データコンペ結果報告

SIGNATE　Beginner 限定コンペ

中三川 紗菜

応用数理学科

2025 年 1 月 9 日

目次

目次

データコンペの選定動機

なぜデータコンペに参加したのか

- 未経験分野への興味
- 将来携わりたい分野への活用

なぜ SIGNATE の Beginner 限定コンペか

- 使用言語が日本語で，また学習データが扱いやすいため，参加のハードルが低い

目次

問題設定

参加したコンペ：【第 50 回_Beginner 限定コンペ】健診データによる肝疾患判定

- 目的：健康診断データに基づいた肝疾患の有無を判定するモデルの構築
  - 850 人の検診データを基に，新たに 350 人の肝疾患の有無を予測
  - 学習データの説明変数は 10 個 (欠損値なし)
- 期間：1ヶ月 (9/1〜9/30)
- １日５回まで投稿可能

説明変数は 10 個で，目的変数'disease' について 0 以上１以下の予測値を提出する．

表 1: 説明変数と目的変数

	項目名	データ型	説明
1	Age	int	年齢
2	Gender	int	性別
3	T_Bil	float	総ビリルビン
4	D_Bil	float	直接ビリルビン
5	ALP	float	アルカリフォスターゼ
6	ALT_GPT	float	アラニンアミノトランスファーゼ
7	AST_GOT	float	アスパラギン酸アミノトランスフェラーゼ
8	TP	float	総蛋白
9	Alb	float	アルブミン
10	AG_ratio	float	アルブミン/グロブリン比
11	disease	int	肝疾患の有無 (0: 無,1: 有)

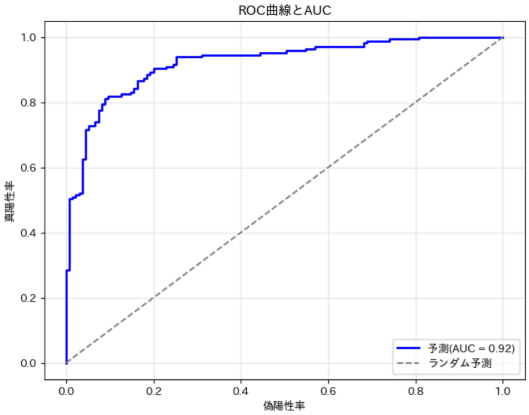
学習データ (イメージ)[8]

id	Age	Gender	T_Bil	D_Bil	ALP	ALT_GPT	AST_GOT	TP	Alb	AG_ratio	disease
0	60	1	2.9	1.3	170.9	42.1	37.1	5.5	2.9	1.01	1
1	28	0	0.7	0.1	158.8	26.0	23.9	6.4	3.7	1.36	0
2	60	1	23.1	12.5	962.0	53.0	40.9	6.8	3.3	0.96	1
3	20	1	1.0	0.5	415.9	33.9	39.0	7.0	3.8	1.31	0
4	44	0	0.6	0.3	152.9	40.9	42.0	4.5	2.1	1.04	0
5	62	1	11.1	5.7	699.0	64.0	100.1	7.4	3.3	0.64	1
6	32	1	12.4	6.0	514.9	48.1	92.1	6.5	2.5	0.81	1
7	37	0	0.8	0.1	152.0	89.9	20.9	7.0	4.3	1.43	0
8	41	1	0.9	0.2	114.0	20.9	22.9	7.0	3.1	1.04	0
9	14	1	1.7	0.6	268.8	58.0	45.1	6.7	3.9	1.21	1
10	63	1	0.7	0.2	184.0	48.9	46.1	6.7	3.9	1.5	0

評価指標：AUC(Area Under the ROC Curve)

ROC 曲線を基に計算される，２値分類における代表的な評価指標．[1]

- ROC 曲線の下部の面積が AUC．
- ROC 曲線は，0 以上１以下の予測値の大小関係を元に描かれる．



目次

大まかな流れ

1. 学習データの各特徴量の分布を確認する (大まかに知る).
2. 良い手法を見つける.
3. パラメータチューニングを行う.
4. モデルのブレンディングを行う.
5. 各特徴量について調べ，変更・新たに生成し追加する.
6. 過学習していないか確認するためにクロスバリデーションを行う.

→最終順位を決めるモデルを１つ選ぶ.

