2020･05･11　ゼミ資料

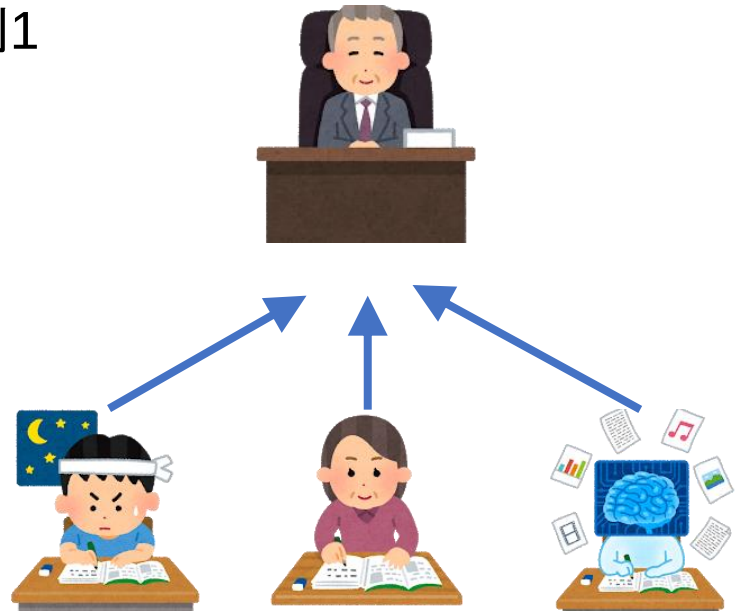
Pythonではじめる教師なし学習
2章9節～12節

1116 17 9036
山口真哉

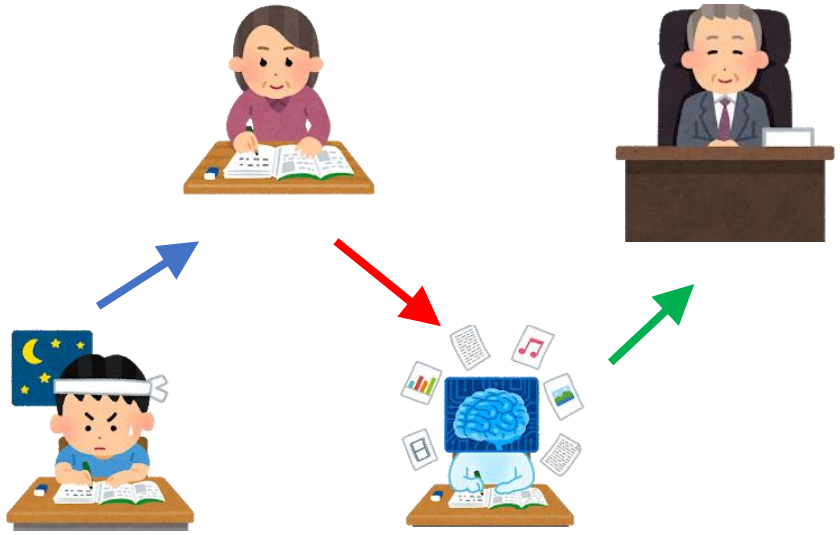
アンサンブル

アンサンブル学習… 複数のモデルに学習させて、全ての予測結果を統合することで、
性能を向上させる手法. (詳しくはKaggle Ensembling Guideに書いてある)

例1

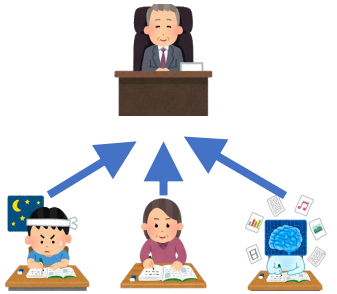


例2

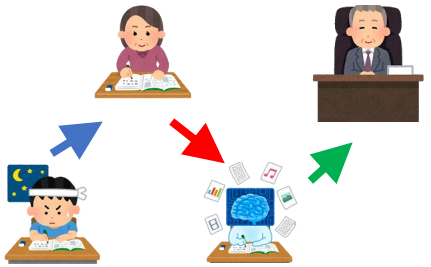


アンサンブル

バギング …　同時に複数のモデルに学習させて、
予測結果の平均をとることで性能を向上させる.
並列処理ができる.



ブースティング …　データの一部を抽出して学習し、
前回の結果を利用する性能を向上させる.
学習結果を次のモデルの学習に反映させるため
同時に処理することはできないが、精度は高まる場合が多い.



