NTT 株価予測モデルの開発

Kalalou Ilias

March 22, 2025

Contents

1 はじめに **3**

2 探索的データ解析 (EDA) **3**
2.1 データ概要 3
2.2 可視化と記述統計分析 3

3 データ前処理と特徴量エンジニアリング **6**
3.1 クリーニングと再インデックス 6
3.2 正規化 6
3.3 時系列シーケンスの作成 6

4 予測モデルと学習 **7**
4.1 ARIMA モデル 7
4.1.1 ARIMA モデルの実装 7
4.1.2 ARIMA モデルの予測評価 8
4.1.3 ARIMA モデルの結果 8
4.1.4 ARIMA モデルの結果の可視化 8
4.2 LSTM モデル 9
4.2.1 LSTM モデルの実装 9
4.2.2 LSTM モデルの評価と結果 10
4.2.3 LSTM モデルの結果の可視化 10
4.3 多変量 GRU 11
4.3.1 利点: 11
4.3.2 欠点: 11
4.3.3 多変量 GRU モデルの実装と学習 11
4.3.4 多変量 GRU モデルの評価と結果 11
4.3.5 多変量 GRU モデルの結果の可視化 12
4.4 TCN（時系列畳み込みネットワーク） 12
4.4.1 利点: 13
4.4.2 欠点: 13
4.4.3 TCN モデルの実装と学習: 13
4.4.4 TCN モデルの結果: 13
4.4.5 TCN モデルの結果の可視化: 14

5 結果のまとめとプレゼンテーション資料の準備 **15**
5.1 結果の概要 15
5.2 結果の可視化 15
5.3 各モデルの改善の可能性 16

6 総合結論 **17**

7 コードの構成と GitHub リポジトリの構造 **18**

8 付録 **20**

1 はじめに

本報告書は、インターンシップの一環として NTT 株価予測モデルの開発プロジェクトについて記述したものです。本プロジェクトの目的は、機械学習手法を用いて時系列データの解析および予測を行うソリューションを開発することにあります。プロジェクトは以下の複数の段階で進められました：

- データの理解と探索的データ解析（EDA）
- データ前処理と特徴量エンジニアリング
- 予測モデルの選定と学習
- パフォーマンス評価と結果分析
- モデルの改善および反復的な改良

2 探索的データ解析 (EDA)

2.1 データ概要

本プロジェクトで使用したデータは、NTT の株価情報に基づいています。各レコードは、以下の情報を含んでいます：

- 日付け (Date)
- 終値 (終値)
- 始値 (始値)
- 高値 (最高値)
- 安値 (最低値)
- 出来高 (出来高)
- 変化率% (変化率%)

最初に、データ行列の最初の 5 行とその次元を表示して概要を把握しました。その後、データ構造を詳細に検証するための複数の表を生成し、データ品質についての結論を導きました。例えば、終値のグラフを描くことで、株価の全体的な傾向を確認しました。

2.2 可視化と記述統計分析

株価の挙動をより深く理解するために、以下のグラフを作成しました：

- **NTT** 終値のトレンド**:**
  図 1は、NTT が株式市場に上場して以来の終値の推移を示しています。NTT は非常に好調な市場デビューを果たしましたが、その後急激な下落を経験し、最終的には安定して徐々に上昇していることが確認されます。

- 終値の分布:
  図 2は、終値のヒストグラムと密度曲線を示しており、最も頻繁に観測される値が 30 ドルから 60 ドルの間にあることから、平均的な株価水準が推定されます。

- 月ごとの終値の変動:
  図 3に示すグラフは、終値の月ごとの変動を表しており、3 月から 6 月および 10 月に高い変動性が見られるため、これらの期間に特に注意を払う必要があることが示唆されます。

