

Otto Group Product Classification Challenge

https://www.kaggle.com/c/otto-group-product-classification-challenge

Otto 2015 再挑戦

生成AIと表形式の基盤モデルで
Grokの基準を更新

最終本命 private 0.36532

Executive Summary

0.36532

最終本命 private logloss

Grok基準 0.37196から 0.00664改善

Grok：TabPFNの予測資産と固定fold

Claude：共通の調査プロンプト

Gemini・ChatGPT・Claude.ai：調査

Claude Fable 5.2：統合戦略

Codex：実験・検証・提出



privateを見て選ばず、事前の評価・停止条件に従って本命を固定

Otto 2015の課題とデータ

1行が1商品。匿名93特徴から、9クラスの確率を予測する

学習61,878行（正解あり）　／　test 144,368行（正解非公開）



train.csv の先頭5行（参考）

id	f1	f2	f3	f4	f5	…	f91	f92	f93	target
1	1	0	0	0	0	…	0	0	0	Class_1
2	0	0	0	0	0	…	0	0	0	Class_1
3	0	0	0	0	0	…	0	0	0	Class_1
4	1	0	0	1	6	…	0	0	0	Class_1
5	0	0	0	0	0	…	0	0	0	Class_1

元CSVの実データ。f1等はfeat_1等の略記。feat_6〜feat_90は中間列を省略。

loglossは、確率の出し方を評価する

正解クラスに付けた確率 p の損失は −log(p)

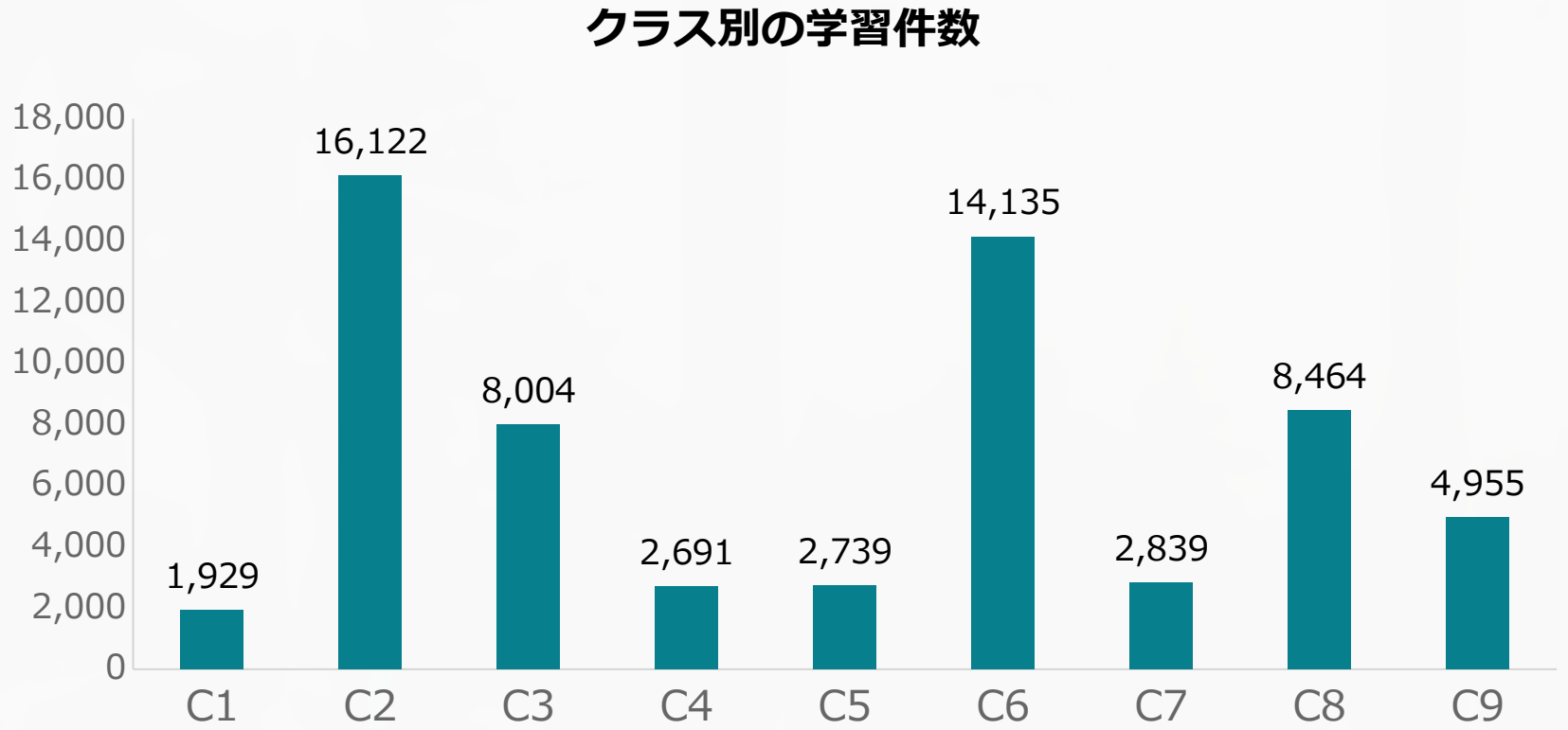
正解クラスの確率	1商品の損失
0.9	0.105
0.5	0.693
0.01	4.605