アンサンブル

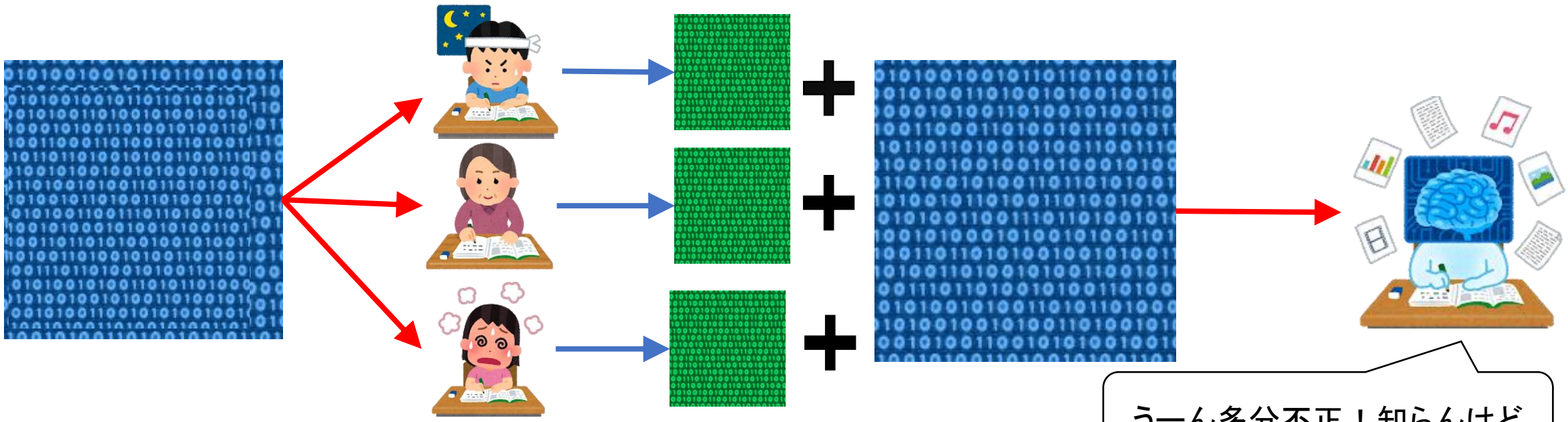
アンサンブルのメリット: 個々のモデルが同程度に強力かつ相関があまりなければ,
個々のモデルの欠点を補い合い元のどれよりも高い性能を示す.

アンサンブルのデメリット:
- 1つが他のモデルより優れている場合,
  一番優れたモデルと同じくらいになり, 他のモデルは機能しない.
- モデルに強い相関があると, アンサンブルで組み合わせも
  多様な判断ができるようにはならない.

アンサンブル

スタッキング　…　・個々のモデルから得た予測値を元の訓練セットの
特徴量集合に追加する.

・それで得た(特徴量+予測値)をk分割交差検証で学習する.

