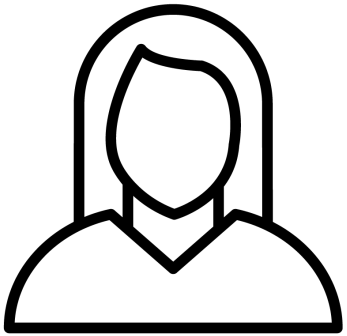
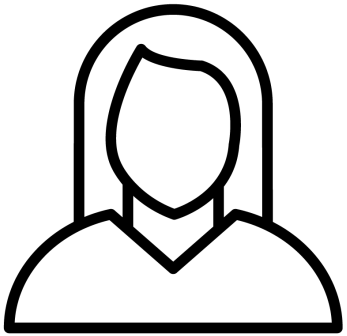
Ищем релевантные признаки из сотен источников для любой модели

Валерия Дымбицкая
Upgini



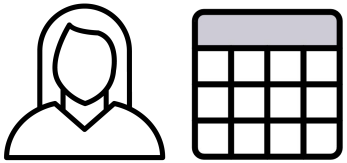
Это Даша.



Это Даша.
Даша хочет найти фичи.

Больше фичей – лучше…

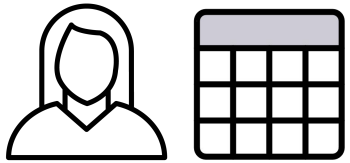
…давайте поможем Даше их найти!



Даша + эталон (выборка) = …

Больше фичей – лучше…

…давайте поможем Даше их найти!

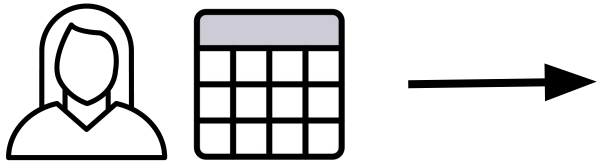


- Google Dataset Search
- Dateno
- etc

страна
формат
описание
владелец
…
имена колонок
типы колонок

Больше фичей – лучше…

…давайте поможем Даше их найти!

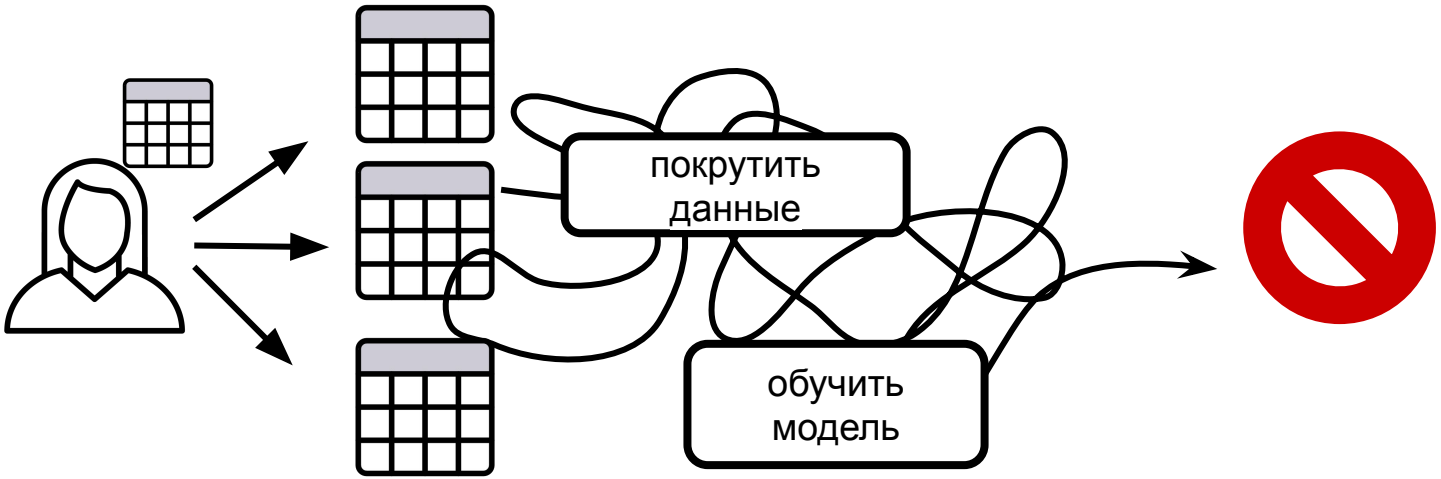


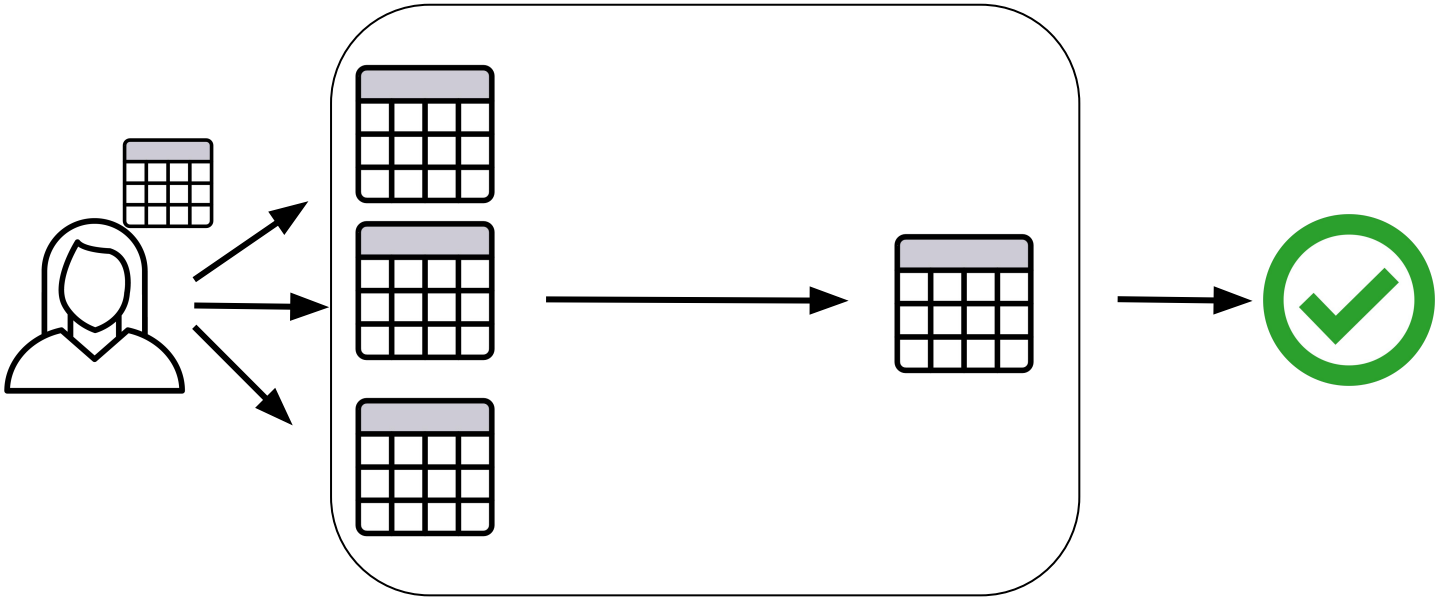
- Google Dataset Search
- Dateno
- etc

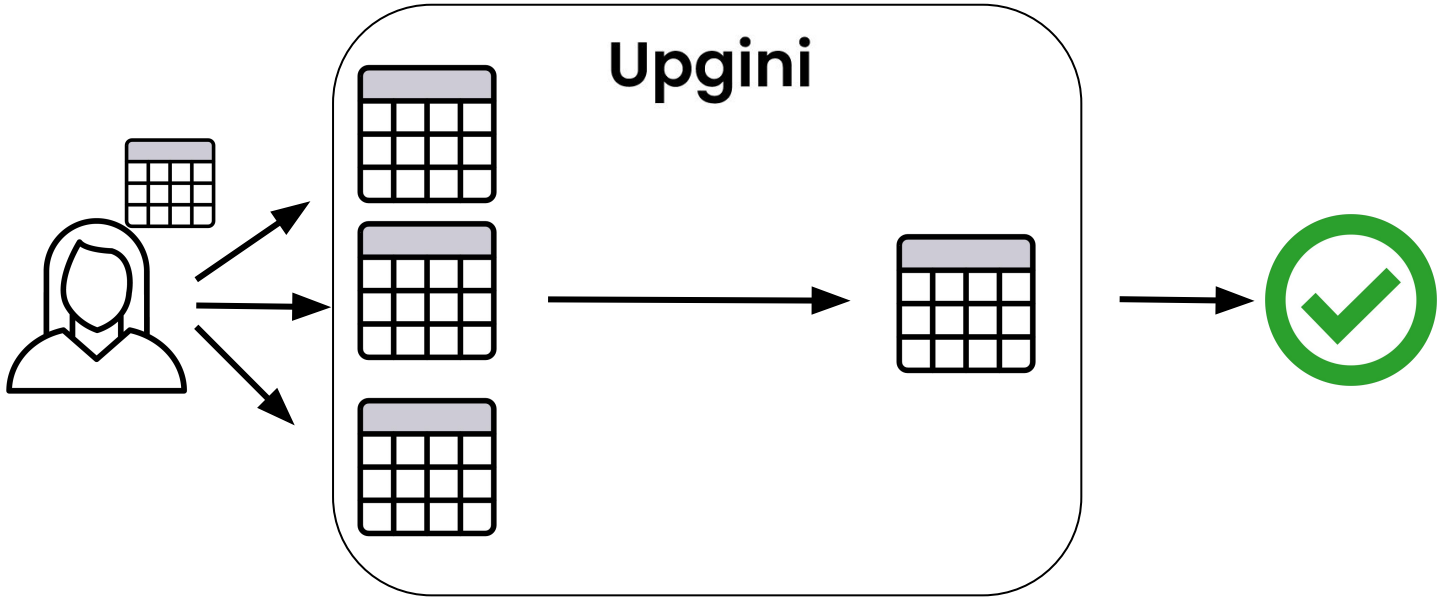
страна
формат
описание
владелец
…
имена колонок
типы колонок

Чего не хватает?