Figure 1: NTT 終値のトレンド

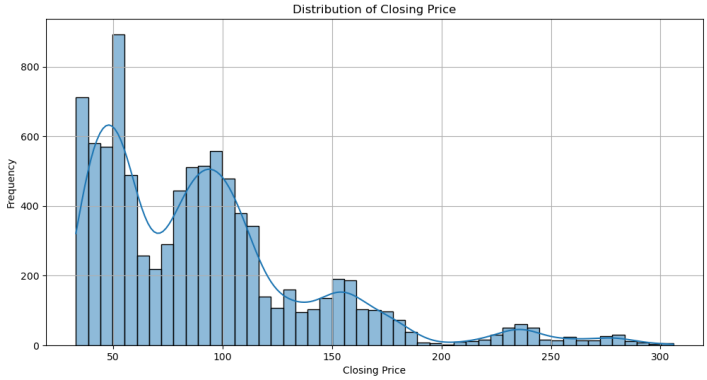

Figure 2: 終値の分布

Figure 3: 月ごとの終値の変動

これらの可視化と記述統計（平均、中央値、標準偏差など）を通じて、データ全体の傾向および潜在的な異常値を把握し、予測モデルの選定の基礎を固めました。

3 データ前処理と特徴量エンジニアリング

3.1 クリーニングと再インデックス

データ前処理は、以下のステップで実施されました：

- 日付の変換: 日付け列を `datetime` 形式に変換し、時系列操作を容易にしました。
- 再インデックス: 最小日付から最大日付までの全営業日を含むように DataFrame を再インデックスし、週末や祝日による欠損値は線形補間で埋め、連続した時系列を作成しました。

3.2 正規化

すべての変数が同じスケールになるように、以下の数値列を `MinMaxScaler` を用いて正規化しました：

- 終値 (終値)
- 始値 (始値)
- 高値 (最高値)
- 安値 (最低値)
- 出来高 (出来高)

出来高列は、専用の関数を用いて前処理（例："79.15M" を絶対数に変換）されました。また、分散の安定化とモデリングの容易化のため、ターゲットである終値に対して対数変換を適用しました：

$$\texttt{log(終値)} = \ln\left(\text{終値} + 10^{-6}\right)$$

3.3 時系列シーケンスの作成

ニューラルネットワークモデル向けに、時系列データをシーケンスに分割しました。本手法では、長さ 20 のシーケンスを用いて、ターゲット（`log(終値)`）の次の値を予測します。この工程は、時系列依存性を捉え、正確な予測のための十分な文脈情報をモデルに提供するために極めて重要です。

4 予測モデルと学習

4.1 ARIMA モデル

ARIMA（自己回帰和分移動平均）モデルは、以下の理由から選択されました：

- 利点**:**
  - シンプルさと解釈性*:* ARIMA は古典的な統計モデルで、自己回帰（AR）および移動平均（MA）の成分を識別し、系列を定常化するために必要な差分（I）を決定します。
  - 一歩先の予測効率*:* 過去の値と誤差に基づいて予測を行うことで、次の時刻の堅牢な予測が可能です。
- 欠点**:**
  - 線形性の仮定*:* 線形モデルであるため、金融データにおける非線形なダイナミクスを十分に捉えられない可能性があります。
  - 多段階予測の劣化*:* 複数ステップ先の予測では、ローリングフォーキャスト法により誤差が蓄積し、予測値が平均に収束する傾向があります。

これらの特徴により、ARIMA はベースラインモデルとして、また時系列の線形ダイナミクスの解析のために適しており、より高度な手法と比較する基準となります。

4.1.1 ARIMA モデルの実装

ARIMA モデルは以下のステップで実装されました：

1. ターゲット変換**:**
   終値列に対して対数変換を適用し、分散を安定化させました：
   $$\text{終値 \_log} = \ln\left(\text{終値} + 10^{-6}\right)$$
   これにより、大きな変動が抑えられ、系列がより定常化されます。

2. 定常性検定**:**
   対数変換後の系列に対して Augmented Dickey-Fuller (ADF) テストを実施し、定常性を確認しました（例：結果 0.34）。この結果から、1 回の差分（d=1）の必要性が示されました。

3. 学習/テスト分割**:**
   対数変換された系列を、約 95% を学習用、5% をテスト用に分割しました。これにより、豊富な過去データでモデルを学習させ、新規データに対する予測性能を評価できます。

4. ローリングフォーキャスト**:**
   実際の予測シナリオを模擬するために、ローリングフォーキャスト手法を採用しました。ARIMA(0,1,1) モデルを使用し、利用可能な履歴に基づいて各イテレーションで一歩先の予測を行い、実際の観測値を履歴に追加することでモデルを連続的に更新しました：
   $$y_{t+1} = ARIMA(y_1, y_2, \ldots, y_t)$$
   予測値は指数変換され、元のスケールに戻されます。

