

Набиуллин Иван

tg: @хенорипел

Здравствуйте!

## 1. Изучение зависимостей

Первое, что я делал после предобработки данных, изучал зависимости переменных друг от друга (в каждой комбинации город - товар).

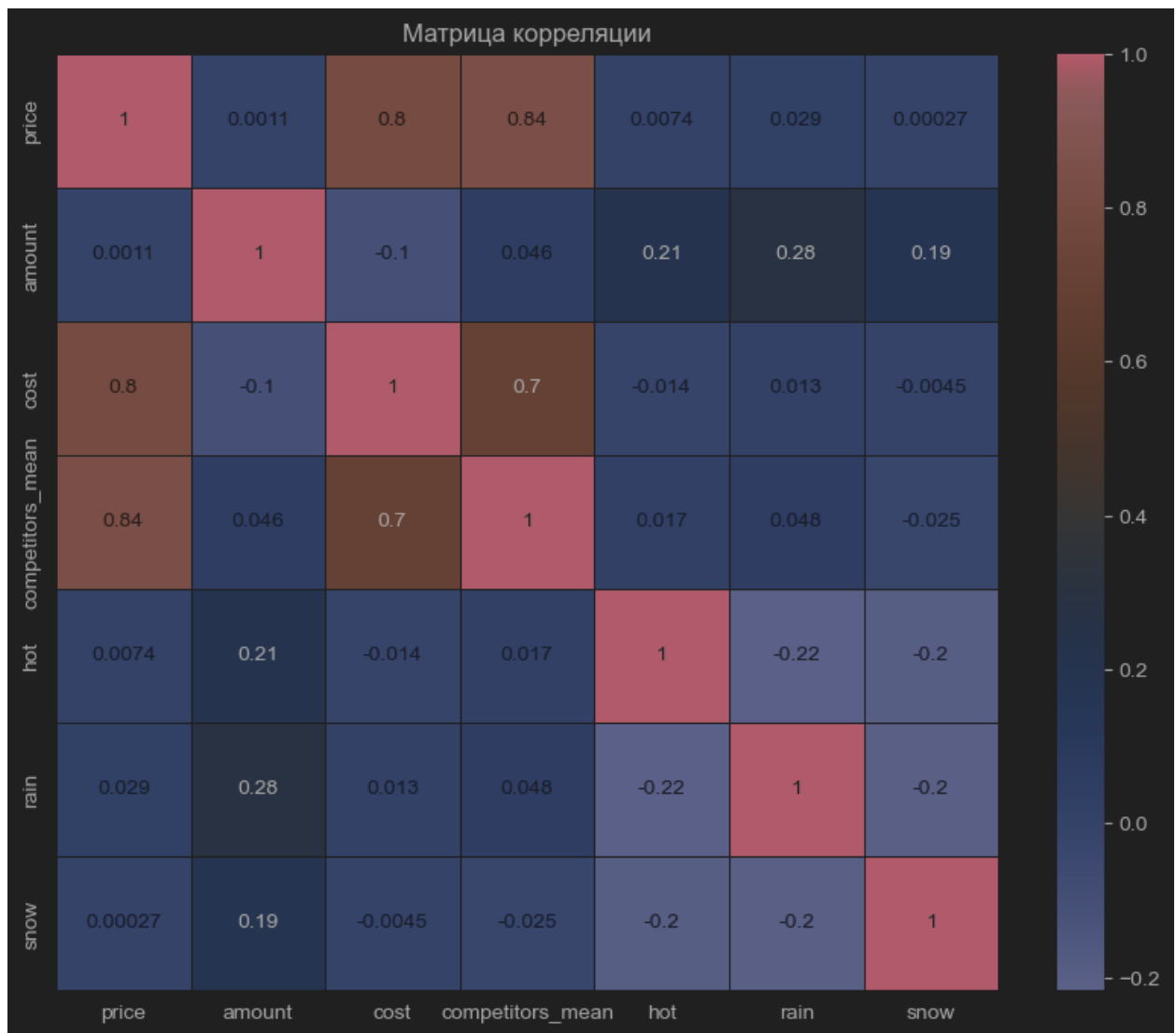


Рисунок 1 Анор Лондо – Эстус

- Цена, себестоимость и средняя цена конкурентов сильно коррелируют друг с другом.
- Объемы продаж склонны зависеть от погоды
- Цена не зависит от объема продаж напрямую

