Regressão: *Naval Propulsion Plants*

Carolina Dias

Maio de 2022

Conteúdo

Aplicações da Regressão Linear

- Legendre em 1805 e Gauss em 1809: primeira aplicação na **predição de movimentos planetários**.
- Prever **quantidade de itens vendidos de um produto** com informações sobre quantidade de pessoas na loja.
- Encontrar a **dosagem de um medicamento deve ser aplicada em um paciente** com informações sobre seu peso e outros dados fisiológicos.
- Análise da **qualidade de alimentos**, utilizando dados como a qualidade da água utilizada.
- Aproximação de **dados sobre a poluição do ar pela concentração de NO em uma cidade**, com informações sobre a quantidade de carros em cada período durante um dia.

Regressão Linear: Método dos Mínimos Quadrados

O Método dos Mínimos Quadrados consiste em encontrar uma solução aproximada, $\tilde{\mathbf{x}}$, de um sistema inconsistente

$$A\mathbf{x} = \mathbf{b}.$$

Isso é feito projetando o vetor $\mathbf{b}$ no espaço-nulo esquerdo de A e resolvendo as equações normais

$$A^T A\tilde{\mathbf{x}} = A^T\mathbf{b}.$$

Pode-se substituir A por sua decomposição QR, $A = QR$, e obter novas equações normais:

$$R\tilde{\mathbf{x}} = Q^T\mathbf{b},$$

sob determinadas condições em A.

Conjunto de Dados: Indústria Naval

Utilizamos um conjunto de dados relacionado à Fábrica de Propulsões Navais (*Naval Propulsion Plants*). Esses dados foram gerados através de um simulador numérico de turbinas de gás.

- 11.934 medições,
- 16 atributos,
- 2 variáveis dependentes,
- Total: vetor com 18 valores.

Tratamos das variáveis dependentes uma por vez, separadamente. Inicialmente, utilizamos a *GT Compressor decay state coefficient* (chamamos de GTC). Para a outra variável dependente, a *GT Turbine decay state coefficient* (GTT), o processo é análogo.

Conjunto de Dados: Indústria Naval

Queremos encontrar solução para o sistema de equações abaixo:

$$\begin{aligned}
\alpha_1 x_{11} + \alpha_2 x_{12} + \ldots + \alpha_{16} x_{116} + \alpha_{17} &= GTC_1 \\
\alpha_1 x_{21} + \alpha_2 x_{22} + \ldots + \alpha_{16} x_{216} + \alpha_{17} &= GTC_2 \\
&\vdots = \vdots \\
\alpha_1 x_{119341} + \alpha_2 x_{119342} + \ldots + \alpha_{16} x_{1193416} + \alpha_{17} &= GTC_{11934}
\end{aligned}$$

Esse sistema equivale, matricialmente, à

$$\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{116} \\ x_{21} & x_{22} & \cdots & x_{216} \\ \vdots & \vdots & \ddots & \vdots \\ x_{119341} & x_{119342} & \cdots & x_{1193416} \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_{16} \end{bmatrix} = \begin{bmatrix} GTC_1 \\ GTC_2 \\ \vdots \\ GTC_{11934} \end{bmatrix}$$

Experimentos

Requisitos: *Python* 3.8.10 e *Jupyter Notebook*.

- Download e leitura do conjunto de dados, utilizando a biblioteca *Pandas*.
- Criação de funções auxiliares.

Queremos que nosso conjunto de dados possua **n** colunas linearmente independentes, onde $\mathbf{n = posto(A)}$. Vamos calcular o posto da matriz de dados original após a remoção das duas variáveis independentes, ou seja, 16 variáveis restantes. Adicionamos uma coluna composta apenas do número 1 ao final da matriz. Ficamos com um vetor de tamanho 17. Ao calcularmos o posto dessa matriz, utilizando a função do *NumPy linalg.matrix_rank()*, obtemos que $\mathbf{posto(A) = 14}$. Ou seja, existem 3 colunas que são linearmente dependentes nesse conjunto de dados.

Experimentos

Para encontramos essas colunas, podemos realizar a decomposição LU da matriz A, $A = LU$ e encontrar as colunas correspondentes às colunas sem pivôs na matriz U. Mas, nesse caso, isso não é necessário. Ao olharmos para a matriz A, conseguimos observar que existem duas colunas constantes e uma coluna que é repetição de outra. Confirmamos que esse é realmente o caso, com funções específicas do *NumPy* e do *Pandas*, e removemos essas colunas do conjunto de dados.

