O Impacto do Pré-processamento dos Dados em Atividades Supervisionadas

Samuel Borges Ferreira Gomes (261663)

Departamento de Telecomunicações (DECOM)
Faculdade de Engenharia Elétrica e de Computação (FEEC)
Universidade Estadual de Campinas (Unicamp)

{Samuel Borges Ferreira Gomes}samuelbfgomes@gmail.com

Abstract – Artificial Neural Networks is a useful tool in pattern recognition and classification tasks. Using the UCI Poker Hands dataset, this paper aims to solve poker hand classification problem using different database arrangements, with MLP Neural Networks as main tool. Experimental results in Python are presented to demonstrate the performance of the proposed system. The paper also makes comparisons over different supervised learning algorithms. One of the proposed database arrangements succeeded in classification of poker hands in 99% classification accuracy.

Keywords – ANN, MLP, Supervised Learning, Data Pre-processing, Poker hands

1. Introdução

Poker é um dos jogos mais famosos de aposta do mundo todo. Não é um jogo totalmente determinístico como Xadrez ou Damas, em que o espaço do jogo é sempre visível para todos os jogadores. No poker, a condição de vitória é determinada de acordo com a combinação das cartas do jogador, das quais são pré-definidas [1]. Além disso, poker é um jogo soma nula, isto é, a soma das jogadas contadas como perdas e vitórias somam 0.

Por sua grande fama, poker é demasiadamente estudado por teóricos de jogos, economistas e matemáticos. Originado nos Estados Unidos na década de 90, diversos modelos numéricos foram investigados, incluindo jogos com mãos discretas e contínuas, com jogos de apostas simultâneas, etc [1, 7]. Além disso, modelos teóricos simples foram vastamente estudados na literatura [9, 6].

O objetivo desse trabalho é classificar 5 cartas retiradas aleatoriamente de um baralho padrão de 52 cartas pelo naipe e valor de cada carta. Os dados foram retirados da *UCI Machine Learning Repository* [11]. Os dados de treinamento possuem mais de 25000 amostras com 11 atributos. 5 atributos são os naipes (1-5) possíveis, outros 5 atributos pros valores (1-13) possíveis, e 1 atributo para a "força"da mão do jogo. A classificação é feita através de redes MLP, com diferentes topologias. Com isso, explorar quatro arranjos diferentes do banco de dados e analisar a performance da rede, bem como o impacto que o pré-processamento desses dados pode causar na acurácia final.

Além disso, é feito uma comparação entre a acurácia das redes MLP, com a de outros algoritmos de aprendizagem supervisionada, onde MLP obteve a melhor performance para a tarefa de classificação. O *dataset* se mostrou difícil de trabalhar. É um exemplo de um conjunto de dados desequilibrado em que as mãos de poker mais comuns, como pares, são fortemente representadas e as mãos menos comuns, como *straight* e *flush*, não são. Diferentes topologias foram analisadas na literatura, utilizando o mesmo *dataset*, considerando aprendizagem supervisionada [12, 5], não-supervisionada [5, 2], e por aprendizado com reforço [10]. Diferentemente destes, o presente trabalho explora diferentes arranjos dos *datasets*.

2. Metodologia

As redes neurais artificiais vem se mostrando potentes ferramentas no aprendizado de máquina, principalmente no que se diz respeito a realização de tarefas de classificação, e problemas de reconhecimento de padrões no geral.

No contexto da inteligência artificial, tem-se um aprendizado supervisionado quando algoritmos aprendem através de dados rotulados com as soluções desejadas, chamadas de *labels*. As redes MLP (do inglês, *multilayer perceptron*), é uma estrutura neural que utiliza neurônios do tipo *perceptron*, dispostos em diferentes camadas, realizando mapeamentos não-lineares. O algoritmo de retropropagação de erro tenta otimizar uma matriz de pesos, através de métodos iterativos (como o Gradiente Descendente), afim de minimizar o erro quadrático médio MSE (do inglês, *mean squared error*). Esse algoritmo aprende os pesos de densas redeus neurais totalmente conectadas de múltiplas camadas, tentando minimizar o MSE entre os valo-

res da saída da rede com os *labels*. Através dessa abordagem, o uso de redes neurais tem-se provado satisfatório em diversas tarefas de reconhecimento de padrões e classificação [14, 15, 16].

