



## Santander Customer Transaction Prediction

Решение 21/8802 места. Solo Gold медаль.

Рязанов Василий















@ryazanoff











vasily.ryazanov@phystech.edu

# Конкурсы Santander на Kaggle

- Santander Customer Satisfaction (апрель 2016, 5123 команды)
  - Шейкап на лидерборде +- 2000 мест
- Santander Value Prediction Challenge (август 2018, 4484 команды)
  - Лик
- Santander Customer Transaction Prediction (апрель 2019, 8802 команды)
  - ?

# Santander Customer Satisfaction

1	▲ 3	Shize & Nir	
2	▲ 848	kg_joi	
3	▲ 6	#1 Leustagos	
4	▲ 1	Why so noise?	
5	▲ 2024	Michael Hartman	
6	▲ 229	Noah Xiao @ Accenture	
7	▲ 4	Rolling Stones (Can't Get No [...]	
8	▲ 718	Matt Motoki	
9	▲ 4	no one	
10	▲ 1240	Bang Nguyen	
11	▲ 27	DataGeek, Kele XU & GDA	
12	▼ 4	uis	
13	▲ 266	Ouranos	
14	▲ 318	g   a	

2115	▼ 1718	Sterby	
2116	▼ 1585	Andrey Shapulin	
2117	▼ 1380	Aneesh Chinubhai	
2118	▼ 1713	Yi Jin	
2119	▼ 1523	Somshubra Majumdar	
2120	▼ 1376	waronzevon	
2121	▼ 1376	javawokr	
2122	▼ 1376	XiyaoLin	
2123	▼ 1375	AppletonTower	
2124	▼ 1372	Puppey	
2125	▼ 1549	jhamann	
2126	▼ 1589	Nelson Dinh	
2127	▼ 1525	dusj	
2128	▼ 1600	Ansup Babu	

# Santander Value Prediction Challenge

Лик (The Data "Property")  
от Giba

Данные - time-series  
одновременно и по  
колонкам и по столбцам.

ID	target	f190486d6	58e2e02e6	eeb9cd3aa	9fd594eec
7862786dc	3513333.3	0	1477600	1586889	75000
c95732596	160000.0	310000	0	1477600	1586889
16a02e67a	2352551.7	3513333	310000	0	1477600
ad960f947	280000.0	160000	3513333	310000	0
8adafb52	5450500.0	2352552	160000	3513333	310000
fd0c7cfc2	1359000.0	280000	2352552	160000	3513333
a36b78ff7	60000.0	5450500	280000	2352552	160000
e42aae1b8	12000000.0	1359000	5450500	280000	2352552
0b132f2c6	500000.0	60000	1359000	5450500	280000

# Santander Customer Transaction Prediction

Задача бинарной классификации

(90%-10%)

1) Данные

Train: 200K\*200

Test: 200K\*200

2) Метрика

ROC-AUC

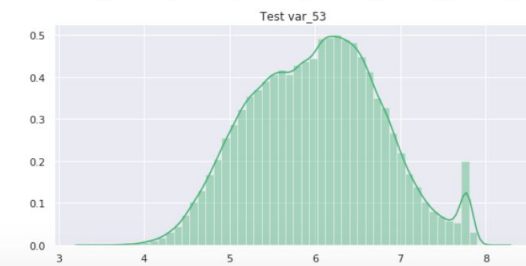
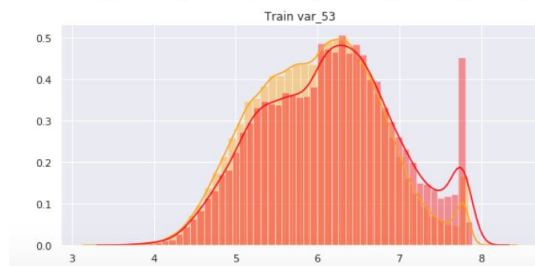
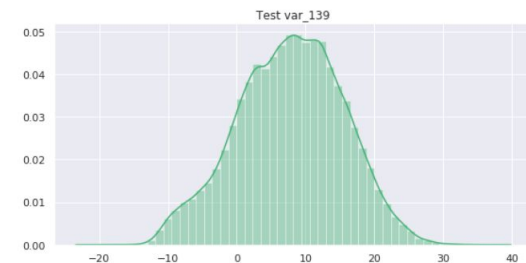
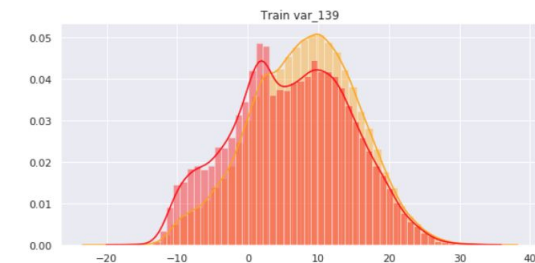
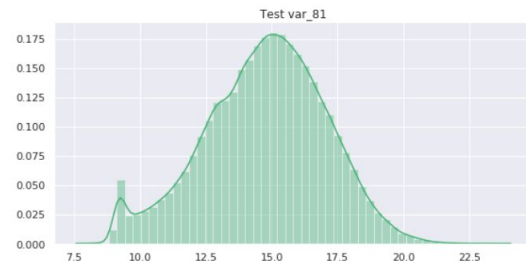
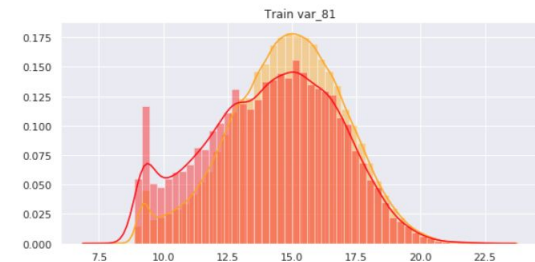
	var_1	var_2	var_3	var_4
0	-6.7863	11.9081	5.0930	11.4607
1	-4.1473	13.8588	5.3890	12.3622
2	-2.7457	12.0805	7.8928	10.5825
3	-2.1518	8.9522	7.1957	12.5846
4	-1.4834	12.8746	6.6375	12.2772
5	-2.3182	12.6080	8.6264	10.9621
6	-0.0832	9.3494	4.2916	11.1355
7	-7.9881	13.8776	7.5985	8.6543
8	2.4426	13.9307	5.6327	8.8014
9	1.9743	8.8960	5.4508	13.6043

