Aplicação de Modelos de Regressão Linear e Rede Neural sobre Dados de Solubilidade

1st Cícero Oliveira da Silva Júnior
Departamento de Engenharia Teleinformática
Universidade Federal do Ceará
Fortaleza, Brasil
junioroliver@alu.ufc.br

2nd David Lima dos Santos
Departamento de Engenharia Teleinformática
Universidade Federal do Ceará
Fortaleza, Brasil
david.santos@alu.ufc.br

3rd Gabriel Moreira de Andrade
Departamento de Engenharia Teleinformática
Universidade Federal do Ceará
Fortaleza, Brasil
gabrielm03@alu.ufc.br

4th Igor Pereira Gouveia
Departamento de Engenharia Teleinformática
Universidade Federal do Ceará
Fortaleza, Brasil
igorpereiragv@alu.ufc.br

***Abstract*—O presente trabalho apresenta a aplicação de diferentes modelos de regressão linear e de rede neural sobre um mesmo conjunto de dados a fim de comparar seus resultados. Para tal, utiliza-se um conjunto de dados sobre solubilidade com 228 váriaveis preditoras e 1267 observações. Os modelos utilizados são: regressão linear ordinária, regressão linear penalizada, regressão parcial de mínimos quadrados e rede neural. Utiliza-se ainda a transformação de Box-Cox para reduzir a assimetria da distribuição dos dados e a técnica "cross-validation" para análise de performance. São obtidas e analisadas métricas para os resultados de implementações dos modelos citados.**

***Index Terms*—regression, linear regression, least squares regression, ridge regression, neural network**

I. Introdução

O principal objetivo da estatística inferencial é construir modelos que, a partir, de *datasets* modelos, consigam realizar predições quando sujeitos a outro conjunto de dados. Entretanto, após um processo de análise exploratória do conjunto de dados que desejamos estudar, é essencial que tratemos-os para reduzir a influência de características que prejudiquem a acurácia do modelo. Assim, é necessário um pré-processamento, pois ele é responsável por escalonar, centralizar e tratar as assimetrias do conjunto de dados antes de aplicar nos modelos. Dentre os modelos, temos os lineares e os não lineares.

Os modelos de regressão linear buscam parâmetros que atribuem diferentes coeficientes aos valores dos preditores para minimizar o erro quadrático médio.

As redes neurais são modelos não-lineares inspiradas sobre teorias de como o cérebro funciona. A saída é modelada por um conjunto intermediário de variáveis não observadas.

É importante no processo de modelagem preditiva analisar diferentes modelos aplicados e comparar seus resultados, a fim de escolher o modelo que seja mais apropriado ao problema.

É importante ainda considerar técnicas de validação, como validação cruzada de k-dobras e transformações dos dados para reduzir assimetria, ou para centralizar e escalar os dados antes de uma análise de componentes principais.

II. Metodologia

A. Análise exploratória e pré-processamento dos dados

O conjunto de dados sobre solubilidade [1] contém D = 228 variáveis preditoras (208 "fingerprints" binárias que indicam a presença ou falta de uma subestrutura química específica, 16 descritores de contagem - como o número de ligações ou o número de átomos de bromo - e 4 descritores contínuos - como peso molecular ou área de superfície) e N = 1267 observações.

A comparação entre as métricas é importante para decidir o modelo adequado a se utilizar, dadas as características do conjunto de dados e as do modelo. Os fatores que levam à escolha de um modelo podem ser variados, como performance, acurácia, quantidade de dados, dentre outros.

Dentre as técnicas de pré-processamento, optamos por aplicar a transformação de Box-Cox, método estatístico apresentando em 1964 por Box e Cox, utilizado neste trabalho para reduzir a assimetria da distribuição dos dados. Refere-se à seguinte família de transformações indexadas pelo parâmetro λ:

$$x^* = \begin{cases} \frac{x^\lambda - 1}{\lambda} & \text{se } \lambda \neq 0 \\ \log(x) & \text{se } \lambda = 0 \end{cases} \quad (1)$$

Para utilizar tal técnica é necessário que os valores dos preditores sejam estritamente positivos. A estimação por máxima verossimilhança é usada para encontrar um valor adequado para λ.