понимания того, будут ли они полезны


покрутить
данные
обучить
модель



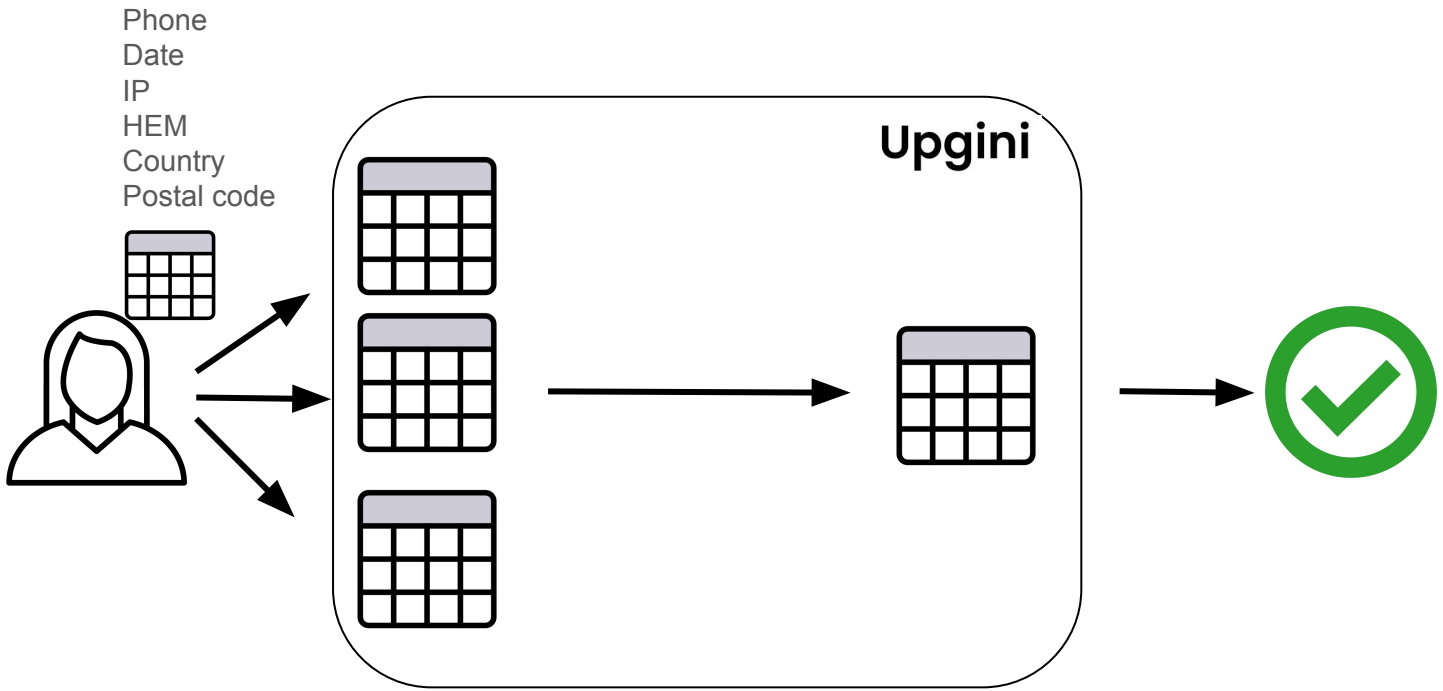

Upgini

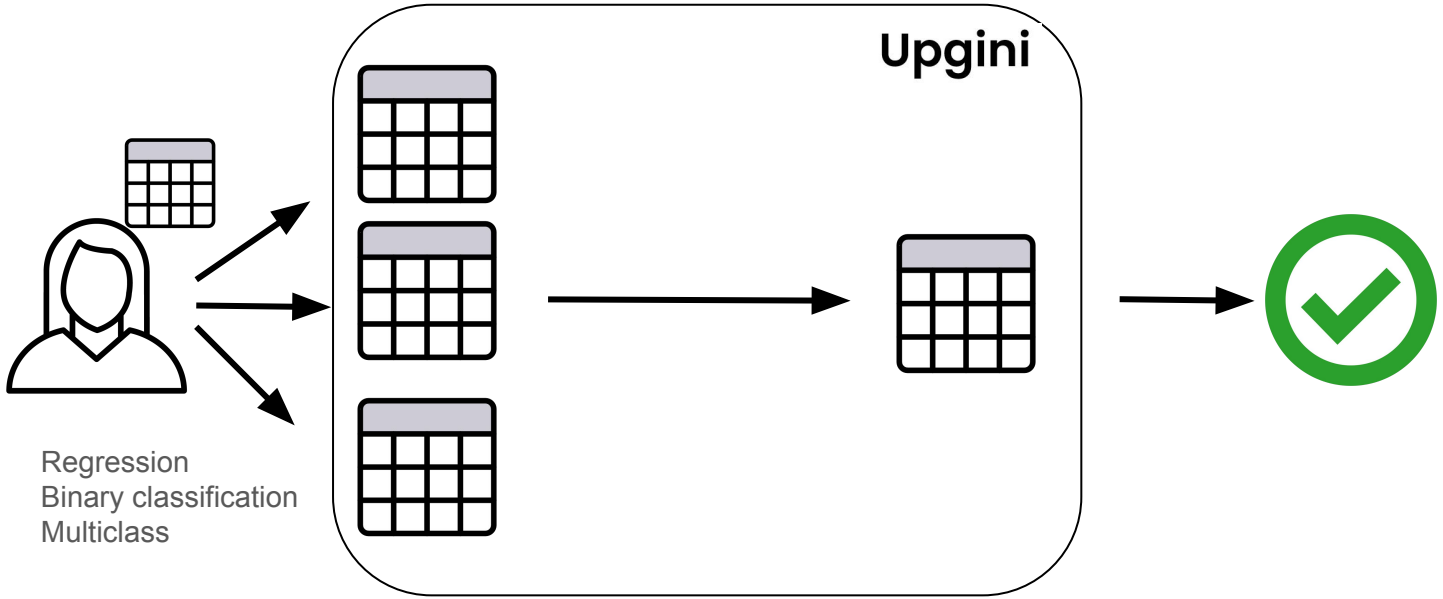
Upgini

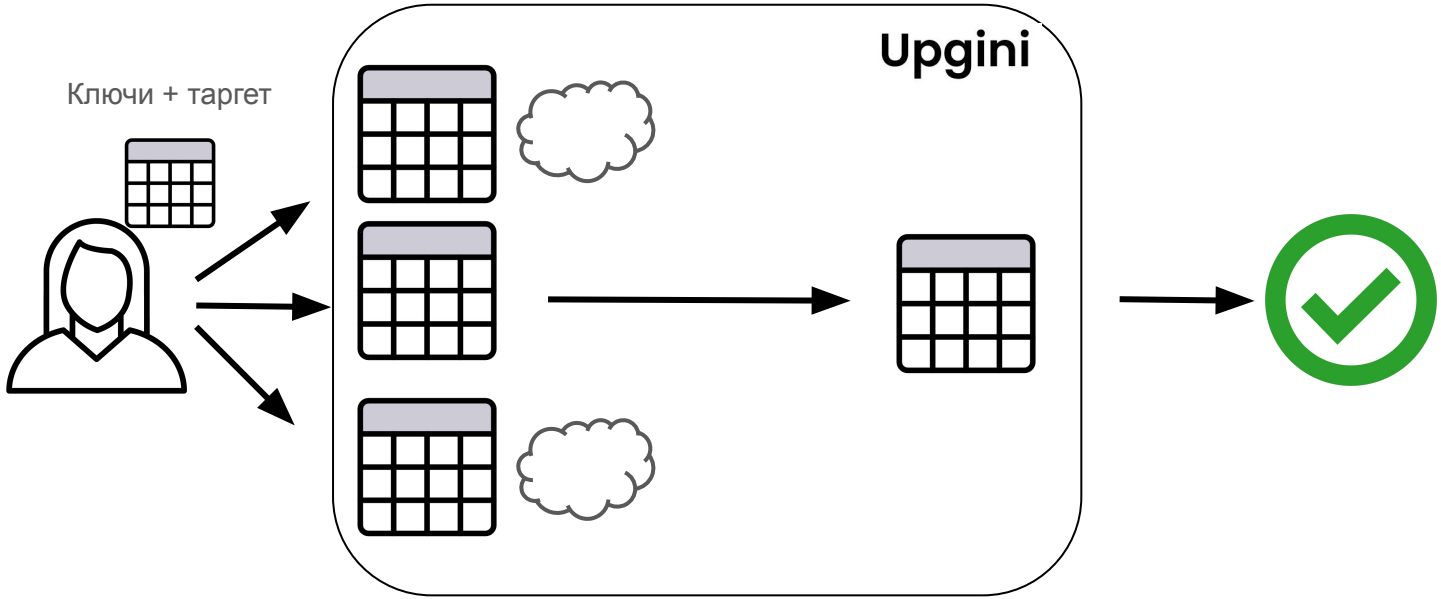
%pip install upgini

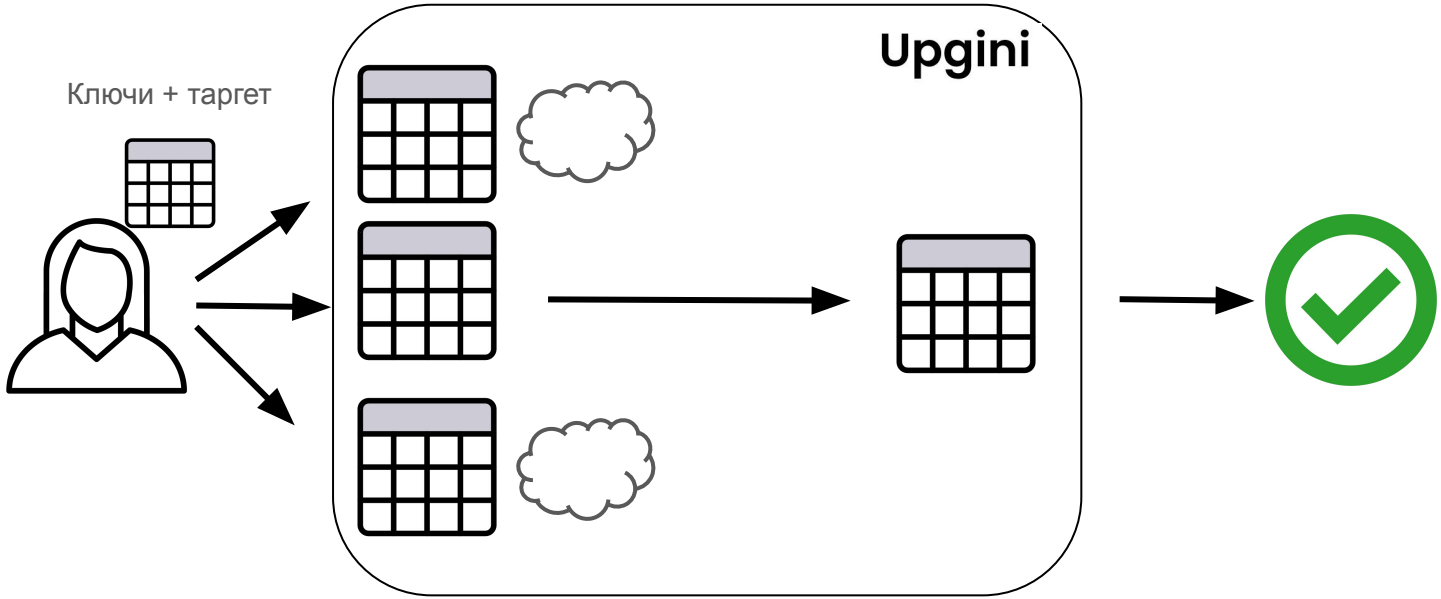
```
df = pd.read_csv(df_path)
train_features = df.drop(['avg_salary'], axis=1)
train_target = df.avg_salary

enricher = FeaturesEnricher(
    search_keys={
    'country': SearchKey.COUNTRY,
    'Postal_code': SearchKey.POSTAL_CODE})
enricher.fit(train_features, train_target, scoring = "mean_absolute_error")
```

https://github.com/upgini/upgini

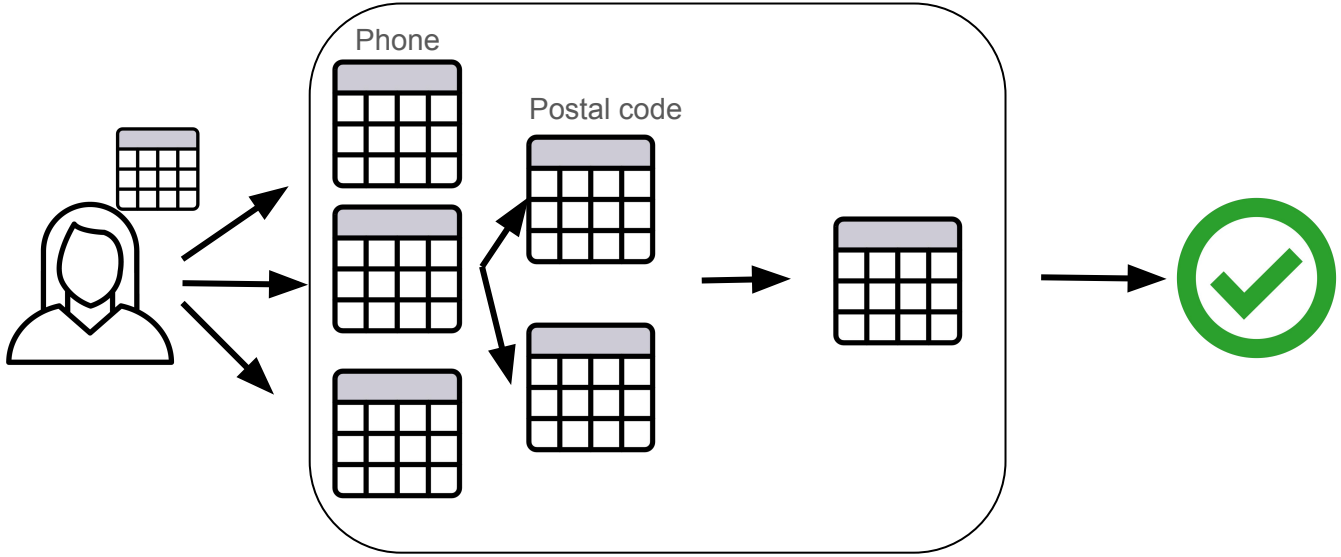

Phone
Date
IP
HEM
Country
Postal code
Upgini


Upgini
Regression
Binary classification
Multiclass

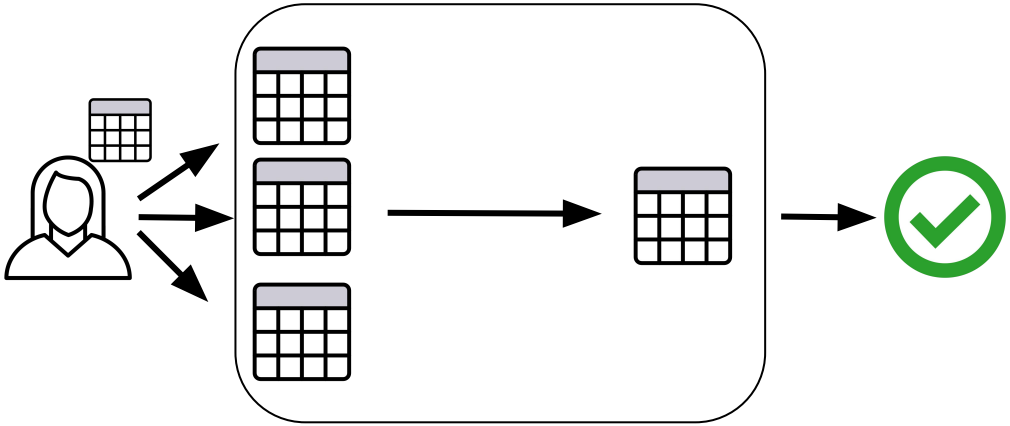

Ключи + таргет
Upgini



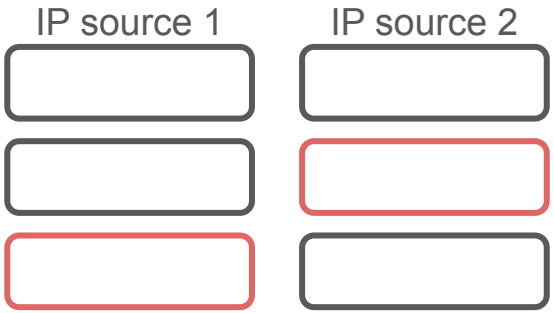
Федеративный поиск **по эталону**



В других датасетах можно найти
дополнительные ключи



Ансамбль данных:
разные источники
дополняют и исправляют
друг друга

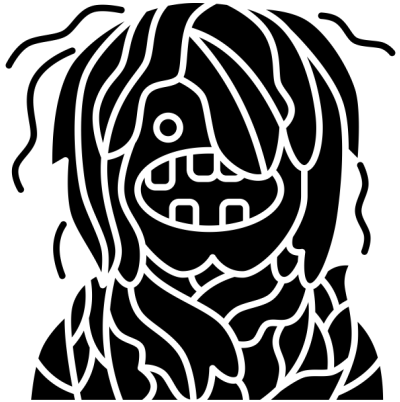


Ура, у нас есть поиск!

- Ищем не по метаданным, а по эталону
- Комбинируем источники в ансамбли

Ура, у нас есть поиск!

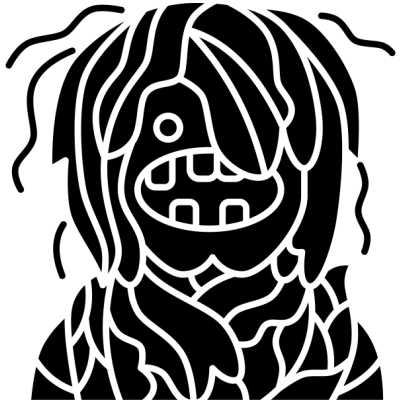
- Ищем не по метаданным, а по эталону
- Комбинируем источники в ансамбли



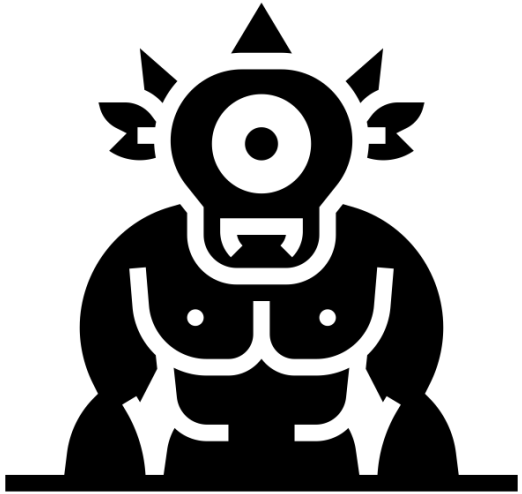
Сырые данные
не работают в модели

Ура, у нас есть поиск!

- Ищем не по метаданным, а по эталону
- Комбинируем источники в ансамбли



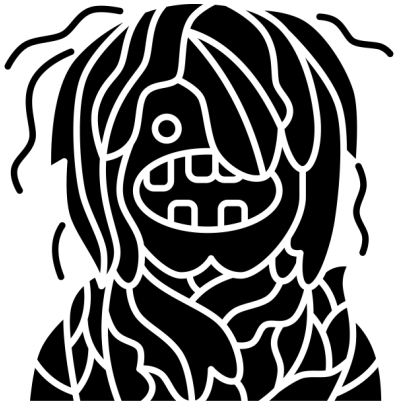
Сырые данные
не работают в модели



Проклятие размерности

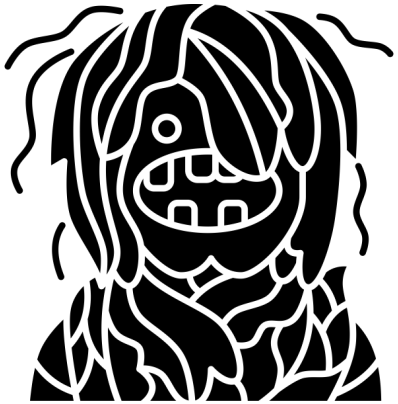
Готовим данные для поиска

- Очистка
- Нормализация
- Feature Engineering



Готовим данные для поиска

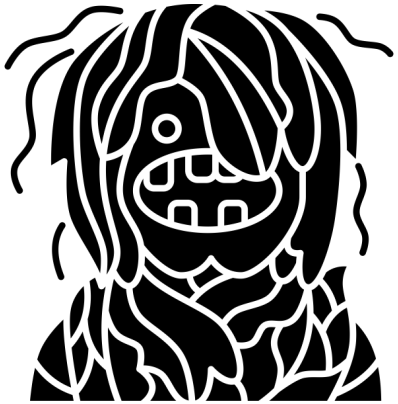
- Очистка
- Нормализация
- **Feature Engineering – самая сложная часть!**