	N1	V1	N2	V2	N3	V3	N4	V4	N5	V5	jogo
0	1	10	1	11	1	13	1	12	1	1	9
1	2	11	2	13	2	10	2	12	2	1	9
2	3	12	3	11	3	13	3	10	3	1	9
3	4	10	4	11	4	1	4	13	4	12	9
4	4	1	4	13	4	12	4	11	4	10	9

Figura 1. Representação do banco de dados *Poker Hands*.

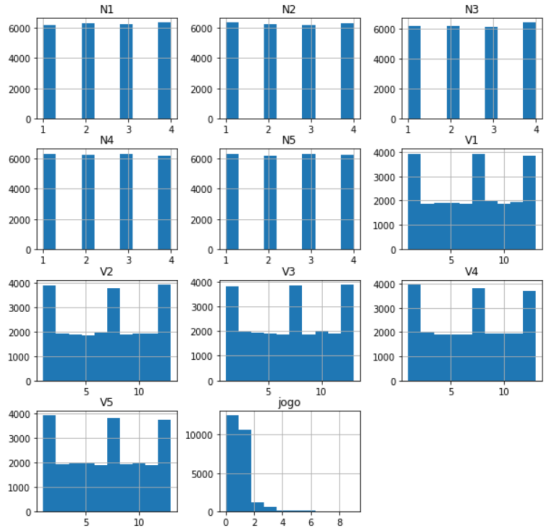

Figura 2. Histograma dos atributos.

2.1. Pré-processamento dos Dados

O banco de dados *Poker Hands* foi obtido do repositório de aprendizado de máquina [11], sendo dividido previamente em um conjunto de treino (matriz com dimensões 25010x11), e outro de teste (matriz com dimensões 10^6x11) Cada padrão de treinamento é uma coleção de cinco cartas retiradas de um baralho comum de 52 cartas. Cada carta é, portanto, descrita como dois atributos (naipe e valor) de um total de 5 cartas, resultando num total de 10 atributos. O último atributo representa a *label* do respectivo padrão.

O atributo de decisão é descrito como numerais de 0 a 9. O numeral nulo representa a falta de jogos em mãos, 1 denota um jogo de carta alta, 2, 3, 4, 5, 6, 7, 8 e 9 denotam um par, dois pares, uma trinca, *straight*, *flush*, *full house*, quadra, *straight flush* e *royal flush*, respectivamente.

Os padrões, portanto, são da forma (C1, C2, C3, C4, C5). Cada elemento C no padrão de entrada denota uma carta, e é representado por seu naipe e seu valor. Os naipes podem ser qualquer um entre paus, ouros, copas e espadas, sendo representados por numerais entre 1 e 4, enquanto que os valores das cartas são representadas por numerais entre 1 e 13, significando ás (1), 2, 3, ..., 10, valetes (11), damas (12) e reis (13). Baseado nisso, os padrões de entradas podem ser representados como uma tupla de 10 elementos: (N1, V1, N2, V2, N3, V3, N4, V4, N5, V5), onde o Ni indica o naipe da i-ésima carta e cada Vi indica o valor da i-ésima carta, como mostra a Figura 1. Além disso, a Figura 2 mostra o histograma de todos os atributos, incluindo as *labels*.