良い手法を見つける

Pycaret

Python で利用可能な主要な機械学習ライブラリが統合され，モデルの作成，比較，チューニング，評価等が行えるライブラリ．

- Pycaret を用いて，様々なモデルを比較し，最も AUC の高いものを見つける．
- Pycaret にはデータの前処理と分割を同時に行う関数'setup'[2] があるが，今回扱うデータは欠損値がなく全て数値データのため，前処理なし．

比較結果

この結果から，上の３つのモデルについて重点的に調べた．

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
catboost	CatBoost Classifier	0.8689	0.9469	0.8335	0.8692	0.8491	0.7333	0.7362	0.3230
rf	Random Forest Classifier	0.8671	0.9431	0.8293	0.8692	0.8452	0.7292	0.7336	0.0300
et	Extra Trees Classifier	0.8656	0.9458	0.8333	0.8648	0.8463	0.7270	0.7305	0.0260
gbc	Gradient Boosting Classifier	0.8587	0.9406	0.8182	0.8593	0.8355	0.7121	0.7155	0.0370
xgboost	Extreme Gradient Boosting	0.8520	0.9381	0.8185	0.8459	0.8306	0.6994	0.7014	0.0210
ada	Ada Boost Classifier	0.8454	0.9167	0.8145	0.8379	0.8241	0.6862	0.6889	0.0160
lr	Logistic Regression	0.8420	0.9100	0.7309	0.9006	0.7994	0.6727	0.6886	0.2850
lightgbm	Light Gradient Boosting Machine	0.8419	0.9393	0.7993	0.8392	0.8165	0.6779	0.6809	0.2290
knn	K Neighbors Classifier	0.8269	0.8867	0.7843	0.8184	0.7973	0.6468	0.6509	0.0070
dt	Decision Tree Classifier	0.8168	0.8153	0.8033	0.7934	0.7960	0.6298	0.6328	0.0040
nb	Naive Bayes	0.7815	0.8594	0.5876	0.8923	0.6968	0.5407	0.5749	0.0040
qda	Quadratic Discriminant Analysis	0.7782	0.8836	0.5840	0.8867	0.6931	0.5340	0.5676	0.0040
lda	Linear Discriminant Analysis	0.7748	0.8580	0.5842	0.8697	0.6916	0.5275	0.5572	0.0040
ridge	Ridge Classifier	0.7731	0.8580	0.5805	0.8693	0.6892	0.5240	0.5540	0.0040
svm	SVM - Linear Kernel	0.6572	0.8559	0.7121	0.7062	0.6443	0.3300	0.3648	0.0040
dummy	Dummy Classifier	0.5547	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0030

CatBoost(Category Boosting) の概略

GBDT(勾配ブースティング木)

勾配降下法とブースティングと決定木を組み合わせた手法.

- 勾配降下法：目的関数の勾配を用いて更新を繰り返し，関数の最小値を見つけるアルゴリズム.
- ブースティング：アンサンブル学習の１つで，同じ種類のモデルを直列的に組み合わせる方法.
- 決定木：入力データを条件に従って枝分かれさせながら分類や予測を行うモデル.

- CatBoost はカテゴリ変数の扱いなどに特徴的な工夫をした GBDT のライブラリ.

Random Forest と Extra Trees の概略

Random Forest

決定木の集合により予測を行うモデル．決定木は並列に作成され，使用する特徴量はそれぞれランダムに選ばれる．

Extra Trees(Extremely Randomized Trees)

RF(Random Forest) とほぼ同じ方法でモデルを作成するが，分岐の際の閾値をランダムに設定する点で RF と異なる．

- いずれも 2 クラス分類問題に対しては，ジニ不純度が最も減少するように分岐を行う．

パラメータチューニング

- Pycaret の関数'tune_model'[2] を用いて，各モデルのチューニングを行なった．
- デフォルトは scikit-learn の RandomizedSearchCV(変更可)
  - ランダムサーチ：パラメータごとにランダムに選んだ複数の組み合わせを計算する方法．
  - グリッドサーチ：各パラメータに対して候補を定め，全ての組み合わせを計算する方法．

```
tuned_model_rf = tune_model(model_rf, optimize="auc",
                            n_iter = 100, choose_better = True)
```

ブレンディング[1]

- Pycaret の関数'blend_models'[2] を用いて，モデルのブレンディングを行った．
- 今回の場合，特に指定しなければ blend_models の予測結果は各モデルのクラス確率の平均となる．

```
blend_model = blend_models(
    estimator_list=[model_rf,model_et,model_cat],
    optimize="auc", choose_better=True)
```

