Mercedes-Benz Competition: 11th place

Daniel Savenkov

#danila_savenkov

https://www.kaggle.com/daniel89

https://www.linkedin.com/in/daniel-savenkov-098b41149/

- https://www.kaggle.com/c/mercedes-benz-greener-manufacturing/discussion/36242#202443
- https://github.com/Danila89/kaggle_mercedes

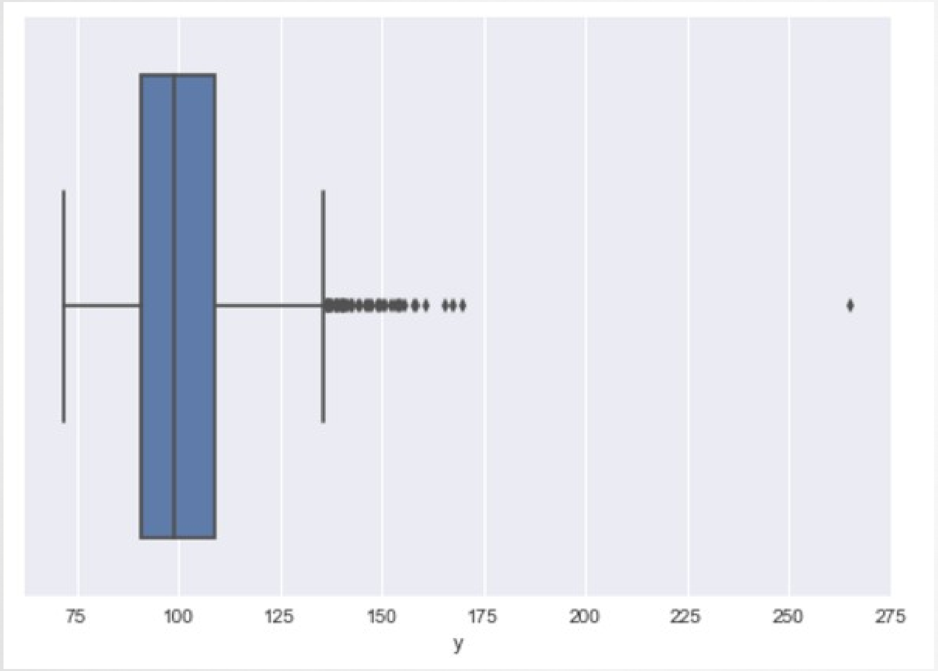
Schedule

- The problem: data, LB-probing, shake-up
- Public Kernel “Stacked and Averaged”
- Gradient boosting and categorical features
- Cross-validation and holdout set importance
- My model
- The 2nd place approach

Problem and data

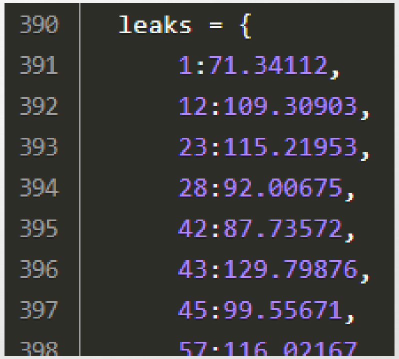
- The target – car testing time (sec)
- Evaluation metric - R2
- Train 4029 rows
- Test 4029 rows: 81% private, 19% public
- Features (378 columns):
  - Binary – tests' characteristics (369 columns)
  - Categorical – car's characteristics (8 columns)
  - ID – numerical order

