Uni_PNRP_Perm_2 Solution: Ensembling and Precise data analysis

Aleksandr Perevalov[1] Sergei Bobkov[1]

[1]Perm National Research Polytechnic University, Russia

Data Mining Cup @ Prudsys retail intellegence summit, 2019

General info about the data

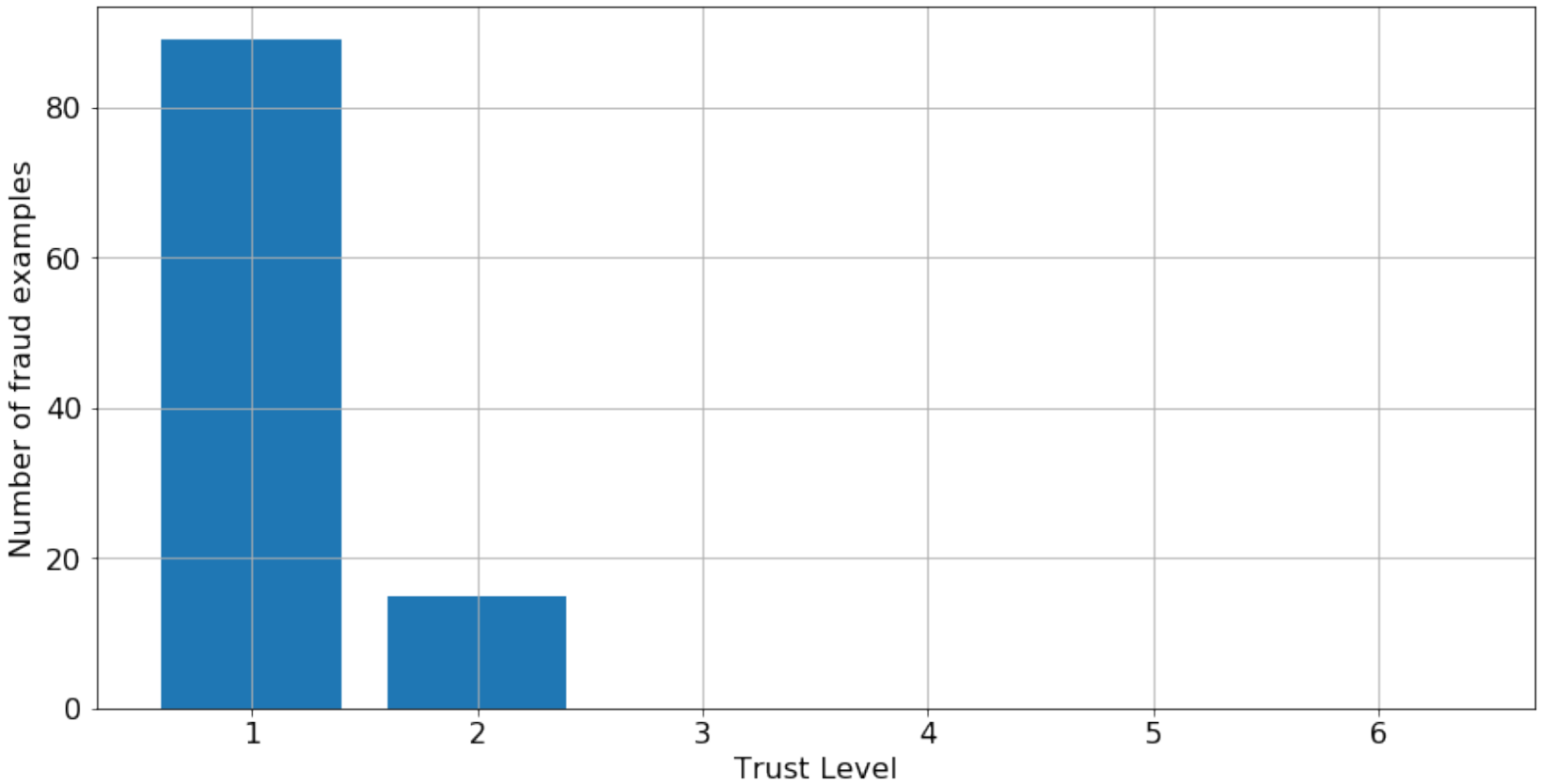
Some simple facts:

- The dataset has no NANs;
- Fraud/Not fraud ratio = 6% to 94%;
- The dataset has 9 features and 1 target;
- The dataset has only 1 categorical feature – **trustLevel**.

