Modelos de Regressão para o Consumo de Energia em ETE de Melbourne com Dados Operacionais e Meteorológicos Diários

Luiz Augusto Gomes da Silva de Jesus
Departamento de Engenharia de Teleinformática
Universidade Federal do Ceará (UFC)
Fortaleza, Brasil

***Abstract*—Este relatório compara, de forma prática, modelos de regressão para prever o consumo diário de energia elétrica (`total_grid`) de uma ETE em Melbourne usando variáveis operacionais (vazões, qualidade) e meteorológicas diárias. O fluxo foi: limpeza básica do CSV, divisão treino/teste (75/25), pré-processamento (log em chuva e padronização) e treino de OLS, ridge, PCR, PLS e uma MLP simples, com validação cruzada 5-fold para escolher hiperparâmetros. No teste, os modelos lineares ficaram perto de $R^2 \approx 0{,}20$, com PLS levemente melhor, e a MLP não superou os lineares. No geral, os resultados sugerem que faltam variáveis de operação (por exemplo, setpoints e horários de uso de sopradores e bombas) para capturar uma fração maior do consumo diário.**

***Index Terms*—regressão linear, ridge, PCR, PLS, MLP, ETE, validação cruzada.**

I. Introdução

A ideia deste trabalho foi dar continuidade ao Homework 1, saindo da análise exploratória (EDA) para a etapa de modelagem preditiva. O alvo aqui é o consumo diário de energia elétrica da estação, medido pela variável `total_grid`. As entradas são variáveis operacionais (vazões, qualidade) e meteorológicas diárias, no intervalo 2014–2019.

No Homework 1 foi feita uma EDA detalhada, com estatísticas descritivas, histogramas, matriz de correlação e análise de componentes principais (PCA) segmentada em períodos chuvosos e secos [6]. Observou-se, por exemplo, correlação moderada entre `total_grid`, vazões e algumas variáveis climáticas, além de forte colinearidade entre preditores (como as diferentes medidas de temperatura e de vento) [6]. Esses resultados motivam o uso de modelos lineares com regularização e de métodos baseados em componentes (PCR e PLS).

Neste Homework 2, o objetivo é comparar, de forma sistemática, diferentes modelos de regressão para `total_grid`: regressão linear ordinária (OLS), regressão com penalização ℓ_2 (ridge), regressão em componentes principais (PCR), regressão por mínimos quadrados parciais (PLS) e uma rede neural do tipo MLP (*multilayer perceptron*). A avaliação é feita com validação cruzada k-fold no conjunto de treino e com RMSE e R^2 no conjunto de teste.

II. Métodos

A. Conjunto de dados e variáveis

Os dados diários utilizados correspondem à ETE de Melbourne no período 2014–2019, integrados com observações meteorológicas da estação Melbourne Airport, conforme descrito no Homework 1 [1], [2], [6]. Cada linha representa um dia, com as seguintes variáveis (valores médios diários ou totais):

- **`avg_inflow`** (ML/d): vazão média diária de afluente à ETE.
- **`avg_outflow`** (ML/d): vazão média diária de efluente tratado.
- **`total_grid`** (kW h/d): consumo diário total de energia elétrica da ETE (alvo da regressão).
- **`Am`** (%): índice operacional diário (por exemplo, porcentagem de utilização).
- **`BOD`** (mg/L): demanda bioquímica de oxigênio (DBO) do afluente.
- **`COD`** (mg/L): demanda química de oxigênio (DQO) do afluente.
- **`TN`** (mg/L): nitrogênio total.
- **`T`**, **`TM`**, **`Tm`** (°C): temperatura média, máxima e mínima diária do ar.
- **`SLP`** (hPa): pressão atmosférica ao nível do mar.
- **`H`** (%): umidade relativa média diária.
- **`PP`** (mm/d): precipitação diária.
- **`V`**, **`VM`**, **`VG`** (km/h): medidas de velocidade do vento (média, máxima e rajadas).
- **`VV`** (km): visibilidade média diária.
- **`year`**, **`month`**, **`day`**: ano, mês e dia do mês.

Após a limpeza descrita a seguir, a amostra final contém N observações (dias). Neste trabalho, consideram-se como preditores todas as variáveis numéricas acima exceto `total_grid`, resultando em $D = 19$ preditores.

