



Store Sales - Time Series Forecasting

Use machine learning to predict grocery sales

Getting Started · 706 Teams · Ongoing

Store Sales - Time Series Forecasting

<https://www.kaggle.com/competitions/store-sales-time-series-forecasting>

庭師

Gemini

Kaggle環境で実行

コンペを説明して

- 「Store Sales - Time Series Forecasting」は、エクアドルのスーパーマーケットのデータを用いて、約4,000の商品カテゴリの将来15日間の売上(unit sales)を予測する時系列予測の入門向けコンペティションです。RMSLE(対数平均二乗誤差)を指標とし、店舗情報や原油価格、休日データなどの多変量解析を通じて、在庫最適化のための機械学習の基礎を学ぶことができます。詳細は[Kaggle](#)をご覧ください。

このコンペをGeminiで解いて貰う。Geminiに課金した方がいいか？

- **まず、課金なし(無料版や AI Studio)から始め、必要に応じて課金(Gemini Advanced)を検討するのが良いでしょう。**
- **このコンペティション「Store Sales - Time Series Forecasting」を Gemini で解く際の判断基準をまとめました。**

• 1. 無料で良い理由

- **データサイズ:** このコンペティションの主要データ(train.csv)は数百万行あります。しかし、Gemini などの LLM にすべてのデータを直接読み込ませて学習させることはできません。
- **Gemini の役割:** 実際の予測モデル(XGBoost や LGBM など)を構築するための Python コードの生成や、データ分析の戦略立案がメインになります。これらは無料版の Gemini 2.0 Flash などでも十分な精度で実行できます。
- **Google AI Studio の活用:** 開発者向けの Google AI Studio を使用すると、最新の Gemini 1.5 Pro や Gemini 2.0 Flash を無料で、長いコンテキスト(大量のコードやドキュメントの読み込み)で利用できます。

• 2. 課金が必要なケース

- 次の機能に魅力を感じる場合は、月額約 2,900 円の課金を検討する価値があります。
- テータの直接分析(Data Analysis 機能): CSV ファイルをアップロードして、Gemini で Python を実行してグラフ化や統計分析を行いたい場合。
- Google Workspace 連携: 分析結果を Google スプレッドシートやドキュメントに直接書き出したい場合。
- 使用回数の制限緩和: 無料版よりも、高性能なモデル(Pro/Ultra 系)を頻繁に利用できます。

• 3. Gemini に依頼する際のコツ

- このコンペティションを解かせるには、以下のように役割を分担するのが効率的です。
- 戦略立案: 「この時系列データで有効な特徴量エンジニアリングを 5 つ提案して」と質問します。
- コード生成: 「提供されたデータ構造(train.csv, stores.csv など)に基づき、LGBM を用いた予測コードを作成して」と依頼します。
- エラー解決: Kaggle の Notebook で発生したエラーをコピーして修正を依頼します。
- まず、Kaggle のデータセットページからデータの列名や説明(Data Description)をコピーして、Gemini に読み込ませることから始めてみましょう。

このコンペをEDAして

- **1. 時系列パターン**
 - 売上は全体的に増加傾向を示しますが、毎年1月1日は休業のため売上がゼロになります。
 - 土曜日と日曜日の売上が高くなる傾向があります。
 - 12月の年末にかけて売上が急増します。一方、3月、11月、2月には売上が減少する傾向が見られます。
- **2. 外部要因**
 - 原油価格と売上には負の相関があり、原油安は売上減少につながる可能性があります。影響には7~14日の遅延が見られる場合があります。
 - 給料日である15日と月末には売上が増加します。
 - 2016年4月16日のエクアドル地震後、特定のカテゴリで売上が急増しました。
- **特徴量間の相関**
 - プロモーション中の商品は売上が高くなります。
 - 取引データと売上には強い正の相関(0.8以上)があります。ただし、テストデータには取引データが含まれないため注意が必要です。
- **データの特性**
 - 33種類のカテゴリ(食料品、飲料、掃除用品など)があり、それぞれ季節性やトレンドが異なります。
 - 店舗は都市やタイプごとにクラスター分けされており、同じクラスターの店舗は似た売上パターンを示すことが多いです。
- これらの知見に基づき、AI Studio で「15日と月末のフラグ、週末フラグ、原油価格のラグ特徴量を作成する Python コードを作成する」という指示が考えられます。