[1]複数の予測モデルを組み合わせて、個々のモデルの予測結果を統合し、全体的な予測精度を向上させる手法

特徴量作成〜基準値を知る〜

表 2: 特徴量名とその基準値

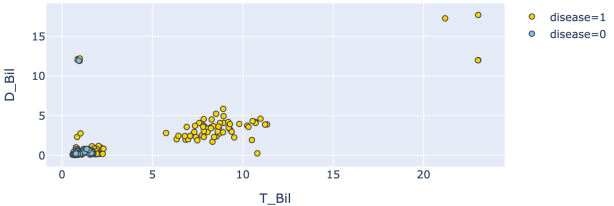
特徴量名	基準値
T_Bil(総ビリルビン)	0.40〜1.50(mg/dL)[4]
D_Bil(直接ビリルビン)	0.03〜0.40(mg/dL)[4]
ALT_GPT	〜30(U/L)[5]
AST_GOT	〜30(U/L)[5]
AG_ratio(アルブミン/グロブリン比)	1.32〜2.23[6]
TP(総蛋白)	6.5〜7.9(g/dL)[5]

特徴量作成〜特徴量間の関係を調べる〜

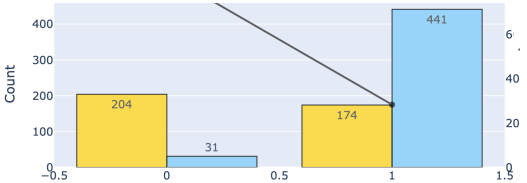
プロット図などを用いて，肝疾患の有無による分布の偏りを可視化した．

- **T_Bil と D_Bil の場合** (※ N1 は T_Bil が 1.5 以下かつ D_Bil が 0.4 以下の場合は 1，それ以外は 0 とする特徴量)

プロット図



特徴量'N1' の分布



特徴量生成

各特徴量の基準値や重要度[2]，特徴量間の相関などを考慮して，Kaggle の notebook[3] を参考に，以下の特徴量を作成した．

1. T_Bil が 1.5 以下かつ D_Bil が 0.4 以下の場合は 1，それ以外は 0 とする特徴量 N1
2. ALT_GPT が 30 以下かつ AST_GOT が 28 以下の場合は 1，それ以外は 0 とする特徴量 N2
3. AG_ratio が 0.6 以上 0.8 以下かつ TP が 6.5 以上 7.0 以下の場合は 1，それ以外は 0 とする特徴量 N3
4. 男性かつ AST_GOT が 30 以上の場合は 1，それ以外は 0 とする特徴量 N4

→ N1 と N4 を追加した際はスコアが向上した

[2] Future Importance: 各特徴量がモデルの予測にどれだけ影響を与えているかを示す指標．

その他試したが上手くいかなかったこと

1. 最も AUC が高い catboost において，分割１のスコアが他の分割と比べ低かったので，分割１をよく予測できている他のモデルとブレンディングした．
2. 肝疾患の年齢階級別総患者数の推移グラフ [7] を参考に特徴量'Age' に対しビニング[a] を行った．
3. 特徴量のうち，重要度が低いものや，相関が高いものを削除．

[a]数値データを一定の範囲で区切り，新たな区分に割り当てる手法．

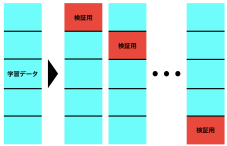
Fold	Accuracy	AUC
0	0.8833	0.9428
1	0.8000	0.8934
2	0.8833	0.9383
3	0.9000	0.9630
4	0.9167	0.9517
5	0.8644	0.9452
6	0.8475	0.9674
7	0.8814	0.9802
8	0.8814	0.9662
9	0.8475	0.9522
Mean	0.8705	0.9500
Std	0.0310	0.0225

その他試したが上手くいかなかったことに対する考察

1. 分割 1 をよく予測できている他のモデルとのブレンディング
   - 分割１のデータ特性 (外れ値など) への分析が不十分だったため，不適切なブレンディングによって過学習が起こった．
2. 'Age' に対するビニング
   - ビニングの範囲が広すぎて，情報が損失した
3. 特徴量の削除
   - 特徴量の重要性や冗長性を過小評価していた．

K-Fold CV[3] によるモデルの評価

1. データを何分割かにし，うち１つを検証用データとする．
2. 学習用データで訓練させたモデルを検証用データで評価し，指定したスコアを算出．
3. どれを検証用データにするかを変えて，それぞれで学習・評価を行う．
4. 各検証で求めたスコアを統合し，それによってモデルを評価する．



→今回は各検証において算出した AUC の平均を CV スコアとし，モデルを評価・比較した．

[3] Cross-Validation の略．分割したデータでモデルを複数回訓練・評価し、モデルが未知のデータに対してどの程度正確に予測できるか (汎化性能) を推定する手法．

K-Fold CV によるモデルの評価 (続き)

