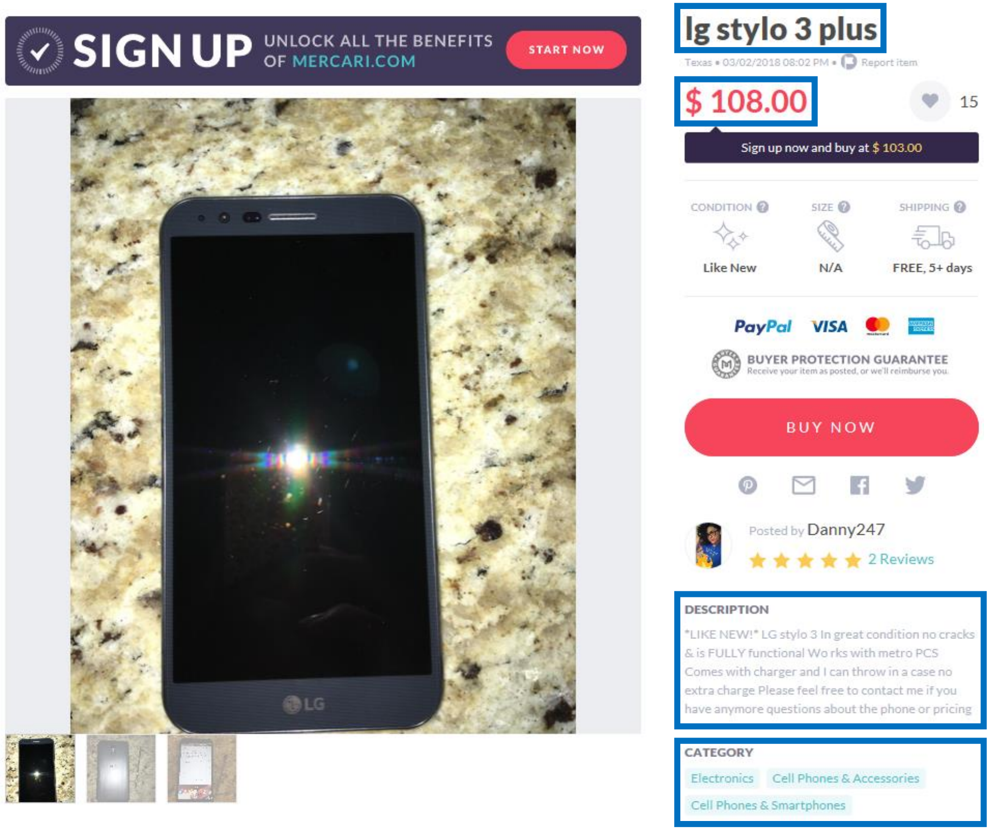
MERCARI PRICE SUGGESTION CHALLENGE

46TH PLACE SOLUTION – BY THOMAS SELECK

COMPETITION CONTEXT AND GOAL

- Mercari: Japan’s biggest community-powered shopping app
- It can be hard to know how much something’s really worth: small details can mean big differences in pricing.
- Goal of the competition: Build an algorithm that automatically suggests the right product prices using user-inputted text descriptions of their products, product category name, brand name, and item condition.



