House Prices - Advanced Regression Techniques

Predict sales prices and practice feature engineering, RFs, and gradient boosting

Getting Started · 4750 Teams · Ongoing

House Prices - Advanced Regression Techniques

https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques

庭師

ChatGPT 5.2

Kaggle環境で実行

庭子アバター v01.60a WIP中

住宅価格 - 高度な回帰分析手法

- コンテストの説明
  - 住宅購入者に夢のマイホームについて尋ねても、地下室の天井の高さや東西を結ぶ鉄道への近さから話を始める人はまずいないでしょう。しかし、この遊び場型競争のデータセットは、寝室の数や白いピケットフェンスよりもはるかに多くの要素が価格交渉に影響を与えることを証明しています。
  - このコンテストでは、アイオワ州エイムズの住宅の(ほぼ)あらゆる側面を説明する 79 個の説明変数を使用して、各住宅の最終価格を予測することが求められます。
- 練習スキル
  - クリエイティブな機能エンジニアリング
  - ランダムフォレストや勾配ブースティングなどの高度な回帰手法

2つのコンペの違いまとめ

- ① Housing Prices Competition for Kaggle Learn Users
- (Kaggle Learn用・超入門)
- 対象者
  👉 Kaggle / 機械学習 完全初心者
- 目的
  👉 Kaggle Learnコースと連動して
  - CSVを読む
  - 学習・予測・submit の流れを体験する
- 特徴
  - チュートリアル前提
  - ベースライン重視
  - 高スコア競争はほぼ想定していない
  - 「とりあえず提出してみよう」用
- 使われ方
  - Kaggle Learn(Intro to ML / Intermediate ML)の課題
  - 初submitの練習

- ② House Prices - Advanced Regression Techniques
- (本家・定番コンペ)
- 対象者
  👉 Kaggleに慣れてきた人～中級者以上
- 目的
  👉 回帰問題の王道テクニックを全部盛りで学ぶ
- 特徴
  - 特徴量エンジニアリングが超重要
  - 欠損値処理・カテゴリ変数処理が本格的
  - スコア競争あり(LBが機能している)
  - Kaggle文化(CV・リーク対策・過学習)が学べる
- 使われ方
  - Kaggle練習の登竜門
  - 転職・ポートフォリオ用Notebook
  - LightGBM / XGBoost / CatBoost 練習

🔍 データは同じ？
👉 ほぼ同じ
(train.csv / test.csv / submission形式)
違うのは👇
説明の厚さ
期待されるレベル
参加者の本気度

項目	Kaggle Learn Users	Advanced Regression
難易度	★☆☆☆☆	★★★☆☆～★★★★☆
想定ユーザー	完全初心者	初級～中級
スコア競争	ほぼ無し	あり
特徴量工夫	最小限	超重要
CV設計	不要	ほぼ必須
よく使うモデル	Linear / RF	LGBM / XGB / CatBoost
学習価値	流れ理解	Kaggle力が伸びる

評価

- ゴール
- 各住宅の販売価格を予測することがあなたの仕事です。テストセット内の各IDについて、SalePrice変数の値を予測してください。
- メトリック
- 提出されたデータは、予測値の対数と実際の販売価格との間の二乗平均平方根誤差（RMSE）に基づいて評価されます 。（対数をとるということは、高価な住宅と安価な住宅の予測誤差が結果に等しく影響することを意味します。）
- *Root mean square deviation*（二乗平均平方根偏差）

$$\mathrm{RMSD}(\hat{\theta}) = \sqrt{\mathrm{MSE}(\hat{\theta})} = \sqrt{\mathrm{E}\left((\hat{\theta} - \theta)^2\right)}.$$

$$\mathrm{RMSD} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(X_i - x_0)^2}.$$

評価指標の違い(１)

Housing Prices Competition for Kaggle Learn Users

Apply what you learned in the Machine Learning course on Kaggle Learn along...