- CV スコアはモデルの汎化性能を評価する指標であり，過学習の有無を判断する際に有用．
- 過学習の兆候として，以下が挙げられる：
  - LB[4] スコアと CV スコアが連動しておらず，後者の方が高い．
  - 分割間の AUC のばらつきが大きい．

[4] テストデータの一部を用いたスコアに基づく順位表が Public Leaderboard として公開される．

提出したモデル

LB のスコアが最も高かったモデル 7 を提出．

- モデル 7 は，特徴量 N1 を追加し，Random Forest と Extra Trees をブレンディングしたモデル．

モデル(提出順)	CV	publicLB
1(cat)	0.9500	0.9219355
2(et)	0.9451	0.9282713
3(rf)	0.9428	0.9237386
4(et+rf+cat)	0.9503	0.9275434
5(et+cat)	0.9509	0.9295451
6(et+rf)	0.9476	0.9334988
7(et+rf, N1)	0.9498	0.9345906
8(et+rf, N4)	0.9500	0.9327709
9(et+rf+cat, N1)	0.9500	0.9331348
10(et+rf+cat, N4)	0.9490	0.9317122
11(et+rf+cat, N1, Age)	0.9449	0.9262862
12(et+rf+cat, N1, N4)	0.9472	0.9262531

目次

結果

- 最終順位：2 位/214 人投稿.
- 下の画像の通り，暫定評価と最終評価が同じ→コンペ中の LB のスコアは全てのデータを用いて算出されていた.

順位	ユーザ名	暫定評価	最終評価 ▼	投稿件数	投稿日時
1	[redacted]	0.9348883	0.9348883	150	2024-09-27 00:54:04
2	中三川	0.9345906	0.9345906	51	2024-09-28 23:07:35

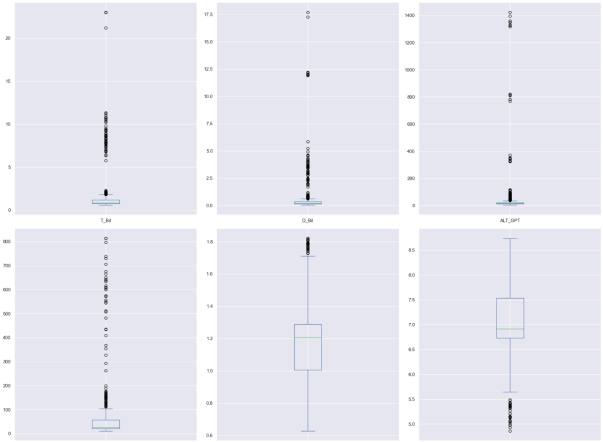
反省点

- 提出時の初歩的な誤り：
  - 提出方法の確認不足から，不要な特徴量名を含めていた．
  - 評価指標への理解不足により，確率ではなく 0 か 1 の予測をしていた．
- パラメータ調整の不足：
  - 分割数など各パラメータへの理解を深め，その設定をさらに最適化する余地があった．
- 過学習の懸念：
  - モデル評価時，LB との連動のみ重視し，CV スコアへの解釈が不十分だった．
  - 採用した新たな特徴量は，知識不足のまま曖昧な基準値を参考に，またプロット図を基に作成しているため，過学習を助長した可能性がある．
- 外れ値への対応不足：
  - 外れ値の影響を全く考慮しなかったため，モデル作成および評価時に悪影響を及ぼした可能性がある．

各特徴量の基準値と箱ひげ図

表 2: 特徴量名とその基準値 (再喝)

特徴量名	基準値
T_Bil	0.40〜1.50(mg/dL)[4]
D_Bil	0.03〜0.40(mg/dL)[4]
ALT_GPT	〜30(U/L)[5]
AST_GOT	〜30(U/L)[5]
AG_ratio	1.32〜2.23[6]
TP	6.5〜7.9(g/dL)[5]



今後に向けて

- CV と LB の連動性：
  - コンペ選びの際は，CV と LB のスコアが連動しているものを優先する．
- データ分析に関する知識の深化：
  - 相関分析[5] への適切な理解を深め，散布図などを用いて相関関係を考察する．
  - 外れ値の特定と適切な処理方法を学び，モデルの安定性向上に繋げる．

→論文や Kaggle の大きいコンペの notebook などを活用する．

- モデル理解に向けた説明可能 AI の導入：
  - 説明可能な AI の学習を通して，予測の根拠やモデル性能向上の理由の理解に繋げる．

[5]特徴量間の関係を定量的に調べるための手法．

目次

GBDT のアルゴリズム [9]

