UNIVERSIDADE FEDERAL DO TOCANTINS

CÂMPUS UNIVERSITÁRIO DE PALMAS

CURSO DE CIÊNCIA DA COMPUTAÇÃO

OTIMIZAÇÃO DOS PARÂMETROS DO ALGORITMO SVM COM O USO DO ALGORITMO PSO

ANDRÉ PEREIRA DOS SANTOS

PALMAS (TO)

2021

ANDRÉ PEREIRA DOS SANTOS

OTIMIZAÇÃO DOS PARÂMETROS DO ALGORITMO SVM COM O USO DO ALGORITMO PSO

Trabalho de Conclusão de Curso II apresentado à Universidade Federal do Tocantins para obtenção do título de Bacharel em Ciência da Computação, sob a orientação do(a) Prof.(a) Dr. Warley Gramacho da Silva.

Orientador: Dr. Warley Gramacho da Silva

PALMAS (TO)

2021

ANDRÉ PEREIRA DOS SANTOS

OTIMIZAÇÃO DOS PARÂMETROS DO ALGORITMO SVM COM O USO DO ALGORITMO PSO

Trabalho de Conclusão de Curso II apresentado à UFT – Universidade Federal do Tocantins – Câmpus Universitário de Palmas, Curso de Ciência da Computação foi avaliado para a obtenção do título de Bacharel e aprovada em sua forma final pelo Orientador e pela Banca Examinadora.

Data de aprovação: 9 / 12 / 2021

Banca Examinadora:

Prof. Dr. Warley Gramacho da Silva

Profa. Dra. Anna Paula de S. P. Rodrigues

Prof. Dr. Marcelo Lisboa Rocha

Dados Internacionais de Catalogação na Publicação (CIP)
Sistema de Bibliotecas da Universidade Federal do Tocantins

S237o Santos, André Pereira dos.

Otimização dos parâmetros do algoritmo SVM com o uso do algoritmo PSO. / André Pereira dos Santos. – Palmas, TO, 2021.

46 f.

Monografia Graduação - Universidade Federal do Tocantins – Câmpus Universitário de Palmas - Curso de Ciências da Computação, 2021.

Orientador: Warley Gramacho da Silva

1. Máquina de Vetores de Suporte. 2. Otimização por Enxame de Partículas. 3. Seleção de parâmetros. 4. Classificação. I. Título

CDD 004

Elaborado pelo sistema de geração automática de ficha catalográfica da UFT com os dados fornecidos pelo(a) autor(a).

A alguém cujo valor é digno

desta dedicatória.

AGRADECIMENTOS

Agradeço primeiramente a Deus, por ter me concedido a oportunidade de cursar uma graduação em uma Universidade Federal e por ter suprido todas as minhas necessidades durante todo o período da graduação.

Agradeço a minha família pelo apoio, em especial a minha avó e minha mãe que tem me ajudado e orado por mim durante toda a minha e vida, as minhas tias Jesus e Conceição e ao meu primo Cristy Anderson.

Também não poderia deixar de agradecer aos meus amigos, irmãos e irmãs de fé da minha igreja Assembleia de Deus congregação Monte Líbano, que os considero como uma família.

Agradeço ao Prof. Warley Gramacho, meu orientador nesta jornada e a todos os professores do curso Ciência da Computação.

RESUMO

A seleção dos parâmetros da Máquina de Vetores de Suporte (SVM) é de grande importância, pois influencia de forma significativa o desempenho do algoritmo SVM. O algoritmo de Otimização por Enxame de Partículas (PSO) é eficiente e bastante utilizado na solução de muitos problemas do mundo real. Neste trabalho é utilizado um método para busca de parâmetros ótimos com base na otimização por enxame de partículas, com o objetivo de melhorar a capacidade de aprendizado e generalização do SVM. O PSO-SVM é utilizado neste trabalho para o diagnóstico de câncer de mama. A eficácia do algoritmo PSO-SVM foi avaliada em relação ao Wisconsin Breast Cancer Dataset (WBCD), que é uma base de dados comumente utilizada entre pesquisadores que utilizam métodos de aprendizagem de máquina para diagnósticos de câncer de mama, e em relação ao Early stage diabetes risk prediction, que é uma base de dados para previsão de risco de diabetes em estágio inicial, e por último em relação a base de dados Sonar, Sonar, Minas vs. Rocha. A função de kernel utilizada é a Função de base radial (RBF). Os resultados do experimento demonstram que o algoritmo PSO-SVM obteve uma precisão bastante considerável quando comparado à outros estudos que utilizam somente o algoritmo SVM para o diagnóstico do cancêr de mama e para a previsão de risco de diabetes. A utilização do PSO mostrou-se eficiente para a busca dos parâmetros do SVM.