Problem: public leaderboard



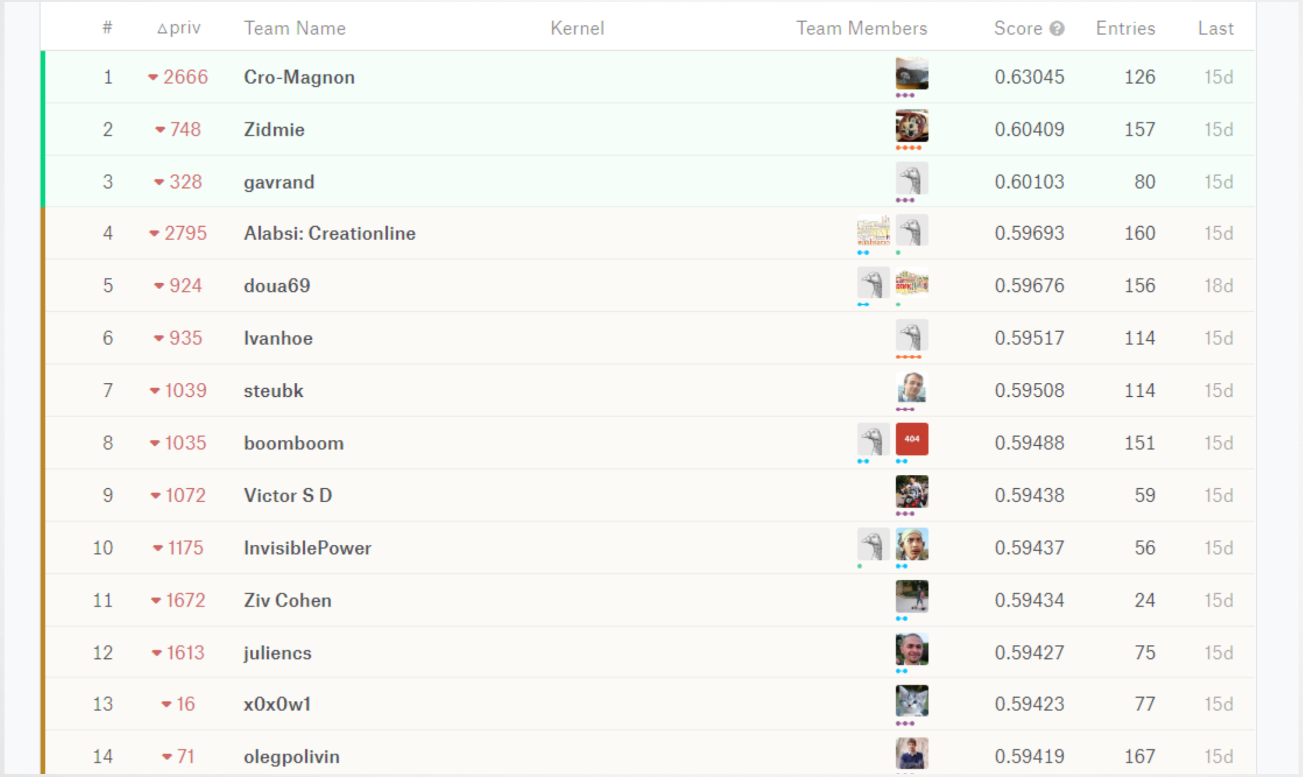
- Train 4029 rows
- Test 4029 rows:

  81% private, 19% public
- 5-fold cross-validation std 0.068



```
leaks = {
    1:71.34112,
    12:109.30903,
    23:115.21953,
    28:92.00675,
    42:87.73572,
    43:129.79876,
    45:99.55671,
    57:116.02167
```

