курс «Машинное обучение»

Генерация признаков
(Feature Engineering / Construction)

Александр Дьяконов

План

признаки / генерация признаков

типы признаков

генерация с помощью EDA

типичные хорошие признаки определённых типов

«Applied machine learning» is basically feature engineering.
Andrew Ng

Генерация признаков

Генерация признаков (feature engineering / construction) – процесс придумывания способов описания данных с помощью простых значений, которые должны отражать характеристики объектов исследований, через которые могут выражаться целевые значения.

Изначально объекты могут быть заданы непризнаковым описанием:

- измерения
- веб-страницы
- файлы
- участники соцсети

и т.д.

Важно понимать

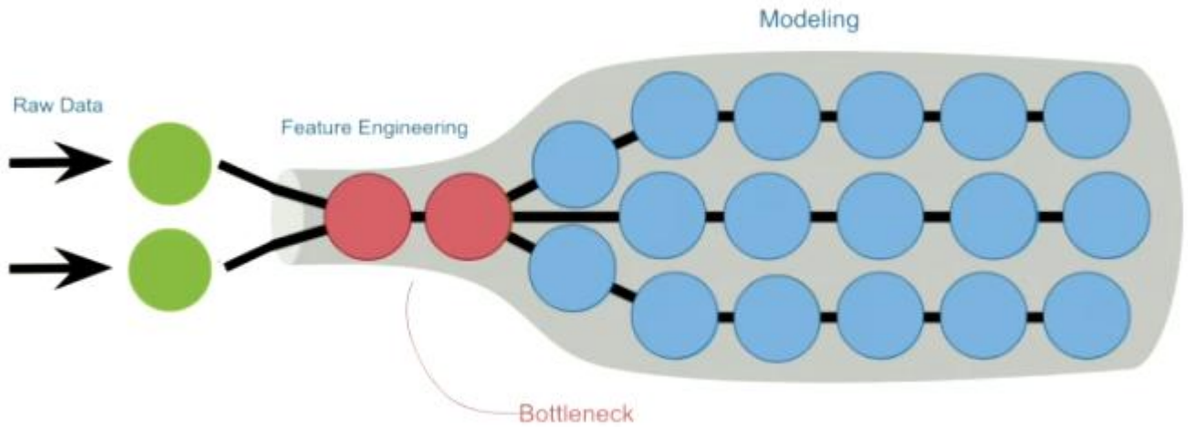
Процесс создания признакового пространства зависит от модели, которую будем использовать

OHE-кодирование предпочтительнее для линейных моделей,
умное кодирование категорий – для деревьев
Выбросы можно не удалять для робастной модели
(– и этапы предработки данных тоже!)

Следует использовать:

- **контекст (знание предметной области)**
- **EDA**

Важно понимать



https://www.youtube.com/watch?v=vwiVKm5_nRA

Признаковое описание – всё, что знает модель о данных

Признаки (Features)

Признак – функция на множестве объектов

$$f : X \to S$$

Признак пол

Клиенты $\to$ {М, Ж}

Признак доход

Клиенты $\to$ {…, 10 000, 20 000, …, NA}

Значения признака могут быть не определены
это тоже важная информация!

Некоторые значения можно восстановить по другим признакам
(например, пол)

Типы числовых признаков $S \subseteq \mathbb{R}$

- **вещественные**

– значения вещественные числа (измерения или характеристики чего-то), часто выделяют дискретные (discrete) и непрерывные (continuous variables)

- **интервальные (Interval)**
- **относительные (Ration)**

- **категориальные (categorical variables)**

– значение переменной – принадлежность к одной из категорий (level), множество категорий актуально конечно (часто фиксировано)

- **неупорядоченные категориальные (номинальные – Nominal, факторные)**
- **порядковые (Ordinal) или упорядоченные категориальные**

отдельно отметим: бинарные

Типы числовых признаков

Тип признака	Операции	Трансформации	Примеры
номинальные	== перестановки mode, entropy, contingency, correlation, X2-test	перестановка	ID, пол, цвет, профессия
порядковые	> median, percentiles, rank correlation, run tests, sign tests	монотонное преобразование	оценка, рейтинг, место в соревновании
интервальные	+, – mean, standard deviation, Pearson's correlation, t and F tests	A∗X + B	дата, температура по Цельсию
относительные	*, / geometric mean, harmonic mean, percent variation	A∗X	возраст, масса, длина, цена, температура по Кельвину

Категориальные

Неупорядоченные

жанры фильмов, имена актёров,
категории товаров

Порядковые

APPAREL

WEIGHT	BACK LENGTH	CHEST GIRTH	SIZE
Up to 8 lbs	6-9	10-13	XS
Up to 15 lbs	9-12	13-15	S
Up to 25 lbs	14-18	15-20	M
Up to 40 lbs	20-24	20-25	L
Up to 60 lbs	22-26	25-30	XL
Up to 120 lbs	24-28	30-35	XXL

COLLARS & BANDANAS

NECK	SIZE
6-12	S
10-16	M
14-24	L

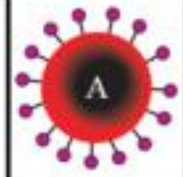
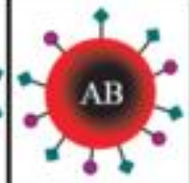
Часто «кольцевой порядок»:

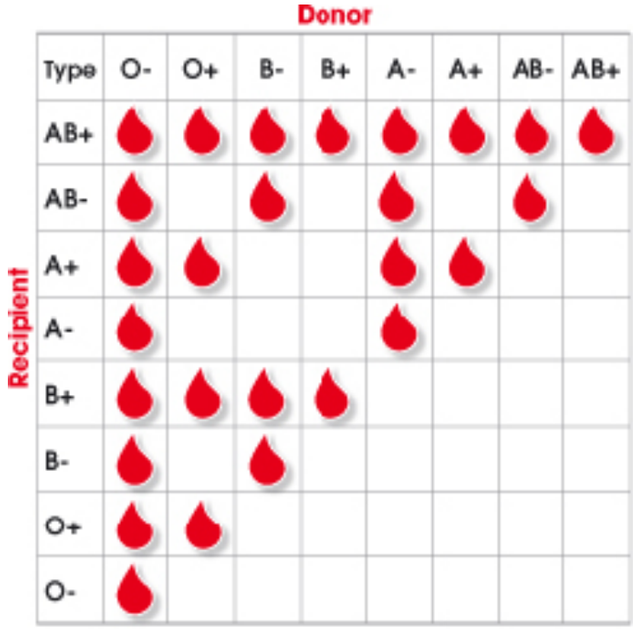
времена года, дни недели, час

Категориальные

С возможным правильным кодированием

группа крови

	Группа 0 (I)	Группа A (II)	Группа B (III)	Группа AB (IV)
Тип эритроцитов				
Антитела в плазме	 α- и β-агглютинины	 β-агглютинины	 α-агглютинины	Нет
Антигены на эритроцитах	Нет	A-агглютиноген	B-агглютиноген	A- и B-агглютиногены



