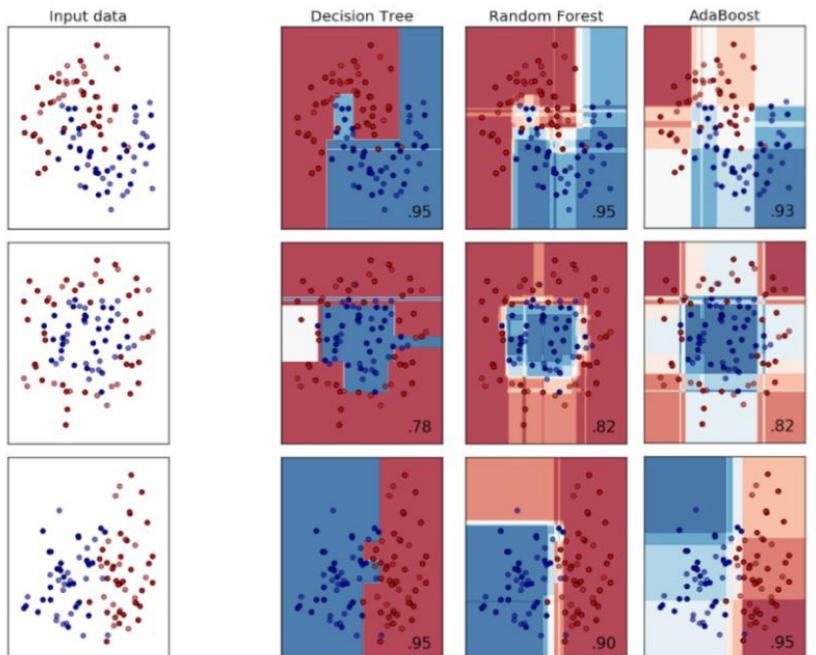
id

Текстовая переменная

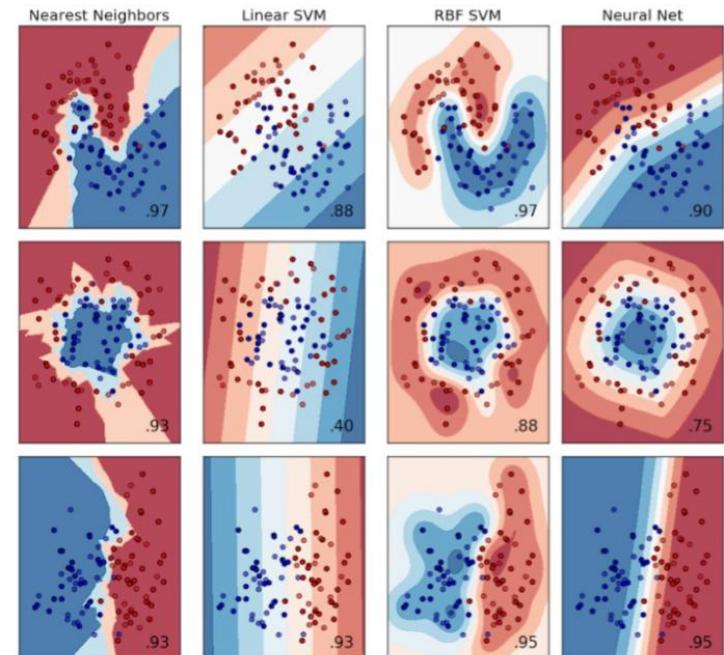
PassengerId	Survived	Pclass	Name				
0	1	0	Braund, Mr. Owen Harris				
1	2	1	Cumings, Mrs. John Bradley (Florence Briggs Th... Th...				
2	3	1	Heikkinen, Miss. Laina				
3	4	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)				
4	5	0	Allen, Mr. William Henry				
5	6	0	Moran, Mr. James				
6	7	0	McCarthy, Mr. Timothy J				
7	8	0	Palsson, Master. Gosta Leonard				
Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0 male	22.000000	1	0	A/5 21171	7.2500	NaN	S
1 female	38.000000	1	0	PC 17599	71.2833	C85	C
2 female	26.000000	0	0	STON/O2. 3101282	7.9250	NaN	S
3 female	35.000000	1	0	113803	53.1000	C123	S
4 male	35.000000	0	0	373450	8.0500	NaN	S
5 male	29.699118	0	0	330877	8.4583	NaN	Q
6 male	54.000000	0	0	17463	51.8625	E46	S
7 male	2.000000	3	1	349909	21.0750	NaN	S