小さいほど良い。強い確信で間違えると、大きな損失になる

表は式を説明する架空の計算例。全商品の損失を平均して評価する



クラスの件数には約8.36倍の差がある





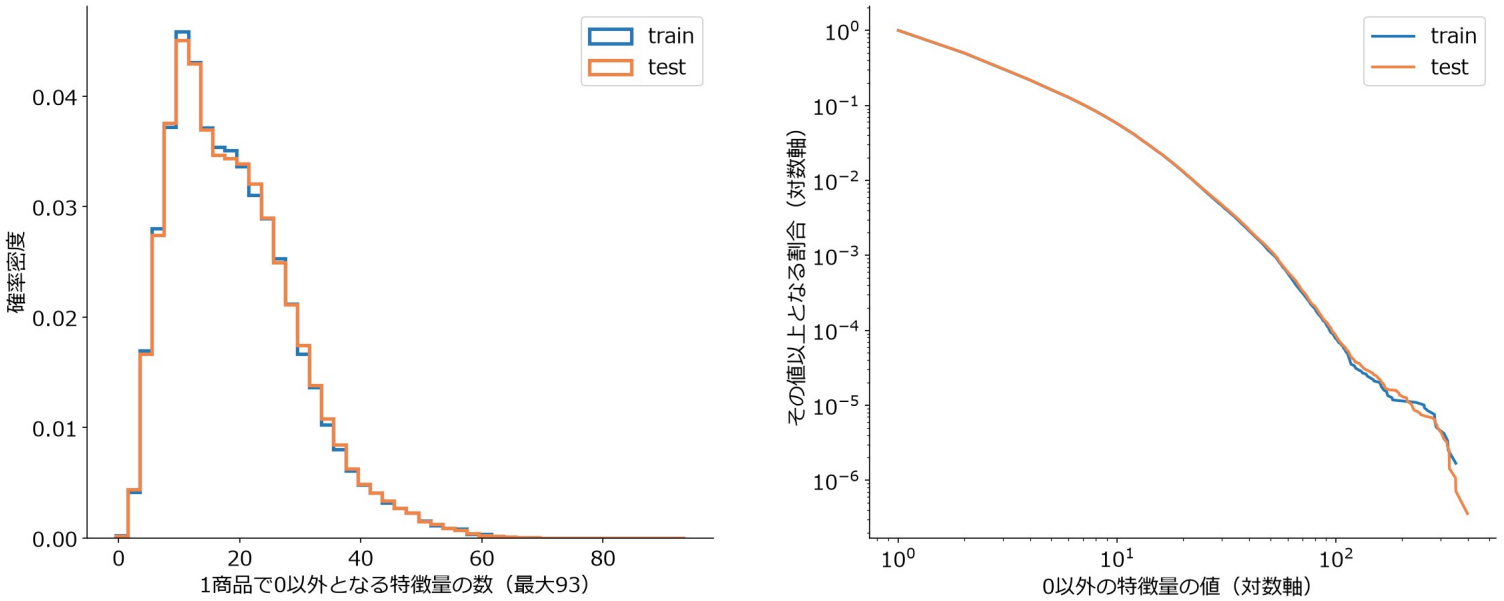
最多 Class_2：16,122件　　最少 Class_1：1,929件

train.csv 全61,878行を集計。クラス構成を保つ層化CVを使用