Простые типы нечисловых признаков

- **временные отметки**
- **множества, наборы** (ex: пары координат)
- **строки** (ex: платёжная строка «00010011»)

	дата	пол	образование	сумма	число просрочек	платёжная строка
0	12/01/2017	1	высшее	5000.0	0	0000
1	13/01/2017	1	высшее	2500.0	1	001000
2	13/01/2017	0		13675.0	3	111
3	25/01/2017	0	начальное	NaN	0	0

Дальнейший план

Всё что касается EDA:

- Контекстные признаки
- Служебные признаки
- Утечки
- Странности в данных

Отдельные виды признаков

- Строковые
- Категориальные
- Вещественные
- Временные
- Географические
- Деньги, количество, ...

Контекстные признаки

– это признаки, смысл которых явно прописан в постановке задачи или понятен из контекста.

Смысл определяет:

- **область значений**
- **примерное распределение в этой области**

	ap_hi	ap_lo	ap_hi_new	ap_lo_new
0	150	1100	150	110
1	11	70	110	70
2	12	80	120	80
3	11	570	115	70
4	1	2080	120	80

Пример: диаметр зрачка

Почему плохо решать задачи без знания предметной области…

Медицинский проект с японцами

異常なし	**нет аномалий**
異常所見なし	**нет аномальных находок**
視神経乳頭陥凹拡大疑い（右）	**подозрение на увеличение выемки головы зрительного нерва (справа)**
（左）乳頭部出血の疑	**(слева) подозрение на папиллярное кровоизлияние**
(両）豹紋眼底	**(оба) лепидоцеллюлярное дно**

+ разный подход к диагностике заболеваний (в Европе и Японии)

японские клинические рекомендации (2016) по сахарному диабету:

HbA1c(NGSP値) > 6.5%

другая лабораторная методика (в другом признаке):

HbA1c(JDS値) > 6.1%

Служебные признаки

**могут не входить в явном виде в признаковую матрицу,
но из значения определяются из способа организации данных.**

- **номер строки** (а также производные признаки, например, чётность номера строки)
- **номер объекта в какой-то внутренней нумерации** (например, id объекта), производные признаки от этого номера
- **порция данных** (если датасет разбит на несколько частей, например, train / test / valid)
- **константный признак**
- **характеристические признаки** (выполняется ли какое-то свойство)
- **характеристика особенностей данных** (например, число пропусков на объекте)
- **имена и характеристики записей** (в задачах, где объекты хранятся отдельно, например как файлы изображений)

Казалось бы, логично не рассматривать такие признаки…

Служебные признаки

могут не входить в явном виде в признаковую матрицу,
но из значения определяются из способа организации данных.

	Пол	Рост	Вес
1	М	170.0	80.0
20	Ж	NaN	70.0
23	М	167.0	75.0
33	М	NaN	NaN
40	Ж	180.0	65.0

	index	Пол	Рост	Вес	#nan	Пол=М
0	1	М	170.0	80.0	0	0
1	20	Ж	NaN	70.0	1	0
2	23	М	167.0	75.0	0	1
3	33	М	NaN	NaN	2	1
4	40	Ж	180.0	65.0	0	0

```
data.reset_index(inplace=True)
data['#nan'] = data.isnull().sum(axis=1)
data['Пол=М'] = (data['Пол'] == 'М').astype(int)
data['Рост2'] = data['Рост'] / 10
data['Рост2'] = data['Рост'] - 10*np.floor(data['Рост2'].values)
```

Служебные признаки

Когда такие признаки важны:

- **особенность значения ⇐ особенность объекта**

«круглый доход» – не знает точного значения

«круглый рост, вес, давление» – не знает точного

- **улучшает качество**

«есть ли пропуск в признаке NAME»,

«сколько пропусков / аномальных значений»

- **поиск утечек в данных**

- **организация эксперимента**

(разбиение на корзины и т.п.)

Пример: области значений целевого (маленькие, средние, большие) + StratifiedKFold

Утечка в данных

– информация, которая повышает качество решения задачи машинного обучения, но теряет эти свойства при тестировании на независимом и правильно организованном контроле

(при эксплуатации алгоритма)

Пол	Рост	Вес	Класс
М	170	80	0
Ж	NA	70	0
М	167	75	0
М	NA	NA	1
Ж	180	65	1

По определению утечка приводит к переобучению

но далеко не единственная причина переобучения

(объём данных, сложность модели и т.п.)

Виды утечек

1) зависимость от служебных признаков,
в том числе

- от порядка (ex: номера строки)
- от организации данных (от каталога)
- от способа представления (названия файлов, времени его создания и т.п.)
- от особенностей (наличия пропусков, дубликатов и т.п.)

2) неявное использование информации

- о целевом векторе
- из будущего или настоящего

3) содержание ответа в исходных данных
как правило, из-за заглядывания в будущее
(пример про номер страницы и число страниц в сессии)

Строковые признаки

T	company	client
12C	Shell	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_4) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.143 Safari/537.36
14	shel	NA
15	bp	NA
11	B&P	NA
10C	Procter&Gamble	NA

отдельная тема – обработка текстов

Строковые признаки

T	company	browser	ip	OS
12	Shell Gas station	Mozilla	53.0.2785.143	Mac
14	Shell Gas station	NA	NA	NA
15	BP Gas station	NA	NA	NA
11	BP Gas station	NA	NA	NA
10	P&G Manufacturer	NA	NA	NA

```
import user_agents
s = 'Mozilla/5.0 ...'
q = user_agents.parse(s)
q.is_mobile
False
q.os.family
'Ubuntu'
```

Категориальные признаки

1. Автоматическое определение категориальности

- если значения – строки
- если мало уникальных значений

	city	sex	income
0	Moscow	M	110
1	London	F	200
2	London	M	140
3	Paris	M	120
4	Moscow	F	190

```
city строка, мало уникальных
sex строка, мало уникальных
```

```
def find_cat(data):
"""
найти все признаки,
в которых первое значение - строка
и / или мало значений
"""
for name in data.columns:
    s = ''
    s += name
    if (type(data[name][0]) == str):
        s += ' строка,'
    if (data[name].nunique() <= 3):
        s += ' мало уникальных'
    if (s!=name):
        print (s)
```