Palavra-chave: Máquina de Vetores de Suporte. Otimização por Enxame de Partículas. Seleção de parâmetros. Diagnóstico de câncer de mama. Previsão de risco de diabetes em estágio inicial.

ABSTRACT

The selection of the Support Vector Machine (SVM) parameters is of great importance, as it significantly influences the performance of the SVM algorithm. The Particle Swarm Optimization (PSO) algorithm is efficient and widely used in solving many real world problems. In this work, a method for finding optimal parameters based on particle swarm optimization is used, with the objective of improving the learning and generalization capacity of the SVM. The PSO-SVM is used in this work for the diagnosis of breast cancer. The effectiveness of the PSO-SVM algorithm was evaluated against the Wisconsin Breast Cancer Dataset (WBCD), which is a database commonly used among researchers using machine learning methods to diagnose breast cancer, and against the Early stage diabetes risk prediction, which is a database for predicting early-stage diabetes risk, and lastly in relation to the Sonar, Mines vs. Rock. The kernel function used is Radial Basis Function (RBF). The results of the experiment demonstrate that the PSO-SVM algorithm achieved a very considerable accuracy when compared to other studies that use only the SVM algorithm for the diagnosis of breast cancer and for the prediction of diabetes risk. The use of PSO proved to be efficient in the search for SVM parameters.

Keywords: Sonar, Minas vs. Rocha. Support Vector Machine. Particle Swarm Optimization. Selection of parameters. Breast Cancer Diagnosis. Early stage diabetes risk prediction. Sonar, Mines vs. Rock.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

O aprendizado de máquina (do inglês, Machine Learning) é a área da inteligência artificial que permite aos sistemas a capacidade de aprender e melhorar automaticamente a si mesmos, com base em dados, sem a necessidade de ser explicitamente programado. Esse processo começa com observações ou dados, através de experiências diretas ou instruções, para através de padrões nesses dados tomar as melhores decisões no futuro sem a necessidade de intervenções humanas. Existem variados tipos de algoritmos de aprendizagem de máquina, este trabalho abordará o algoritmo Máquina de Vetores de Suporte (SVM).

Desenvolvido por (VAPINK, 1995), Máquina de Vetores de Suporte, do inglês Vectors Machine - SVM, é um algoritmo de aprendizagem de máquina supervisionado comumente utilizado para resolver problemas de classificação e de regressão, é fundamentada no princípio indutivo da Minimização do Risco Estrutural, e este princípio é baseado na Teoria da Aprendizagem Estatística. Para obter a melhor capacidade de generalização, ele busca o melhor compromisso entre a complexidade do modelo e a capacidade de aprendizagem com base em informações amostrais limitadas (ZHU; ZHANG; LIU, 2006).

A seleção de parâmetros é um problema muito importante na área de pesquisa do algoritmo SVM e a capacidade de aprendizagem e o desempenho de generalização dependem da seleção dos parâmetros de SVM.

Uma meta-heurística é uma estrutura algorítmica independente do problema de alto nível que fornece um conjunto de diretrizes ou estratégias para desenvolver algoritmos de otimização heurística (SÖRENSEN; GLOVER, 2013). Nas últimas décadas os métodos meta-heurísticos têm sido muito estudado, através disso surgiram muitos algoritmos heurísticos de alta qualidade. Contudo, dificilmente utilizamos uma mesma meta-heurística a diversos tipos de problemas.

Uma meta-heurística é um modelo algorítmico genérico que pode ser usado para encontrar soluções de alta qualidade para problemas de otimização combinatória difíceis. Um problema é considerado difícil se a solução requer recursos significativos, independentemente do algoritmo usado, e nem sempre é possível encontrar uma solução possível em um tempo viável. O pior tempo de execução dos algoritmos exatos mais conhecidos para problemas difíceis cresce exponencialmente com o número de variáveis de decisão, o que pode facilmente levar a bilhões de anos para instâncias de problemas maiores.

Algoritmos de otimização fazem a ponte: eles trocam a qualidade da solução por tempo de execução, encontrando soluções muito boas (mas não necessariamente ótimas) no tempo viável.

Proposto pelo psicólogo James Kennedy e pelo engenheiro Russell Eberhart em

1995, o algoritmo de otimização por enxame de partículas (do inglês, particle swarm optimization, PSO) foi apresentada como uma meta-heurística estocástica para modelar o comportamento social de pássaros e cardumes. Originalmente foi utilizada para otimizar funções contínuas não lineares, e tem sido utilizada na resolução de muitos problemas reais. A inspiração para o algoritmo PSO teve início na década de 1990, onde diversos estudos a respeito do comportamento social dos grupos de animais, mostraram que animais pertencentes a seu grupo, como, por exemplo, aves e peixes, conseguem compartilhar informações entre os próprios membros de seu grupo (KENNEDY; EBERHART, 1995).