## 2. Метрики

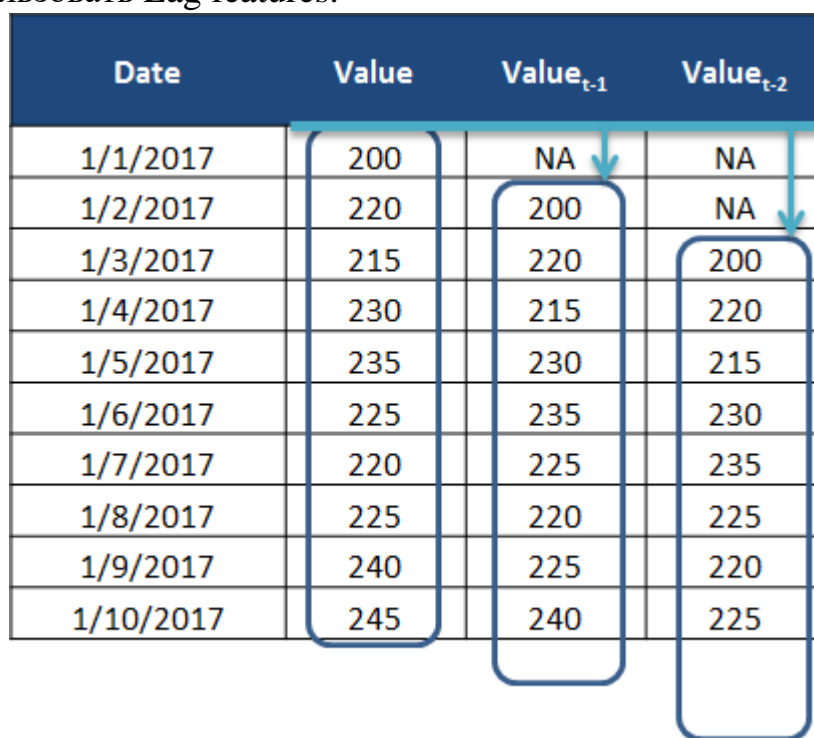
Использовал **корень** из значения, полученного кросс-валидацией по временному ряду с MSE. Корень, чтобы ошибка была в тех же единицах, в которых измеряется цена.

Так же в процессе работы пригодились RMSE и MAPE при разбиении всей выборки на train, valid и test. Так я мог строить графики по train и по valid и наблюдать за ошибками тщательнее.

## 3. Выбор модели

Первой идеей было просто построить модель регрессии через случайный лес, чтобы было с чем сравнивать. Модель дала плохие результаты с  $RMSE = 9,42$  (эту часть решил не вставлять в .ipynb файл).

Я посмотрел на графики по всему, чему можно было, и принял решение строить модель линейной регрессии, однако тут возникла проблема. Линейная регрессия не учитывает временную зависимость. Чтобы решить эту проблему я решил использовать Lag features.



Date	Value	Value <sub>t-1</sub>	Value <sub>t-2</sub>
1/1/2017	200	NA	NA
1/2/2017	220	200	NA
1/3/2017	215	220	200
1/4/2017	230	215	220
1/5/2017	235	230	215
1/6/2017	225	235	230
1/7/2017	220	225	235
1/8/2017	225	220	225
1/9/2017	240	225	220
1/10/2017	245	240	225

Сдвигать можно не один признак, и не на одну дату, поэтому на этом этапе наступил момент подбора параметров для лучших результатов.

Лучшее решение (без регуляризации), которое я нашел это сдвинуть 10 раз на [90:100] дат соответственно следующие признаки:

- Цена
- Скользящее среднее с окном в 50 (сгенерировал ранее)
- Скользящее среднее себестоимости с окном в 25

Скользящее среднее себестоимости, а не сама себестоимость, потому что данные содержали много пропусков. Я восстановил их, используя линейную интерполяцию, но, в любом случае, лучше сгладить.

```
RMSE: 0.5913
MAPE: 3.7465
CrossValidation: 0.8626
```

Рисунок 2 – RMSE и MAPE на **валидационной** выборке. Кросс валидация учитывает все.

Так же пробовал генерировать новые фичи с помощью библиотеки tsfresh, которая автоматически достает признаки из временных рядов. Результаты получились интересными, думаю она плохо работает вместе с Lag фичами, поэтому результат слегка ухудшился.

```
RMSE: 0.7062
MAPE: 4.6629
CrossValidation: 1.0036
```

Рисунок 3- Lag features + tsfresh

Результат с Lag фичей по цене, но теперь только в одном количестве. Один сдвиг на 90 дней. + tsfresh:

```
RMSE: 0.6392
MAPE: 3.9448
CrossValidation: 0.815
```

Рисунок 4 -1 Lag + tsfresh

CV лучше, но две другие ошибки хуже, и графики мне не понравились, поэтому я решил отказаться от этой комбинации параметров.