Категориальные признаки

2. Создание новых категориальных признаков

- **конъюнкция признаков**

	city	sex	income	city + sex
0	Moscow	M	110	Moscow + M
1	London	F	200	London + F
2	London	M	140	London + M
3	Paris	M	120	Paris + M
4	Moscow	F	190	Moscow + F

```
def make_conj(data, feature1, feature2):
    """
    конъюнкция двух признаков
    """
    data[feature1 + ' + ' + feature2] =
                        data[feature1].astype(str) + ' + ' +
                                    data[feature2].astype(str)
    return (data)

# пример использования
make_conj(data, 'city', 'sex')
```

Категориальные признаки

2. Создание новых категориальных признаков

- **конъюнкция признаков**

может быть очень полезно: kNN, линейные алгоритмы

- **создание новых признаков по контекстным**

Пример: верхние уровни иерархии

Категориальные признаки

3. Простейшее кодирование – по номеру категории Label Encoding

- **Лексикографический порядок (sklearn)**

sklearn.preprocessing.LabelEncoder

- **В порядке появления (pandas)**

pandas.factorize

Хорошая идея! Можно использовать сортировки по признакам

$\Rightarrow$ по порядку индуцированным каким-то признаком

- **Случайное кодирование**

dict + map

многократное случайное кодирование иногда хорошо работает с RF

Категориальные признаки

	city	sex	income	city_le	city_fz	city_rnd	sex_le	sex_fz	sex_rnd
0	Moscow	M	110	1	0	0.63	1	0	0.22
1	London	F	200	0	1	0.75	0	1	0.20
2	London	M	140	0	1	0.75	1	0	0.22
3	Paris	M	120	2	2	0.50	1	0	0.22
4	Moscow	F	190	1	0	0.63	0	1	0.20

```
from sklearn import preprocessing
le = preprocessing.LabelEncoder()

for name in cols:
    # лексикографический порядок
    data[name + '_le'] = le.fit_transform(data[name])
    # в порядке упоминания
    data[name + '_fz'] = pd.factorize(data[name])[0]
    # случайно
    dct = dict(zip(data[name].unique(),
               np.random.rand(data[name].nunique()).round(2)))
    data[name + '_rnd'] = data[name].map(dct)
```

Категориальные признаки

3. Простейшее кодирование – по номеру категории Label Encoding

- не подходит для линейных алгоритмов
- проблема новых категорий

(средним?)

Категориальные признаки

4. Dummy-кодирование / One-hot-encoding

	city	sex	income	city=Moscow	city=London	city=Paris	sex=M	sex=F
0	Moscow	M	110	1	0	0	1	0
1	London	F	200	0	1	0	0	1
2	London	M	140	0	1	0	1	0
3	Paris	M	120	0	0	1	1	0
4	Moscow	F	190	1	0	0	0	1

Мнение: OHE $\rightarrow$ m, dummy $\rightarrow$ m – 1
drop_first=True в get_dummies
(теперь False)

Категориальные признаки

OneHotEncoder (по умолчанию sparse, раньше – только с числами)

```
from sklearn import preprocessing
ohe = preprocessing.OneHotEncoder(sparse=False)
tmp = ohe.fit_transform(data[cols]).astype(int)
tmp = pd.DataFrame(tmp,
                   columns=['OHE_' + str(i) for i in range(tmp.shape[1])])
data = pd.concat([data, tmp], axis=1)
```

Ручное решение

```
def code_myohe(data, feature):
    """
    ручной способ OHE
    """
    for i in data[feature].unique():
        data[feature + '=' + i] = (data[feature] == i).astype(int)

for name in cols:
    code_myohe(data, name)
```

Самый простой способ

```
pd.get_dummies(data)
```

Категориальные признаки

ONE

- **Хороши для линейных алгоритмов**

можно кодировать N-1 категорию

- **Все признаки в одной шкале (на [0, 1])**

но есть тонкость, что категории разной мощности

- **Большое число категорий → сильно разреженные матрицы**

часто используют sparse-формат

- **Плохо, что после ONE значительная часть признаков – бинарные – описывают категориальные зависимости**

некоторые алгоритмы (RF) могут терять качество

- **Плохо, что значительная часть – признаки с большим числом категорий**

Проблема мелких и новых категорий

– возникает почти для всех способов кодирования

Часто: мелкие категории → в одну

Здесь: можно не кодировать!

простая и понятная интерпретация

Категориальные признаки

5. По значениям вещественного признака

	city	sex	income	city_mean_income	sex_mean_income
0	Moscow	M	110	150	123.3
1	London	F	200	170	195.0
2	London	M	140	170	123.3
3	Paris	M	120	120	123.3
4	Moscow	F	190	150	195.0

```
def code_mean(data, cat_feature, real_feature):
    """
    кодирование средним значением
    """
    mn = data.groupby(cat_feature)[real_feature].mean()
    return data[cat_feature].map(mn)

for name in cols:
    data[name + '_mean_income'] = code_mean(data, name, 'income')
```

Категориальные признаки

5. По значениям вещественного признака

- **естественная интерпретация:**

товары какой категории дороже

- **можно использовать другие статистики**

они не всегда логичны и лучше интерпретируются

- **можно кодировать по разным признакам**

Ниже отдельно рассмотрим случай

кодирования по целевому признаку

Категориальные признаки

6. По значениям категориального признака

- просто по мощности **Count Encoding**
- по частоте **Frequency Encoding**

	city	sex	income	city_vc	sex_vc	city_vcn	sex_vcn
0	Moscow	M	110	2	3	0.4	0.6
1	London	F	200	2	2	0.4	0.4
2	London	M	140	2	3	0.4	0.6
3	Paris	M	120	1	3	0.2	0.6
4	Moscow	F	190	2	2	0.4	0.4

```
data[name + '_vc'] = data[name].map(data[name].value_counts())
data[name + '_vcn'] = data[name].map(data[name].value_counts(normalize=True))
# другой способ
for name in cols:
    data[name + '_vc'] = data[name].map(data.groupby(name).size())
```

Категориальные признаки

Недостатки Count Encoding

- **коллизии (несколько категорий – один код)**

добавляем шум (не всегда это проблема)

- **проблема шума (мелкие категории)**

мелкие категории объединяем в одну, и новые!