Some thoughts after reviewing these facts:

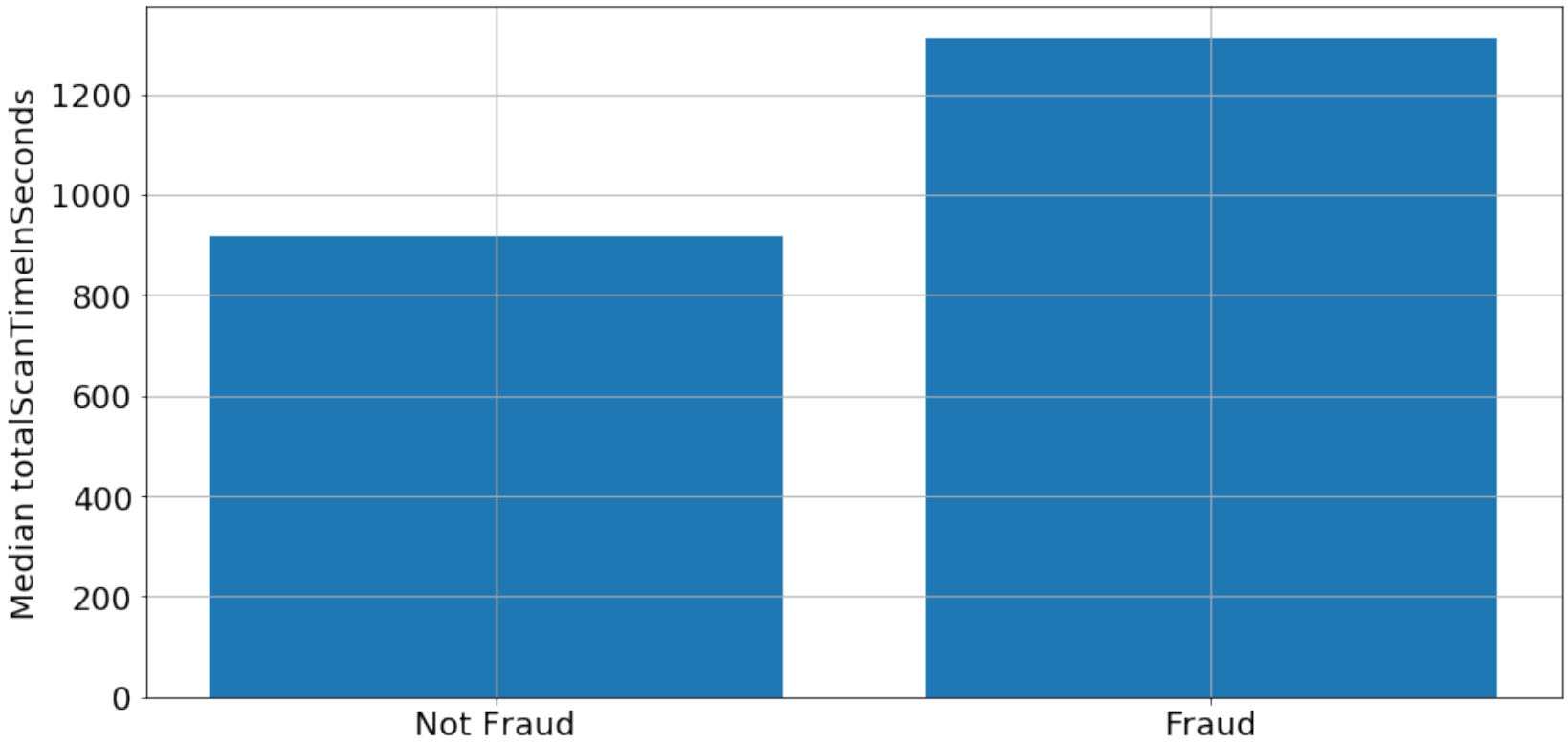
- The data is artificially generated (?);
- Don't use Accuracy (use Precision and Recall instead as additional metric for validation);
- Probably, **trustLevel** will have big weight.