Готовим данные для поиска

- **Feature Engineering**

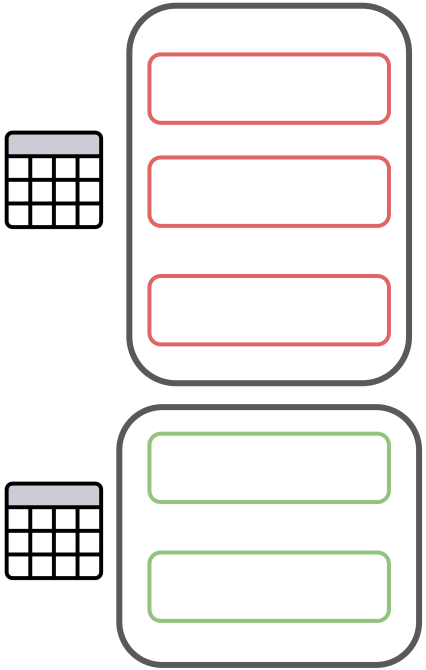
Featuretools

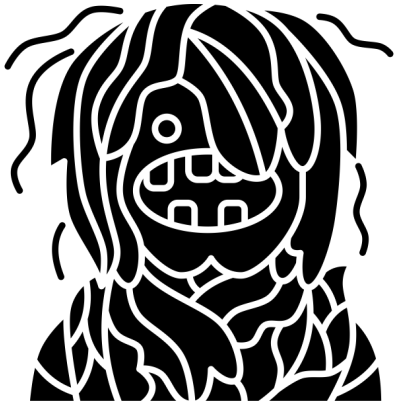


autofeat

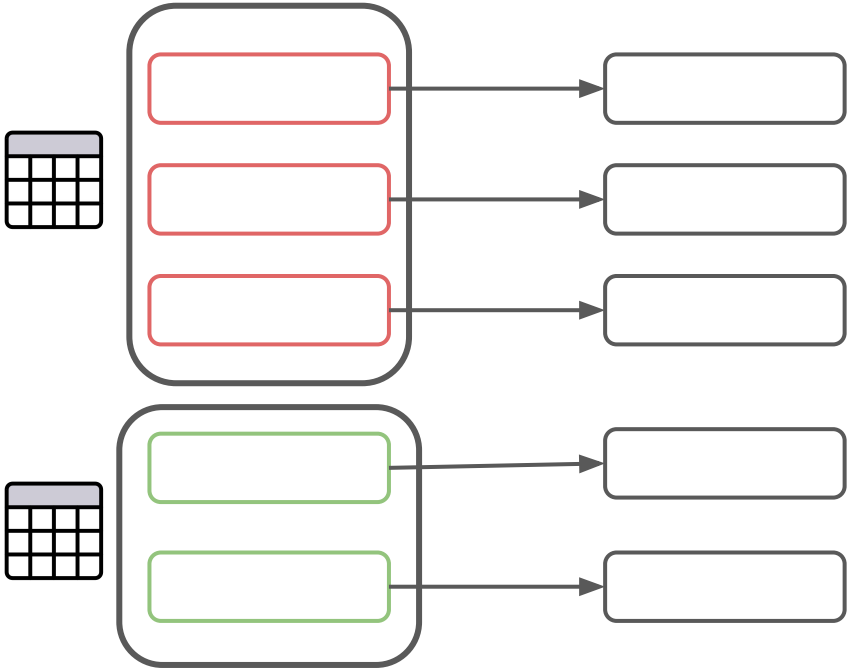
OpenFE

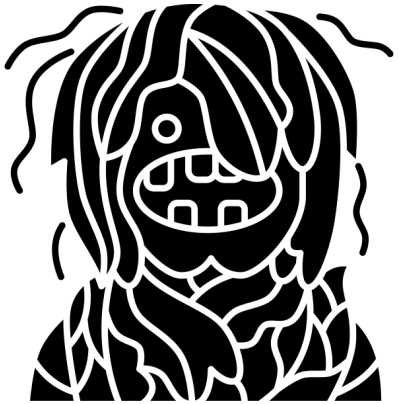
Табличный Feature Engineering



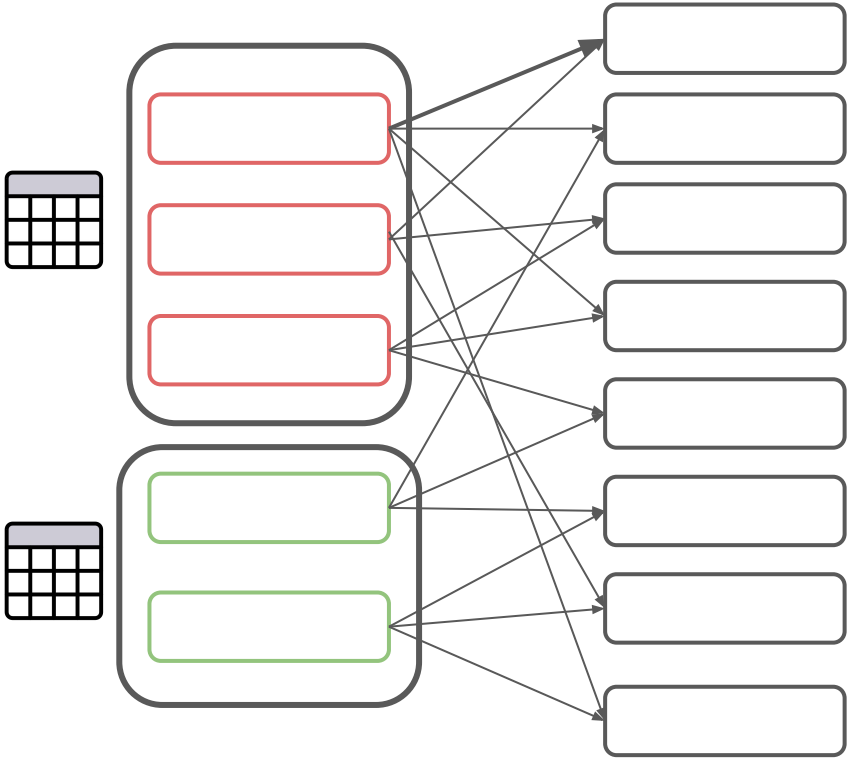


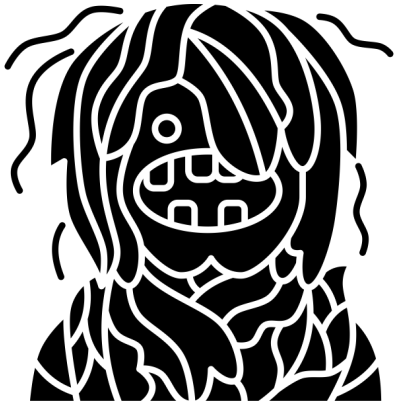
Табличный Feature Engineering



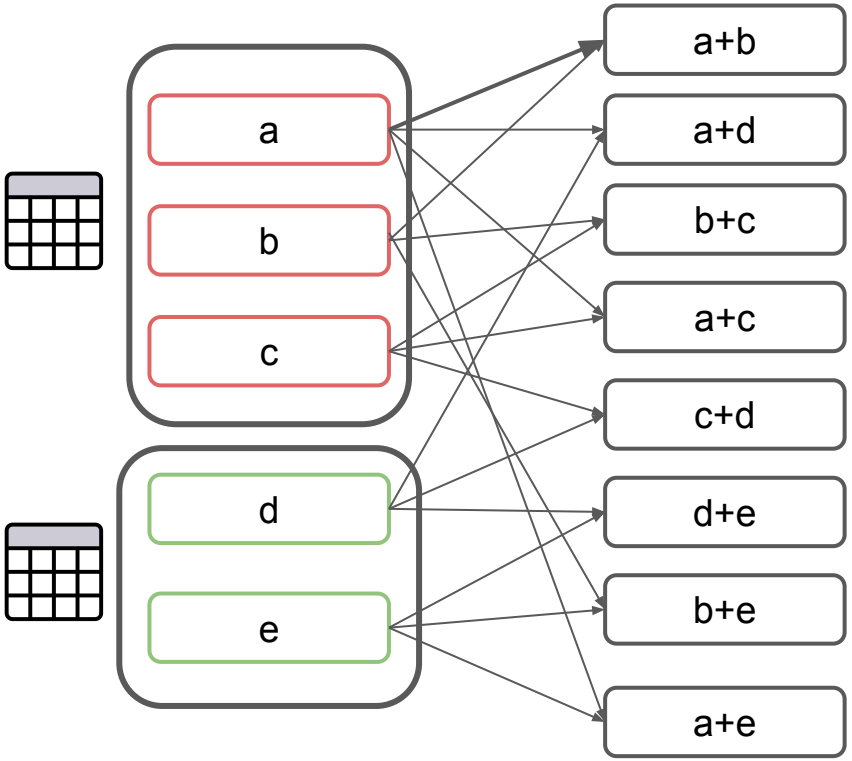


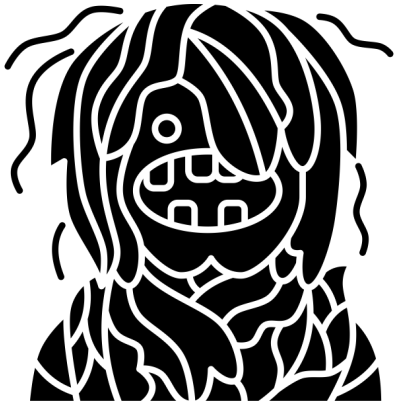
Табличный Feature Engineering



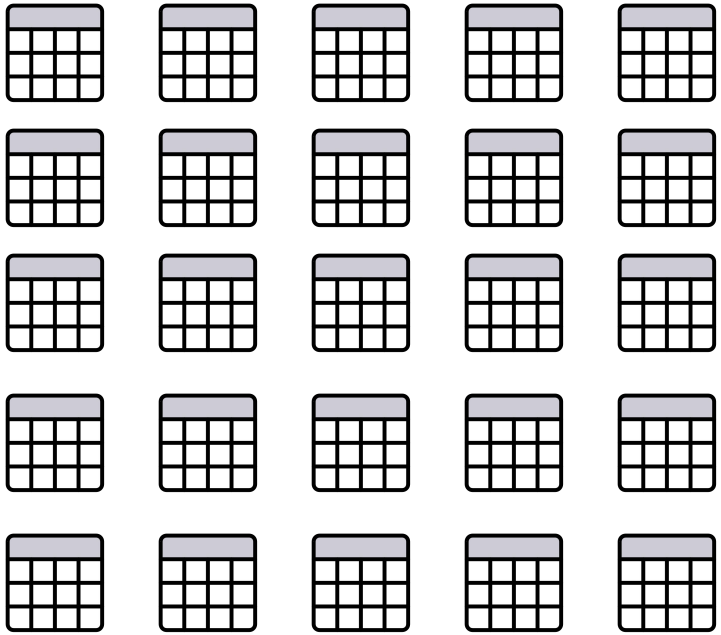


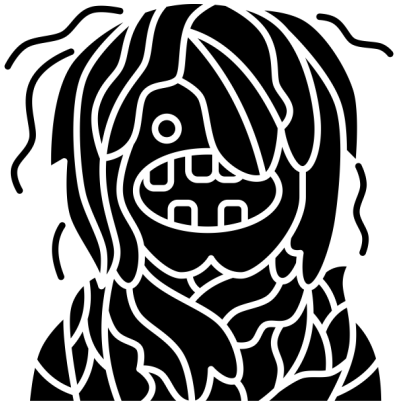
Табличный Feature Engineering



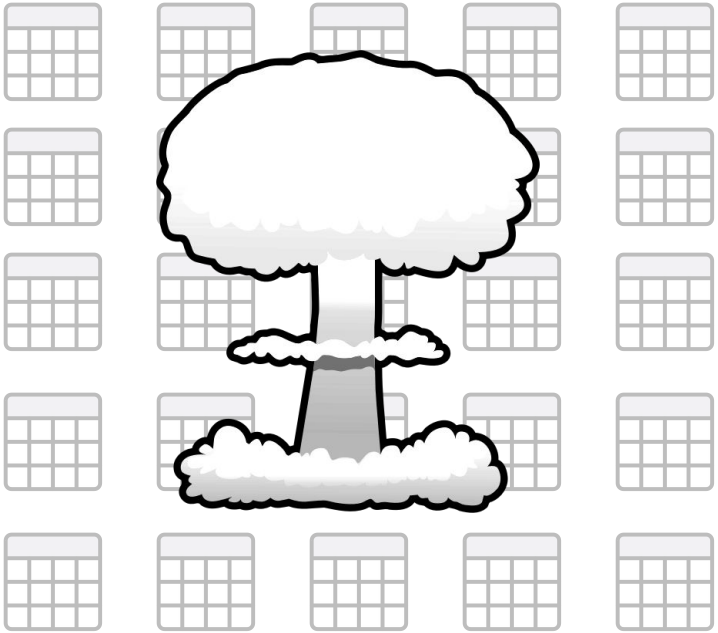


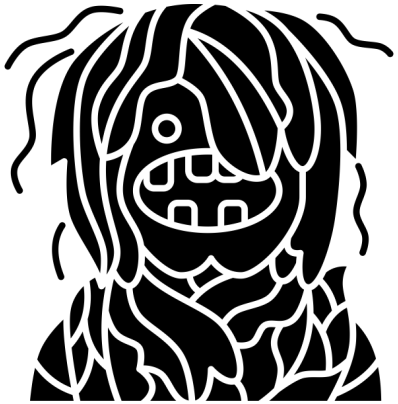
Табличный Feature Engineering





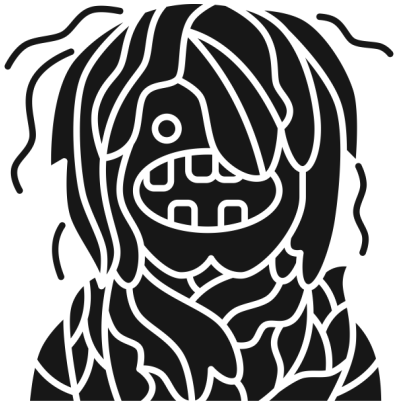
Табличный Feature Engineering





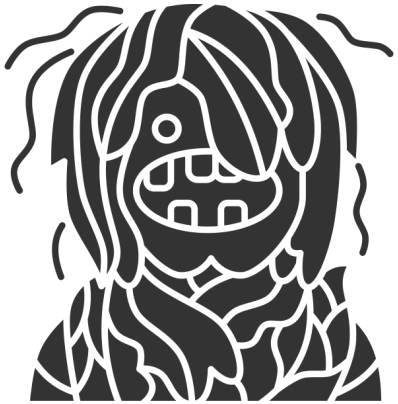
Табличный Feature Engineering

- Вычисляем для фичей конкретного поиска



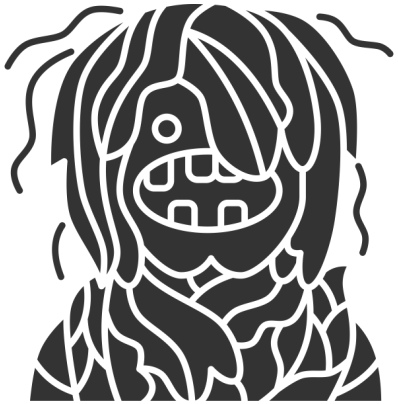
Табличный Feature Engineering

- Вычисляем для фичей конкретного поиска
- Только подмножество фичей и комбинаций



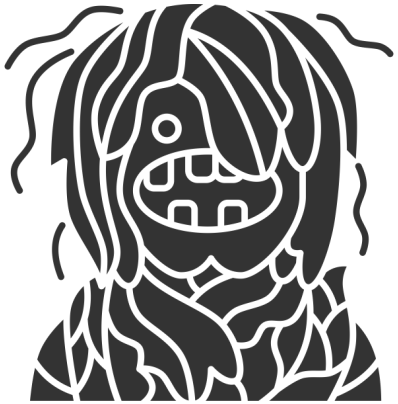
Табличный Feature Engineering

- Вычисляем для фичей конкретного поиска
- Только подмножество фичей и комбинаций
  - Простые правила



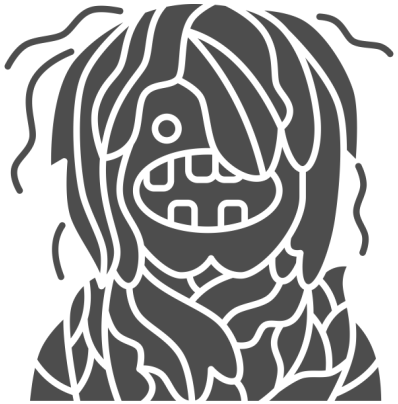
Табличный Feature Engineering

- Вычисляем для фичей конкретного поиска
- Только подмножество фичей и комбинаций
  - Простые правила
  - Только важные фичи



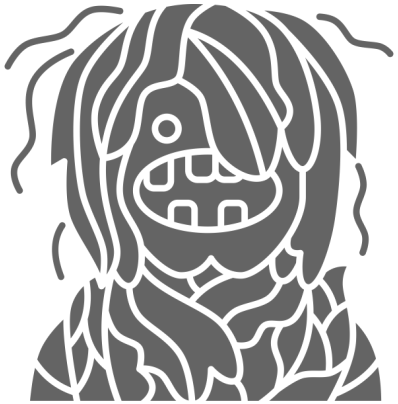
Табличный Feature Engineering

- Вычисляем для фичей конкретного поиска
- Только подмножество фичей и комбинаций
- Операнды должны иметь смысл

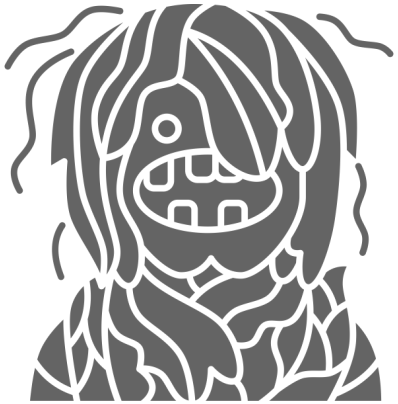


Табличный Feature Engineering

- Вычисляем для фичей конкретного поиска
- Только подмножество фичей и комбинаций
- Операнды должны иметь смысл
    - abs, log, floor, /, sigmoid, sim, date_diff, etc

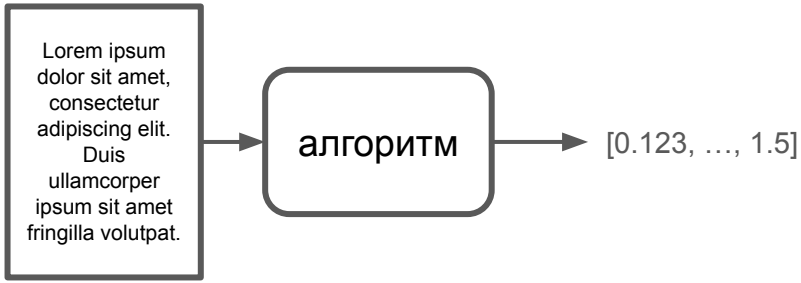


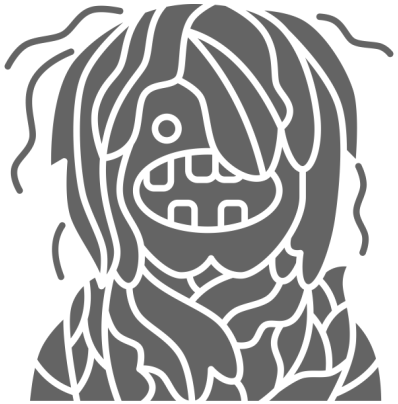
Фичи на текстах



Фичи на текстах

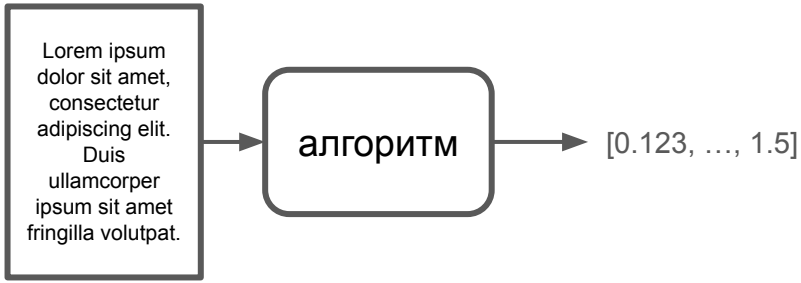
- Преобразование текста в вектор

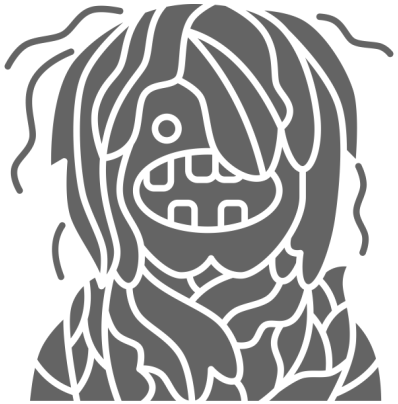




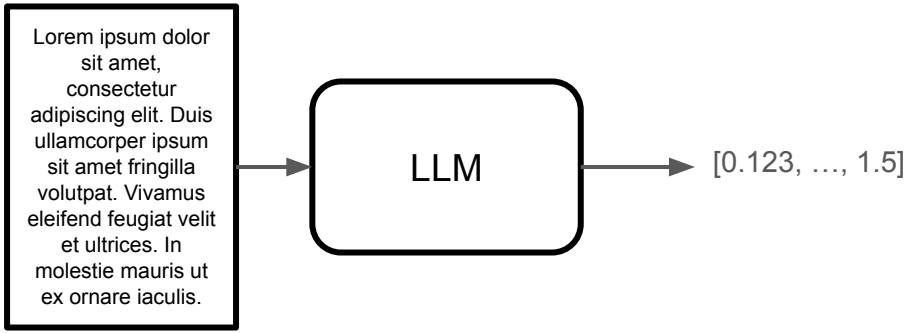
Фичи на текстах

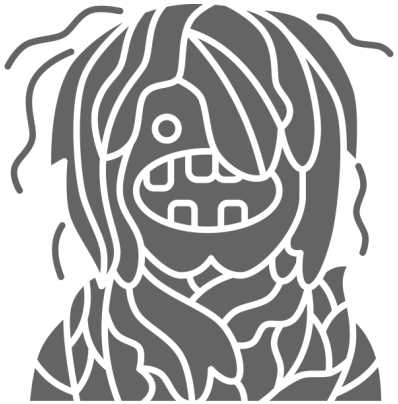
- Преобразование текста в вектор
  - TF-IDF, word2vec, Fasttext, эмбеддинги LLM, etc