- **проблема новых категорий**

см. выше

- **проблема утечки информации**

честно: без заглядывания в будущее

это не страшная утечка (допустима в соревнованиях)

Категориальные признаки

8. Хэш-кодирование

- средство против сильно разреженных данных
- могут быть коллизии (можно выполнять разные хэш-кодирования)

	city	sex	income	city_0	city_1	sex_0	sex_1
0	Moscow	M	110	1.0	-1.0	1.0	0.0
1	London	F	200	-2.0	2.0	-1.0	0.0
2	London	M	140	-2.0	2.0	1.0	0.0
3	Paris	M	120	1.0	-2.0	1.0	0.0
4	Moscow	F	190	1.0	-1.0	-1.0	0.0

```
from sklearn.feature_extraction import FeatureHasher
fh = FeatureHasher(n_features=2, input_type='string')
for name in cols:
    tmp = fh.fit_transform(data[name]).toarray()
    tmp = pd.DataFrame(tmp, columns=[name + '_' + str(i) for i in range(tmp.shape[1])])
    data = pd.concat([data, tmp], axis=1)
```

Категориальные признаки

9. По значению целевого – Target Encoding

- Mean Target Encoding
- Std Target Encoding
- ...

	city	sex	income	city_mt	sex_mt	target
0	Moscow	M	110	0.5	0.67	1
1	London	F	200	0.5	0.50	1
2	London	M	140	0.5	0.67	0
3	Paris	M	120	1.0	0.67	1
4	Moscow	F	190	0.5	0.50	0

- **это форма стекинга** (по одной переменной)
- **подходит для любых алгоритмов** (если правильно сделана)

Категориальные признаки

9. По значению целевого – Target Encoding

Почему хорошая идея –
упорядочивание категорий исходя из смысла задачи

Многие кодировки «случайны»,
а кодирование по значению целевого «логично»

Категориальные признаки: Target Encoding

наверное главная проблема:

неадекватная кодировка мелких категорий +
слияние этих категорий

Категориальные признаки: Target Encoding

Нельзя допустить утечки значений целевого!

особенно проблемно для мелких категорий

способы борьбы аналогичны стекингу

- **кодирование по отложенной выборке**

– сокращаем выборку для обучения

Хороший пример: кодирование по куску испорченных данных

- **k-fold-кодировка**

Идея: не использовать метку объекта при кодировании

разбиваем на фолды и кодируем

LOO-кодировка – проблемы утечки остаются

- **кодирование по случайным подвыборкам**

- **кодирование по предыдущим объектам (CatBoost)**

Категориальные признаки: сглаживание

добавляем среднее значение целевого с весом

$$\text{mean} = \frac{k_1 + \alpha \frac{m_1}{m}}{k + \alpha}$$

+ борьба с редкими категориями

на них оценка ненадёжна

см. оценку среднего

тонкий вопрос: как кодировать на обучении, как на тесте

обычно на тесте пересчитывают (по всему обучению)...

Категориальные признаки: добавление шума

Чаще мультипликативный шум...

```
def add_noise(series, noise_level):
        return series * (1 + noise_level * np.random.randn(len(series)))
```

Категориальные признаки: почему плохо LOO-кодировать

	sex	target	sex_loo
0	M	1	0.2
1	M	1	0.2
2	M	0	0.4
3	M	0	0.4
4	M	0	0.4
5	M	0	0.4

```
d1 = data.groupby('sex')['target'].sum()
d2 = data.groupby('sex').size()

data['sex_loo'] = (data['sex'].map(d1) - data['target']) / data['sex'].map(d2)
```

В явном виде утечка…

Ещё и упорядочивание неестественное!

Категориальные признаки: Кодирование по предыдущим объектам (CatBoost)

	sex	sex_cb	target
0	M	NaN	1
1	F	NaN	1
2	M	1.0	0
3	M	0.5	1
4	F	1.0	0

- **есть в CatBoost**

там аналогичный приём и для вычисления градиента

- **одна категория в обучении кодируется по-разному, а на контроле фиксировано**

- **нет гиперпараметров**

(хотя напрашивается ввести)

- **можно использовать разные порядки**

~ при построении разных деревьев

случайно сортировать порядки, индуцированные признаками

```
gb = data.groupby(name)
data[name + '_cb'] = (gb['target'].cumsum() - data['target']) / gb.cumcount()
```

Категориальные признаки: Target Encoding

На практике хороша смесь подходов!

Категориальные признаки

10. Экспертное кодирование

Если признак порядковый, то есть естественная нумерация

см. также кодирование названий географических регионов

11. Вложение категориальных признаков в маломерное пространство (Category Embedding)

отдельная тема

пример: транзакции упорядочены во времени, категория покупки – слово, естественным образом получаем текст ⇒ word2vec

Категориальные признаки: пропуски

- **создание отдельной категории «нет значения»**

- **игнорирование пропусков**

(например, в dummy-кодировке сопоставить им нулевую строку, т.е. соответствующие объекты не принадлежат ни одной категории)

- **стандартные методы обработки пропусков**

(при плотном кодировании, например, по целевому вектору)

Категориальные признаки: пример кода

http://contrib.scikit-learn.org/categorical-encoding/

https://www.kaggle.com/mlisovyi/9-ways-to-treat-categorical-features-updated#
https://www.kaggle.com/ogrellier/python-target-encoding-for-categorical-features#
https://www.kaggle.com/vprokopev/mean-likelihood-encodings-a-comprehensive-study#
https://github.com/DenisVorotyntsev/CategoricalEncodingBenchmark

Вещественные признаки

были способы генерации признаков (в обработке данных):

- **дискретизация (binning / quantization)**
  округление (rounding)
- **деформация (функция над признаком)**
- **сглаживание**
- **нормировка (специальный вид деформации)**
- **новые признаки (функции над несколькими)**

Продажа планшетов	Продажа телефонов	Продажа ноутбуков	Общие продажи
10	30	2	32
12	42	2	56
10	20	1	31
15	31	2	48
5	15	0	20

Вещественные признаки: новые признаки

- суммы групп признаков
- мономы
- расстояние (ядро) до какого-то объекта

	f	g	f^2	fg	g^2
0	1.0	0.0	1.0	0.0	0.0
1	2.0	2.0	4.0	4.0	4.0
2	3.0	1.0	9.0	3.0	1.0
3	4.0	1.0	16.0	4.0	1.0

```
from sklearn.preprocessing import PolynomialFeatures

pf = PolynomialFeatures(degree=2,
                        interaction_only=False,
                        include_bias=False)
data = pf.fit_transform(data)
```

Вещественные признаки

как искать взаимодействия

- **использование предметной области**

- **перебор операций из словаря**

пример генерации признаков для сигналов

- **анализ взаимодействий признаков в моделях**

например, последовательные сплиты в деревьях

Временные признаки

```
train_d[:5]
```

	ID	Office_PIN	Application_Receipt_Date	Applicant_City_PIN	Applicant_Gender	Applicant_BirthDate	Applicant_Marital_Status
0	FIN1000001	842001	4/16/2007	844120	M	12/19/1971	M
1	FIN1000002	842001	4/16/2007	844111	M	2/17/1983	S
2	FIN1000003	800001	4/16/2007	844101	M	1/16/1966	M
3	FIN1000004	814112	4/16/2007	814112	M	2/3/1988	S
4	FIN1000005	814112	4/16/2007	815351	M	7/4/1985	M

1. Преобразование признаков

```
data[name] = pd.to_datetime(data[name], errors='coerce')
```

м.б. преобразование в вещественный признак

2. Генерация новых признаков

Каких?