# Данные

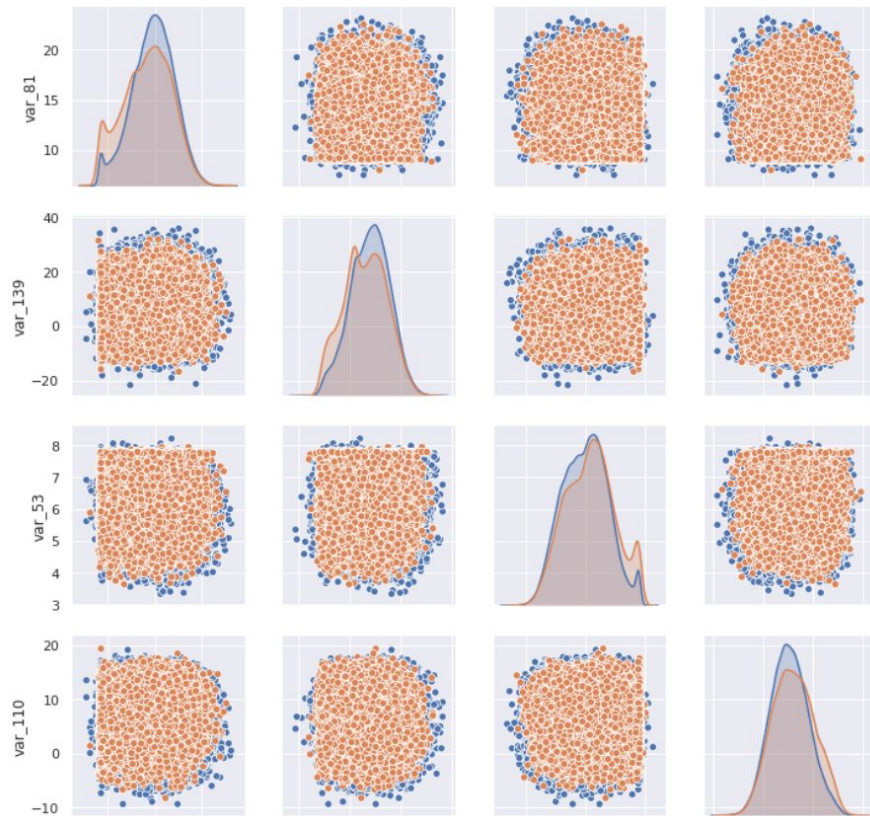
1. 200 признаков в разных диапазонах (общий [-90;90])
2. Признаки округлены до 4 знака
3. Признаки независимы (корреляции близки к 0)
4. nunique от 461 до 236483
5. Нет nan

	var_1	var_2	var_3	var_4
0	-6.7863	11.9081	5.0930	11.4607
1	-4.1473	13.8588	5.3890	12.3622
2	-2.7457	12.0805	7.8928	10.5825
3	-2.1518	8.9522	7.1957	12.5846
4	-1.4834	12.8746	6.6375	12.2772
5	-2.3182	12.6080	8.6264	10.9621
6	-0.0832	9.3494	4.2916	11.1355
7	-7.9881	13.8776	7.5985	8.6543
8	2.4426	13.9307	5.6327	8.8014
9	1.9743	8.8960	5.4508	13.6043

# Данные



# Данные





# Baseline

1. Сырые признаки

# Baseline

1. Сырые признаки
2. Разности признаков

# Baseline

1. Сырые признаки
2. Разности признаков
3. row-wise статистики (sum, mean, min, max, ...)