Insights from data



Maybe just train on examples with trustLevel (TL) = 1,2 and constantly mark examples with TL more than 2 as not fraud?

Insights from data



Fraudulent self-checkout process takes more time than non-fraudulent one
Mean overall total scan time is **15 minutes** – why so long?

Train and Test feature descriptive statistics

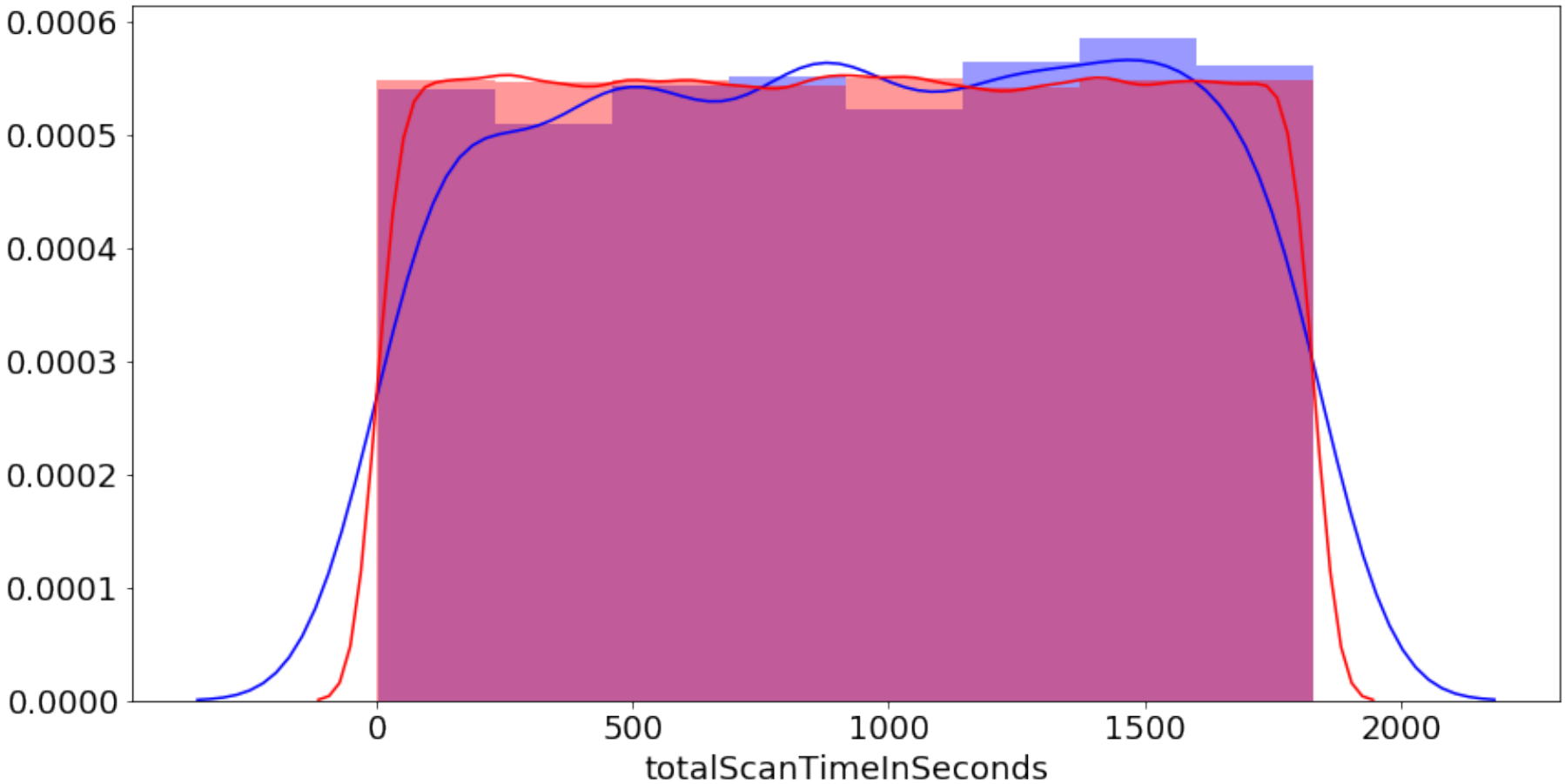
The statistics (min, max, mean, std. etc) of the Train data and Test data are the same except: **scannedLineItemsPerSecond** and **valuePerSecond**;