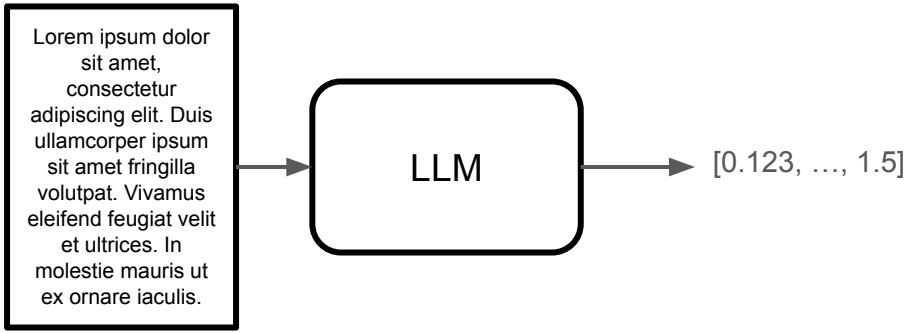


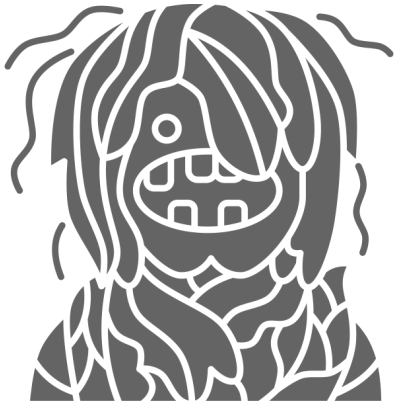
Фичи на текстах





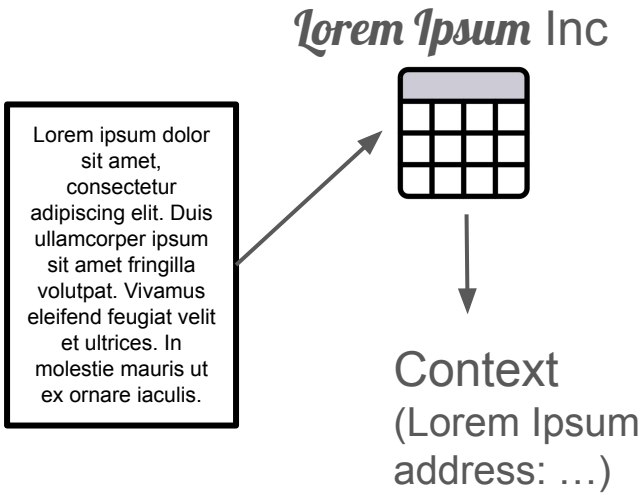
Фичи на текстах

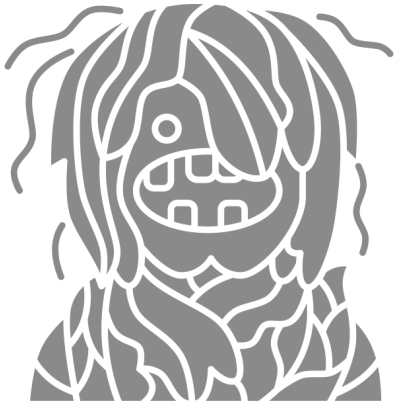




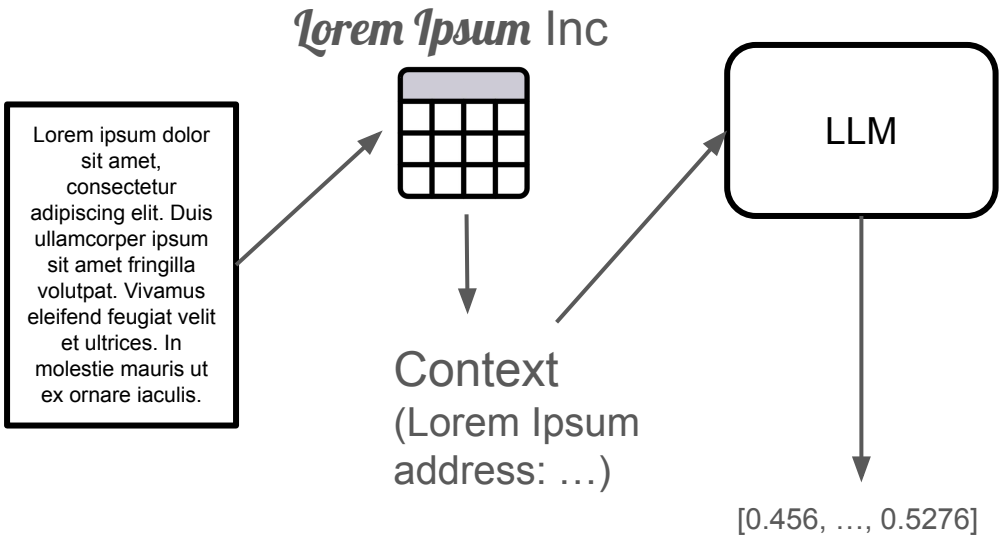
Lorem Ipsum
AKA
Челябинский трубопрокатный завод

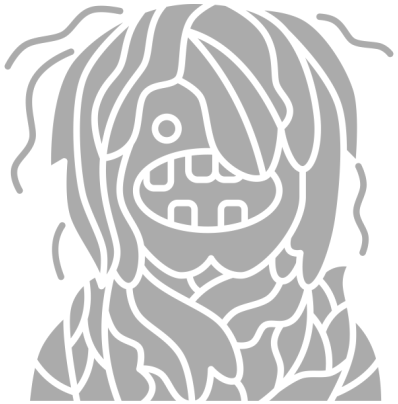
Фичи на текстах



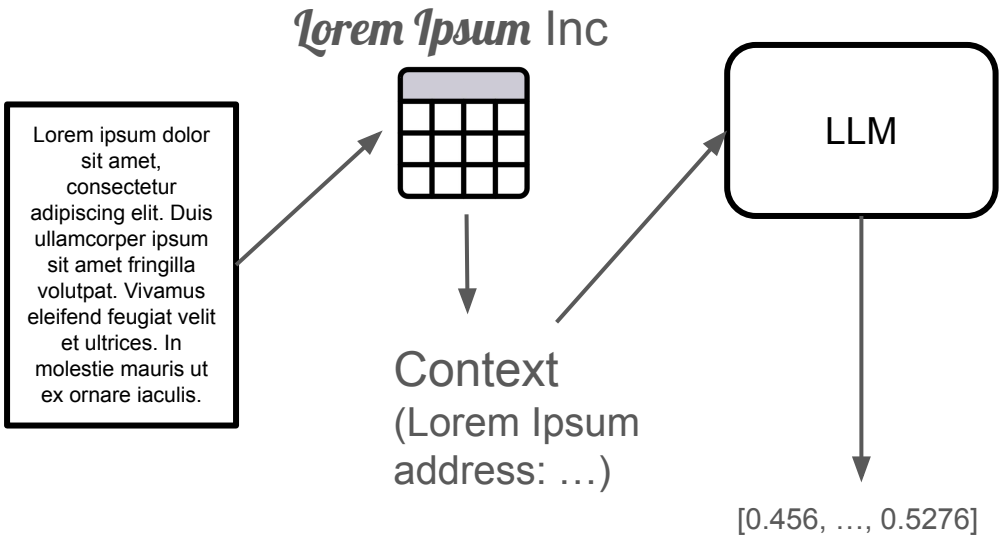


Фичи на текстах





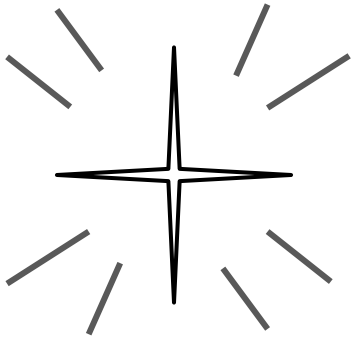
Фичи на текстах



Retrieval Augmented Generation (RAG)

Бенчмарк: https://medium.com/p/2d67cf595b9c

Умеем готовить фичи!



Умеем готовить фичи!

- Преобразовали текстовые данные в наилучший набор эмбеддингов

Умеем готовить фичи!

- Преобразовали текстовые данные в наилучший набор эмбеддингов
- Сгенерировали оптимальный набор комбинаций с помощью ряда эвристик

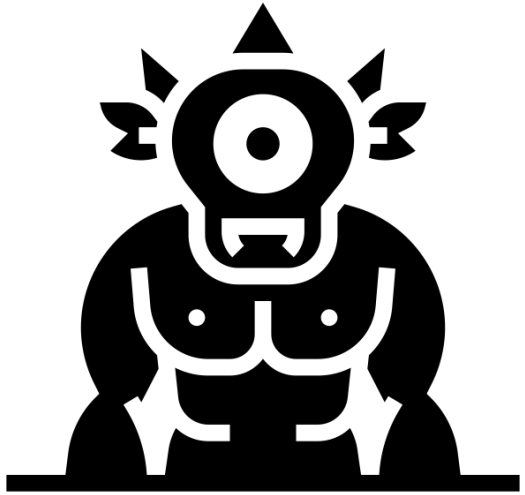
Умеем готовить фичи!

- Преобразовали текстовые данные в наилучший набор эмбеддингов
- Сгенерировали оптимальный набор комбинаций с помощью ряда эвристик

…

Умеем готовить фичи!

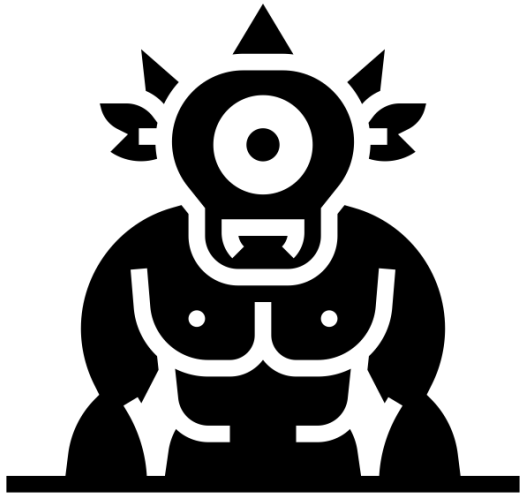
- Преобразовали текстовые данные в наилучший набор эмбеддингов
- Сгенерировали оптимальный набор комбинаций с помощью ряда эвристик



Получили **десятки тысяч** фичей

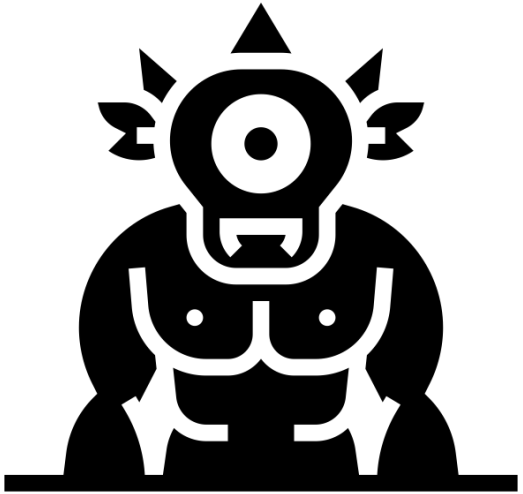
Ранжируем результат поиска

- Поиск фичей – это задача feature selection



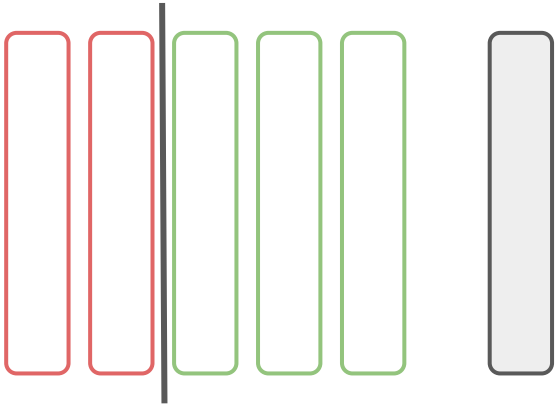
Ранжируем результат поиска

- Поиск фичей – это задача feature selection
    - filter, wrapper, embedded

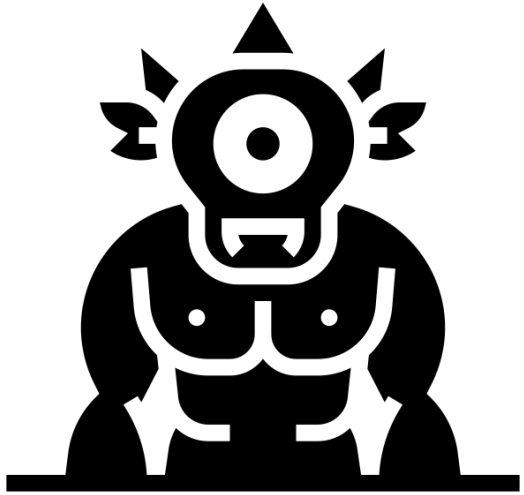


Ранжируем результат поиска

- Поиск фичей – это задача feature selection
    - **filter**, wrapper, embedded

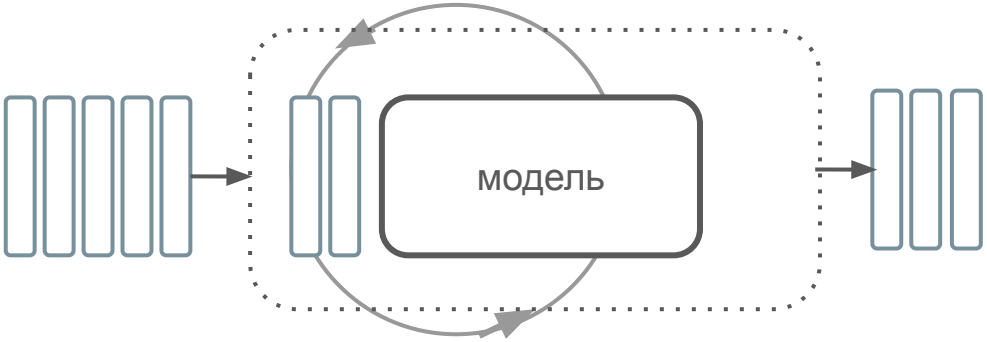


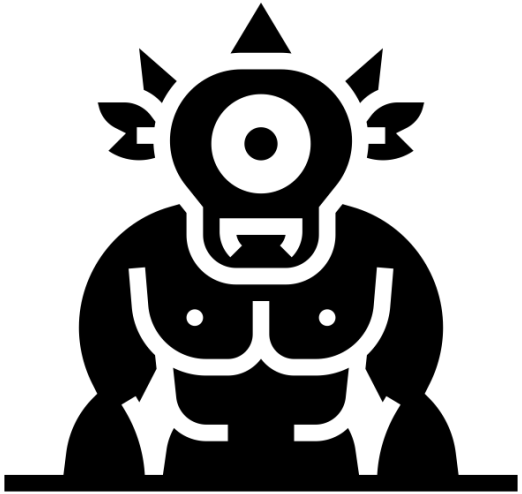
correlation
mutual information



Ранжируем результат поиска

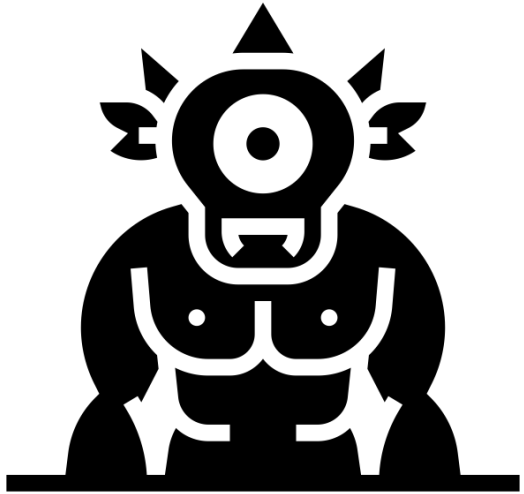
- Поиск фичей – это задача feature selection
    - filter, **wrapper**, embedded





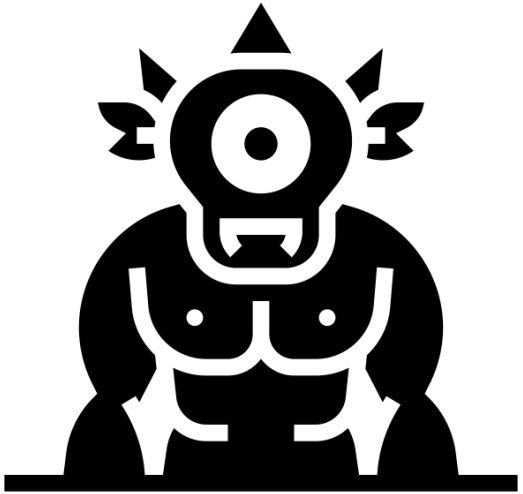
Ранжируем результат поиска

- Поиск фичей – это задача feature selection
    - filter, wrapper, **embedded**



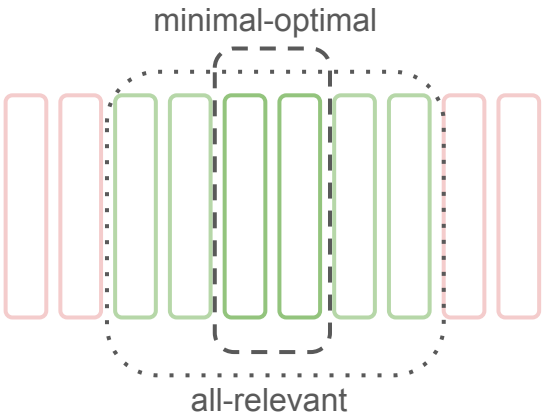
Ранжируем результат поиска

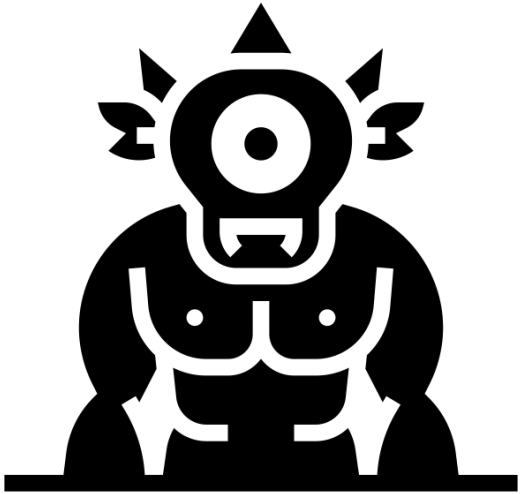
- Поиск фичей – это задача feature selection
    - minimal-optimal vs all-relevant