Este trabalho propõe investigar o uso do PSO para busca dos parâmetros ótimos do SVM e esta é uma abordagem eficiente para seleção de parâmetros do SVM.

1.1 Objetivos

1.1.1 Objetivo Geral

O objetivo geral do projeto consiste em utilizar o algoritmo PSO na seleção dos melhores parâmetros que melhore a acurácia do algoritmo SVM.

1.1.2 Objetivos Específicos

Mais especificamente o presente projeto tem como proposito:

- Investigar o processo de otimização dos parâmetros do algoritmo de aprendizado de maquina SVM através da meta-heurística PSO;
- Definir uma base de dados teste;
- Analisar e comparar os resultados obtidos.

1.2 Estrutura da Monografia

Inicialmente, são abordados os conceitos referentes a toda a fundamentação teórica, descrevendo os conceitos de um processo de otimização em que determinam os pontos extremos de uma função, é descrito os conceitos de meta-heurística, mais especificamente uma meta-heurística baseada em inteligência de enxame, a particle swarm optimization (PSO), descrevendo o seu processo de otimização, estrutura e sua representação matemática. Ainda na fundamentação teórica é abordado o conceito de máquina de vetores de suporte demonstrando sua formalização através de definição matemática. Em metodologia é descrito as ferramentas e base de dados, além de abordar o processo de otimização dos parâmetros SVM com base no algoritmo PSO. E por último são apresentados os resultados da otimização usando o conjunto de dados de diagnóstico de câncer de mama, o conjunto de dados de previsão de risco de diabetes em estágio inicial e a base de dados Sonar.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Processo de Otimização

Otimizar é algo muito comum em nosso cotidiano, trata-se da tarefa de encontrar a melhor solução possível sob um determinado conjunto de restrições. A otimização engloba as mais diversas áreas do conhecimento, com as mais diversas categorias de problemas, e independente do problema, o primeiro passo para se ter uma otimização é a modelagem matemática.

Problemas de otimização determinam os pontos extremos de uma função, seja mínimo, onde os problemas serão de minimização, ou máximo, que tratará da maximização. É importante a identificação de um objetivo que representa o desempenho do sistema a ser otimizado, para que um processo de otimização seja iniciado. Esses objetivos vão desde os mais simples, como ir por um caminho mais curto para chegar em casa, até problemas mais complexos, como escalar tripulações e aeronaves de modo que o custo das viagens sejam minimizados.

Os objetivos estão associados às variáveis que são as características do sistema, e essas variáveis devem ser tratadas sob algumas restrições. Todo esse processo é conhecido como modelagem.

A modelagem do sistema é representada pela função objetivo, que também é conhecida como função fitness, a escolha dessa função determinará se o problema proposto terá uma resolução rápida ou lenta, e se encontrará a solução considerada ótima.

A forma mais básica de se demonstrar matematicamente um problema de otimização é:

$$\min_{X \in R^n} sujeito\ a \begin{cases} c_i(x) = 0, i \in \varepsilon \\ c_i(x) \geq 0, i \in \varrho \end{cases} \tag{1}$$

onde

- x: é a função objetiva
- f(x): $R^n \rightarrow$ R é a função objetiva para ser minimizada sobre a variável x
- c: é o vetor que contém as restrições que as variáveis x devem satisfazer
- ε e ϱ: são os conjuntos de índices

Para maximizar um problema de otimização, basta utilizar o negativo da função objetivo, ou seja, $\min[f(x)] = \max[-f(x)]$.

Na Figura 1 percebe-se que na função modal possui diversas cristas e vales, a imagem demonstra que o ponto em destaque mais abaixo é um valor mínimo, ou seja, foi utilizado uma função de minimização, e esse ponto é a solução ótima global para o problema, os demais pontos são mínimos locais. De forma geral, a busca global procura garantir que o algoritmo irá se mover por uma região a partir do ponto em que uma busca local conseguirá encontrar um minimizador.