Como trataremos de um dataset que possui amostras de entrada e saída, usaremos algoritmos de uma aprendizagem supervisionada. Em termos científicos, um processo de modelagem de dados supervisionado consiste em estimar uma função (f) que descreva a relação entre um conjunto de dados de entrada ($X = x_1, ..., x_D$), o qual contém D variáveis preditores cada uma com n observações, com um conjunto que representa a saída ($Y = y_1, ..., y_D$). O principal objetivo consiste em inferir um modelo cujo erro, isto é, a diferença entre o valor

do conjunto de saída (y_i) e o valor predito pelo modelo ($\hat{y}_i$), seja mínimo, como mostra a equação 2.

$$e_1 = y_i - \hat{y}_i \tag{2}$$

B. Métricas de avaliação da qualidade do modelo

Com o objetivo de servir de parâmetro de avaliação para os métodos que aplicarmos no conjunto de dados, usaremos o erro quadrático médio (3) (*Root Mean squared error*, em inglês). Ele nos fornece a distância média entre o valor observado e a predição do modelo [2].

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\epsilon_i)^2} \tag{3}$$

Outro parâmetro foi o Coeficiente de determinação (4), que pode ser interpretado como a proporção da informação do *dataset* que é explicada pelo modelo [2]. Mesmo não sendo uma medida de acurácia, ele fornece o quanto que a variância em Y é afetada por X.

$$R^2 = \frac{\sum(y_i^2 - \bar{y})^2 - \sum_{i=1}^{n}\epsilon_i^2}{\sum_{i=1}^{n}\epsilon_i^2} \quad \therefore \quad R^2 = \frac{TSS - RSS}{TSS} \tag{4}$$

onde

TSS : Soma total dos quadrados (*total sum of squares*, em inglês)

RSS : Soma dos quadrados residuais (*residual sum of squares*, em inglês)

Além dessas métricas, as técnicas de reamostragem são aplicadas no processo de modelagem para garantir que bons parâmetros de modelos sejam gerados. O método escolhido para este artigo foi a validação cruzada (*K-Fold cross validation*, em inglês), a qual consiste em dividir o *dataset* em *k* subamostras (ou dobras), separando uma para ser a amostra de teste, enquanto que as outras são amostras de treinamento. A primeira é usada como um conjunto de validação, e as demais servem para serem aplicadas ao método durante o treinamento do modelo [2]. Ademais, cada um dessas dobras serão a amostra teste durante uma execução do processo, o qual se repetirá *k* vezes. Com isso, cerca de *k* métricas $RMSE$ e R^2 serão geradas e a média de cada uma delas servirão para a análise da acurácia do modelo. O nosso *dataset* em questão possui um *train set* de N = 951, e um *test set* de N = 316.

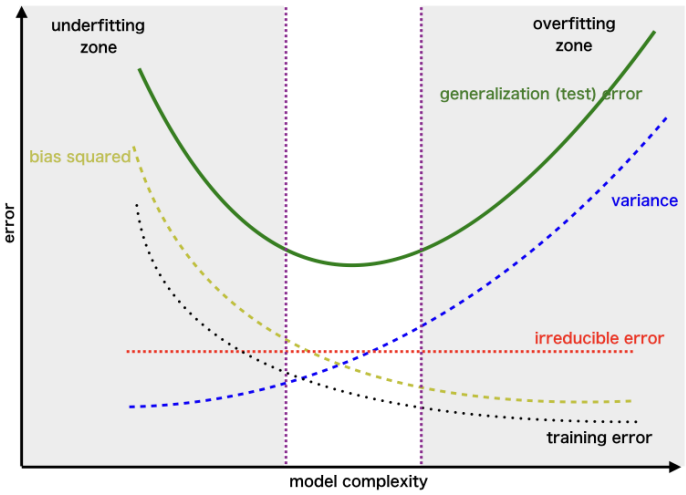
Em posse dessas métricas, conseguimos mensurar uma forma de alcançar um modelo que obtenha um resultado favorável na relação de *trade-off* entre o viés e a variância (*Bias-Variance trade-off*, em inglês), como mostra a equação 5.

$$E[MSE] = \sigma^2 + (Bias)^2 + Var(\epsilon) \tag{5}$$

Para que o valor esperado do MSE seja alcançado, o Bias e a variância devem ser mínimos, uma vez que, de qualquer maneira, o modelo sempre terá um erro irredutível (σ^2). O Viés ou *Bias* corresponde a uma medida do quanto o modelo está da verdadeira relação entre X e Y, enquanto que a variância mede o grau de espalhamento dos dados [2]. Modelos com um elevado *Bias* tendem a ser mais simples, enquanto que aqueles com alta variância possum uma complexidade maior. Entretanto, ambos tendem a ser prejudicial para o modelo quando um ou outro são muito elevados. Enquanto que o primeiro, causa *underfitting*, isto é, quando o modelo não consegue capturar a complexidade dos dados de maneira satisfatória, o segundo causa *overfitting*, que, por sua vez é quando o modelo se ajusta muito ao conjunto de treinamento de modo que ele falha em generalizar para novos dados [6].