Временные признаки

2.1. Характеристика момента времени:

- час, минута, секунда (=0 ?)
- время суток
- день, день недели, день года
- неделя, месяц
- время года, год
- праздник / выходной / особый день

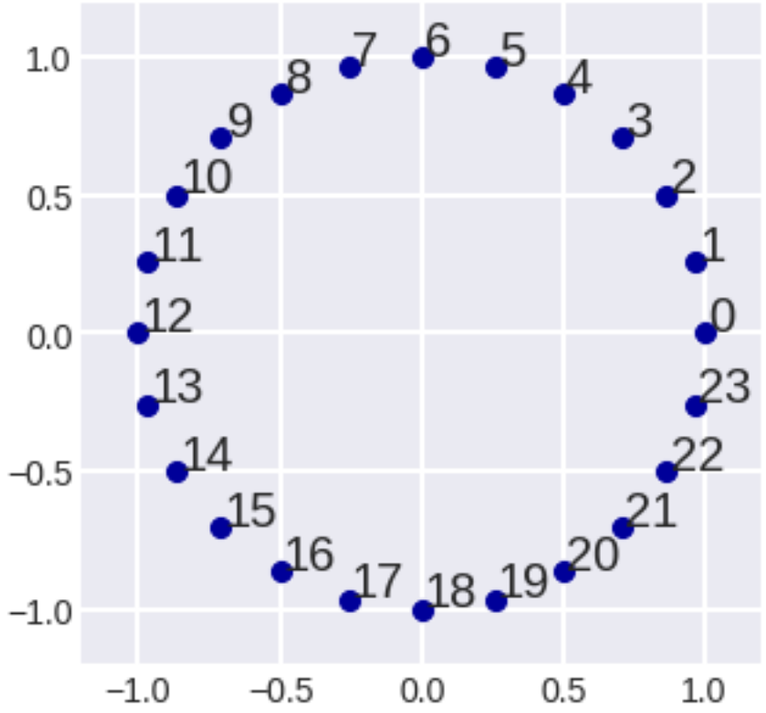
(первый понедельник месяца, начало Олимпиады)

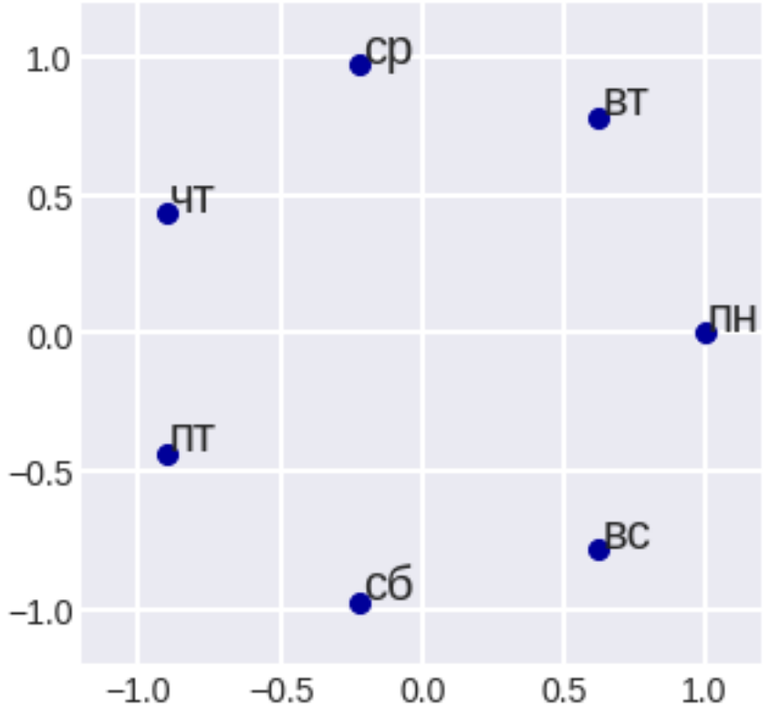
```
data[name + '_day'] = data[name].dt.day
data[name + '_dayofweek'] = data[name].dt.dayofweek
data[name + '_dayofyear'] = data[name].dt.dayofyear
data[name + '_month'] = data[name].dt.month
```

	date	date_day	date_dayofweek	date_dayofyear	date_month
0	2017-12-10	10	6	344	12
1	2016-11-13	13	6	318	11
2	2008-01-01	1	1	1	1
3	2017-05-06	6	5	126	5

Временные признаки

Кодирование циклических признаков






```
t = np.linspace(0, 2*np.pi, 8)
x = np.cos(t)
y = np.sin(t)
```

Временные признаки

2.2. Взаимодействие пары признаков

- разница времён
- в один ли день недели/год и т.п.

м.б. другое время задано неявно:

- близость к дедлайну ($T_{max} - T$)
- возраст ($T - T_{день\ рождения}$)
- сколько после/до праздника / большой покупки / регистрации

+ нормировать на некоторые шаблоны (max разность)

2.3. Использование для других признаков

- устаревание в весовых схемах
- что за последний промежуток T (операции по карте за последний месяц)
- формирование разбиения обучение / тест

Временные признаки

2.3. Использовать для генерации других признаков (продолжение)

