UNIVERSIDADE FEDERAL DO PARANÁ

MATEUS NORONHA DOS SANTOS

TÉCNICAS DE INTELIGÊNCIA ARTIFICIAL APLICADAS À PREVISÃO DA CLASSIFICAÇÃO FINAL DO CAMPEONATO BRASILEIRO DE FUTEBOL

CURITIBA
2018

MATEUS NORONHA DOS SANTOS

TÉCNICAS DE INTELIGÊNCIA ARTIFICIAL APLICADAS À PREVISÃO DA CLASSIFICAÇÃO FINAL DO CAMPEONATO BRASILEIRO DE FUTEBOL

Trabalho de conclusão de curso apresentado ao Curso de Bacharelado em Matemática Industrial, do Departamento de Matemática, Setor de Ciências Exatas, Universidade Federal do Paraná, como parte das exigências para a obtenção do título de Bacharel em Matemática Industrial.

Orientadora: Professora. Dra. Mariana Kleina

CURITIBA
2018

RESUMO

Este trabalho apresenta uma previsão de classificação em grupos para as equipes do campeonato brasileiro de futebol tanto da série A quanto da série B a partir dos resultados do primeiro turno de cada campeonato. Aplicam-se as técnicas rede neural artificial *Multilayer Perceptron* (MLP) e *Support Vector Machine* (SVM) para obter tal classificação. Resultados indicam que para os dados de treinamento, a rede neural apresenta raiz do erro médio quadrático (REMQ) de aproximadante 1 e coeficiente de correlação de 0,7, já para os dados de validação, tem-se REMQ de 0,96 e correlação de 0,78. Analisando os resultados do SVM, tem-se REMQ de 0,54 e correlação de 0,92 para os dados de treinamento e para a validação 1,13 para a REMQ e 0,69 para a correlação.

Palavras-chave: Redes Neurais Artificiais, Previsão de Jogos, Rede Neural MLP, *Support Vector Machine*.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

MLP	*Multi-Layer Perceptron*
RBF	*Radial Basis Function*
REMQ	Raiz do Erro Médio Quadrático
RNAs	Redes Neurais Artificiais
SVM	*Support Vector Machine*

SUMÁRIO

1 INTRODUÇÃO

O futebol é considerado o esporte mais popular do mundo, além disso, ele é o mais valorizado no Brasil, visto que é o quinto país com o maior faturamento nesse esporte, ficando atrás da França(4°), Itália(3°), Alemanha(2°) e Inglaterra(1°). O futebol brasileiro teve um faturamento de R$545 milhões de reais em 2017, onde 65% são originados por meio de patrocínios (CAPELO, 2018). A Confederação Brasileira de Futebol (CBF) é responsável pela administração da verba, segundo Capelo (2018) R$530 milhões usados em despesas no ano de 2017, 2% foram destinados à série B e apenas 1% a série A. Além das séries A e B, existem outros campeonatos durante o ano que também recebem investimentos, por exemplo, Copa do Brasil (R$ 13,3 milhões); Taça Libertadores (R$ 24,7 milhões) e Mundial de clubes (R$ 15,5 milhões), onde esses valores se referem apenas aos campeões, sendo também remunerados aqueles que finalizam o campeonato até uma certa posição, de acordo com a competição que está participando (OHATA, 2017).

O Campeonato Brasileiro (série A e B) é composto por 40 equipes de todo o país, onde, em cada série eles disputam entre si dois turnos, em jogos de ida (partidas no estádio de residência do adversário) e volta (partidas no estádio de residência da própria equipe), com o objetivo de somar o maior número de pontos. O formato dos pontos corridos começou no ano de 2003 e até hoje é utilizado como forma de quantificar o desempenho de uma equipe no campeonato. O time vencedor de uma partida recebe 3 pontos; no caso de empate cada time recebe 1 ponto e o time derrotado não pontua. Ao final das 38 rodadas, o time com maior pontuação é o campeão, e os 4 últimos colocados são rebaixados para a série imediatamente inferior (essa regra vale para as séries A e B), os 4 quatro primeiros colocados são promovidos a série A. Além da pontuação, outros fatores são importantes para critério de desempate, tais como

saldo de gols, gols marcados, número de cartões, entre outros.

Desde o início do Campeonato Brasileiro, ano após ano, vários times participam da disputa com o propósito de alcançar a liderança ou alcançar posições que favoreçam a participação no mesmo campeonato em anos seguintes. Desde o início de cada ciclo competitivo surgem especulações de quem será o campeão, ou quais serão os times que encerrarão o torneio nas posições mais almejadas que oferecem benefícios diferenciados aos mais bem colocados, como a oportunidade de participar de campeonatos mais importantes além do retorno financeiro. Isso faz com que essas especulações e projeções sejam cada vez mais estudadas e observadas, não apenas pelos torcedores dos times, mas também para empresas patrocinadoras.

A aplicação das técnicas de inteligência artificial para a previsão de resultados esportivos já acontece há alguns anos e tem se difundido cada vez mais, principalmente com o avanço da tecnologia e pela quantidade imensurável de dados coletados e disponíveis para estudos dessa espécie.

1.1 JUSTIFICATIVA

Uma previsão com boa acurácia é interessante, pois poderá auxiliar futuras projeções, possíveis alterações nas estratégias das equipes, entre outros benefícios. A proposta deste trabalho é apresentar uma previsão que informará em qual intervalo de posições uma equipe encerrará o campeonato dada sua posição, pontuação, saldo de gols e número de vitórias ao final do primeiro turno (metade de todo o tempo da competição). Portanto, essa previsão se daria com antecedência de 19 rodadas do campeonato.

As técnicas de aprendizado de máquina Redes Neurais Artificiais (RNA's) e *Support Vector Machine* (SVM) são amplamente utilizadas nas mais diversas áreas, tais como saúde, economia, meteorologia, entre outras. Vêm sendo empregadas com sucesso e serão utilizadas neste trabalho para realizar as projeções da classificação final dos times.

1.2 OBJETIVOS

1.2.1 Objetivo Geral

O objetivo geral deste trabalho é utilizar as técnicas de inteligência artificial, rede neural artificial Perceptron de Múltiplas Camadas e *Support Vector Machine*, para prever com o menor erro possível, em qual grupo uma equipe ficará dada a sua posição, pontuação, número de vitórias e de gols na metade da competição.

1.2.2 Objetivos Específicos

Para alcançar o objetivo geral, objetivos específicos foram traçados:

- Coleta de dados dos times das séries A e B de 2003 a 2018;
- Estudar a teoria das técnicas de RNA's e SVM;
- Aprimorar conhecimentos em programação com a linguagem *Python*, onde será executada a metodologia proposta;
- Escolha de parâmetros para as técnicas;
- Testes numéricos e análise de erro.

1.3 LIMITAÇÕES DO TRABALHO

Este trabalho se atém à:

- Utilizar dados disponíveis na internet sobre os campeonatos brasileiros da série A e série B no período de 2003 a 2018.

1.4 DISPOSIÇÃO DO TRABALHO

O presente trabalho está disposto de tal forma que no Capítulo 2 encontram-se as teorias das técnicas utilizadas assim como citações de trabalhos relevantes que

utilizaram-se das mesmas técnicas com objetivos similares. No Capítulo 3, têm-se a descrição de como foram aplicadas as técnicas bem como a forma que se optou para escolher a melhor configuração de cada uma, em seguida, no mesmo capítulo é feita uma classificação do campeonato brasileiro de futebol de 2018 a fim de comparar todos os resultados. Após os resultados, segue a conclusão do trabalho desenvolvido e as referências utilizadas.