# Baseline

1. Сырые признаки
2. Разности признаков
3. row-wise статистики (sum, mean, min, max, ...)
4. mean target на признаках

# Baseline

1. Сырые признаки
2. Разности признаков
3. row-wise статистики (sum, mean, min, max, ...)
4. mean target на признаках
5. стекинг lr, knn, bayes, lightgbm

# Baseline

1. Сырые признаки
2. Разности признаков
3. row-wise статистики (sum, mean, min, max, ...)
4. mean target на признаках
5. стекинг lr, knn, bayes, lightgbm
6. **ROC\_AUC 0.899**



# Baseline

1. Сырые признаки
2. Разности признаков
3. row-wise статистики (sum, mean, min, max, ...)
4. mean target на признаках
5. стекинг lr, knn, bayes, lightgbm
6. Тюнинг гипер параметров на основе форума
7. **ROC\_AUC 0.9**



## Состояние leaderboard на тот момент

1. 2-4 команды со скором 0.904+
2. 5-10 команд со скором 0.901
3. По несколько тысяч команд 0.9 и ниже



# Состояние форума на тот момент

**A part of the magic?**

**CoreyLevinson** 24 days ago

**Is this magic ?**

**tarobxl** 25 days ago

**Discover The Magic Of Santander**

**Darko Androcec** a month ago

**Still looking for magic : Timestamp**

**François** 23 days ago

**"Shuffling the features" seems the key magic**

**RunningZ** a month ago

**I found some magic features by some way**

**J MA** a month ago

# Некоторые открытия на тот момент

1. Всем стало понятно что есть лик (“магия”)
2. Оказалось что признаки можно перемешивать в пределах таргета

```
df_train0[c] = df_train0[c].sample(frac=1, random_state=rs_).values  
df_train1[c] = df_train1[c].sample(frac=1, random_state=10*rs_+1).values
```



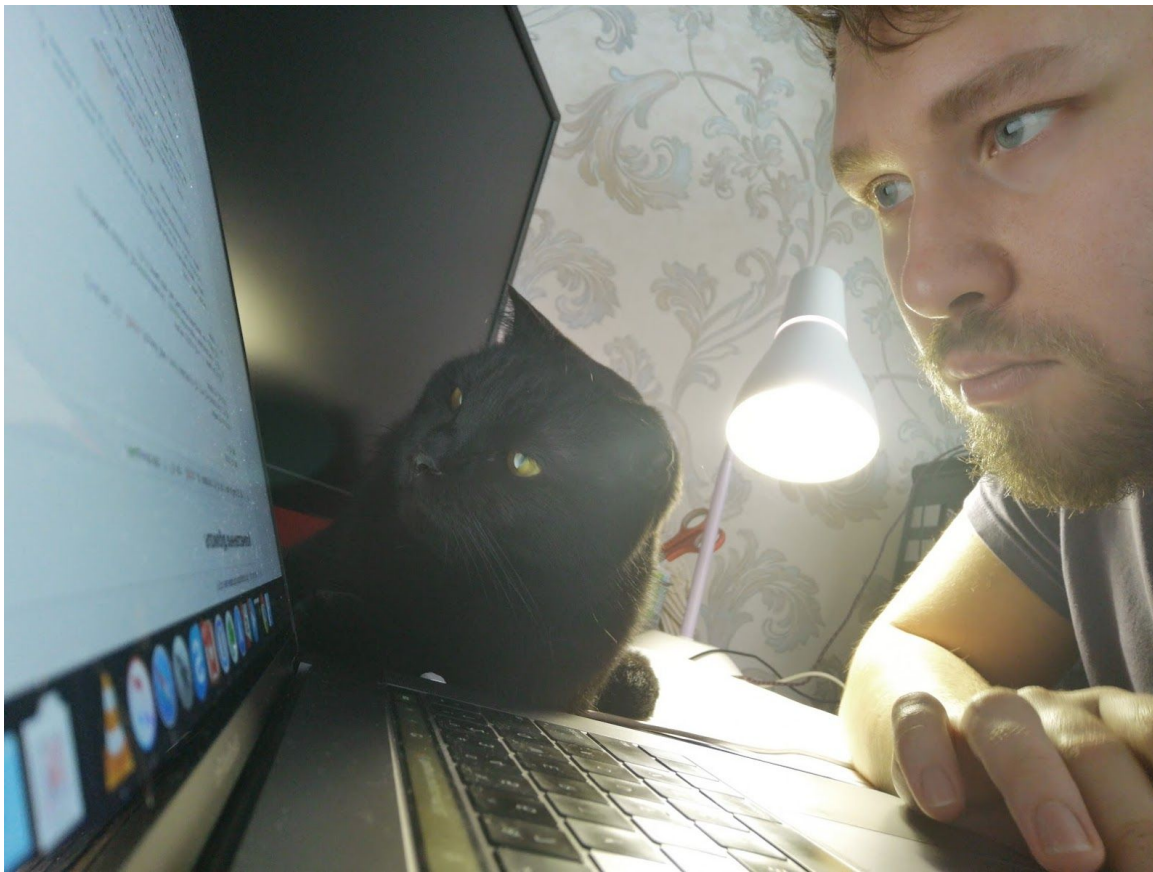
Но это не была “магия”