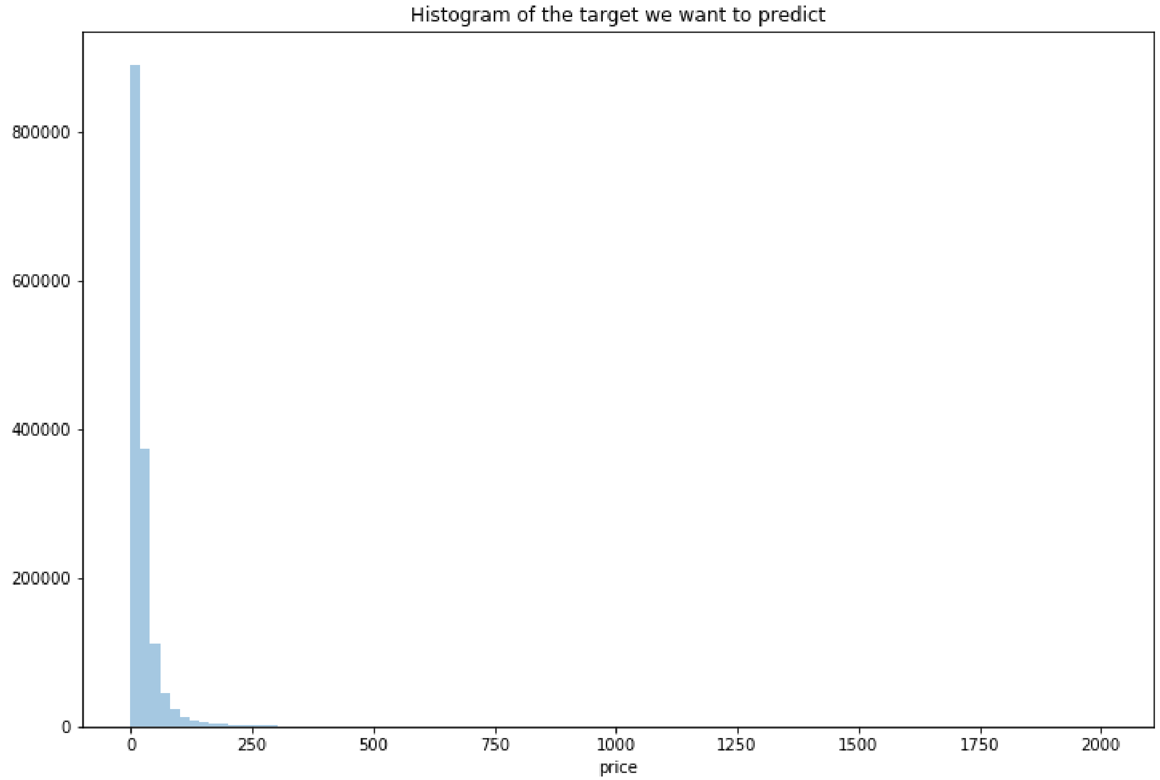
EVALUATION METRIC AND DATA

- Evaluation metric: RMSLE (Root Mean Squared Logarithmic Error)

$$RMSLE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\log(p_i + 1) - \log(a_i + 1))^2}$$

  - n is the total number of observations in the data
  - p_i is the predicted price
  - a_i is the actual sale price for the i^{th} article

- Data:
  - 1,482,535 samples in train set
  - 693,359 samples in test set (phase 1)
  - 3,5 million samples in test set (phase 2)
  - 6 features: *name*, *item_condition_id*, *category_name*, *brand_name*, *shipping*, *item_description*



FEATURE ENGINEERING (1 / 2)

- 874 items had a zero price in train set ⇒ remove those items
- *brand_name* feature: 42.67% of items in train set don't have a brand ⇒ Create dummy: 1 == no brand
- For *item_condition_id*: goes from 1 to 5; 1 = brand new and 5 = broken; reverse levels order (1 = 5, 2 = 4, …)
- Compute average price for each category found in *category_name*.
  - Group categories than have less than 10 samples to avoid overfitting.
  - Create 7 new features based on it using binning: avg $\in$ [0;10[, [10;20[, [20;40[, [40;50[, [50;75[, [75;+∞[
- Extract first 3 levels out of 5 from *category_name* and fill missing values with “missing” string
  - Last 2 levels are only used by 0.3% of samples
  - E.g. Women/Jewelry/Necklaces ⇒ *category_1* = Women, *category_2* = Jewelry, *category_3* = Necklaces
- Fill missing values in *brand_name*, *name* and *item_description* with “missing” string
- Group least occuring brands, as 1,243 brands out of 4,810 only appears once in train set

FEATURE ENGINEERING (2 / 2)

- Luxury brands are expensive ⇒ Dummy equal to 1 for 22 luxury brands *(Louis Vuitton, Rolex, Apple, …)*
  - Do the same for cheapest brands
- Look for important keywords
  - “dust” ⇒ refers to “dust bag”, a women’s handbag accessory: brand new luxury bags have one, others not
  - “gold” ⇒ as gold is a precious metal, having gold in the item raises its price
  - “lularoe” ⇒ clothes brand that was one of the most important features for LightGBM
  - “bundle” ⇒ this means several objects to sell and can increase the price
- Compute some statistics on *name* and *item_description*
  - Number of characters, tokens, words, numbers, letters, digits
- Add brand groups depending on *category_1*
- Normalize the created features (substract mean and divide by standard deviation)
- Use WordBatch on *name* and *item_description* (text processing), LabelBinarizer for *brand_name*, CountVectorizer for *category_1*, *category_2* and *category_3*