約8割のセルが0、一部の値は大きい

0の割合 79.34%　欠損 0　非ゼロ列数の中央値 18／93

多くのセルは0、一部の値は大きい



0は欠損値として扱わない。右図は正のセルだけの分布。大きな値の裾が長い。

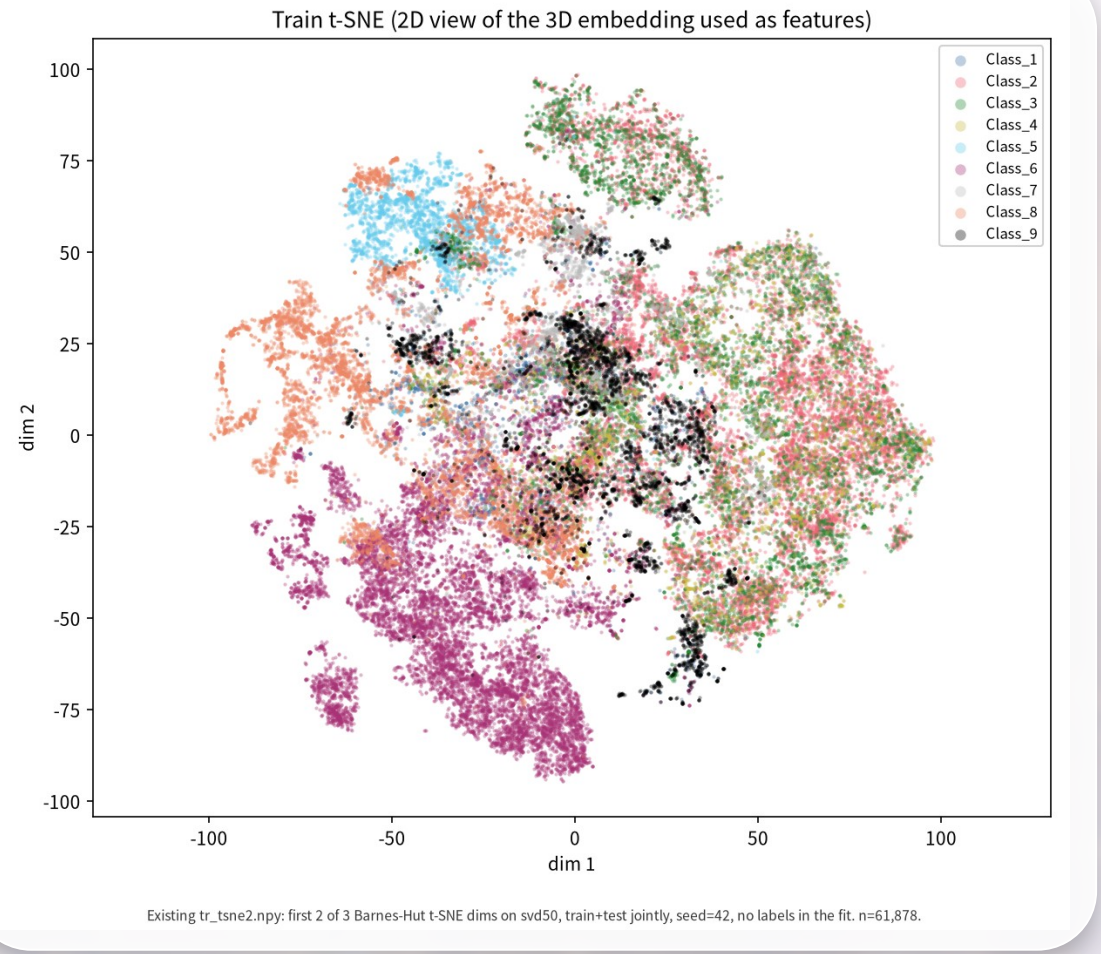
train/testの全体的な形は似るが、これだけでは分布差や将来の性能は判断できない。

