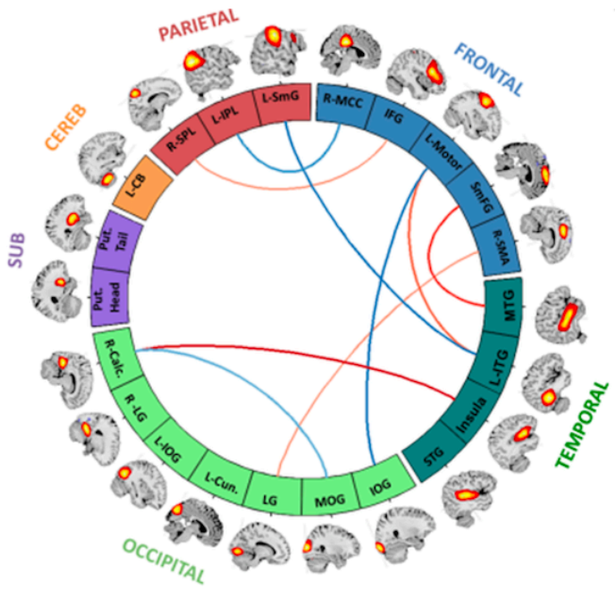
TReNDS Neuroimaging
まとめ

成田泰基

どんなコンペだったか



https://www.kaggle.com/c/trends-assessment-prediction/overview

説明変数

・rsfMRIデータ

・sMRIデータ(構造データ)

目的変数

- 年齢
- 匿名変数1-1
- 匿名変数1-2
- 匿名変数2-1
- 匿名変数2-2

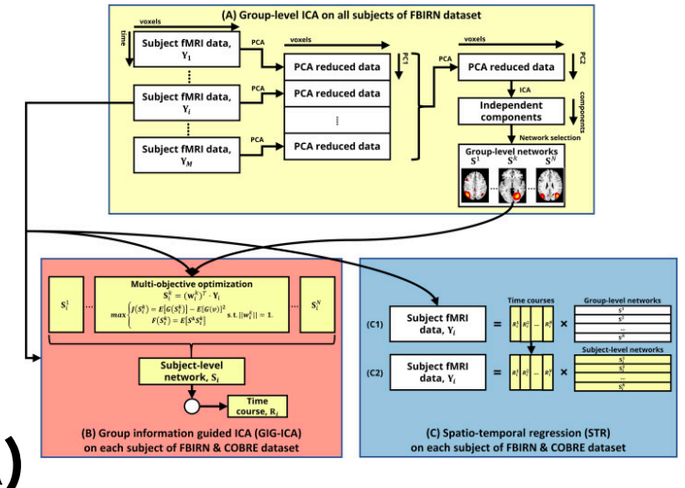
使用データ

4D image

- fMRIデータ
- (IC,x,y,z)で構成される4Dの画像データ
- subject-level空間コンポーネントが53存在(by GIG-ICA)

fnc.csv

- fMRIデータ
- functional network connectivity
- 53コンポーネント同士の時間の相関行列(by GIG-ICA)