Para o presente trabalho, iremos utilizar quatro vertentes do banco de dados *Poker Hands*. D_1) O *dataset* (conjuntos de treino e teste) original, sem nenhuma alteração. D_2) *Dataset* sem as duas primeiras classes (0,1), representando mãos que não possuem nenhum jogo, e jogo de carta mais alta, respectivamente. O motivo desse tipo de processamento, é que devido a distribuição irregular dos padrões, onde aproximadamente 42% dos padrões são da classe 1 (um par) e classe 0 (nada na mão). D_3) Um esquema de codificação binária nos conjuntos do *dataset* é utilizado, onde cada atributo é codificado na forma *one-hot*. Por exemplo, o naipe coração é representado por {1 0 0 0}, espada é codificado para {0 1 0 0}, etc. O mesmo ocorre para os valores das cartas, codificados no mesmo esquema, por binários de tamanho 13 (número de valores possíveis de um determinado naipe), tal que um ás pode ser representado por {1 0 0 0 0 0 0 0 0 0 0 0 0}, um 2 por {0 1 0 0 0 0 0 0 0 0 0 0 0}, e assim sucessivamente, resultando em 85 atributos. D_4) Por fim, a última vertente testada foi dividida em duas partes. A primeira é a junção dos conjuntos de treino e teste do *dataset*, ja a segunda utiliza o método de *holdout* de validação cruzada, destinando 30% aleatório do *dataset* agrupado para o conjunto de teste.

Com isso, as dimensões das matrizes dos *datasets* são ilustradas na tabela 2.

	D_1	D_2	D_3	D_4
Treino	25010x11	1918x11	25010x85	1025010x11
Teste	10^6x11	76293x11	10^6x85	307503x11

Tabela 1. Dimensões das matrizes dos quatro *dataset* analisados.

2.2. Especificações dos Modelos

Um passo importante na construção de redes neurais é a escolha de hiperparâmetros. Dessa forma, a escolha do número de camadas intermediárias e o número de neurônios devem ser escolhidos cuidadosamente [13]. De uma forma geral, foi adotada uma topologia da forma $(N_{in}, 52, 26, N_{out})$, onde N_{in} é o número de atributos de entrada e N_{out} os de saída. Portanto, a topologia do *dataset* original D_1 pode ser representado na Figura 3.

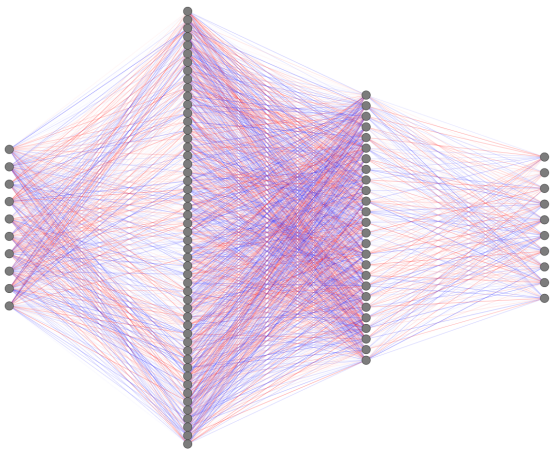

Figura 3. Topologia de D_1.

A Figura 4 ilustra a topologia do *dataset* sujeito ao esquema de codificação D_3.

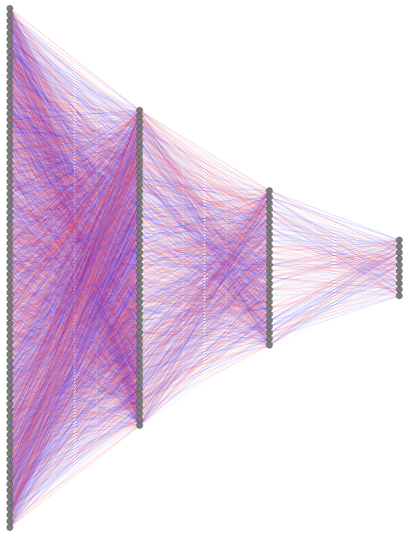

Figura 4. Topologia de D_3.

As funções de ativação foram padronizadas como sendo a tangente hiperbólica, por na prática, geralmente, ter um aprendizado mais rápido [8]. A escolha do número de neurônios da primeira camada intermediária ser 52 foi arbitrária, na esperança do modelo aprender a identificar todas cartas em diferentes neurônios. Além disso, o algoritmo de otimização escolhido foi o *Adam*, uma combinação entre RMSProp e SGD com momentum, onde utiliza os quadrados dos gradientes para escalar a taxa de aprendizado como RMSprop, e aproveita o momento usando a média móvel do gradiente em vez do próprio gradiente como SGD com momentum [3]. Por fim, a taxa de aprendizado foi fixada em $\alpha = 0.01$.