Пробовал использовать L1 и L2 регуляризации. Про L1 сразу было понятно, что все коэффициенты у лаг фичей превратятся в 0. А вот L2 было интересно посмотреть. Alpha = 50, 10 лаг фич по трем признакам, как и до этого.

```
RMSE: 0.6569
MAPE: 4.0627
CrossValidation: 0.8067
```

Рисунок 5 - Все lag + L2

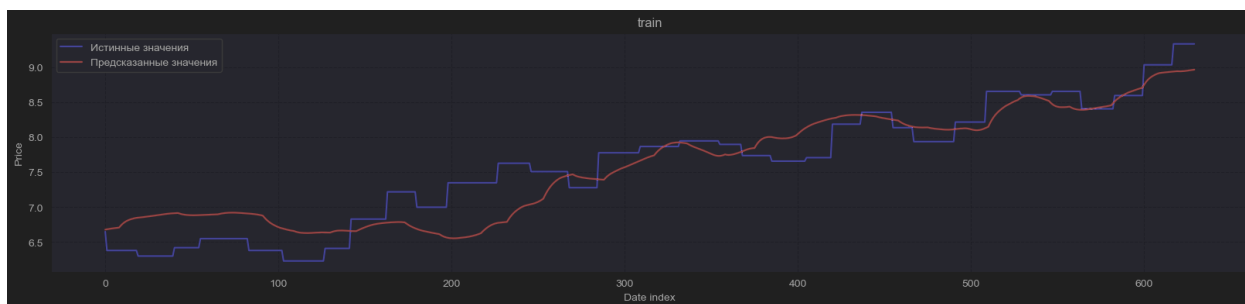


Рисунок 6 - train

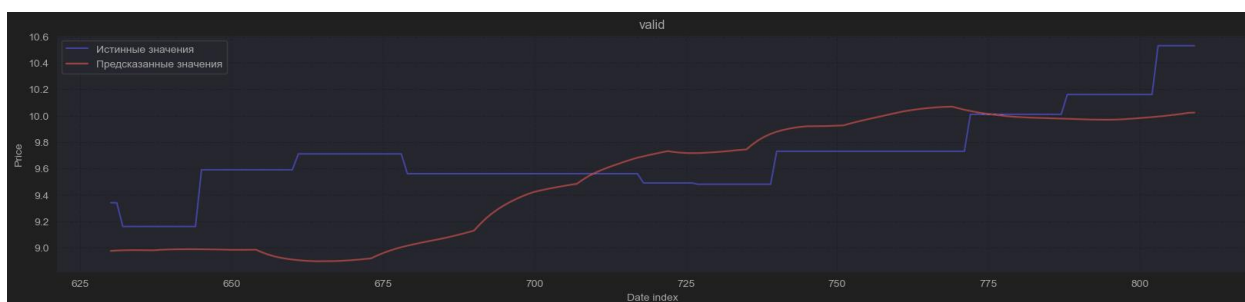


Рисунок 7 - valid

Мне понравились графики, и CV меньше, поэтому остался на этом варианте несмотря на возросшие RMSE и MAPE.

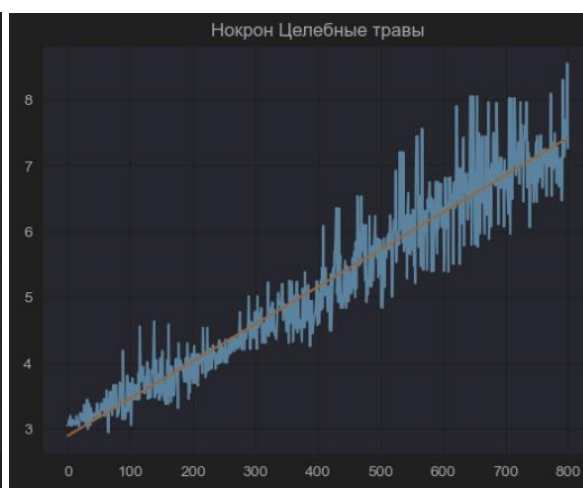
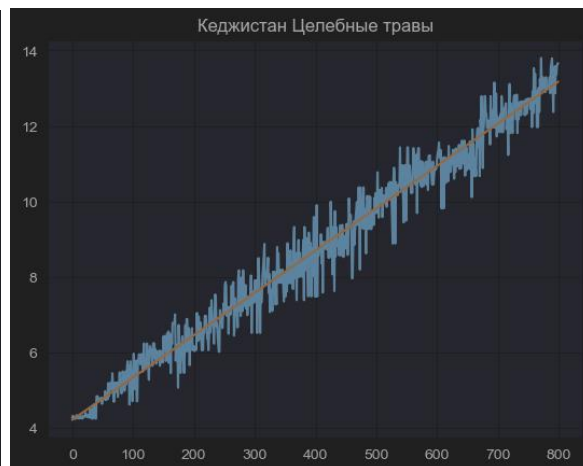
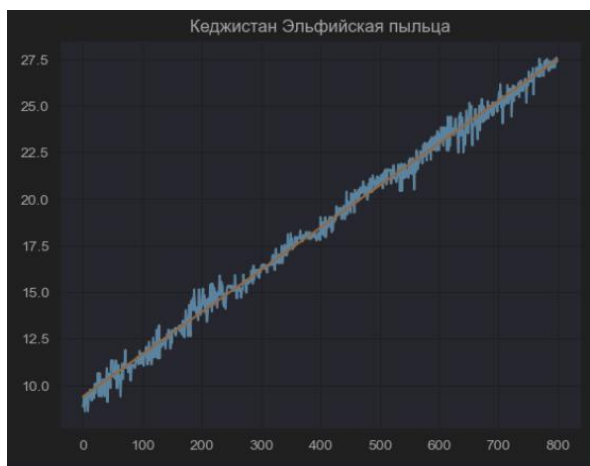
Стоит заметить, что себестоимость можно было заменить средней ценой конкурентов, т.к. у них большая корреляция между друг другом (рис.1). Вместе их использовать нельзя, появится мультиколлинеарность (я пробовал поставить их вместе и использовать *L2 регуляризацию* для уменьшения мультиколлинеарности, вышло не очень).