	scannedLineItemsPerSecond	valuePerSecond
count	1879.000000	1879.000000
mean	0.058138	0.201746
std	0.278512	1.242135
min	0.000548	0.000007
25%	0.008384	0.027787
50%	0.016317	0.054498
75%	0.032594	0.107313
max	6.666667	37.870000

	scannedLineItemsPerSecond	valuePerSecond
count	498121.000000	498121.000000
mean	0.068054	0.222182
std	0.521092	1.717867
min	0.000546	0.000000
25%	0.008682	0.027348
50%	0.016940	0.054550
75%	0.033929	0.109091
max	30.000000	99.710000

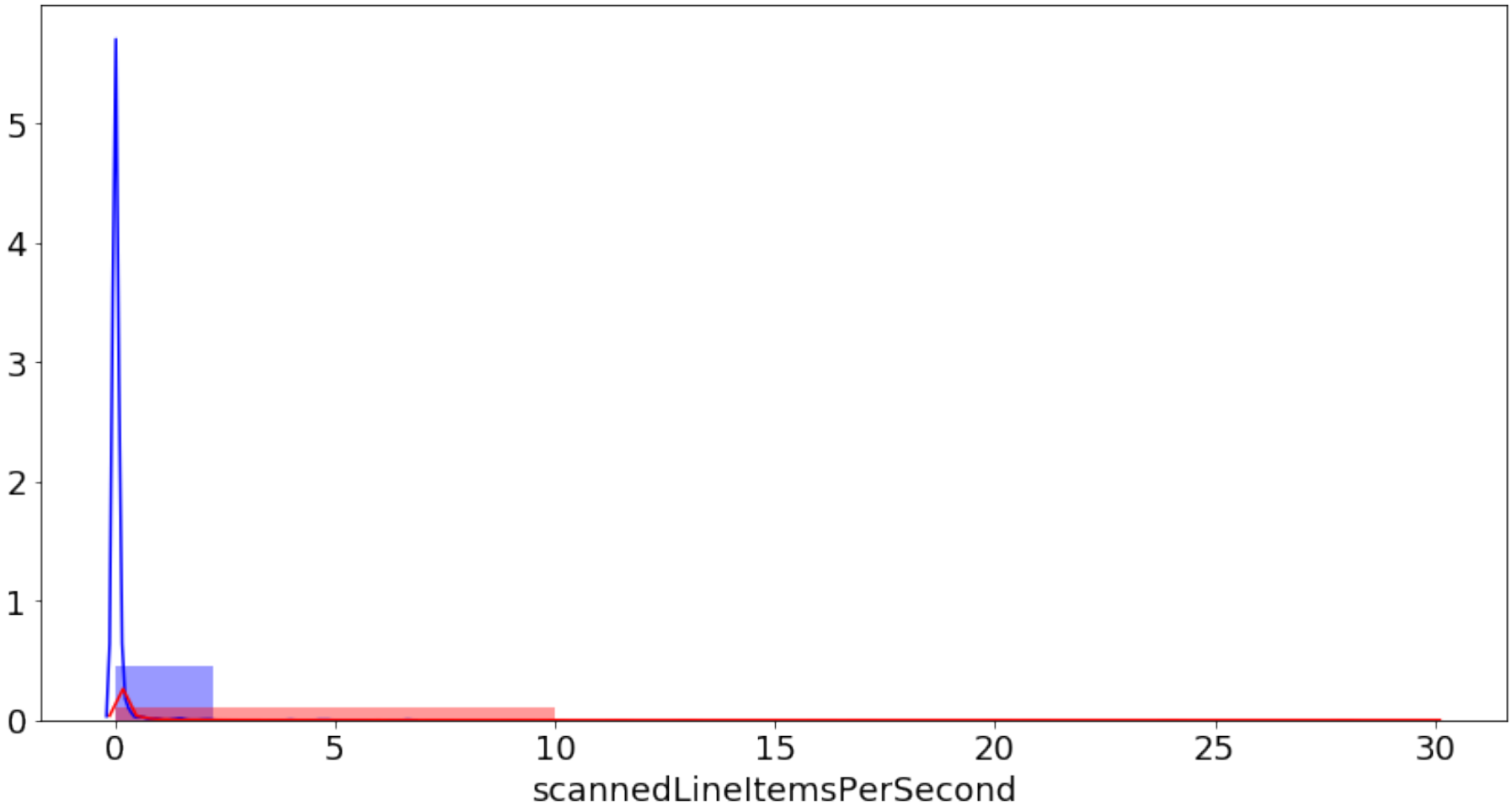
