Projeto Sistemas Inteligentes 2024.1

Maria Eduarda Oliveira de Melo
Centro de Informática
Universidade Federal de Pernambuco (UFPE)
Recife, Brasil
meom@cin.ufpe.br

***Abstract*—Since John McCarthy coined the term Artificial Intelligence (AI) in 1956, the field has gained increasing popularity and prominence. Given the relevance and potential impact of AI in several sectors, the present work aims to test the efficiency of four Machine Learning algorithms, K-Nearest Neighbors (KNN), Decision Trees, Artificial Neural Networks (RNN) and Support Vector Machines (SVM), in a regression problem, in the context of the Intelligent Systems discipline at the Informatics Center of the Federal University of Pernambuco. The problem addressed involves predicting the age of abalone molluscs, using a database with 4177 instances and nine attributes. The results showed that Neural Networks presented the highest accuracy.**

***Index Terms*—Artificial Intelligence (AI), Artificial Neural Networks (RNN), Machine Learning (ML), Regression Problems, K-Nearest Neighbors (KNN), Decision Trees, Support Vector Machines (SVM), Abalone**

***Abstract*—Desde que John McCarthy cunhou o termo Inteligência Artificial (IA) em 1956, a área tem ganhado crescente popularidade e destaque. Dada a relevância e o impacto potencial da IA em diversos setores, o presente trabalho objetiva testar a eficiência de quatro algoritmos de Aprendizado de Máquina, K-Nearest Neighbors (KNN), Árvores de Decisão, Redes Neurais Artificiais (RNN) e Support Vector Machines (SVM), em um problema de regressão, no contexto da disciplina de Sistemas Inteligentes do Centro de Informática da Universidade Federal de Pernambuco. O problema abordado envolve a previsão da idade de moluscos da espécie abalone, utilizando uma base de dados com 4177 instâncias e nove atributos. Os resultados mostraram que as Redes Neurais apresentaram a maior precisão.**

***Index Terms*—Inteligência Artificial (IA), Redes Neurais Artificiais (RNN), Aprendizado de Máquina (AM), Problemas de Regressão, K-Nearest Neighbors (KNN), Árvores de Decisão, Support Vector Machines (SVM), Abalone**

I. Introdução

Apesar do termo Inteligência Artificial (IA) ter sido proposto por John McCarthy em uma conferência realizada em Darthmouth já em 1956 para caracterizar alguns tópicos que há tempo vinham sendo discutidos [3], é inegável o crescimento da popularidade e destaque que essa tecnologia têm ganhado mais recentemente. Essa intensificação no interesse na área pode ser justificado por vários motivos, entre eles a significativa melhora no poder computacional, a maior disponibilidade de dados e uma performance notável que sistemas baseados em IA conseguem alcançar diante de problemas já conhecidos e debatidos na literatura [5].

Diante, desse cenário apresentado e do impacto potencial em diversos âmbitos da sociedade, é imprescindível que haja investimento, suporte e ampliação de pesquisas científicas, inovações tecnológicas e disseminação de conhecimento. Tais ações, muito comumente, ficam a cargo das instituições de ensino e pesquisa, com grande destaque para o papel desenvolvido pelas universidades federais brasileiras [3].

Portanto, é nesse contexto que surge o presente trabalho, o qual tem como objetivo testar a eficiência de quatro algoritmos de Aprendizado de Máquina (AM) em um problema simples de regressão, elaborado para fins avaliativos na disciplina de Sistemas Inteligentes, do Centro de Informática da Univesidade Federal de Pernambuco, cujo propósito é introduzir a base desse conhecimento para alunos de computação.

II. Base de Dados

O problema explorado no presente estudo trata-se do uso de IA na predição rápida, fácil e precisa da idade de uma espécie de molusco chamada abalone, cujo método clássico, amplamente aderido, é muito demorado e trabalhoso, por ser necessário considerar o corte de sua concha, pintando e contando os seus anéis. Com isso, outras características da espécie foram levantadas e consolidadas na base de dados explorada nesse estudo com a intenção de buscar um melhor método de previsão.

