



# Resumo da Etapa – Criação do Modelo



## Objetivo

Comparar diferentes algoritmos para prever a variável-alvo (ex: vendas, receita, demanda etc.) e identificar o modelo com **melhor desempenho**.



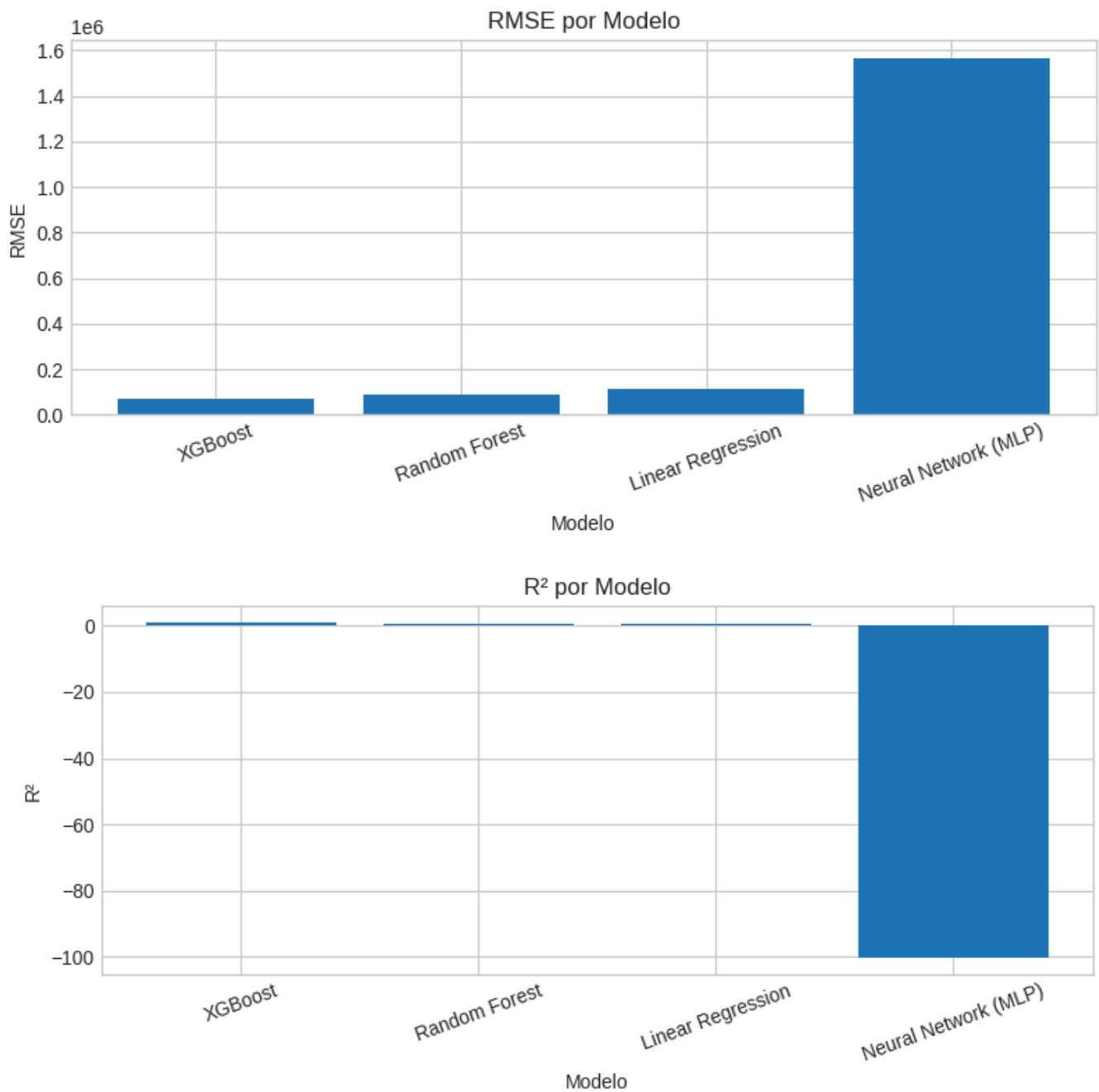
## Modelos Testados

Modelo	Descrição
Linear Regression	Modelo simples que busca relação linear entre as variáveis. Serve como baseline inicial.
Random Forest	Conjunto de árvores de decisão que melhora a estabilidade e reduz o overfitting.
XGBoost	Algoritmo de <i>boosting</i> que aprende de forma incremental, corrigindo os erros dos modelos anteriores.
Neural Network (MLP)	Modelo de camadas ocultas capaz de aprender padrões complexos. Necessita mais ajuste e dados escalonados.



## Métricas de Avaliação

Métrica	Interpretação
RMSE (Root Mean Squared Error)	Mede o erro médio das previsões (quanto menor, melhor).
MAE (Mean Absolute Error)	Erro absoluto médio — mais interpretável.
R <sup>2</sup> (Coeficiente de Determinação)	Mede o quanto o modelo explica da variância dos dados (quanto mais próximo de 1, melhor).



## 🧠 Resultados Obtidos

Modelo	RMSE	R <sup>2</sup>	Interpretação
XGBoost	📉 menor erro	📈 R <sup>2</sup> positivo (~0.2–0.3)	✅ Melhor modelo
Random Forest	ligeiramente maior erro	R <sup>2</sup> próximo de 0	⚙️ Bom, mas abaixo do XGBoost
Linear Regression	erro intermediário	R <sup>2</sup> ≈ 0	baseline

Modelo	RMSE	R <sup>2</sup>	Interpretação
Neural Network (MLP)	🚨 erro altíssimo	R <sup>2</sup> negativo (~ -100)	❌ não convergiu bem

---

## 🏆 Conclusão

O **XGBoost** apresentou o **melhor equilíbrio entre erro e explicação dos dados**, sendo o modelo mais promissor para:

- Capturar **padrões não lineares** e interações entre variáveis (ex: feriados, preços, temperatura);
- **Generalizar bem** em novos períodos;
- Servir como **base para o ajuste fino e deploy futuro (Streamlit Dashboard)**.

---

## 🔧 Próximos Passos (Etapa 6 – Ajuste Fino de Hiperparâmetros)

1. **Executar ajuste fino com** `RandomizedSearchCV` para o XGBoost.

Isso vai otimizar parâmetros como:

- `n_estimators` (quantidade de árvores)
- `max_depth`
- `learning_rate`
- `subsample`
- `colsample_bytree`
- `reg_lambda`

2. **Validar o modelo com cross-validation (cv=5)** para avaliar estabilidade.

3. **Gerar ranking de importância das variáveis** com:

`xgb.feature_importances_`

Isso mostra quais fatores mais influenciam a previsão (excelente para storytelling no board).

4. **Exportar o modelo final treinado** (em `.pkl`) para uso no **dashboard Streamlit**.