2 REVISÃO DA LITERATURA

2.1 REDES NEURAIS ARTIFICIAIS

Redes Neurais Artificiais, por hora chamadas apenas de “redes neurais” são técnicas que ganham cada vez mais espaço tanto na pesquisa quanto na indústria devido a sua capacidade de agir, mesmo que em uma escala mínima, como o cérebro humano. As redes neurais mais utilizadas foram inspiradas no aprendizado através da experiência, fenômeno que acontece com o cérebro através de sinais transmitidos por neurônios dentro do sistema nervoso. Com isso, tais técnicas são capazes de realizar previsões, reconhecimento de padrões e até mesmo gerir sistemas computacionais.

A grande capacidade das redes neurais possibilita modelar problemas complexos, tanto lineares quanto não-lineares (HAYKIN, 2001).

2.1.1 Definição de Neurônio

Um neurônio é responsável por processar informações em uma rede neural. A Figura 1 mostra o modelo de um neurônio base. Podem ser identificados alguns elementos:

1. Um conjunto de *elos de conexão*, cada um caracterizado por um peso.
2. Um *somador* $\sum$ para as informações de entrada, ponderadas pelos respectivos pesos.
3. Uma *função de ativação* φ para delimitar a amplitude da saída das informações do neurônio.

O exemplo da Figura 1 também inclui um *bias*, representado por b_k que pode aumentar ou diminuir o dado da função de ativação.

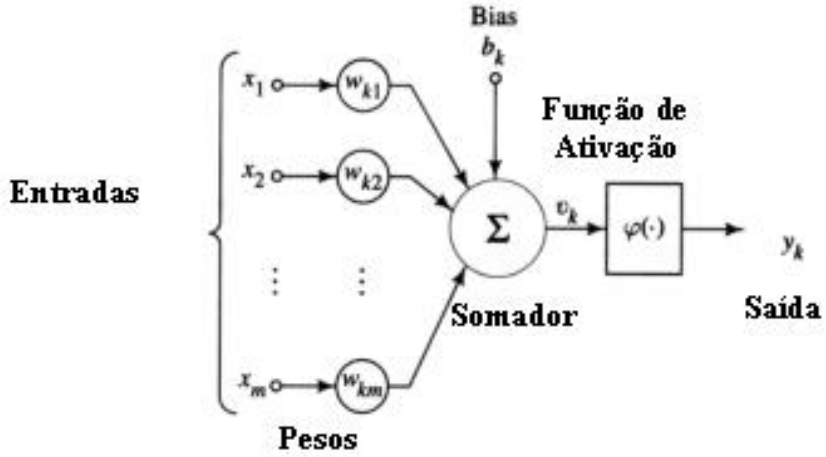

FIGURA 1: Exemplo de um neurônio
FONTE: Haykin (2001)

Na linguagem matemática, pode-se apresentar um neurônio seguindo as equações 1 e 2 :

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \tag{1}$$

e

$$y_k = \varphi(u_k + b_k) \tag{2}$$

onde $x_1, x_2, \ldots, x_m$ são as informações de entrada; $w_{k1}, w_{k2} \ldots, w_{km}$ são os pesos do neurônio k; u_k é a saída da combinação linear devido aos valores de entrada; b_k é o bias; $\varphi(\cdot)$ é a função de ativação; e y_k é a saída do neurônio. Utilizar o bias tem o efeito de aplicar uma transformação afim à saída u_k, conforme a equação 3

$$v_k = u_k + b_k \tag{3}$$

2.2 PERCEPTRON DE MÚLTIPLAS CAMADAS

Os perceptrons de múltiplas camadas (*Multi-Layer Perceptron* - MLP) têm sido aplicados com sucesso em diversos problemas através de um treinamento supervisionado e um algoritmo conhecido, o algoritmo de retropropagação (*back-propagation*) baseado na regra de aprendizagem pela correção do erro. Essa aprendizagem é basi-

camente composta por dois passos: um passo para frente, chamado de *propagação*, e um para trás, *retropropagação*. No primeiro passo, um padrão de entrada é aplicado aos nós da rede e seu efeito se propaga camada por camada. Por fim, a saída é produzida com a resposta real da rede. No decorrer do passo de propagação, os pesos permanecem fixos, e durante o passo para trás os pesos são todos ajustados de acordo com a regra de correção do erro. De forma específica, a resposta da rede é subtraída da resposta desejada para produzir um sinal de erro. Então, na retropropagação os pesos são ajustados fazendo com que a resposta real da rede se aproxime da resposta desejada (HAYKIN, 2001).

2.2.1 Características da rede MLP

1. O modelo de cada neurônio inclui uma função de ativação não-linear. Enfatizando que a função é diferenciável em qualquer ponto.
2. A rede possui uma ou mais camadas de neurônios ocultos, que não compõem a parte da entrada ou saída da rede.
3. Alto grau de conectividade, determinado pelas ligações da rede.

Combinando essas características com a habilidade de aprender através do treinamento, tem-se a rede neural MLP.

A Figura 2 apresenta a arquitetura de um perceptron de múltiplas camadas com uma camada de saída e uma oculta. A rede exibida aqui é totalmente conectada, significando que um neurônio qualquer está ligado a todos os nós da camada anterior.

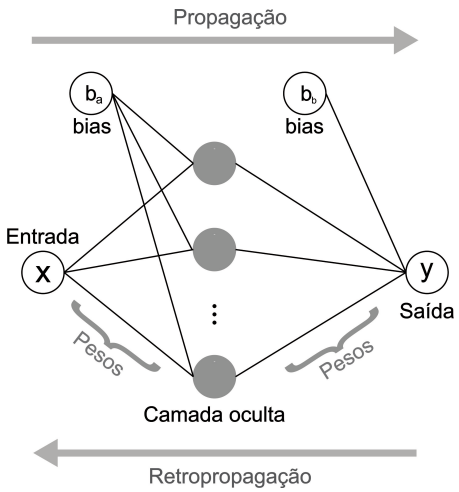

FIGURA 2: Arquitetura de uma rede neural perceptron de múltiplas camadas
FONTE: Santos (2014)

Cada nó de uma camada oculta ou de saída é desenvolvido para cumprir dois cálculos (HAYKIN, 2001):

1. O cálculo do sinal funcional que aparece na saída de um neurônio, que é uma função não-linear do sinal de entrada e dos pesos relacionados com aquele neurônio.

2. O cálculo de uma estimativa do vetor gradiente, necessário para a retropropagação pela rede.

O algoritmo de retropropagação pode ser encontrado no Apêndice A.

2.2.2 Função de Ativação - Tangente Hiperbólica

O modelo de cada unidade da rede pode incluir uma não-linearidade na sua saída. É importante enfatizar que a não-linearidade deve ser suave, ou seja, diferenciável, diferentemente da função sinal utilizada pelo perceptron original.

A função de ativação representa o efeito que a entrada e o estado atual de ativação exercem na definição de ativação da próxima camada. Quando propriedades dinâmicas estão envolvidas na definição do estado de ativação, equações diferenciais (caso

contínuo) ou a diferenças (caso discreto) são empregadas. Tendo em vista a simplicidade desejada para as unidades processadoras, geralmente define-se seu estado de ativação como uma função algébrica da entrada atual, independente de valores passados do estado de ativação ou mesmo da entrada. Geralmente, esta função é monotonicamente não-decrescente e apresenta um tipo de não-linearidade associada ao efeito da saturação (HAYKIN, 2001).

A função logística é uma das mais utilizadas em redes do tipo MLP, porém, ela apresenta valores de ativação apenas no intervalo (0,1), então ela pode ser substituída pela função tangente hiperbólica, que preserva a forma sigmoidal da função logística, mas assume valores positivos e negativos. A função tangente hiperbólica e sua derivada são dadas pelas equações em 4:

$$f(x) = \frac{e^{px} - e^{-px}}{e^{px} + e^{-px}} = tanh(px), \qquad f'(x) = p(1 - f(x)^2) \tag{4}$$

Na Figura 3 tem-se a função tangente hiperbólica (a) e a sua derivada (b).