A base de dados disponibilizada por [4] contém quatro mil cento e setenta e sete instâncias, um target e oito *features*, sendo elas, uma categórica representando o sexo do molusco; uma inteira, representando a quantidade de anéis do animal; e sete contínuas, representando a largura, diâmetro, altura, peso total, peso sem casca, peso das vísceras e peso da casca.

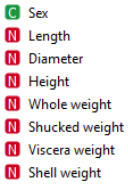

Fig. 1. Features da base de dados

Para que as informações desse repositório pudessem ser melhor aproveitadas, foi necessário uma fase de tratamento e pré-processamento dos dados, que envolveu, primeiramente, a criação de uma nova coluna na tabela, denominada ”Age”, cujo valor era o mesmo presente na coluna ”Rings” multiplicado por 1.5 para estimar a idade do animal e a eliminação da coluna ”Ring”. Posteriormente, os dados foram normalizados para apresentarem média zero e desvio padrão de um e as variáveis categóricas foram transformadas, de forma a serem

representadas por três colunas (F, M e I), cada uma contendo o valor 1 para indicar pertencimento àquela classificação ou 0 quando pertencente a outra. Por último, foi realizada uma amostragem dos dados, em que 80% das amostras ficou no grupo de treinamento e 20% foi para o grupo de teste.

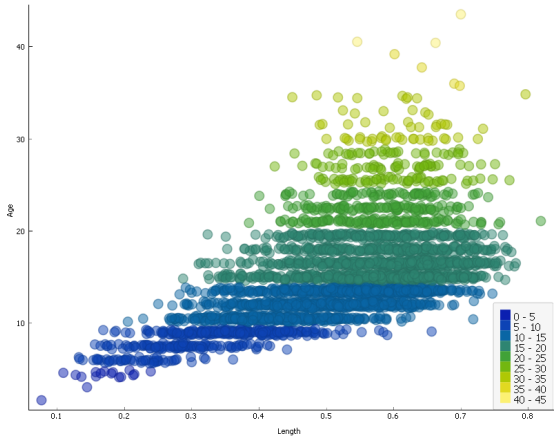

Fig. 2. Distribuição dos dados

III. Algoritmos

Quatro principais algoritmos supervisionados foram vistos e trabalhados durante a disciplina: K-Nearest Neighbors (KNN), Árvores de Decisão, Redes Neurais Artificiais (RNN) e Support Vector Machines (SVM); resultando em suas escolhas como possíveis candidatos para tentar solucionar o problema do abalone.

A. *Árvores de Decisão*

Uma Árvore de Decisão é uma estrutura usada para criar um conjunto de regras a partir dos dados de treinamento. O algoritmo iterativamente encontra a *feature* que melhor categoriza os dados e as usa para criar as divisões até chegarmos às folhas em que existem apenas uma classe ou o limite de profundidade da árvore foi atingido [6].

Os parâmetros desse modelo utlizados nesse estudo foram um booleano para indicar se deve ser induzido uma árvore binária, o número mínimo de instâncias nos nós das folhas, o número mínimo de amostras necessárias para dividir um nó e o limite de profundidade da árvore. No nosso problema, os valores utilizados foram, respectivamente: True, 2, 5 e 100.

Um dos benefícios dessa técnica é que, uma vez que o conjunto de regras foi definido, a análise de uma nova instância pode ser realizada em tempo real [6]. Além disso, esse é um algoritmo simples, mas bastante poderoso, pois nos permite aprender durante o seu uso, é bastante sensível a *outliears*, funciona bem com tanto dados numéricos e como categóricos mas, podem não ser muito bons na previsão de valores contínuos devido à sua natureza disjunta.

B. *K-Nearest Neighbor (KNN)*

O funcionamento do KNN é baseado em aprender de amostras de dados para criar classes ou grupos. A vizinhança é definida como um número K de pontos de acordo com uma métrica de distância, geralmente, a Euclidiana, os votos dos K vizinhos decidem em qual classe o novo ponto será alocado [6].

Para isso, esse modelo recebe como argumento o valor de K, ou seja da vizinhança que será utilizada para ”votar” na classe do novo ponto; a métrica de distância que deverá ser utilizada para calcular a distância entre os pontos; e os pesos que cada ponto terá na votação. No nosso contexto, os valores utilizados foram, respectivamente: 7, distância Euclidiana e distribuição uniforme.