4.1.2 ARIMA モデルの予測評価

ARIMA モデルの信頼性を評価するために、以下の指標を計算しました：

- **RMSE**（平方根平均二乗誤差）と **MAE**（平均絶対誤差）: 予測値と実際の値との全体的な乖離を測定します。
- カスタムトレンドおよび近接性評価**:**
  毎日の変化の方向（上昇または下降）の予測精度と、予測値と実際の値の近接性を評価するためのアルゴリズムを開発しました。各日（2 日目以降）において、実際の変化と予測変化を比較し、以下の指標を算出します：
  - トレンド精度*:* 変化の符号が正しく予測された日の割合。
  - 近接性精度*:* 予測値と実際の値の絶対差に基づき、差がゼロの場合は 100%、1 ドル以上の場合は 0% となるスコアが日々割り当てられます。

  グローバルスコアは、これら 2 つの精度の平均として定義されます。

4.1.3 ARIMA モデルの結果

ARIMA モデルは以下の結果を示しました：

- **RMSE:** 1.71
- **MAE:** 1.32

さらに、カスタム評価では：

- トレンド精度**:** 学習/テスト分割により 54% から 63% の範囲となりました。

これらの結果は、ARIMA モデルが次の値の予測において堅牢な性能を発揮する一方、誤差の蓄積により長期予測能力には限界があることを示しています。

4.1.4 ARIMA モデルの結果の可視化

図 4は、ARIMA モデルのローリングフォーキャスト手法によって得られた予測曲線と実際の値を比較したものです。予測曲線は株価全体の傾向を概ね捉えていますが、いくつかのずれが見受けられます。これらのずれは、ARIMA の線形性および多段階予測における誤差累積に起因しています。

多数のテストの結果、欠損値が補完された完全なデータを用い、正規化されていない状態で ARIMA モデルを使用した場合に、より良好な予測性能が確認されました。正規化は値のスケールをわずかに変化させ、予測精度に影響を与えるためです。

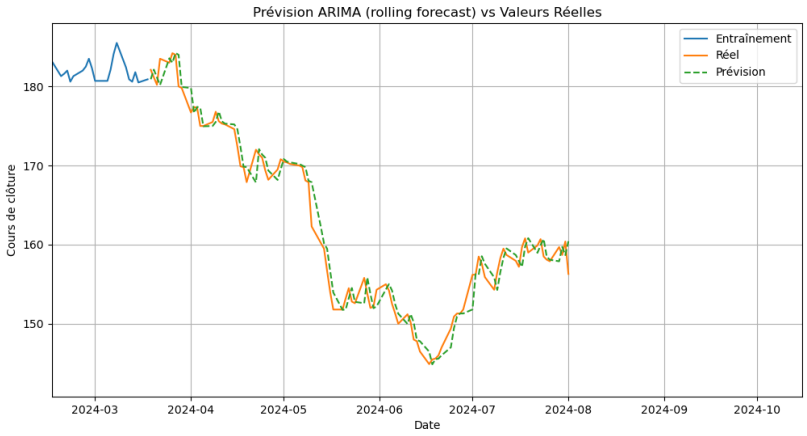

Figure 4: ARIMA ローリングフォーキャスト vs. 実際の値

4.2 LSTM モデル

基準および ARIMA モデルとの性能比較のために、単変量 LSTM モデルが開発されました。LSTM（Long Short-Term Memory）の選択は、標準的なリカレントニューラルネットワークにおける勾配爆発や勾配消失の問題を克服し、時系列データの長期依存性を捉える能力に基づいています。

利点:

- 長期依存性の管理: LSTM の内部メモリ構造により、長期間にわたって情報を保持でき、複雑なダイナミクスを持つ金融時系列に有用です。
- 非線形関係への適応: ARIMA のような線形モデルと異なり、LSTM は変数間の非線形な関係をモデル化できるため、非常に変動性の高い系列に対して潜在的な改善が期待されます。

欠点:

- 計算複雑性: ハイパーパラメータが適切に調整されない場合、LSTM の学習には多くの計算時間とリソースが必要となります。
- 大量データの必要性: LSTM は信頼性のある性能を発揮するために大量の観測値を必要とするため、小規模な時系列データでは難しい場合があります。

4.2.1 LSTM モデルの実装

LSTM モデルは、以下のステップで実装されました：

1. データ準備:
   終値（株価終値）列を生データまたは正規化された形で抽出し、固定長の時系列シーケンス（例：10 タイムステップ）を作成してネットワークの入力としました。
2. 学習/テスト分割:
   系列は、例えば 95% を学習用、5% をテスト用に分割され、大量の過去データで学習し、新たなデータに対する予測性能を評価できるようにしました。