3. Resultados e Análises

Nessa seção, os principais resultados e análises são apresentados. As curvas foram feitas em *python*, e todo o código gerador pode ser encontrado no meu repositório pessoal [4]. É observado que os resultados não são satisfatórios com apenas uma camada intermediária e o número de neurônios menor que 10 [5]. O treino das redes neurais só tiveram uma performance satisfatória com três ou mais camadas, e o número de neurônios maior ou igual a 10. Para evitar o *overfitting* (termo dado para modelos que generalizam mal), foi utilizado, em todas as arquiteturas das redes, um esquema de validação cruzada *k-fold cross validation* de 5 pastas. Somente para as redes do *dataset* D_4 que a estratégia de validação foi a de *holdout*, com a proporção 70-30 para os conjuntos de treino e teste.

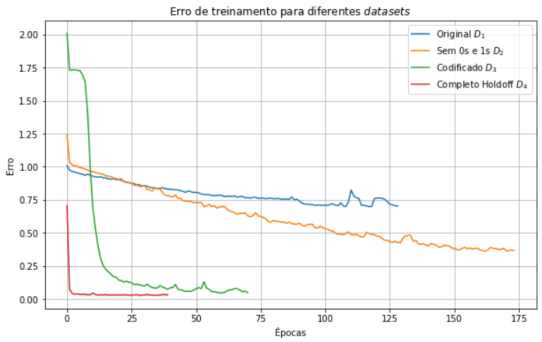

Figura 5. Erros de treinamento dos diferentes bancos de dados.

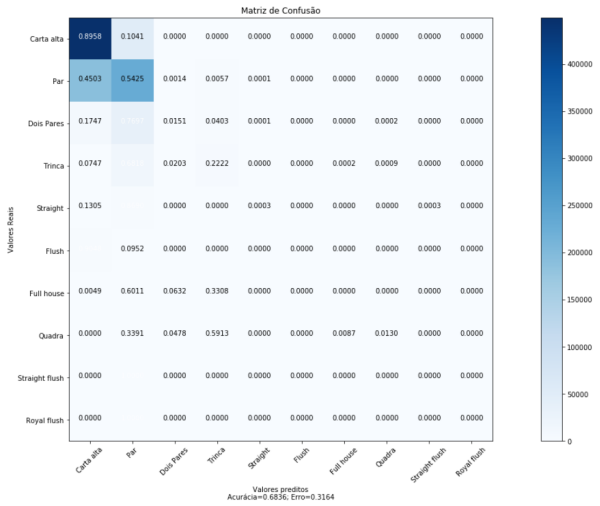

Figura 6. Matriz de confusão para D_1.

A Figura 5 mostra a progressão do erro (logarítmico) em função das épocas de treinamento. As curvas em azul, laranja, verde e vermelho são referentes aos *datasets* D_1, D_2, D_3 e D_4, respectivamente. É imporante notar que, após a época 25, o *dataset* D_2 sem as classes $\{0$ e $1\}$ começa a ter um erro de treinamento menor que o modelo que treinou o *dataset* original D_1. O treinamento dos dados codificados D_3 obteve uma melhora significativa na performance em relação aos sem condificação, chegando no mínimo local de forma mais rápida e eficaz. Surpreendentemente o modelo de treinamento dos dados com o esquema de validação *holdout* obteve também uma ótima performance.

	D_1	D_2	D_3	D_4
Acurácia	68%	57%	97%	99%

Tabela 2. Acurácia (no conjunto de teste) dos quatro *datasets* analisados.

É imediato notar que, para os *datasets* D_3 e D_4, os modelos de treinamento obtiveram uma acurácia excelente junto ao conjunto de teste. Em vista disso, a matriz de confusão da Figura 6 nos informa que, para os bancos de dados testados, os modelos conseguem classificar um par de maneira satisfatória. Pode haver diversos motivos pelo qual o modelo está com uma performance tão baixa. Poderia estar preso em um mínimo local que identifica apenas pares e nada mais. Ademais, a falta de amostras para classes "altas"(como *royal flush*) pode gerar uma desbalanceamento e acabar viesando o modelo.