# EDA и попытки найти магию

1. PCA, t-SNE
2. Denoising Autoencoder
3. Поиск категориальных признаков
4. Поиск time-series внутри признака и признаков
5. Важность признаков
6. Перемешивание, аугментация
7. Анализ дубликатов
8. Сортировки, кумулятивные суммы

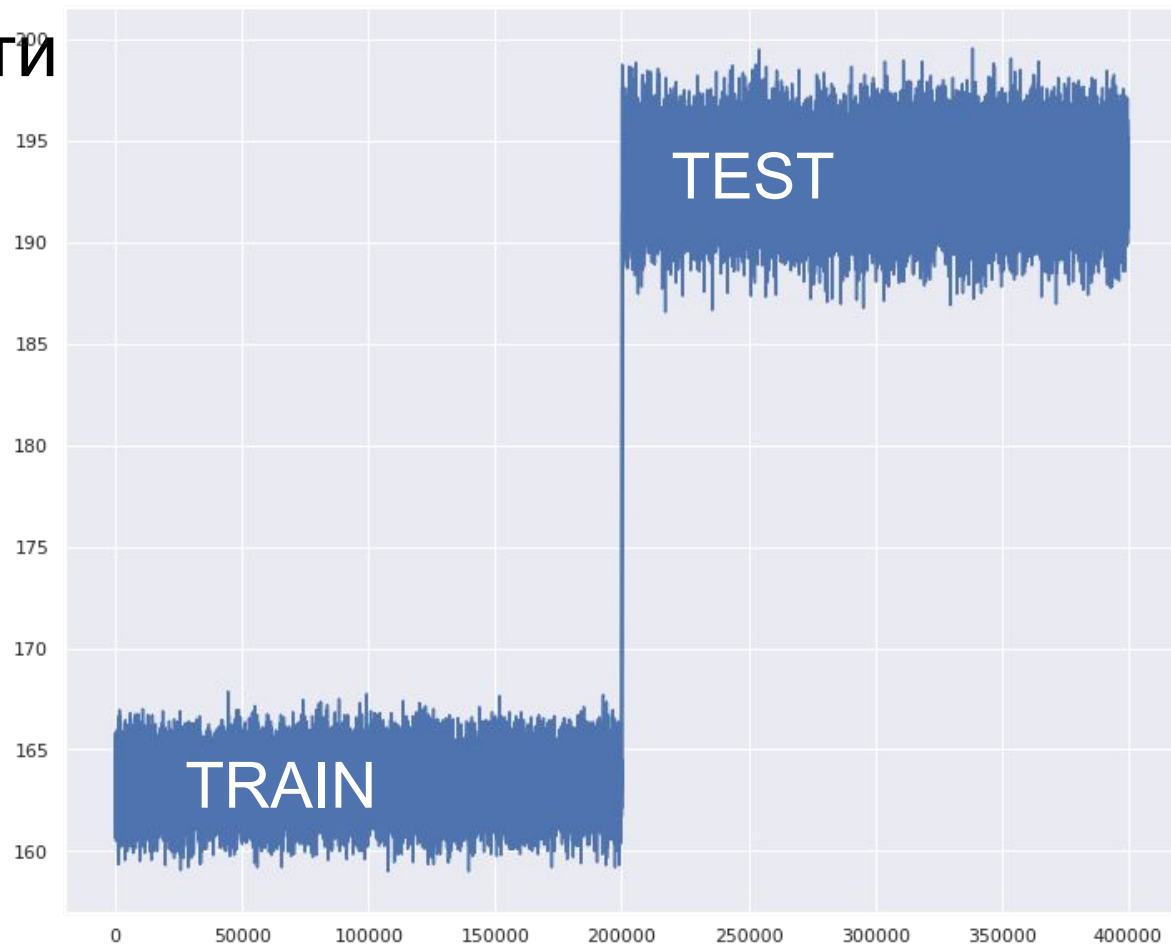
... и многое-многое другое

# Процесс поиска магии



# Первые странности

rolling mean количества  
дубликатов по  
признакам объекта в  
train/test



# Количество дубликатов\* внутри объекта

200K объектов train и **100K** объектов теста - нормальное распределение (mean около **160**)

100K объектов теста - ровно **200**!