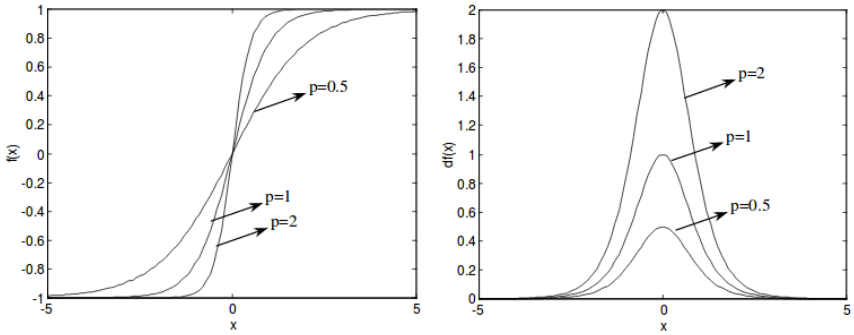

FIGURA 3: (a) Função tangente hiperbólica. (b) Sua derivada em relação a entrada.
FONTE: Silva (1998)

2.2.3 Critério de Parada

Para estabelecer um critério de parada, procura-se por um mínimo global ou local na superfície de erro. Dessa forma, suponha que w* seja um vetor de pesos e represente um mínimo, global ou local. A condição necessária para que w* de fato seja

mínimo é que o gradiente g(w) da superfície em relação ao peso w seja zero para w = w*. Pode-se então, formular um critério de convergência para a aprendizagem por retropropagação como a seguir (HAYKIN, 2001):

Considera-se que o algoritmo de retropropagação tenha convergido quando a norma euclidiana do vetor gradiente alcançar um limiar suficientemente pequeno.

Este critério pode causar problemas quando procura-se por tentativas bem sucedidas, pois os tempos de aprendizagem podem ser longos. Além de precisar calcular o vetor gradiente.

Um critério diferente pode ser apresentado, visto que a função de medida de erro $\varepsilon_{med}(\mathrm{w})$ é estacionária em w = w*, é o seguinte:

Considera-se que o algoritmo de retropropagação tenha convergido quando a taxa absoluta de variação do erro médio quadrático por iteração for suficientemente pequena. (HAYKIN, 2001)

Esta taxa do erro médio quadrático normalmente é considerada pequena se estiver no intervalo de $0,1$ a 1 por cento.

2.3 SUPPORT VECTOR MACHINE

A técnica SVM (*Support Vector Machine*) ou sua nomenclatura traduzida não muito utilizada, Máquinas de Vetor de Suporte, é fundamentada na teoria estatística. Seu desenvolvimento fez-se com o objetivo de classificar padrões, em outras palavras, através de um treinamento a partir de um banco de dados, é possível obter uma classificação satisfatória em muitos casos. O SVM tem a capacidade de classificar dados linearmente separáveis ou não-linearmente separáveis. Na Figura 4 pode-se acompanhar os dois casos, respectivamente.

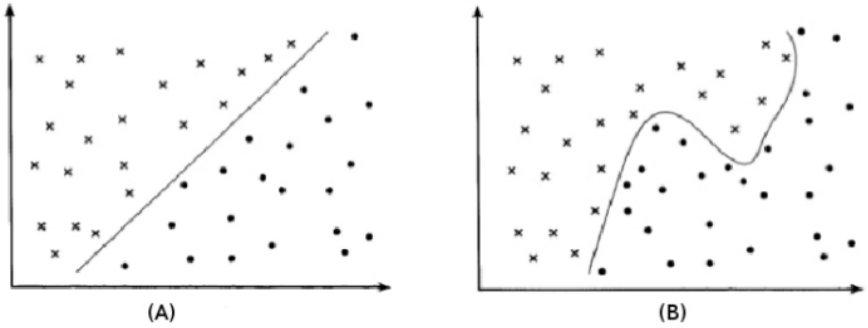

FIGURA 4: Padrões linearmente separáveis (A) e não-linearmente separáveis (B)
FONTE: Gonçalves (2009)

Quando trabalha-se com padões linearmente separáveis, o SVM constrói um hiperplano que divide os padrões, buscando a separação máxima.

A Figura 5 apresenta dois conjuntos de treinamento simbolizados por “quadradinhos” e “bolinhas”, e os hiperplanos que atuam na separação das classes. O melhor hiperplano, em vermelho, é o que maximiza a distância entre os pontos de cada classe.

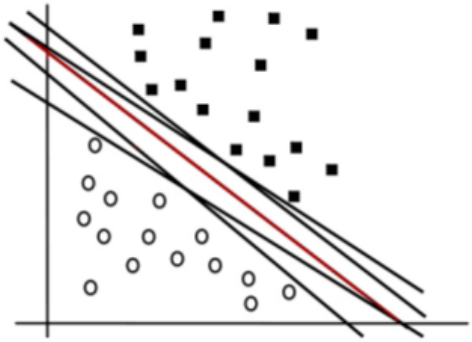

FIGURA 5: Hiperplanos de separação
FONTE: Liu (2007)

Por outro lado, no caso não-linear, o SVM utiliza uma função kernel que mapeia os padrões de treinamento para um espaço linear de alta dimensão, e a partir disso, encontra o hiperplano ótimo que separa as classes.

2.3.1 SVM com Margens Rígidas - Caso Linear

Considere dados de treinamento linearmente separáveis, com n vetores x_i e seu par resposta y_i. Na busca pelo hiperplano, encontra-se uma função linear da forma

$$f(x) = \langle w, x \rangle + b \tag{5}$$

sendo $\langle w, x \rangle$ o produto escalar entre vetores. O vetor de pesos, chamado de w, é normal ao hiperplano e com a variação de b (bias) é movido paralelamente a ele mesmo.

O SVM pode classificar os padrões em dois ou mais conjuntos, no caso em que tem-se duas classes, devido à metodologia de margens de separação para os conjuntos, que são deslocamentos com distância igual a 1 ($f(x) = \pm 1$), tendo o conjunto de classe -1 e outro de classe $+1$. Tais margens estão ilustradas na Figura 6 pelos hiperplanos H_1 e H_2. Dessa forma, um padrão é atribuido à classe positiva se $f(x_i) \geq 0$ e à outra classe no caso contrário. Esse padrão é considerado classificado de forma correta se estiver fora da margem de separação, ou seja, se $y_i = +1$ então deve-se ter que $\langle w, x_i \rangle + b \geq 1$, para a outra classe, trabalha-se de forma análoga. Tem-se então, como restrição do SVM:

$$(\langle w, x_i + b \rangle) y_i, \ \ \forall i = 1, ..., n \tag{6}$$

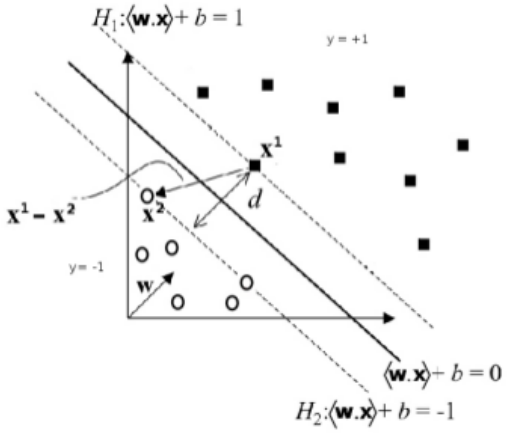

FIGURA 6: Distância d entre os hiperplanos
FONTE: Oliveira (2012)

