TReNDS Neuroimaging 3rd place solution

2020/07/14

Mobility Technologies

Introduction

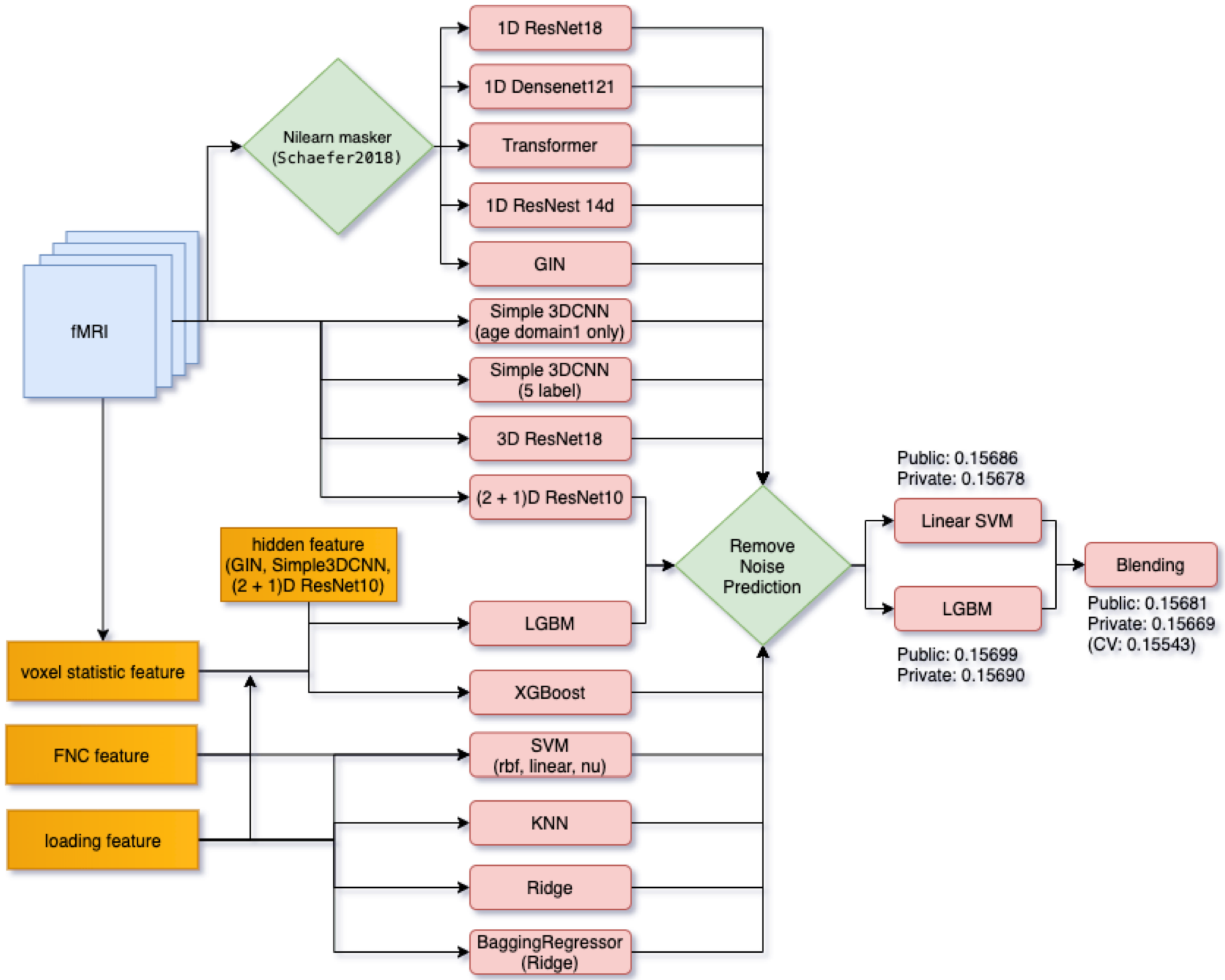
- Name
  - Naoto Shimakoshi (Kaggle name: shimacos)
- Background
  - Master of Mechanical Engineer
  - Working at Mobility Technologies Co., Ltd.
    - Seconded from DeNA Co., Ltd.
    - Providing taxi dispatch service in Japan (like Uber, DiDi).
  - ONODERA’s former colleague :D

My experience working with machine learning at work and at Kaggle has been very useful in this competition!