Зависимость от используемых моделей

Модели на деревьях



Остальные модели



Числовые признаки: масштабирование

- MinMax, RobustScaler, StandardScaler...
- Rank:
 $\text{rank}([-100, 0, 1e5]) == [0,1,2]$
 $\text{rank}([1000, 1, 10]) = [2,0,1]$
- Логарифмирование:
 $\text{np.log}(1 + x)$
- Возведение в степень <1:
 $\text{np.sqrt}(x + 2/3)$

Числовые признаки: выбросы

В признаках  и таргете 

- Винсоризация (когда выброс - действительно очень большое значение)
- Генерация фичей (когда выброс означает событие которое может быть важным)
- Восстановление истинных значений (когда возможно)
- Удаление объектов
- Rank

$\text{rank}([-100, 0, 1e5]) == [0, 1, 2]$

$\text{rank}([1000, 1, 10]) = [2, 0, 1]$

Категориальные и порядковые признаки

-  Типичные способы обработки
 - Оставить как есть
 - LabelEncoding
 - One-hot-encoding
 - Кодирование частотой
 - Кодирование числовым признаком
 - Кодирование целевой переменной
-  На что обратить особое внимание
 - Новые категории
 - Количество категорий в признаке
 - Мелкие категории

Временные признаки

- Периодичность

Номер дня в неделе, месяц, сезон, год, секунда, минута, час

- Сколько времени прошло/осталось

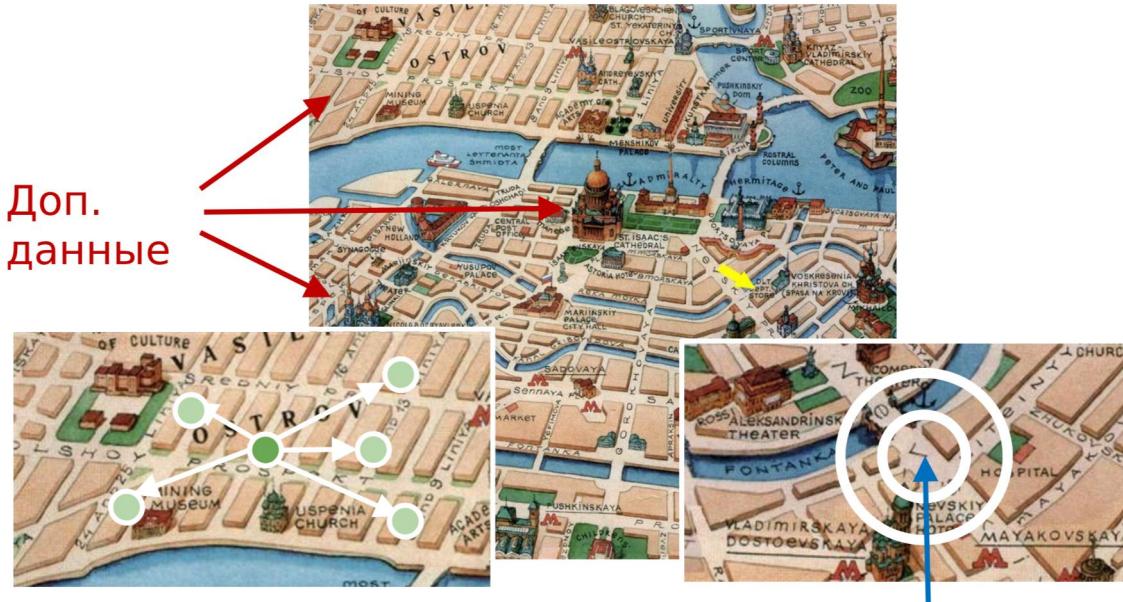
- С одного момента для всех данных
 - Момент зависит от выбора объекта (сколько дней осталось до выходных)

- Разница между моментами времени

- Использование для генерации признаков со скользящим окном

Пространственная структура

- Соседние («интересные») объекты
- Агрегаты и статистики
- Дополнительные данные



St. Petersburg

Выбор и тюнинг моделей

Инструкция из четырех шагов

1. Разобраться, как устроены алгоритмы
2. Выбрать подходящие алгоритмы
3. Выбрать наиболее важные гиперпараметры
4. Тюнить – вручную или автоматически

Линейные модели



Легко интерпретировать

Требует подготовки данных

Просто реализовать

Не лучший результат

“Какой признак изменить,
чтобы стало вот так?”