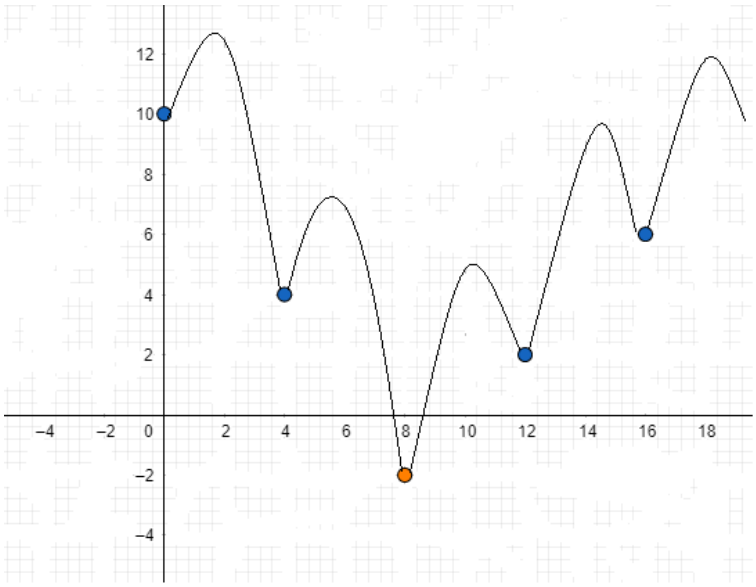

Figura 1 – Função multimodal

2.2 Meta-heurística

Utilizado pela primeira vez em 1986 por Fred Glover (1986), para demonstrar o método heurístico sem característica específica do problema. A meta-heurística é uma técnica de nível superior que orienta uma heurística subjacente e pode ser aplicada para resolver qualquer problema de otimização.

Uma meta-heurística é um processo de geração iterativo que orienta uma heurística subordinada combinando de forma inteligente conceitos para explorar os espaços de busca usando estratégias de aprendizagem para estruturar informações de modo a encontrar soluções quase ótimas de maneira eficiente (OSMAN; KELLY, 1997).

Algoritmos meta-heurísticos são bastante eficientes e eficazes na solução de problemas de ciência e engenharia. Esses algoritmos possuem dois esquemas de busca que são bastante significativos: diversificação e intensificação. A diversificação busca a melhor solução em regiões não exploradas, enquanto a intensificação tende a invadir novos espaços considerados como regiões promissoras para obter melhores soluções. Em média, todas as meta-heurísticas possuem o mesmo desempenho, diante disso ao longo dos anos foram desenvolvidos diversos algoritmos de meta-heurística que se adaptam a vários problemas.

Pode-se destacar algumas características que são fundamentais das meta-heurísticas, como:

- Busca uma solução "boa o suficiente"
- Normalmente não são exatas, e sim aproximadas
- Não são usadas em problemas específicos
- Possui implementação fácil
- Podem ser descritas por nível de abstração
- Podem ir da pesquisa básica até técnicas de aprendizagem avançadas
- Usa conceitos derivados de inteligência artificial, biológico, matemático, ciências naturais e física para orientar uma heurística com o intuito de melhorar seu próprio desempenho

2.3 Inteligência de Enxame

Tal expressão surgiu em 1989 por Beni e Wang, através de um trabalho onde um conjunto de bibliotecas era utilizado para otimização global no domínio de enxame de robôs. A inteligência de enxame surgiu com a tentativa de imitar o comportamento coletivo de seres em uma comunidade, como pássaros e insetos, através de simulações computacionais.

Sistemas fundamentados em inteligência de enxame normalmente possuem uma população de agentes simples, onde as possíveis soluções são atualizadas por meio da interação local umas com as outras e com o seu próprio ambiente, ou seja, dependente do princípio da descentralização.

Os princípios gerais do enxame são:

- Proximidade: a capacidade de memorizar o espaço e os cálculos do tempo.
- Qualidade: a capacidade de responder a fatores de qualidade, como por exemplo ser capaz de determinar a segurança de um local.
- Diversificação: os recursos não devem ficar juntos em uma única região estreita.
- Estabilidade: as mudanças de ambientes não devem alterar o comportamento da população.
- Adaptabilidade: quando houver mudanças no ambiente, se necessário, a população deverá ter a capacidade de adaptação.

Existem diversos problemas de otimização que podem ser resolvidos através de algoritmos de inteligência de enxames, como, por exemplo, problema do caixeiro-viajante, roteamento de veículos, treinamentos de redes neurais, dentre outros. E um dos algoritmos mais famosos de inteligência de exame é o PSO (Otimização por Enxame de Partículas).

2.4 Enxame de Partícula (PSO)

Proposto pelo psicólogo James Kennedy e pelo engenheiro Russell Eberhart em 1995, o algoritmo de otimização por enxame de partículas (do inglês, particle swarm optimization, PSO) foi apresentada como uma meta-heurística estocástica para modelar o comportamento social de pássaros e cardumes. Originalmente foi utilizada para otimizar funções contínuas não lineares, utilizada na resolução de muitos problemas reais. A inspiração para o algoritmo PSO teve início na década de 1990, onde diversos estudos a respeito do comportamento social dos grupos de animais, mostraram que animais pertencentes a seu grupo, como, por exemplo, aves e peixes, conseguem compartilhar informações entre os próprios membros de seu grupo (KENNEDY; EBERHART, 1995).

Craig Reynolds (1987) através de um computador simulou o comportamento de bando e propôs 3 regras simples para a implementação de tal comportamento, são elas: coesão, separação e alinhamento. A coesão centraliza o bando, cada ave do bando permanece perto da ave que está mais próxima e o movimento individual dessa ave faz com que ela fique mais perto do centroide dos companheiros. A separação evita colisão com membros do bando, e no alinhamento cada ave alinha o seu caminho na direção média dos membros do bando.

