DATA SCIENCE GAME

Qualifiers + Final Phase

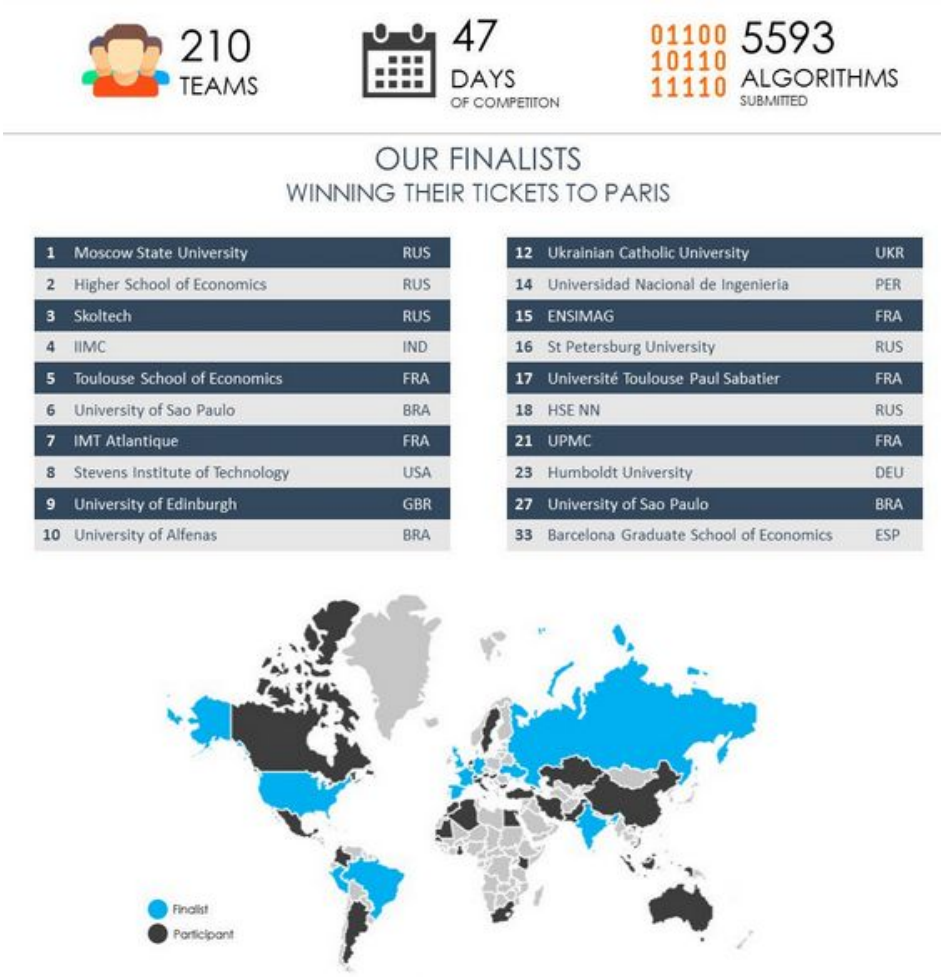
Luis Moneda, Data Scientist at Nubank

Overview

- Data Science Game 2017
- Team
- Qualifiers & Finals
  - The challenge
  - Data
  - Our strategy
  - What we could have done better?
  - Other teams approach (14th and 2nd)
  - Take away

Data Science Game

- Annual student competition since 2014;
- Teams composed of 4 students from the same university (Phd, master or undergraduate);
- 2 phases;
- Hosted in Kaggle;
- Organized by students from France;
- Data from french companies;



Team

- Name: Team Maia!
- University: USP São Paulo
- Members:
  - Luis Moneda
  - Pedro Cicolin
  - Arthur Lacerda
  - Wanderson Ferreira
  - Paulo Castro
- Qualifiers: 27th from 210
- Finals: 12th from 21

Sponsors:

serasa experian

DIGESTO

Big Data

The challenge

- Deezer play registers data:
  - Information about user, song, service and a target “is_listened”;
- **Goal**: Predict the probability of listening a song suggested by the “flow” service;
- Metric: AUC;
- The submission set consist of one register per unique user in the data, their last flow play;
- We need to learn how to relate user historical data and his probability of listening a certain song in the flow in the future;