Постпроцессинг
(отрицательные продажи)

Быстро работает

KNN



Модель для ансамбля

Координаты, габариты,
расстояния

Обучение на эмбеддингах

Реализация sklearn
медленная

Требует подготовки данных

Не лучший результат

В проде долго и затратно

Градиентный бустинг



Обычно лучший, как для соревнований, так и для прода

Относительно быстрая скорость работы

Легко переобучить, накосячить с ликами и неправильной валидацией

Не умеет использовать линейные зависимости между признаками

Нейросети



Незаменимы в ряде задач

Много интересных
применений

Много готовых архитектур и
сохраненных весов

Иногда можно красиво
интерпретировать

Бывают неустойчивы на
небольших датасетах

Побеждает тот, у кого
больше GPU

Сложнее в тюнинге и выборе
архитектуры

Часто - “черный ящик”

Кодирование средним

- Проблема: Категориальные признаки с большим количеством уникальных значений - это трудно для алгоритмов ML
- Решение: Применим некоторое преобразование признака, которое сделает новые значения упорядоченными по
 -  другому признаку (например, при предсказании стоимости м² заменим признак "город" на признак "средняя зарплата")
 -  таргету (кодирование средним)

Id	Стоимость квартиры	Город	Средняя зарплата	Средняя стоимость м^2
1	200000	Москва	80000	180000
2	80000	Волгоград	30000	75000
3	110000	Казань	40000	100000
4	100000	Уфа	35000	90000
5	60000	Березняки	25000	75000
6	80000	Новосибирск	30000	60000

Чем можно кодировать

- Регрессия
 - Средним или медианой таргета
- Классификация
 - Частотой значения таргета
 - $\#P - \#N$
 - $\text{LN}(\#P - \#N)$

Основная проблема - переобучение

- подсчет значений нового признака с помощью K-Fold внутри трейна
- сглаживание - ближе к среднему `y_train.mean()` для редких категорий
- `expanding mean` - при подсчете для объекта i используем только объекты с 1 по $i-1$

Ансамблирование

Тroe голосуют независимо, вероятность правильного ответа 0.7

Исход	Формула	Вероятность
Правы трое	$0.7 * 0.7 * 0.7$	0.3429
Правы двое	$0.7 * 0.7 * 0.3 * 3$	0.4409
Прав один	$0.7 * 0.3 * 0.3 * 3$	0.189
Никто не прав	$0.3 * 0.3 * 0.3$	0.027

Вероятность ответить верно - 0.7838

Ансамблирование

-  Ингредиенты
 - Данные, как можно разнообразней
 - Модели, как можно разнообразней
-  Рецепты
 - Усреднение
 - Линейные комбинации
 - Смеси экспертов
 - Бустинг
 - Стекинг

Источники разнообразия в данных

-  Предобработка
 - Декомпозиция данных по признакам и объектам
 - Преобразования признаков: `log`, `sqrt`, `TFIDF`, `PCA`, `NMF`, `tSNE`, `encoder-decoder`, `w2v`, `fasttext`, `BERT`, `mean encoding`, `OHE`
 - Комбинации групп и преобразований
-  Сэмплирование
 - По объектам (`bootstrap`, `undersampling`, `oversampling`)
 - По признакам (`RSM`)

Источники разнообразия в моделях

-  Алгоритмы одного семейства с разными параметрами
 - Глубина дерева
 - Число соседей
 - Число слоев в сетке
 - Регуляризация
-  Разные семейства алгоритмов
 - Линейные модели
 - Наивный байес
 - Факторизационные машины
 - Деревья
 - Unsupervised learning
 - Нейронные сети (разная топология)