Train and Test feature distributions

"Good" train and test distributions nearly match each other



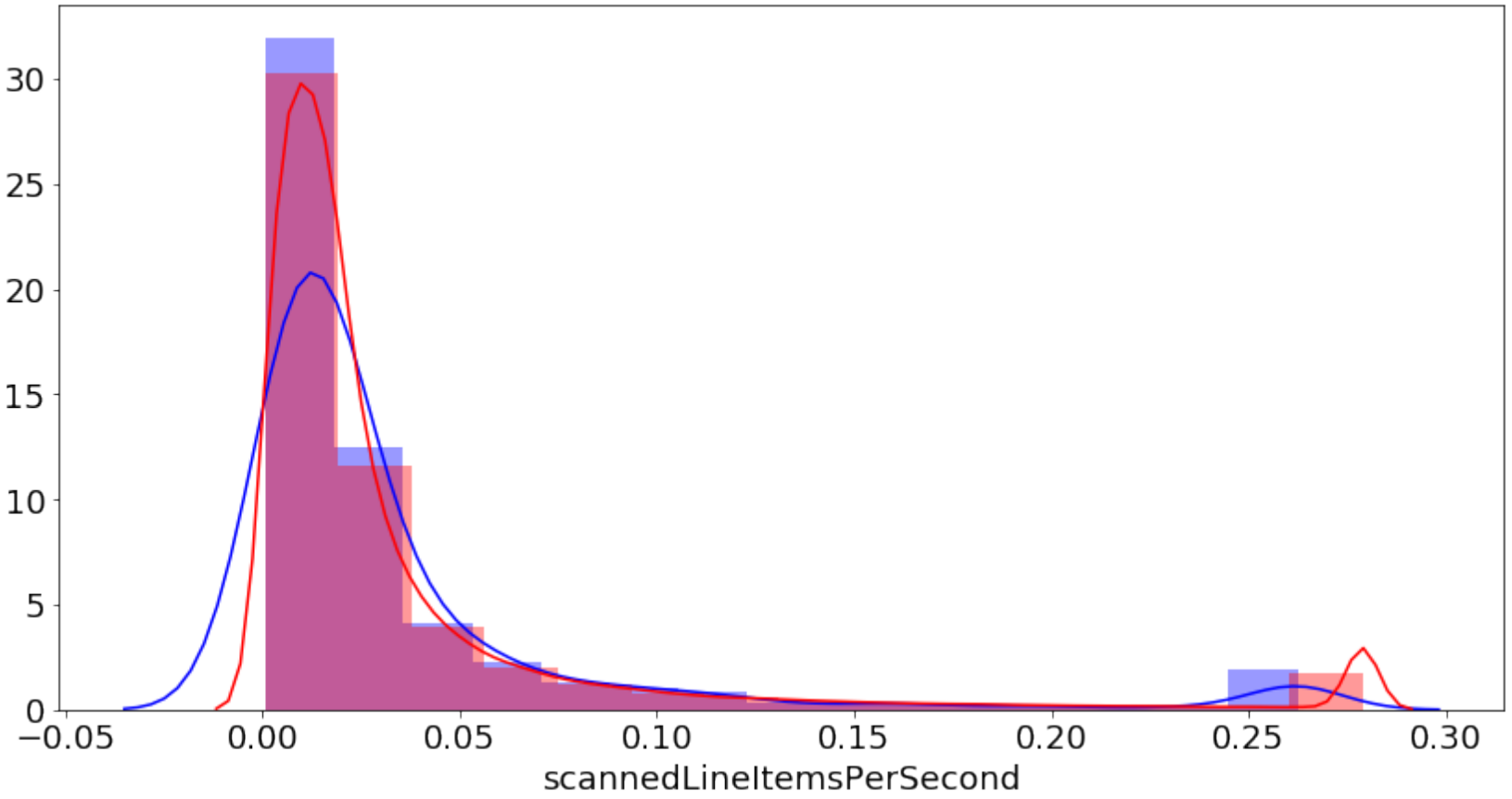
Train and Test feature distributions

This is not surprise that only **scannedLineItemsPerSecond** and **valuePerSecond** distributions doesn't match each other



Fix distributions by removing outliers

We fixed "bad" distributions by clipping their values by 1-percentile at lower bound and 97-percentile at upper bound



This procedure made our local validation score worse, but we believe that it made it better on the whole data

Feature engineering

14 features were generated manually, only 6 of them passed the validation test.
Some of the features:

- $avgTimePerScan * valuePerSecond$
- $scannedLineItemsPerSecond * totalScanTimeInSeconds$
- $\frac{1}{scannedLineItemsPerSecond}$

PCA, Truncated SVD, LDA. Number of components: 2
We also tried polynomial features such as $lineItemVoids^2$ and so on, but they showed worse result on final model.