Na Figura 1, temos uma representação bem explicada do *trade-off*. Nela, podemos observar que o ideal é encontrar um ponto de equilíbrio a depender do contexto da aplicação ao qual o modelo será usado. O principal é tentar reduzir o erro de generalização. Por isso, o tratamento da colinearidade dos dados, é essencial, uma vez que é de extrema importância que o modelo consiga reconhecer novos padrões, sem cair na generalização para novos dados [6].

Fig. 1. Trade-off entre Viés e variância



C. Regressão Linear Ordinária

O modelo de regressão linear ordinária 6 consiste em tentar predizer a variável em Y a partir de uma combinação linear das variáveis em X [3]. Dessa forma, temos:

$$Y = \beta_0 + \beta_1 x_1 + \beta_1 x_2 + ... + \beta_D x_D + \epsilon \tag{6}$$

Os parâmetros β_i representam a relação o efeito que a variável x_i possui na variável Y, considerando os outros preditores fixos. Para que um bom modelo seja gerado, é essencial que as variáveis utilizadas estejam tratadas, com colinearidade reduzida, relação linear com a variável Y, e sejam importantes para predir a variação em Y [4]. Em vista disso, para estimar esses parâmetros, usamos o método dos mínimos quadrados (LSM, *Least Square Method*, em inglês), o qual, por sua vez, consiste em minimizar a soma dos quadrados

residuais (RSS), objetivando obter um bom custo benefício entre *bias* e a variância.

$$\hat{\beta} = \text{argmin}\left\{ \frac{1}{n} \cdot \sum_{i=1}^{n} L(\hat{y}(x_i, \beta), y_i) \right\} \tag{7}$$

onde

$L(\hat{y}(x_i, \beta), y_i)$: Função de custo

A função de custo é uma medida única e geral que determina a perda gerada pela ação do modelo [5]. No nosso contexto, ela é soma dos resíduos quadrados. Dessa forma, podemos concluir que $\hat{\beta}$ corresponde em minimizar o RMSE, uma vez que a função de custo é dividida por n. Em um modelo matricial, os parâmetros são obtidos pela equação 8.

$$\hat{\beta} = (X^T X)^{-1} X^T Y \tag{8}$$

Em suma, após calcularmos os parâmetros que moldam o modelo usando o código próprio do *python* que segue a equação 8, usamos a amostra de validação, que foi separada anteriormente, para predizer um um vetor resposta $\hat{y}$, para, então, as métricas de avaliação (raiz do erro quadrático médio e coeficiente de correlação) tendo como ponto de comparação os valores de validação de y. Esse procedimento é repetido cerca de k vezes, por meio de um *loop for* no código elaborado por nós mesmos que obedece a técnica do CV.

D. Regressão Linear L2-Penalizada (Ridge)

A regressão Ridge (L2-penalizada) é uma técnica de regressão linear que aplica uma penalidade à soma dos erros quadráticos para melhorar seu desempenho preditivo. Segue a fórmula:

$$\text{SSE}_{L2} = \sum_{i=1}^{n} e_i^2 + \lambda \sum_{j=1}^{P} \beta_j^2, \tag{9}$$

onde:

- $e_i = y_i - \hat{y}_i$ diferença entre os valores reais (y_i) e os valores preditos ($\hat{y}_i$);
- λ é o parâmetro de regularização que controla a intensidade da penalização aplicada aos coeficientes β_j;
- β_j são os coeficientes de regressão associados aos preditores.

Adicionando essa penalidade na soma dos quadrados dos erros, o modelo é forçado a reduzir os valores absolutos dos coeficientes, minimizando o impacto de preditores altamente correlacionados, o que reduz a multicolinearidade. Entretanto, ao que pode, eventualmente, acontecer um viés, já que os coeficientes são ajustados para minimizar tanto os erros quanto a magnitude dos coeficientes [9]. No código, assim como fizemos na regressão linear ordinária, usamos o *dataset* transformado em nosso pré-processamento. Em seguida, aplicamos nosso próprio código que simula, na forma matricial, a estimação dos parâmetros e do λ a partir da equação 9. A expressão é simular a mostrada na equação 8, sendo diferente por ter o termo de ridge somado dentro do primeiro termo da multiplicação.