Essa técnica costuma ser bastante afetada por dados desbalanceados, pode ser computacionalmente custosa, já que para cada ponto, é preciso calcular sua distância para todos os outros pontos, mas em compensação não é preciso treinar o algoritmo ou retreiná-lo com a presença de novos dados ou classes.

C. *Support Vector Machine (SVM)*

Os modelos SVMs estendem modelos de regressão linear, ao classificar amostras de dados, um plano que separa as amostras de dados em duas classes é definido. O hiperplano de separação pode ser modelado de forma linear, não linear, polinomial, Gaussiana, radial, sinóide, etc, dependendo da função Kernel aplicada [6].

Para esse tipo de modelos, precisamos passar: o parâmetro de regularização C, para controlar a margem de decisão; a função Kernel a ser utilizada; e o número máximo de iterações. Nos foi oportuno, nesse problema, preencher esses valores, respectivamente, como: 1, RBF e 100.

Esses modelos são bastante eficazes em espaços de alta dimensionalidade e a opção de diferentes tipos de Kernels permitem a modelagem de funções complexas, entretanto, geralmente aplicados em problemas de classficiação, podem não obter resultados muito bons em problemas de regressão como o nosso.

D. *Rede Neural Artificial (RNN)*

A técnica de aprendizado das Redes Neurais Artificiais foram inspiradas nos neurônios de seres humanos de verdade, esses neurônios são modelados em termos de equações matemáticas que lêem uma série de amostras de dados para gerar um valor alvo. O algoritmo dessas redes itera até que o valor do output esteja dentro de um range de erro aceitável do valor alvo, em cada iteração, os neurônios aprendem corrigindo os valores de seus pesos e medindo o quão longe eles estão do valor pretendido, a partir de determinados padrões identificados em dados de entrada [6].

Os parâmetros que precimos preencher na definição desses modelos são: o número de neurônios nas camadas escondidas, a função de ativação a ser utilizada, qual otimizador utilizaremos, a taxa de aprendizado e o número máximo de iterações. Esse parâmetros foram preenchidos, respectivamente, com os seguintes valores: 100, ReLU, Adam, 0.0001 e 200.

Essas redes são capazes de modelar relações não lineares complexas nos dados e podem ser aplicadas para uma ampla

gama de problemas, entretanto, treinar e ajustar uma rede neural pode ser computacionalmente caro e tendem a ser mais propensas à *overfitting*.

IV. Resultados Obtidos

Para avaliar os algoritmos mencionados na seção passada de forma prática e sem que houvesse a necessidade de implementação de código, foi utilizada a ferramenta Orange, de código aberto para análise de dados e aprendizado de máquina, desenvolvida pela Universidade de Ljubljana, na Eslovênia [2]. Nela, além das fases de pré-processamento e definição dos modelos já mencionadas, também é possível realizar os treinamentos e testar e validar os resultados obtidos.

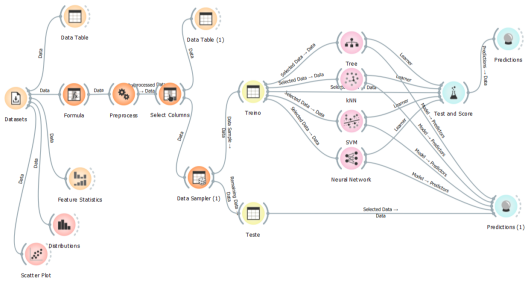

Fig. 3. Esquema montado na plataforma Orange

Considerando que nosso caso envolve um problema de regressão, as métricas utilizadas para avaliar o desempenho dos modelos foram a média dos quadrados dos erros (MSE), a raiz quadrada do MSE (RMSE), a média dos valores absolutos dos erros (MAE) e a proporção de variância explicada pelo modelo (R2), obtendo os seguintes valores no conjunto de treinamento e teste:

TABLE I
Métricas do conjunto de treinamento

Model	Train	MSE	RMSE	MAE	R2
Tree	16.492	0.782	0.884	0.623	0.218
SVM	0.340	1.335	1.155	0.909	-0.336
NN	13.198	0.429	0.655	0.461	0.571
kNN	0.107	0.504	0.710	0.497	0.496