Metric and Validation

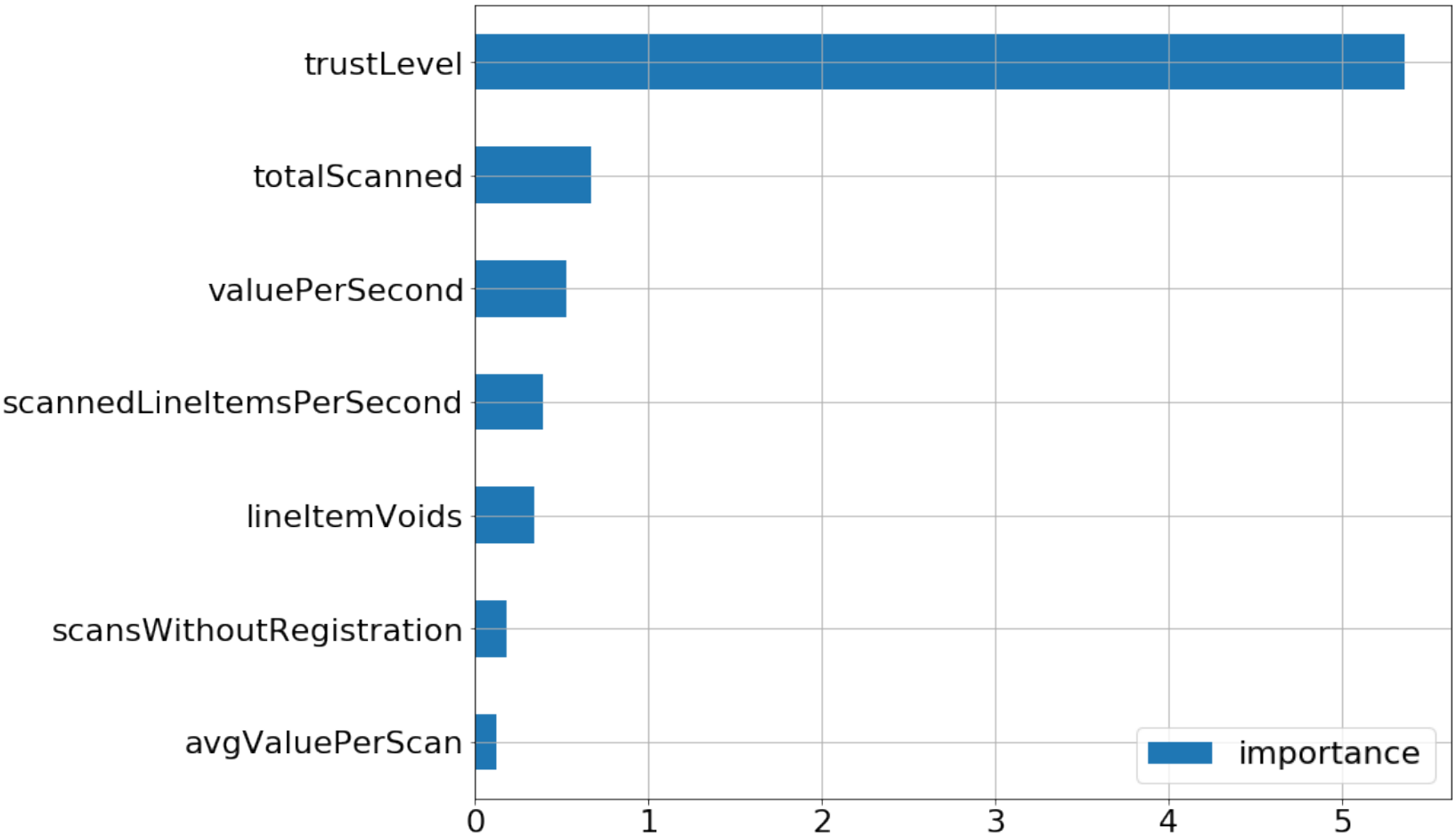
Metrics: F-Score, Precision, Recall, Organizer's metric
Validation: K-Fold stratified cross-validation (3, 5, 7, 10 folds)

	Actual values	
	Fraud	**Not fraud**
Fraud	5 Euro (TP)	-25 Euro (FP)
Not fraud	-5 Euro (FN)	0 Euro (TN)

Maybe use threshold to make our predictions more confident?

Baseline model – Logistic regression

Score = 285 Euros (520 max)