O foco a seguir é encontrar o hiperplano que separa os dados com a maior distância. Sendo assim, toma-se x_1 um ponto do hiperplano $H_1 : \langle w,x\rangle + b = 1$ e x_2 um ponto em $H_2 : \langle w,x\rangle + b = -1$. A distância nomeada d entre os hiperplanos é encontrada pela projeção de $(x_1 - x_2)$ na direção de w, definido nas equações 7, 8 e 9:

$$d = proj_w(x_1 - x_2) = \frac{\langle (x_1 - x_2), w\rangle w}{\|w\|^2} = \frac{w, \langle (x_1 - x_2)\rangle w}{\|w\|^2} = \frac{(\langle w,x_1\rangle - \langle w,x_2\rangle)w}{\|w\|^2} \quad (7)$$

Sabendo que $\langle w,x_1\rangle + b = 1$ e $\langle w,x_2\rangle + b = -1$, tem-se:

$$d = \frac{(1 - b - (-1 - b))w}{\|w\|^2} = \frac{2w}{\|w\|^2} \quad (8)$$

Logo, o comprimento de d é:

$$\|d\| = \left\| \frac{2w}{\|w\|^2} \right\| = \frac{2\|w\|}{\|w\|^2} = \frac{2}{\|w\|} \quad (9)$$

Para minimizar a distância encontrada, busca-se minimizar $\|w\|$, definindo-se o problema de otimização:

$$Minimizar_{w,b} = \frac{1}{2}\|w\|^2$$

sujeito a $\quad (\langle w,x_i\rangle + b)y_i - 1 \geq 0, \qquad \forall i = 1,...,n. \quad (10)$

Pode-se então, associar ao problema 10, a função Lagrangeana:

$$L(w,b,\mu) = \frac{1}{2}\|w\|^2 - \Sigma_{i=1}^{n}((\langle w,x_i\rangle + b)y_i - 1) \tag{11}$$

Aplicando-se as condições de Karush-Kuhn-Tucker (KKT) para o problema primal 10, obtém-se:

$$\nabla_w L = 0 \Longleftrightarrow w = \Sigma_{i=1}^{n}\mu_i y_i x_i \tag{12}$$

$$\nabla_b L = 0 \Longleftrightarrow w = \Sigma_{i=1}^{n}\mu_i y_i = 0 \tag{13}$$

A função objetivo é convexa e possui um único mínimo global. Pode-se de forma equivalente resolver o problema dual, maximizando o lagrangeano (LORENA A. C.; CARVALHO, 2007).

2.3.2 Caso Não-Linear - Funções Kernel

A técnica SVM para problemas não-lineares, introduz funções reais que mapeiam o conjunto de treinamento (dados de entrada) para um espaço linearmente separável. Este novo espaço é de alta dimensão onde serão mapeados os dados de entrada por meio de uma função Φ, obtendo um novo conjunto de dados linearmente separável, representado por ($\Phi(x_1), y_1$),...,($\Phi(x_n), y_n$)). Substituindo o mapeamento $\Phi(x)$ no lugar de x no problema primal 10, obtém-se o problema dual a seguir:

$$Maximizar_{\mu} = \Sigma_{i=1}^{n}\mu_i - \frac{1}{2}\Sigma_{j=1}^{n}\mu_i\mu_j y_i y_j \langle\Phi(x_i),\Phi(x_j)\rangle$$

$$\text{sujeito a} \qquad \mu_i \geq 0, i = 1,...,n, \qquad \Sigma_{i=1}^{n}\mu_i y_i = 0 \tag{14}$$

Obtém-se então a função que representa o hiperplano ótimo:

$$g(x) = sgn(f(x)) = sgn(\Sigma_{x_j \in SV}\mu_i^* y_i \langle\Phi(x_i),\Phi(x)\rangle + b^*) \tag{15}$$

Como este novo espaço pode ter dimensão muito alta, o cálculo de Φ pode ser demorado ou até inviável. Devido a isso, são utilizadas funções kernel, que têm a capacidade de representar espaços abstratos e simplificar os cálculos, dessa forma,

não é necessário conhecer o mapeamento Φ, que é gerado de forma implícita.

Essas funções recebem dois pontos x_i e x_j do espaço de entradas e calculam o produto euclidiano desses dados:

$$K(x_i, y_i) = \langle \Phi(x_i), \Phi(x_j) \rangle \tag{16}$$

Para garantir a convexidade do problema de otimização 14 utilizam-se funções que obedecem as condições estabelecidas pelo Teorema de Mercer, onde diz que para uma função ser dita função de kernel, ela precisa dar origem a matrizes M, semidefinidas positivas, em que cada elemento m_{ij} é definido por $m_{ij} = K(x_i, y_i) = \langle \Phi(x_i), \Phi(x_j) \rangle, \forall i = 1, ..., n$ (LORENA A. C.; CARVALHO, 2007). As funções kernel mais utilizadas são: Linear, Gaussiana, Sigmoidal e Polinomial.

2.4 TRABALHOS CORRELATOS

Na literatura encontram-se diversos trabalhos de pesquisa que envolvem técnicas de aprendizado de máquina aplicas à previsão de jogos esportivos. No trabalho de Fernandes (2015) fez-se um desenvolvimento da rede MLP para prever a quantidade de gols que pode acontecer em uma partida. Segundo o autor, os resultados foram satisfatórios, com média de 57,85% de acertos. Sua base de dados originou-se das temporadas de 2013/2014 e 2014/2015 do campeonato inglês.

Em outra pesquisa, desenvolvida por Chang (2010) a rede MLP é utilizada para verificar a previsão de vitórias dos jogos da Copa do Mundo de Futebol de 2006. Fez-se uso de uma rede neural com arquitetura de 8 neurônios de entrada, uma camada intermediária com 11 neurônios e uma única saída. Sua acurácia para essa previsão foi de 76,9% e seus conjuntos de dados são informações estatísticas anteriores à competição.

Na pesquisa desenvolvida por Bezerra (2014) utilizou-se as técnicas de mineração de dados SVM, *Random Forest* (RF) e árvore *C4.5* (*Weka J48*). Seus dados proce-

dentes do campeonato brasileiro de futebol de 2013 e 2014. A intenção era prever os resultados de uma partida da série A. Alguns resultados, mostram que o classificador RF foi o que obteve os melhores resultados, com taxa de acerto de aproximadamente 90%. O melhor resultado para o SVM foi uma taxa de 89% de acerto.

Em outro trabalho, feito por Schmidt (2017) fez-se uma aplicação do SVM, de redes neurais e *Random Forest* para prever resultados de jogos de futebol tendo como dados de entrada a posição de cada equipe na rodada atual, na temporada anterior e a média de gols marcados e sofridos. Chegou-se a um resultado de 58,77% de acerto, seu objetivo era dizer se uma equipe venceria, empataria ou perderia a partida. O melhor resultado foi encontrado com a rede neural, seguida do SVM com resultado de 57,03% de acertos na previsão.

Na pesquisa de Duarte (2015) aplicou-se o SVM linear e *Random Forest* no intuito de prever resultados de jogos de futebol da Liga Portuguesa. Sua taxa de acertos chegou a 59% nos jogos da Liga referente a competição de 2012/2013. São feitas mudanças interessantes nos dados utilizados para o aprendizado das técnicas para obter uma melhor performance. Todas as aplicações apresentaram resultados similares com coeficientes de correlação muito próximos.