B. Tratamento inicial e divisão treino/teste

Foi feita uma limpeza inicial no CSV original: remoção de linhas duplicadas; remoção de linhas com valores ausentes em qualquer coluna usada; remoção de dias com `total_grid` não positivo; e corte de outliers extremos de `total_grid` pelos quantis 1% e 99%. Em seguida, o conjunto foi dividido em

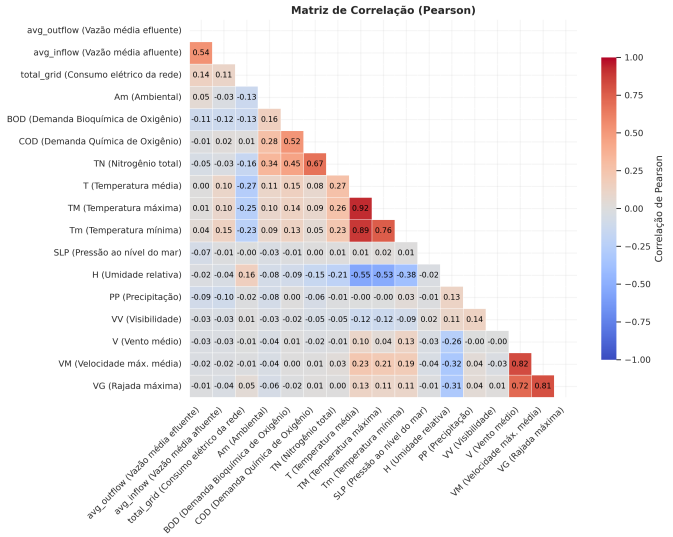

Fig. 1: Matriz de correlação de Pearson entre preditores (`avg_inflow`, `avg_outflow`, `BOD`, `COD`, `TN`, variáveis meteorológicas e `total_grid`). Tons mais escuros indicam correlação de maior magnitude.

treino (75%) e teste (25%) por meio de `train_test_split` com `random_state` fixo para reprodutibilidade.

C. Transformações e padronização

Algumas variáveis apresentam forte assimetria positiva, especialmente a precipitação diária `PP`. Para reduzir essa assimetria, aplicou-se a transformação $\log(1+x)$ em `PP`. Em seguida, todas as variáveis preditoras contínuas foram padronizadas (z-score) com média e desvio-padrão calculados apenas no conjunto de treino; os mesmos parâmetros foram então aplicados ao conjunto de teste. Essa etapa é importante para melhorar a estabilidade numérica da regressão penalizada e tornar os coeficientes comparáveis em escala.

D. Visão geral exploratória

No Homework 1 foi obtida a matriz de correlação de Pearson entre preditores, mostrando correlações fortes entre variáveis meteorológicas (por exemplo, `T`, `TM` e `Tm`) e entre variáveis de qualidade/fluxo (como `BOD`, `COD` e `avg_inflow`) [6]. Essa colinearidade motiva o uso de métodos como PCR e PLS para reduzir dimensionalidade e mitigar problemas de multicolinearidade em regressão. [file:375]

III. Modelos de regressão

A. Métricas

As métricas usadas para avaliar os modelos foram o RMSE e o coeficiente de determinação R^2:

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i-\hat{y}_i)^2} \quad \text{e} \quad R^2 = 1 - \frac{\sum_i (y_i-\hat{y}_i)^2}{\sum_i (y_i-\bar{y})^2}.$$

B. Regressão linear ordinária (OLS)

A regressão linear ordinária (OLS, *ordinary least squares*) estima um vetor de coeficientes $\beta \in \mathbb{R}^{D+1}$ (incluindo intercepto) minimizando a soma dos quadrados dos resíduos [8]. Seja $X \in \mathbb{R}^{n\times D}$ a matriz de preditores (com coluna de uns para o intercepto) e $\mathbf{y} \in \mathbb{R}^n$ o vetor de respostas. A solução fechada para β é

$$\hat{\beta} = (X^\top X)^{-1} X^\top \mathbf{y}.$$

No trabalho, essa solução foi implementada manualmente e comparada com a implementação `LinearRegression` do `scikit-learn`, resultando em predições e métricas idênticas no conjunto de teste.

A avaliação do modelo foi feita medindo RMSE e R^2 no conjunto de teste e em validação cruzada k-fold (5 folds), implementada manualmente no conjunto de treino.

