Relatório do Projeto: Previsão Preditiva do IBOVESPA

Curso: Pós-Graduação em Data Analytics

Autor: Jorge Diego Guarda Gibin

---

1. Introdução e Objetivo do Projeto

Este projeto foi desenvolvido para cumprir o desafio de criar um modelo preditivo de séries temporais para prever diariamente os dados de fechamento da bolsa de valores da IBOVESPA.

O requisito principal era desenvolver uma estratégia de dados completa que resultasse num modelo com uma **"assertividade" de pelo menos 80%**.

1.1. Definição da Métrica de Sucesso

O termo "assertividade" é ambíguo e precisou ser traduzido para uma métrica estatística de erro. A métrica escolhida foi o **SMAPE (Symmetric Mean Absolute Percentage Error)**, que mede o erro percentual de forma simétrica.

A "assertividade" foi definida como `100% - SMAPE %`. Portanto, o objetivo do projeto era alcançar um **SMAPE inferior a 20%**.

2. Metodologia Aplicada

Foi implementado um *pipeline* de *Data Science* robusto, focado em garantir a comparabilidade e a validade estatística dos modelos. O processo seguiu 5 fases principais:

1. **Carga e Limpeza de Dados:** Tratamento do arquivo `dados10anos.csv` para converter formatos de texto complexos em dados numéricos utilizáveis.
2. **Análise de Estacionaridade:** Uso do Teste ADF (Augmented Dickey-Fuller) para diagnosticar a série temporal.
3. **Engenharia de Features e Validação:** Criação de variáveis preditivas (diferenciação, uso do Volume) e definição de uma estratégia de validação cronológica.
4. **Modelagem Comparativa:** Treinamento de três modelos distintos (Naive, ARIMA, LSTM) para competir entre si.
5. **Avaliação e Conclusão:** Comparação justa dos modelos na escala original e verificação do atingimento da meta.

3. Execução do Projeto Passo a Passo

A seguir, detalha-se a execução de cada fase implementada no *notebook* Python.

Fase 1: Carga e Pré-processamento dos Dados

O conjunto de dados `dados10anos.csv` apresentava desafios significativos de formatação que impediam a sua utilização direta.

- **Problema 1 (Preços):** Os preços estavam formatados como *strings* com `.` como separador de milhar (ex: `"145.517"`).
- **Problema 2 (Volume):** O volume de negociação usava sufixos `M` (Milhões) e `B` (Bilhões) e `,` como separador decimal (ex: `"8,34B"`, `"3,74M"`).
- **Problema 3 (Datas):** As datas estavam no formato `dd.mm.AAAA`.

Solução Aplicada:

1. **Datas:** A coluna `Data` foi convertida para o formato `datetime` do pandas.
2. **Preços:** Foi aplicada uma função para remover os `.` e converter a *string* para um tipo numérico (`int64`).
3. **Volume:** Foi criada uma função personalizada para:
    - Trocar `,` por `.` para padronizar o decimal.
    - Identificar o sufixo `M` ou `B`.
    - Remover o sufixo e multiplicar o número pela constante apropriada (1.000.000 para `M` ou 1.000.000.000 para `B`).

Ao final desta fase, tínhamos um *DataFrame* limpo, ordenado cronologicamente, contendo duas colunas de interesse: `y` (preço de fechamento) e `volume`.

Fase 2: Análise de Estacionaridade e Engenharia de Features

Os modelos de séries temporais, especialmente os da família ARIMA, funcionam de forma mais eficaz ou assumem que os dados são **estacionários** (onde a média e a variância não mudam ao longo do tempo).

Diagnóstico (Teste ADF):

- O **Teste Augmented Dickey-Fuller (ADF)** foi aplicado à série de preços (`y`).
- O resultado (P-Valor > 0.05) **confirmou que a série de preços NÃO é estacionária**, como esperado.

Solução (Engenharia de Features):

1. **Diferenciação:** Para tornar a série estacionária, foi aplicada a **primeira diferenciação** (`y_diff = y.diff()`). Esta nova série representa a *variação diária* do preço.

2. **Confirmação (Teste ADF):** O teste foi re-executado em `y_diff`, resultando num P-Valor baixo (< 0.05), **confirmando que a série de variações É estacionária**.
3. **Variável Exógena:** O mesmo processo foi aplicado à coluna `volume`, criando a `volume_diff`.

Decisão Metodológica: Todos os modelos seriam treinados para prever a `y_diff` (a variação), usando a `volume_diff` como uma *feature* de apoio.

Fase 3: Estratégia de Validação e Modelagem

Uma divisão de dados aleatória (`train_test_split`) é incorreta para séries temporais. A validação deve respeitar a cronologia.

- **Divisão:** O conjunto de dados foi dividido em **Treino** (todos os dados, exceto os últimos 100 dias) e **Teste** (os últimos 100 dias).
- **Horizonte (H):** O objetivo dos modelos seria prever `H=100` passos à frente.