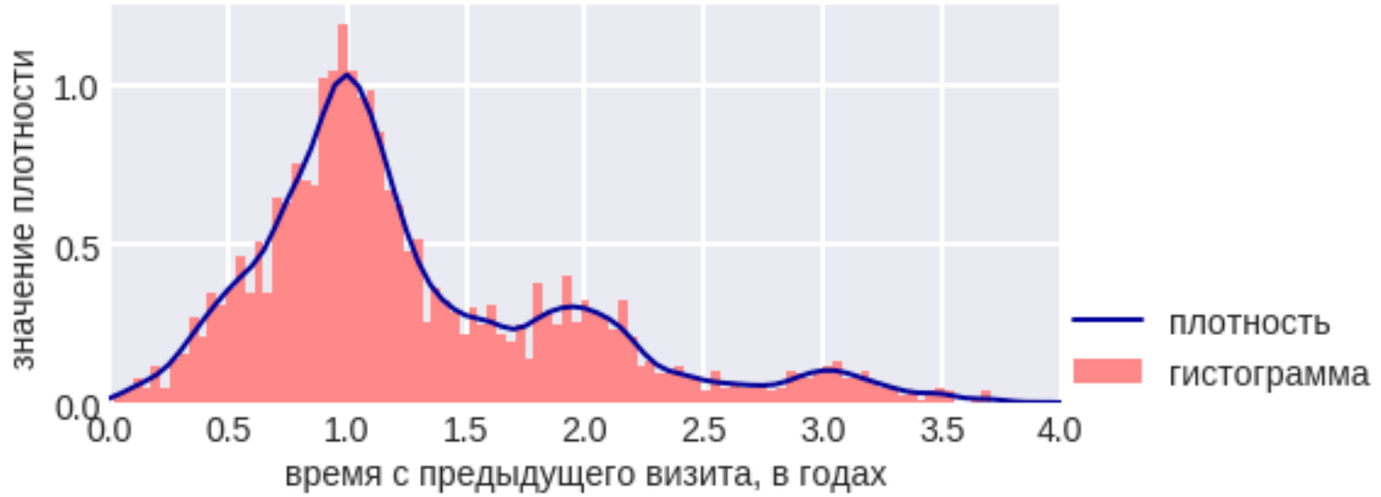
сколько транзакций в этот день

число транзакций клиента в день / число всех транзакций

какой по счёту закрыл сделку перед концом торгов

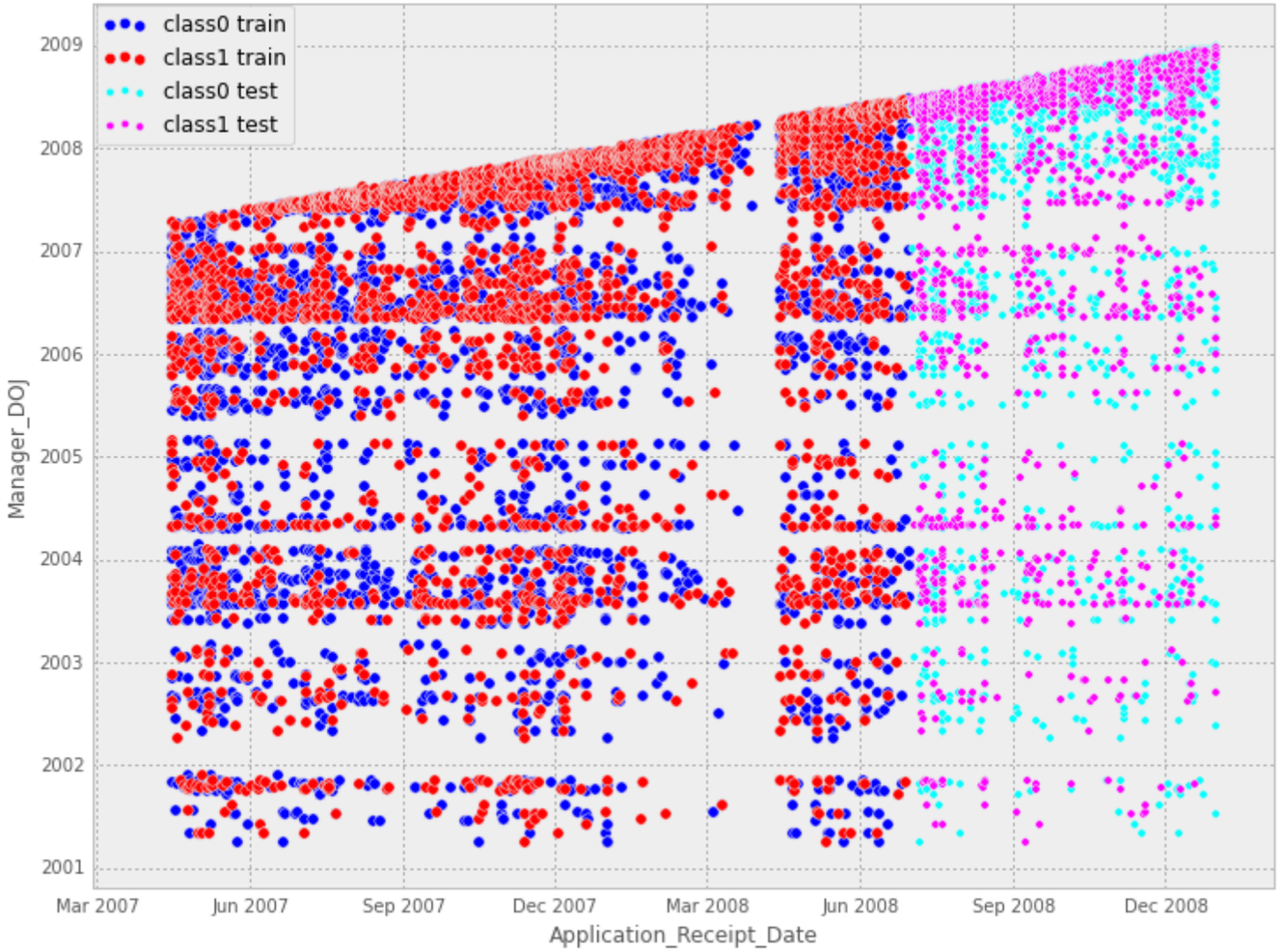
2.4. Использование для уточнения задачи, генерации признаков

Регулярность медицинских исследований:



Идея: использовать признак «ежегодный визит»

Пример признака



День сделки / когда менеджер начал работать

Разница – опыт менеджера

Временные признаки: 2.3. Взаимодействие с другими признаками

Можно смотреть на стабильность признаков!

Как меняются распределения признаков / значение целевого

Временные признаки: 2.3. Взаимодействие с другими признаками

Это позволяет:

- выявить стабильные признаки

- правильно сформировать разбиения обучение / тест

Приём: **по старой истории кодировать признаки, по новой обучать!**

Географические (пространственные) признаки: Spatial Variables

– отражают локализация в пространстве

- **GPS-координаты**
- **города**
- **страны**
- **адреса**
- **траектории / скорости перемещения и т.п.**

Можно сконвертировать в координаты

Географические (пространственные) признаки

Проекции на разные оси

Чтобы реализовывались более сложные «поверхности разделения»

Кластеризация

Чтобы выделить отдельные регионы

Идентификация, привязка, характеристики окрестности

В случае точных координат:

- где находится объект
- какие объекты также рядом (плотность объектов)
- что ещё рядом