В этом было что-то полностью неестественное и я начал копать дальше

\*кол-во фичей у которых значение встречалось в других объектах

# Некоторые открытия

1. 100К с 200 повторяющимися значениями полностью не попадали в Public (я думал это Private)
2. 100К с 200 повторяющимися значениями были составлены из значения другой половины теста (думал это Public)
3. Становится ясно зачем добавили фейки (чтобы на сырых данных частотные статистики не работали)

# EDA и попытки найти магию

1. PCA, t-SNE
2. Denoising Autoencoder
3. Поиск категориальных признаков
4. Поиск time-series внутри признака и признаков
5. Важность признаков
6. Перемешивание, аугментация
7. **Анализ дубликатов**





# 0.901 Leaderboard

200 новых признаков

1) По трейну:

```
df_train[c].duplicated(keep=False)
```

2) По тесту:

```
df_test[c].isin(df_train[c].unique())
```

Распределения получались одинаковые.

## 0.904 - 0.914 Leaderboard

- 200 сырых признаков
- 200 признаков-дубликатности
- 200 признаков частотности (посчитанные по train)
- 200 признаков с заменами:
  - Если значение - дубликат - оставляем сырое
  - Если значение уникальное - заменяем на среднее по признаку

# Самый важный Kernel



## 🔖 List of Fake Samples and Public/Private LB split

Python notebook using data from [Santander Customer Transaction Prediction](#) · 12,824 views

Основная идея:

- Ищем 100K фейков по алгоритму ранее
- Ищем кандидаты из которых строились фейки
  - По каждому значению фейка ищем объекты у которых такое значение встретилось 1 раз
  - Следовательно это должен быть объект-генератор

Получаем разбиение 50K/50K/100K!

Проверкой убеждаемся что это Public/Private/Fake(?)

## 0.922 Leaderboard

1. Я пересчитал статистики по 200K train + 100K real test
2. Все больше людей начало отрывать магию и получать результаты 0.901-0.914

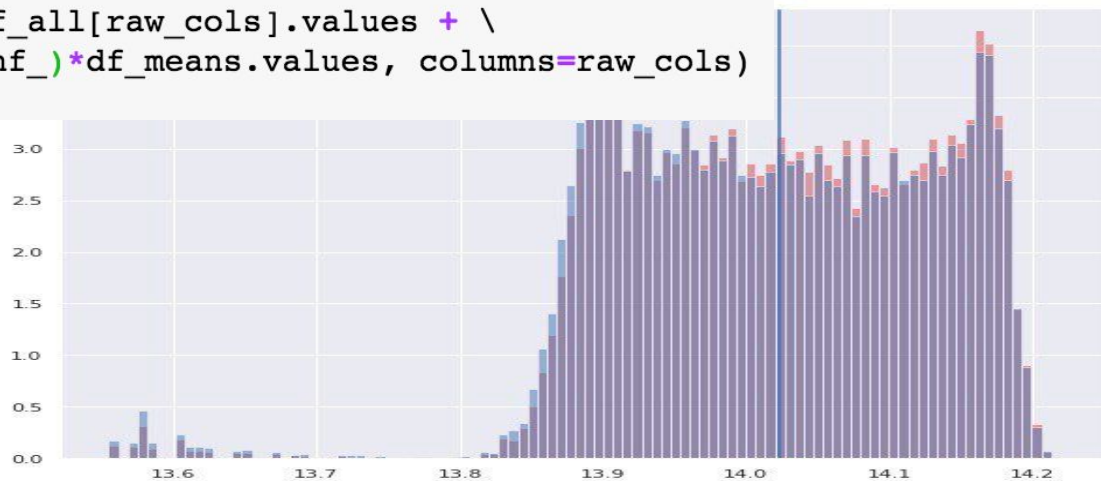
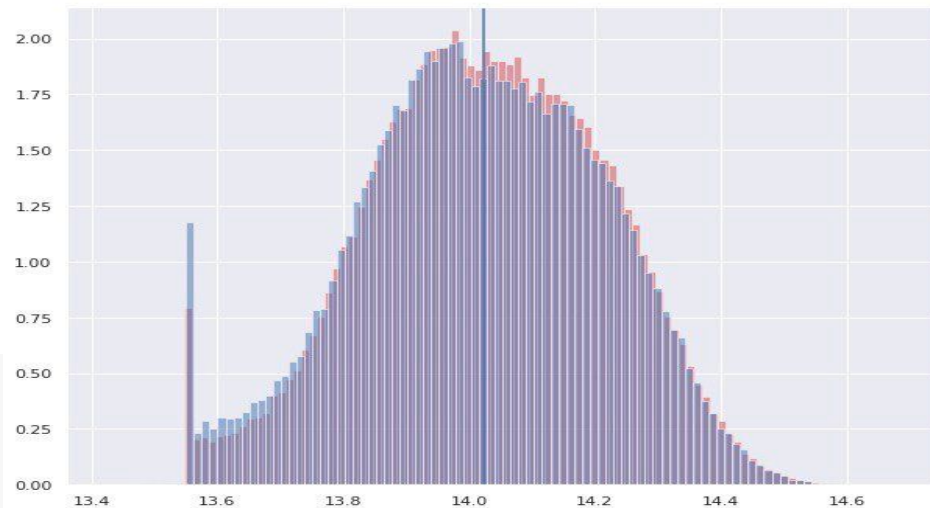
# Сдвиги к среднему