Por fim, a Figura 7 trás um comparativo da métrica de Acurácia para diferentes algoritmos e comitês que operam sob o paradigma de aprendizagem supervisionada. O resultado nos mostra que, para esse conjunto de dados, redes MLP possuem a melhor estrutura para essa tarefa.

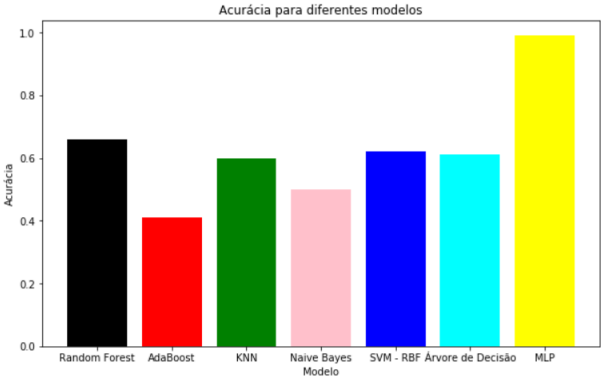

Figura 7. Acurácia para diferentes algoritmos de aprendizado.

4. Conclusões

Resultados experimentais mostraram o impacto que o pré-processamento nos dados pode causar na performance geral, em algoritmos de aprendizado supervisionado. Além disso, redes neurais MLP possuem uma melhor performance de classificação do que outros métodos de aprendizado, para essa tarefa no banco de dados de jogos de poker.

Além disso, foi discutido uma das possíveis razões em que o *dataset* é considerado difícil de trabalhar em cima. Esse conjunto de dados pode se beneficiar de um processo de otimização que pondera

erros de previsão pela proporção inversa da frequência da classificação do registro. Isso forçaria o modelo a considerar as raras mãos de pôquer no começo, em vez de fazer os ganhos mais rápidos se encaixarem no jogo de um par. Suspeita-se que o modelo se estrutura para identificar pares, mas não pode se afastar disso para identificar outras mãos também. Essencialmente, fica preso em um grande mínimo local.

Referências

[1] Emile Borel. On games that involve chance and skill of the players. *Econometrica*, 1953.

[2] Jasmine Daly. K-means cluster analysis of poker hands in python. `https://jasminedaly.com/2016-05-25-kmeans-analysis-in-python/`. (acessado em 20/06/2019).

[3] Jimmy Ba Diederik P. Kingma. Adam: A method for stochastic optimization. *3rd International Conference for Learning Representations*, 2014.

[4] Samuel Gomes. Github personal repository. `https://github.com/SamuelBFG/IA006`. (acessado em 20/06/2019).

[5] Suraiya Jabin. Poker hand classification. *International Conference on Computing, Communication and Automation (ICCCA)*, 2016.

[6] L Shapley John Nash. Essays on game theory. *Edward Elgar Publishing*, 35(10):91, 1996.

[7] Oskar Morgenstern John Vohn Neumann. Theory of games and economic behavior. *Princeton university press*, 2007.

[8] Tom Mitchell. Machine learning. *McGraw Hill 45*, 1997.

[9] David Blackwell Richard Bellman. Some two-person games involving bluffing. *National Academy of Science*, 35(10):600–605, 1949.

[10] Annija Rupeneite. Building poker agent using reinforcement learning with neural networks. *(Science and Technology Publications*, 2014.

[11] Vo Hoang Trong. Decision tree versus multilayer perceptron in classification problems.

[12] Vo Hoang Trong. Decision tree versus multilayer perceptron in classification problems. *Chonnam National University*.

[13] Zhen Wong. Automatic hyperparameter tuning of machine learning models under time constraints. *IEEE BigData 2018 workshop*.

[14] Jabin Zareen. A comparative study of the recent trends in biometric signature verification. *IEEE Computer Graphics and Applications*, pages 354–358, 2013.

[15] Jabin Zareen. Biometric signature verification. *International Journal of Biometrics*, 7(2):97–118, 2015.

[16] Jabin S Zareen, F. J. An authentic mobile-biometric signature verification system, iet biometrics. *International Journal of Biometrics*, 1(2):97–118, 2016.