Data

- media_id - identifiant of the song listened by the user
- album_id - identifiant of the album of the song
- Genre_id - identifiant of the song genre
- context_type - playlist, album...
    - release_date - release date YYYYMMDD
    - ts_listen - timestamp of the listening in UNIX time
    - platform_name - type of os
    - artist_id - identifiant of the artist of the song

	genre_id	ts_listen	media_id	album_id	context_type	release_date	platform_name
0	25471	1480597215	222606	41774	12	20040704	1
1	25571	1480544735	250467	43941	0	20060301	2
2	16	1479563953	305197	48078	1	20140714	2
3	7	1480152098	900502	71521	0	20001030	0
4	7	1478368974	542335	71718	0	20080215	0

Data II

- media_duration - duration of the song
- user_gender - gender of the user
- user_id - anonymized id of the user
- platform_family - type of device
- user_age - age of the user
- artist_id - identifiant of the artist of the song
- is_listened - 1 if the track was listened, 0 otherwise
- listen_type - if the songs was listened in a flow or not

platform_family	media_duration	listen_type	user_gender	user_id	artist_id	user_age	is_listened
0	223	0	0	9241	55164	29	0
1	171	0	0	16547	55830	30	1
1	149	1	1	7665	2704	29	1
0	240	0	1	1580	938	30	0
0	150	0	1	1812	2939	24	1

Data III

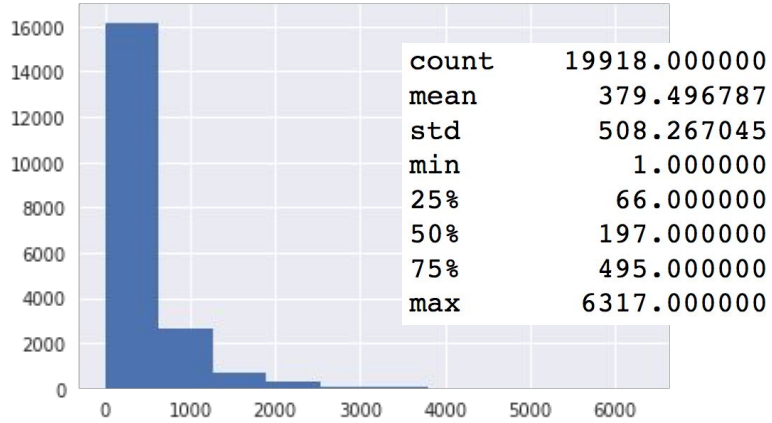
- Further Media Info:
  - {**"media_id"**:213952,
    "sng_title":"Maria Cristina",
    "alb_title":"El Son de Cuba",
    "art_name":"Septeto Nacional De Ignacio Pineiro"}
    {**"media_id"**:223014,
    "sng_title":"Love stealer",
    "alb_title":"Sounds from the fourth world",
    "art_name":"Calvin Russell"}

- API Information (Album, Artist and Song):
  - Artist_albums: num of released albums
  - Artist_fans: num of fans in deezer
  - Artist_radio: if it has a radio in deezer
  - Bpm: song bpm
  - Song_rank: song ranking

Overall Strategy

- Create as many features as possible;
- Validate using the last flow execution for each user (closer to the submission set than random sampling);
- Use simpler models to users with few registers;
- Blend different models results (different sets of features)
- Approaches:
  - (Benchmark): mean target for flow mode;
  - Random Forest;
  - XGBoost with all features;
  - XGBoost with basic features;
  - Blend the predictions;

User distribution x number of registers



Feature Engineering I

- Transform the original features using:
  - Difference between release date and listening date (in days, months and years);
  - Expansion on release date: day (hehe), month and year;
  - Day of week (mon, tue, wed..) and period of day (morning, afternoon) the user is listening;
  - Binning for release date (song from 70s, 80s..) and user age
  - Difference between song and user age;

Feature Engineering II

- User-specific features:
  - Successful reproduction in flow and non-flow mode;
  - Proportion between normal and flow registers;
  - How many std above the mean for flow registers;
  - How many platform_family and platform_name the user uses;
  - Maximum number of executions of the same song;
  - How many different songs he has listened during training time (flow and non-flow);
  - How many different artists, genres, difference decades;
  - Mean / std of age, duration, bpm and rank from listened and not listened songs;
  - Proportion of playlist execution;
  - Do all the above for flow and non-flow;