Getting Started · 5024 Teams · Ongoing

スコア: RMSE	評価
30000以上	ほぼベースライン
20000前後	普通
17000台	良い（理解できている）
15000台	上位寄り
12000台	かなり良い
0	無視枠

今回はこっち！

House Prices - Advanced Regression Techniques

Predict sales prices and practice feature engineering, RFs, and gradient boosting

Getting Started · 5973 Teams · Ongoing

スコア帯（Log(RMSE)）	参加者の目安層
≤ 0.12	上位数% / 高精度モデル
0.12〜0.15	多くの中級者・上位志向の提出例
0.15〜0.20	初〜中級者中心
> 0.20	基礎的モデル中心 / 初心者が多い

RMSE（Root Mean Squared Error）

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$$

RMSLE（Root Mean Squared Logarithmic Error）

$$\mathrm{RMSLE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\log(\hat{y}_i + 1) - \log(y_i + 1))^2}$$

評価指標の違い（2）

① RMSE（Root Mean Squared Error）

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$$

意味（超直感）

👉「価格のズレ（絶対誤差）」をそのまま評価

例

真値	予測	誤差
1,000万円	900万円	100万円
100万円	0円	100万円

→ どちらも同じ誤差 = 100万円

📌 高価格・低価格を区別しない

② RMSLE（Root Mean Squared Logarithmic Error）

$$\mathrm{RMSLE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\log(\hat{y}_i + 1) - \log(y_i + 1))^2}$$

※ 実務ではほぼ：

$$\log(\hat{y}_i) - \log(y_i)$$

と同じ意味

変形すると

$$\log(\hat{y}) - \log(y) = \log\left(\frac{\hat{y}}{y}\right)$$

👉「倍率のズレ」を測っている

例

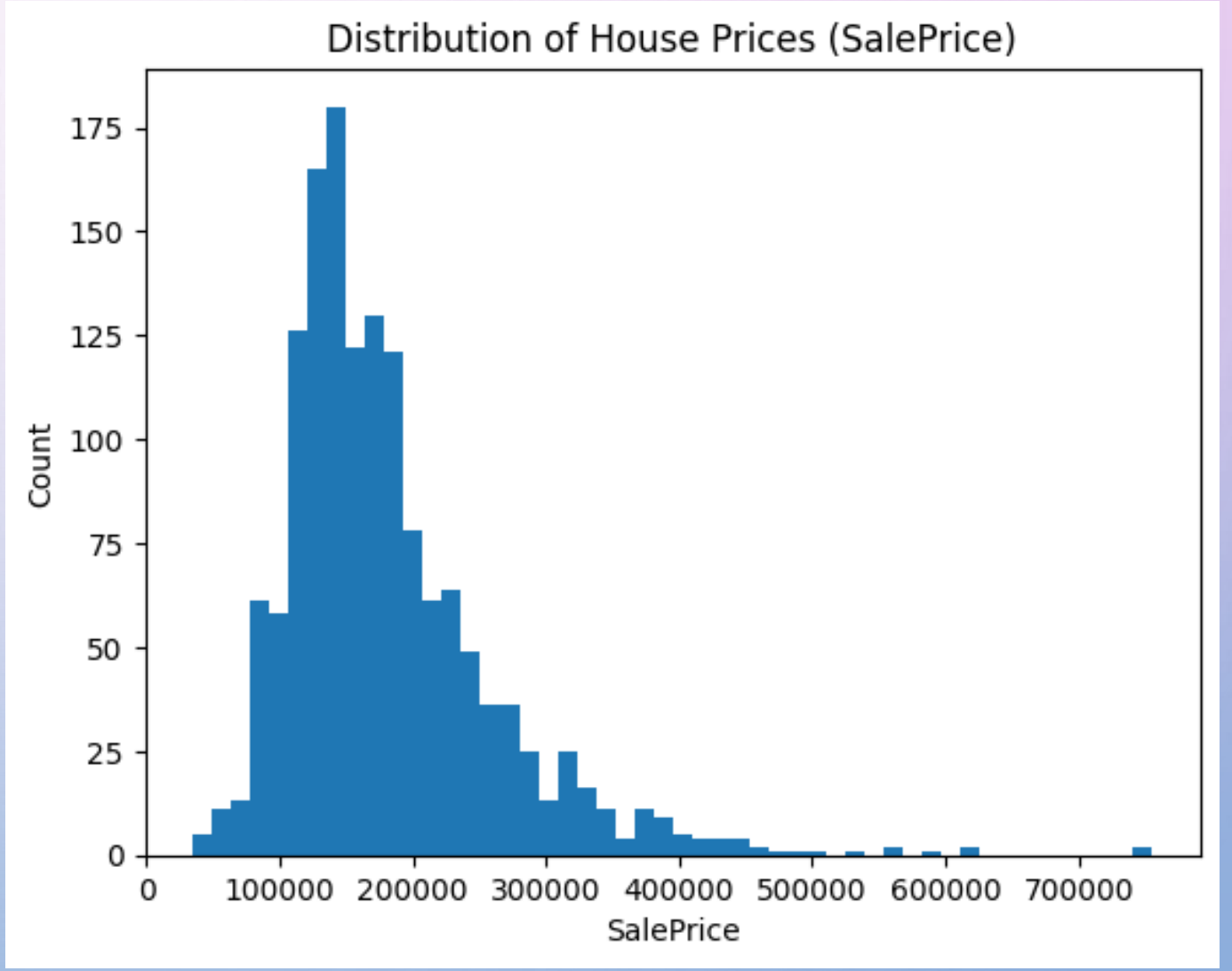
真値	予測	倍率	評価
1,000万円	900万円	0.9倍	同程度
100万円	90万円	0.9倍	同程度