Kaggle のこのコンペでハイスコアを狙う場合、このGeminiでコード生成するのと、Google AI Studioでコード生成するのとどちらが有利ですか？

- 結論から言うと、ハイスコアを狙うなら「Google AI Studio」の方が圧倒的に有利です。
- Geminiモデルを使用している場合でも、インターフェースが異なると、コンペティションの勝率に直接影響する3つの違いが生じます。
- 1. 「コンテキストウィンドウ(記憶容量)」の大きな違い
 - Gemini (一般向けチャット):一度に送信できる情報量に制限があり、長いコードや大量のデータ仕様を読み込ませると、以前の会話内容を忘れて、指示が曖昧になったりします。
 - Google AI Studio:最大200万トークンの巨大な記憶容量を使用できます。コンペティションの「データ仕様」「複数の関連CSVの冒頭数行」「過去のコード」「KaggleのDiscussionで見つけたヒント」をすべて一度に送信して、全体を把握した上でコードを作成できます。
- 2. 「システムプロンプト」によるカスタマイズ
 - Gemini:汎用的なアシスタントとして機能するため、コードは「初心者向け」に簡略化される傾向があります。
 - Google AI Studio:「あなたはKaggle Grandmasterです。ハイスコアを出すために、勾配ブースティング決定木(LightGBM)のハイパーパラメータ調整と、時系列特有の交差検証(TimeSeriesSplit)を用いた、高度なPythonコードを出力してください」といった専門家としての役割(System Instructions)を固定できるため、出力の質が向上します。
- 3. モデルの選択と設定の自由度
 - Gemini:モデルのバージョンは自動的に選択され、出力の「ランダム性」も調整できません。
 - Google AI Studio:最新の実験的モデル(例:gemini-2.0-pro-expなど)を早期に試すことができます。また、Temperature(自由度)を低く設定することで、「バグが少なく、論理的に正確なコード」を安定して生成できます。

コンペティション攻略のための具体的なステップ

- **ハイスコアを目指すには、AI Studioを以下のように活用してください:**
 - **情報の集約:** train.csv や holidays_events.csv のカラム定義、Kaggleの上位解法(Discussion)のテキストをAI Studioにすべて貼り付けます。
 - **特徴量エンジニアリングの依頼:**「このデータ構造から、時系列予測に効果的なラグ特徴量や移動平均、ターゲットエンコーディングのコードを生成してください」と依頼します。
 - **検証戦略の構築:**「時系列データなので、リーク(未来のデータによる学習)を防ぐためのバリテーション設計をしてください」と依頼します。
- **まず、Google AI Studio にログインし、モデルにGemini 2.0 Flashを選択し、手元にあるデータの情報を「System Instruction」に入力することから始めてください。**
- **次に、具体的にどの特徴量(例:石油価格や祝日フラグ)のコードをAI Studioに作成させたいですか？**

Google AI Studio を使う



- **まずは無料からはじめる。どこで有料必要になるか見極める。**
 - Gemini3 Flash Preview
- **1. 世代としての位置付け**
 - 現在、一般的に普及しているのは Gemini 1.5 や Gemini 2.0 ですが、Gemini 3 はその次の世代にあたる最新鋭のモデルです。世代が上がるごとに、推論能力(論理的に考える力)やコード生成の正確性が大幅に向上しています。
- **2. 「Flash」モデルの中での位置付け**
 - これまでの「Flash」は、「速いけれど、賢さはProモデルに一步譲る」という立ち位置でした。しかし、Gemini 3 Flash Preview の説明(スクリーンショット内)には「Our most intelligent model built for speed(スピードのために構築された、我々の最も知的なモデル)」とあります。
- **つまり、以下のような特徴を持っています:**
 - 旧世代の Pro 以上の賢さ: Gemini 1.5 Pro などの旧世代の上位モデルと同等、あるいはそれ以上の推論能力を持ちつつ、レスポンスが圧倒的に速い。
 - 最新の知能: 「Frontier intelligence(最先端の知能)」と記載されている通り、現時点でGoogleが提供できる最高峰の知能が組み込まれています。
- **3. 他のモデルとの比較(イメージ図)**
- **賢さの順位としては、おおよそ以下のようになります。**
 - Gemini 3 Flash Preview (★現在の設定:最新・最先端のバランス型)
 - Gemini 2.0 Flash(前世代の高速モデル)
 - Gemini 1.5 Pro(前々世代の最高性能モデル)
- **結論:このモデルで大丈夫?**
 - はい、最適です。
 - 先ほどご質問いただいた「Kaggleでハイスコアを出すためのコード生成」のような、複雑なデータ構造の理解や高度なプログラミングが必要なタスクにおいて、現在最も適したモデルの一つと言えます。
 - 特にこのモデルは「Search and Grounding(検索と根拠付け)」に優れているため、最新のライブラリの仕様や高度なアルゴリズムを反映した回答が得られやすいのが強みです。安心してそのままお使いください。