Modeling

Two kinds of model

- Performance (random sample):
  - 0.861694 (w/ user-specific features)
  - 0.8262 (general features)
- Score:
  - Public: 0.6423
  - Private: 0.6393

After blending predictions scores:

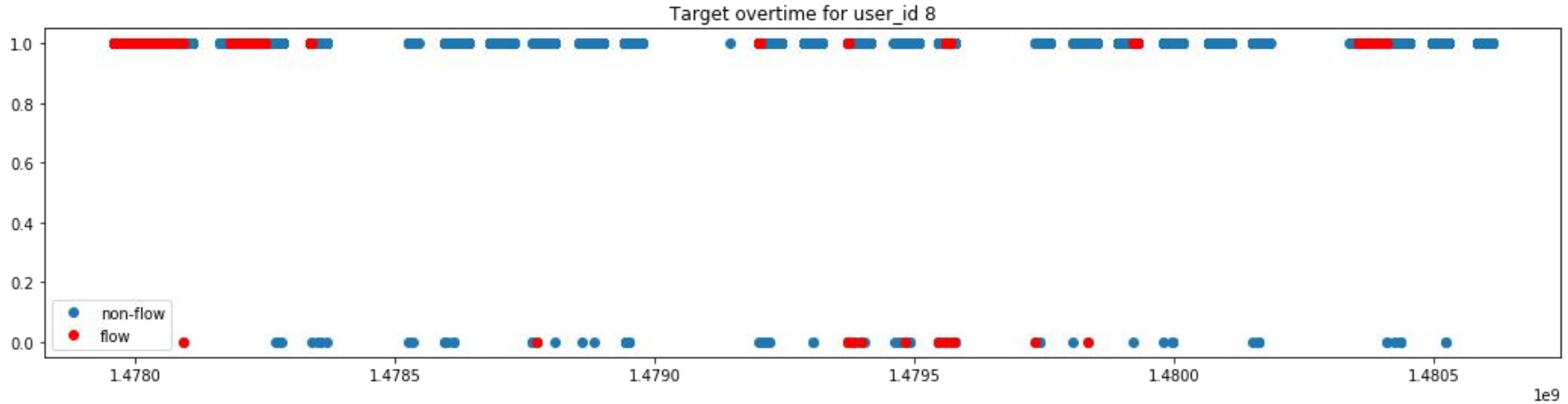
- Public: 0.6647
- Private: 0.6718

Feature Importance in XGBoost Model

```
('media_rank', 108554)
('media_bpm', 105514)
('diff_user_song_age', 100350)
('artist_fans', 99141)
('artist_albuns', 95015)
('album_id', 80906)
('diff_ts_listen_AND_release_date_Y', 77166)
('media_id', 73054)
('nmidia_regular_listened_PROP', 71187)
('nmidia_regular_listened', 69962)
('nmidia_PROP_diff', 64373)
('nmidia_with_flow_listened', 62142)
('nmidia_regular', 59499)
('nmidia_with_flow', 57739)
('user_id', 56269)
('nmidia_with_flow_listened_PROP', 55801)
('genre_id', 36985)
('context_type', 29875)
('ts_listen_PERIOD_OF_DAY_1', 26963)
('media_duration', 23570)
('release_date_YEAR', 18939)
('user_gender_1', 15002)
('ts_listen_PERIOD_OF_DAY_2', 10208)
('ts_listen_DAY_OF_WEEK_1', 9843)
('ts_listen_DAY_OF_WEEK_3', 9767)
('ts_listen_DAY_OF_WEEK_4', 9611)
('ts_listen_DAY_OF_WEEK_5', 9029)
('user_age_group_1', 8835)
('platform_name_1', 7579)
```

What we could have done better?