Solution overview

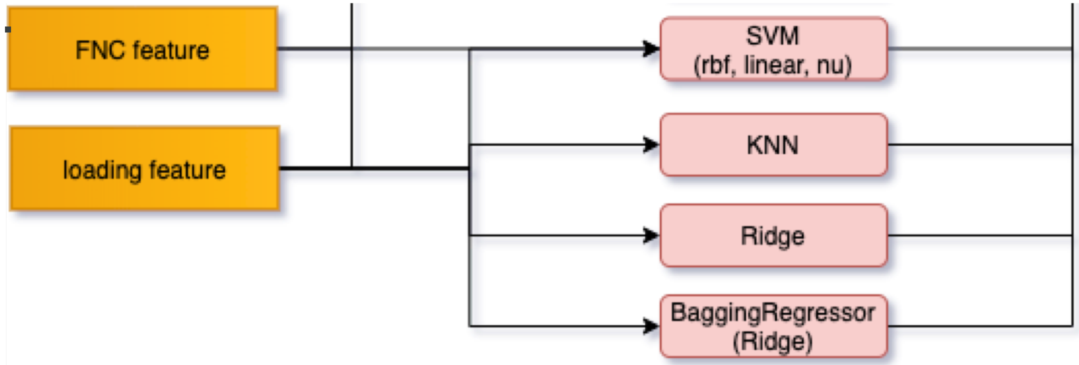
■ Solution Points

1. Stacking
2. Diversity of models for stacking
3. Remove noise caused by the discrepancy distribution between train and test data



Solution Details | Simple model

- Preprocessing
  - I shifted IC_20 feature of train data by the difference between mean(IC_20_test) and mean(IC_20_train).
- I trained simple models by fnc and loding feature.
  - SVM (rbf kernel and linear kernel)
  - NuSVM
  - KNN
  - Ridge
  - BaggingRegressor(Ridge)

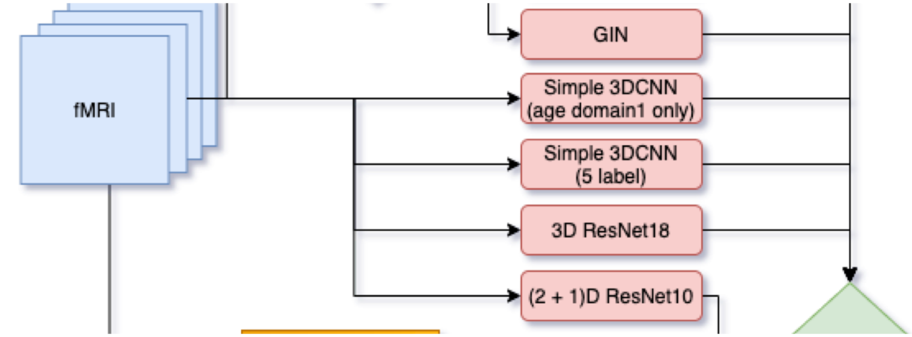


- Some Insights
  - Simple stacking SVM and Ridge, LGBM got private score 0.15905 (bronze zone)
  - BaggingRegressor boosted score (-0.0001)
    - Diversity through feature subset was probably important.

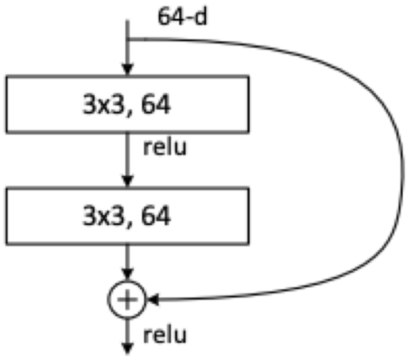
Solution Details | NN model

NN model using 3D voxel

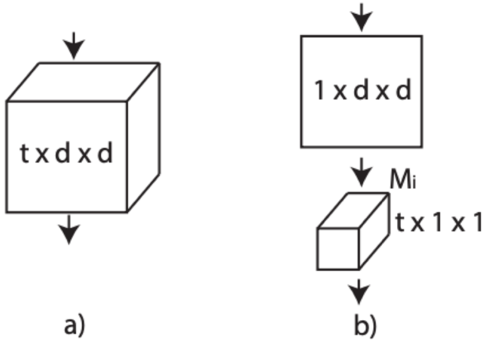
- I applied sample wise and component wise normalization.
- ResNet18 replaced by 3D modules.
- Simple 3dConvNet (Conv3D x 3, MaxPool3D x 3)
  - trained by all label
  - trained by age, domain1*var1 and domain1var2*
- (2 + 1)D CNN
  - better than ResNet18