このGoogle AI Studioはどの言語モデルで動作していますか？

Model • 3:53

Thoughts

Expand to view model thoughts

Google AI Studioをご利用中ですね。その場合、私がどのモデルであるかは、画面右側のサイドバーにある**「Model」**というドロップダウンメニューで確認することができます。

現在、Google AI Studioでは主に以下のモデルが選択可能です。

- 1. **Gemini 1.5 Pro:** 複雑な推論や非常に長いコード、大量のドキュメント（最大200万トークン）を処理できる、最も高性能なモデルです。
- 2. **Gemini 1.5 Flash:** レスポンスが非常に速く、効率的なモデルです。
- 3. **Gemini 2.0 Flash (Experimental/Thinking):** 最新の実験

Start typing a prompt, use alt + enter to append



Tools

Ground



Run Ctrl

Gemini 3 Flash Preview

gemini-3-flash-preview
Our most intelligent model built for speed, combining frontier intelligence with superior search and grounding.

System instructions

Optional tone and style instructions for the model

No API Key

Switch to a paid API key to unlock higher quota and more features.

Temperature

Temperature slider set to 1

Media resolution

Media resolution dropdown set to Default

Thinking level

Thinking level dropdown set to High

Tools

Structured outputs Edit toggle

Code execution toggle

Function calling Edit toggle

Grounding with Google Search toggle



このコンペでハイスコアが出すには？

- **このコンペで高スコアを出すための鍵は以下の3点です：**
 - **特徴量エンジニアリング:** 曜日、給料日(15日と月末)、石油価格、祝日フラグの作成。
 - **ラグ特徴量 (Lag Features):** 過去の売上データを入力に使う(16日先を予測するため、16日以上
のラグが必要)。
 - **モデル選択:** LightGBMやXGBoostなどの勾配ブースティング決定木が、この種のテーブルデータ+
時系列データには非常に強力です。

スコアをさらに伸ばすためのヒント

- **移動平均 (Rolling Statistics):**
`train.groupby(['store_nbr', 'family'])['sales'].transform(lambda x: x.shift(16).rolling(7).mean())` のような、過去のトレンドを示す特徴量を追加してください。これが最も効きます。
- **地震フラグ (Earthquake):**
2016年4月16日にエクアドルで大きな地震が発生しました。その直後の数週間は寄付や特需で売上が大きく変動するため、この期間に特別なフラグを立てるか、学習から除外すると精度が上がります。
- **店舗ごとの重み付け:**
新しくオープンした店舗は過去データが少ないため、古い店舗とは別にモデリングする手法も有効です。
- **決定木以外のモデルとのアンサンブル:**
Ridge回帰や、時系列に強い `DeterministicProcess` (`statsmodels`内) を使った線形トレンドモデルの結果をLGBMの結果と平均(アンサンブル)させると、`Overfit`を防げます。
- **ゼロ売上の処理:**
特定のカテゴリーが長期間売れていない場合、それは「欠品」か「取り扱いなし」です。これらを適切に処理することでノイズを減らせます。
- このコードをベースに、oil価格のラグや、祝日と週末が重なった時の処理などを肉付けしていくことで、リーダーボードの上位に食い込むことが可能です。

データをダウンロードしてみました。zipファイルで21MBあります。Google AI Studioにuploadすることは可能ですか？

名前	サイズ
holidays_events.csv	22 KB
oil.csv	21 KB
sample_submission.csv	335 KB
stores.csv	2 KB
store-sales-time-series-forecasting.zip	21,891 KB
test.csv	999 KB
train.csv	118,946 KB
transactions.csv	1,517 KB