3 APLICAÇÕES E RESULTADOS

3.1 BANCO DE DADOS

O período de dados selecionado foi de 2003 à 2018, considerando os campeonatos das séries A e B do campeonato brasileiro de futebol. Todos os dados encontram-se na internet e foram retirados das seguintes fontes em Outubro de 2018: globo.com; uol.com.br; cassiozirpoli.com.br; mat.ufmg.br; jornalheiros.blogspot.com; cbf.com.br; diariodepernambuco.com.br; torcedores.com; terra.com.br; futebolinterior.com.br; ogol.com.br; resultados.com; bolanaarea.com.

3.1.1 Dados utilizados

Os dados coletados e utilizados são:

1. **Posição ao final do primeiro turno:** este dado informa qual é a posição da equipe no final do primeiro turno;
2. **Pontuação ao fim do primeiro turno:** este dado informa quantos pontos foram obtidos ao final do primeiro turno;
3. **Saldo de gols:** esta informação apresenta o desempenho prático de cada equipe durante o primeiro turno da competição, estes dados podem sem positivos ou negativos;
4. **Número de vitórias, derrotas e empates:** este dado assim como o anterior, está diretamente relacionado ao desempenho de cada equipe;
5. **Classificação no final do campeonato:** estes dados são utilizados como resposta da técnica para o treinamento. Estas respostas estão separadas nos gru-

pos: 1 (posições de 1°a 4°), 2 (posições de 5°a 8°), 3 (posições de 9°a 12°), 4 (posições de 13°a 16°) e 5 (posições de 17°a 20°).

3.1.2 Filtro

A coleta dos dados referentes à número de vitórias, empates e derrotas, foram sintetizados em uma única variável que considera a soma do número de vitórias com os empates e estes são subtraídos do número de derrotas. Optou-se por essa escolha pois tais dados são de mesma origem e têm o mesmo objetivo na rede neural. Como tanto vitórias quanto empates garantem pontos a mais, optou-se por somá-los.

As informações sobre quantidade de gols feitos e sofridos também são justificados em uma única variável: saldo de gols.

Os dados foram separados entre conjuntos de treinamento e de validação, sendo 70% para treinamento e 30% para validação.

3.2 ÍNDICES DE COMPARAÇÃO

Para fins de comparação dos resultados, utilizou-se a *Raiz do Erro Médio Quadrático* (REMQ) e o Coeficiente de Correlação ρ, todos definidos em Ramírez (2003).

• Raiz do Erro Médio Quadrático (REMQ): é uma das medidas dos padrões de acurácia das estimativas que indica a magnitude média do erro. Esse índice será mais influenciado quando no conjunto de estimativas verificadas existam erros de maior magnitude, mesmo que sejam poucos, do que quando ocorrem muitos erros pequenos, já que ao calcular a potência dois os erros maiores serão realçados. A REMQ é definida como:

$$\text{REQM} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(P_i - P_i^*)^2} \tag{17}$$

onde P_i são os resultados das previsões, P_i^* são os resultados exatos conhecidos e N é o número de padrões.

• Coeficiente de Correlação de Pearson(ρ): é uma medida da dependência linear

entre dois conjuntos de dados, e pode ser expressa por:

$$\rho = \frac{N\sum_{i=1}^{N} P_i P_i^* - \sum_{i=1}^{N} P_i \sum_{i=1}^{N} P_i^*}{\sqrt{\left[N\sum_{i=1}^{N} P_i^2 - \left(\sum_{i=1}^{N} P_i\right)^2\right]\left[N\sum_{i=1}^{N} (P_i^*)^2 - \left(\sum_{i=1}^{N} P_i^*\right)^2\right]}} \tag{18}$$

onde P_i são os resultados das previsões, P_i^* são os resultados exatos conhecidos e N é o número de padrões. Quanto mais próximo de 1 for este índice, melhor é o resultado.

3.3 APLICAÇÃO DA REDE MLP

A aplicação da rede MLP, como já citado, se da através do algoritimo de aprendizagem pela retropropagação do erro de cada sinal de entrada. Para este estudo, aplicou-se a rede MLP da biblioteca *scikit-learn* versão 0.20.1 (Scikit-Learn, 2018), da linguagem de programação Python na versão 2.7.

Com o objetivo de encontrar uma configuração apropriada para os dados em questão, fez-se uma variação do número de neurônios na camada intermediária. Para cada configuração diferente da rede, calculou-se: a raiz do erro médio quadrático, o coeficiente de correlação de Pearson e também estipulou-se alguns indicadores de acertos e erros. Acerto significa que fez-se a classificação da equipe de forma totalmente correta em relação o grupo que ela pertence. Os Acertos ± 1 referenciam classificações que apresentaram um grupo de diferença. Por fim, os Erros contemplam todas as outras classificações que são consideradas muito distantes do esperado.

Na Tabela 1, pode-se encontrar todos estes índices.

Resultados Rede MLP – Treinamento															
Neurônios	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**15**	**20**	**30**	**50**	**100**
REQM	1.14	1.01	1.02	1.05	1.03	1.05	0.98	1.17	1.00	1.00	1.10	0.98	0.99	1.07	1.02
ρ	0.74	0.74	0.75	0.74	0.74	0.74	0.73	0.73	0.76	0.73	0.76	0.74	0.73	0.72	0.76
Acertos(%)	41.89	48.67	46.97	42.62	45.52	45.28	47.46	41.65	47.70	48.43	46.25	48.18	45.76	48.91	44.07
Acertos±1(%)	40.44	36.32	39.47	42.37	41.40	40.44	39.71	35.59	38.98	41.16	36.56	39.47	42.37	34.87	41.65
Erros(%)	17.68	15.01	13.56	15.01	13.08	14.29	12.83	18.89	13.32	10.41	17.19	12.35	11.86	16.22	14.29

TABELA 1: Resultados da rede MLP - Treinamento

FONTE: O autor (2018)

Além da Tabela 1, que apresenta os resultados para o período de treinamento, fez-se uma segunda tabela para os dados de validação. Na Tabela 2 pode-se acompanhar os valores da REMQ, coeficiente de correlação de Pearson e os indicadores de acertos assim como na Tabela 1.

Resultados Rede MLP – Validação															
Neurônios	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**15**	**20**	**30**	**50**	**100**
REQM	1.14	1.12	1.05	1.09	1.07	1.04	0.96	1.05	1.24	1.07	1.13	1.07	1.18	1.04	1.12
ρ	0.73	0.69	0.74	0.73	0.75	0.76	0.78	0.77	0.67	0.75	0.69	0.72	0.72	0.75	0.71
Acertos(%)	45.76	40.68	49.72	44.07	46.33	44.07	42.94	43.50	38.98	44.07	42.94	41.81	33.33	40.11	44.63
Acertos±1(%)	36.72	43.50	35.59	40.11	39.55	42.37	47.46	34.60	39.55	40.11	37.85	45.20	44.63	45.76	39.55
Erros(%)	17.51	15.82	14.69	15.82	14.12	13.56	9.60	21.90	21.47	15.82	19.21	12.99	22.03	14.12	15.82

TABELA 2: Resultados da rede MLP - Validação

FONTE: O autor (2018)

Para encontrar a melhor configuração da rede MLP, analisou-se as Tabelas 1 e 2 e as Figuras 7 e 8 a seguir que apresentam tanto os dados de treinamento quanto de validação para as diferentes arquiteturas da rede neural, respectivamente.