[1]



[2]



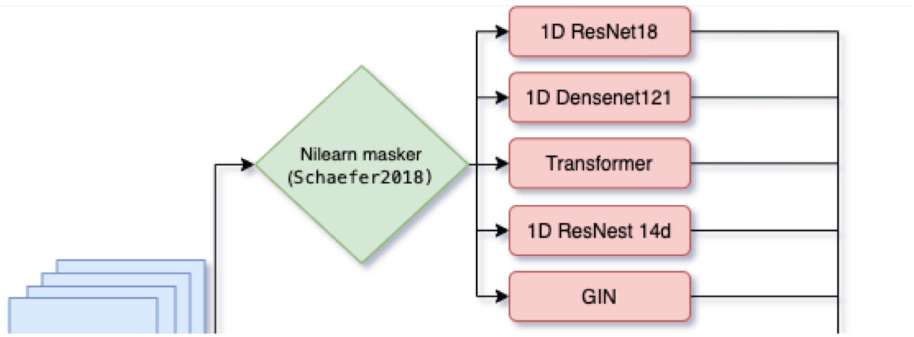
[1] https://arxiv.org/abs/1512.03385
[2] https://openaccess.thecvf.com/content_cvpr_2018/papers/Tran_A_Closer_Look_CVPR_2018_paper.pdf

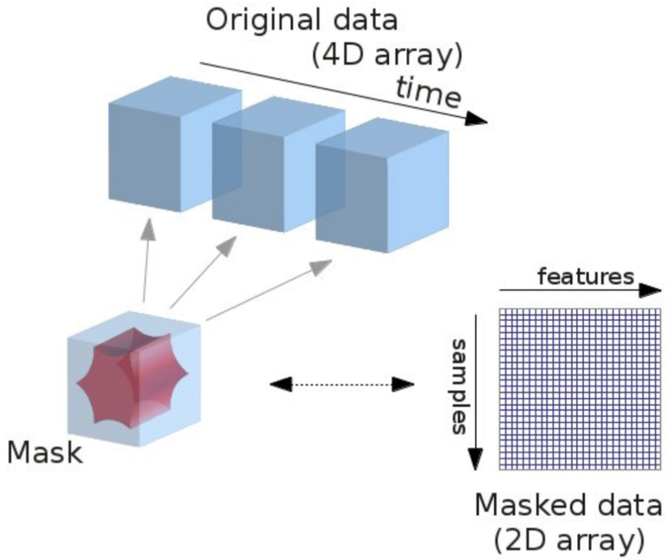
Mobility Technologies

Solution Details | NN model

NN model using masked signal

- Preprocessing
  - I applied masker of schaefer_2018 parcellation.
  - Reduced shape [53, 52, 64, 53] to [53, 400]
- CNN models (input channel is 400)
  - 1D ResNet18
  - 1D densenet121
  - 1D ResNest 14d
- Transformer (feature axis is 400)
  - Self Attention of components by components
- GIN (Graph Isomorphism Network)
  - I used the network architecture like [2] which number of layers I replaced five to three.





[1]

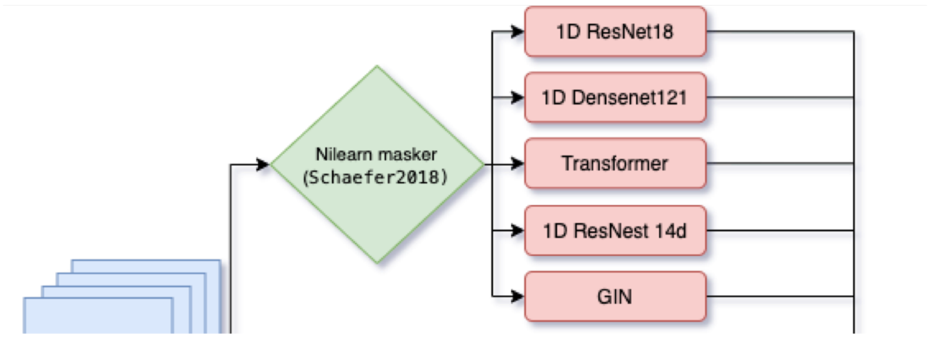
[1] https://nilearn.github.io/auto_examples/plot_decoding_tutorial.html#sphx-glr-auto-examples-plot-decoding-tutorial-py