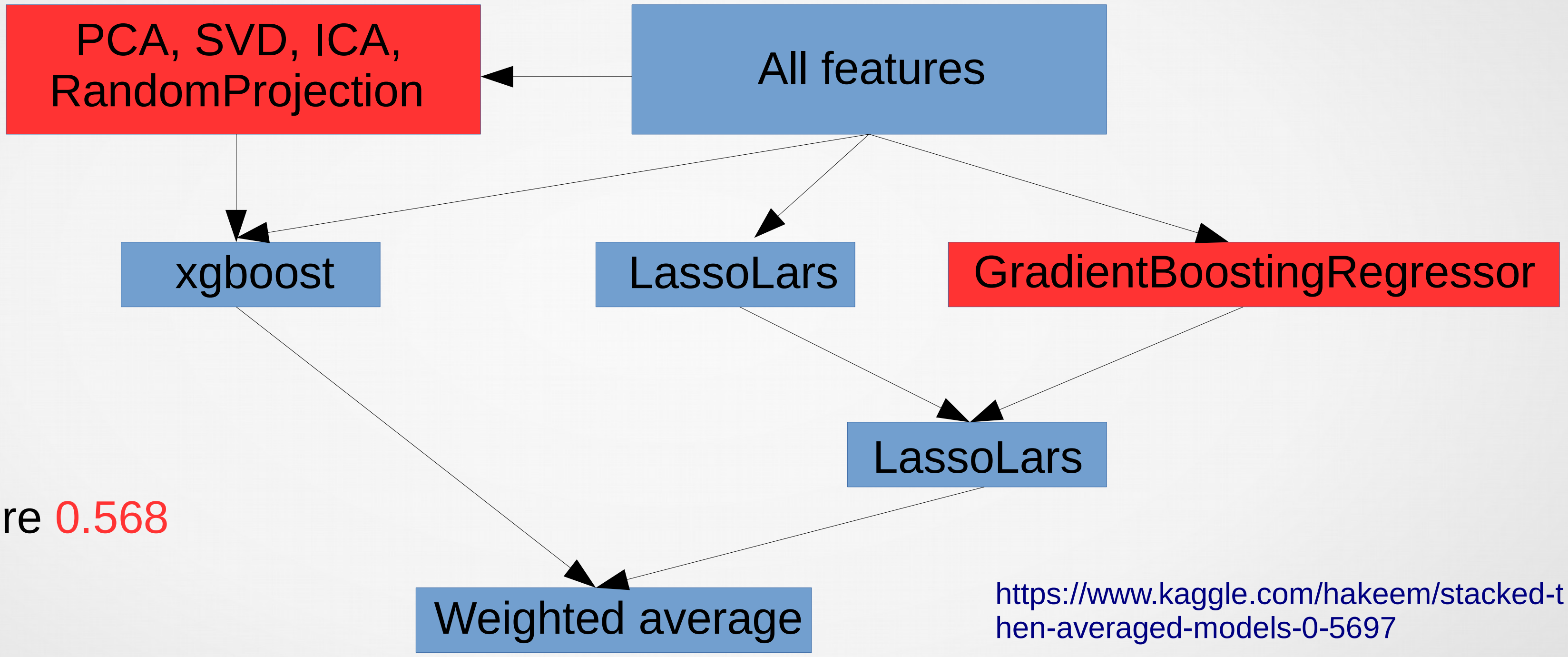
Problem: public leaderboard



#	∆priv	Team Name	Kernel	Team Members	Score	Entries	Last
1	▾ 2666	Cro-Magnon			0.63045	126	15d
2	▾ 748	Zidmie			0.60409	157	15d
3	▾ 328	gavrand			0.60103	80	15d
4	▾ 2795	Alabsi: Creationline			0.59693	160	15d
5	▾ 924	doua69			0.59676	156	18d
6	▾ 935	Ivanhoe			0.59517	114	15d
7	▾ 1039	steubk			0.59508	114	15d
8	▾ 1035	boomboom			0.59488	151	15d
9	▾ 1072	Victor S D			0.59438	59	15d
10	▾ 1175	InvisiblePower			0.59437	56	15d
11	▾ 1672	Ziv Cohen			0.59434	24	15d
12	▾ 1613	juliencs			0.59427	75	15d
13	▾ 16	x0x0w1			0.59423	77	15d
14	▾ 71	olegpolivin			0.59419	167	15d

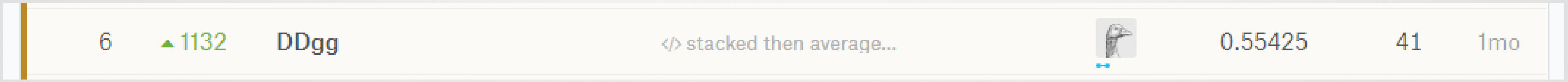
Winner's private score 0.55551

Public Kernel “Stacked and Averaged”



Public score 0.568

https://www.kaggle.com/hakeem/stacked-then-averaged-models-0-5697

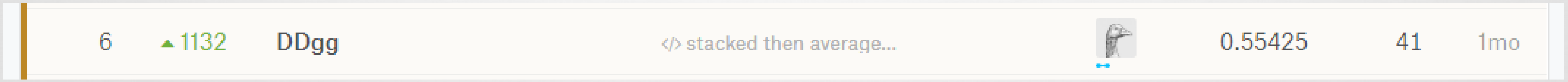


6	▲1132	DDgg	</> stacked then average...	0.55425	41	1mo

Public Kernel “Stacked and Averaged”: PCA, SVD...

PCA, SVD, ICA, RandomProjection

- The projections were performed without pre-scaling of features
- The scale of ID is much higher than the scale of the categorical features
- Binary features are almost not presented at the projections
- If we delete all projections from the script, we get statistically significant improvement on CV, improvement of private score, but public score would decrease in 0.01
- Public Leaderboard Overfitting

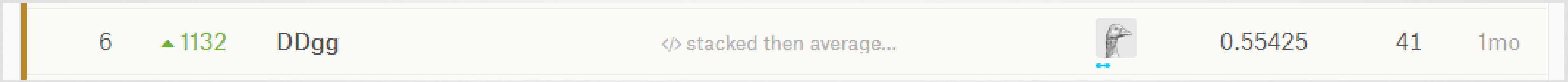


6 ▲1132 DDgg </> stacked then average... 0.55425 41 1mo

Public Kernel “Stacked and Averaged”: GradientBoostingRegressor

GradientBoostingRegressor

- GradientBoostingRegressor(learning_rate=0.001, loss="huber", max_depth=3, max_features=0.55, min_samples_leaf=18, min_samples_split=14, subsample=0.7)
- Random_state is not fixed
- The script has 503 forks, DDgg has the luckiest seed