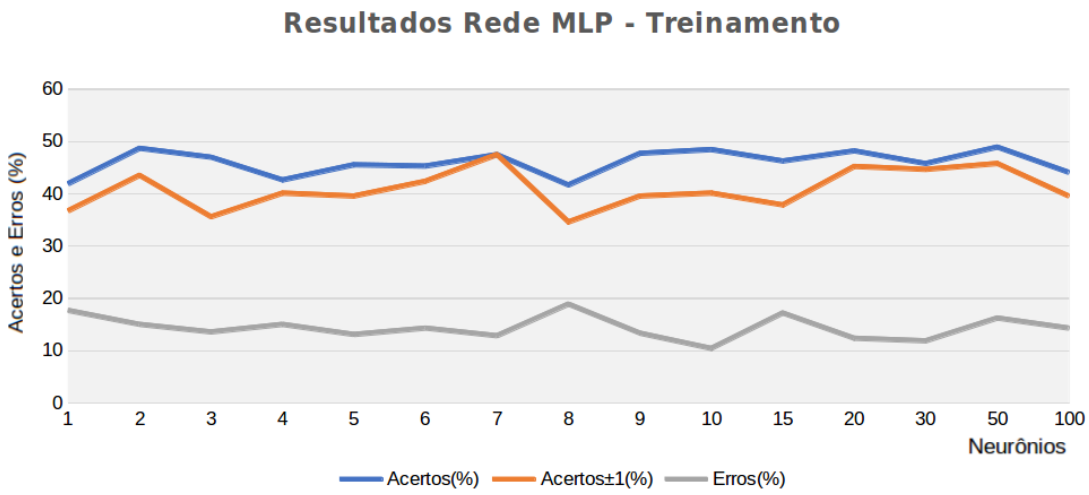

FIGURA 7: Acertos e erros da rede MLP para o treinamento

FONTE: O autor (2018)

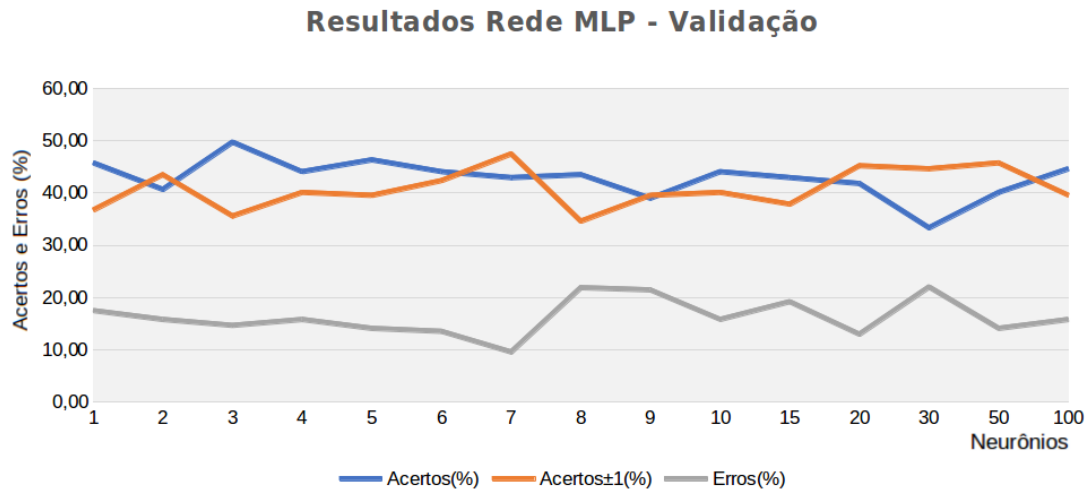

FIGURA 8: Acertos e erros da rede MLP para a validação

FONTE: O autor (2018)

Analisando estas informações, conclui-se que a rede neural MLP apresentou resultados satisfatórios, no entanto não viu-se diferenças significativas para as diferentes configurações. Escolheu-se então, a configuração com 7 neurônios na camada intermediária por apresentar a REQM razoavelmente menor que as outras configurações e o coeficiente de correlação minimanente mais próximo de 1, além de apresentar um dos menores índices de Erros na comparação entre os grupos.

3.4 APLICAÇÃO DO SVM

A aplicação do SVM deu-se de forma similar a rede MLP, através da biblioteca *scikit-learn* versão 0.20.1 (Scikit-Learn, 2018) do Python. Utilizou-se os dados de entrada com as mesmas informações que na rede MLP. No entanto, escolheu-se analisar a diferença de possíveis resultados através da mudança da função Kernel, sendo assim, as diferentes configurações se baseiam na troca entre as funções: linear, rbf (função de base radial), sigmoidal e quadrática.

Para o SVM também calculou-se os Acertos, Acertos ± 1 e Erros, além da REMQ e a correlação de Pearson, tanto para dados de treinamento quanto para dados de validação. Tais resultados são encontrados nas Tabelas 3 e 4.

Treinamento SVM					
Função	REMQ	Correlação	Acertos(%)	Acertos±1(%)	Erros(%)
Linear	1,07	0,74	45,52%	38,74%	15,73%
RBF	0,54	0,92	88,37%	7,74%	3,87%
Sigmoidal	1,41	0,45	21,30%	38,74%	39,95%
Quadrática	1,03	0,72	46,48%	38,49%	15,01%

TABELA 3: Resultados do SVM - Treinamento

FONTE: O autor (2018)

Validação SVM					
Função	REMQ	Correlação	Acertos(%)	Acertos±1(%)	Erros(%)
Linear	1,16	0,69	43,50%	36,15%	20,33%
RBF	1,13	0,69	44,63%	39,54%	15,81%
Sigmoidal	1,39	0,42	18,07%	44,63%	37,28%
Quadrática	0,95	0,77	49,71%	37,85%	12,42%

TABELA 4: Resultados da SVM - Validação

FONTE: O autor (2018)

Após a análise dos resultados da técnica SVM, escolheu-se como melhor configuração aquela que utilizou-se da função kernel RBF, visto que para os dados de treinamento seus resultados foram significativamente melhores que as outras configurações. A REMQ encontrada é de $0,54$ enquanto que as outras funções kernel a

REQM é maior do que 1, e o coeficiente de correlação de Pearson é de $0,92$, resultado que não pode ser ignorado.

Para os dados de validação não houveram diferenças significativas, apenas a configuração com a função quadrática apresentou uma pequena melhoria em relação às outras, porém não se destaca como a função rbf para os dados de treinamento.

Sendo assim, as configurações escolhidas para gerar o próximo resultado são:

- Para a rede MLP: Configuração com 7 neurônios na camada intermediária;
- Para o SVM: Configuração com a função kernel rbf.

3.5 COMPARAÇÃO MLP E SVM

Para ambas as técnicas, fez-se um pós processamento dos dados de saída motivados pelo fato que as técnicas apresentam algumas vezes, mais ou menos de quatro equipes em um mesmo grupo, fez-se então uma seleção através de *scores* calculados utilizando a posição da equipe no final do primeiro turno e sua pontuação. Observou-se que na maioria dos casos, este pós processamento trouxe respostas mais acertivas que a saída direta da MLP e do SVM.

Exemplo do calculo do $score$: em cada grupo deve-se ter apenas quatro times classificados, tanto a MLP quanto o SVM apresentam em alguns grupos mais de 4 times e em outros grupos menos de 4 times. Para encontrar o $score$ de todos os times, fez-se a normalização do dados dividindo a posição de todos pelo maior valor de posição e também dividindo a pontuação pela maior valor das pontuações. Criou-se então uma escala que analisa a quantidade de times classificados como grupo 1, se já forem 4 equipes, então segue a análise para o grupo de classificação 2, caso contrário, as equipes com os 4 maiores $scores$ permanecem no grupo 1 e as outras são colocadas no grupo 2, então segue a análise da quantidade de times no grupo 2 e assim por diante. De maneira análoga, fez-se a análise de quando a quantidade de times classificados no grupo for menos do que 4. Simplificando, o $score$ é calculado

como na equação 19.

$$score = \text{Pontuação - Posição} \tag{19}$$

Para comparar as duas técnicas estudadas e aplicadas neste trabalho, utilizou-se as configurações escolhidas para cada uma e os resultados do campeonato brasileiro do ano de 2018, Séria A. Os dados de entrada são: a posição, pontuação, saldo de gols e saldo de jogos (que envolvem vitórias, empates e derrotas). Os resultados são vistos na Tabela 5.