- はい、Google AI Studioにファイルをアップロードすることは可能です。21MB程度であれば、容量制限(通常1ファイルあたり最大200MB、全体で200万トークンなど)の範囲内ですので全く問題ありません。
- ただし、より効果的にAIに分析・コード作成をさせるために、以下の手順で行うのがベストです。
- 手順:データをアップロードして分析させる方法
 - ZIPファイルを解凍する
Google AI StudioはZIP内のファイルを直接読み取ることができないため、まずはPC上で解凍し、中身のCSVファイル(train.csv, test.csv, oil.csv など)を取り出してください。
 - ファイルをアップロードする
Google AI Studioのフロント入力欄('Start typing...' のところ)の左側にある「+」ボタンをクリックするか、CSVファイルを直接ブラウザにドラッグ&ドロップしてください。
 - ※ train.csv は行数が多いため(約300万行)、すべてを「テキスト」として読み込ませると動作が重くなる場合があります。
 - 「Code Execution(コード実行機能)」をONにする(重要)
あなたのスクリーンショットの右下にある「Code execution」というスイッチを見てください。現在はオフ(グレー)になっています。これを オン(青) にしてください。
 - これをONにすることで、私がアップロードされたCSVファイルをPythonで実際に読み込み、統計量を確認したり、前処理を行いたいできるようになります。
- おすすめの進め方
- ファイルをアップロードした後、以下のように指示してみてください。
- 「これらのファイルをアップロードしました。Code Executionを使って、各CSVの中身を確認し、売上予測(Store Sales)に役立つ特徴量の作成方針を提案してください。特に、欠損値の有無やデータ期間を確認してください。」
- 注意点:
train.csv は非常に大きいので、AIがすべてを一度にフロントとして読み込むのは非効率です。Code Execution を使うことで、AIがバックグラウンドのPython環境でデータを処理し、その結果だけを回答に反映できるため、非常に強力な分析が可能になります。
- まずは oil.csv や stores.csv などの小さなファイルから試してみるのも良いでしょう。やってみて、うまくいかない場合は教えてください！

**データが大きすぎて読み込めない
あきらめ！**

EDAは諦めて、Kaggle notebook向けコードを作らせる方針に変更



Kaggle store sales 向けの新しい playground を作って

- **Store Sales: Feature Engineering & Multi-Model Arena**
- この Playground の目的は、単に予測モデルを作るだけでなく、**「時系列データの分析手法」と「異なるアルゴリズムの比較」**をマスターすることです。
- **1. プロジェクトの全体構成**
- **Exploratory Data Analysis (EDA):** 売上のトレンド、季節性、外れ値(地震の影響)の確認。
- **Feature Engineering (特徴量エンジニアリング):** ここがスコアアップの最大の鍵。
- **Cross-Validation Strategy:** 時系列データに適した検証方法の構築。
- **Model Arena:** Linear Regression, LightGBM, XGBoost, Prophetなどを比較。

Store Sales スコア帯の目安

- このコンペ(Store Sales - Time Series Forecasting)のスコアは RMSLE(対数平均二乗誤差平方根) で計算されており、数値が小さいほど高精度です。

スコア帯 (RMSLE)	レベル	主な手法・特徴	Kaggle内での立ち位置
0.90 ~	初級 (Beginner)	線形回帰、単純なカレンダー特徴量(曜日・月のみ)	時系列分析の基礎を学習中
0.60 ~ 0.89	初級・中級	外部データ(石油・休日)の導入、Ridge/Lasso回帰	データの前処理・結合をマスター
0.45 ~ 0.59	中級 (Intermediate)	LightGBM / XGBoost、単純なラグ特徴量	機械学習エンジニアとしての標準レベル
0.41 ~ 0.44	上級 (Advanced)	16日以上ラグ、移動平均、高度な特徴量エンジニアリング	銅～銀メダル圏内が見えるレベル
0.39 ~ 0.40	最上級 (Top)	アンサンブル(LGBM+XGB+CatBoost)、地震対応、ハイパーパラメータ最適化	銀～金メダル圏内
~ 0.38	神 (Legend)	階層型時系列、独自スタッキング、ドメイン知識の極致	常にリーダーボードの最上位