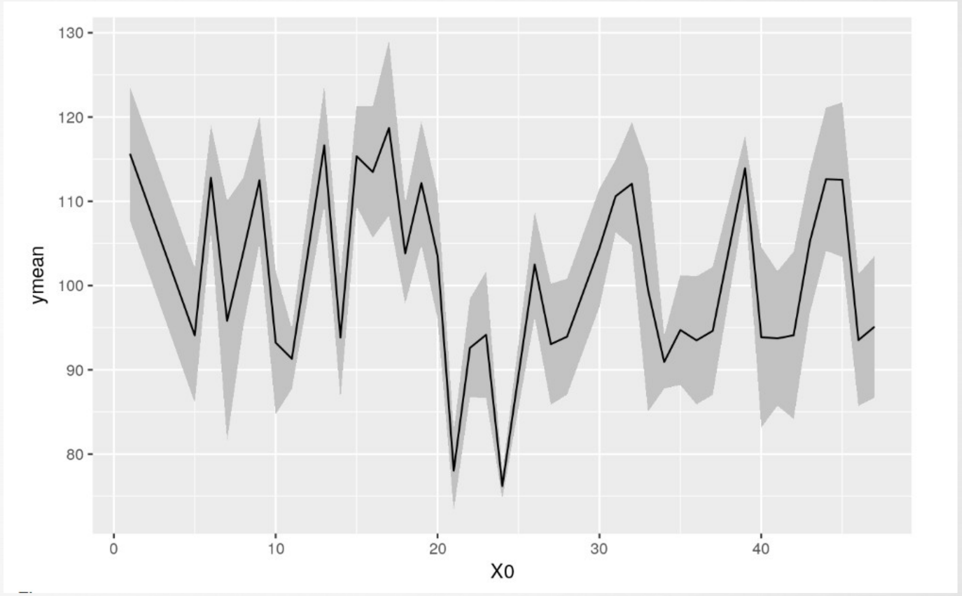
6	▲ 1132	DDgg	</> stacked then average...	0.55425	41	1mo

Public Kernel “Stacked and Averaged”: the best stable result

- The best script based on this kernel yields 0.554 at private leader-board, which is the 10th place: https://www.kaggle.com/adityakumarsinha/stacked-then-averaged-models-private-lb-0-554/code
- It has several crucial changes:
  - It does not use the projections
  - Every object has it's own features plus features of two neighboring IDs. It's more advanced way to use the fact that the ID's order is not random

Gradient boosting and categorical features

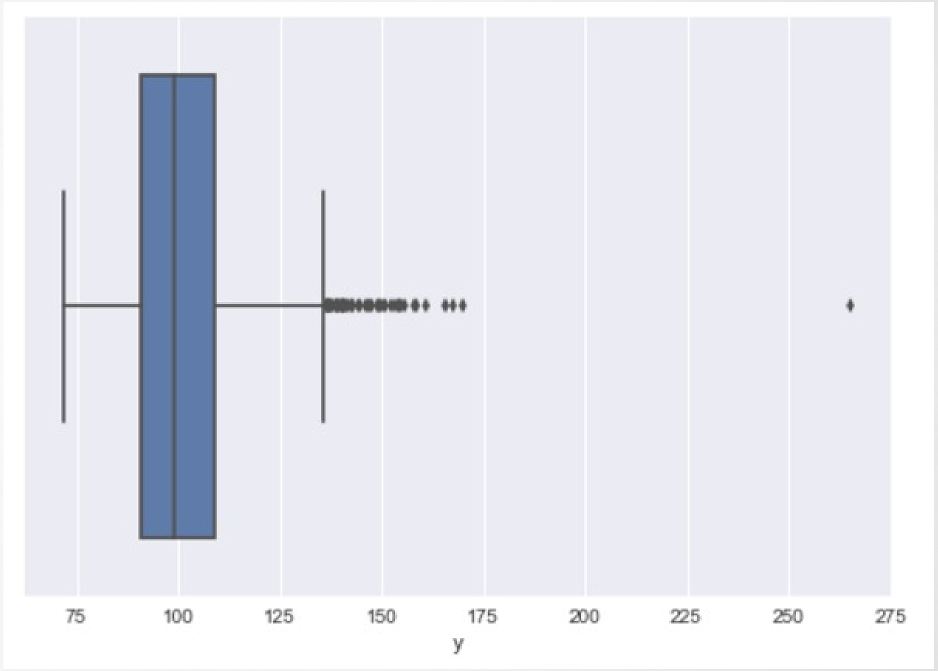
- Is it possible to approximate this dependency with a piecewise-constant function?
- LightGMB implements trees with == condition on splits (categorical_features)
- CatBoost has several methods of working with categorical features
- h2o has different ways of categorical features encoding



https://www.kaggle.com/headsortails/mercedas-2-feature-interactions

Cross-validation v1

- Many outliers – 5-fold cross-validation std 0.05
- Dropping outliers or making stratification with respect to the outliers does not help