Agora possuímos um conjunto de dados em forma de matriz com 14 variáveis e, ao conferir o posto dessa matriz, vemos que ele é 14. Isso nos diz que agora todas as colunas são linearmente independentes, e podemos prosseguir para o cálculo da decomposição QR de A, do modo detalhado acima.

Experimentos

- Com a matriz com todas as colunas linearmente independentes, a separamos em dados de treino e teste. Ficamos, assim, com 4 matrizes: *X_train*, *y_train*, *X_test*, *y_test*.
- Finalmente, calculamos a decomposição QR da matriz de treinamento *X_train*, já com a coluna de 1s $[1\ 1\ \ldots\ 1]^T$ adicionada.
- Agora encontramos os coeficientes lineares $[\alpha_1, \alpha_2, \ldots, \alpha_{14}]$ com o comando

```
coefs_lineares =np.linalg.solve(R, np.dot(Q.T,
y_train))
```

Experimentos

- Com os coeficientes lineares podemos calcular os valores de GTC, a primeira variável dependente, para cada um dos vetores medidos, tanto para o conjunto de treino, como para o conjunto de teste.
- Com isso podemos calcular o erro entre essa predição e o valor de fato. Utilizamos, aqui, a métrica da **raiz quadrada do erro médio quadrático** (RMSE), dada por

$$RMSE = \sqrt{\frac{(y_1 - \hat{y}_1)^2 + (y_2 - \hat{y}_2)^2 + \ldots + (y_m - \hat{y}_m)^2}{m}},$$

sendo m a dimensão dos vetores y e y_preds, tanto para treino como para teste. Nesse caso, $m_{treino} = 8353$ e $m_{teste} = 3581$.