3. モデルアーキテクチャ:
   LSTM モデルは、50 個のニューロンを持つ 2 層のリカレント層と、予測を生成する全結合層で構成されます。各ミニバッチの初期状態はゼロに設定されます。

4. 学習と最適化:
   Adam オプティマイザと平均二乗誤差（MSE）損失関数を用いて、50 エポックで学習が行われ、学習損失はプログレスバーで監視されました。

5. 予測と評価:
   学習後、テストセットに対する予測が生成され、RMSE および MAE が計算されました。また、カスタム評価アルゴリズムにより、トレンド精度と近接性が測定され、全体の信頼性スコアが算出されました。

4.2.2 LSTM モデルの評価と結果

LSTM モデルの評価は、生データおよび正規化データの両方で行われました。生データの場合、RMSE が 159.84、MAE が 159.49 という非常に高い誤差が観測され、一方、正規化データでは RMSE と MAE がそれぞれ 0.14 と極めて低い値となりました。この乖離は、正規化によるスケールの縮小が誤差値を低下させた結果であり、実際の予測性能を十分に反映しているとは言い難いです。総じて、LSTM は非線形関係をモデル化する潜在能力を示すものの、本プロジェクトにおけるその性能は限定的であり、さらなる改善（例：ハイパーパラメータの最適化、より高度なアーキテクチャの採用）が必要です。したがって、LSTM モデルは、ARIMA などの古典的手法との比較におけるベースラインとしての役割を果たします。

4.2.3 LSTM モデルの結果の可視化

図 5は、生データおよび正規化データに対して LSTM モデルが生成した予測と実際の値との比較を示しています。モデルは全体的な傾向を追う場合もありますが、顕著なずれが存在し、本特定時系列に対する LSTM アプローチの限界を浮き彫りにしています。

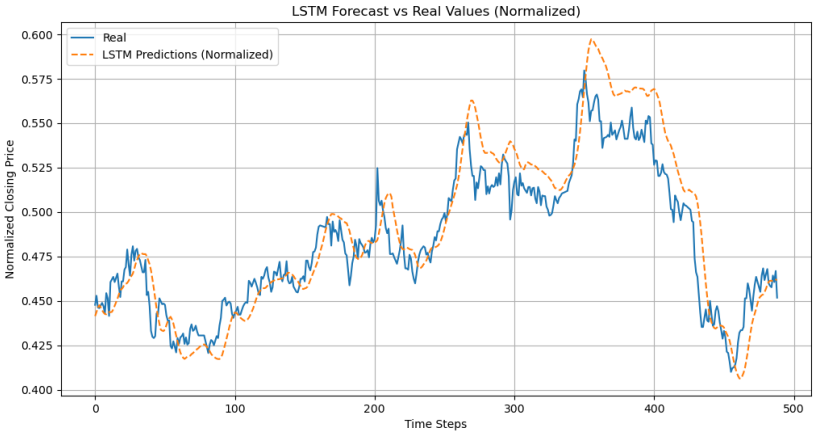

Figure 5: LSTM 予測 vs. 実際の値

4.3 多変量 GRU

本モデルは、複数の特徴量を活用して終値の対数変換値を予測します。単変量モデルとは異なり、多変量 GRU は、始値、最高値、最低値、出来高などの各種指標を統合することで、金融データの相互作用およびダイナミクスをより正確に捉えます。

4.3.1 利点:

- 多変量情報の活用*:* 複数の特徴量を同時に使用することで、複雑な相互作用を検出し、予測精度を向上させることが可能です。
- パラメータ効率*:* GRU は一般的に LSTM よりも少ないパラメータで済むため、大規模データセットの学習が容易です。
- 学習の安定性*:* 勾配クリッピングなどの技術により、非常に変動の激しい金融時系列でも学習が安定します。

4.3.2 欠点:

- ハイパーパラメータへの感度*:* シーケンス長、学習率、学習/テスト分割などの選択がモデル性能に大きく影響するため、広範な実験が必要です。
- 前処理の複雑さ*:* 多変量データの準備には、各変数のスケールの違いが学習プロセスに影響しないよう、厳格な正規化が求められます。