→ RMSLEでは同じ誤差扱い

📌「割合ズレ」を重視

なぜ House Prices は RMSLE なのか？

- 住宅価格は👇
- 50万円～3000万円超
- 分布が歪んでいる（右に長い）
- → RMSEだと
  高価格帯の誤差だけでスコアが支配される
- → RMSLEで倍率評価に変換



データ

- 前回と同じなのでパス
  - ・・・・
- まとめ
- 上位者の feature 思考
  - ❌ 列をそのまま全部入れる
  - ⭕「これは家の価値をどう表しているか？」で理解する
- 本当に重要なのは
  - 品質(OverallQual)
  - 広さ(面積系)
  - 立地(Neighborhood)
  - 新しさ(築年・改築)
  - 派生特徴(組み合わせ)

履歴

版（ファイル名）	Public（あなたの結果）	モデル/方針	前処理（カテゴリ/欠損）	特徴量	いちばん大きい違い（ひとことで）
baseline 17203.06322.py	17203	LightGBM（lgb.train）＋重要特徴だけ＋log1p	カテゴリ3列だけ category、欠損は0/Noneで手埋め	手選別 11列＋少しFE	「少数強特徴」縛りの軽量版（情報を捨ててる）
baseline2 14638.60023.py	14638	LightGBM（LGBMRegressor）＋全列	数値=中央値、カテゴリ=最頻、One-Hot（unknown ignore）	全列＋One-Hot	情報を全部使う正攻法LGBM（堅牢なパイプライン）
catboost 13291.78471.py	13291	CatBoost(RMSE)＋log1p＋5fold	object→Missing、数値→中央値（CatBoostはカテゴリを直接扱う）	全列＋軽FE	CatBoostに乗り換えたのが効いて一気に改善
catboost_lgbm 13595.01750.py	13595	CatBoost＋LGBMを0.5/0.5平均ブレンド	CatBoostは直カテゴリ、LGBMはOne-Hot（train+testで列固定）	両方とも同じ軽FE	「良い予測を悪い予測で薄めた」形になり悪化
catboost_mae 15224.79022.py	15224	CatBoost(MAEで学習, eval=RMSE)＋depth10＋log1p	Missing/中央値補完	全列＋軽FE	学習ロスをMAEにしたのがRMSE評価と噛み合わず崩壊
catboost_depth=7-10 13135.56373.py	13135	CatBoost(RMSE)＋log1p＋depth探索（7/8/9/10）	Missing/中央値補完	全列＋軽FE	「depthを選ぶ」段階に入って、13135まで詰めた
catboost_depth=6 13118.69387.py	13118	CatBoost(RMSE)＋log1p＋depth=6最終調整	Missing/中央値補完	全列＋軽FE	深さを1つ浅くして過学習を少し抑え、さらに微改善