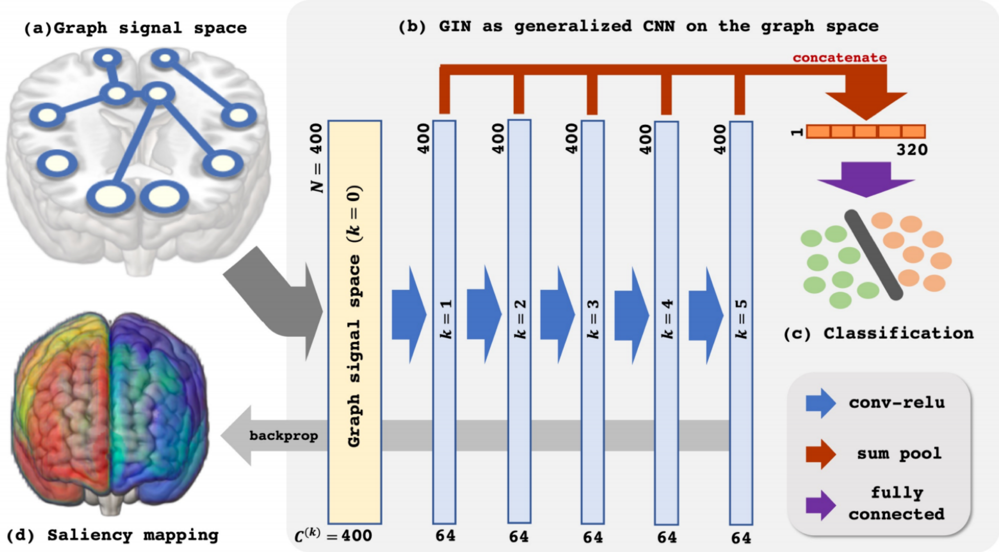
[2] https://arxiv.org/pdf/2001.03690.pdf

Solution Details | NN model

NN model using masked signal

- Preprocessing
  - I applied masker of [schaefer_2018] parcellation.
  - Reduced shape [53, 52, 64, 53] to [53, 400]
- CNN models (input channel is 400)
  - 1D ResNet18
  - 1D densenet121
  - 1D ResNest 14d
- Transformer (feature axis is 400)
  - Self Attention of components by components
- GIN (Graph Isomorphism Network)
  - I used the network architecture like [2] which number of layers I replaced five to three.



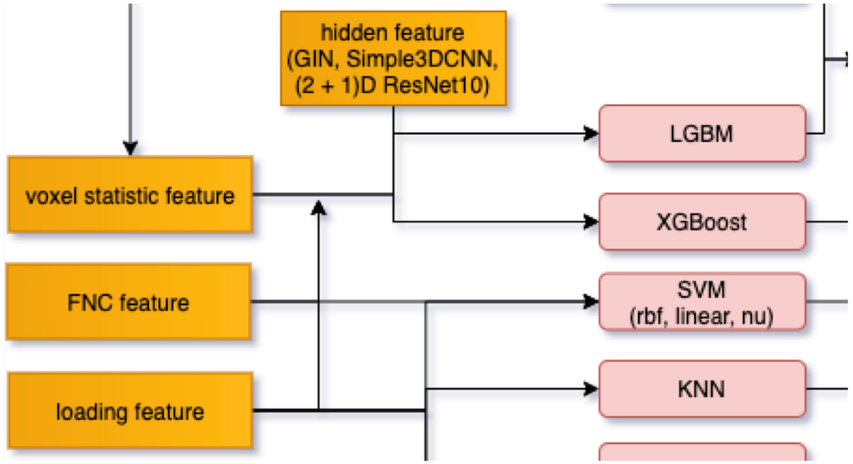


[2]

[1] https://nilearn.github.io/auto_examples/plot_decoding_tutorial.html#sphx-glr-auto-examples-plot-decoding-tutorial-py
[2] https://arxiv.org/pdf/2001.03690.pdf