Classificação Campeonato Brasileiro 2018 – Série A

Equipe	Resultado	MLP	Pós-MLP	SVM	Pós-SVM
Palmeiras	1	1	1	1	1
Flamengo	1	1	1	1	1
Internacional	1	1	1	1	1
Grêmio	1	1	1	1	1
São Paulo	2	2	2	2	2
Atlético-MG	2	1	2	2	2
Atlético-PR	2	4	4	2	2
Cruzeiro	2	5	5	5	2
Botafogo	3	4	3	4	4
Santos	3	3	3	5	3
Bahia	3	3	2	2	3
Fluminense	3	4	4	4	3
Corinthians	4	5	5	4	4
Chapecoense	4	3	3	5	5
Ceará	4	1	2	4	4
Vasco	4	4	3	4	4
Sport	5	4	4	5	5
América	5	5	5	5	3
Vitória	5	4	4	5	5
Paraná	5	5	5	4	5
Acertos		**10**	**10**	**13**	**17**

TABELA 5: Comparação entre as técnicas MLP e SVM

FONTE: O autor (2018)

Na Tabela 5 tem-se as equipes que participaram da série A do campeonato brasileiro de 2018. Na coluna 'Resultado' tem-se os grupos exatos em que cada equipe finalizou o campeonato. Na coluna 'MLP' são apresentadas as saídas da rede MLP que apresentou um total de 10 classificações corretas (campos em azul), na coluna seguinte 'Pós-MLP' tem-se a classificação da rede MLP após a análise do $score$ para que se tenha exatamente 4 equipes por grupo, onde também encontrou-se um total

de 10 acertos.

Na coluna seguinte ’SVM’, apresentam-se os resultados do SVM, com 13 acertos, e por fim, na última coluna, tem-se os reultados após o processamento da saída do SVM para que se tenha exatamente 4 equipes por grupo, com isso, tem-se o melhor resultado, com 17 acertos.

4 CONCLUSÃO

Neste trabalho apresentou-se a aplicação de técnicas como a rede neural MLP e o *Support Vector Machine* para a classificação das equipes do campeonato brasileiro de futebol. Ambas as técnicas são reconhecidas na academia e exaustivamente utilizadas para fins como este de classificação e reconhecimento de padrões. Buscou-se pela melhor configuração da rede MLP que apesar de poucas diferenças, escolheu-se a rede com 3 camadas (entrada, intermediária e saída), onde os testes aplicados na camada intermediária sugere que sejam utilizados 7 neurônios em sua composição.

Para o SVM, fez-se testes similares para encontrar qual função kernel melhor se ajusta aos dados e retorna os melhores valores quando se fala na raiz do erro quadrático médio e coeficiente de correlação. Encontrou-se tal configuração com a função de base radial (RBF), que retornou pequenos erros e correlação alta entre os dados verídicos e dados previstos.

Para ambas as técnicas fez-se um pós processamento para obrigar que a classificação apresente exatamente 4 equipes por grupo, o que não acontecia com a saída direta.

Após a análise de todos os resultados, conclui-se que o SVM com pós processamento dos dados apresenta o melhor resultado, classificando de forma correta 17 das 20 equipes pertencentes ao campeonato brasileiro de 2018, que participaram da série A.

Obter uma informação como esta, na metade do campeonato pode estimular estratégias futuras de cada equipe, assim como obter uma margem do prêmio final que cada equipe arrecada ao ser anunciado o campeão. Por exemplo, uma equipe classificada no grupo 2, pode esperar que sua participação no campeonato gere um retorno

entre R$2.072.655,00 (8°) e R$4.092.165,00 (5°), já os últimos 4 colocados, ou seja, com classificação no grupo 5, além de serem rebaixados ao campeonato da série B, não arrecadam prêmios financeiros.

Por fim, este trabalho incentivou o estudo das técnicas de inteligência artificial, ciência e modelagem de dados, assim como o estudo da linguagem de programação Python. Conclui-se que tais técnicas são ferramentas poderosas quando utilizadas corretamente para fins de classificação e previsão.

REFERÊNCIAS

BEZERRA, C. A. C. Previsão de resultados de jogos do campeonato brasileiro de futebol: Uma abordagem de mineração de dados. **Universidade Federal de Lavras**, 2014.

CAPELO, R. **A CBF fatura alto com a seleção brasileira - eis de onde vem e para onde vai o dinheiro**. 2018. Https://epoca.globo.com/esporte/epoca-esporte-clube/noticia/2018/06/cbf-fatura-alto-com-selecao-brasileira-eis-de-onde-vem-e-para-onde-vai-o-dinheiro.html.

CHANG, K.-Y. H. W.-L. A neural network method for prediction of 2006 world cup football game. **The 2010 International Joint Conference on Neural Networks**, 2010.

DUARTE, L. **1X2 - Previsão de Resultados de Jogos de Futebol**. Tese (Doutorado) — Faculdade de Engenharia - Universidade do Porto, 2015.

FERNANDES, F. A. P. Umo modelo de redes neurais artificiais para predição do limite mínimo de gols em uma partida de futebol. **Centro Federal de Educação Tecnológica de Minas Gerais**, 2015.

GONçALVES, A. R. Máquina de vetores support. 2009. Disponível em: <www.dca.fee.inicamp.br/ andreric.>.

HAYKIN, S. **Redes Neurais: princípios e prática**. Porto Alegre: Bookman, 2001.

LIU, B. Web data mining. **Springer**, 2007.

LORENA A. C.; CARVALHO, A. C. P. L. F. Uma introdução às support vector machines. **RITA**, 2007.

OHATA, E. **Fifa deixa congelado prêmio de Mundial de Clubes**. 2017. Https://blogdoohata.blogosfera.uol.com.br/2017/09/08/fifa-deixa-congelado-premio-de-mundial-de-clubes-veja-quanto-campeao-leva/.

OLIVEIRA, C. de. **Uso de Support Vector Machine para detectar ecos de terrenos em dados do radar meteorológico do Simepar**. 2012. Monografia (Bacharel em Matemática Industrial), UFPR (Universidade Federal do Paraná), Brasil.

RAMÍREZ, M. C. V. **Previsão e Análise da Precipitação sobre as Regiões Sudeste e Sul do Brasil Utilizando Redes Neurais Artificiais**. Tese (Doutorado) — Instituto Nacional de Pesquisas Espaciais - INPE, 2003.

SANTOS, T. N. dos. **Redes Neurais Artificiais e Relação ZR aplicadas à Estimativa de Chuva**. Dissertação (Mestrado) — Universidade Federal do Paraná, 2014.

SCHMIDT, H. L. **Uso de Técnicas de Aprendizado de Máquina no Auxílio em Previsão de Resultados de Partidas de Futebol**. Tese (Doutorado) — Universidade de Santa Cruz do Sul, 2017.

Scikit-Learn. **Scikit-Leran - Machine Learning in Python**. [S.l.], 2018. Disponível em: <https://scikit-learn.org/stable>.

SILVA, L. N. C. **Análise e Síntese de Estratégias de Aprendizado para Rades Neurais Artificiais**. Dissertação (Mestrado) — Universidade Estadual de Campinas, 1998.

APÊNDICE A – ALGORITMO DA REDE NEURAL MLP

O algoritmo a seguir é baseado na sequência de passos apresentada na obra de Haykin (2001).