E. Mínimos Quadrados Parciais

O método dos Mínimos Quadrados Parciais (assim como o método de Regressão por Componentes Principais) utiliza análise de componentes principais para gerar combinações lineares não correlacionadas entre si dos preditores. Esse método pode ser recomendado quando o grau de correlação entre preditores é grande ou o número de observações é menor do que a quantidade de preditores, com regressões lineares ordinárias podendo ser instáveis nesses casos. A quantidade de dimensões retidas pelo método dos Mínimos Quadrados Parciais é direcionada a otimizar a correlação com a saída.

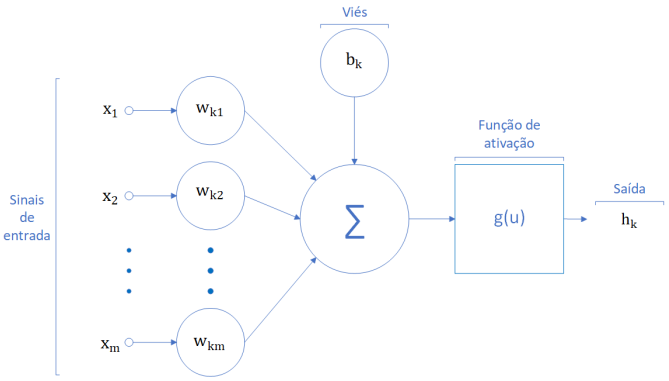
Para utilizar o método, primeiro transformou-se os dados centralizando-os e escalando-os. Em seguida, computou-se o erro quadrático médio da raiz para diferentes números de componentes sendo retidos, de 1 a 50, utilizando validação cruzada com k-dobras, a fim de escolher a dimensão adequada para o método.

F. Rede Neural

Uma rede neural é um modelo de aprendizado de máquina inspirado no cérebro humano, que reproduz a interação entre neurônios biológicos para identificar padrões, avaliar possibilidades e tomar decisões de forma eficiente [10].

O análogo ao neurônio biológico é a estrutura denominada perceptron. Esta é basicamente um sistema que recebe múltiplas entradas, multiplica cada uma pelo peso correspondente, soma esses valores e, por fim, aplica uma função de ativação para determinar o resultado final. Estes quando organizado em camadas permitem a extração de padrões e a representação de relações complexas nos dados. A estrutura básica de um neurônio artificial é dada pela figura 2.

Fig. 2. Neuron



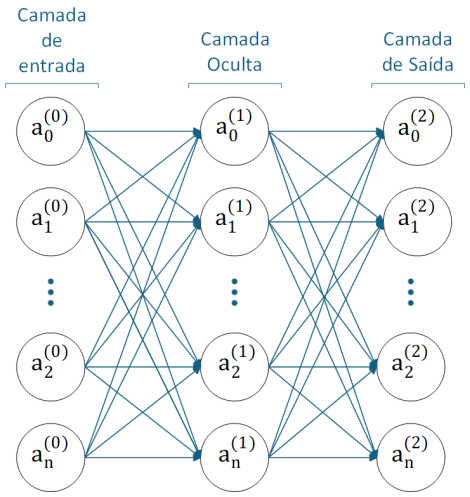
De forma matemática temos que a saída de cada perceptron é dada por:

$$h_k(\mathbf{x}) = g\left(b_k + \sum_{i=1}^{m} x_j w_{jk} \right) \tag{10}$$

Quando as camadas de uma rede neural são organizadas de forma sequencial, em que cada camada repassa suas informações processadas para a camada seguinte, o modelo

resultante é chamado de Perceptron Multicamadas (MLP). Sua estrutura apresenta uma camada de entrada que recebe os dados (sinais), camadas ocultas que realizam o processamento destes dados e uma camada de saída que fornece o resultado do processamento realizado no decorrer da rede neural. Nesse modelo, cada neurônio em uma camada recebe os sinais de entrada, realiza o processamento e, quando ativado, transmite os resultados para a próxima camada, permitindo o aprendizado e a captura de padrões complexos por meio dessa estrutura hierárquica. Segue a Figura 3 desta estrutura:

Fig. 3. Estrutura hierárquica do Perceptron Multicamadas



Apesar da figura não representar, temos que o número de camadas ocultas pode ser arbitrário e não limitado a somente uma.