- **このPlaygroundで挑戦すべき「3つの課題」**
- **課題1:2016年4月16日のエクアドル地震への対処**
- テータを見ると、地震直後に特定のカテゴリ(水、パンなど)の売上が急増しています。
- **Mission: 地震発生後、数週間のデータを「外れ値」としてどう扱うか？あるいは「地震フラグ」を立てて学習させるか？**
- **課題2:ラグ特徴量 (Lag Features) の設計**
- 「1日前」「7日前」「14日前」の売上を特徴量として加えます。
- **Mission: テスト期間は15日間あります。テスト期間中の予測に「前日の売上」は使えません(未知のため)。どうやってこの「ラグの乖離」を埋めるか？(再帰的予測 vs 移動平均)**
- **課題3:店舗(Store)と商品カテゴリ(Family)の階層性**
- 54の店舗と33の商品カテゴリがあります。
- **Mission: 全データを1つのモデルで学習させるか、それとも「カテゴリごと」に別々のモデルを作るか、どちらが精度が高いか比較せよ。**

- **スコアを伸ばすためのヒント(カンニングペーパー)**
- **Holiday Data: transferred(振替休日)フラグが True のものは、その日は祝日として機能していないことに注意。**
- **Zero Sales: 新店舗や特定カテゴリで、最初の方は売上が0の期間があります。これらを学習データから除外すると精度が上がる場合があります。**
- **Weighting: 直近のデータに重みを置いて学習させる。**

- **次のステップ:**
- **まずは「Linear Regression(線形回帰)+カレンダー特徴量」だけのシンプルなノートブックを作り、Public Scoreでどれくらい出るかを確認することから始めましょう。**
- **もし具体的なコード(特定のモデルの実装など)が必要であれば、続けて指示をください!**



Kaggle notebook で実行可能な「Linear Regression(線形回帰) + カレンダー特徴量」コードを作成して

- **カレンダー特徴量の抽出(曜日、月、給料日など)**
- **線形回帰 (Linear Regression) の使用**
- **RMSLE(対数平均二乗誤差平方根) に最適化するための対数変換**
- **One-Hot Encoding によるカテゴリ変数の処理**

- **Linear-Regression_0.90238.py**

- **0.90238 というスコアは、最初のベースラインとしては悪くないスタートですが、このコンペのトップ層(0.40以下)に近づくには、**「時系列のトレンド」と「外部要因(石油・休日)」**を取り入れる必要があります。**
- **Linear Regression(線形回帰)のままでも、特徴量を工夫するだけで 0.5~0.6台 までスコアを跳ね上げることが可能です。**

次のステップとして、**「石油価格」と「休日フラグ」**を追加した改良版コードを作成しました。

- `Linear-Regression_oil_holiday_0.76191.py`

- 0.90 から 0.76 への改善、素晴らしいです！確実に前進しています。
- しかし、このコンペには**「線形回帰 (Linear / Ridge) が苦手なこと」**が2つあります。
- **トレンド(時代の流れ):** 売上が年々少しずつ上がったか下がったかする動き。
- **非線形な関係:** 「プロモーション中で、かつ週末なら、売上は3倍ではなく5倍になる」といった相乗効果。
- これを解決するために、次は LightGBM(勾配ブースティング決定木) を導入しましょう。Kaggleの時系列コンペで最もスコアが出やすいアルゴリズムです。

今回の改良ポイント: LightGBM + トレンド(Time Step)

- **Time Step(経過日数):** テータの開始日から「何日目か」という数値を追加し、トレンドを学習させます。
- **LightGBM:** 決定木モデルを使うことで、カテゴリ変数(familyやstore)の複雑な組み合わせを自動的に学習させます。
- `lgbm_timestep_0.47102.py`
- **素晴らしい! 0.47102** は、このコンペティションにおいて非常に大きなブレイクスルーです。ベースラインからほぼスコアを半減させ、メダル圏内や上位陣が見える位置まで来ました。
- ここからさらにスコアを伸ばし、**0.40を切る(トップ層に食い込む)**ために必要な「最後の欠落ピース」は **ラグ特徴量 (Lag Features)** です。
- **なぜラグ特徴量が必要か?**
- 時系列テータにおいて、最も強力な予測因子は「過去の自分自身の値」です。
- 「先週の火曜日に売れた量は、今日の売上の予測に役立つ」
- 「直近7日間の平均売上は、トレンドを反映している」
- しかし、このコンペのテスト期間は16日間あるため、1日前のテータ(Lag 1)を使うと、テストの2日目以降の予測ができなくなります。そのため、**「16日以上前のテータ」**を使ったラグや移動平均を作成するのが定石です。