https://www.sciencedirect.com/science/article/pii/S221315821930097X

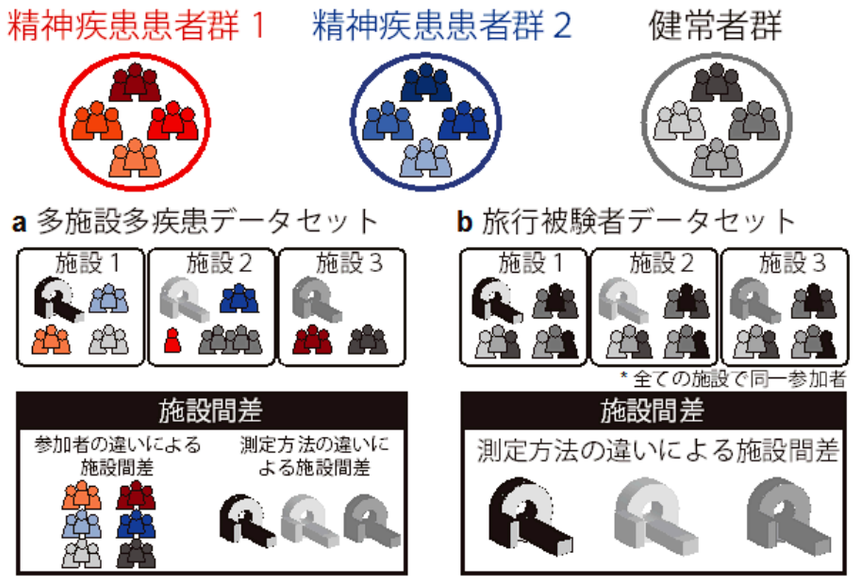
loading.csv

- sMRIデータ
- source-based morphometry (SBM) loadings
- 灰白質濃度マップをGroup-level ICA

異なるsite

訓練データセット

→ site1のみで構成されていた



https://www.amed.go.jp/news/release_20190419-01.html

テストデータセット

→ site1+site2

site1データを用いて、
site1データとsite2データを汎化させる必要があった

4Dimageでの予測

3DCNN + ResNet

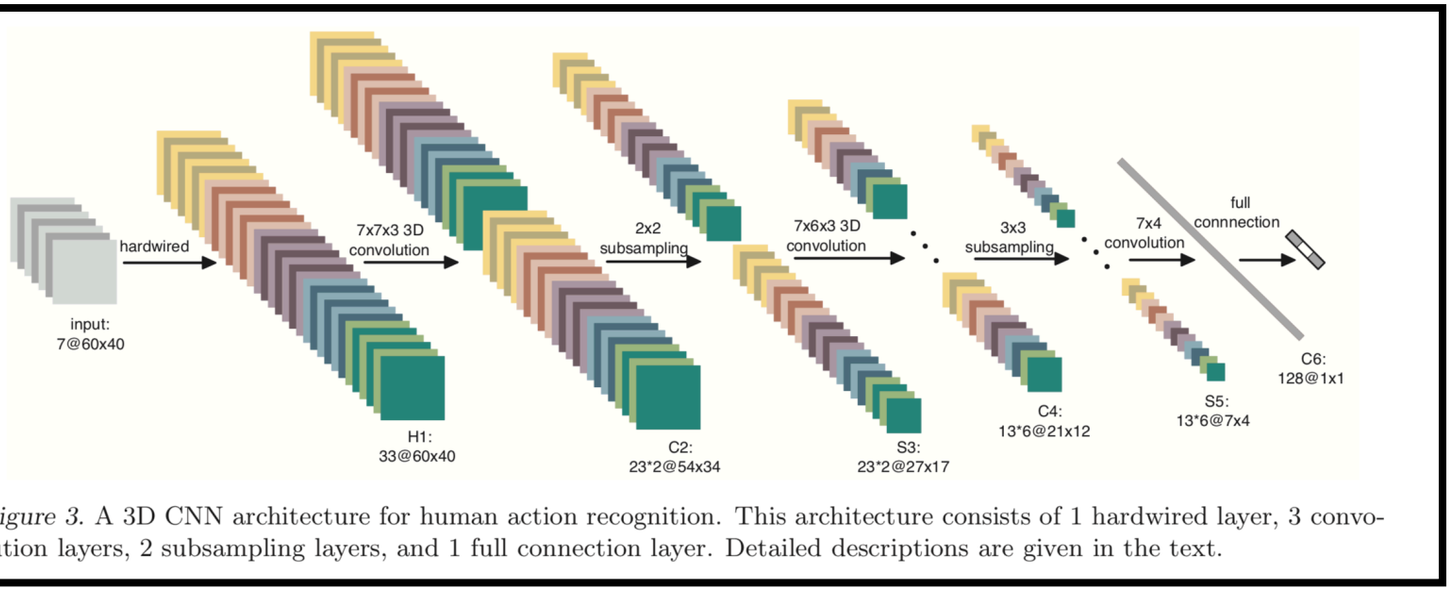


Figure 3. A 3D CNN architecture for human action recognition. This architecture consists of 1 hardwired layer, 3 convolution layers, 2 subsampling layers, and 1 full connection layer. Detailed descriptions are given in the text.