TABLE II
Métricas do conjunto de teste

Model	MSE	RMSE	MAE	R2
Tree	0.856	0.925	0.649	0.161
SVM	1.644	1.282	1.049	-0.612
NN	0.425	0.652	0.463	0.583
kNN	0.489	0.700	0.495	0.520

Diante das tabelas apresentadas, podemos verificar que o modelo do KNN foi o que obteve melhor desempenho com relação ao tempo de treinamento e a Árvore de decisão foi o que se saiu pior nesse quesito. Já com relação à acurácia, o modelo de Rede Neural foi o que mais se destacou, aprensentando taxas de erros mais baixas que todos os outros modelos e uma boa variância explicada, tanto nos dados de treinamento, quantos nos de teste. Um destaque pode ser dado para o resultado negativo obtido pelo SVM na métrica R2 e o menos discrepante, mas também bastante baixo, valor da Tree na mesma métrica, que podem ser justificados pelas suas naturezas em lidar com problemas de classificação.

V. Implementação do Algoritmo

Perante a análise dos dados obtidos através da ferramenta do Orange e consequente definição da Rede Neural como modelo com o melhor desempenho estimando as idades dentre os modelos analisados, mesmo apresentando tempo de treinamento consideravelmente superior que os modelos KNN e SVM, foi realizada sua implementação em código na linguagem Python, usando a plataforma do Google Colab e a biblioteca do keras [1].

A. Carregamento e Tratamento dos Dados

Para facilitar os cálculos e operações em nossos dados, após importar a base de dados, utilizamos a estrutura de *DataFrame* da biblioteca Pandas do Python para armazená-la. Com essa estrutura pudemos aplicar as operações para gerar a coluna ”Age” e deletar a coluna ”Rings”. Em seguida, a partir do módulo *preprocessing* da biblioteca do *sklearn*, utilizamos a função *OneHotEncoder* para transformar as variáveis categóricas em valores binários e a função *StandardScaler* para normalizar os dados de treinamento e teste, os quais já aviam sido separados com auxílio da função *train_test_split* do módulo *model_selection*, também da biblioteca *sklearn*.

B. Definição do Modelo

Para criar o modelo da Rede Neural, primeiro o definimos como um modelo sequencial e em seguida foram adicionadas duas camadas densas com 64 neurônios com a função de ativação *ReLU* e mais uma camada densa de saída com apenas um neurônio, do qual sairão nossas predições. O modelo terminou com um total de quatro mil novecentos e vinte e nove parâmetros, todos treináveis.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 64)                704

 dense_1 (Dense)             (None, 64)                4160

 dense_2 (Dense)             (None, 1)                 65

=================================================================
Total params: 4929 (19.25 KB)
Trainable params: 4929 (19.25 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Fig. 4. Arquitetura do modelo de Rede Neural

O modelo foi compilado usando o otimizador *Adam* com o *learning rate* de 0.001 e para auxiliar na realização dos ajustes nos pesos, foi utilizada a métrica MSE. Depois de compilado, o modelo foi treinado por vinte épocas e um *batch* de tamanho trinta e dois com os dados de treinamento e seus rótulos, por se tratar de um modelo supervisionado, e desses dados, 2% foi

utilizado para validação. Durante as vinte épocas os valores da *loss* diminuíram tanto para o conjunto de treinamento, quanto para o conjunto de validação, sugerindo que não houve *overfitting*, ajuste excessívo aos dados, durante o treinamento.

C. Resultados

Para que fosse possível ter uma noção visual do desempenho do modelo preditivo com relação aos dados reais, foi criada duas visualizações em formato de gráfico. O primeiro gráfico foi um scatterplot com os valores preditos no eixo y e os valores reais no eixo x e ao meio, uma linha cruzando o gráfico que representa a posição ideal em que os valores preditos são iguais aos reais, logo, quanto mais próximo os pontos se encontram dessa linha melhor. A segunda representação gráfica foi um histograma, em que no eixo x temos cada uma das ocorrências e no eixo y o range dos valores que os pontos atingiram, as linhas representam a variação dos valores reais e a variação dos valores preditos.