Conhecido a estrutura do perceptron e avaliando a imagem y temos que um neurônio de uma camada qualquer, exceto da de entrada, é dada por:

$$a_k^{(l)} = g\left(w_{0,0}a_l^{(l-1)} + w_{0,1}a_1^{(l-1)} + \cdots + w_{0,n}a_n^{(0)} + b_k\right) \tag{11}$$

Generalizando para a camada e representando de forma matricial obtém-se:

$$\mathbf{a}^{(l)} = g\left(\mathbf{W}\mathbf{a}^{(l-1)} + \mathbf{b}\right) \tag{12}$$

$$a^{(l)} = g\left(\begin{bmatrix} w_{0,0} & w_{0,1} & \dots & w_{0,n} \\ w_{1,0} & w_{1,1} & \dots & w_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{k,0} & w_{k,1} & \dots & w_{k,n} \end{bmatrix} \begin{bmatrix} a_0^{(l-1)} \\ a_1^{(l-1)} \\ \vdots \\ a_n^{(l-1)} \end{bmatrix} + \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_k \end{bmatrix}\right) \tag{13}$$

Contudo a mera geração desta estrutura não fornecerá resultados desejados, é necessário que a rede neural ajuste seus parâmetros internos, como pesos e vieses, de forma a melhorar sua capacidade preditiva. Este processo é chamado de aprendizado e é proveniente de um algoritmo chamado backpropagation (retropropagação) [11].

O backpropagation é um método utilizado para treinar redes neurais, ajustando iterativamente os pesos das conexões entre os neurônios para minimizar a função de custo. Esse algoritmo calcula o gradiente da função de custo em relação aos pesos da rede, propagando o erro da camada de saída para as camadas anteriores. O intuito é encontrar o mínimo global da função de erro, contudo nem sempre tal mínimo é encontrado e a rede pode ficar em um mínimo local.

Essa minimização da função de erro é coordenada pelo algoritmo de descida do gradiente (do inglês *gradient descent*) e, no caso de regressão linear, a função de erro é dada como o erro quadrático médio. A cada iteração, o valor dos pesos (w) e vieses (b) são atualizados com os valores da iteração anterior e parametrizados por λ, a taxa de aprendizado, onde J é a função de perda:

$$w = w - \lambda\left(\frac{\partial J(w,b)}{\partial w}\right) \tag{14}$$

$$b = b - \lambda\left(\frac{\partial J(w,b)}{\partial b}\right) \tag{15}$$

Um valor de taxa de aprendizado muito pequeno pode levar a muitos passos de computação para encontrar o mínimo, enquanto um valor muito grande pode levar o algoritmo a ficar alternando entre valores próximos do mínimo no vale da função custo, evitando que o valor mínimo seja alcançado.

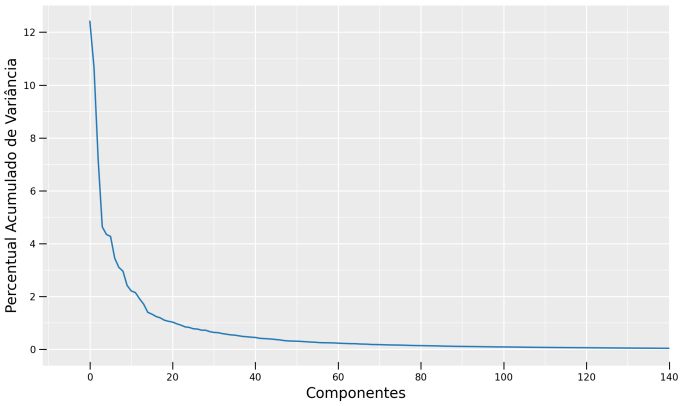
III. Resultados

A. Resultados da Análise exploratória e o pré-processamento

A assimetria da distribuição apurada dos dados originas tem amplitude de 0.67 a 3.84, com média de 1.65. Assim, transformou-se os dados utilizando a técnica Box-Cox para reduzir a assimetria da distribuição dos dados, aplicando-os a todos os preditores contínuos, somando uma unidade a todas os valores de todos os preditores contínuos para que sejam estritamente positivos.