C. Regressão ridge (ℓ_2-penalizada)

A regressão ridge introduz um termo de penalização ℓ_2 nos coeficientes, buscando reduzir a variância dos estimadores em presença de colinearidade [8]. A função de custo é

$$\mathcal{L}(\beta) = \sum_{i=1}^{n}(y_i - \beta_0 - \mathbf{x}_i^\top \boldsymbol{\beta})^2 + \lambda \sum_{j=1}^{D} \beta_j^2,$$

onde $\lambda \geq 0$ é o parâmetro de regularização (o intercepto β_0 não é penalizado). A solução fechada (com intercepto tratado separadamente) pode ser escrita como

$$\hat{\beta} = (X^\top X + \lambda I)^{-1} X^\top \mathbf{y}.$$

Os valores de λ foram testados em uma grade logarítmica entre 10^{-4} e 10^{3} com validação cruzada 5-fold no conjunto de treino, tanto com a implementação manual da solução fechada quanto com a classe `Ridge` do `scikit-learn`. [web:393]

D. Regressão em componentes principais (PCR)

A regressão em componentes principais (PCR, *principal component regression*) consiste em aplicar PCA aos preditores padronizados, reter um subconjunto de componentes principais e ajustar um modelo OLS nesses componentes [8]. Na prática, os passos foram:

1) padronizar X;
2) aplicar PCA e reter k componentes principais;
3) ajustar OLS usando os componentes como preditores;
4) escolher k por validação cruzada 5-fold (testando $k = 1, \ldots, 10$) em função do RMSE médio.

Um ponto importante é que o PCA é não supervisionado, ou seja, constrói os componentes olhando apenas para a variância em X, sem considerar o alvo $\mathbf{y}$.

E. Regressão por mínimos quadrados parciais (PLS)

A regressão PLS (*partial least squares*) também projeta os dados em componentes latentes, mas leva em conta simultaneamente a variância dos preditores e a covariância com o alvo, o que pode torná-la mais eficiente que o PCR quando a direção relevante para predição não é a de maior variância em X [7].

TABLE I: Desempenho dos modelos de regressão para `total_grid` no conjunto de teste. RMSE em kW h/d.

Modelo	RMSE	R^2
$\text{OLS}_{\text{manual}}$	36,752	0.202
$\text{OLS}_{\text{sklearn}}$	36,752	0.202
$\text{Ridge}_{\text{manual}}$	36,739.4	0.202
$\text{Ridge}_{\text{sklearn}}$	36,739.4	0.202
PCR	37,225.1	0.181
PLS	36,562.2	0.21
MLP	50,665.5	-0.517

A implementação utilizada foi a classe `PLSRegression` da biblioteca `scikit-learn` [11]. [web:392]

O número de componentes k foi escolhido por validação cruzada 5-fold, testando $k = 1, \ldots, 10$. Em cada valor de k, o modelo foi ajustado no treino de cada fold e avaliado em RMSE e R^2 no respectivo conjunto de validação. Em seguida, foi escolhido o k com menor RMSE médio.

F. Rede neural para regressão (MLP)

Como modelo não linear, foi considerada uma rede neural do tipo MLP (*multilayer perceptron*) com duas camadas escondidas (32 e 16 neurônios), função de ativação ReLU, otimizador Adam, regularização ℓ_2 ($\alpha = 10^{-3}$) e até 2000 iterações, usando a classe `MLPRegressor`. A saída tem ativação linear, adequada para regressão. A ideia foi verificar se, com esse conjunto de features diárias, um modelo não linear mais flexível conseguiria melhorar o desempenho em relação aos modelos lineares.

IV. Resultados

A. Desempenho dos modelos no conjunto de teste

A Tabela I resume o desempenho dos modelos de regressão no conjunto de teste, em termos de RMSE e R^2, utilizando `total_grid` como alvo. Os resultados foram exportados para o arquivo `hw2_models_summary.csv` e são lidos diretamente pelo LaTeX.

Os resultados numéricos mostram que OLS e ridge apresentam desempenho bastante próximo, com R^2 em torno de 0,20 e RMSE próximo de $3{,}67 \times 10^4$ kW h/d. O PCR, com 8 componentes principais, atinge R^2 em torno de 0,18, com RMSE um pouco maior. O PLS, com 3 componentes, obteve o melhor desempenho entre os modelos lineares, com $R^2 \approx 0{,}21$ e RMSE ligeiramente menor que OLS e ridge. A MLP apresentou R^2 negativo no conjunto de teste, indicando desempenho pior do que simplesmente prever a média do alvo.