Ранжируем результат поиска

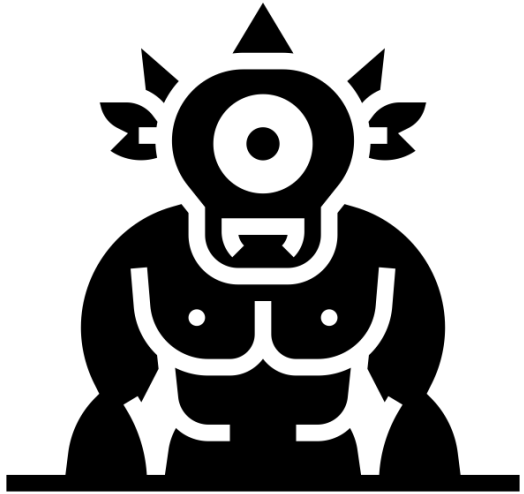
- Поиск фичей – это задача feature selection
    - minimal-optimal vs all-relevant





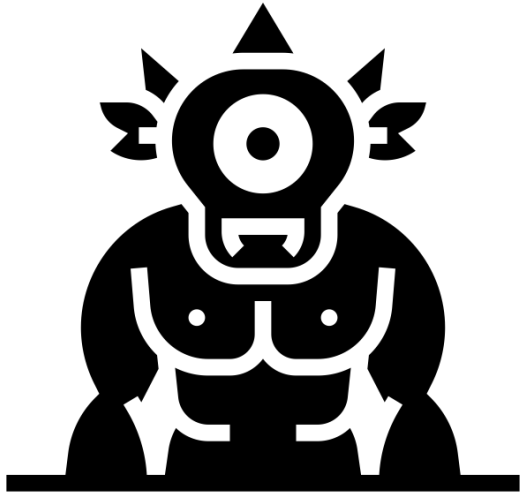
Ранжируем результат поиска

- Minimal-optimal алгоритмы
    - Embedded
    - MRMR



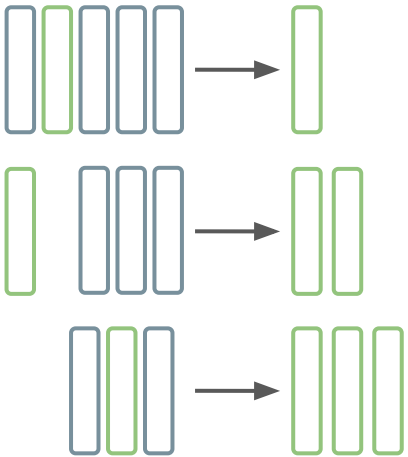
Ранжируем результат поиска

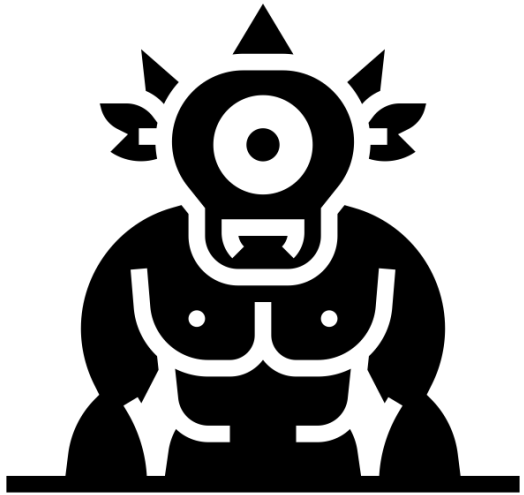
- Minimal-optimal алгоритмы
  - **Embedded**
  - MRMR



Ранжируем результат поиска

- Minimal-optimal алгоритмы
  - Embedded
  - **MRMR**

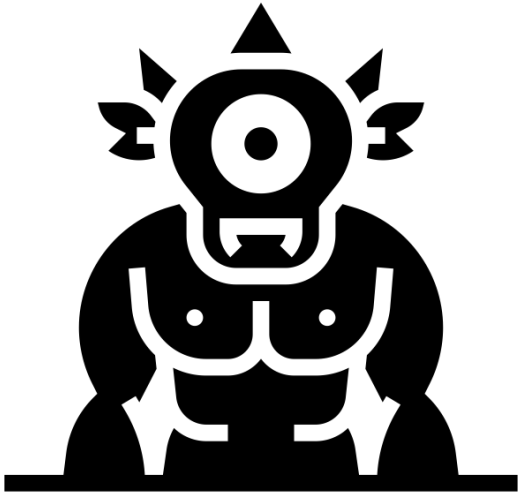




Ранжируем результат поиска

- Minimal-optimal алгоритмы
    - Embedded
    - **MRMR**

$$score_i(f) = \frac{relevance(f \mid target)}{redundancy(f \mid features\ selected\ until\ i-1)}$$

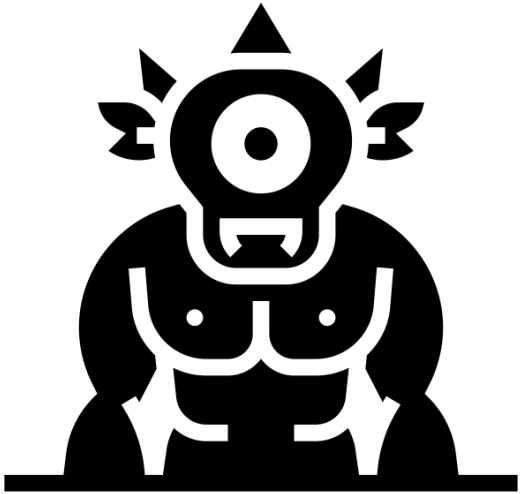


Ранжируем результат поиска

- Minimal-optimal алгоритмы
  - Embedded
  - **MRMR**

$$score_i(f) = \frac{relevance(f \mid target)}{redundancy(f \mid features\ selected\ until\ i-1)}$$

Пример:
relevance = F-score
redundancy = Pearson correlation



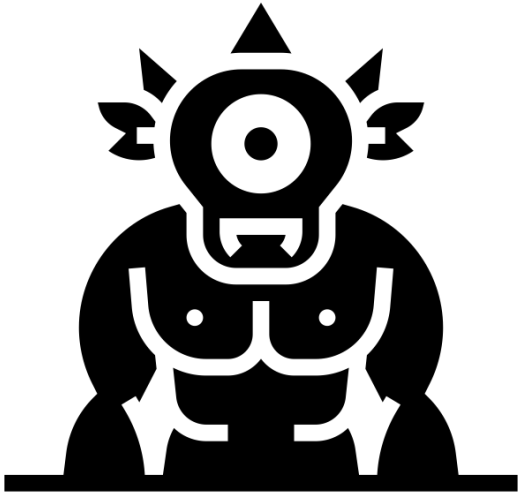
Uber paper (2019): https://arxiv.org/pdf/1908.05376

Ранжируем результат поиска

- Minimal-optimal алгоритмы
  - Embedded
  - **MRMR**

$$score_i(f) = \frac{relevance(f \mid target)}{redundancy(f \mid features\ selected\ until\ i-1)}$$

Сколько брать фичей?



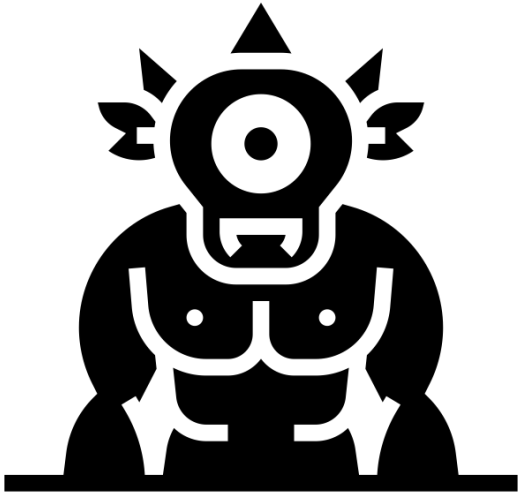
Ранжируем результат поиска

- Minimal-optimal алгоритмы
  - Embedded
  - **MRMR**

$$score_i(f) = \frac{relevance(f \mid target)}{redundancy(f \mid features\ selected\ until\ i-1)}$$

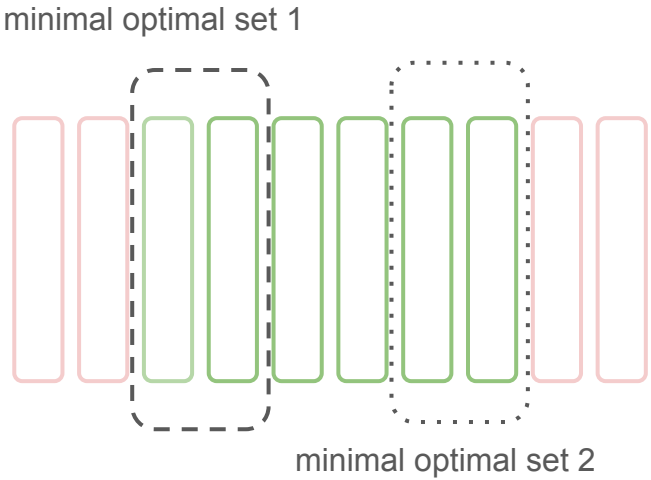
Сколько брать фичей?

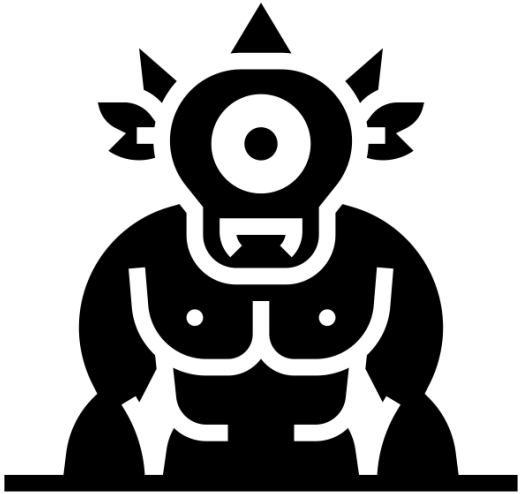
Uber: минимальное количество, при котором будет максимальный AUC по итогам **50** прогонов Random Forest



Ранжируем результат поиска

- Minimal-optimal алгоритмы
  - Embedded
  - MRMR

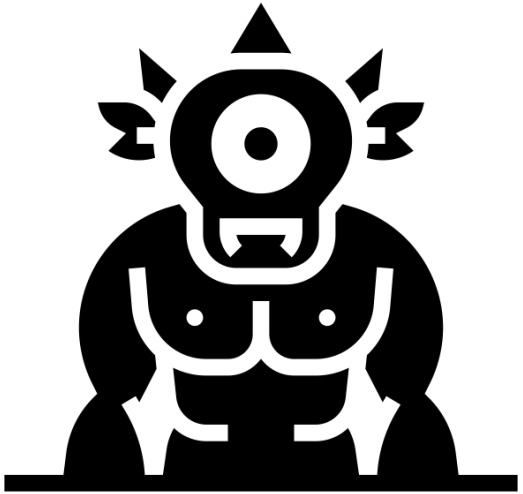




Ранжируем результат поиска

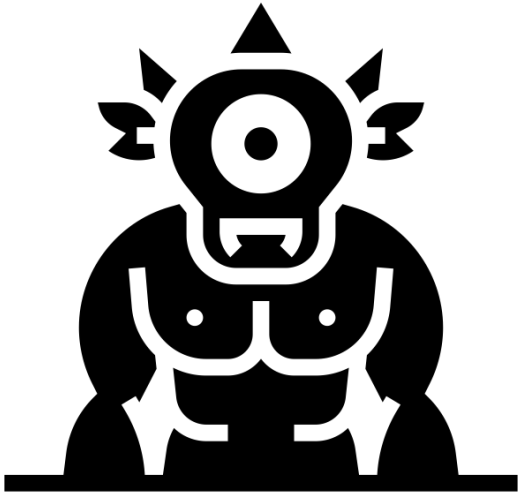
- Minimal-optimal алгоритмы
    - Embedded
    - MRMR

Проблема: мы не знаем конечную модель



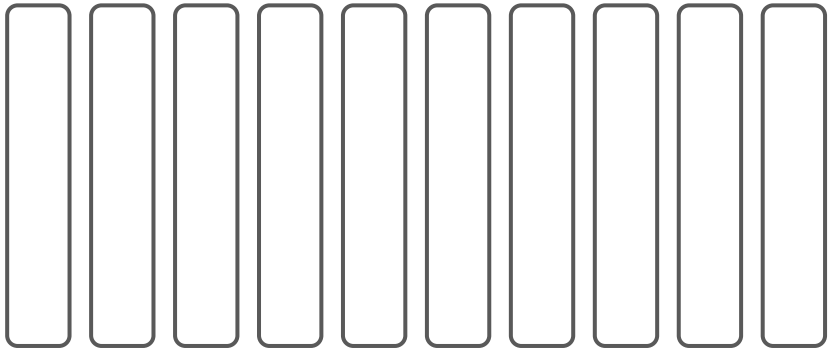
Ранжируем результат поиска

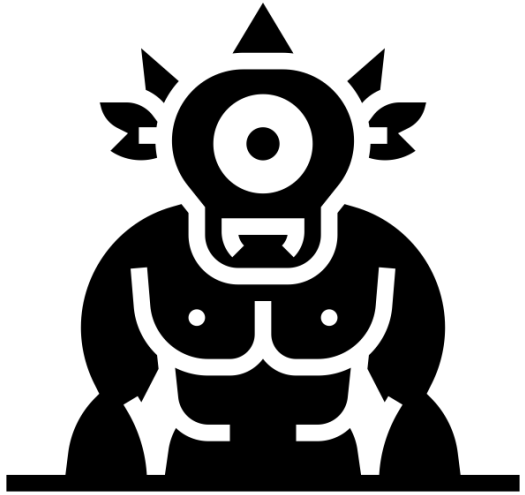
- All-relevant алгоритмы
    - Recursive Feature Elimination
    - Permutation importance selection
    - Boruta