Анализ траекторий

если изменение координат во времени

Деанонимизация данных

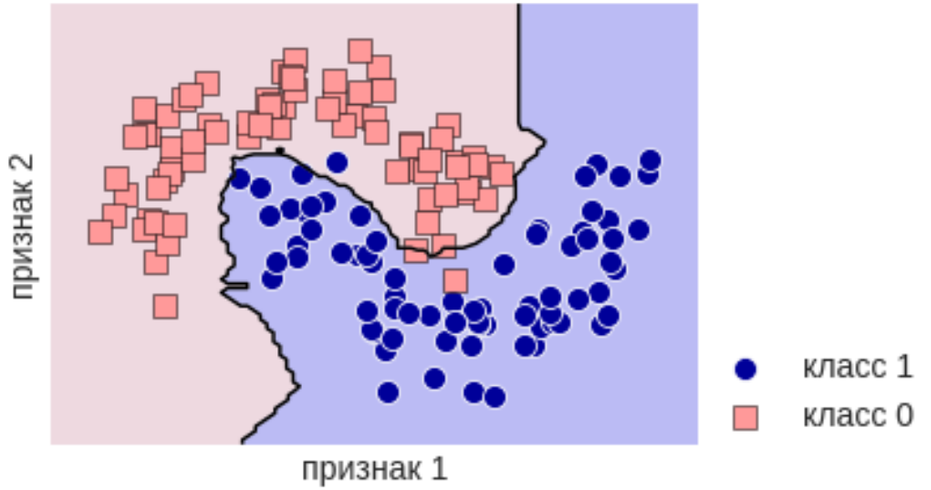
Часто география «вскрывает» местоположение

Использование контекста и исследование странностей

Телепортации, слишком частые координаты и т.п.

Проекции на разные оси

RF, число деревьев = 10



Rot + RF, число деревьев = 10

RF, число деревьев = 1000

Rot + RF, число деревьев = 1000

Географические (пространственные) признаки

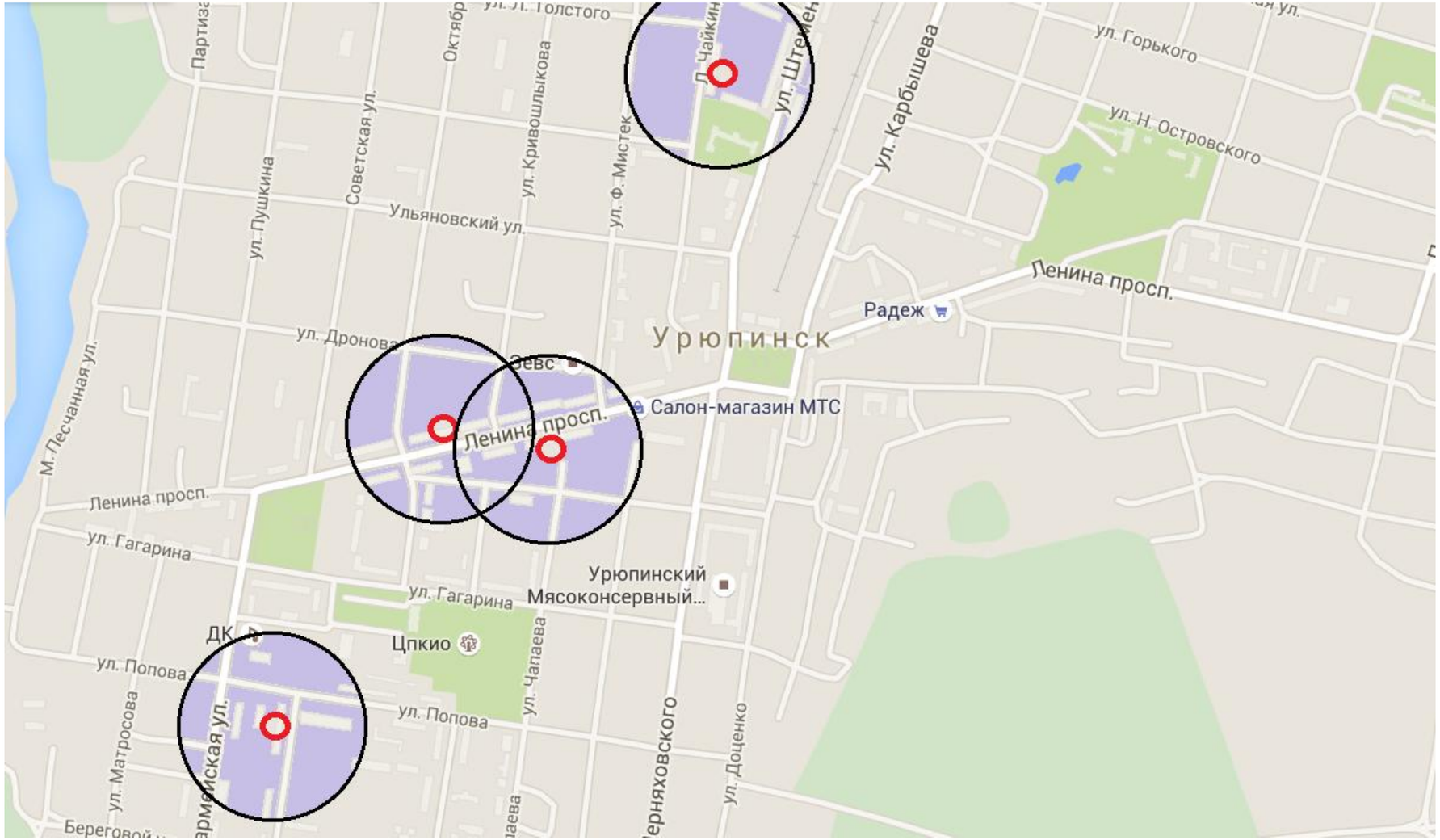
Генерация признаков

- **расстояния до**
  - **объектов** (дорога, АЗС, метро, банк, школа, остановка, магазин, город и т.п.)
  - **вычисленных объектов** (самого дорогого дома, скопления людей)
  - **границ**
  - **кластеров**