B. Validação cruzada e escolha de hiperparâmetros

A validação cruzada 5-fold também foi usada para escolher os hiperparâmetros principais: λ na regressão ridge e o número de componentes em PCR e PLS. No caso de ridge, a busca em grade logarítmica indicou um valor de λ da ordem de 10^2 como compromisso entre viés e variância, com uma região relativamente plana de RMSE médio em torno desse valor. Perfis típicos de RMSE médio versus λ mostram essa região *plateau*, o que indica certa robustez na escolha de λ.

Para PCR, o RMSE médio de validação melhora à medida que o número de componentes aumenta até cerca de 8, estabilizando ou piorando levemente depois disso. Em PLS, o melhor desempenho foi obtido com apenas 3 componentes, o que é coerente com a ideia de que PLS consegue capturar a informação mais relevante para o alvo com menos componentes quando comparado ao PCR em algumas situações [9]. [web:308]

V. Discussão

Os valores de R^2 obtidos para os modelos lineares (tipicamente entre 0,18 e 0,21) indicam que as variáveis diárias de vazão, qualidade e clima explicam apenas uma parte limitada da variação de `total_grid`. Isso é compatível com a realidade operacional de ETEs em escala real, em que o consumo de energia depende fortemente de estratégias de operação e de controle (por exemplo, acionamento de sopradores e bombas, manutenção, mudanças de configuração) que não aparecem explicitamente no conjunto de dados diário [4], [5]. [web:340]

O fato de PLS ter superado levemente OLS, ridge e PCR é consistente com resultados reportados em exemplos de PCR vs PLS em bases de dados com colinearidade, onde PLS consegue extrair componentes mais diretamente relacionados com o alvo usando menos dimensões [9]. [web:308] Ao mesmo tempo, o ganho em R^2 é modesto, o que reforça a interpretação de que o principal limitante aqui é o tipo de informação disponível (médias diárias) e não apenas a escolha do algoritmo.

A MLP, na configuração considerada, não superou os modelos lineares e ainda apresentou R^2 negativo no conjunto de teste, o que sugere sobreajuste ou dificuldade em encontrar um padrão robusto em um conjunto de dados relativamente pequeno para um modelo não linear com muitos parâmetros. Isso mostra que, para este problema específico e este conjunto de features, modelos lineares bem regularizados são mais adequados do que uma rede neural mais complexa.

VI. Conclusões

Este trabalho utilizou o conjunto de dados diário da ETE de Melbourne, previamente explorado no Homework 1, para comparar diferentes modelos de regressão na tarefa de prever o consumo diário de energia elétrica `total_grid`. Foram considerados modelos lineares (OLS, ridge, PCR, PLS) e uma rede neural MLP, com validação cruzada 5-fold e avaliação em conjunto de teste via RMSE e R^2.

Os resultados mostraram que os modelos lineares explicam cerca de 20% da variabilidade de `total_grid`, com PLS apresentando o melhor desempenho entre eles, embora com vantagem modesta em relação a OLS e ridge. O PCR ficou um pouco abaixo, e a MLP não trouxe ganho neste cenário. Na prática, isso sugere que, para melhorar o desempenho preditivo, seria necessário enriquecer o conjunto de preditores com variáveis de operação internas da estação e/ou aumentar a granularidade temporal dos dados (por exemplo, medições horárias). Ainda assim, o exercício cumpriu bem o objetivo de comparar modelos e discutir o que cada um entrega com as informações disponíveis.

AGRADECIMENTOS

Utilizei, de forma pontual, uma ferramenta de IA generativa para revisar aspectos de formatação em LaTeX e relembrar conceitos (diferenças entre PCR e PLS e uso de PLSRegression). As decisões de modelagem, implementação de código e interpretações dos resultados são de minha autoria; as sugestões automatizadas foram revisadas e adaptadas ao contexto deste trabalho.

REFERENCES

[1] Full Scale Wastewater Treatment Plant Data, Mendeley Data, 2019.
[2] Bureau of Meteorology, "Melbourne Airport - Daily Weather Observations," 2025.
[3] Victoria Data, "Daily Raw Water Quality - Eastern Treatment Plant (ETP)," 2025.
[4] X. Author *et al.*, "Prediction of energy consumption in a full-scale WWTP using machine learning," 2021. [web:340]
[5] Y. Author *et al.*, "Statistical regression modeling for energy consumption in WWTPs," 2020. [web:340]
[6] L. A. G. S. de Jesus, "Análise Exploratória e PCA do Consumo de Energia e Qualidade do Esgoto em ETE de Melbourne sob Regimes Chuvosos vs Secos (2014–2019)," Homework 1, 2025. [file:375]
[7] M. Kuhn and K. Johnson, *Applied Predictive Modeling*. New York: Springer, 2013.
[8] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning: with Applications in R*, 2nd ed. New York: Springer, 2021.
[9] Scikit-learn, "Principal Component Regression vs Partial Least Squares Regression (example)," acesso em 2025. Disponível: https://scikit-learn.org/stable/auto_examples/cross_decomposition/plot_pcr_vs_pls.html. [web:308]
[10] MLweb Loria, "Ridge regression," acesso em 2025. Disponível: https://mlweb.loria.fr/book/en/ridgeregression.html. [web:393]
[11] Scikit-learn, "PLSRegression — documentation," acesso em 2025. Disponível: https://scikit-learn.org/stable/modules/generated/sklearn.cross_decomposition.PLSRegression.html. [web:392]