Cross-validation v1

- 10 splits into 5 folds – 50 scores for each fold
- To compare two models we the t-test for related samples (scipy.stats.ttest_rel)
- We asses the significance of the difference between two models

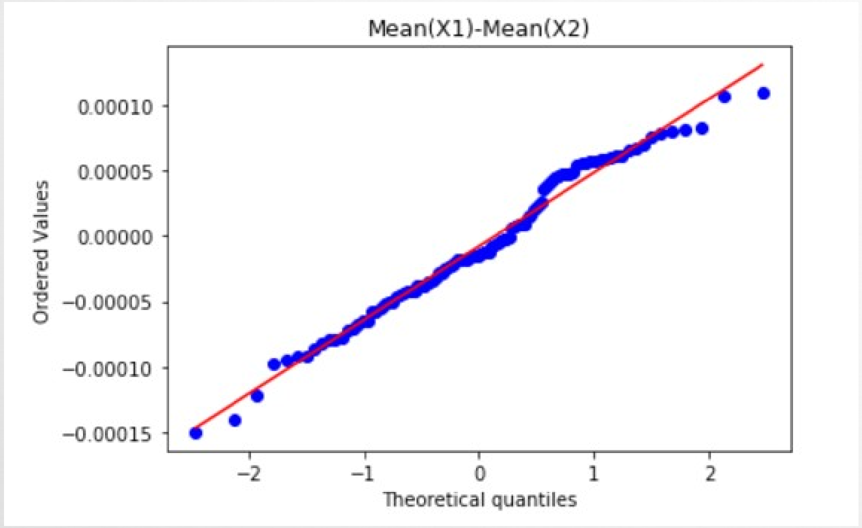
$$T\left(X_1^n, X_2^n\right) = \frac{\bar{X}_1 - \bar{X}_2}{S/\sqrt{n}}$$

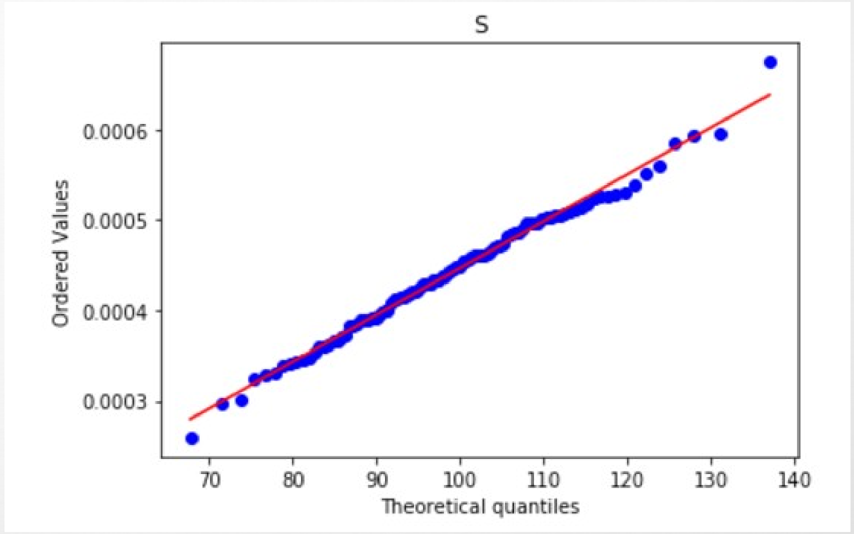
X_1,X_2 – out-of-fold values of R2 for the respective folds, S – deviance of the pairwise differences of X_1 and X_2, n – number of folds