Ранжируем результат поиска

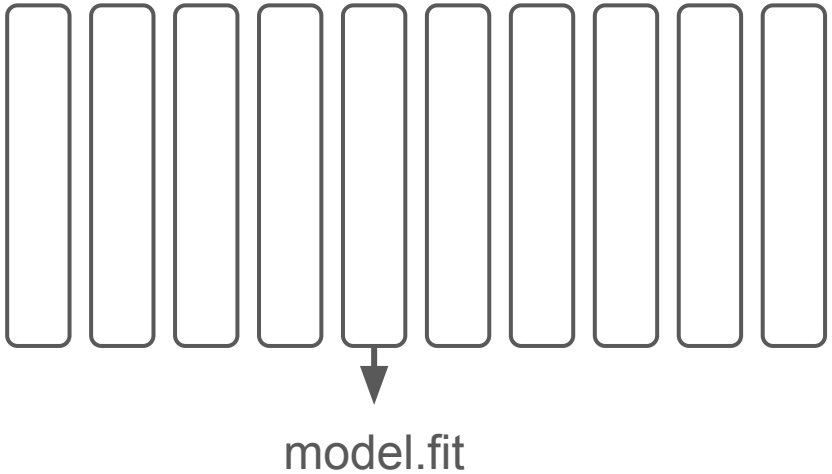
- Recursive Feature Elimination

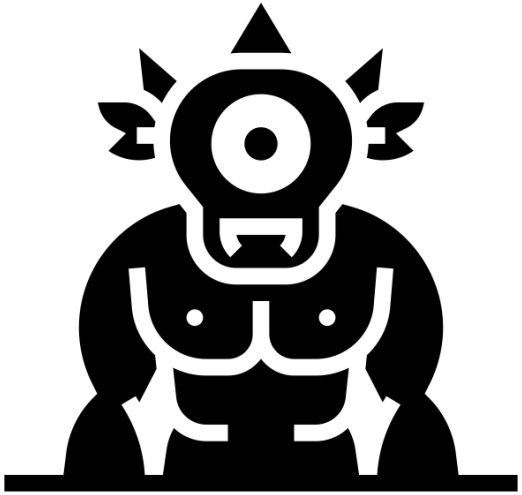




Ранжируем результат поиска

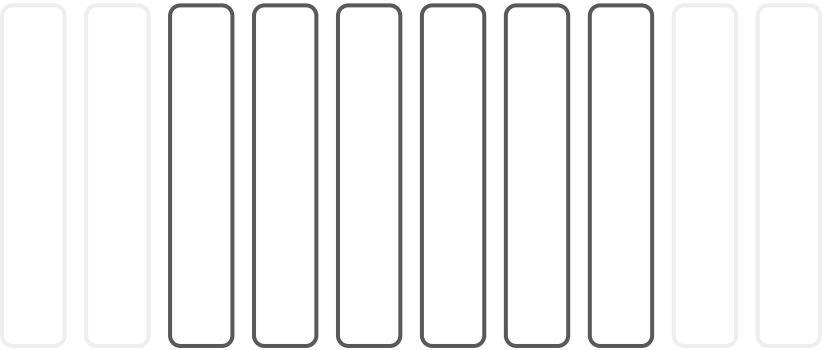
- Recursive Feature Elimination

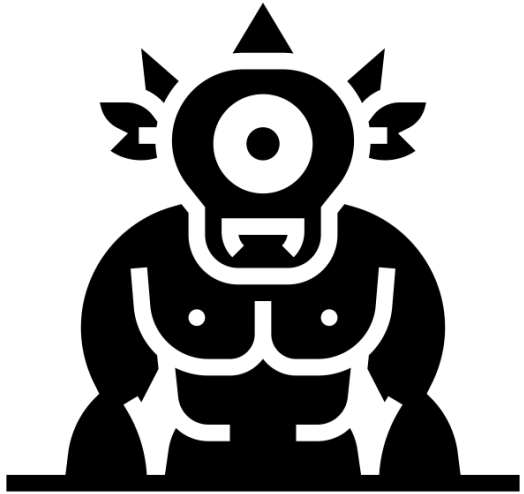




Ранжируем результат поиска

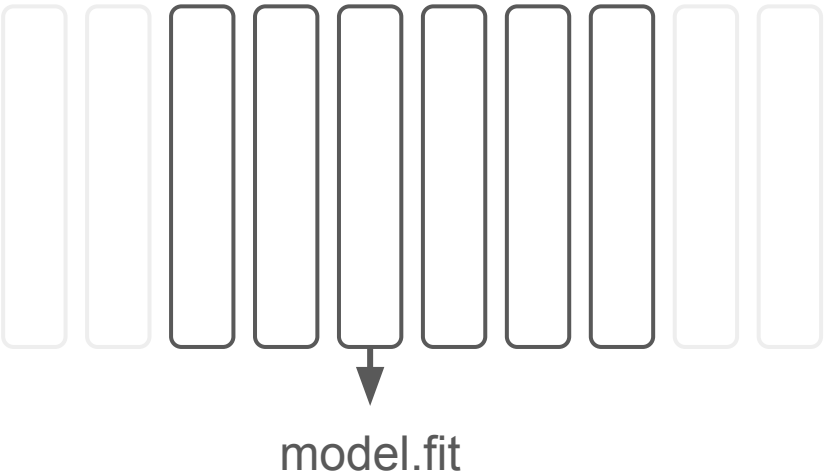
- Recursive Feature Elimination

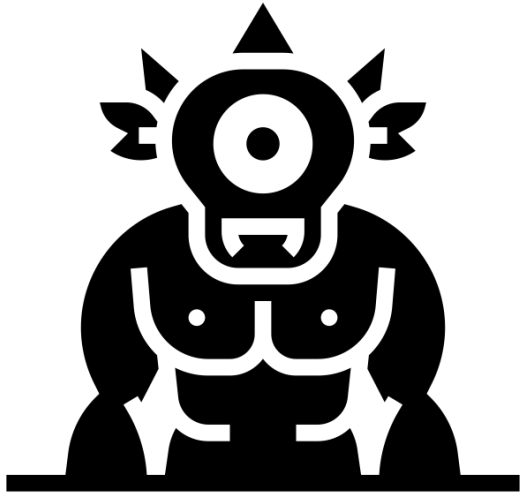




Ранжируем результат поиска

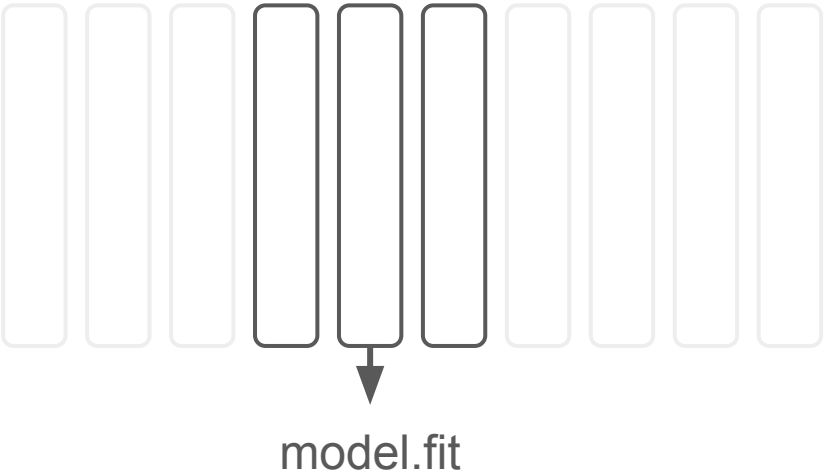
- Recursive Feature Elimination

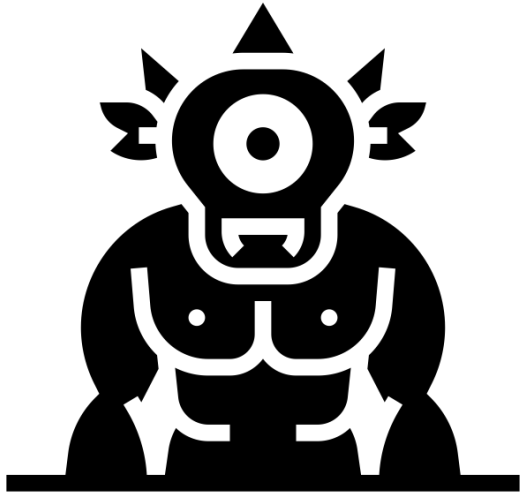




Ранжируем результат поиска

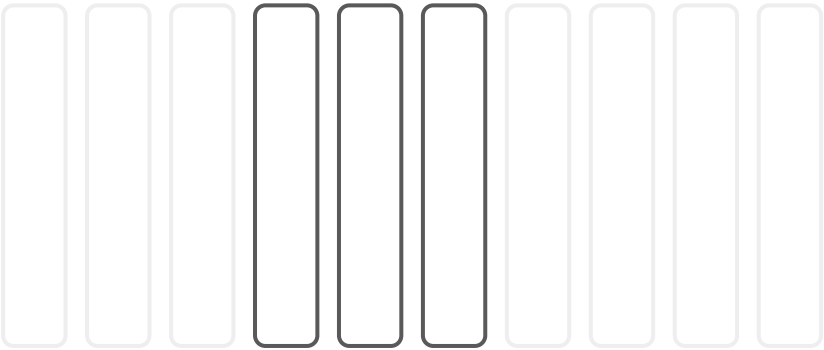
- Recursive Feature Elimination



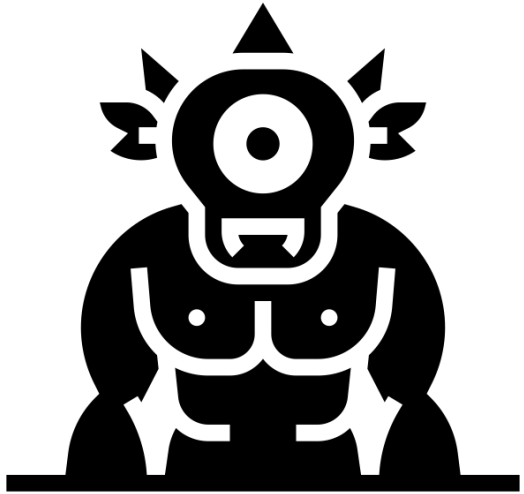


Ранжируем результат поиска

- Recursive Feature Elimination

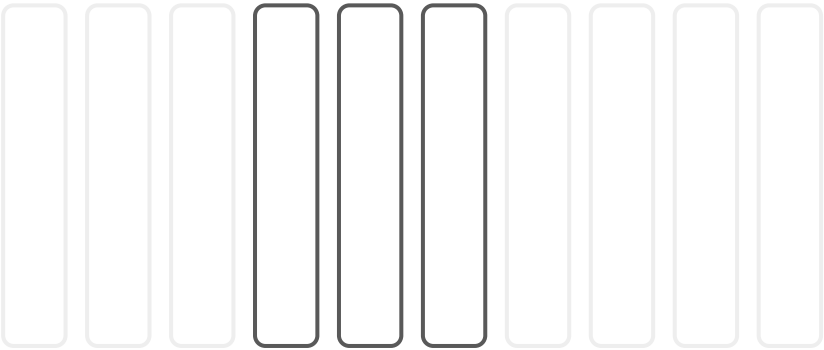


и т.д. до остановки

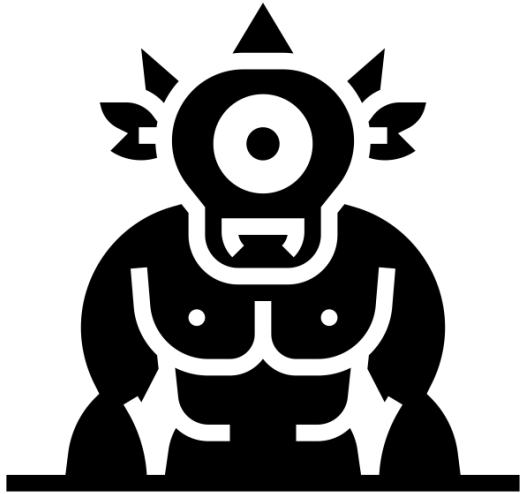


Ранжируем результат поиска

- Recursive Feature Elimination

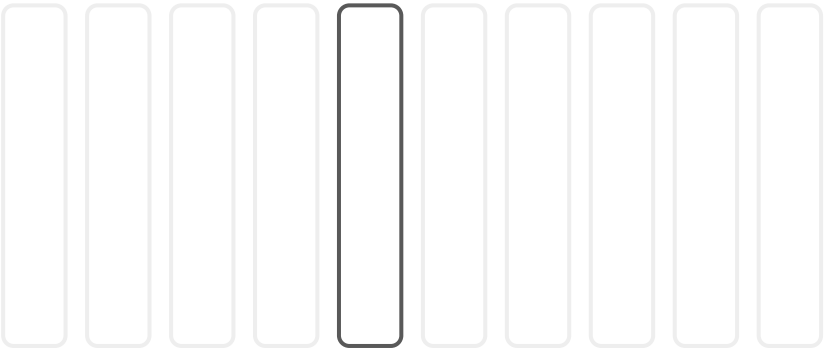


Сколько отбрасывать?

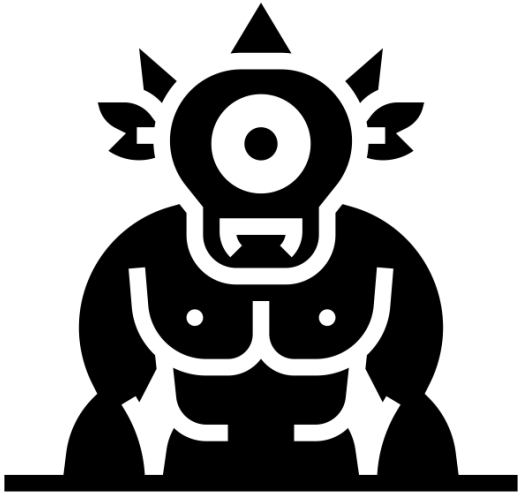


Ранжируем результат поиска

- Recursive Feature Elimination



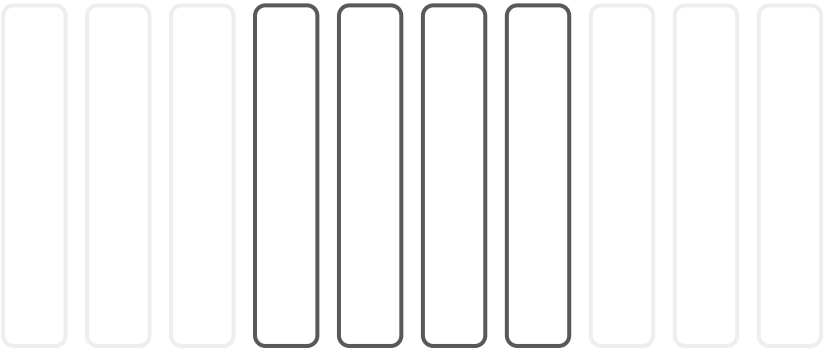
Сколько отбрасывать?



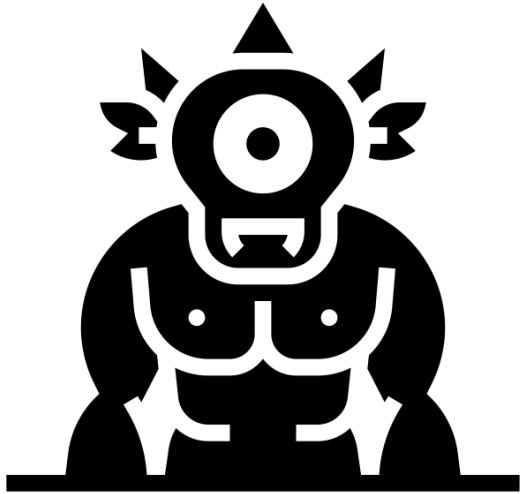
```
train_df.shape = (46152, 10667)
Iteration time: 1 min
```

Ранжируем результат поиска

- Recursive Feature Elimination

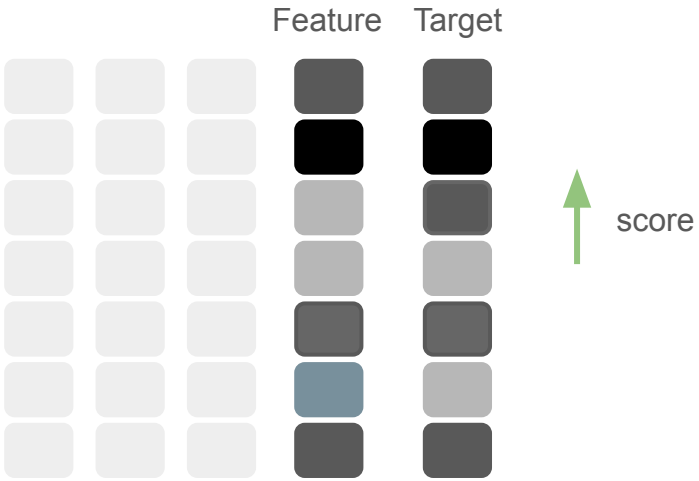


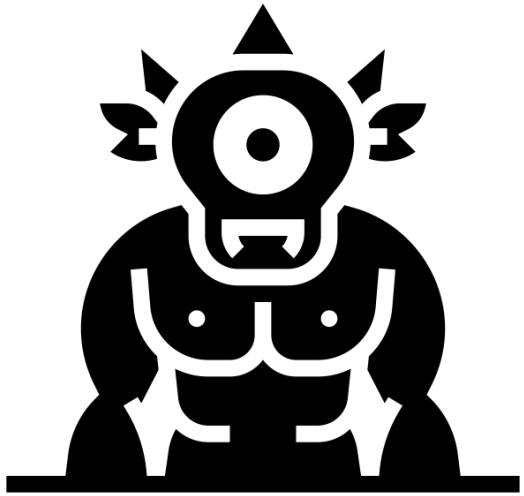
Сколько отбрасывать?
Сколько оставлять?



Ранжируем результат поиска

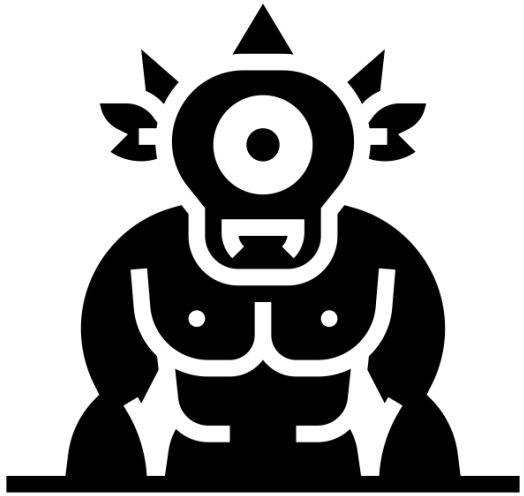
- Permutation importance selection





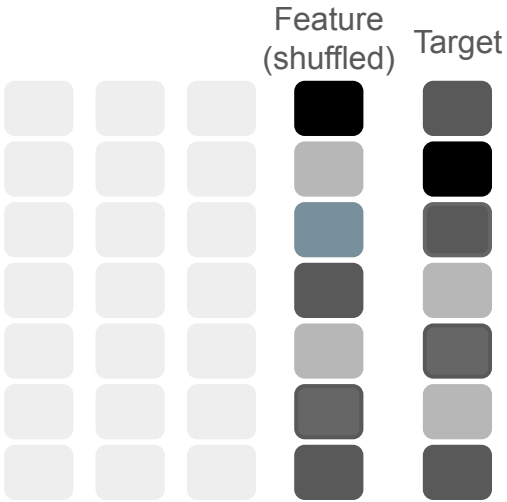
Ранжируем результат поиска

- Permutation importance selection

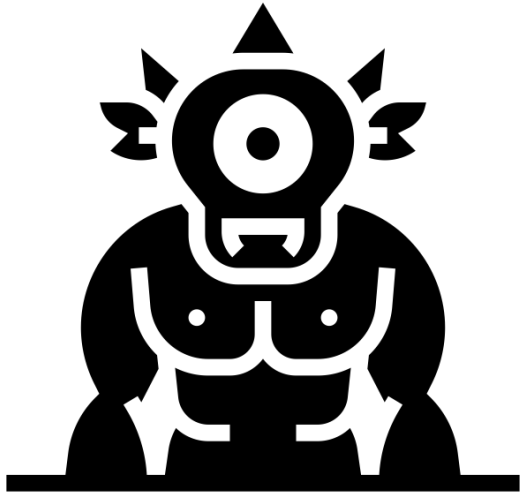


Ранжируем результат поиска

- Permutation importance selection

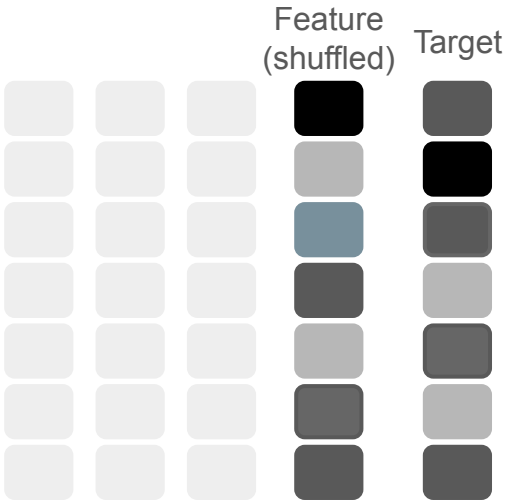


PFI = score - corrupted_score



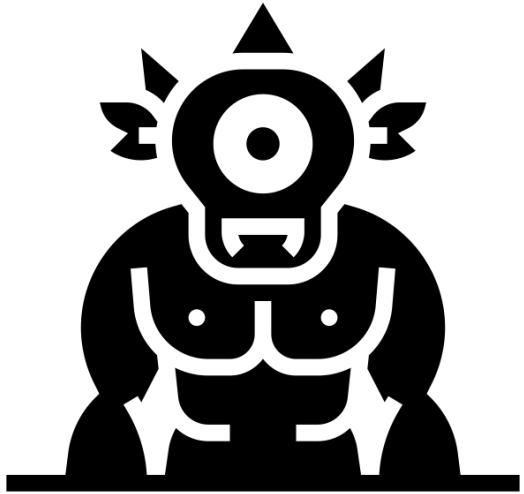
Ранжируем результат поиска

- Permutation importance selection



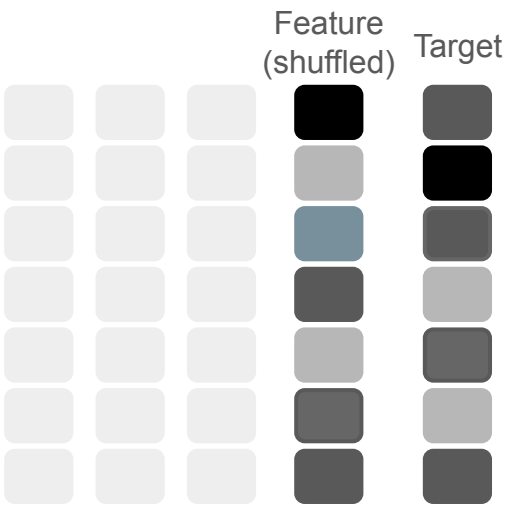
PFI = score - corrupted_score
(в среднем за K итераций)