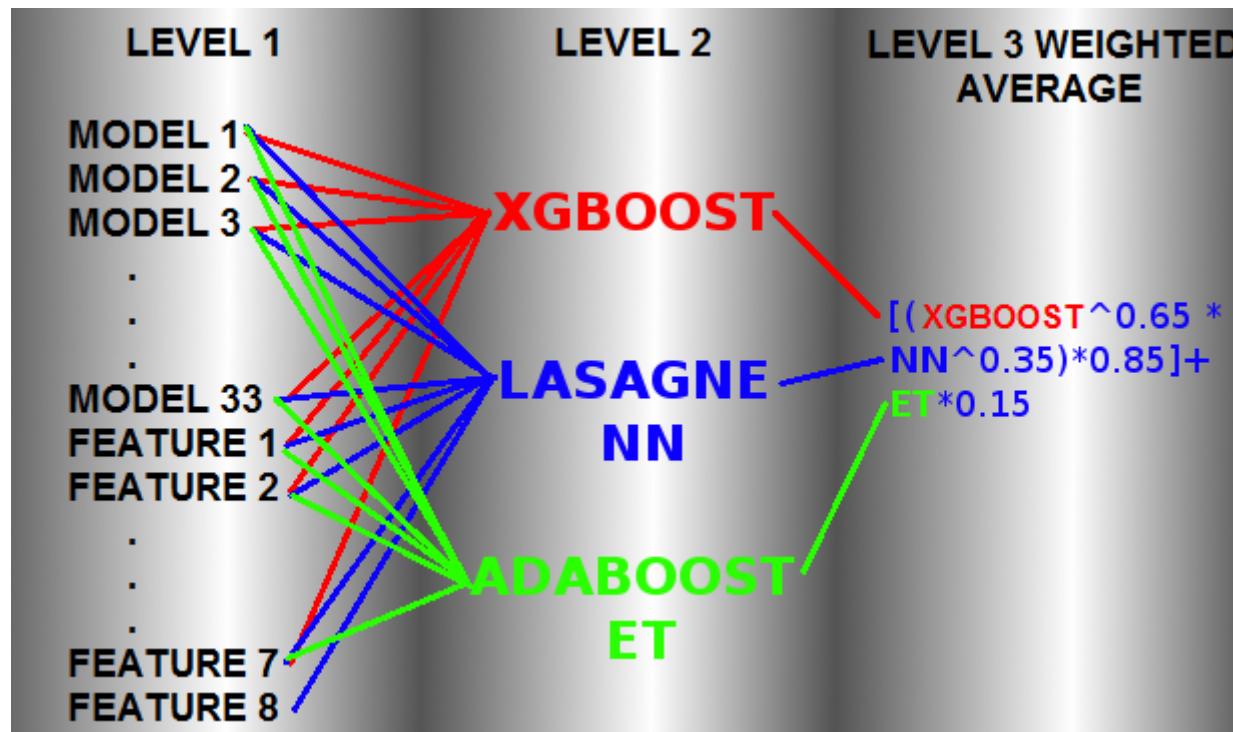
Bootstrap, bagging и голосования: примеры

- Стандартный метод
 - DecisionTree + сэмплирование + усреднение $\sim=$ RandomForest
- kaggle-avito-prohibited-content
 - kNN + линейная модель = решение в топ5
- kaggle-malware
 - xgboost + oversample($a=7$) + усреднение
- kaggle-tube-pricing
 - NN + dropout\dropconnect + усреднение
- любой конкурс
 - xgboost + сэмплирование(subsample, colsample_bytree) + усреднение

Bootstrap, bagging и голосования

-  Плюсы
 - легко реализовать
 - эффективен
 - снижает разброс
 - предотвращает переобучение
 - коэффициенты смеси (веса) можно подбирать по лидерборду
-  Минусы
 - довольно простой класс отображений

Стекинг



Стекинг

-  Идея: предоставим создание ансамбля алгоритмам ML
-  Плюсы
 - Нелинейные отображения (RF / GBM / NN)
 - Один из самых эффективных способов
-  Минусы
 - Самый трудоемкий
 - Легко переобучиться

Стекинг

-  Нельзя допускать банальных ошибок
 - обучение и валидация в каждый момент не должны пересекаться
-  Если данных было бы много - не было бы проблем
 - их мало, пользуемся техникой аналогичной кроссвалидации
-  Обязателен честный `holdout` при тестировании идей
 - хотя бы простое разбиение 40 / 30 / 30

Как решать соревнования - ещё раз план

1. Изучить задачу - EDA
2. Изучить метрику - что делать с популярными и странными метриками
3. Сделать константный бейзлайн
4. Написать фреймворк
5. Разбить на трейн/тест
6. Генерировать признаки
7. Выбрать и тюнить модели
8. Кодировать средним
9. Ансамблировать

Организация работы - по одиночке и в команде

Что нужно для участия

1. Время!
2. Умение программировать (желание научиться)
3. Машинка (можно арендовать)

Время

- Нужно заранее определиться с целью
 - Закончить в топ-5
 - Закончить в топ-10%
 - Поучиться у других
 - Попробовать новые задачи, подходы, библиотеки
- Сколько времени у DS уходит на решение задачи в индустрии или на проведение ресерча для статьи?

Умение программировать или желание научиться

- Bash для некоторых библиотек и работы с машинкой
- Python, R, Matlab, другие языки
- Для Python: ipython, jupyter notebook, numpy, pandas, matplotlib, scikit-learn, lightgbm, многие другие

Машинка

- ■ Удобно решать с ноутбука с 16+ гб RAM, core i5+, SSD■ Можно арендовать сервер на aws, google cloud, microsoft azure, selectel, многие другие
- ■ Больше датасеты => требуется больше CPU, RAM и/или SSD■ Больше интенсивных вычислений (фичи и модели) => CPU■ Соревнования с картинками/текстами => нужны GPU и SSD

Эмпирические советы

1. Как можно раньше проверьте, что ваша валидация корректна
2. Выписывайте свои идеи
3. Читайте форум
4. Изучайте код других участников

Воспроизводимость решений

- Наиболее важно
 1. random_state
 2. git
 3. submission files
- Советы
 - Удобно хранить код всех сабмитов (git)
 - Удобно хранить файлики с сабмитами (filesystem, git lfs, cloud storage...)