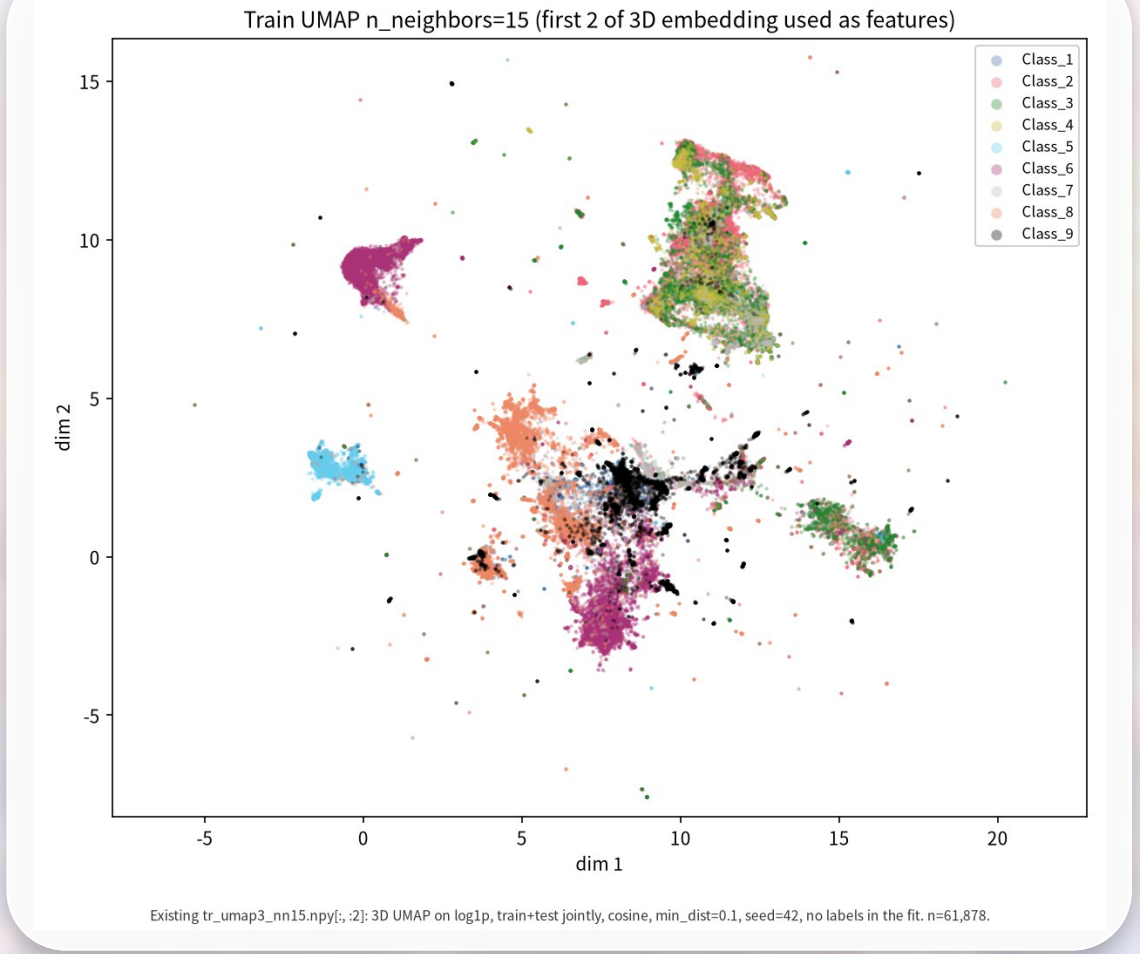
大きなカウントの裾が長い。0は欠損として補完しない。



t-SNE・UMAPで見る商品の位置関係







学習61,878点、3次元の先頭2軸。Class 5・6は比較的分かれ、2・3・4は混在（目視）

なぜ、t-SNE・UMAPの座標を特徴量に加えると効くのか

**93列の組合せにある「近い商品同士」の関係を、
少数の座標へまとめてLightGBMに渡せる。**

木は1列ずつしきい値で分割する。複数列にまたがる近傍関係は、
埋め込み座標を足すことで扱いやすくなる可能性がある。

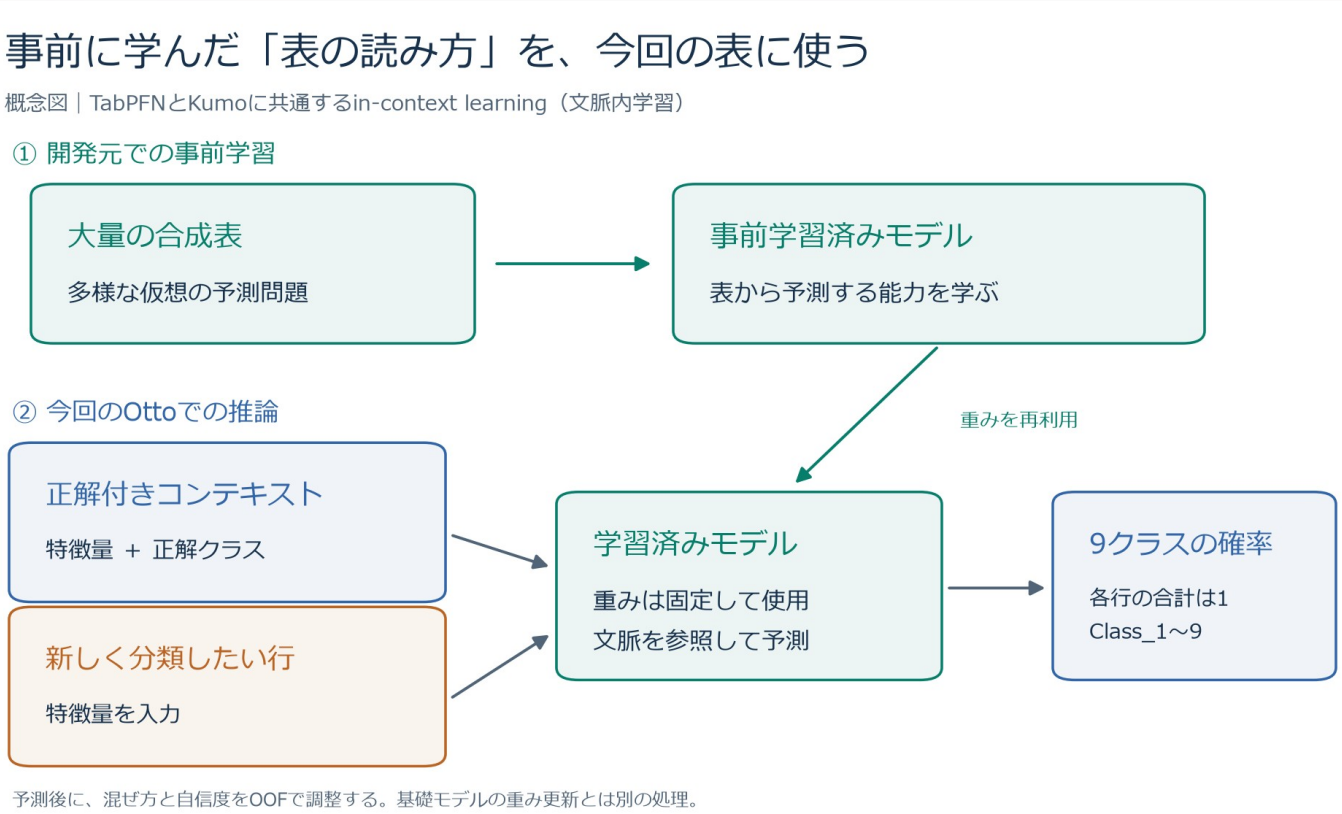
同条件の入力	OOF logloss	追加の効果
土台A	0.469951	—
A＋t-SNE3	0.465469	−0.004481
A＋UMAP2種	0.456226	−0.013725
raw93特徴	0.453295	—
raw93＋UMAP2種	0.438755	−0.014541

土台A：log1p＋統計量＋SVD50（154列）

改善は実測（各5/5 fold）。近傍構造が効いたという機序は仮説。最終TabPFN/Kumoには未使用。



TabPFN・Kumoは、表の読み方を事前に学ぶ



事前に、多くの仮想的な表の予測問題を経験したモデル。

Ottoでは正解付きの行を文脈として参照し、新しい行の9クラスの確率を計算する。

今回、基礎モデルの重みは固定。混合重みと温度は、予測後にOOFで調整した。



開発元での事前学習、Ottoでの文脈内推論、予測後の校正を分けて理解する。

TabPFNが強い基準を作った

項目	GrokがつくったTabPFN基準
モデル	TabPFN-3.5
入力	raw93特徴
予測のまとめ方	seed 0・1・2の幾何平均＋温度
private logloss	0.37196