Исходное распределение

```
mms = MinMaxScaler()  
df_vc_nf_ = df_vc_nf.copy()  
df_vc_nf_ = mms.fit_transform(df_vc_nf_)  
df_vc_nf_ = df_vc_nf_**(1/4)
```

```
vals_shift = pd.DataFrame(df_vc_nf_*df_all[raw_cols].values + \  
                          |(1 - df_vc_nf_)*df_means.values, columns=raw_cols)
```

Распределение после  
сдвига



## 0.924 признаки

- 200 сырых фичей
- 200 фичей-дубликатов (0/1)
- 200 фичей-частот
- 200 фичей с заменами уникальных значения на среднее
- 200 фичей со сдвигом к среднему пропорционально частоте
- 200-600 фичей с заменами на среднее по определенным порогам частоты
- Некоторые row-wised статистики по частотам

## 0.924 модель

- Отбор признаков по Shap Importance (-700 признаков)
- Аугментация train на фолдах (x1, x2, x3)
- Усреднение рангов CTB, LGB, разных аугментаций, сидов фолдов.

### Реализации бустинга:

- LightGBM - быстро и хорошо
- CatBoost - медленней и хуже
- Xgboost - медленно и плохо, убивал kernel :(

# Железо

- Workstation 64RAM + i7 + 2 GPU
  - Для обучения основных моделей
- Ноутбук с 32RAM
  - для EDA
- Под конец Memory Optimized AWS EC2
  - для монотонных генераций признаков



# Что не хватило для победы?


- Выдохся, не хватало воображения
- Использование нейронок и блендинг с ними (использовали в top10)
- Модель на простых фичах (хинт 4 места)
  - Table 4 columns: [value, count, var number, target], \*var number \* in range 0-199.
  - 40M of rows. Training on this table with LGBM you achieve not so big AUC around ~0.530
  - Reshape and get ~0.925 CV using just product of rows.

Сборник золотых решений:

<https://www.kaggle.com/c/santander-customer-transaction-prediction/discussion/88926>

# Хинт с разворачиванием таблицы

- Table 4 columns: [value, count, var number, target], \*var number\* in range 0-199.
- 40M of rows. Training on this table with LGBM you achieve not so big AUC around ~0.530
- Reshape and get ~0.925 CV using just product of rows.

200K	target	var_0	var_1	var_2		var_num	var	target	200K*200 = 40M
	0	5.7154	-1.2824	10.781100		0	5.715400	0	
	0	11.3034	6.1088	17.723301		0	11.303400	0	
	0	6.8898	0.8730	12.859800		0	6.889800	0	
	1	11.1349	7.0059	11.175100		0	11.134900	1	
	0	14.2565	-5.3368	12.220100		0	14.256500	0	
	0	10.4030	-5.9898	15.166200		0	10.403000	0	
	0	8.8095	-5.5346	12.791400		0	8.809500	0	
	0	10.8458	0.4223	14.916100		0	10.845800	0	
	0	9.6543	-0.6606	12.334600		0	9.654300	0	
	0	8.7797	3.5260	13.280600		0	8.779700	0	
						1	-1.282400	0	

Что были за данные и почему “магия” работала?