4.3.3 多変量 GRU モデルの実装と学習

多変量 GRU モデルは、以下の手順で実装されました：

1. データ準備**:**
   終値列を（生データまたは正規化された形で）抽出し、他の特徴量と統合した DataFrame を作成。専用の関数を用いて、固定長の時系列シーケンス（例：10 タイムステップ）に分割しました。
2. 学習/テスト分割**:**
   シーケンスは、例えば 95% を学習用、5% をテスト用に分割されました。
3. モデルアーキテクチャ**:**
   多変量 GRU モデルは、各 100 個の隠れニューロンを持つ 2 層の GRU と、予測出力用の全結合層で構成されています。学習安定化のため、勾配クリッピングが適用されました。
4. 学習**:**
   モデルは、学習率 0.001 の Adam オプティマイザを用いて 50 エポックで学習され、各エポックで予測値と実際の値との平均二乗誤差（MSE）が最小化されました。

4.3.4 多変量 GRU モデルの評価と結果

多変量 GRU モデルの性能は、RMSE および MAE といった古典的指標と、トレンド精度および近接性を評価するカスタム指標によって評価されました。例えば、正規化された終値系列を使用した場合、モデルは以下の結果を示しました：

- 生データの場合:
  - *RMSE:* 155.58
  - *MAE:* 155.22
  - トレンド精度*:* 48.77%
- 正規化データの場合:
  - *RMSE:* 0.017
  - *MAE:* 0.014
  - トレンド精度*:* 48.15%

これらの結果は、多変量 GRU が単変量アプローチに比べて予測誤差をある程度削減できることを示唆していますが、正規化データでの極端に低い誤差は、実際の予測精度よりもスケーリングの影響を反映している可能性があります。また、トレンド精度が約 48〜49% に留まることは、日々の変動方向を正確に予測するのに苦戦していることを示しています。これらの結果から、さらなる特徴量の改善やハイパーパラメータの微調整が必要であることが示唆されます。

4.3.5 多変量 GRU モデルの結果の可視化

図 6は、多変量 GRU モデルが生成した予測と実際の値の比較を示しています。モデルは全体的な傾向を概ね捉えていますが、日々の変動に関しては明確なずれが見受けられます。

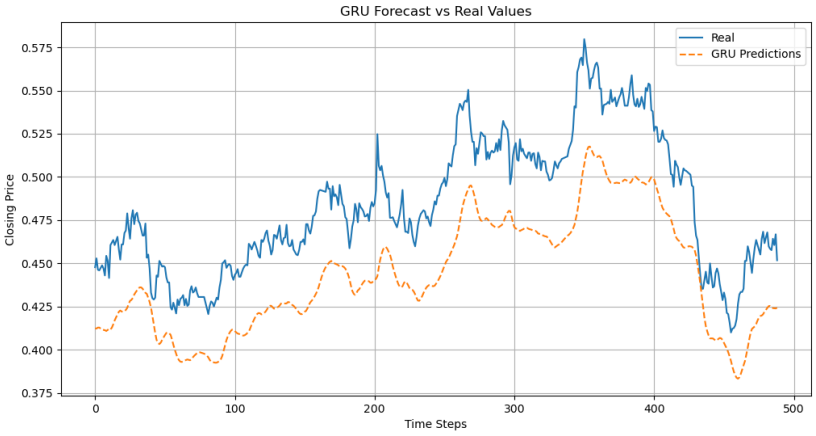

Figure 6: 多変量 GRU 予測 vs. 実際の値

4.4 TCN（時系列畳み込みネットワーク）

TCN モデルは、拡張畳み込みブロックを用いて複数のスケールで時系列依存性を捉えるモデルです。クラス `TCN` は、各ブロックが 2 層の拡張畳み込み、ドロップアウト、ReLU 活性化で構成され、最後のタイムステップの出力に線形層を適用して予測値を生成します。

4.4.1 利点:

- マルチスケールキャプチャ*:* 拡張畳み込みにより、短期および長期の依存関係を再帰的な手法を使わずに捉えられます。
- 並列処理*:* 畳み込み操作は並列化可能であり、リカレントモデルに比べて学習時間を短縮できます。
- 勾配の安定性*:* 畳み込みアーキテクチャにより、勾配爆発や勾配消失の問題が緩和されます。

4.4.2 欠点:

- ハイパーパラメータへの感度*:* TCN の性能は、カーネルサイズ、層の数、拡張率、ドロップアウト率に敏感であり、慎重な調整が必要です。
- 解釈性の低さ*:* 畳み込みおよび拡張操作により、従来の線形モデルに比べてモデル内部の動作の解釈が困難になる場合があります。

4.4.3 TCN モデルの実装と学習:

TCN モデルは、以下の手順で実装されました：

1. データ準備:
   終値列に対応する時系列データを（生データまたは正規化データで）抽出し、専用の関数を用いて固定長のシーケンス（例：10 タイムステップ）に分割しました。
2. モデルアーキテクチャ:
   TCN は、複数の時系列ブロック（各ブロックは拡張畳み込み、ReLU 活性化、ドロップアウトおよび残差接続から構成）を積み重ね、最後のブロックの出力を線形層に渡して最終予測を生成します。
3. 学習:
   学習は、学習率 0.001、平均二乗誤差（MSE）損失関数を用いて 50 エポックで実施され、プログレスバーで進捗が監視され、10 エポックごとに損失が報告されました。
4. 評価:
   学習後、モデルはテストセット（例：全体の 10%）で評価され、RMSE および MAE が算出されました。さらに、カスタム評価によりトレンド精度および近接性が評価され、全体の信頼性スコアが導出されました。

4.4.4 TCN モデルの結果:

正規化された終値系列で学習した TCN モデルのテストにより、以下の指標が得られました：

- 生データの場合:
  - *RMSE:* 8.79
  - *MAE:* 8.35

  - トレンド精度*:* 49.28%

- 正規化データの場合**:**
  - *RMSE:* 0.213
  - *MAE:* 0.199
  - トレンド精度*:* 51.63%

これらの結果は、TCN が複数のスケールで時系列依存性を効果的に捉え、従来のリカレントモデルと比べて競争力のある予測を提供していることを示しています。ただし、生データと正規化データとの間の指標の差は、モデルがデータスケールに対して非常に敏感であることを示しており、適切な正規化およびハイパーパラメータの最適化が不可欠であることを示唆しています。

4.4.5 TCN モデルの結果の可視化**:**

図 7は、TCN モデルが生成した予測と実際の値との比較を示しています。モデルは全体的な傾向を概ね捉えていますが、いくつかの逸脱が予測能力の改善余地を示しています。

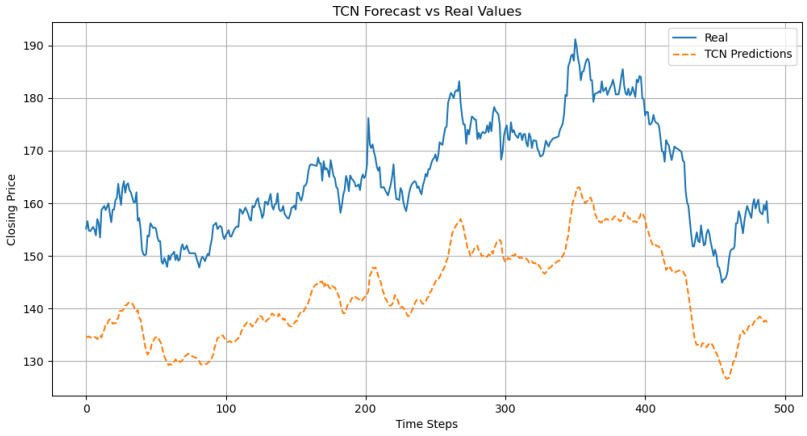

Figure 7: TCN 予測 vs. 実際の値

5 結果のまとめとプレゼンテーション資料の準備

本節では、ARIMA、LSTM、多変量 GRU、TCN といった各予測モデルの学習結果を総合し、これらの結果を明確かつ分かりやすく伝えるためのプレゼンテーション資料の作成について述べます。

5.1 結果の概要

各予測モデルは、RMSE（平方根平均二乗誤差）および MAE（平均絶対誤差）といった従来の指標、さらに日々の変化の方向を正しく予測する能力（トレンド精度）と、予測値と実際の値の近接性を評価するカスタム評価を用いて評価されました。主な結果は以下の通りです：