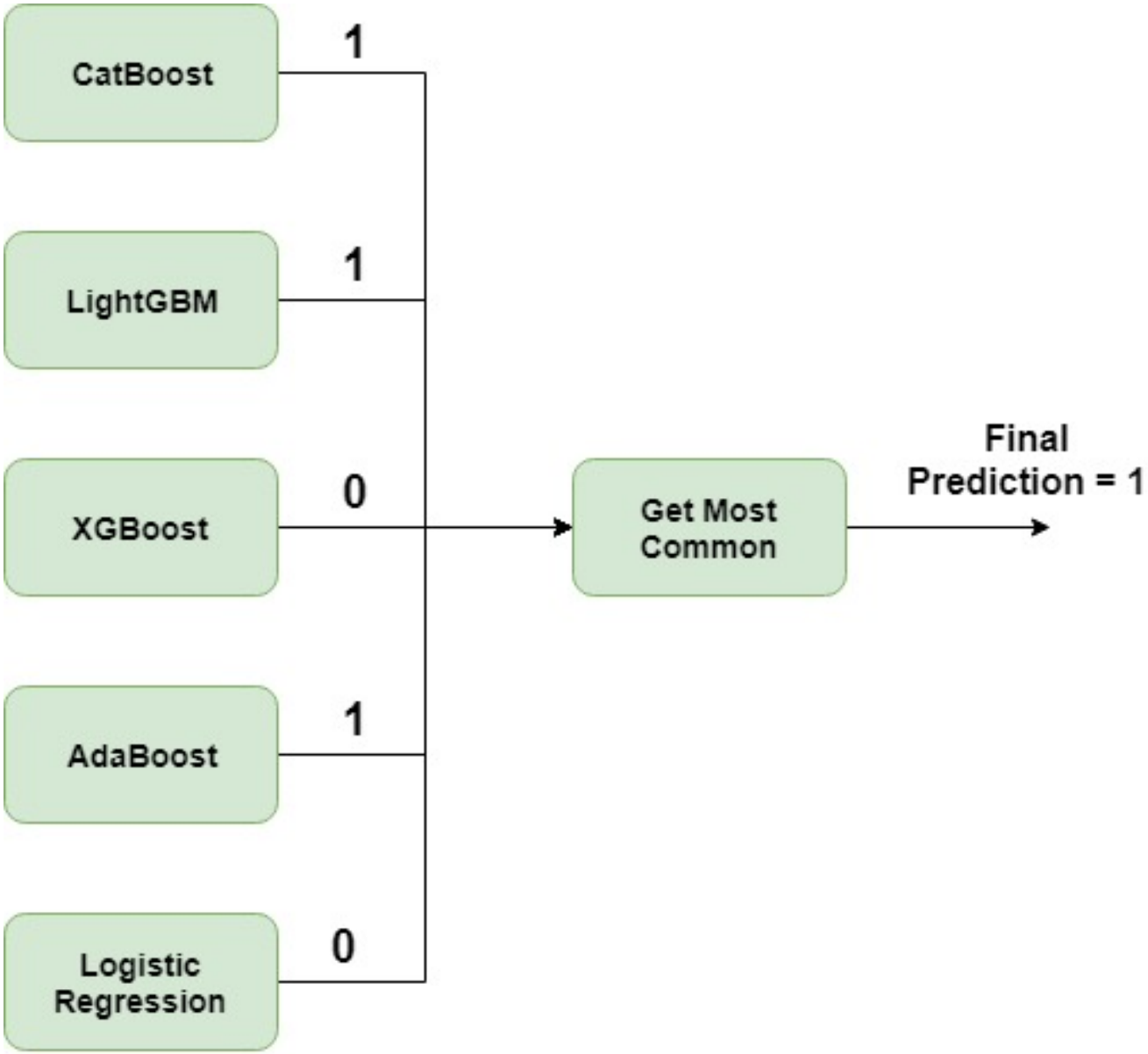


Final model building

1. Searching for 5 best models (Gradient boosting, Extra trees, kNN, SVM, NNs...);
2. Optimizing hyperparameters tuning algorithm;
3. Neural networks didn't work well;
4. Hardware: i5, 3.2 GHz, GTX 1060;
5. Training time: 30 minutes.

Final model architecture

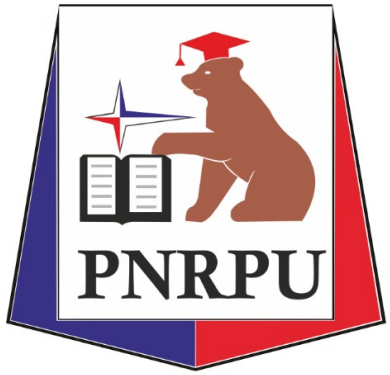
Score = 355 Euros (520 max)



Questions

Possible ways of solution improvement:

- Oversampling and undersampling;
- One model for one customer level;
- Stacking techniques and meta-features;
- Threshold for classifier (> 0.5 confidence for fraud);
- Polynomial features;
- Consultations with people from business;
- More models in ensemble;
- Add weights to ensemble models;
- ...



Bernburg
Dessau
Köthen

Hochschule Anhalt
Anhalt University of Applied Sciences

Thanks for attention!

Questions?

perevalovA@pstu.ru