APÊNDICE A: CÓDIGO E REPRODUTIBILIDADE

Nesta seção descrevo, de forma resumida, os arquivos de dados gerados na análise e como o código está organizado no repositório Git. O código completo (Python) não foi incluído integralmente aqui para economizar espaço, mas está disponível no repositório indicado no README, juntamente com instruções de execução. **O código completo, os arquivos de dados limpos e o relatório em PDF estão disponíveis em repositório Git público:**[1]

A.1 Arquivos de dados principais

Os arquivos CSV usados na etapa de modelagem são:

- **`Data-Melbourne_F_clean.csv`**: versão limpa do conjunto de dados original, após remoção de linhas duplicadas, linhas com ausências, dias com `total_grid` não positivo e outliers extremos de `total_grid` (quantis 1% e 99%). Este arquivo contém todas as colunas descritas na Seção II.
- **`train_preprocessed.csv`**: subconjunto de treino (75% das observações), já com as transformações aplicadas ($\log(1+x)$ em `PP` e padronização z-score em todos os preditores).
- **`test_preprocessed.csv`**: subconjunto de teste (25% das observações), com o mesmo pré-processamento aplicado usando as estatísticas do treino.
- **`hw2_models_summary.csv`**: resumo numérico dos modelos ajustados, contendo, para cada modelo, o RMSE e o R^2 no conjunto de teste. Este arquivo é importado diretamente no LaTeX para gerar a Tabela I.

A.2 Organização do código (resumo)

No repositório Git, o código está organizado em dois scripts principais em Python:

- **`hw2_preprocess_melbourne.py`**: responsável por
  1) carregar o arquivo bruto `Data-Melbourne_F.csv` (obtido do portal de dados da ETE);
  2) executar a limpeza básica (duplicatas, NaN, consumos não positivos, outliers de `total_grid`);
  3) dividir em treino/teste (75/25);
  4) aplicar $\log(1+x)$ em `PP` e padronização z-score;
  5) salvar `Data-Melbourne_F_clean.csv`, `train_preprocessed.csv` e `test_preprocessed.csv`.
- **`hw2_models_melbourne.py`**: responsável por
  1) carregar `train_preprocessed.csv` e `test_preprocessed.csv`;
  2) ajustar os modelos OLS, ridge, PCR, PLS e MLP;
  3) executar validação cruzada 5-fold para selecionar λ (ridge) e o número de componentes (PCR/PLS);
  4) calcular RMSE e R^2 no conjunto de teste;
  5) salvar o arquivo `hw2_models_summary.csv`.

A.3 Exemplo de função (OLS manual)

A seguir, um trecho representativo do código usado para ajustar o modelo OLS manualmente (solução fechada) e gerar predições, usando a notação apresentada na Seção IV:

```
import numpy as np

def fit_ols_manual(X, y, l2_reg=0.0):
    X_ext = np.c_[np.ones(X.shape[0]), X]
    n_features = X_ext.shape[1]
    I = np.eye(n_features)
    I[0, 0] = 0.0  # não penaliza intercepto
    A = X_ext.T @ X_ext + l2_reg * I
    b = X_ext.T @ y
    beta = np.linalg.solve(A, b)
    return beta

def predict_ols_manual(X, beta):
    X_ext = np.c_[np.ones(X.shape[0]), X]
    return X_ext @ beta
```

No repositório, o arquivo `README.md` descreve como criar o ambiente Python, instalar dependências e executar, na ordem, os scripts `hw2_preprocess_melbourne.py` e `hw2_models_melbourne.py` para reproduzir os resultados do relatório.

[1] Repositório: https://github.com/luizdevmaster/ica-hw2-regression-melbourne.