Codexは、この予測と固定foldを引き継いだ

基盤モデルの重みをOttoで追加学習した実験ではない



Kumoの導入が大きな改善につながった

Kumoは、セルから表の情報を組み立てる

概念図｜NVIDIAの公式3段階説明を簡略化。実際の層数・テンソル形状は省略。

1　数値を表現する

表の1マスの値を
モデル内部の
数値の並びへ変換

→

2　列と行を読む

列：値の分布を扱う
行：特徴を組み合わせる
各行の要約を作る

→

3　文脈で予測する

正解付き行を参照
新しい行について
9クラスの確率を出す

Transformerのattention：予測に関係する情報を集め、組み合わせる計算。

TabPFN-3.5も、値・列の分布・特徴の組合せを扱うTransformer型のICLモデル。

出典：NVIDIA NIM API Reference / SDM KumoTabular API。図は本レポートで作成。

Large v1.0.1、8推定器、1seed

単体＋温度 private 0.36642。1 fold 約4分、ピーク割当 約3.9GB。

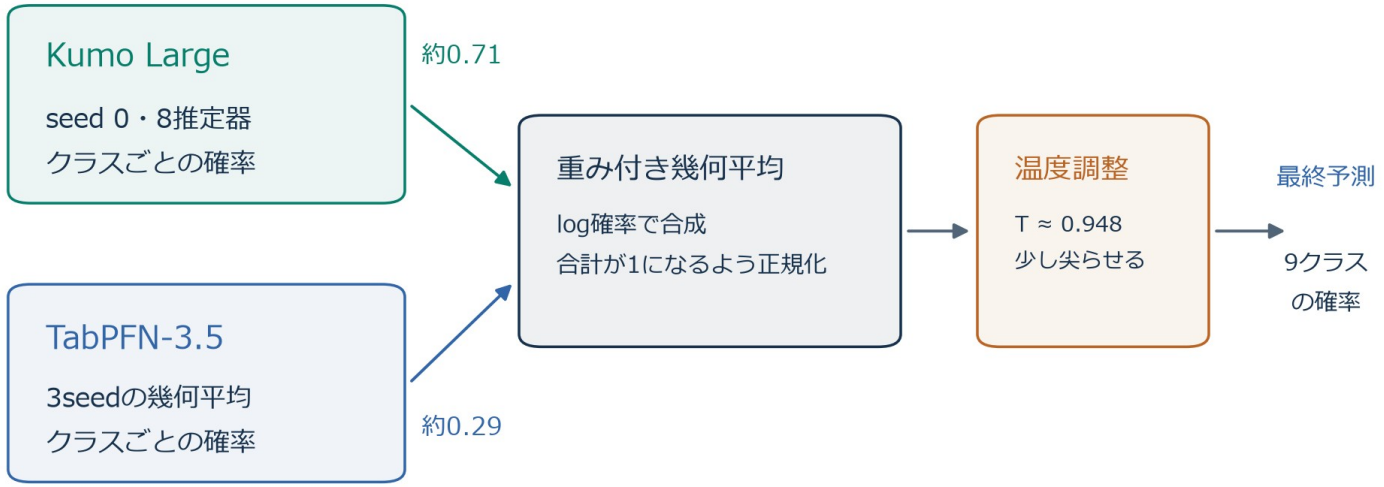


温度は、予測確率の自信度を調整する



確率を混ぜ、自信度を調整する

今回の最終本命｜係数・温度はOOFで決定。privateスコアで選ばない。



最終本命のprivate log loss：0.36532

0.71／0.29は、正解率や担当する行数ではなく、予測確率を混ぜる係数。

比較候補の最高値0.36516とは区別する。実測条件と選択理由は本文を参照。

T＝1：元の確率のまま
T＜1：確率を鋭くする
T＞1：確率をなだらかにする

全クラスに共通の温度なら、
最大確率のクラスは変わらない。
確率の自信度を整える。

温度T	3クラスの確率（架空例）
1	0.700 / 0.200 / 0.100
0.8	0.771 / 0.161 / 0.068
1.2	0.645 / 0.227 / 0.128