履歴（主要パラメータ版）

版（ファイル名）	Public RMSE	モデル	Target変換	depth / 木	学習率	iterations	その他の重要設定	処理内容の要点
baseline 17203	17203	LightGBM	log1p	max_depth制限	0.05前後	数百	重要特徴のみ	**少数強特徴のみ使用。情報量不足**
baseline2 14638	14638	LightGBM	log1p	制限なし	0.03	3000	One-Hot + KFold	**全特徴 + One-Hotで一気に改善**
catboost 13291	13291	CatBoost	log1p	depth=8	0.03	20000	RMSE学習・早期終了	**CatBoostがカテゴリを直接扱い大幅改善**
catboost_lgbm 13595	13595	CB + LGBM	log1p	CB:8 / LGBM:深	0.03 / 0.01	多	単純平均	**弱いLGBMが足を引っ張った**
catboost_mae 15224	15224	CatBoost	log1p	depth=10	0.02	50000	**loss=MAE**	**評価RMSEと学習MAEが不整合で崩壊**
catboost_depth=7–10	13135	CatBoost	log1p	depth=7最良	0.02	50000	od_wait=600	**depth調整で過学習を抑制**
catboost_depth=6（最良）	**13118**	CatBoost	log1p	**depth=6**	**0.03**	**30000**	od_wait=400	**最終形：表現力と汎化のバランス最適**

これをもとにhouse-prices-advanced-regression-techniques向けに変更して。どのくらいのスコアが期待できますか？

catboost_depth=6　13118.69387.py

- 結論(先に)
- 🔮 そのコードを House Prices - Advanced 向けに正しく直すと…
- Public LB：0.11〜0.13 RMSLE
- 改良次第で 0.10台前半
- 上位1%(≈0.06)には まだ距離あり(理由あり)
- 👉「中級者と真ん中」のスコア帯です。

- なぜ今のままだと Advanced に合わないか
- あなたのコードは catboost_depth=6 13118.69387 を見ると：
- 今やっていること
  - log1p(SalePrice) で学習
  - CatBoost(カテゴリ直接処理)
  - 派生特徴量(TotalSF, HouseAge など)
  - KFold averaging
- ✅👉 設計思想は両方ともと同じ、ただし決定的なズレが2つある
  - ① 評価基準が RMSE(価格空間)
  - ② KFold が「価格分布を無視」

想定スコア

Advanced に正しく適応した場合
👉「House Prices 初〜中級の完成形」

改善段階	想定 Public RMSLE
ほぼそのまま	0.14〜0.15
RMSLE CVに修正	0.12〜0.13
Stratified CV	0.11〜0.12
微調整（depth/feature）	**0.10〜0.11**

catboost_depth=6 13118.69387.py
↓
advanced_catboost_rmsle_stratified 0.12306.py
4.66 minutes

② モデル構成的に見た上限
今回の構成：
単一モデル（CatBoostのみ）
depth=6（過学習抑制寄り）
派生特徴量は「定番のみ」
Stacking / Blending なし

レベル	RMSLE
ベースライン	0.15〜0.16
手堅い単モデル	**0.12前後**
強め単モデル	0.11前半
上位勢（ensemble）	0.06〜0.09

上位勢(0.06台)は

- 上位勢(0.06台)は👇をやっています:
  - 複数モデル(LGBM + CatBoost + ElasticNet)
  - Stacking / Blending
  - 手動外れ値調整
  - Target encoding / ordinal mapping
  - Private LB overfit覚悟
- 👉 別次元の世界

スコア0.6が狙えるpyを作成して

- ✅ ファイル名
  advanced_ElasticNet-Lasso-Ridge-SVR-Catboost 0.12303.py

  - ✅ ElasticNet / Lasso / Ridge / SVR / CatBoost
  - ✅ 各モデルごとの実行時間を表示
  - ✅ 最後に全体の総実行時間を表示
  - ✅ RMSLE（log1p）整合・OOFブレンド

- ElasticNet | OOF RMSLE: 0.10991 | time: 0.02 min
- Lasso | OOF RMSLE: 0.10989 | time: 0.02 min
- Ridge | OOF RMSLE: 0.11093 | time: 0.01 min
- SVR | OOF RMSLE: 0.11379 | time: 0.08 min
- CatBoost | OOF RMSLE: 0.11811 | time: 9.47 min
- Meta(Ridge) | OOF RMSLE: 0.10820 | time: 0.00 min

0.12306 →0.12303
ちょっと上がったけど、非効率。

Catboost単体上げもトライしたけど
良い結果にならず。

もとのCatboostがそれなりに良かったと思われる。

今回はここまで。