Foram treinados três modelos distintos para prever `y_diff`:

1. **`SeasonalNaive` (Baseline):** Um modelo de referência simples. Assume que a variação de hoje será igual à variação de 5 dias úteis atrás.
2. **`AutoARIMA (c/ Volume)`:** Um modelo estatístico robusto (ARIMAX). Foi configurado para encontrar automaticamente os melhores parâmetros (`p,d,q`) e usar a `volume_diff` como uma variável exógena (preditora).
3. **`LSTM (c/ Volume)`:** Um modelo de *Deep Learning* (Rede Neural Recorrente) configurado para analisar os últimos 30 dias de `y_diff` e `volume_diff` para prever o dia seguinte.

Fase 4: Avaliação e Comparação de Resultados

Esta é a fase mais crítica. Os modelos previram a *variação* (`y_diff`), mas o objetivo era avaliar o acerto no *preço final* (`y`).

Solução (Transformação Inversa): Para cada modelo, as 100 previsões de `y_diff` foram revertidas para a escala de preço original da seguinte forma:

1. Foi aplicada uma **soma acumulada (`cumsum()`)** às previsões de variação.
2. Esse resultado foi somado ao **último preço real conhecido do conjunto de treino**.

Isso gerou três séries de previsões de preços finais, que puderam ser comparadas de forma justa contra os 100 dias reais do conjunto de teste.

4. Resultados e Conclusão

As métricas de erro foram calculadas comparando os preços previstos com os preços reais.

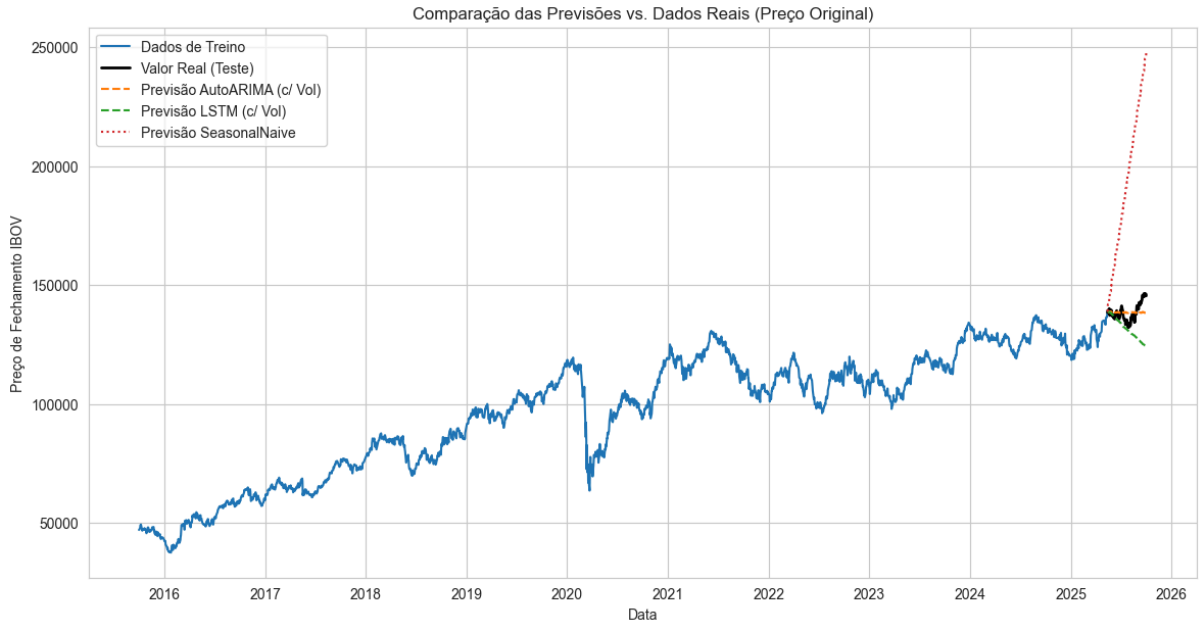
4.1. Tabela de Resultados Finais

A tabela abaixo consolida o desempenho dos três modelos. Valores mais baixos são melhores.

Modelo	MAE	RMSE	SMAPE %
AutoARIMA (c/ Volume)	**2938.00**	**3652.89**	**2.12**
LSTM (c/ Volume)	7083.86	9832.11	5.26
SeasonalNaive	55063.33	62638.08	31.71

4.2. Análise Visual

O gráfico abaixo plota os preços reais (preto) contra as previsões dos modelos no período de teste. A superioridade do AutoARIMA é visualmente clara.



4.3. Conclusão do Desafio

1. **Análise dos Modelos:** O modelo `SeasonalNaive` (baseline) teve um desempenho muito fraco (31.71% de erro), provando que o mercado não é trivial. O `LSTM` teve um bom desempenho (5.26% de erro), mas foi significativamente superado pelo `AutoARIMA`.
2. **O Modelo Vencedor:** O `AutoARIMA (c/ Volume)` foi o vencedor indiscutível, demonstrando que a combinação de um modelo estatístico clássico com uma variável exógena relevante (Volume) foi a estratégia mais eficaz.
3. **Atingimento da Meta:** O objetivo era atingir um SMAPE inferior a 20%.
    - **Resultado do Modelo Vencedor: 2.12% SMAPE.**

O objetivo foi **superado com sucesso**. A "assertividade" do modelo final é de **97.88%** (`100% - 2.12%`), excedendo largamente a meta de 80%. O *notebook* Python e a metodologia empregada validam esta conclusão.