- **ARIMA:**
  - *RMSE:* 1.71
  - *MAE:* 1.32
  - トレンド精度*:* 54%〜63%（学習/テスト分割による）
- **LSTM**（単変量）**:**
  - 生データの場合: RMSE = 159.84, MAE = 159.49, トレンド精度 = 47.74%
  - 正規化データの場合: RMSE = 0.14, MAE = 0.14, トレンド精度 = 49.59%
- 多変量 **GRU:**
  - 生データの場合: RMSE = 155.58, MAE = 155.22, トレンド精度 = 48.77%
  - 正規化データの場合: RMSE = 0.017, MAE = 0.014, トレンド精度 = 48.15%
- **TCN:**
  - 生データの場合: RMSE = 8.79, MAE = 8.35, トレンド精度 = 49.28%
  - 正規化データの場合: RMSE = 0.213, MAE = 0.199, トレンド精度 = 51.63%

総合すると、本プロジェクトにおいては ARIMA モデルが最も適した手法として浮上しました。ARIMA モデルは、RMSE が 1.71、MAE が 1.32 という結果から、元のスケールでの堅牢な予測能力を示しています。また、トレンド精度が 54%〜63% と比較的良好であり、日々の変化の方向を十分に予測できることを示しています。一方、ニューラルネットワークベースのモデル（LSTM、多変量 GRU、TCN）は、理論的には非線形関係をより良くモデル化できる可能性があるものの、本ケースでは生データに対する性能が非常に低く、ほとんど無視できるレベルに留まっています。これらの低い性能は、正規化に対する過敏性や正規化スケール上での過学習に起因している可能性があり、元のスケールで信頼性のある予測を行う能力が制限されています。

5.2 結果の可視化

生成されたグラフは、各モデルの実際の値と予測値との比較を示しています。これらのグラフから、ARIMA モデルは全体的に一貫性があり堅牢な予測を提供している一方、ニューラルネットワークベースのモデルは生データ上で有意な結果を出すのに苦戦していることが明らかです。これにより、本プロジェクトにおいて ARIMA が最適な手法であることが確認され、他のアプローチの限界が浮き彫りになりました。

5.3 各モデルの改善の可能性

性能と堅牢性をさらに向上させるため、各アプローチに対して以下の改善の可能性が提案されました：

- **ARIMA:**
  - 外部変数の組み込み*:* ARIMAX モデルへの移行により、マクロ経済指標や業界シグナルなど外部要因を統合することで予測の精度を向上できる可能性があります。
  - 季節性モデル*:* SARIMA の利用により、金融データに存在する周期的変動をより正確に捉えることができます。
  - 学習/テスト分割の最適化*:* 異なる比率や誤差の重み付け技法を検討することで、多段階予測における誤差累積を抑制できる可能性があります。
- **LSTM**（単変量）**:**
  - より深いまたは双方向のアーキテクチャ*:* より深いネットワークまたは双方向 LSTM を検討することで、複雑な時系列依存性をより効果的に捉えられる可能性があります。
  - アテンション機構の導入*:* アテンション機構を組み込むことで、重要な期間に焦点を当て、予測精度を向上させることが期待されます。
  - ハイパーパラメータの最適化*:* ニューロン数、学習率、正則化（ドロップアウト、L2 など）を微調整することで、特に変動の激しい時系列における汎化性能を向上させることができるかもしれません。
- 多変量 **GRU:**
  - 特徴量の選択とエンジニアリング*:* 追加のテクニカル指標を取り入れるなどして、利用可能な情報の質を向上させることが鍵となります。
  - ハイパーパラメータの微調整*:* 隠れ層のサイズ、層数、勾配クリッピングなどを調整することで、学習の安定性をさらに向上させることができます。
  - 正則化の強化*:* ドロップアウトや L2 正則化のような追加の正則化技術を導入することで、過学習の防止が期待されます。
- **TCN:**
  - 多様なアーキテクチャとカーネルサイズ*:* より深いアーキテクチャや異なるカーネルサイズを試すことで、さまざまなスケールの依存性をより正確に捉えられる可能性があります。
  - 拡張率とドロップアウトの最適化*:* これらのパラメータの微調整は、モデルの柔軟性と安定性のバランスを取る上で不可欠です。
  - 残差接続の強化*:* 残差接続を強化し、追加の正規化技術を導入することで、収束を促進し、長期予測精度の向上が期待されます。

6 総合結論

本プロジェクトは、技術的にも概念的にも大きな挑戦を伴いました。ARIMA のような古典的手法から、LSTM、多変量 GRU、TCN といった先進的なニューラルネットワーク技術まで、様々な予測モデルの実装と評価には、時系列に適した AI モデルを特定し、その利点と限界を理解するための広範な調査が必要でした。