データセット：$\{\boldsymbol{x}_i, y_i\}_1^N$, 損失関数：$L(y_i, \rho)$, 決定木の出力：$h$

1. **初期値設定：** $F_0(\boldsymbol{x}) = \arg\min_{\rho} \sum_{i=1}^{N} L(y_i, \rho)$
2. $m = 1 \ldots, M$ **に対し，以下を繰り返す (**M **：決定木の数)：**
   1. **損失関数の勾配を計算：**
   $$\tilde{y}_i = -\left[\frac{\partial L(y_i, F(\boldsymbol{x}_i))}{\partial F(\boldsymbol{x}_i)}\right]_{F(\boldsymbol{x})=F_{m-1}(\boldsymbol{x})}, \quad i = 1, \ldots, N$$
   2. **勾配と決定木の二乗和誤差を最小化する：**
   $$\boldsymbol{a}_m = \arg\min_{\boldsymbol{a},\beta} \sum_{i=1}^{N} [\tilde{y}_i - \beta h(\boldsymbol{x}_i; \boldsymbol{a})]^2, \quad (\beta : \textbf{スケーリングパラメータ})$$
   3. **損失関数を最小化する：** $\rho_m = \arg\min_{\rho} \sum_{i=1}^{N} L(y_i, F_{m-1}(\boldsymbol{x}_i) + \rho h(\boldsymbol{x}_i; \boldsymbol{a}_m))$
   4. **モデルを更新：** $F_m(\boldsymbol{x}) = F_{m-1}(\boldsymbol{x}) + \rho_m h(\boldsymbol{x}; \boldsymbol{a}_m))$
3. **最終的なモデル** $F_M(\boldsymbol{x})$ **を出力．**

CatBoost の工夫 (一部)

- GBDT の問題点：以下の手順でターゲットリーケージ[6] が発生し，過学習しやすくなる．
  1. カテゴリーの種類が多数の変数に対し，各カテゴリを y の統計量 (Target Statistics: TS) に基づいてエンコーディングする．
  2. 損失関数の勾配を計算する際，全ての目的変数 y_i の情報を用いて計算する．
- これらを改善するため，CatBoost では以下の工夫を施している [10]：
  1. TS の代わりに，データの分割を工夫するなどの改良を行った，Greedy TS，Holdoout TS, Leave-one-out TS,Orderd TS を使用．
  2. 一般的な勾配法の代わりに，Ordered Boosting という勾配法を使用．

[6] モデルの学習過程で，モデルが予測しようとしている目的変数に関する情報が，予測に不適切に含まれてしまう現象．

参考文献 I

[1] 門脇大輔, 阪田隆司, 保坂桂佑, 平松雄司. (2019). Kaggle で勝つデータ分析の技術. 技術評論社.

[2] GitHub. An open-source, low-code machine learning library in Python.
https://github.com/pycaret/pycaret/blob/master/pycaret/classification/functional.py.
更新日 2024 年 7 月 25 日. 閲覧日 2024 年 12 月 25 日.

[3] Kaggle. Indians Diabetes - EDA & Prediction (0.906).
https://www.kaggle.com/code/vincentlugat/pima-indians-diabetes-eda-prediction-0-906.
更新日 2020 年 1 月 27 日. 閲覧日 2024 年 12 月 25 日.

[4] 帝京大学医学部附属病院. 検査値の見方.
https://www.teikyo-hospital.jp/hospital/section/inspection/pdf/inspection03.pdf.
更新日 2016 年 2 月. 閲覧日 2024 年 12 月 26 日.

[5] 東京メディカルクリニック　日本人間ドック・健診センター. 肝機能の検査.
https://www.dock-tokyo.jp/results/liver-function/ast-alt.html.
更新日 2024 年 4 月. 閲覧日 2024 年 12 月 26 日.

[6] 人間ドックのマーソ. A/G 比 - 血液検査でわかること.
https://www.mrso.jp/inspection/209.html. 更新日 2016 年 2 月. 閲覧日 2024 年 12 月 26 日.

参考文献 II

[7] メットライフ生命. 肝疾患とは？肝疾患の平均入院日数と入院治療費用の例.
https://www.metlife.co.jp/products/medical/contents/liver-disease/.
更新日 2024 年 8 月 29 日. 閲覧日 2024 年 12 月 26 日.

[8] SIGNATE. 【練習問題】健診データによる肝疾患判定.
https://signate.jp/competitions/265/data

[9] Jerome H. Friedman. (1999). Greedy function approximation: A gradient boosting machine.
https://projecteuclid.org/journals/annals-of-statistics/volume-29/issue-5/Greedy-function-approximation-A-gradient-boosting-machine/10.1214/aos/1013203451.full

[10] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, Andrey Gulin. (2017).
CatBoost: unbiased boosting with categorical features.
https://arxiv.org/abs/1706.09516