Em um processo de otimização, o movimento das partículas é ilustrado na Figura 2, onde inicialmente demonstra que as partículas estão posicionadas no espaço de solução e passa por um processo de estreitamento, fazendo com que elas sejam direcionadas através da busca em uma região de valores que são mais interessantes com movimentos no espaço de solução em busca dos melhores pontos que satisfaçam o objetivo da otimização.

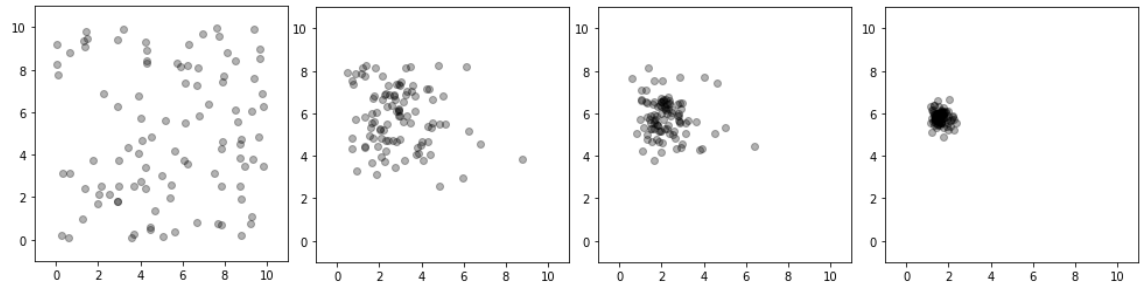

Figura 2 – Movimento das Partículas

2.4.1 Estrutura do enxame de partículas

O algoritmo PSO usa uma população de indivíduos onde os membros se movimentam no espaço de busca com o objetivo de encontrar a melhor solução para um determinado problema. Cada partícula no algoritmo é formada por três vetores: velocidade, posição no espaço de busca em D dimensões e a melhor posição encontrada até o momento. Respectivamente representadas como $\vec{V_i} = (v_{i1}, v_{i2}, \ldots, v_{iD})$, $\vec{X_i} = (x_{i1}, x_{i2}, \ldots, x_{iD})$ e $\vec{P_i} = (p_{i1}, p_{i2}, \ldots, p_{iD})$.

O algoritmo é inicializado aleatoriamente por uma população de soluções, as suas velocidades e posições podem ser atualizadas conforme o ambiente mudar, não limita seus movimentos e continuamente procura a solução ideal no possível espaço de solução conforme atualiza as gerações. O movimento estável das partículas podem ser mantidas no espaço de busca, e se adaptam às mudanças no ambiente conforme mudam seu modo de movimento.

O movimento da partícula que é ilustrado na Figura 3 demonstra serem necessários três elementos para ocorrer o próximo movimento da partícula, são eles: cognição, onde força a partícula seguir para um caminho que seja melhor que a atual, a inércia, que força a partícula a manter sua mesma direção, e o termo de aprendizado social, que força a partícula ir em direção de seus melhores vizinhos, ou seja, a melhor posição encontrada.

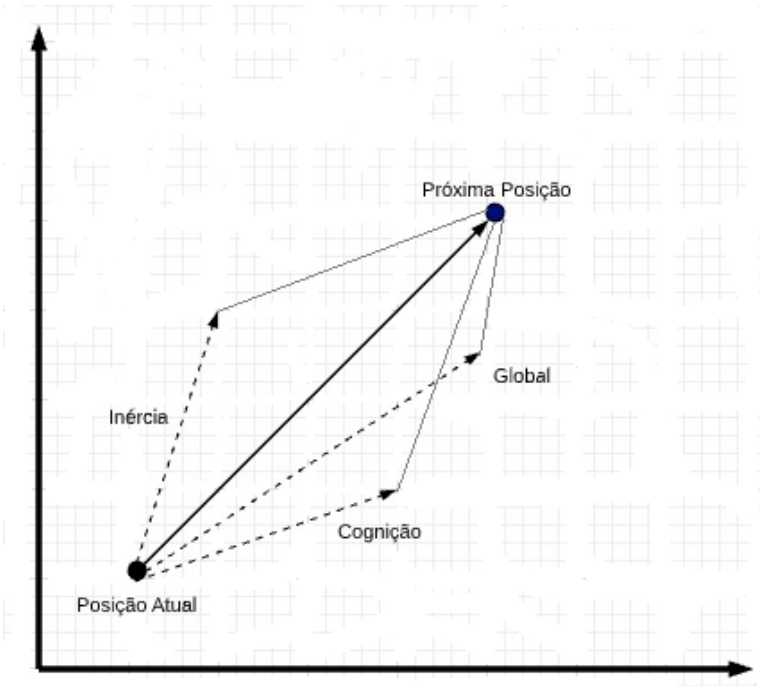

Figura 3 – Atualização da Posição da Partícula Fonte: Adaptado de (Yoshida, Kawata, Fukuyama, Takayama & Nakanishi, 2000).