Режимы работы

Сделайте два режима работы своего ноутбука/скрипта:
validation и test, например

```
X = pd.read_csv('data/train.csv')
y = pd.read_csv('data/y_train.csv')

MODE = 'validation'
if MODE == 'validation':
    x_train, x_test, y_train, y_test = train_test_split(X, y, ...)
elif MODE == 'test':
    x_train = X
    y_train = y
    x_test = pd.read_csv('data/test.csv')
```

Random_state

1. `train_test_split(random_state=0)`
2. `RandomForestClassifier(random_state=0)`
3. `np.random.seed(0); np.random.choice(...)`

Submissions



1. Организаторы могут попросить воспроизвести финальный сабмит
2. Вам может понадобиться воспроизвести какой-то свой сабмит
3. Вы можете захотеть смешать сделанные вами сабмиты

Регулярные коммиты – в конце вашего ноутбука...

```
COMPETITION = 'kaggle.com/...'
```

```
FILE = 'submissions/_LGBM_NN.csv'  
MESSAGE = 'LGBM-super-stacked + NN averaged 10 fold'  
submit.to_csv(FILE)
```

```
%%bash  
git pull  
git add StackingLGBM.ipynb  
git commit -m '{MESSAGE}'  
git push  
kaggle competitions submit \  
-c {COMPETITION} -f {FILE} -m {MESSAGE}
```

<https://www.kaggle.com/docs/api>

Работа в команде

-  Плюсы
 - Быстрее всего учиться новому у других или вместе с другими
 - Больше людей => больше шансов выиграть
-  Минусы
 - Члены команды могут перестать что-то делать
 - Могут возникнуть другие разногласия
-  Удалять участников из команды на kaggle нельзя

Что нужно для успешной работы в команде



1. Планировать работу
2. Обмениваться идеями
3. Обмениваться кодом (git)
4. Обмениваться сабмитами (git / cloud storage) и смешивать ваши решения

Как обычно идет работа в команде



1. Формируется общая валидационная выборка
2. Все готовят свои решения: по-своему обрабатывают данные, генерят фичи и учат модели
3. Решения участников регулярно смешиваются

Планирование работы и обмен идеями



1. Планировать заранее - правильно
2. Нужно регулярно синхронизироваться
3. Созвоны - для синков, чат - для обсуждений
4. Нужно быть готовым объединять ваши решения
5. Кто-то должен (нести ответственность за работу команды)

Обмен кодом



1. Заведите приватный репозиторий и дайте доступы
2. Комитьте туда код регулярно

Смешивание решений



1. Выделите валидационную выборку вначале! (и перепроверьте)
2. Напишите скрипт/ноутбук, который будет смешивать предсказания
3. Делайте это регулярно, чтобы понимать ваш прогресс

Проверка валидационной выборки

День 1:

А: сделал валидацию, `train_test_split(test_size=0.2, random_state=1)`
Б: отлично, го

День 31:

А: Перепроверь, у тебя `test_ids [5821, 2440, 2143, ...]`?
Б: `[1734, 2314, 4325, ...]`
А: :'(

Частые причины:

1. Разные версии python и библиотек
2. Разный порядок строк в разбиваемом датафрейме

Смешивание решений



- Напишите скрипт/ноутбук, который может подобрать по валидационной выборке способ смешивания ваших моделей.
- Чтобы смешивать не об лидерборд, вам нужно сохранять прогнозы (иногда вероятности!) для валидации (1) и теста (2)
- Внизу ноутбука – тот код, который делает коммит и сабмит.

Смешивание решений

-  Шаги по усложнению:
 1. Усреднение
 2. Усреднение с весами (== блендинг линейной моделью)
 3. Стекинг
-  Делайте смешивание регулярно
 1. Чтобы понимать, есть ли от этого толк
 2. Чтобы тратить время более перспективно

Соревнования не только про ML. Ещё про то, как



1. Заставить код работать
2. Распарсить данные в непонятных форматах
3. Работать с датасетом, который не влезает в память
4. Найти в данных баг, который пропустили организаторы
5. Использовать алгоритмы, которые вы не понимаете
6. Как ускорить ваши вычисления
7. Просмотреть глазками большое количество данных

Но и реальная жизнь тоже!