- **использовать для генерации других признаков**

средняя цена квартиры в районе, число школ в районе, плотность населения

Пример: анализ трафика и конверсии в различных точках продаж



Пример: анализ трафика и конверсии в различных точках продаж

Данные заказчика

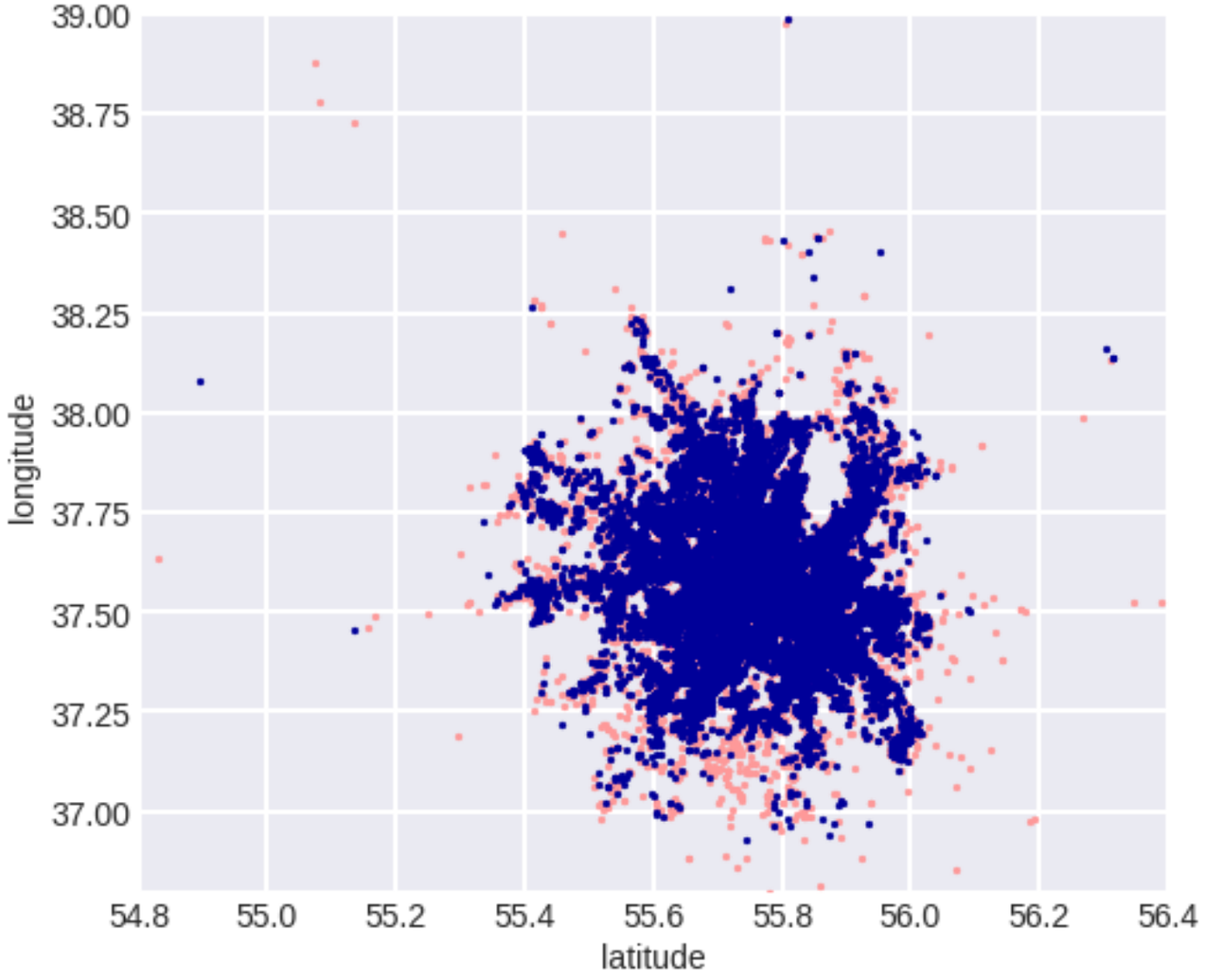
- статистика посещений
  - визиты
  - покупки/конверсия
  - ...
- данные магазина
  - площадь
  - персонал
  - категория
  - ...

Наши данные

- Анализ окрестности салона
  - наличие остановок / метро
  - конкурентов
  - где находится магазин (ТЦ)
  - численность населения

Деанонимизация данных

Достаточно очертаний множества всех координат



Финансовые признаки (деньги), количество и т.п.

- **абсолютные суммы → относительные**

ex: стоимость, площадь → стоимость кв. м.

- **остатки от деления / округления**

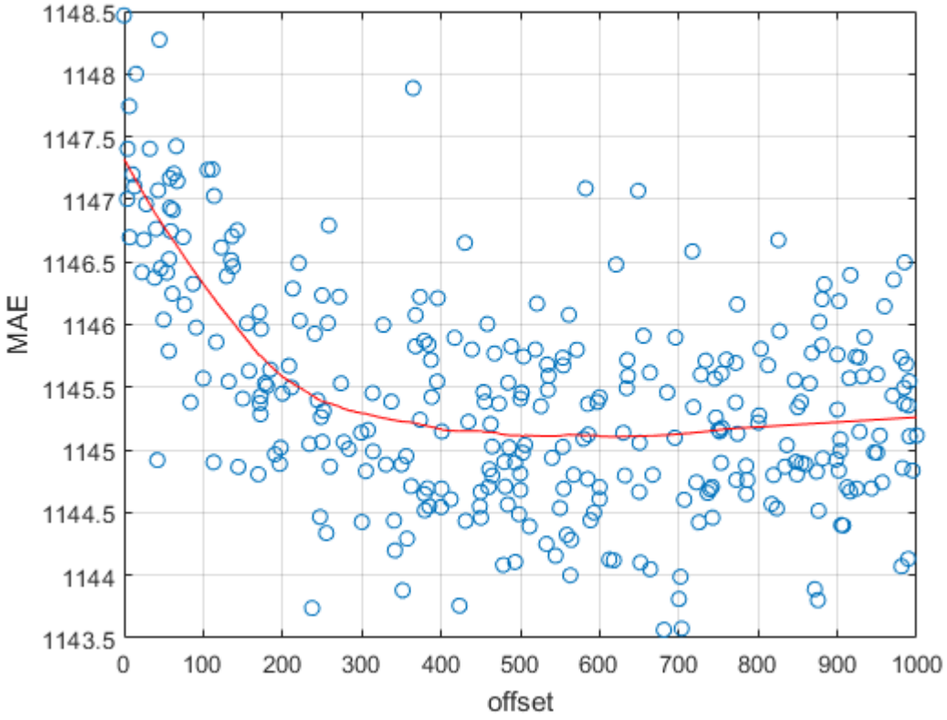
	price	f_p
0	12.50	0.50
1	0.99	0.99
2	10.00	0.00
3	18.01	0.01

позволяет отделить точные от приблизительных

- **статистики транзакций**

Генерация целевого признака

Исследование сдвига



Преобразование целевого признака log(y + offset)

Другие примеры при генерации признаков

Использование контекста

бинарные «первый/последний этаж», «балкон» и т.п.

LTV (loan to value) – отношение суммы кредита к стоимости жилья

признак «значение плотности признака»

Итог

генерация признаков…

- **из знания предметной области**
- **из EDA**
- **переборно**

учитывая особенности модели

Литература

Серия постов Understanding Feature Engineering

https://towardsdatascience.com/understanding-feature-engineering-part-1-continuous-numeric-data-da4e47099a7b

Книга по генерации признаков

Alice Zheng, Amanda Casari Feature Engineering for Machine Learning, Principles and Techniques for Data Scientists // O'Reilly Media, 2018, pp. 218

Книга по генерации признаков

http://www.feat.engineering

Курс «How to Win a Data Science Competition: Learn from Top Kagglers»

https://ru.coursera.org/learn/competitive-data-science