K * N прогонов



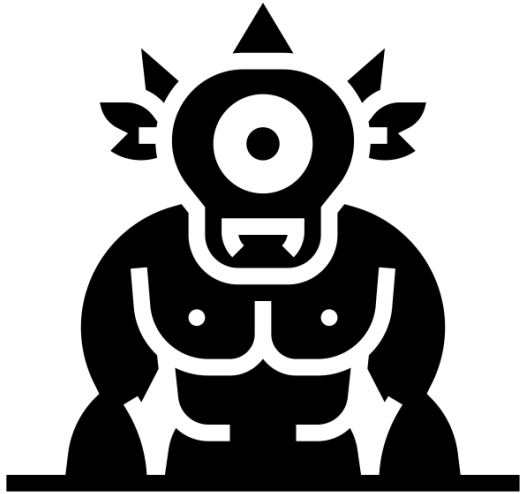
Ранжируем результат поиска

- Permutation importance selection



PFI = score - corrupted_score
(в среднем за K итераций)

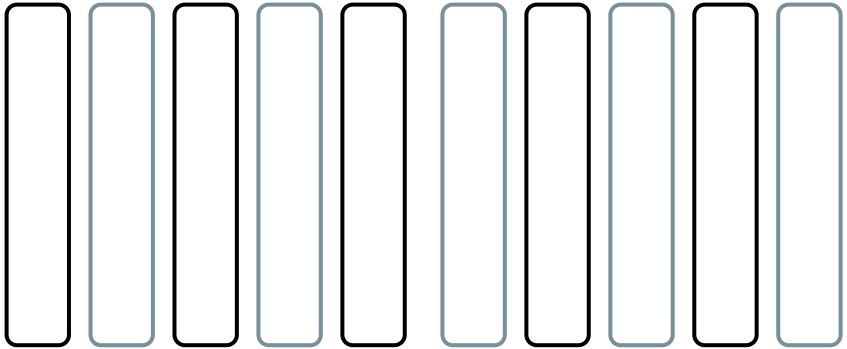
K * N прогонов



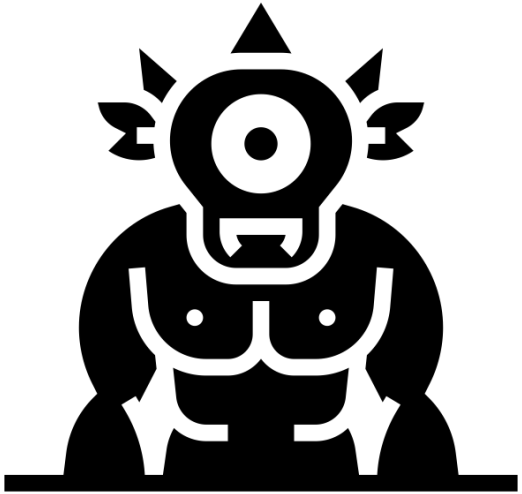
```
train_df.shape = (46152, 10667)
K = 3
```

Ранжируем результат поиска

- Boruta

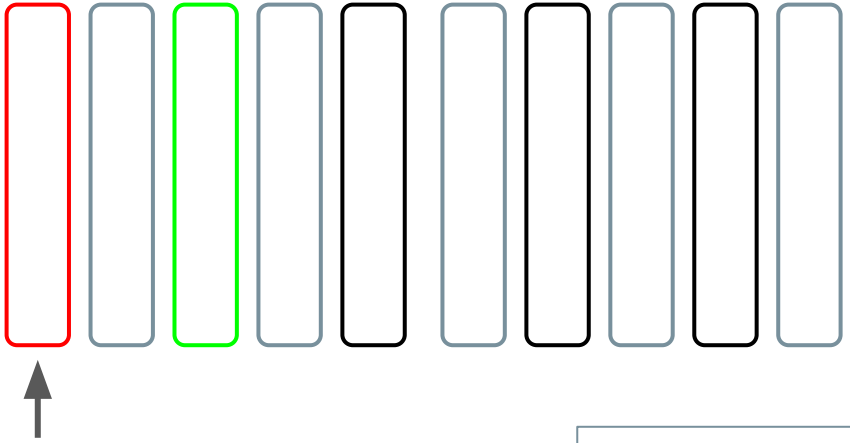


shadow features = shuffled



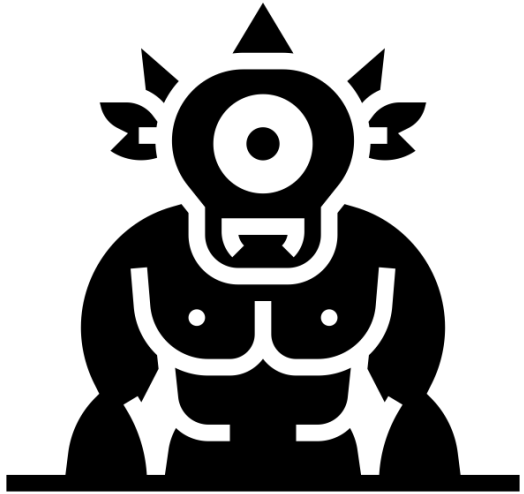
Ранжируем результат поиска

- Boruta



z-score < max(shadow z-score)

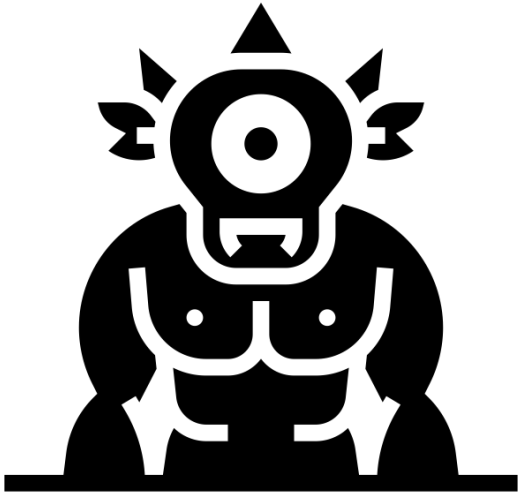
z-score = avg(importance) / std(importance)



Ранжируем результат поиска

- Featurewiz

https://github.com/AutoViML/featurewiz

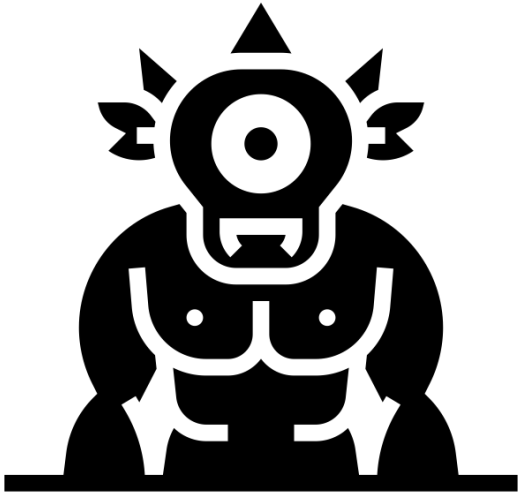


Ранжируем результат поиска

- Featurewiz

https://github.com/AutoViML/featurewiz

+ Сам определяет количество фичей
+ Встроенный FE, набор энкодеров



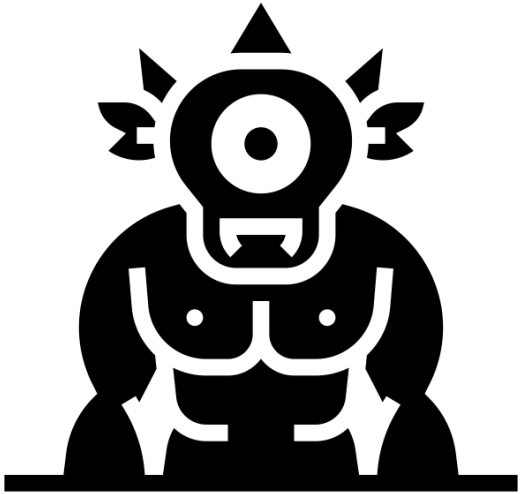
Ранжируем результат поиска

- Featurewiz

https://github.com/AutoViML/featurewiz

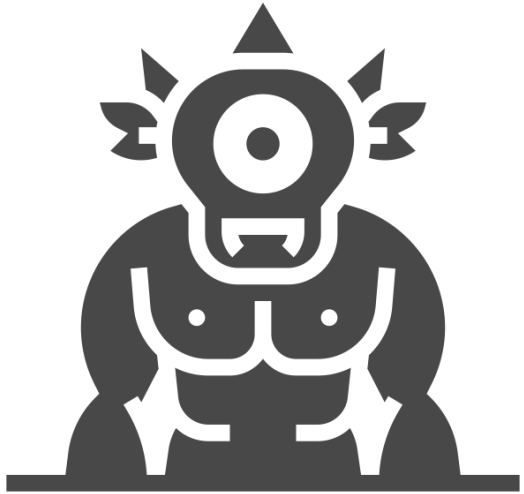
+ Сам определяет количество фичей
+ Встроенный FE, набор энкодеров

- Требует предобработки датасета
- Медленный
- Отбирает слишком много



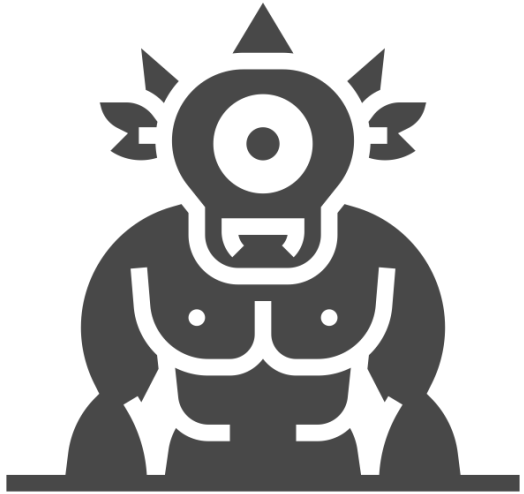
Ранжируем результат поиска

- Свой обёрточный метод



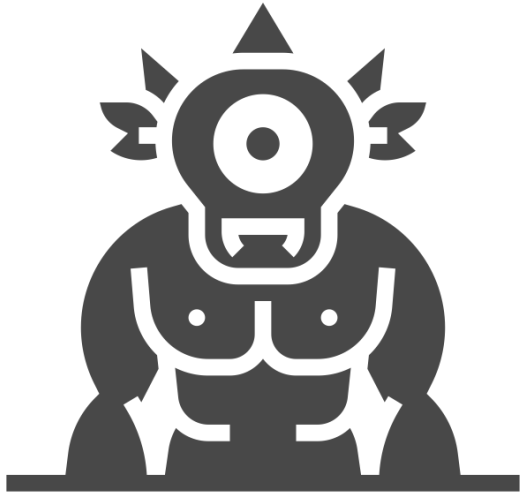
Ранжируем результат поиска

- Свой обёрточный метод
    - баланс скорости и качества



Ранжируем результат поиска

- Свой обёрточный метод
    - баланс скорости и качества
    - стабильность (защита от overfit)



Ранжируем результат поиска

- sklearn.feature_selection.RFE(step=137)
- https://github.com/smazzanti/mrmr
- FeatureWiz
- Upgini

Ранжируем результат поиска

```
train_df.shape = (46152, 10667)
Финтех
Бинарная классификация
Baseline: GINI=0.394962
```

Алгоритм	Время	GINI	Uplift
RFE	79 минут	0.489	0.095
MRMR	77 минут	0.492	0.097
Upgini	58 минут	0.487	0.093
FeatureWiz	8 часов	0.481	0.086

48588 строк в eval_df
6 CPU, 32Gb RAM

Ранжируем результат поиска

```
train_df.shape = (46152, 10667)
Финтех
Бинарная классификация
Baseline: GINI=0.394962
```

Алгоритм	Время	GINI	Uplift
RFE	79 минут	0.489	0.095
MRMR	77 минут	0.492	0.097
Upgini	58 минут	0.487	0.093
FeatureWiz	**8 часов**	0.481	0.086

Ранжируем результат поиска

```
train_df.shape = (46152, 10667)
Финтех
Бинарная классификация
Baseline: GINI=0.394962
```

Алгоритм	Время	GINI	Uplift
RFE	79 минут	0.489	0.095
MRMR	77 минут	0.492	0.097
Upgini	58 минут	0.487	0.093
FeatureWiz	**8 часов**	0.481	0.086

…и это сэмпл 25к строк

Ранжируем результат поиска

```
train_df.shape = (46152, 10667)
Финтех
Бинарная классификация
Baseline: GINI=0.394962
```

Алгоритм	Время	GINI	Uplift
RFE	79 минут	0.489	0.095
MRMR	**77 минут**	**0.492**	0.097
Upgini	**58 минут**	**0.487**	0.093
FeatureWiz	8 часов	0.481	0.086

Ранжируем результат поиска

```
train_df.shape = (46152, 10667)
Финтех
Бинарная классификация
Baseline: GINI=0.394962
```

Алгоритм	Время	GINI	Uplift
RFE	79 минут	0.489	0.095
MRMR	**77 минут**	**0.492**	0.097
Upgini	**58 минут**	**0.487**	0.093
FeatureWiz	8 часов	0.481	0.086

-25% время работы за -0.9% качества

Ранжируем результат поиска

```
train_df.shape = (46152, 10667)
Финтех
Бинарная классификация
Baseline: GINI=0.394962
```

Алгоритм	Время	GINI	Uplift
RFE	79 минут	0.489	0.095
MRMR	**77 минут**	**0.492**	0.097
Upgini	**58 минут**	**0.487**	0.093
FeatureWiz	8 часов	0.481	0.086

N

Ранжируем результат поиска

```
train_df.shape = (46152, 10667)
Финтех
Бинарная классификация
Baseline: GINI=0.394962
```

Алгоритм	Время	GINI	Uplift
RFE	79 + 58 минут	0.489	0.095
MRMR	**77 + 58 минут**	**0.492**	0.097
Upgini	**58 минут**	**0.487**	0.093
FeatureWiz	8 часов	0.481	0.086

N

Ранжируем результат поиска

```
train_df.shape = (46152, 10667)
Финтех
Бинарная классификация
Baseline: GINI=0.394962
```

Алгоритм	Время	GINI	Uplift
RFE	79 + 58 минут	0.489	0.095
MRMR	**77 + 58 минут**	**0.492**	0.097
Upgini **(65)**	**58 минут**	**0.487**	0.093
FeatureWiz **(2308)**	8 часов	0.481	0.086

N

Ранжируем результат поиска

```
train_df.shape = (46152, 10667)
Финтех
Бинарная классификация
Baseline: GINI=0.394962
Убрали ближайший к train месяц
```

Алгоритм	Время	Uplift	Uplift с “дыркой”
RFE	79 минут	0.095	0.079 (-0.014)
MRMR	77 минут	0.097	0.067 (-0.030)
Upgini	58 минут	0.093	0.080 (-0.013)
FeatureWiz	8 часов	0.086	0.065 (-0.021)

Ранжируем результат поиска

```
train_df.shape = (46152, 10667)
Финтех
Бинарная классификация
Baseline: GINI=0.394962
Убрали ближайший к train месяц
```

Алгоритм	Время	Uplift	Uplift с “дыркой”
RFE	79 минут	0.095	0.079 (-0.014)
MRMR	77 минут	0.097	**0.067 (-0.030)**
Upgini	58 минут	0.093	0.080 (-0.013)
FeatureWiz	8 часов	0.086	0.065 (-0.021)

Ранжируем результат поиска

```
train_df.shape = (46152, 10667)
Финтех
Бинарная классификация
Baseline: GINI=0.394962
Убрали ближайший к train месяц
```

Алгоритм	Время	Uplift	Uplift с “дыркой”
RFE	79 минут	0.095	0.079 (-0.014)
MRMR	77 минут	0.097	**0.067 (-0.030)**
Upgini	58 минут	0.093	0.080 (-0.013)
FeatureWiz	8 часов	0.086	0.065 (-0.021)

наименее стабильные фичи

Ранжируем результат поиска

```
train_df.shape = (12080, 2345)
Цены на недвижимость
Регрессия
Baseline: R2=0.493
```

Алгоритм	Время	R2	Uplift
RFE	0.5 минут	0.533	0.040
MRMR	2 минуты	0.523	0.030
Upgini	7 минут	0.566	0.073
FeatureWiz	12 минут	0.514	0.021

1812 строк в eval_df

Ранжируем результат поиска

```
train_df.shape = (12080, 2345)
Цены на недвижимость
Регрессия
Baseline: R2=0.493
```