今回の温度 T≈0.948は少し鋭くする方向。OOFで決定し、privateでは選んでいない。

検証行を、予測の参照例から外す

OOFのfold	コンテキスト行	検証行	重複
0〜2（各fold）	49,502	12,376	0
3〜4（各fold）	49,503	12,375	0

重み・温度は固定OOF上の校正クロスフィットで評価

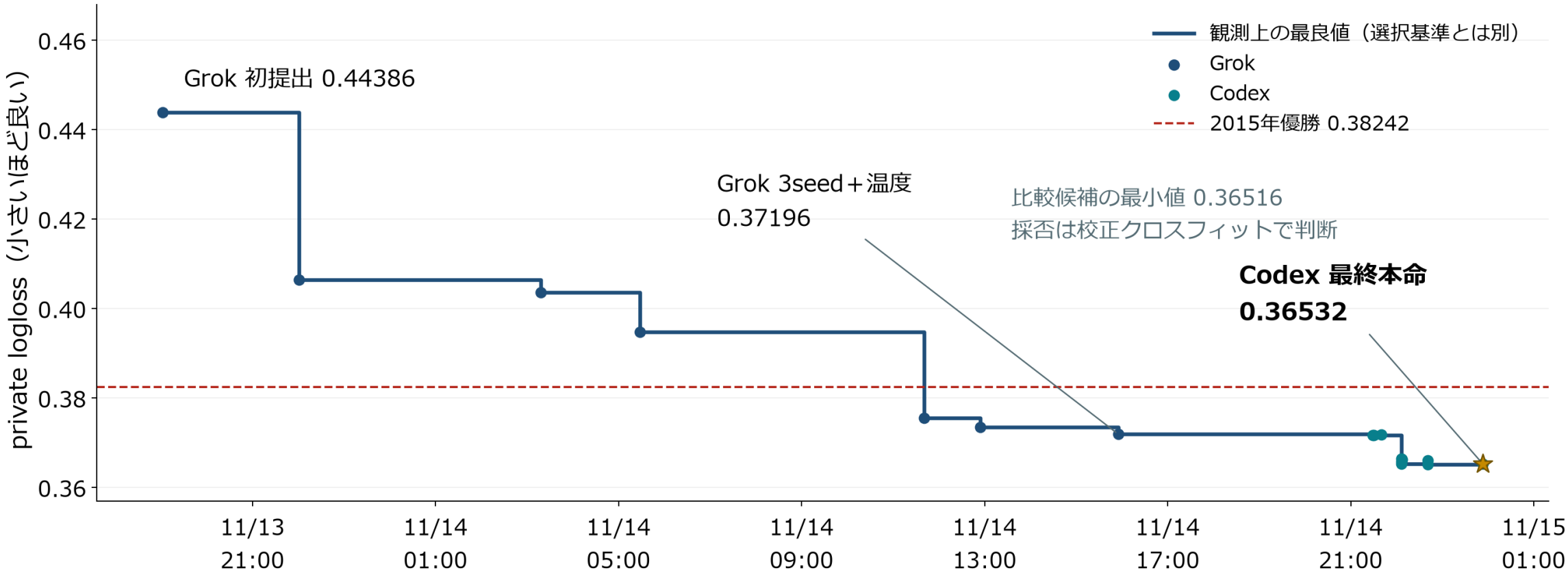
testだけ全61,878行をコンテキストに使用

基礎モデルまで再学習する完全な入れ子CVとは区別する



Grokの探索からCodexの更新へ

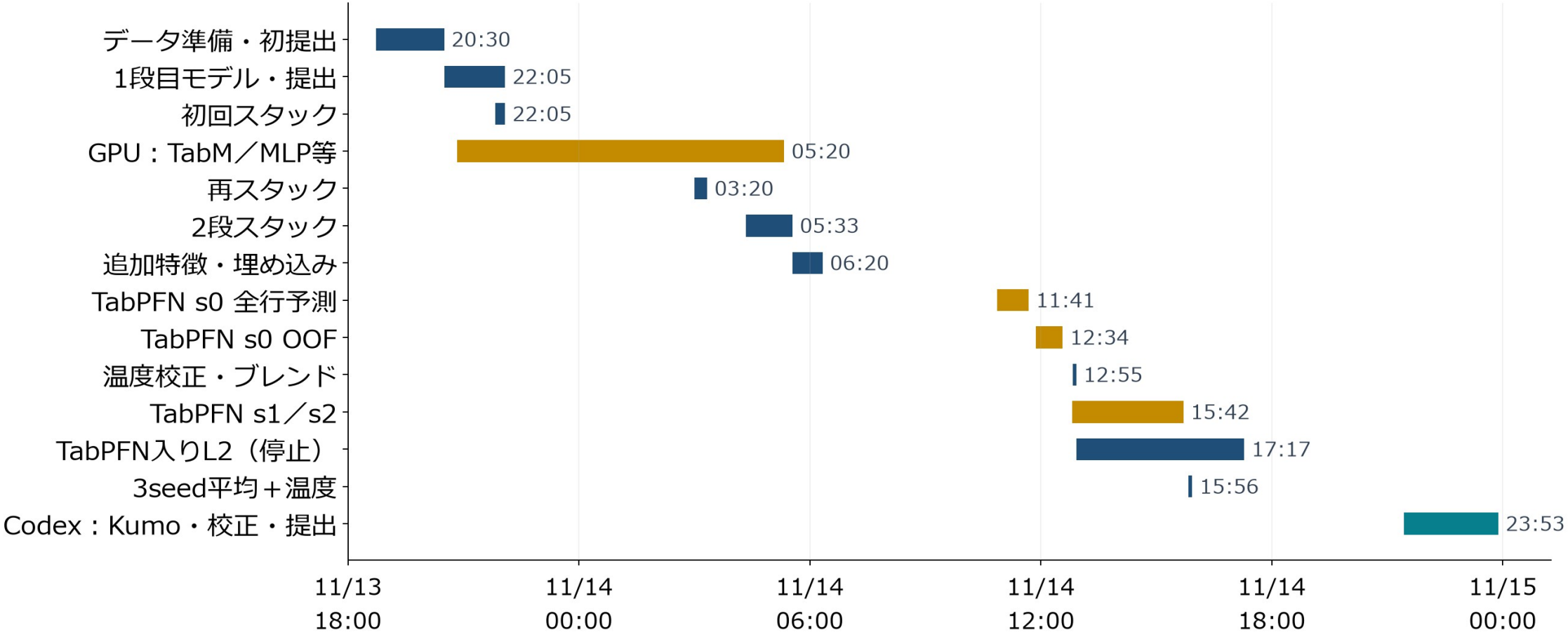




時刻はJST。Grok主要7点とCodex全11提出。本命0.36532／比較候補0.36516。

Grokの工程を引き継いだ、Codexの追加実験





時刻はJST。Grok全体22時間33分／Codex追加2時間27分8秒（既存予測・foldを継承）。

今回の最終スコアの位置づけ

位置づけ	private logloss	本命 – 参考値（負ほど改善）
2015年優勝（公式LB）	0.38242	−0.01710
TabPFN-3.5 公開ノート	0.37549	−0.01017