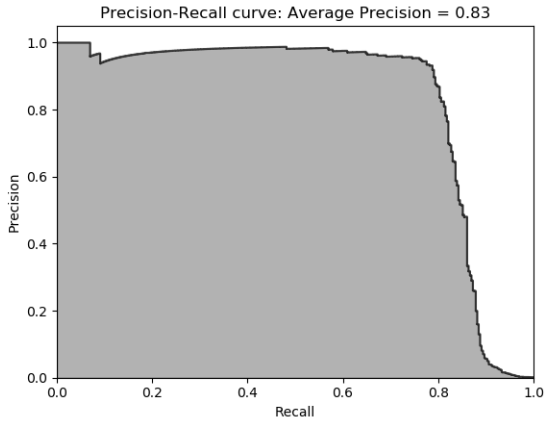


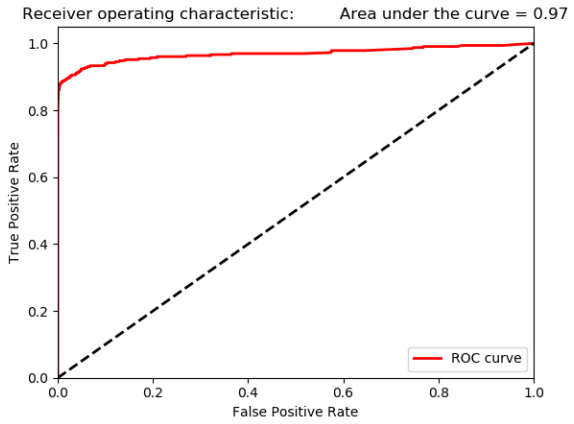
アンサンブル

やってみた

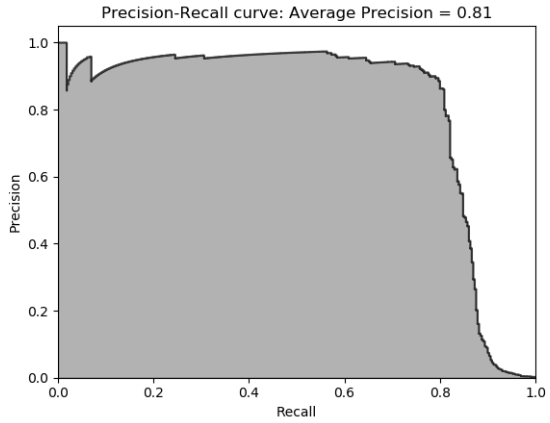
LightGBM→

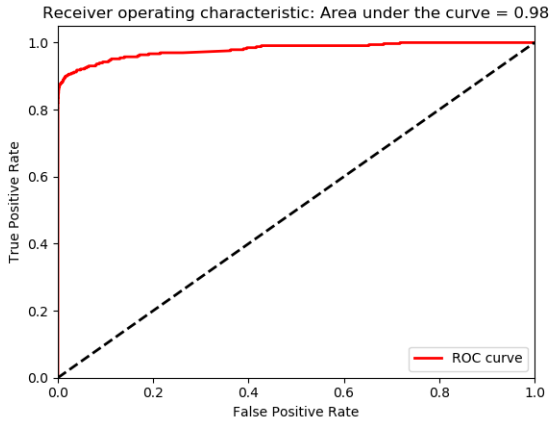
対数損失:0.00297





対数損失:0.00288





←スタッキング

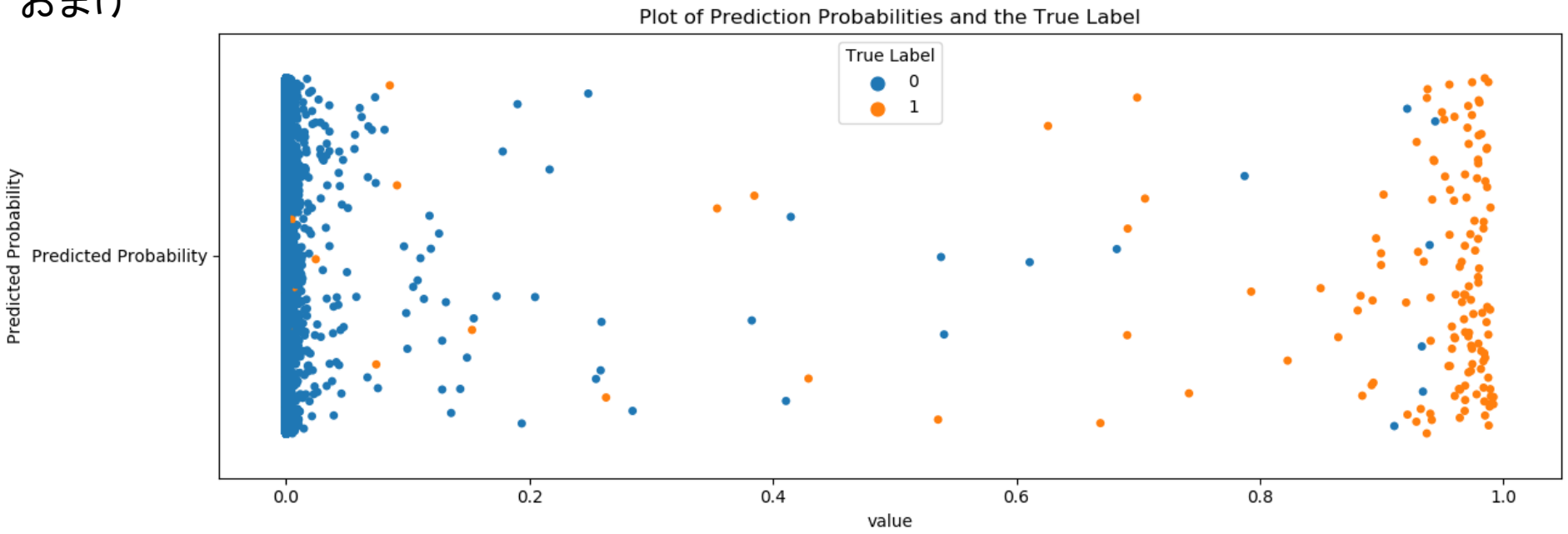
改善は
見られない

⇒最良の2つが
両方勾配ブースティング
だったためと思われる.

～まとめ～

・アンサンブルではいい結果が得れなかったので
実運用では高速なLightGBMを単独で使った方が良い.

おまけ



実運用システムパイプライン

実運用するときは以下のような手順を踏む

1. 新しいデータを取り込んで’newData’というDataFrameに格納

2. データをスケール変換（標準化)

```
newData.loc[:,スケール変換する特徴量] =
    sX.transform(newData[スケール変換する特徴量]
```

3. LightGBMで予測

```
gbm.predict(newData, num_iteration=gbm.best_iteration)
```