Cross-validation v1: normality

$$T\left(X_1^n, X_2^n\right) = \frac{\bar{X}_1 - \bar{X}_2}{S/\sqrt{n}}$$

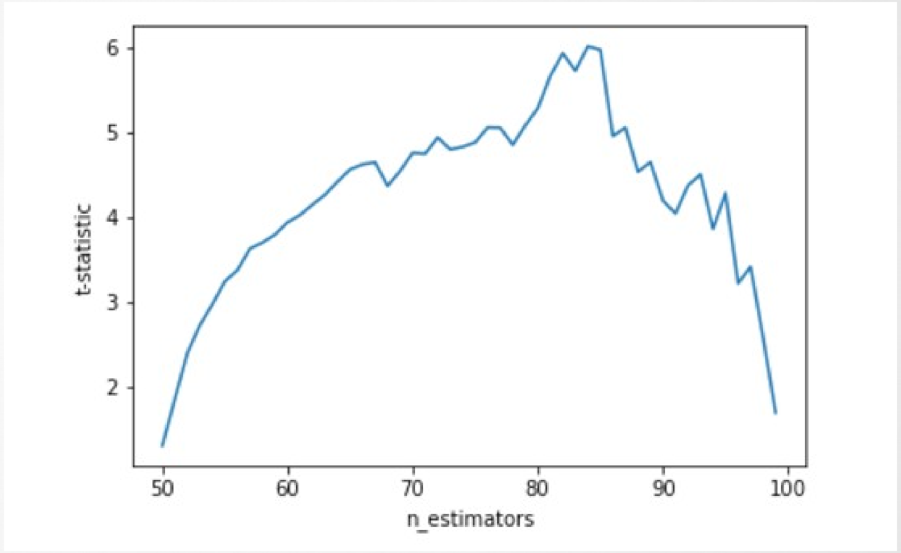
X_1,X_2 – out-of-fold valuer of R2 for the respective folds, S – the deviance of the pairwise differences, n – number of folds





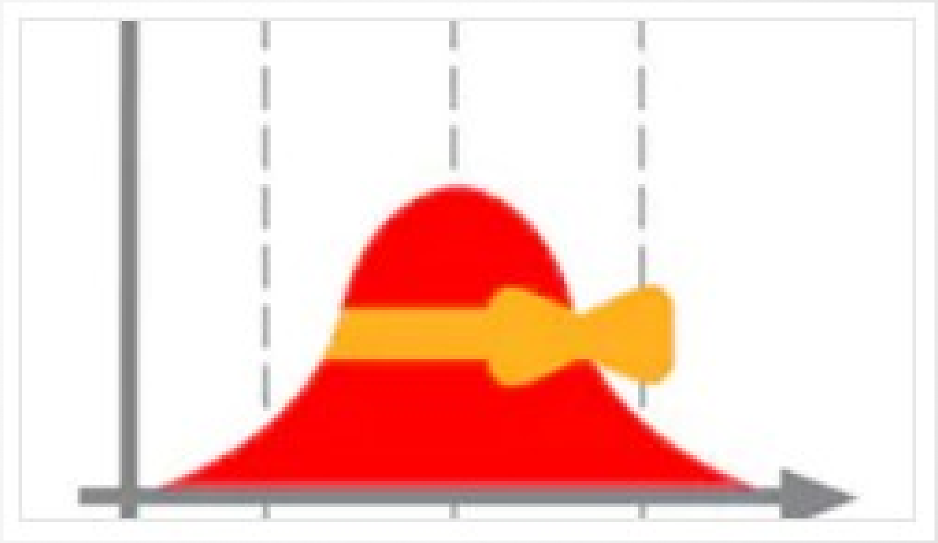
Cross-validation v1: multiple comparisons

- The main strategy is to test less hypothesis
- When changing a hyper-parameter we should observe the tendency of the t-statistic's changing. If t-statistics changes smoothly and has an optimum, where p-value < 0.05, then I believe that I've found the statistically significant difference
- If t-statistic changes rapidly with small changes of hyper-parameter, I believe that there is no difference



Have you encountered unfamiliar concepts in last three slides?

- Very good course in statistics applied to machine learning problems (Russian language):
- https://ru.coursera.org/learn/stats-for-data-analysis
- Actually these are the very basics of statistics and you can find them in any course in statistics



Cross-validation v1: something got wrong
Importance of the holdout set

- I trusted my CV and arrived to R2 = 0.58 on CV by improving my model
- Winner's private score is 0.55551
- My model with CV R2 = 0.58 got public score of 0.54. The baseline model gets 0.55 on public
- Public leaderboard in my case acted as the holdout set

Cross-validation v2: what did get wrong? Importance of holdout set

- I encountered over-fitting of the outliers in the train set
- Cross-validation should compare models rather than asses the model's performance on the new data
- We want our model to predict “normal” (non outlier) objects well and we accept it's inability to predict outliers
- Thus we don't want the outliers to influence the CV score
- It is possible to go further and to drop outliers from the train set. But in this problem it shifts the mean prediction