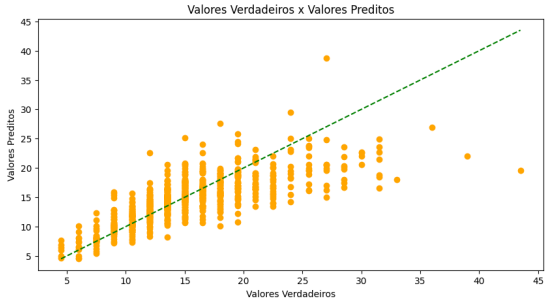

Fig. 5. Relação dos valores reais e previstos diante da situação ideal

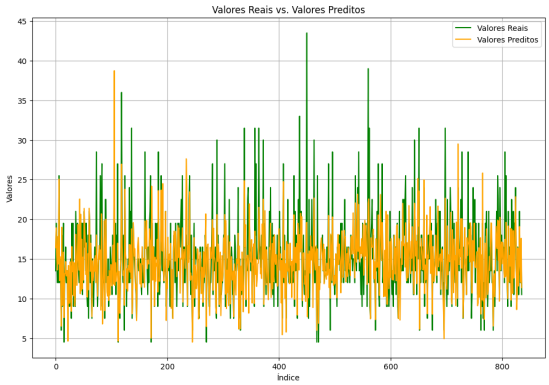

Fig. 6. Valores atingidos realmente e preditos

Apesar de satisfatório no nosso contexto, várias técnicas poderiam ser aplicadas na nossa RNN para que pudéssemos ampliar o desempenho atingido. Uma delas seria o ajuste dos hiperparâmetros, aumentando o número de camadas da nossa rede, ou a quantidade de neurônios em cada camada; treinar o modelo por uma maior quantidade de épocas e com um batch de tamanho menor; testar diferentes otimizadores e funções de ativação. Outra técnica bastante poderosa é a utilização de *Ensemble Learning*, para combinar o conhecimento de vários modelos para se chegar no output final. Dois exemplos dessa técnica são o *Bagging*, em que os modelos são treinados paralelamente e o conhecimento aprendido por cada um é utilizado na geração do output final e o outro exemplo é *Boosting*, em que os modelos são treinados em sequência, nesse sentido, um modelo vai aprender a partir dos erros de um modelo anterior.

VI. Conclusão

Diante da análise realizada sobre o desempenho dos algoritmos de aprendizado de máquina no problema de regressão da idade de abalones, fica evidenciada a capacidade desses algoritmos em lidar com problemas complexos. O KNN, apesar de sua simplicidade apresentou desempenho considerável, apesar do melhor resultado da RNN, por se tratar de um método mais sofisticado para lidar com problemas complexos.

Portanto, concluímos esse trabalho reforçando a relevância contínua de investimento em pesquisas e desenvolvimento de técnicas de inteligência artificial, não apenas para fins acadêmicos, mas também para aplicações práticas que podem melhorar diversos setores da sociedade, destacando a crescente capacidade desses modelos em lidar com problemas complexos e variados com potencial significativo de para contribuição para o avanço científico e tecnológico em um futuro próximo.

References

[1] Abadi, M., et al. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

[2] Demsar, J., et al. (2013). Orange: Data Mining Toolbox in Python. Journal of Machine Learning Research, 14(Aug), 2349-2353.

[3] GROENNER, Luciana Castro; FARIA, Leandro Innocentini Lopes de; PERISSINI, Rodrigo César; GRACIOSO, Luciana de Souza. Um estudo bibliométrico sobre a pesquisa em inteligência artificial no Brasil. Brazilian Journal of Information Science, v. 16, p. 8, 2022.

[4] Nash,Warwick, Sellers,Tracy, Talbot,Simon, Cawthorn,Andrew, and Ford,Wes. (1995). Abalone. UCI Machine Learning Repository. https://doi.org/10.24432/C55C7W.

[5] Zhou, Zhi Hua. “Machine learning: Recent progress in China and beyond”. National Science Review, vol. 5, no. 1, 2018, pp. 20, doi:10.1093/nsr/nwx132. Acessado 19 jul. 2021.

[6] Zeadally, S., Adi, E., Baig, Z. and Khan, I.A., 2020. Harnessing artificial intelligence capabilities to improve cybersecurity. Ieee Access, 8, pp.23817-23837.