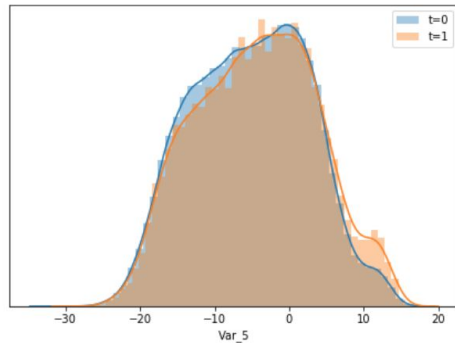
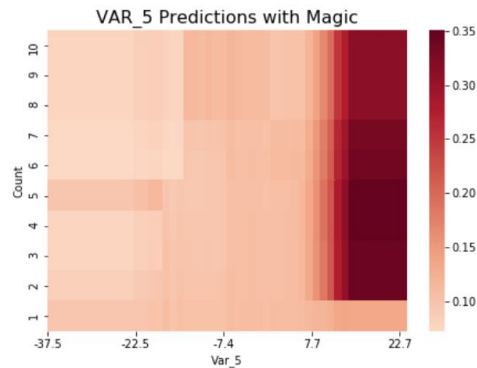
# Данные

1. Данные синтетические, скорее всего сгенерированы из распределения
2. Либо реальные данные перемешали неким образом
3. Это может быть связано с тем что в Европе запрещают анализировать данные клиентов, но разрешают анализировать “похожие” данные
4. Либо Santander боится за свои данные

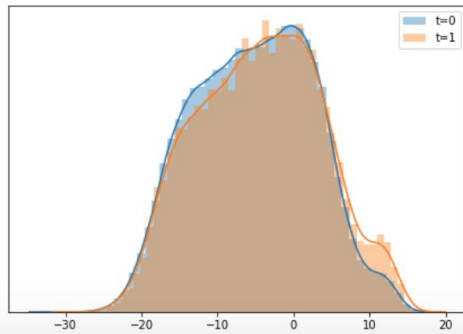
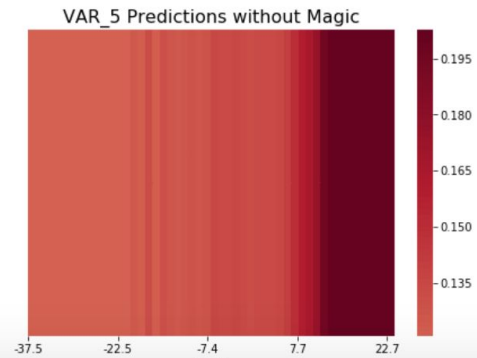
# Почему магия работает

1. Возможно результат семплирования и округления до 4 знаков
2. <https://www.kaggle.com/c/deotte/200-magical-models-santander-0-920>
3. Никто так до конца и не понял :(

VAR\_5 with magic val\_auc = 0.5208



VAR\_5 without magic val\_auc = 0.51441



# Некоторые заключительные советы

1. Делайте коммиты
2. Используйте tqdm и tg\_tqdm, логируйте в телеграм
3. Используйте SHAP чтобы понять/отобрать признаки
4. Добавляйте скор и время в название файла
5. Читайте Discussion первых мест LB
6. Можно чекать Github
7. Не выбирайте рандомных тиммейтов

# Делайте коммиты

1. Легко отправить код организаторам
2. Легко найти место с наилучшим решением и не запутаться
3. Добавляйте в message текущий скор

```
import os
def make_commit(mes, filename='baseline.ipynb'):
    os.system("git add {}".format(filename))
    os.system("git commit -m '{}'.format(mes))
    os.system("git push origin master")
```