- Explore the time series nature in everything we have done:
  - Features in time (all the user specific for the last month, day, "listening chunk"...)
  - Create a more robust out-of-time validation schema;



14th place solution

Source: Team E3 Analytics - Peru

- Likelihood Features strong dependent on the time the song was listened;
- Users with less than 20 registers have their likelihood features set to NaN;
- Train only with registers in the flow mode;
- Use non-flow data to build non-flow behavior features for the users;
- Score
  - Public: 0.68036 (5th)
  - Private: 0.67310 (14th)

14th place solution

- Different features engineered:
  - Last_is_listened
  - Song / Album name features:
    - "Remastered" ,"Tribute to", "Version", "Edition", "Deluxe", "Special", "Remix", "Live";
  - Listening daytime;
  - Song duration comparison with the last one;

They use 100+ features;

Single model performance in local validation: 0.7275

The challenge

- Demand for Valeo products:
  - Information about past demand, product, price, competidor and so on;
- **Goal**: Predict the demand for a certain Material from a certain Organization
- Metric: MAE (Mean Absolute Error);
- The submission set consist of 38676 Material-Organization series for 3 periods: 2017-04, 2017-05 and 2017-06

The data

Variable	Description	Definition
ID	Row identifier	Row Identifier
ordre	Command number	Unique ID # of order from a customer (several products can be ordered within the same order)
First_MAD	Demand date	The date when the customer ordered the products
SalOrg	Sales Organization code	ID of the Valeo distribution center that sold the products (different geographical location = Country)
DC	Distribution channel code	Represents different channels of sales
Ship_To	Ship to customer code	ID of the customer entity that received the products (customer warehouse)
Plant	Warehouse	ID of a Valeo distribution warehouse shipping the products (different geographical location)
Material	Product code	Product reference #
ItemCat	Item Category	Type of the demand for a particular order item (represents level of variation from average demand for a particular product)
OrderQty	Quantity ordered	Ordered quantity

The data

LT	Lead Time	Lead time. A procurement parameter representing number of days needed to produce the product and deliver it to Valeo distribution warehouse
LogABC	Materials' logistic classification	Represents different categories of product (high-/medium-/low- movers) from procurement standpoint
MOQ	Minimum of Quantity	A procurement parameter. Represents minimum size of the batch (in pieces) in which product is produced and delivered to distribution warehouse
ROP	ROP	A procurement parameter. Represents the level of inventory which triggers an action to replenish that particular product in a distribution warehouse
SafetyStk	Safety stock	A procurement parameter. Represents the level of extra (buffer) stock that is maintained to mitigate risk of stockouts in a distribution warehouse
PL	Material's product line/line of business	Product line of business (e.g. lighting products / climat control products)
MktABC	Marketing classification	Represents different categories of product (fast-/medium-/slow- movers) from sales standpoint
SubFct	Personnal car or Truck	Differentiates between the types of the vehicle this product is for (Passenger car, Truck, Agricultural, etc…)

The data

Gross_Weight	Material gross weight	Product gross weight
Length	Material length	Product length
Width	Material width	Product width
Height	Material height	Product height
Volume	Material volume	Product volume
Gamma	Materials' type of application	Represents different geographical groups of car brands (e.g. European cars, Asian cars, American cars,…)
Manufacturer	Manufactuer	Car brand
Business	Business name	Differentiates the products that is just replacing the old one (like a new headlamp, or a new cabin air filter) from the other types (e.g. maintenance kits)
Month	Month name	The month when the customer ordered the products (calculated from date)
CBO_CBO_Qty_Shortage	Customer Back Order (CBO). Material's last Quantity	The product quantity in shortage (in pieces) at the end of month (see field 'Month') in the particular distribution warehouse

The data

Age_ZN_ZI_years	Age of a product in the market	Age of a product in the market
DP_FAMILY_CODE	Subsegment of material's product line	Family of product (sub-line of business). E.g. head lamps / cabin air filters
PRODUCT_STATUS	The current status of the product (Material)	Status of the product within product life cycle
ORIGINAL_SUPPLIER	The original supplier name of the Material	Code of the factory producing the product
SUBRANGE	Sub range of the Material	Sub-family of product (specific ranges within same family)
Comp_reference_number	Number of competitors references	The number of similar Materials on the market belonging to the competitors
Name_Of_Competitor	Number of the competitors for the current Material	Number of the competitors for the current Product
COMP_PRICE_MIN	Competitors minimum price	Competitors minimum price for the current Product
COMP_PRICE_AVG	Competitors average price	Competitors average price for the current Product
COMP_PRICE_MAX	Competitors maximum price	Competitors maximum price for the current Product
PRICE	Valeo price	Product price (Valeo)
NEAREST_COMP_PRICE_MIN	Competitors minimum nearest price	Competitors minimum nearest price. Take the lower one if equality.
NEAREST_COMP_PRICE_MAX	Competitors maximum nearest price	Competitors maximum nearest price. Take the higher one if equality.