Cross-validation v2

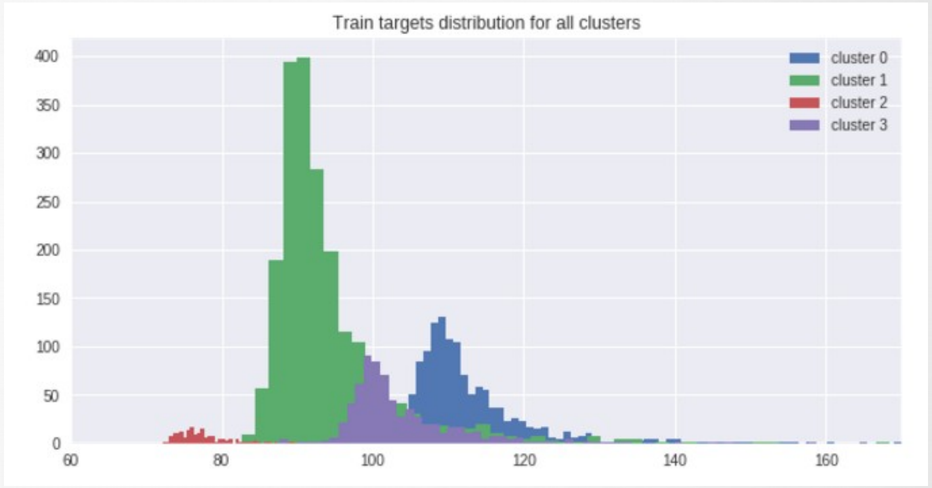
- We get out-of-fold predictions for the entire train set (cross_val_predict)
- We mark as outliers objects with high prediction error
- During CV model model fits all objects in the train folds, but the score is calculated only for “normal” (non outlier) objects of the test folds
- Function cross_validation_score_statement from this repository is used: https://github.com/Danila89/cross_validation_custom

Cross_validation v2

- My feature selection routine appeared to be over-fitting
- The n_estimators parameters appeared to be much to high

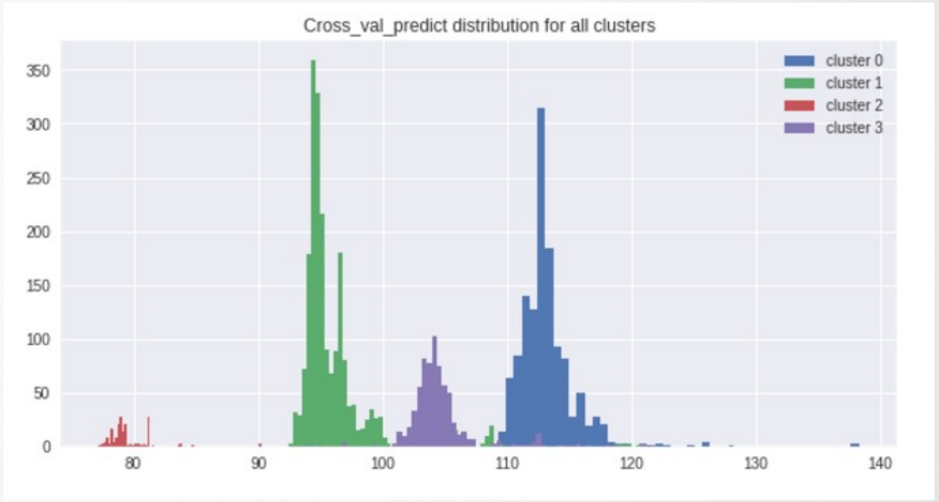
My model: clustering

- We can find four car types using X0 feature: https://www.kaggle.com/daniel89/mercedes-cars-clustering



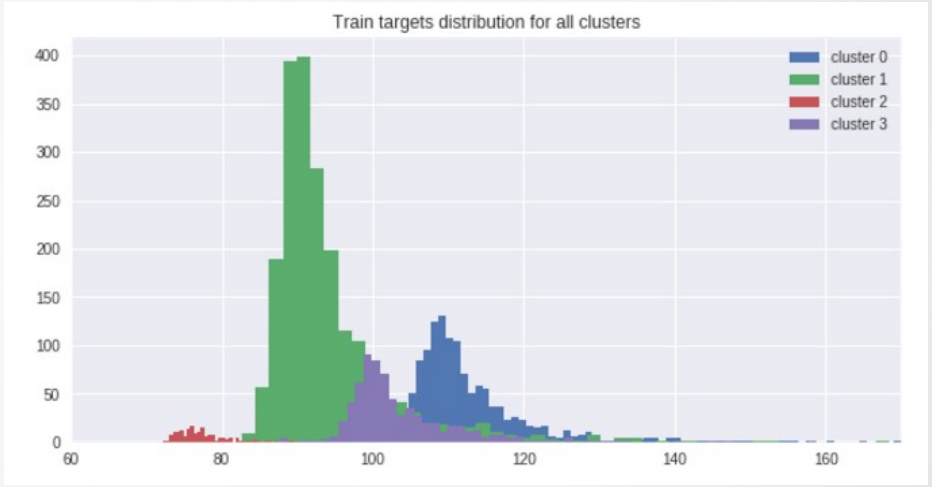
My model: clustering

- Gradient boosting models are finding these clusters even without our help
- The problem is that inside the clusters models perform very poorly and have zero or even negative R2