また、これらのモデルの実装においては、正規化、シーケンス作成、学習/テスト分割などのデータ前処理の複雑さや、生成されたグラフの解釈、ハイパーパラメータの調整、アーキテクチャの最適化など、多くの困難が伴いました。しかし、これらの課題を克服することで、各モデルの比較に基づいた堅牢なアプローチを構築することができました。各モデルは、金融時系列のダイナミクスに関する補完的な洞察を提供し、複雑性、解釈性、および予測性能の間にあるトレードオフを明らかにしました。

最終的に、本研究は予測モデリングと時系列分析における自らのスキルを大幅に向上させたとともに、結果を明確に伝えることの重要性も強調しました。よく構造化されたプレゼンテーション資料の作成は、複雑な技術概念を伝え、データに基づいた意思決定を促進する上で不可欠であるといえます。

まとめると、本プロジェクトは、時系列予測に応用される AI アプローチの豊かさと多様性を示すとともに、その実装および解釈に内在する課題を浮き彫りにしています。

7 コードの構成と GitHub リポジトリの構造

保守性と可読性を向上させるため、コードは予測プロセスの各ステップに特化した複数のファイルに分割されています。以下に、主要ファイルとその機能の概要を示します：

- **data_preprocessing.py**:
  - *load_and_preprocess_data*: CSV ファイルからデータを読み込み、日付列を `datetime` 型に変換し、（特に出来高列の変換を含む）データのクリーニングを行い、全営業日を含むように DataFrame を再インデックスします。
  - *normalize_data*: 指定された列を `MinMaxScaler` を用いて正規化します。
  - *create_sequences*: 時系列データを固定長のシーケンスに分割し、モデルへの入力準備を行います。
  - *visualisation*: トレンド、分布、月次変動など、探索的データ解析（EDA）用のグラフを生成します。
  - *information*: 記述統計を提供し、欠損している日付を特定します。
  - *convert_volume*: 「79.15M」などの出来高の値を絶対数に変換します。
  - *plot_forecast*: 予測曲線と実際の値を比較して表示します。
  - *adjust_monthly_volatility*: 異常なピークを低減することで、時系列の変動性を平滑化します。
- **model_arima.py**:
  - ARIMA モデルの実装。
  - *rolling_forecast_arima*: 各イテレーションで ARIMA モデルを再学習し、ローリングフォーキャストを実行します。
- **model_lstm.py**:
  - 単変量 LSTM モデルを実装する `LSTMForecast` クラスの定義。
  - *train_and_evaluate*: LSTM モデルの学習を行い、予測結果と評価指標（RMSE, MAE）を返します。
- **model_gru.py**:
  - 多変量 GRU モデルを実装する `GRUMultivariate` クラスの定義。
  - `create_sequences` 関数(data_preprocessing.py と類似)および *train_and_evaluate_gru* 関数を含み、モデルの学習と評価を行います。
- **model_tcn.py**:
  - 拡張畳み込みブロック（`TemporalBlock` クラス）の定義（パディングとドロップアウトを含む）。
  - 複数の `TemporalBlock` を積み重ね、最終的に線形層を通じて予測を生成する `TCN` クラスの定義。
  - *create_sequences* および *train_and_evaluate_tcn* 関数を含み、TCN モデルの学習と評価を実施します。

- **evaluation.py**:
  - トレンド精度および近接性などのカスタム評価指標を計算する *evaluate_forecast* 関数の実装。
  - RMSE や MAE などの従来の指標も計算します。
- **main.py**:
  - データの読み込み、前処理、各モデル（ARIMA、LSTM、GRU、TCN）の学習および評価、グラフ生成など、プロジェクトの全工程を統括します。
  - 各ファイルに定義された関数の呼び出しを調整し、最終結果を生成します。
- **stock_price.csv**:
  - 株価情報（例：終値, 始値, 高値, 安値, 出来高など）を含む生データファイルです。

各ファイルには、関数の役割が詳細にコメントされており、コードの保守や拡張が容易になるように配慮されています。

8 付録

最終的な所感

- GitHub リポジトリには、上記すべてのファイルに加えて、コードの実行方法や結果の再現方法を詳述した README が含まれます。
- プレゼンテーション用のスライドは、本報告書に基づき、データ理解からモデル検証までの全プロセスを示す内容となります。