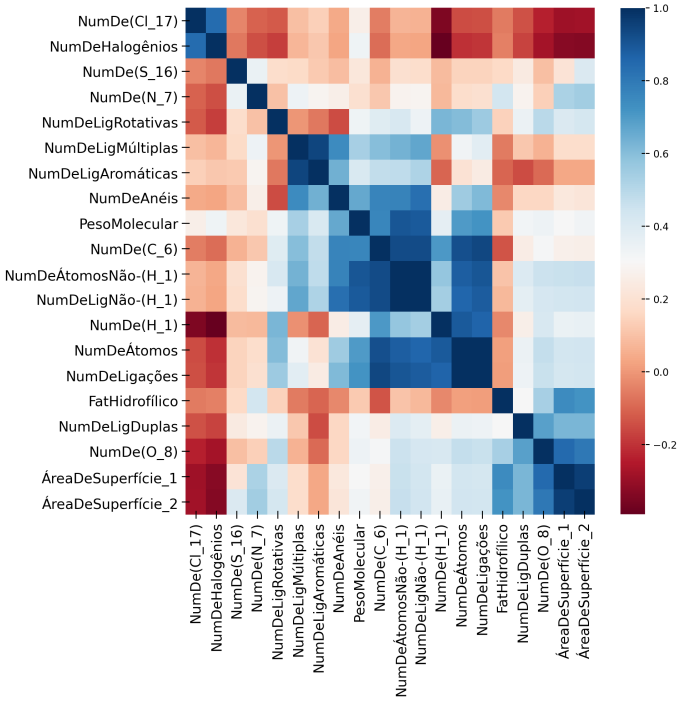
A partir da Figura 4, que apresenta a variância explicada por componente principal, percebe-se que nenhum dos componentes explica mais do que 13% da variância e a maioria explica menos do que 2% de variância, o que indica que a estrutura dos dados está contida em um espaço de menor dimensão do que o original, devido a um alto grau de colinearidade.

Fig. 4. Análise de componentes principais dos preditores



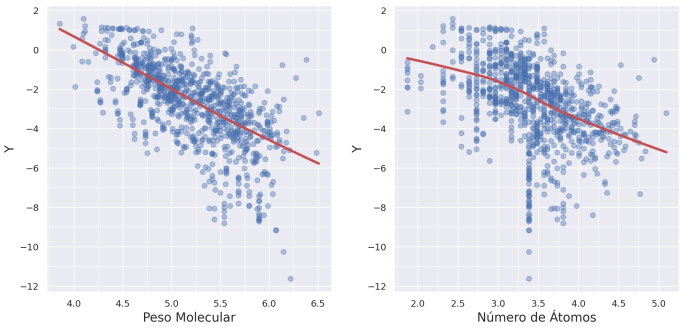
Utilizando da matriz de correlação representada pela Figura 5, percebe-se que há vários pares de preditores correlacionados, o que confirma um efeito de colinearidade de grande impacto.

Fig. 5. Matriz de correlação entre preditores contínuos



Observam-se nas Figuras 6 e 7 relações lineares entre preditores e saída como o peso das moléculas (MolWeight) e número de ligações (NumBonds), bem como outras relações não-lineares como número de átomos de cloro (NumChlorine) e número de atômos halógenos (NumHalogen). Utilizou-se o suavizador "lowess" para desenhar as curvas que representam as relações.

Fig. 6. Gráficos de dispersão da relação entre preditores contínuos e saída



B. Resultados da Regressão Linear Ordinária

O modelo de regressão linear foi avaliado com validação cruzada com 5 e 10 dobras. Na tabela I temos os resultados das métricas no conjunto de treino.

Fig. 7. Gráficos de dispersão da relação entre preditores contínuos e saída

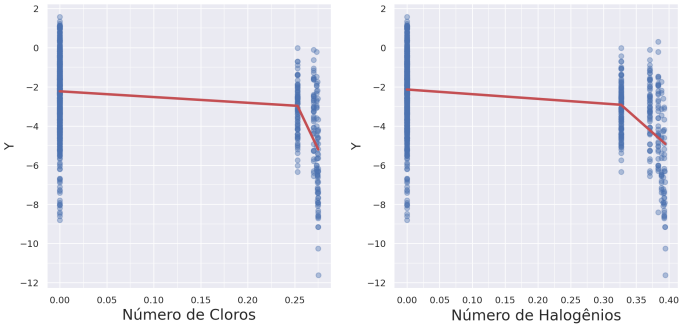


TABLE I
COMPARAÇÃO DOS RESULTADOS DA REGRESSÃO LINEAR ORDINÁRIA PARA 5 E 10 DOBRAS.

Nº de Dobras	RMSE	R^2
5	0.7453	0.8626
10	0.7105	0.8743

Aplicando o modelo de regressão no conjunto teste obtemos RMSE igual a 0.7969 e R^2 igual a 0.8525. Comparando-o com os resultados do treinamento, os quais, como explicados, foram obtidos a partir da média dos valores calculados em cada repetição, podemos concluir que estão bem condizentes entre si, sem uma elevada diferença entre eles. Ademais, por se tratar de um modelo simples, o *Bias* tende a influenciar para que o erro nao seja tao reduzido.