The series

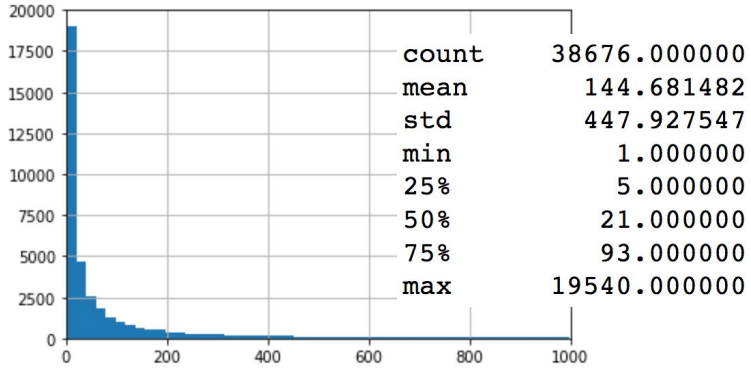
Like the qualifiers: very different data in the same dataset;

Series differ by:

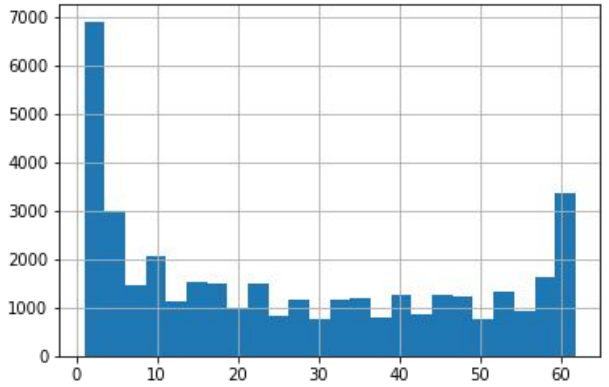
- Quantity ordered;
- Length;
- Distance to the prediction periods;

Good series: long and close to the prediction period;

Count distribution for series



Distance in months to 2017-04



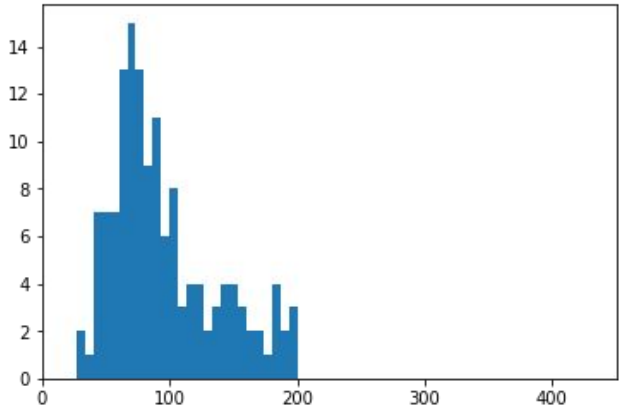
Overall Strategy

- Use different models to the good and bad series;
- Use a simple model to bad / corner cases;
- Validate out-of-time: 2017-01, 2017-02 and 2017-03;
- Create as many features as possible (lags and its statistics, max, min..)
- Try to use different models to get uncorrelated predictors;
- Blend different models results;
- Approaches:
  - (Benchmark): Last 3 months averaged;
  - Lasso for good series;
  - XGBoost them all!
  - Entity Embedding;
  - Not included in final submission: Prophet and LSTM;

Lasso

- Simple model;
- One model for each Material-Organization;
- Intended to be used with the good series;
- OrderQty in past 8 periods to predict the next;

MAPE distribution for 178 series



XGBoost

- With all series, using different sets of features; material and organization ids;
- Performance (local):
  - Using only 5 lags: 10.23
  - Using all features: 9.62
- Score:
  - 12.63 (public), 11.82 (private)

FI for simple model

```
('OrderQty_1', 8198)
('OrderQty_2', 7302)
('OrderQty_3', 7221)
('OrderQty_5', 6748)
('OrderQty_4', 6675)
('Material', 5731)
('SalOrg', 1170)
```