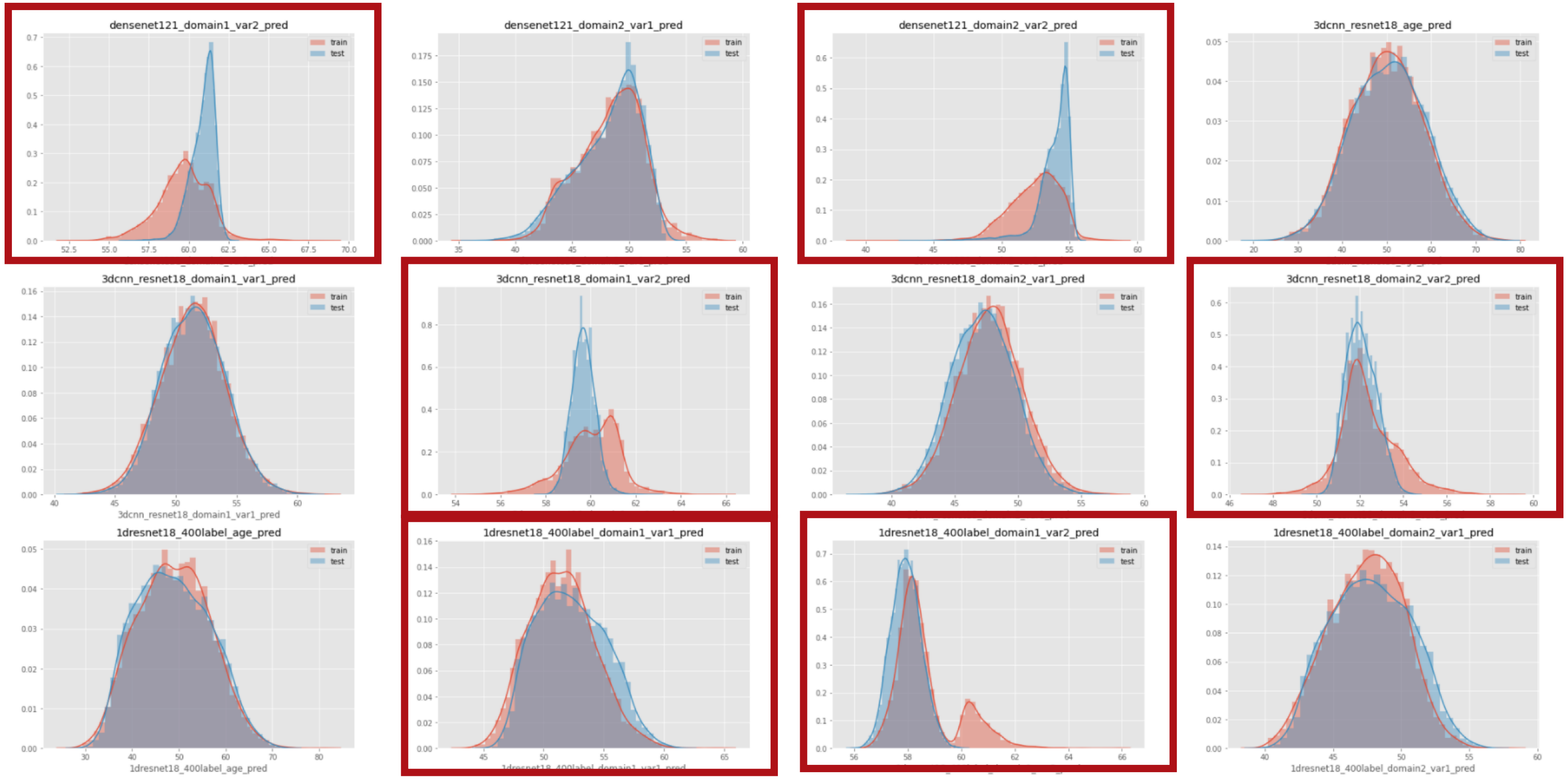
Solution Details | GBM with NN features

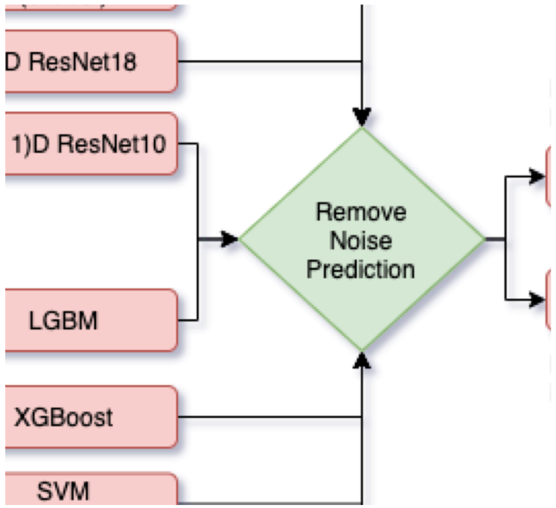
- Preprocessing
  - I computed voxel statistical feature (mean, max, kurt, skew) of each components.
  - Extract hidden feature from (2 + 1)D CNN, Simple 3dConvNet and GIN.
- GBM models with NN features and tabular features.
  - LGBM(XGBoost) model with hidden feature of (2 + 1)D CNN
  - LGBM(XGBoost) model with hidden feature of Simple 3dConvNet
  - LGBM(XGBoost) model with hidden feature of GIN
- Parameter tuning
  - I selected 1024 feature of each models by feature importance and reduced the number of leaves to avoid over-fitting (n_leaves = 2).



Solution Details | Remove Noisy Predictionn

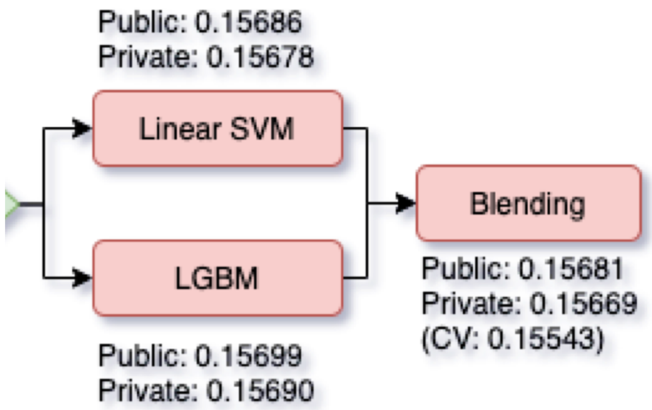
- I just looked at the distribution of predictions in the training and test data and erased them if there were too much discrepancy.