Grok：3seed＋温度	0.37196	−0.00664
Kumo 外部報告（参考）	0.36924	−0.00392
今回の最終本命	0.36532	—

複数AIで資産・調査・戦略・実行を引き継ぎ、基準を更新。

Claude Fable 5.2：対象Proの特典を活用（2026/11/9〜11/16 PT）

終了は日本時間11/17 16:59。対象契約・週間上限の条件あり。

異なる年代・設定の参考比較。0.36924は第三者報告で、NVIDIA公式値とは表記しない。



補足：モデル・混合・校正の区別

処理	意味
基礎モデル	TabPFN・Kumoがクラス確率を計算
混合	2つのモデルの対数確率を組み合わせる
校正	温度で予測の自信度を整える

$$p = \mathrm{softmax}((wK \log pK + wT \log pT) / T)$$

wK ≈ 0.7138　wT ≈ 0.2862　T ≈ 0.9475

同じOOFで調整・評価した値と、校正クロスフィットの値は区別



補足：学習データと今回の検証

Kumoの合成データのみの事前学習はNVIDIAの公表内容

根拠はNIM APIリファレンス

今回のOOF分離はコード・行番号・予測記録を照合

最終CSVは再構成し、SHA256一致を確認



メーカー公表情報と、今回のローカル実装監査を別の根拠として示す