Além do sistema ser inicializado por uma população de forma aleatória, a cada solução potencial, também conhecida como partículas, atribui-se uma velocidade aleatória e então são lançadas através do espaço do problema. Quando cada partícula se adéqua a

uma solução que até então é a melhor, elas conseguem fazer o controle de suas coordenadas no espaço do problema. A versão global do algoritmo consegue rastrear e armazenar a melhor posição alcançada, indicada com pbest, e também o até então melhor valor geral e sua localização, conhecida como gbest. A Figura 4 demonstra a topologia da versão global do algoritmo, onde todas as partículas, representadas pelas bolinhas da figura, possuem informações sobre as demais.

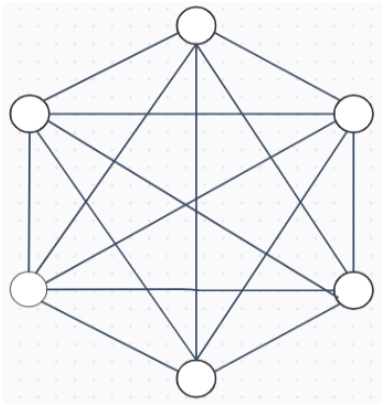

Figura 4 – Topologia global

O algoritmo PSO busca a solução ótima, atualizando a velocidade e localizações Pbest e gbest de cada partícula. A aceleração possui números aleatórios separados, gerados para a velocidade em direção às localizações pbest e gbest.

Com o objetivo de reduzir a dependência do processo de busca nos limites rígidos da velocidade, o conceito de peso de inércia w foi introduzido no algoritmo PSO (SHI; EBERHART, 1998). A velocidade e a posição de cada partícula são atualizadas conforme as equações 2 e 3, respectivamente da seguinte forma:

$$v_{k+1} = wv_k + c_1 r_1(Pbest_k - x_k) + c_2 r_2(gbest_k - x_k) \tag{2}$$

$$x_{k+1} = x_k + v_k \tag{3}$$

onde c_1 é um coeficiente que contribui com a aprendizagem da própria partícula e c_2 é o índice de aprendizado global da partícula. Para garantir a melhor busca pelas soluções, foi adotado neste trabalho o conceito de Coeficientes de Aceleração Variados no Tempo (TVAC), para melhorar o equilíbrio do espaço de busca entre a exploração global e a exploração local, introduzidos em (RATNAWEERA; HALGAMUGE; WATSON, 2004). A principal ideia desse conceito é fazer com que c_1 diminua de seu valor inicial de c_{1i} para c_{1f}, enquanto c_2 aumenta de c_{2i} para c_{2f}, como em (RATNAWEERA; HALGAMUGE; WATSON, 2004). O conceito TVAC é representado matematicamente da seguinte forma:

$$c_1 = (c_{1f} - c_{1i})\frac{t}{t_{max}} + c_{1i}, \tag{4}$$

$$c_2 = (c_{2f} - c_{2i})\frac{t}{t_{max}} + c_{2i}. \tag{5}$$

onde c_{1f}, c_{1i}, c_{2f} e c_{2i} são constantes, t é a iteração atual do algoritmo e t_{max} é o número máximo de iterações.

Nas Equações 2 e 3 v_k é a velocidade atual da partícula k e essa velocidade é restrita ao intervalo $[-v_{max}, v_{max}]$, de modo a evitar que as partículas voem para fora do espaço de solução. w é o fator de inércia da partícula, $Pbest_k$ é a melhor posição alcançada da partícula k, $gbest_k$ é o melhor valor geral e sua localização, x_k é a posição da partícula k, v_{k+1} é a velocidade atualizada da partícula, x_{k+1} é a posição atualizada da partícula e r_1 e r_2 são números aleatórios entre 0 e 1.

O fator de inércia (w) melhora a velocidade de convergência do algoritmo, e a sua presença também dá um equilíbrio na velocidade das partículas. Shi e Eberhart (1998) mostra que se o w é grande ($>$ 1,2) terá um comportamento de busca global e tentará sempre explorar as novas áreas, porém se w é médio ($0,8 < w < 1,2$) o algoritmo terá mais oportunidade de chegar ao ótimo global, porém levará um número moderado de iterações. Com o objetivo de reduzir o peso sobre as iterações possibilitando ao algoritmo a exploração de algumas áreas específicas, o peso de inércia w é atualizado conforme mostra a seguinte equação:

$$w = w_{min} + (w_{max} - w_{min})\frac{(t_{max} - t)}{t_{max}} \tag{6}$$

A Equação 6 é conhecida como o peso de inércia variável no tempo (TVIW) (EBERHART; SHI, 2000), onde w_{max} e w_{min} são valores máximos e mínimo predefinidos do peso de inércia w, t_{max} é o número máximo de iterações e t é a iteração atual do algoritmo. Essa Equação melhora de forma significativa o desempenho do PSO, acrescentando ao algoritmo uma maior capacidade de pesquisa global logo no início da execução e acrescenta maior capacidade de pesquisa local próximo ao fim da execução.