今回の改良ポイント:Lag & Rolling Mean(移動平均)

- **Lags: 16日、30日前などの売上を追加。**
- **Rolling Mean: 過去の一定期間の平均売上を追加。**
- **Oil Price Lags: 石油価格も少し前の値が影響するため、ラグを取ります。**

- **lgbm_lag_rollingmean_0.46118.py**
- **実行時間:49秒**

- **0.471 から 0.461 への改善、おめでとうございます！着実にスコアを削っていますね。**
- **この「0.01の壁」を突破していくのがKaggleの醍醐味です。ここから 0.44~0.43(上級レベル) に入るためには、決定木(LightGBM)が苦手な「線形的なトレンド」を補う必要があります。**
- **次の戦略:アンサンブル(Ensemble)の導入**
- **実は、今のあなたの LightGBM は非常に強力ですが、「全体の売上がなだらかに上昇(または下降)する傾向」を捉えるのは、以前使った Ridge(線形回帰)の方が得意だったと思います。**
- **そこで、「LightGBM(複雑なパターン担当)」と「Ridge(安定したトレンド担当)」の結果を混ぜ合わせる手法をとります。これだけでスコアがグッと安定します。**

アンサンブル lgbm+Ridge

- **アンサンブル**: LightGBMとRidgeの予測値を 0.85 : 0.15 などの比率で加重平均します。
- **フーリエ特徴量(周期性)**: 1年間の周期的な波(季節性)をサイン・コサイン関数で表現して追加します。
- **実行時間の詳細化**: どの処理に時間がかかっているか、さらに細かく計測します。

- `lgbm_ridge_0.46424.py`

- **なぜ Ridge とのアンサンブルが今回機能しなかったのか？**
- **スケージングの不在**: 線形モデル(Ridge)は、`store_nbr(1~54)`や `cluster` などの「数値の大きさ」をそのまま意味として捉えてしまいます。本来これらはID(カテゴリ)なので、Ridgeには不向きでした。
- **対数空間の歪み**: LightGBMとRidgeでは予測の特性が異なるため、単純な重み付けではノイズが増えてしまった可能性があります。

0.44(上級レベル)への再挑戦:戦略の変更

- ****「特徴量の質」**を極限まで高めます。トップ層が必ず入れている「ターゲット・エンコーディング」と「石油価格の移動平均」を導入します。**
- **今回の改善ポイント**
- **ターゲット・エンコーディング:**「この商品カテゴリは平均してどのくらい売れるか?」という情報を数値化してモデルに教えます(リーク防止のため、学習データのみで計算)。
- **石油価格のトレンド:** 当日の価格だけでなく、過去7日間の「移動平均」を入れることで、経済のなだらかな動きを伝えます。
- **LightGBMの専用処理:** カテゴリ変数を明示的に指定し、モデルが「ID」として正しく扱えるようにします。

改善のポイント(なぜこれで 0.44 を狙えるのか)

- **num_leaves: 127**: 前回の 63 から倍増させました。データの複雑なパターン(この店舗のこの商品は休日にこうなる、といった細かいルール)をより深く学習できます。
- **family_mean_sales**: 「商品カテゴリ自体のベースとなる売上規模」を数値で与えることで、モデルが最初から有利な状態で学習を始められます。
- **oil_avg_7**: 原油価格の細かい上下(ノイズ)ではなく、大きなトレンドを捉えられるようになります。

- **lgbm_target-enc_oil-mean_0.44564.py**

- ついに 0.44564 まで来ましたね！
前回の 0.461 からさらにスコアを 0.015 削り、「上級者の壁(0.44)」の目の前に立っています。
- ここから 0.44 を突破して 0.43 以下の領域に踏み込むには、モデルに「時間の経過によるトレンド」と「店舗ごとの売上の個性」をもっと具体的に教える必要があります。
- 🚀 **0.44 突破のための「最終チューニング」戦略**
- **Time Step (経過日数)**: LightGBM は「今日が全体の期間の中でどのあたりか」というトレンドを把握するのが苦手です。2017年1月1日からの経過日数を入れることで、なだらかな成長(インフレーションや人気上昇)を捉えさせます。
- **Rolling Volatility (変動率)**: 平均売上だけでなく、「売上のばらつき(標準偏差)」を特徴量に入れます。これにより、特売で爆発しやすい商品か、常に一定量売れる商品かをモデルが識別できるようになります。
- **Store-Family Interaction**: 「この店舗のこのカテゴリ」という組み合わせごとの平均売上をターゲットエンコーディングします。