C. Resultados da Regressão L-2 Penalizada (Ridge)

Os modelos de regressão Ridge foram avaliados com validação cruzada com 5 e 10 dobras, variando o valor de λ de 0.1 a 30 divididos em 100 valores igualmente espaçados. Segue a tabela com as comparações:

TABLE II
COMPARAÇÃO DOS RESULTADOS DA REGRESSÃO RIDGE PARA 5 E 10 DOBRAS.

Nº de Dobras	Melhor λ	RMSE	R^2
5	30.000	0.697	0.881
10	29.698	0.690	0.881

Os resultados mostram que, para ambas as configurações (5 e 10 dobras), o modelo alcançou valores consistentes de R^2 (Figura 9), superiores a 85%. Além disso, os valores de $RMSE$ foram próximos para ambos os modelos (Figura 8), o que reforça a capacidade do modelo em diferentes esquemas de validação.

D. Resultado do método dos Minímos Quadrados Parciais

Optou-se por escolher o método dos Mínimos Quadrados Parciais (do inglês Partial Least Squares, ou PLS) como método que utiliza componentes principais. Na Figura 10, é possível analisar o erro quadrático médio da raiz (RMSE) e o coeficiente de determinação médio por número de componentes após uma validação cruzada de k igual a 8. Utilizando a regra do "um

Fig. 8. Gráfico de Comparação do RMSE Médio para 5 e 10 dobras

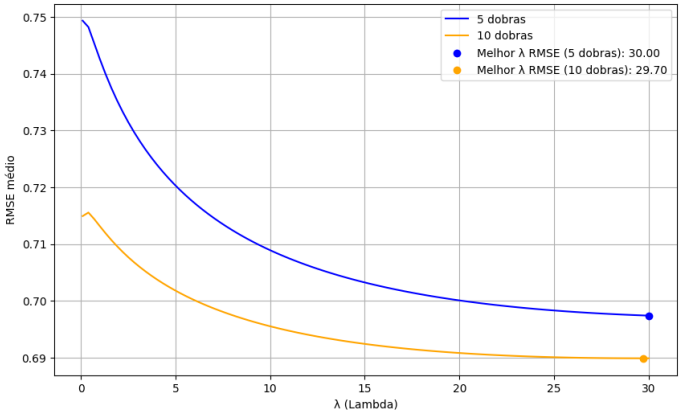
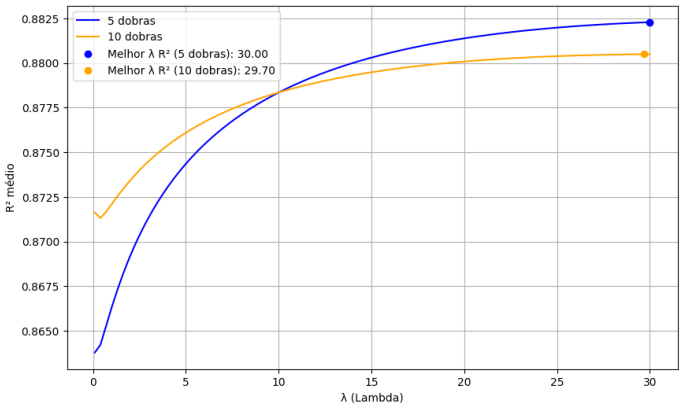


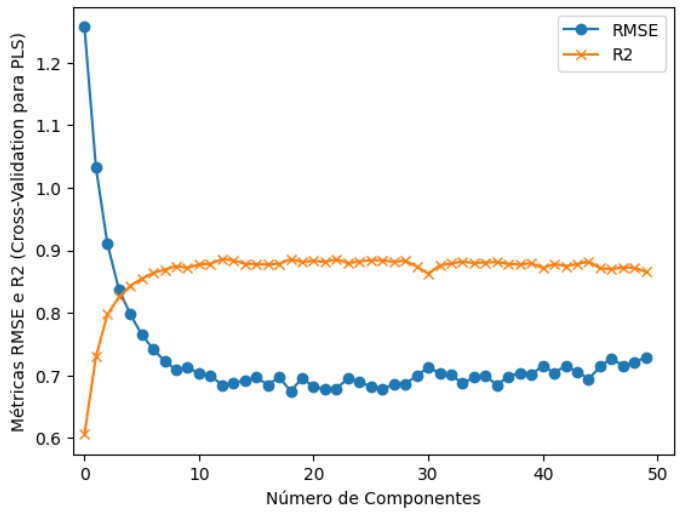
Fig. 9. Gráfico de Comparação do R² Médio para 5 e 10 dobras



erro padrão" (do inglês *one-standard error*), o número de componentes escolhidos a ser retido foi determinado como 10.