PREDICTIVE MODELS

- FTRL (Follow The Regularized Leader): kind of adaptive-learning-rate sparse linear regression with efficient L1-L2-regularization
  - Comes from WordBatch package: https://github.com/anttttti/Wordbatch
  - Original paper: H. B. McMahan, "Follow-the-Regularized-Leader and Mirror Descent: Equivalence Theorems and L1 Regularization": https://static.googleusercontent.com/media/research.google.com/fr//pubs/archive/37013.pdf

- FM_FTRL: Factorization Machine with linear effects estimated with FTRL and factor effects estimated with adaptive SGD.
  - Comes from WordBatch package: https://github.com/anttttti/Wordbatch

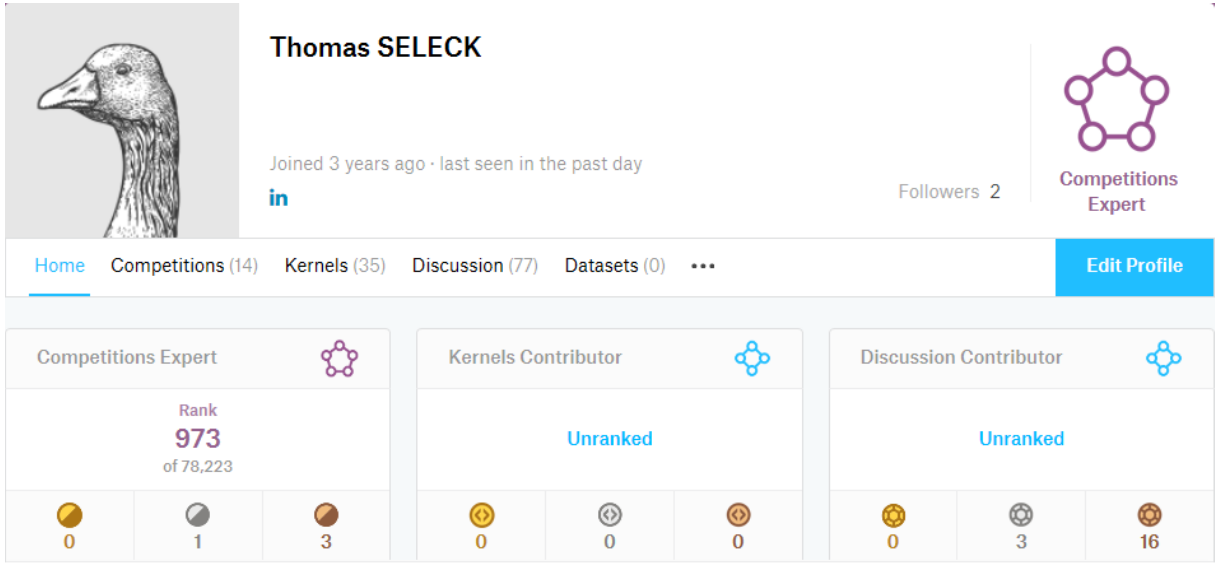
- LightGBM: Gradient boosted trees library developped by Microsoft
  - More info here: https://github.com/Microsoft/LightGBM

BLENDING STRATEGY

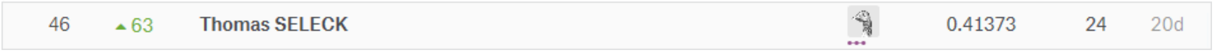
- Quick and Dirty way:
  - Split train set into 2 sets: *X_train* and *X_valid* (*y_train* and *y_valid* for target)
  - Group predictions in a DataFrame and split it by *category_1*
  - Fit a linear regression without intercept for each split, trying to predict price using predictions made by all three models.
  - Use linear regression coefficients as weights for blending
  - E.g. “Men”: [-0.01, 0.65, 0.36] ⇒ preds for “Men” split = -0.01 × FTRL + 0.65 × FM_FTRL + 0.36 × LightGBM

FINAL RESULTS AND CONCLUSION

- Public LB score: 0.41316
- Private LB score: 0.41373



- Code and slides available here:
  - https://github.com/ThomasSELECK/Kaggle_Mercari_competition



ANY QUESTIONS?

DON’T FORGET: HTTPS://GITHUB.COM/THOMASSELECK/KAGGLE_MERCARI_COMPETITION