My moel: clustering

- In the 'red' cluster the R2 is negative. For this cluster we simply predict the cluster mean.
- For every other cluster we train it's own xgboost with own parameters and own features to fit
- But all xgboosts fit the entire train set



My solution: features

- I tried different methods of feature selection as well as dimension reduction methods but ended up with over-fitting
- At the final model I use all binary features, ID and cluster number. Cluster number is calculated using X0, has 4 cardinalities and represents the car type

My model: the xgboost's parameters

- max_depth = 2 for all xgboosts
- n_estimators is less than 100
- I used feature subsampling actively (colsample_bytree, colsample_bylevel)
- Gamma hyper-parameter helped a lot to control the number of leaves

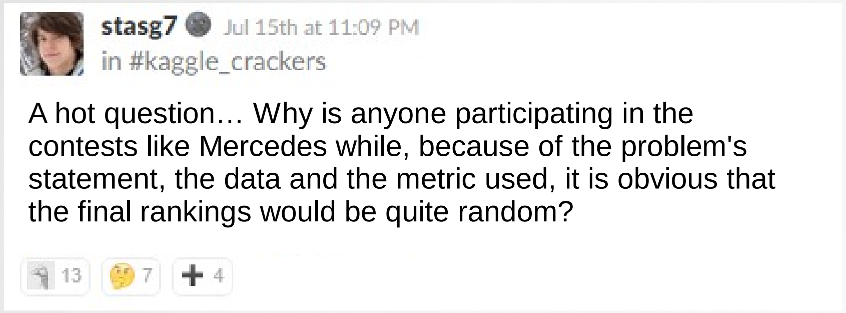
My model: summary

- Divide objects into 4 clusters using X0 feature
- In the worst cluster predict by cluster mean for all objects
- For other clusters tune their own xgboosts, but each of them fits on the entire train set

The 2nd place approach

- Clip the outliers at y=155
- Feature Learner: features are selected randomly, the selection probability is higher for features that have already given higher scores
- https://www.kaggle.com/c/mercedes-benz-greener-manufacturing/discussion/36390
- Now the 1st place approach is also available: https://www.kaggle.com/c/mercedes-benz-greener-manufacturing/discussion/37700

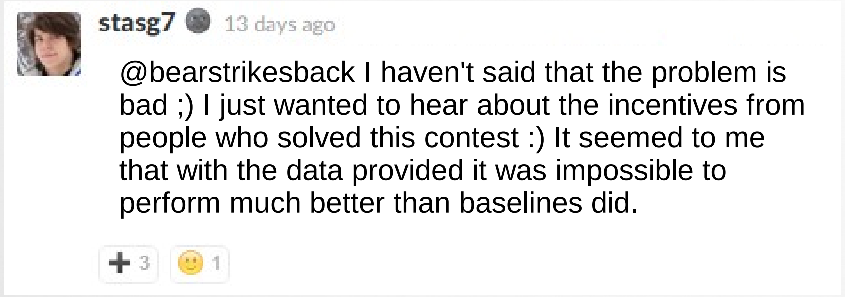
Questions from the colleagues at ods.ai



stasg7 Jul 15th at 11:09 PM
in #kaggle_crackers

A hot question… Why is anyone participating in the contests like Mercedes while, because of the problem's statement, the data and the metric used, it is obvious that the final rankings would be quite random?

13 7 4



stasg7 13 days ago

@bearstrikesback I haven't said that the problem is bad ;) I just wanted to hear about the incentives from people who solved this contest :) It seemed to me that with the data provided it was impossible to perform much better than baselines did.

3 1

- I wanted another medal after Sberbank :)
- The difference between the leader-board's top and baselines is ~ 0.015, but solving this problem I found a lot of new insights:
  - Cross-validation with respect to the significance
  - CV without outliers
  - Necessity of holdout set
  - Risk of outlier over-fitting
  - Clustering data set and cluster-specific approach when building models
  - gamma parameter in xgb
  - Feature Learner from the 2nd place solution
  - ...

Thank you

- Questions
- Comments
- Criticism
- ...