Notação:

- Os índices $i, j,$ e k referenciam neurônios diferentes; o neurônio j se encontra uma camada à direita do neurônio i, e o neurônio k à direita do neurônio j, quando o neurônio j é de uma camada oculta.
- Na iteração n, o n-ésimo padrão é apresentado à rede.
- $\varepsilon(n)$ se refere à soma instantânea dos erros quadráticos na iteração n.
- $e_j(n)$ se refere ao sinal do erro na saída do neurônio j.
- $d_j(n)$ se refere à resposta desejada para o neurônio j.
- $y_j(n)$ se refere ao sinal funcional que aparece na saída do neurônio j.
- $\omega_{ji}(n)$ representa o peso conectando a saída do neurônio i à entrada do neurônio j, na iteração n. A correção aplicada a este peso é representada por $\Delta\omega_{ji}(n)$.
- $v_j(n)$ representa a soma ponderada de todas as entradas acrescida do *bias* do neurônio j.
- $\varphi_j(\cdot)$ representa a função de ativação que descreve a relação de entrada-saída na não-linearidade associada ao neurônio j.
- O *bias* aplicado ao neurônio j é representado por b_j; seu efeito é representado por um peso $\omega_{j0} = b_j$ vinculado a uma entrada fixa $+1$.
- O i-ésimo termo do padrão de entrada é representado por $x_i(n)$.
- O k-ésimo termo do padrão de saída é representado por $o_k(n)$.
- A taxa de aprendizagem é representada por η.
- m_l representa o número de nós da camada l; $l = 0, 1, \ldots, L$, onde L é a profundidade da rede. Dessa forma, m_0 representa o tamanho da camada de entrada, m_L representa o tamanho da camada de saída e os outros representam as camadas intermediárias. A notação $m_L = M$ também é usada.

A.0.1 O Algoritmo da Rede MLP

O sinal de erro da saída do neurônio j, na iteração n, é

$$e_j(n) = d_j(n) - y_j(n), \tag{20}$$

Define-se o valor instantâneo do erro para o neurônio j como $\frac{1}{2}e_j^2(n)$. O valor instantâneo $\varepsilon(n)$ total do erro é encontrado somando-se os termos $\frac{1}{2}e_j^2(n)$ de todos os neurônios da camada de saída.

$$\varepsilon(n) = \frac{1}{2}\sum_{j\in C} e_j^2(n) \tag{21}$$

onde C inclui todos os neurônios da camada de saída. Seja N o número total de padrões do conjunto de treinamento. A média do erro quadrático pode ser obtida somando-se os $\varepsilon(n)$ e normalizando em relação a N, como a seguir

$$\varepsilon_{med} = \frac{1}{N}\sum_{n=1}^{N} \varepsilon(n) \tag{22}$$

O erro $\varepsilon(n)$ e consequentemente a média ε_{med}, é uma função dos pesos e *bias* da rede. Para um certo conjunto de treinamento, ε_{med} é a função de custo medindo o desempenho na aprendizagem. A aprendizagem deve ajustar os parâmetros livres (pesos e *bias*) minimizando ε_{med}. Para tal minimização, considera-se um método no qual os pesos são atualizados de padrão em padrão até completar uma apresentação inteira do conjunto de treinamento. Os novos pesos são calculados conforme os erros de cada padrão. A média aritmética destes erros individuais sobre o conjunto de treinamento é uma estimativa da alteração real dos pesos baseada na minimização da função ε_{med} sobre todo o conjunto de treinamento.

Considere um neurônio j sendo alimentado por um conjunto de neurônios de uma camada à sua esquerda. O campo local induzido $v_j(n)$ produzido na entrada da função de ativação é

$$v_j(n) = \sum_{i=0}^{m} \omega_{ji}(n) y_i(n) \tag{23}$$

onde m é o total de entradas (com exceção do *bias*) no neurônio j. O peso ω_{j0} que corresponde à entrada $y_0 = +1$ é igual ao b_j aplicado ao neurônio j. Então, $y_j(n)$ aparente na saída do neurônio j na iteração n é

$$y_j(n) = \varphi_j(v_j(n)) \tag{24}$$

O algoritmo de retropropagação aplica ao peso $\omega_{ji}(n)$, uma correção $\Delta\omega_{ji}(n)$ dada por:

$$\Delta\omega_{ji}(n) = -\eta\frac{\partial\varepsilon(n)}{\partial\omega_{ji}(n)} \tag{25}$$

sendo η a taxa de aprendizagem do algoritmo. O sinal negativo na Eq. (25) indica a

descida do gradiente em busca de uma mudança nos pesos que reduza $\varepsilon(n)$. A Eq. (25) pode ser escrita como

$$\Delta\omega_{ji}(n) = \eta\,\delta_j(n)y_i(n) \tag{26}$$

e $\delta_j(n)$ é o gradiente local, definido por

$$\delta_j(n) = e_j(n)\varphi'_j(v_j(n)) \tag{27}$$

O gradiente local aponta as mudanças que devem ser feitas nos pesos.

Neste ambiente pode-se identificar dois casos, dependendo do local que o neurônio j está na rede. No primeiro caso, trata-se o neurônio j como sendo de saída. No segundo, o neurônio j é um neurônio oculto.

O neurônio j é de saída

Estando o neurônio j contido na camada de saída, ele é munido com uma resposta desejada. Pode-se usar a Eq. (20) para calcular $e_j(n)$ referente a este neurônio. Determinando-se $e_j(n)$, calcula-se o gradiente local com a Eq. (27).

O neurônio j é oculto

Com o neurônio j sendo de uma camada oculta, não tem-se uma resposta desejada. Logo, o sinal de erro deve ser determinado recursivamente, utilizando os sinais de erro de todos os neurônios aos quais j está conectado. De acordo com a Eq. (27), o gradiente local pode ser redefinido como

$$\delta_j(n) = \varphi'_j(v_j(n))\sum_k \delta_k(n)\omega_{kj}(n). \tag{28}$$

onde k referencia um neurônio de saída.

O fator $\varphi'_j(v_j(n))$ no cálculo do $\delta_j(n)$ na Eq. (28) depende apenas da função de ativação relacionada ao neurônio oculto j. O somatório sobre k depende dos $\delta_k(n)$, requerendo conhecimento dos sinais de erro $e_k(n)$, para todos os neurônios da camada à direita do neurônio j, e dos termos ω_{kj}, que são os pesos associados às conexões.

Voltando as relações determinadas anteriormente para o algoritmo de retropropagação. Primeiramente, a correção $\Delta\omega_{ji}(n)$ sobre o peso conectando o neurônio i ao j é definida pela regra delta (HAYKIN, 2001):

$$\begin{pmatrix} \textit{Correção} \\ \textit{de peso} \\ \Delta\omega_{ji}(n) \end{pmatrix} = \begin{pmatrix} \textit{Parâmetros da} \\ \textit{taxa de aprendizagem} \\ \eta \end{pmatrix} \times \begin{pmatrix} \textit{Gradiente} \\ \textit{local} \\ \delta_j(n) \end{pmatrix} \times \begin{pmatrix} \textit{Sinal de entrada} \\ \textit{do neurônio } j \\ y_i(n) \end{pmatrix} \tag{29}$$

Segundo, a dependência sobre o gradiente local $\delta_j(n)$ se o neurônio j é da camada de saída ou de alguma camada oculta:

1. Se o neurônio j é de saída, $\delta_j(n)$ é o produto de $\varphi'_j(v_j(n))$ pelo sinal do erro $e_j(n)$.

2. Se o neurônio j é oculto, $\delta_j(n)$ é o produto de $\varphi'_j(v_j(n))$ pela soma ponderada dos gradientes locais locais calculados para os neurônios na camada seguinte.