2.5 Classificação

Classificação é o processo de classificar um determinado conjunto de dados em classes, que podem ser realizadas em dados estruturados ou não estruturados. O processo começa prevendo a categoria de certos pontos de dados. As classes são frequentemente

chamadas alvos, rótulos ou categorias.

A modelagem de classificação preditiva é a tarefa de aproximar a função de mapeamento de variáveis de entrada para variáveis de saída discretas. O objetivo principal é determinar em qual classe ou categoria os novos dados se encaixam.

No aprendizado de máquina, a classificação é um conceito de aprendizado supervisionado, que basicamente classifica um conjunto de dados. Os problemas de classificação mais comuns são: reconhecimento de voz, detecção de rosto, classificação de documentos, dentre outros. Pode ser um problema de classificação binária ou pode ser um problema de várias classes. Existem muitos algoritmos de aprendizado de máquina usados para classificação no aprendizado de máquina, dentre esses esta a máquina de vetores de suporte (SVM).

2.6 Máquina de Vetores de Suporte

Desenvolvido por (VAPINK, 1995), Máquina de Vetores de Suporte, do inglês Vectors Machine - SVM, é um algoritmo de aprendizagem de máquina supervisionado comumente utilizado para resolver problemas de classificação e de regressão, é fundamentada no princípio indutivo da Minimização do Risco Estrutural, e este princípio é baseado na Teoria da Aprendizagem Estatística.

O princípio indutivo de Minimização do Risco Estrutural idealizado por Vapnik, busca minimizar o erro do conjunto de treinamento e do conjunto de teste, possui vantagens de uma teoria forte e boa capacidade de generalização, buscando o melhor compromisso entre a complexidade do modelo e a capacidade de aprendizagem com base em informações amostrais limitadas.

Segundo (HAYKIN, 1999) a máquina de vetores de suporte é outra categoria das redes neurais alimentadas adiante, ou seja, redes cujas saídas dos neurônios de uma camada alimentam os neurônios da camada posterior, não ocorrendo a realimentação.

O SVM funciona primeiro mapeando dados para um espaço de feição de alta dimensão para que os pontos de dados possam ser categorizados, mesmo quando os dados não são separáveis linearmente. Em seguida, um separador é estimado para os dados. Os dados devem ser transformados de tal forma que um separador possa ser desenhado como um hiperplano.

Basicamente, SVMs são baseados na ideia de encontrar um hiperplano que melhor divide um conjunto de dados em duas classes, como mostra a Figura 5.

Como o espaço é bidimensional, pode-se pensar no hiperplano como uma linha que separa linearmente os pontos azuis dos pontos vermelhos.

Uma escolha razoável como o melhor hiperplano é aquele que representa a maior separação ou margem entre às duas classes, para minimizar erros de classificação e problemas de enviesamento do modelo (overfitting). Então o objetivo é escolher um hiperplano

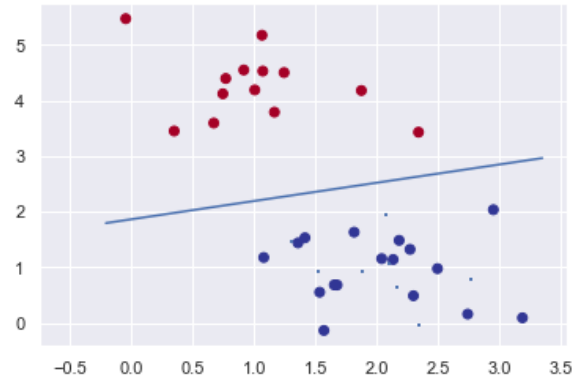

Figura 5 – Conjunto de dados divididos em duas classes

com uma margem tão grande quanto possível. Exemplos mais próximos do hiperplano são vetores de suporte, em verde na Figura 6. É intuitivo que apenas os vetores de apoio importam para alcançar o objetivo, e assim, outros exemplos de tendências podem ser ignorados. Há a tentativa de encontrar o hiperplano de tal forma que ele tenha a distância máxima para suportar vetores.