FI for complex model

```
('month', 3568)
('Last5_Mean', 2631)
('SafetyStk', 2361)
('OrderQty_1', 2164)
('OrderQty_5', 2092)
('Last3_Mean', 1988)
('OrderQty_4', 1871)
('OrderQty_2', 1596)
('Last5_Qty_Max', 1587)
('OrderQty_3', 1506)
('diff2', 1473)
('Last5_Qty_std', 1435)
('Material', 1372)
('Last3_Qty_std', 1362)
('diff1', 1233)
('Last5_Price_std', 1191)
('SalOrg', 1135)
```

Entity Embedding

- We know the code snippet is small, check it online later;
- From Rossmann Sales Prediction Kaggle Competition;
- Sources: Repository and Paper;
- Another way to use all the data in a single model;
- Performance
  - Local 9.34
  - Public: 11.68
  - Private: 10.96

```python
def create_model(features, max_org, max_prods):
    # Aqui eu vou receber o ID das Orgs, é um inteiro
    orgs_input = Input(shape=(1, ), dtype='int32', name='orgs_id_input')

    # Aqui eu vou receber o ID dos produtos, é um inteiro
    prods_input = Input(shape=(1, ), dtype='int32', name='prods_id_input')

    # Essa layer fará o encoding da entrada
    # Vai transformar um inteiro em um vetor de 2 dimensões
    # Preciso informar o ID máximo de uma org
    orgs_out = Embedding(output_dim=2, input_dim=max_org, name="orgs_embedding_layer")(orgs_input)
    orgs_out = Flatten()(orgs_out)

    # Essa layer fará o encoding da entrada
    # Vai transformar um inteiro em um vetor de 300 dimensões
    # Preciso informar o ID máximo de um produto
    prods_out = Embedding(output_dim=300, input_dim=max_prods, name="prods_embedding_layer")(prods_input)
    prods_out = Flatten()(prods_out)

    # Aqui entram as outras features que serão usadas na previsão
    auxiliary_input = Input(shape=(len(features), ), name='aux_input')
    x = keras.layers.concatenate([orgs_out, prods_out, auxiliary_input], name="concat_layer")

    # E aqui vem uma fully-connected no topo
    x = Dense(16, activation='relu', name="dense_1")(x)
    x = Dense(8, activation='relu', name="dense_2")(x)
    x = Dense(4, activation='relu', name="dense_3")(x)

    # E a saída, que faz a regressão de fato.
    main_output = Dense(1, activation='linear', name='output')(x)

    model = Model(inputs=[orgs_input, prods_input, auxiliary_input], outputs=[main_output])

    model.compile(loss='mean_absolute_error', optimizer='adam')

    return model
```

2nd place solution

Average of 3 models from two approaches:

- The median of last n periods;
- The ratio method;

Source: LSTeAm solution

- Score
  - Public: 9.45 (4th)
  - Private: 10.45 (2nd)

- Models performance (MAE):
  - Random Forest 1: 11.25
  - Random Forest2: 10.25
  - Last 15 months median: 9.89
  - Last 15 + ratio: 9.84

(Local validation, all for 2017-March)

2nd place solution

- Median:

"If a couple SalOrg - Material has less than i months of history (we consider that the history of a couple SalOrg-Meterial begins when the demand takes a non zero value for the first time), we take the median over the available history."

2nd place solution

- Ratio I (not for Material-Org, but by Product Line)

$$ratio\_medians = \frac{nov_{2016} + nov_{2015} + nov_{2014} + nov_{2013}}{median(nov_{2015} : oct_{2016}) + median(nov_{2014} : oct_{2015}) + median(nov_{2013} : oct_{2014}) + median(nov_{2012} : oct_{2013})}$$

- Ratio II: "computed based on PL + [low, mid, high] where low/mid/high indicates the order of magnitude of the median : low [0;1[, mid [1,10[, and high [10+]"

Basically, see the past errors of using median / mean as a prediction and apply a factor to the next prediction;

Take aways

- After hitting a performance plateau the iteration must include the Exploratory Data Analysis;
- Time always plays a role in the data, check if the dataset makes it possible to use it;
- It worths to have someone only engineering new features (and checking if they make sense);
- Try to segment the data and find corner cases, use a different strategy to them, measure how good each approach is in every segment you have identified;
- Be russian;

Questions?

Luís Moneda
E-mail: lgmoneda@gmail.com
Kaggle: @lgmoneda