Resultados

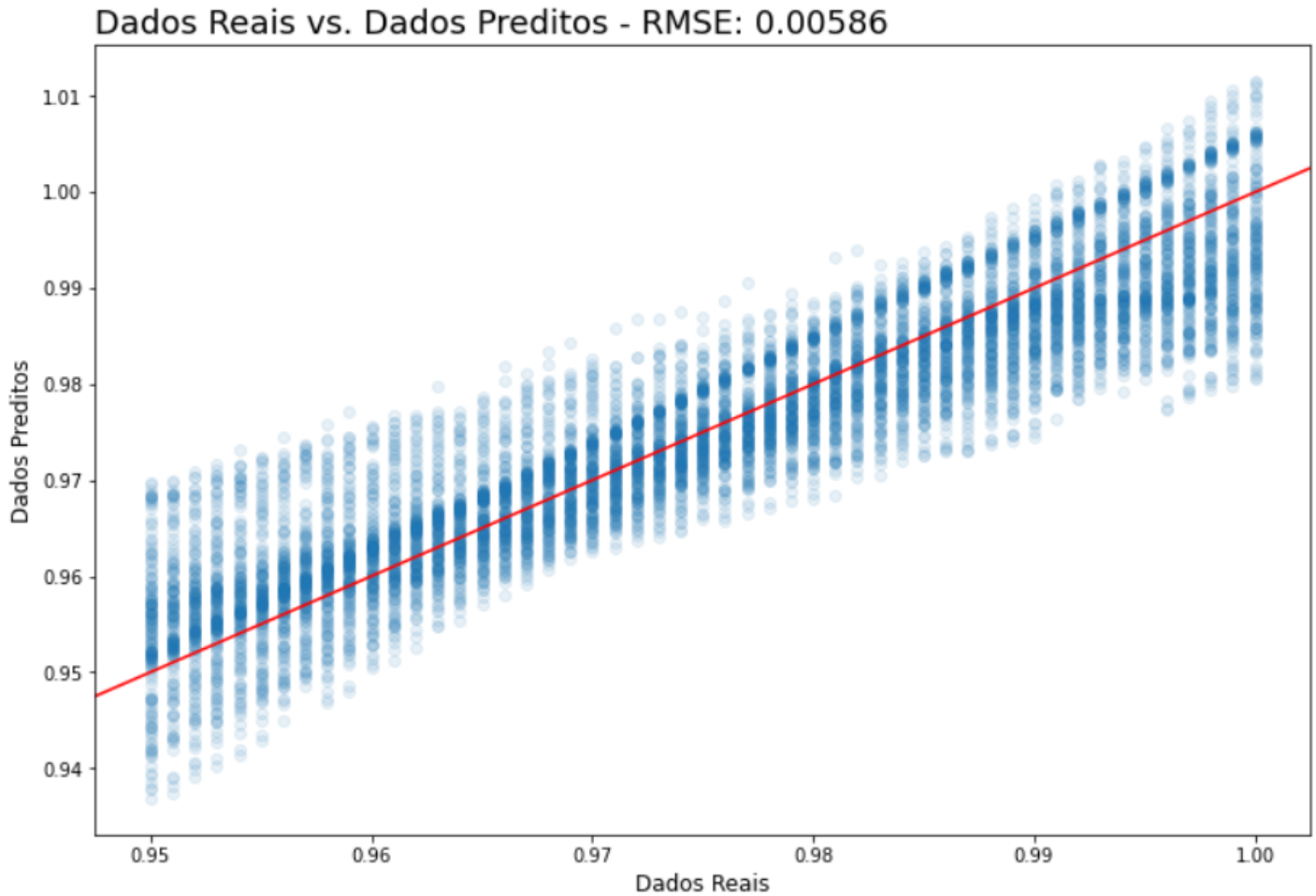

Figura: Resultado da predição de treino para a variável dependente *GTC*.

