Изучение данных в задаче кредитного скоринга

Каюмов Эмиль, 517
ММП ВМК МГУ
Курс «Прикладные задачи анализа данных»

Начнём с простого

- Выделяется не только тип кредита (микрозаймы)

```
train.groupby('CREDIT_ACTIVE')['DEF'].mean()
```

```
CREDIT_ACTIVE
0    0.030186
1    0.039676
2    0.035062
3    0.028169
Name: DEF, dtype: float64
```

- «1» соответствует «договор активен» — логично
- «3» соответствует «безнадёжный долг» — редко встречается

Начнём с простого

- Некоторые признаки лучше удалять

```
train.groupby('CREDIT_COLLATERAL')['DEF'].count()
```

```
CREDIT_COLLATERAL
0    1787570
1          1
Name: DEF, dtype: int64
```

Начнём с простого (2)

- AMT_REQ = sum(AMT_REQ_*) — запросы к истории

```
train.groupby('ID')[['AMT_REQ', 'DEF']].first().reset_index().groupby('DEF')['AMT_REQ'].mean()
```

```
DEF
0    2.992704
1    2.816376
Name: AMT_REQ, dtype: float64
```

- Больше обращений = больше истории = больше информации о клиенте (можно предложить «правильные условия)

Начнём с простого (3)

- Больше платёж — сложнее платить

```
train.groupby('DEF')['AMT_ANNUITY'].mean()
```

```
DEF
0    2874.937464
1    3080.046587
Name: AMT_ANNUITY, dtype: float64
```

```
labels = train.groupby('ID')['DEF'].first()
df = train.groupby('ID')['AMT_ANNUITY'].sum().reset_index()
df['DEF'] = df['ID'].map(labels)
df.groupby('DEF')['AMT_ANNUITY'].mean()
```

```
DEF
0    28588.418575
1    31471.262791
Name: AMT_ANNUITY, dtype: float64
```

Если нет ID людей…

- Возьмём характерные для кредита параметры (дата начала, сумма, тип, источник, …)
- С помощью kNN поищем в точности такие же в тестовой части…
- 15% строк (=запросов) из тестовой части встречаются в трейне, у половины заявок в тесте встречаются в истории запросы, которые мы уже видели в трейне…
  - Актуализировать данные
  - Детектировать людей
  - …
- Можно и что-нибудь ещё так поискать