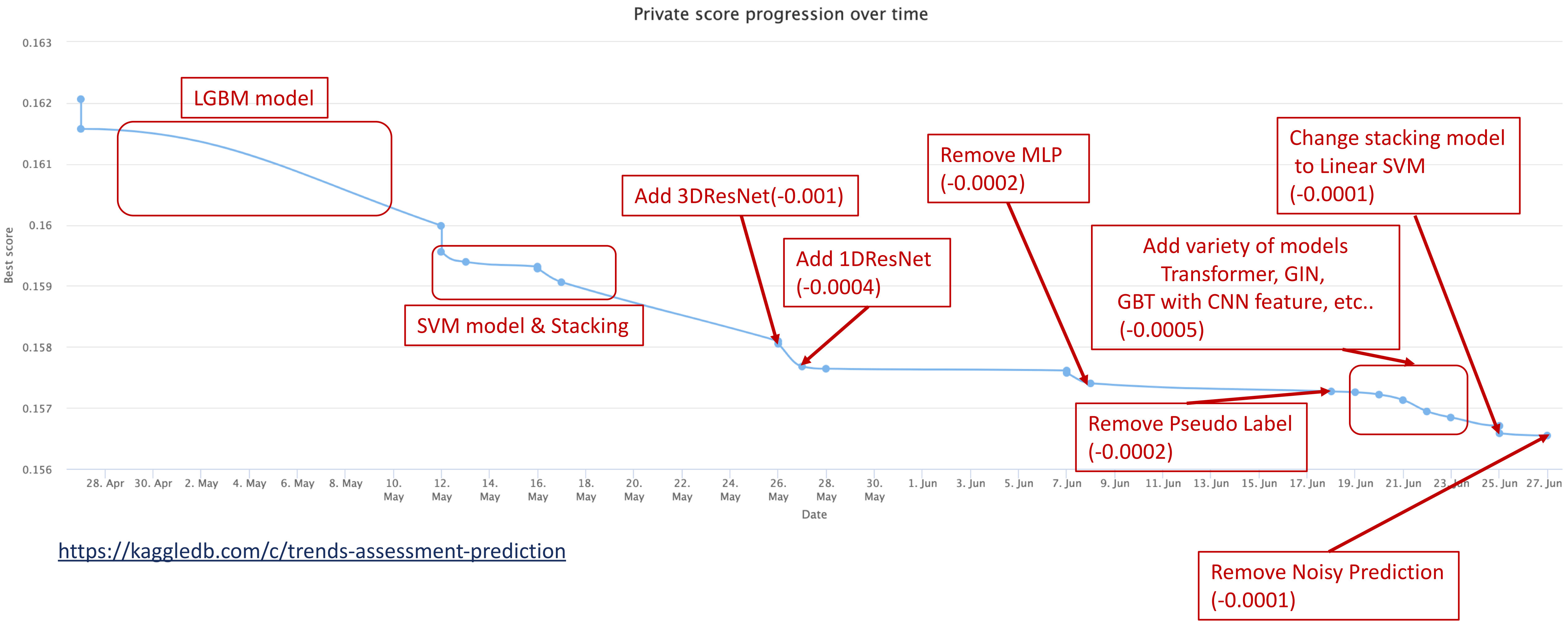


Solution Details | Stacking & Blending

- Preprocessing
  - I made domain interaction features per each models.
    - domain1_var1_pred + domain1_var2_pred, …
    - abs(domain1_var1_pred - domain1_var2_pred), …
    - domain1_var1_pred x domain1_var2_pred, …
- Age prediction
  - I used only prediction features of age.
- Domain prediction
  - I used prediction features of all target and interaction features.
- Stacking
  - Linear SVM (better than LGBM)
  - LGBM
- Blending
  - Weighted mean predictions of above two models



My Leaderboard Chart



https://kaggledb.com/c/trends-assessment-prediction

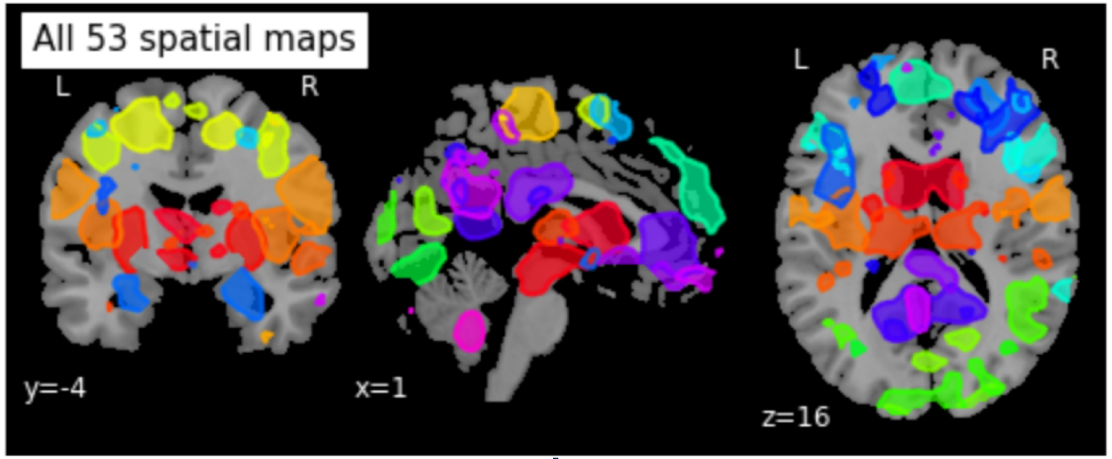
Some Insights

- My best single model was 3DCNN with tabular feature. (Private: 0.15845, Public: 0.15854)
  - But when I added this model to stacking, I got worse score.
  - This is because model using only fMRI extracted more important information than those using tabular feature.
- Removing noisy prediction had important role on shake-up.
  - Maybe the discrepancy of distribution between train and test is caused by the difference between site1 and site2.

	Public LB	Private LB
Without Removing noise	0.15672	0.15690
With Removing noise	0.15669	0.15679

This competition points in my opinion

- How to extract feature from fMRI data.
  - fMRI data have different information from table features, and it is very important to boost score.



?

fMRI Feature

・PCA (1st place solution)
・Neural Networks (2nd, 3rd , 5th)
・Masker (3rd , 4th)
・Auto Encoder (6th)
・Other feature engineering

This competition points in my opinion

- How to produce diversity.
  - Different model architecture. (my solution)
  - Feature subset selection (1^{st} , 4^{th})
  - Different input area of fMRI to NN(2^{nd})