Resultados

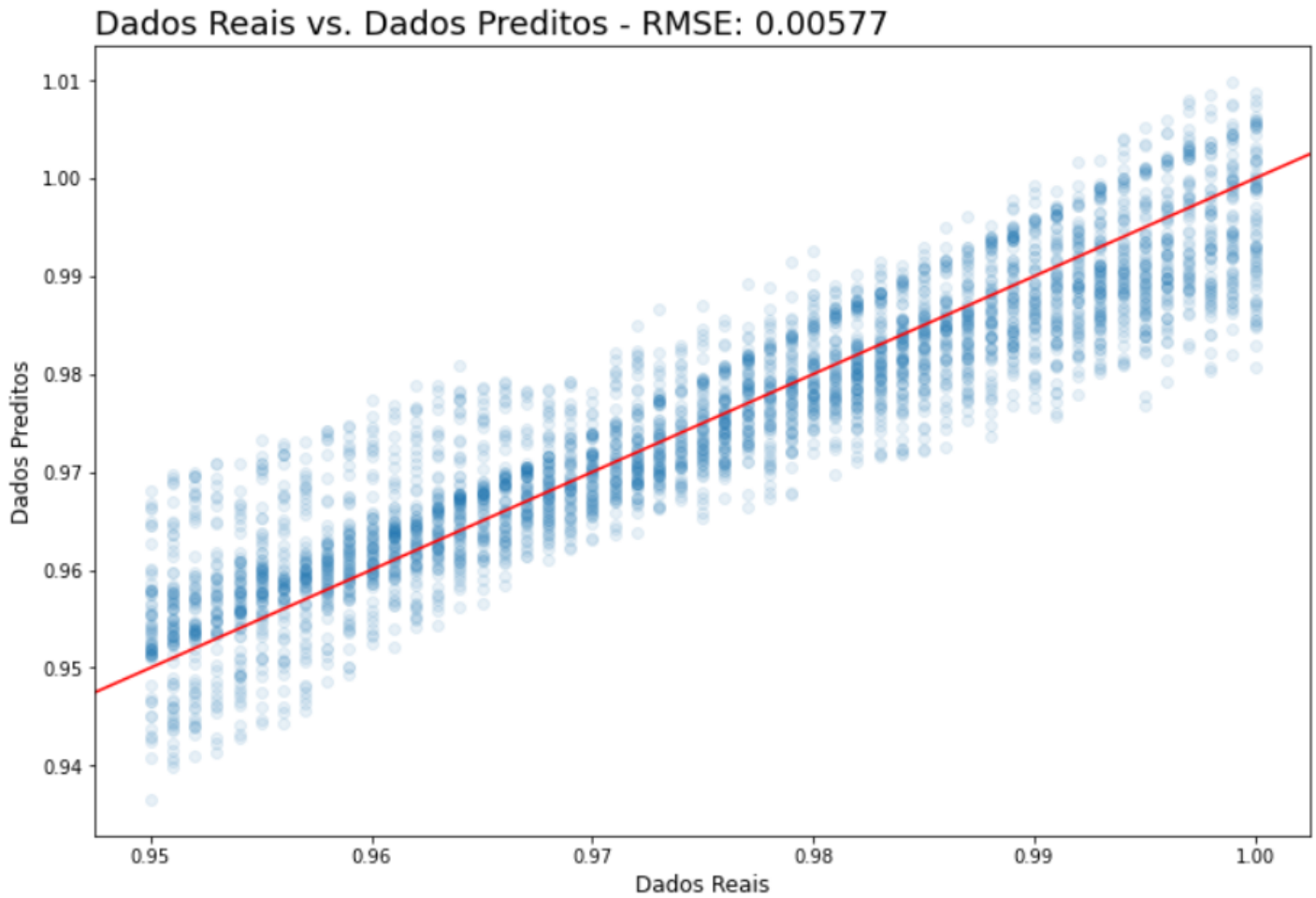

Figura: Resultado da predição de teste para a variável dependente GTC.

Resultados

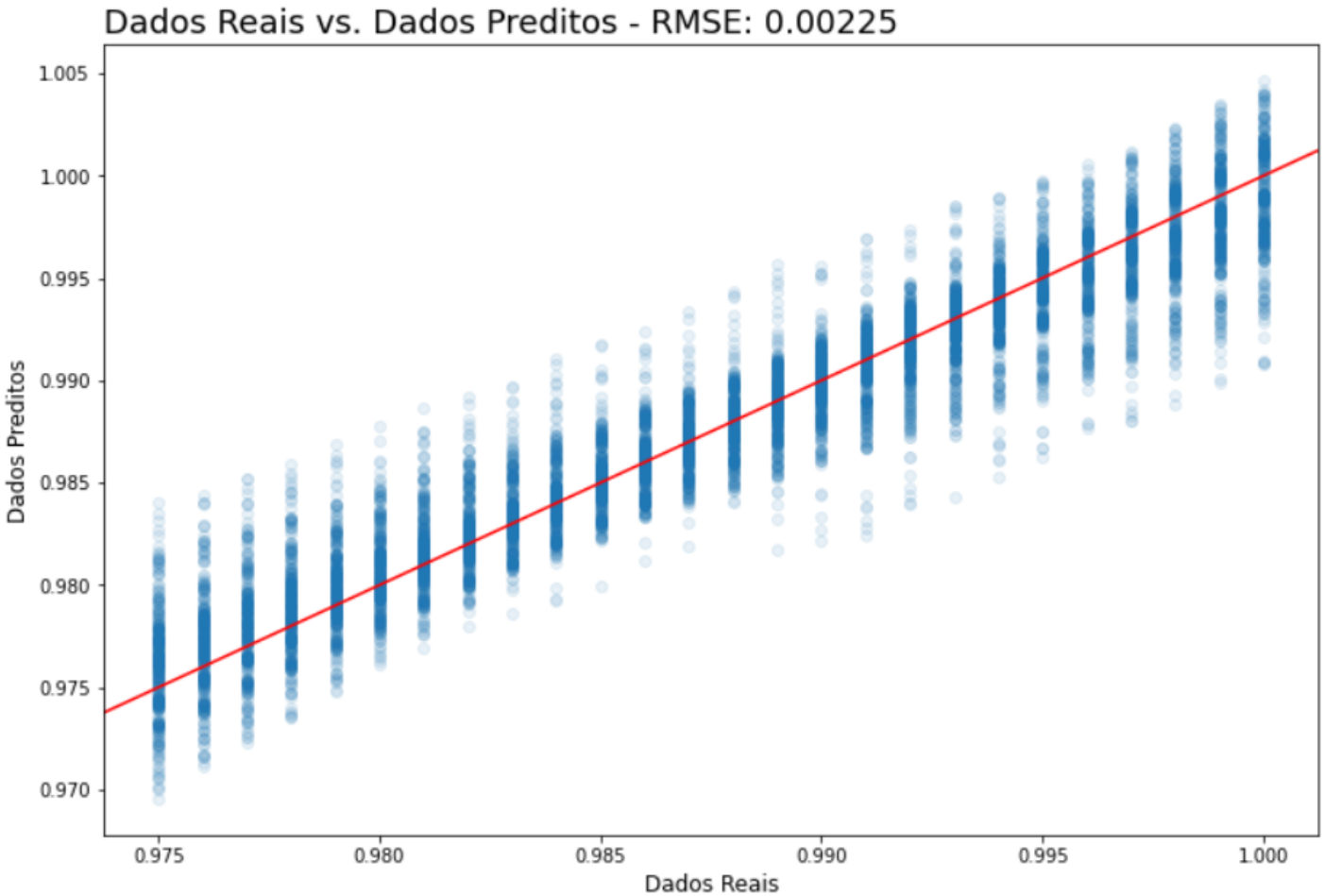

Figura: Resultado da predição de treino para a variável dependente *GTT*.