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.442.8617&rep=rep1&type=pdf

Resnet参考:https://deepage.net/deep_learning/2016/11/30/resnet.html

▼

・過学習しまくる

・Public: 0.17~（Tableデータのみ:0.15台）

・学習が難しくて4Dimageでの予測は微妙。。。

fnc+loadingでの予測

- (rapids)SVR
  → Public : 0.159
- Ridge Regression
  → Public : 0.160
- Bagging Regresser (Ridge)
  → Public : 0.159

比較的出ていた

- LGBM
  → Public : 0.162
- MLP
  → Public : 0.161
- kNN,RF,etc...
  → Public :0.166~

そこそこ

fnc+loading（テーブルデータ）でのアンサンブルが賢明か・・・？

我々のモデル

1層目

・SVM(特徴選択、チューニング後)

・bayesian Ridge(特徴選択)

・bayesian Ridge(PCA+ICA)

・LightGBM(raw+PCA+ICA+proba_site2)

・GPR(特徴選択)

・GPR(loadingのみ)

・Ridge(いぶきくんのやつNo2)

・BaggingRegressor(PCA+ICA)

2層目

・bayesian Ridge(targetごとに個別の予測値を使用)

submit = bayesian RIdge*0.7 + stacking kernel*0.3

fnc+loading
(テーブルデータのみ)
のスタッキング

・Public : 0.15880
93位/1047位

・Private : 0.15901
66位/1047位

備考

- site adversarial validation(site予測)
  - ー 非常に困難だった
  - ー 成功した人がいるらしい・・・

- 4Dimageデータについて
  - ー 同じような値を出力する微妙モデルができる
  - ー 次元圧縮には時間的コストとメモリ的にきつかった
  - ー スタッキングで用いることで効果を発揮したらしい

備考

- site adversarial validation(site予測)
  - 非常に困難だった
  - 成功した人がいるらしい・・・

- 4Dimageデータについて
  - 同じような値を出力する微妙モデルができる
  - 次元圧縮には時間的コストとメモリ的にきつかった
  - スタッキングで用いることで効果を発揮したらしい

命運を分けた

1st solution

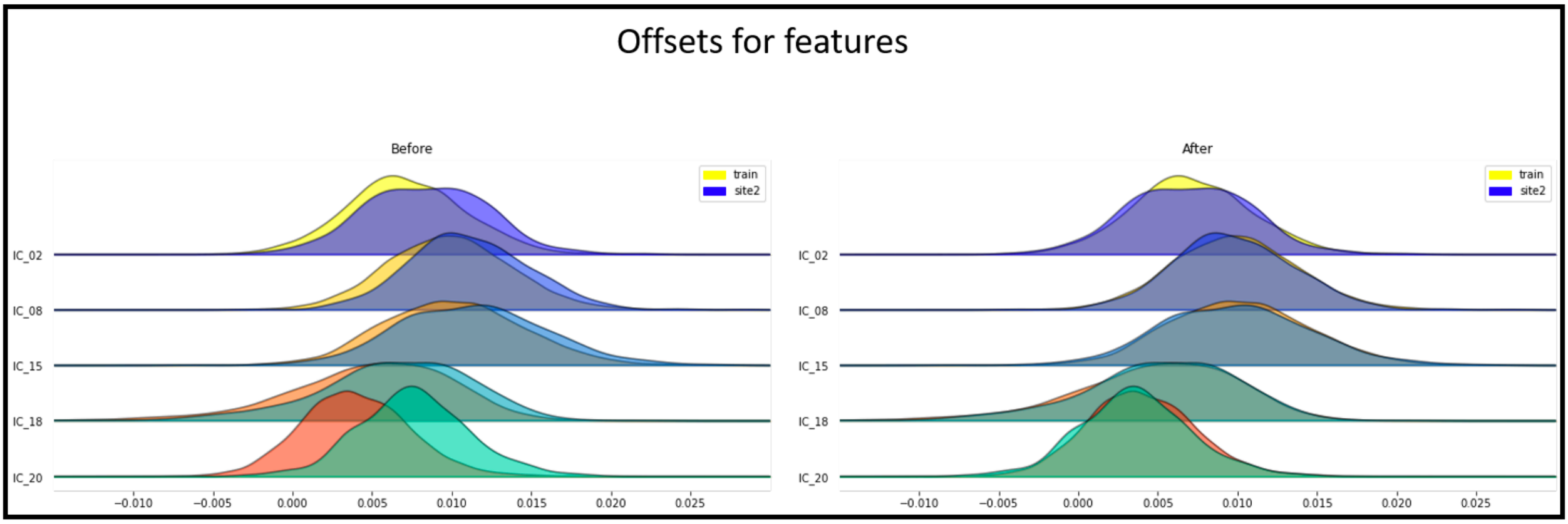
I. Generate features from 3D fMRI data.