Ao testar o modelo obteve-se erro quadrático médio da raiz igual a 0.78 e coeficiente de determinação igual a 0.86.

Fig. 10. Métricas por quantidade de componentes usando validação-cruzada



E. Resultado da Rede Neural

O modelo de rede neural aplicado foi avaliado com validação cruzada de 5 e 10 dobras. Durante cada execução, o treinamento realizava 200 épocas, utilizando 64-48-24-1 neurons, sucessivamente, tendo a função *reLU* como ativação.

TABLE III
COMPARAÇÃO DOS RESULTADOS DA REDE NEURAL PARA 5 E 10 DOBRAS.

Nº de Dobras	**RMSE (Treino)**	**R^2 (Treino)**
5	0.7330	0.8701
10	0.7005	0.8771

Na tabela III, podemos verificar que os resultados são satisfatórios possuindo, por exemplo, um R^2 melhor do que encontrado nos outros modelos (OLS e PLS), especialmente, quando o número de dobras é igual a 10. Dessa forma, concluímos que por a rede neural ser um modelo não linear e complexo o bias é reduzido e o nível de variância se apresenta mais elevado dentre os resultados, uma vez que ela obteve ótimas métricas de acurácia durante a validação cruzada.

Em suma, o de PLS aquele que obteve o menor coeficiente de determinação entre todos e o maior RMSE, enquanto que o de regressão ordinária se apresentou com métricas de treino medianas, ficando em terceiro lugar. Por conseguir ter excelente desempenho mesmo em preditores enviesados, o modelo de *Ridge* adquiriu os melhores resultados, superando inclusive a rede neural.

REFERENCES

[1] Igor V. Tetko, J. Huuskonen, *Estimation of Aqueous Solubility of Chemical Compounds Using E-State Indices*, Applied Predictive Modelling (Springer book). [Online]. Disponível em: http://appliedpredictivemodeling.com/data

[2] M. Kuhn, K. Johnson, *Applied Predictive Modeling*, 2013rd ed., Springer, ISBN 978-1-4614-6849-3.

[3] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning: with Applications in Python*, 2023rd ed., Springer Texts in Statistics, 2023.

[4] A. Hayes, *Multiple Linear Regression (MLR) Definition, Formula, and Example*, 16-jul-2024, Investopedia. [Online]. Disponível em: https://www.investopedia.com/terms/m/mlr.asp

[5] C. M. Bishop, *Pattern Recognition and Machine Learning*, 2006th ed., Springer, ISBN 978-0387-31073-2.

[6] R. Duarte, *Underfitting, Overfitting e o Princípio de Bias-Variance Trade-off*, 21-nov-2023, Sigmoidal. [Online]. Disponível em: https://sigmoidal.ai/underfitting-overfitting-e-o-principio-de-bias-variance-trade-off/

[7] GeeksforGeeks, *Gradient Descent Algorithm and its variants*, *GeeksforGeeks*, 22-set-2021. [Online]. Disponível em: https://www.geeksforgeeks.org/gradient-descent-algorithm-and-its-variants/. [Acessado: 26-jan-2025].

[8] Google Colab: Códigos Utilizados. Link: https://colab.research.google.com/drive/1vZI6WAq_ybxJ2Xvy46caOuqV_RD70BZz?usp=sharing

[9] K. Marshall, *What is the benefit of ridge regression?*, *DeepChecks*. [Online]. Disponível em: https://www.deepchecks.com/question/what-is-the-benefit-of-ridge-regression/#:~:text=Ridge%20helps%20you%20normalize%20. [Acessado: 28-jan-2025]

[10] IBM, *What is a neural network?*, 02-jul-2024, IBM. [Online]. Disponível em: https://www.ibm.com/think/topics/neural-networks

[11] D. Bergmann, C. Stryker, *O que é retropropagação?* , 02-jul-2024, IBM. [Online]. Disponível em: https://www.ibm.com/br-pt/think/topics/backpropagation#:~:text=Retropropaga%C3%A7%C3%A3o%20%C3%A9%20uma%20t%C3%A9cnica%20de,IA)%20moderna%20%22aprendem%22