Resultados

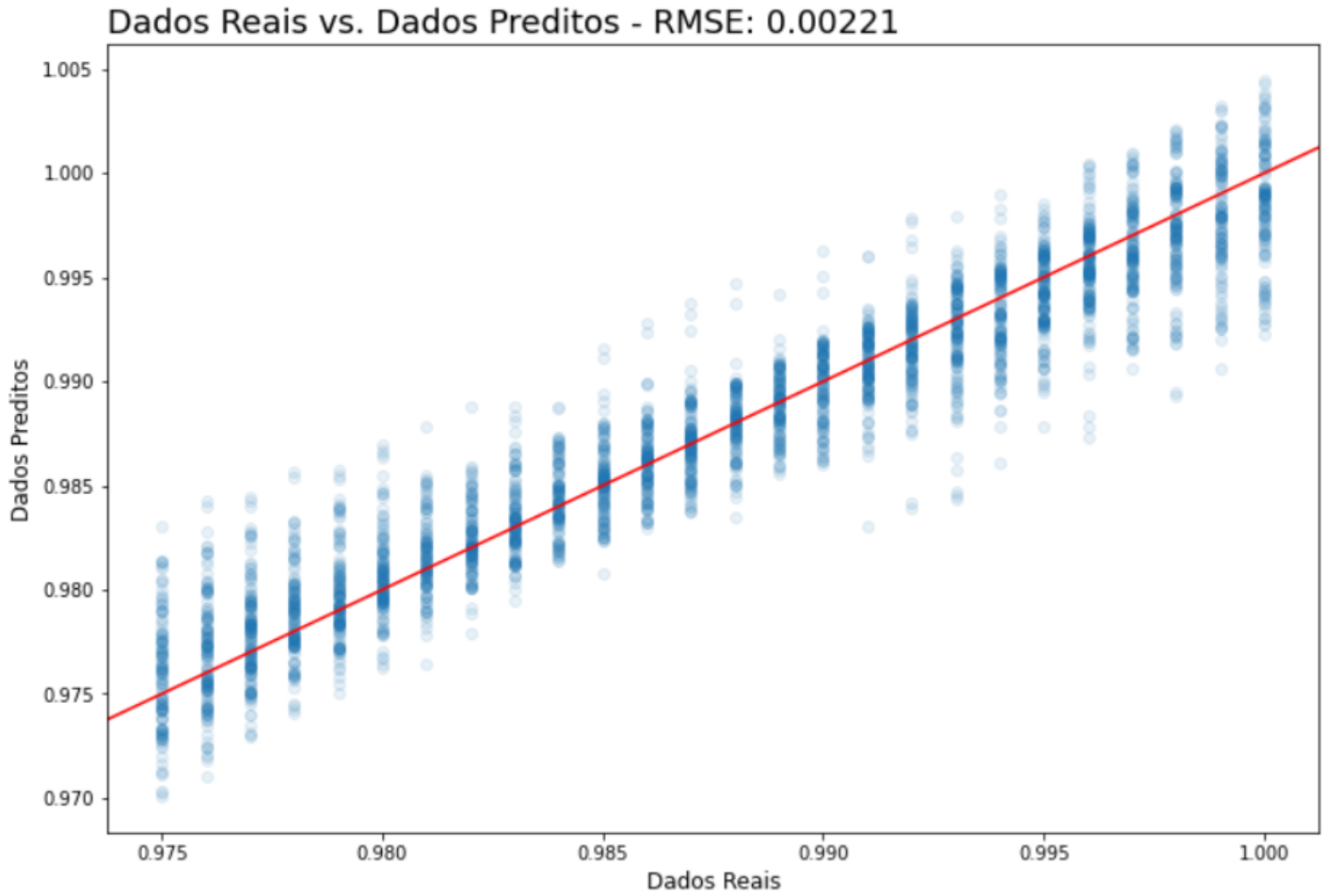

Figura: Resultado da predição de teste para a variável dependente *GTT*.