- 3DCNNは微妙だったので4Dimageを次元削減して特徴抽出する方法に移行
- データの高次元性が問題
- scikit-learnによるバッチ学習で次元削減
    - Incremental PCAのpartial_fit()
        - 54個のICコンポーネントは６つのグループに分けてflatten
        - n_components 200、バッチサイズ200でそれぞれ実行
          合計1200個の特徴量を作成
        - 24スレッドプロセッサと64GB RAMで9時間
    - dictionary-learning
        - n_components=100, batch_size=100
        - 計算に2日
- どちらもtrain+testを結合して学習

1st solution

II. Train-test matching

- train/testで分布が離れている列に対してバイアスを追加
- train[col]とtest [col] +の間のコルモゴロフスミルノフ検定の統計の最小化
- site2のデータ（分類器で予測された）、testのデータそれぞれにこの処理をした
- submitするデータに対しても同様な後処理を施した



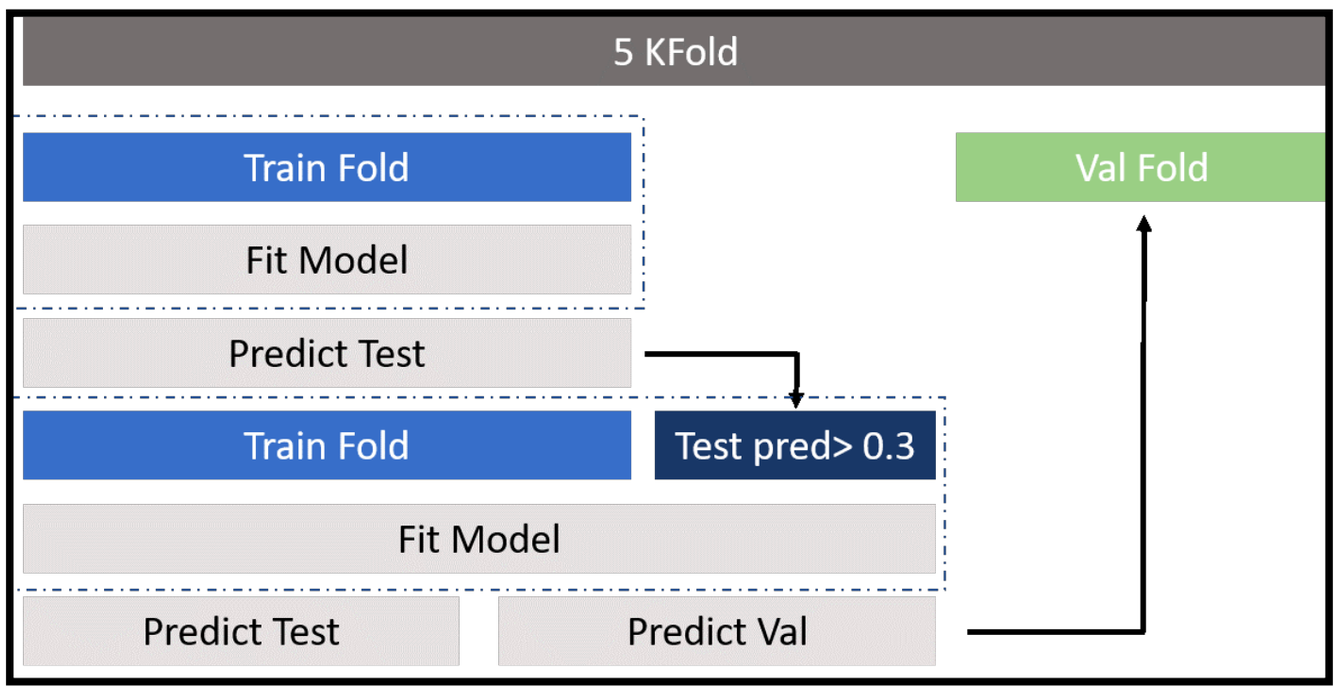
1st solution

II. Train-test matching

- site予測
  - loading, fnc, pca featureを使用
  - Elastic Netを使用
  - 2回の学習を行う

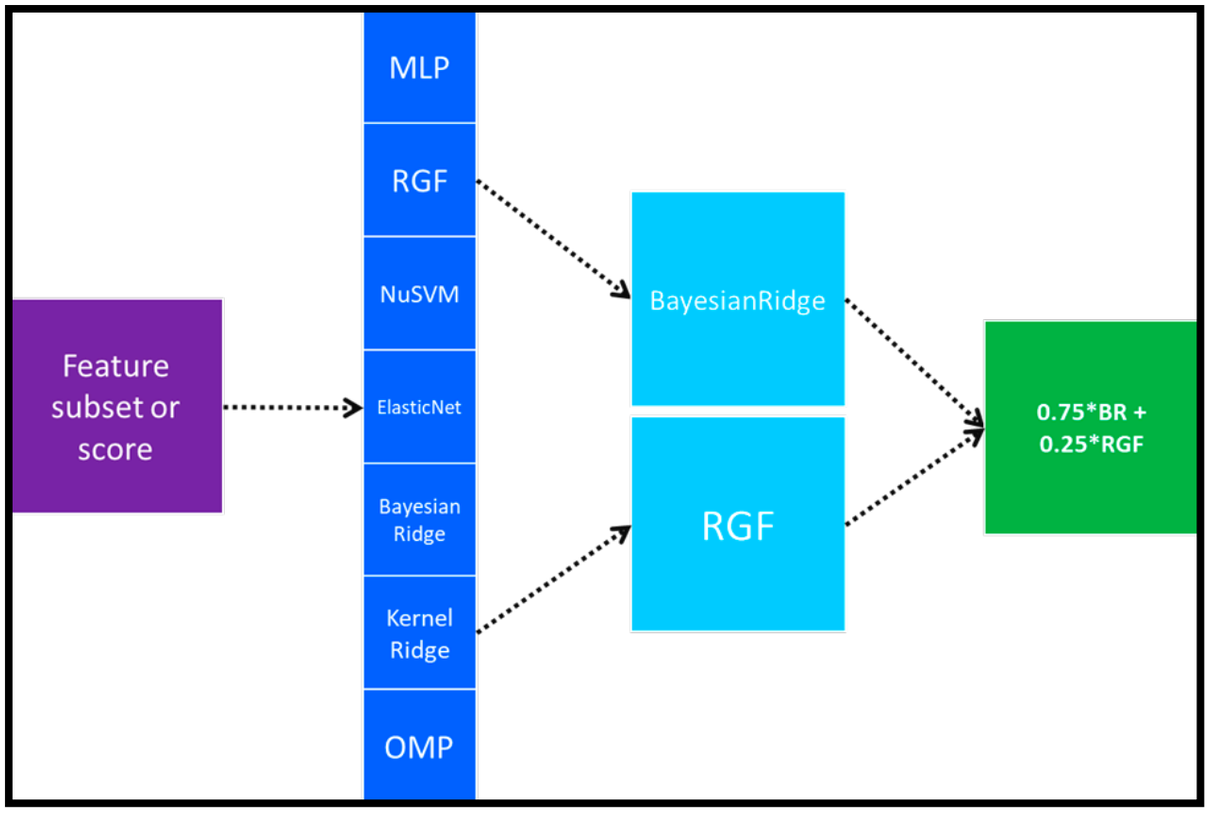