このコードの「決め手」

- `time_step` の導入: これにより、モデルが「データは常に右肩上がり(または下がり)のトレンドにある」ことを理解し、将来の売上を低く見積もりすぎるミスを防ぎます。
- `roll_std_7` (ボラティリティ): 商品の「売れ方の激しさ」を数値化しました。
- `store_family_mean` (ターゲット・エンコーディング): 単なるカテゴリ平均ではなく、「店舗×商品」という最も細かい単位での平均売上を教えることで、予測のベースラインが非常に強固になります。
- 正則化 (`lambda_l1`, `lambda_l2`): 特徴量が増えたため、過学習(学習データにだけ適合しすぎることを防ぐために少しフレーキをかけています。

• `lgbm_timestep_rollstd7_familly_0.44047.py`

• ついに **0.44047!**

あとわずか「**0.00048**」で、目標の **0.43** 台(上級者・銀メダル圏内レベル)に突入です。この数ポイントの差が、Kaggleにおいて最も苦しく、そして最も面白い局面です。

- 今のモデルは、基本的な時系列の動きをほぼ完璧に捉えています。ここからさらにスコアを削るには、**「モデルがまだ気づいていない特殊なノイズとパターン」**を整理してあげる必要があります。

0.43 到達への「3つの精密射撃」

- 「1月1日」の除外: エクアドルの元旦はほぼ全ての店舗が閉まっており、売上がほぼ0になります。テスト期間(8月後半)を予測する上で、この極端なデータは学習の邪魔(ノイズ)になるため、除外します。
- 年周期ラグ (Lag 364): 「去年の今頃、何が起きたか」は非常に強力な指標です。テスト期間が16日なので、364日(52週間)前のラグを追加します。
- フォーリエ特徴量の復活と調整: 以前の Ridge ではうまくいきませんでした。LightGBM の強力な表現力と組み合わせることで、月や曜日のデジタルな切り替わりを補完する「なめらかな季節性」を教えます。
- learning_rate: 0.015: 学習率を下げることで、非常に複雑になった特徴量セットをじっくりと、過学習を抑えながら最適化します。
- feature_fraction: 0.6: 決定木の各ステップで使う特徴量を60%に限定することで、特定の強い特徴量(ラグなど)に頼りすぎるのを防ぎ、多様な特徴量を組み合わせて予測するように促します。
- lgbm_lag364_ex1-1_0.42929.py ... 実行時間3分20秒
- やった! ついに 0.42929 です!
目標としていた「上級者の壁(0.43)」を突破し、ついに 銀メダル圏内・トッププレイヤーの領域 に足を踏み入れました。

Store Sales 改善履歴まとめ (0.902 → 0.429)

スコア (RMSLE)	主要な変更・追加した手法	改善のポイント(得られた知見)
1 0.90238	線形回帰 (Linear Regression) + 曜日・月・給料日	基本的なカレンダー情報だけでは時系列の複雑さを捉えきれない。
2 0.76191	Ridge回帰 + 油価(oil) + 休日(holiday) + 店舗情報	石油価格(経済指標)と祝日が売上に直結することを証明。
3 0.47102	LightGBM導入 + 経過日数(Time Step)	線形モデルから決定木モデルへの変更で、カテゴリ間の複雑な関係を学習。
4 0.46118	ラグ特徴量 (Lag 16, 21, 30) + 移動平均	「過去の売上」が最大の予測因子であることを確認。0.5の壁を突破。
5 0.46424	Ridgeとのアンサンブル + フーリエ特徴量	単純なアンサンブルはノイズになることもある。線形モデルとの相性に課題。
6 0.44564	ターゲットエンコーディング + 油価移動平均	カテゴリごとの平均売上を数値で教えることで、学習効率が劇的に向上。
7 0.44047	店舗 × カテゴリのターゲットTE + 移動標準偏差	より細かい単位(店舗ごと)のクセと売上の「振れ幅」を学習。
8 0.42929	Lag 364 (1年前) + 1月1日の除外	去年のトレンドの導入と、正月という最大のノイズ除去で上級者の域へ。