Подумал, что цена логичнее описывается себестоимостью, чем ценой конкурентов (конкуренты вполне могут зависеть от нашей себестоимости в данном городе на данный продукт, вдруг у нас похожие себестоимости).

## 4. Предсказание цены соперников

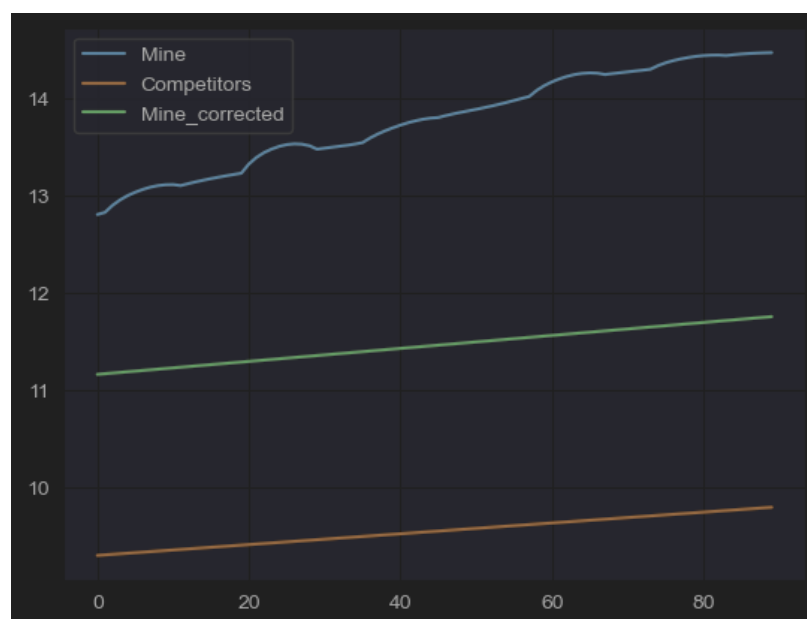
Всё бы хорошо, но есть правило о 20%, постарался придумать как его соблюсти.

Если мы посмотрим на графики средних цен конкурентов, то сразу увидим, что они растут достаточно линейно, поэтому я подумал, что для предсказания нам хватит регрессии по индексам дат.



После предсказаний цен соперников мы можем поэлементно сравнить каждое предсказание нашей цены с конкурентами, и если наше превышает конкурента больше чем на 20%, то наша цена принимает значение цены конкурента\*1.2.

Вот пример проблемного предсказания (их, на удивление было всего 2, в остальных случаях предсказания не вылезают за примерные 20%)



После этого применил функцию, которая привела все предсказания наших цен к ступенчатому виду, а также округлила до 2 серебряных (соблюдаем правила).

## **5. Что не успел реализовать.**

Я не успел предсказать количество, а его, скорее всего, получится достаточно точно предсказать, т.к. это в большинстве своем красивые временные ряды, я с общими максимумами, минимумами. У них почти везде одинаковая амплитуда, а еще, на количество продаж влияет погода (они скоррелированы положительно, я не стал предполагать, что от объемов продаж зависит погода).

Само собой, если бы я просто предсказал объемы продаж, это бы ничего не дало, нужно знать и себестоимость, но по тем данным с огромным количеством пропусков было бы тяжело что-то спрогнозировать. В любом случае, можно предсказать, предварительно сгладив кривую. И после этого можно было бы посчитать прибыль, но я не успеваю. А в целом, это конец моего отчёта.