→1回目で0.3以上の確率だったものを加えて再学習

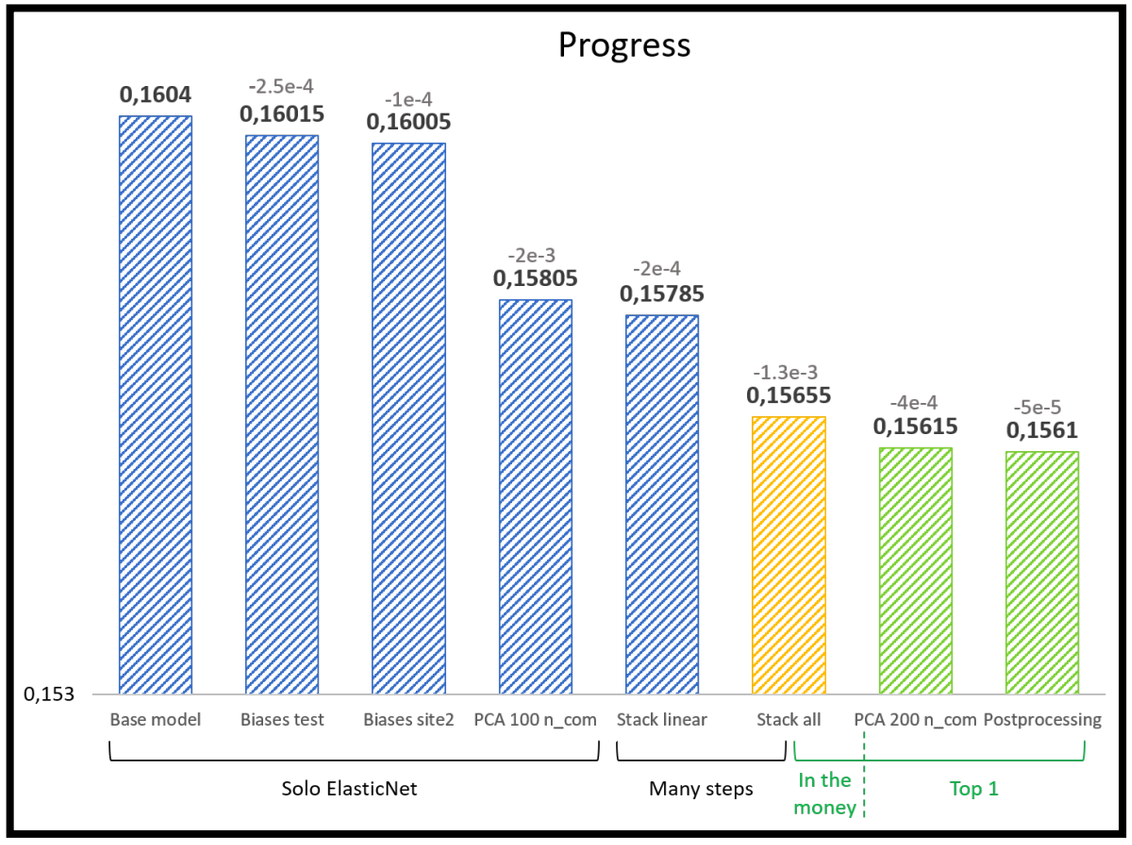
- 最終的にAUC：0.973



1st solution

III. Training of a diverse ensemble





スタッキングの構造　　スコアの推移

Public : 0.15600　Private : 0.15612

https://www.kaggle.com/c/trends-assessment-prediction/discussion/163017

反省点

- 知識量（引き出しの多さ）の大切さ
- 序盤からスタッキングを想定したコードを書く
- ニューラルネットをもっと気軽に扱えるように

反省点

おつかれさまでした！！！