Алгоритм	Время	R2	Uplift
RFE	**0.5 минут**	**0.533**	**0.040**
MRMR	2 минуты	0.523	0.030
Upgini	**7 минут**	**0.566**	**0.073**
FeatureWiz	12 минут	0.514	0.021

Ранжируем результат поиска

```
train_df.shape = (12080, 2345)
Цены на недвижимость
Регрессия
Baseline: R2=0.493
```

Алгоритм	Время	R2	Uplift
RFE	**0.5 минут**	**0.533**	**0.040**
MRMR	2 минуты	0.523	0.030
Upgini	7 минут	0.566	0.073
FeatureWiz	**12 минут**	**0.514**	**0.021**

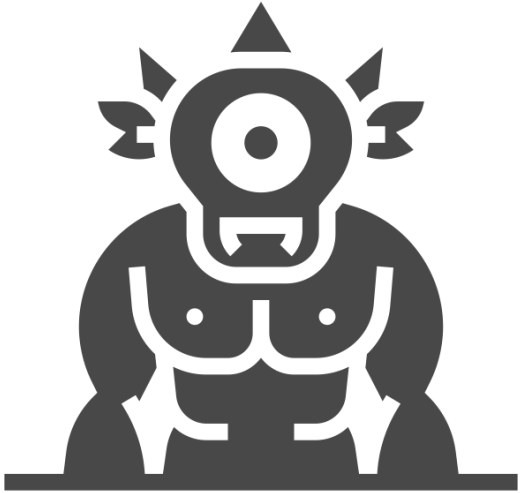
Ранжируем результат поиска

```
train_df.shape = (12080, 2345)
Цены на недвижимость
Регрессия
Baseline: R2=0.493
```

Алгоритм	Время	R2	Uplift
RFE	**0.5 минут**	**0.533**	**0.040**
MRMR	2 минуты	0.523	0.030
Upgini	7 минут	0.566	0.073
FeatureWiz	**12 минут**	**0.514**	**0.021**

Самый быстрый – не самый лучший
Самый медленный – тоже

Камни на пути



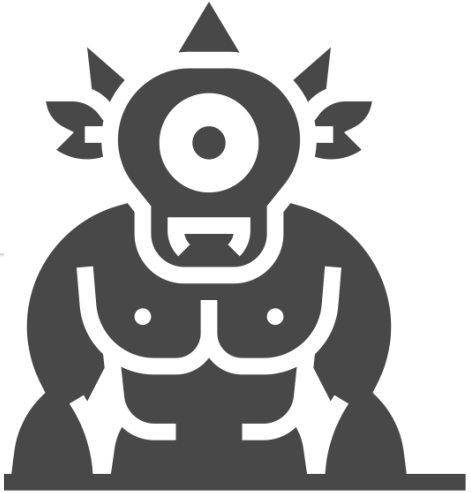
Камни на пути

Fun fact: LightGBM не может обеспечить идемпотентный feature importance на разном порядке колонок

guolinke commented on Mar 31, 2018 Collaborator

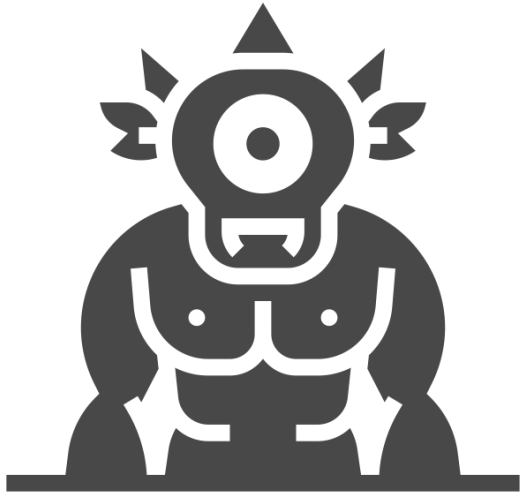
This is by design. feature order will affect the accuracy.
The reason is, when choose a feature to split tree node, if two features have the same split gain, the feature with smaller index(id) will be chosen.

👍 4



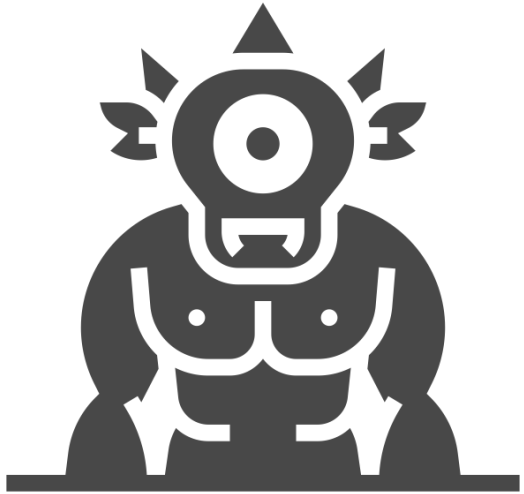
Камни на пути

- Строго следим за сортировкой

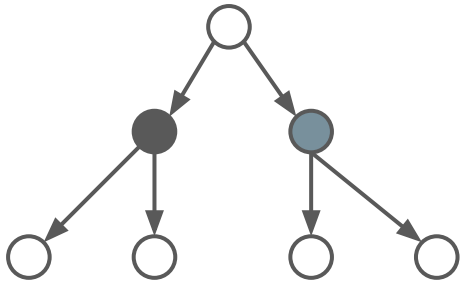


Камни на пути

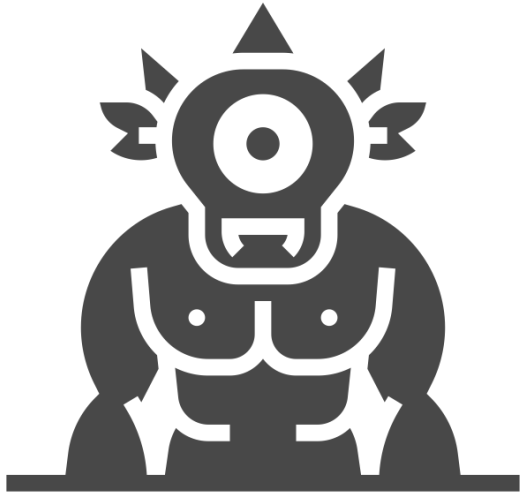
Деревья решений “продвигают” скоррелированные фичи



Камни на пути

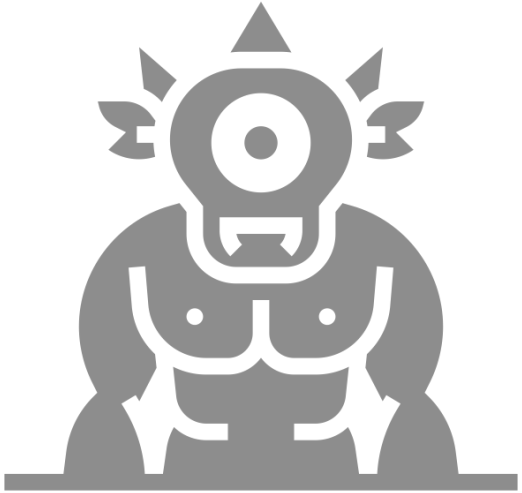


Количество сплитов будет похожим!



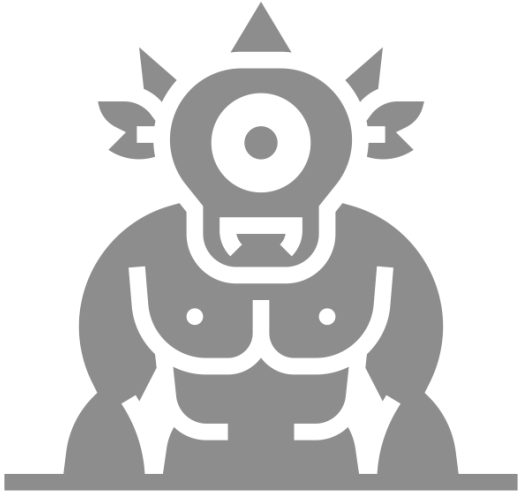
Камни на пути

- Строго следим за сортировкой
- Убираем фичи, скоррелированные с другими



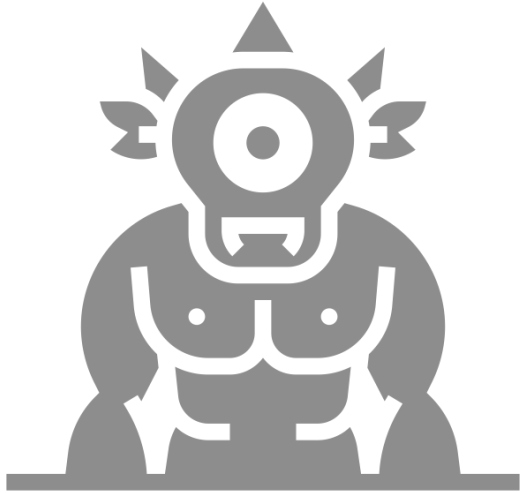
Камни на пути

- Строго следим за сортировкой
- Убираем фичи, скоррелированные с другими
    - SULOV в FeatureWiz (**s**earch for **u**ncorrelated **l**ist of **v**ariables)



Камни на пути

Стандартные методы плохо работают с разреженными фичами



Камни на пути

```
train_df.shape = (12080, 2345)
Цены на недвижимость
Регрессия
36 фичей с hit <= 40%
```

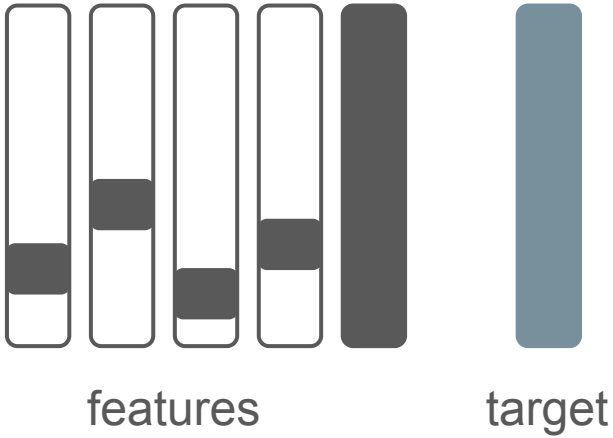
Алгоритм	Uplift без разреженных фичей	Uplift с разреженными фичами
RFE	0.050	0.040
MRMR	0.012	0.030
Upgini	0.053	0.073
FeatureWiz	0.043	0.021

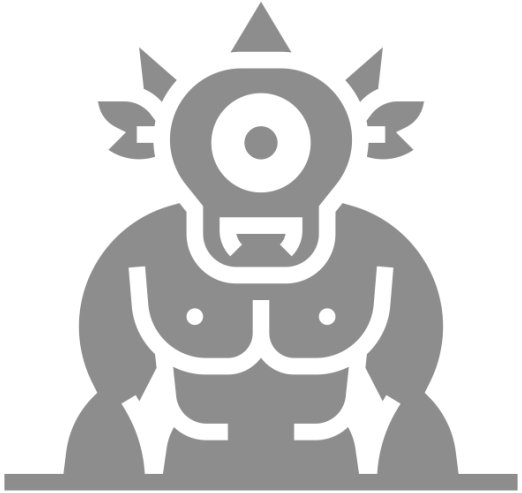
Камни на пути

```
train_df.shape = (12080, 2345)
Цены на недвижимость
Регрессия
36 фичей с hit <= 40%
```

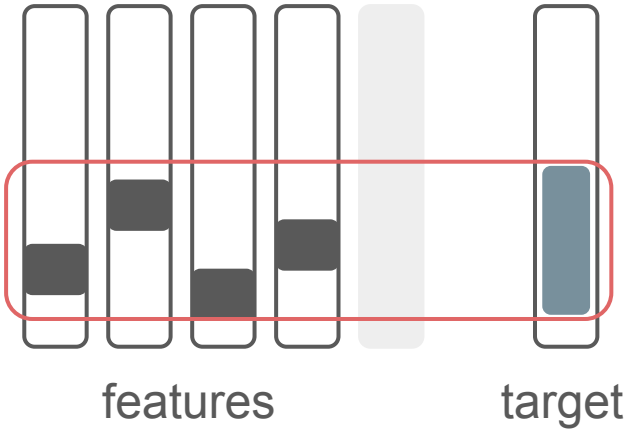
Алгоритм	Uplift без разреженных фичей	Uplift с разреженными фичами
RFE	**0.050**	**0.040**
MRMR	0.012	0.030
Upgini	0.053	0.073
FeatureWiz	**0.043**	**0.021**

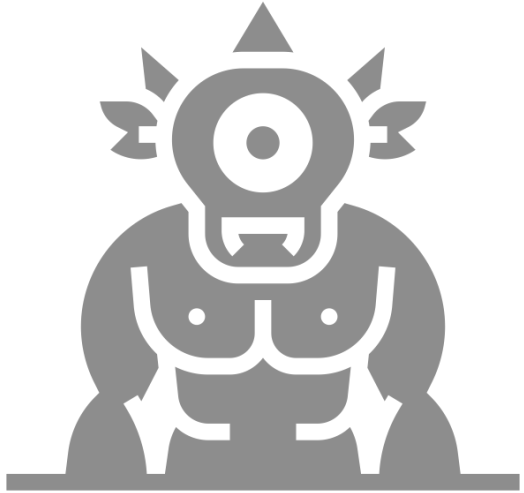
Камни на пути





Камни на пути





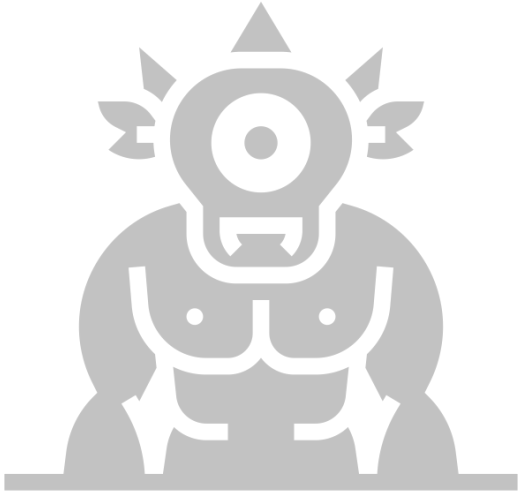
Камни на пути

- Строго следим за сортировкой
- Убираем фичи, скоррелированные с другими
- Учитываем сегменты заполненности для разреженных фичей

И ещё раз про FE

```
train_df.shape = (100000, 3311)
Предсказание оборота
Регрессия
Многомерный временной ряд
Upgini
```

	Время	Uplift
Только отбор	12 минут	0.067
FE + отбор	25 минут	0.147

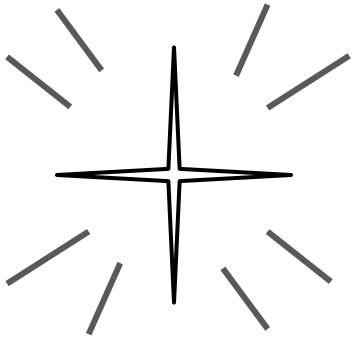


```
149 relevant feature(s) found with the search keys:
['date', 'phone', 'hashed_email', 'ip']
```

Accuracy after enrichment

Copy Share

Dataset type	Rows	Mean target	Baseline GINI	Enriched GINI	Uplift
Train	55920	0.4464	0.4352	0.6575	0.2223
Eval 1	6061	0.4267	0.3496	0.4759	0.1263



Итог

Итог

- Вместо метаданных – ищем по эталону

Итог

- Вместо метаданных – ищем по эталону
- Нашли данные? А надо – фичи

Итог

- Вместо метаданных – ищем по эталону
- Нашли данные? А надо – фичи
- Чтобы всё работало, нужно:
    - Обходить комбинаторику

Итог

- Вместо метаданных – ищем по эталону
- Нашли данные? А надо – фичи
- Чтобы всё работало, нужно:
    - Обходить комбинаторику
    - Улучшать эмбеддинги

Итог

- Вместо метаданных – ищем по эталону
- Нашли данные? А надо – фичи
- Чтобы всё работало, нужно:
    - Обходить комбинаторику
    - Улучшать эмбеддинги
    - Ускорять алгоритмы

Итог

- Вместо метаданных – ищем по эталону
- Нашли данные? А надо – фичи
- Чтобы всё работало, нужно:
    - Обходить комбинаторику
    - Улучшать эмбеддинги
    - Ускорять алгоритмы
    - Заботиться о скоррелированных и разреженных фичах

Итог

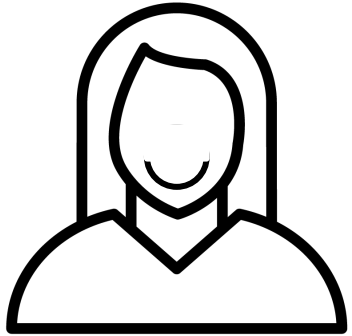
- Вместо метаданных – ищем по эталону
- Нашли данные? А надо – фичи
- Чтобы всё работало, нужно:
    - Обходить комбинаторику
    - Улучшать эмбеддинги
    - Ускорять алгоритмы
    - Заботиться о скоррелированных и разреженных фичах
    - …и много чего ещё

Итог

- Вместо метаданных – ищем по эталону
- Нашли данные? А надо – фичи
- Чтобы всё работало, нужно:
    - Обходить комбинаторику
    - Улучшать эмбеддинги
    - Ускорять алгоритмы
    - Заботиться о скоррелированных и разреженных фичах
    - …и много чего ещё
      (чистка эталона, правильный тип кросс-валидации, time series задачи, перебалансировка, расчёт метрик…)

Итог

- Вместо метаданных – ищем по эталону
- Нашли данные? А надо – фичи
- Чтобы всё работало, нужно:
    - Обходить комбинаторику
    - Улучшать эмбеддинги
    - Ускорять алгоритмы
    - Заботиться о скоррелированных и разреженных фичах
    - …и много чего ещё
      (чистка эталона, правильный тип кросс-валидации, time series задачи, перебалансировка, расчёт метрик…)



Спасибо за внимание!

Upgini https://upgini.com/

@Suncelesta

В ролях:

Даша

Болотный монстр
by Amethyst Studio

Гигант
by Eucalyp