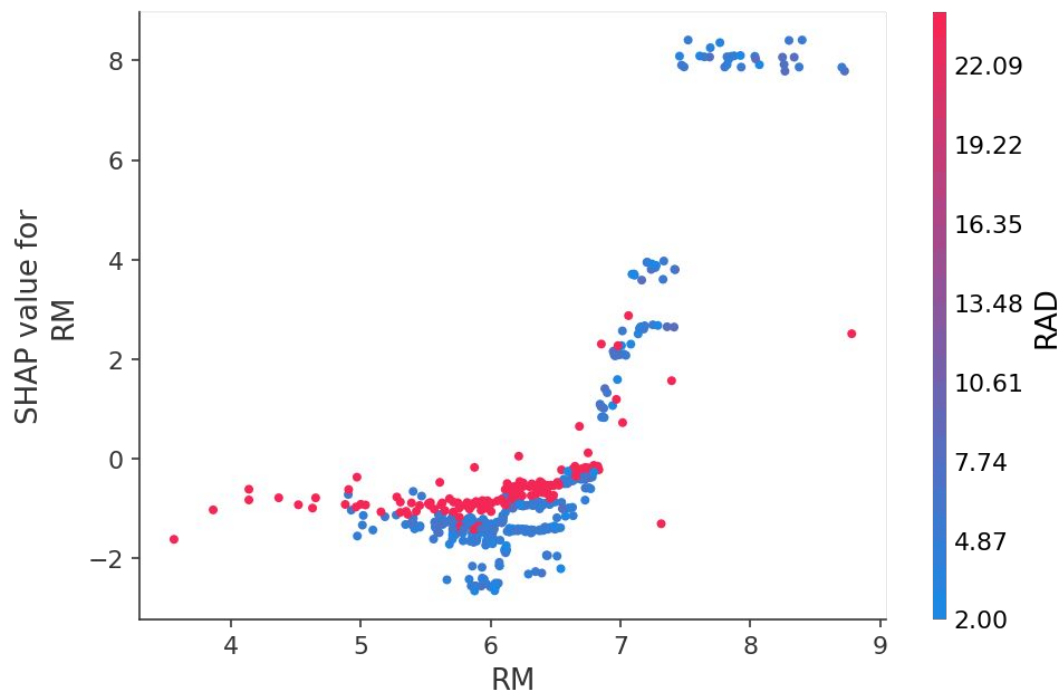
# tqdm / Telegram / логирование

1. tqdm <https://github.com/tqdm/tqdm>
2. tg\_tqdm [https://github.com/ermakovpetr/tg\\_tqdm](https://github.com/ermakovpetr/tg_tqdm)
3. telepot <https://github.com/nickoala/telepot>



# SHAP





<https://github.com/slundberg/shap>




# Добавляйте скор/время в submission

```
COMMENT = 'MEAN SCORE {} \t NFOLD:{}'.format(np.array(scores).mean(), nfold)
ts = time.time()
st = datetime.datetime.fromtimestamp(ts).strftime('%m_%d_%H_%M_%S')
ss.to_csv('../submissions/{}_{}.csv'.format(st, COMMENT), index=None)
```

# Discussion лидеров

1	—	Wizardry		0.92573
2	—	三人寄れば文殊の知恵（本当か？）	  	0.92552




### Silogram

Freelance data scientist at Self-employed  
Zurich, Switzerland  
Joined 6 years ago · last seen in the past day  
[in](#)




Followers 1394  
Following 40


**Competitions Grandmaster**

[Home](#) [Competitions \(74\)](#) [Kernels \(4\)](#) [Discussion \(179\)](#) [Datasets \(1\)](#) [Followers \(1,394\)](#) [Contact User](#) [Follow User](#)




**Competitions Grandmaster** 


**Rank**  
**4**  
of 109,601

 15  28  11




**Kernels Contributor** 

**Unranked**

 0  0  1

**Discussion Expert** 

**Current Rank** **Highest Rank**  
**21** **11**  
of 91,096

 24  28  82

# Проверка Github

The screenshot shows the GitHub search interface. The search bar at the top contains the text 'santander customer'. Below the search bar, the left sidebar displays statistics for the search results: Repositories (355), Code (23K), Commits (166), Issues (85), Marketplace (0), Topics (0), Wikis (23), and Users (0). The main content area shows 355 repository results. The first result is 'mohammed-alayyaf/Santander-Customer-Transaction-Prediction', which is a Jupyter Notebook. The second result is 'dmitrikuksik/santander-customer-transaction-prediction', also a Jupyter Notebook. A red circle highlights the search bar, and another red circle highlights the 'Sort: Recently updated' dropdown menu. Red arrows point from the search bar and the sort menu to the first result.

santander customer

Pull requests Issues Marketplace Explore

Sort: Recently updated

355 repository results

**mohammed-alayyaf/Santander-Customer-Transaction-Prediction**  
My Attempt for Kaggle's *Santander Customer* Transaction Prediction Competition  
Updated 2 hours ago

**dmitrikuksik/santander-customer-transaction-prediction**  
Kaggle competition: *Santander Customer* Transaction Prediction  
Updated 23 hours ago

Jupyter Notebook

Jupyter Notebook

Repositories	
Repositories	355
Code	23K
Commits	166
Issues	85
Marketplace	0
Topics	0
Wikis	23
Users	0

Languages	
Jupyter Notebook	155
Python	77

Обычно слабо отличается от публичных Kernel

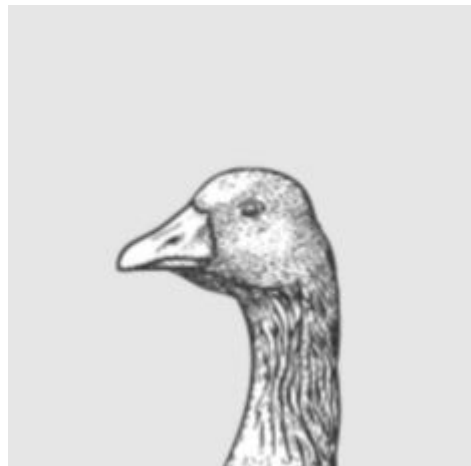
# Про выбор тиммейтов

Если хоть один тиммейт нарушил правила - вся команда лишается наград! В Santander человек так лишился золота.

Нарушения:

- 1) Несколько аккаунтов
- 2) Приватный обмен кодом

Поэтому не берите в команду случайных





Рязанов Василий  
@ryazanoff  
vasily.ryazanov@phystech.edu