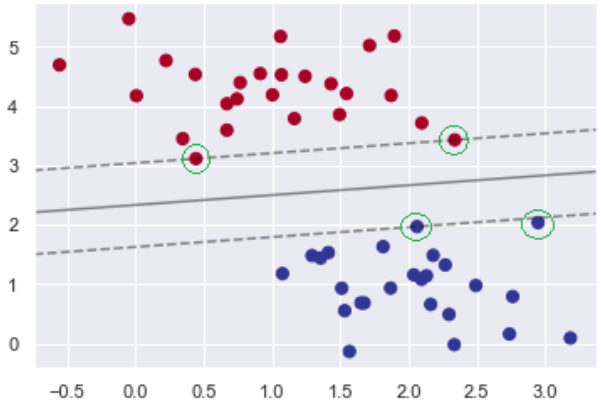

Figura 6 – Conjunto de dados divididos por um hiperplano

A equação do hiperplano que divide as classes está localizado no ponto médio entre eles, e pode ser facilmente escrita como:

$$\vec{w}.\ \vec{x} + b = 0 \tag{7}$$

onde $\vec{w}$ é perpendicular ao hiperplano e é também o vetor que caracteriza esse hiperplano, $\vec{x}$ é um ponto qualquer pertencente ao hiperplano, e b é uma constante que determina o deslocamento do hiperplano em relação à origem, além de determinar a regra de decisão.

O hiperplano é aprendido a partir de dados de treinamento usando um procedimento de otimização que maximiza a margem. Portanto, a saída do algoritmo é os valores w e b para a linha. Pode-se fazer classificações usando essa linha estimada, suficiente para conectar valores de entrada na equação de linha, em seguida, pode-se calcular se um ponto

desconhecido está acima ou abaixo da linha, se a equação retorna um valor maior que 0, então o ponto pertence à primeira classe que está acima da linha, e vice-versa.

Encontrar o hiperplano ótimo pode ser formalizado usando definições matematicamente convenientes, como descrito em (8), onde é definido uma variável de classificação y_i que assume os valores +1 e -1 para amostras positivas e negativas respectivamente.

$$\begin{aligned} &\vec{w}.\ \vec{x}_i + b \geq +1, se\ y_i \geq 1 \\ &\vec{w}.\ \vec{x}_i + b \leq -1, se\ y_i \leq -1 \end{aligned} \tag{8}$$

Ao introduzir a variável y transformamos numa única equação, com isso, podemos definir a margem e consequentemente os vetores de suporte:

$$y(\vec{w}.\ \vec{x}_i + b) - 1 \geq 0, \forall(\vec{x}_i, y_i) \in T \tag{9}$$

Para maximizar a margem e por consequência definir o melhor hiperplano possível, devemos encontrar os valores de w e b, logo há a necessidade de encontrar a distância entre a margem inferior e superior.

$$width = \frac{(x_+ - x_-)\ .\ \vec{w}}{||\vec{w}||} \tag{10}$$

Levando em consideração que y é positivo para amostras à direita e negativo para amostras à esquerda, usamos a equação (8) para simplificar a equação (10), ficando da seguinte forma:

$$width = \frac{b+1}{||\vec{w}||} = \frac{b-1}{||\vec{w}||} = \frac{1}{||\vec{w}||}(b + 1 - b + 1) = \frac{2}{||\vec{w}||} \tag{11}$$

Com isso, podemos perceber que o objetivo é maximizar $\frac{2}{||\vec{w}||}$, obedecendo às restrições estabelecidas da equação (8) e classificando corretamente os dados.

Uma vez que queremos maximizar a margem, analogamente pode-se dizer que também é possível minimizar $||\vec{w}||$ ou podemos transformá-lo em um problema quadrático conforme demostrado na equação (12), o tornando matematicamente mais conveniente:

$$\begin{aligned} &minimize : \tfrac{1}{2}||\vec{w}||^2 \\ &sujeito\ a\ \ y_i(\vec{w}.\ \vec{x}_i + b) - 1 \geq 0,\ para\ i = 1, \ldots,\ n \end{aligned} \tag{12}$$

A forma como solucionar este problema de otimização de programação quadrática

sob o aspecto de aprendizado de máquina é usando o método dos multiplicadores de Lagrange, definida em termos de w e b:

$$L_p = \frac{1}{2}||\vec{w}||^2 - \sum_{i=1}^{N} \alpha_i [y_i(\vec{w}.\ \vec{x}_i + b) - 1] \tag{13}$$

onde, os α_i são denominados de multiplicadores de Lagrange não-negativos.

Como o problema agora passou a ser a minimização da equação (13) que é em relação a w e b e maximização de α_i, com $\alpha_i > 0$, a derivada de L_p deve ser nula. A solução para este problema é chamada de ponto de sela ótimo. Através das condições fornecidas pelo teorema de Karush-Kuhn-Tucker (KKT) (CRISTIANINI; SHAWE-TAYLOR et al., 2000) encontramos seu valor.

Como primeira condição, as derivadas de L_P com relação a $\vec{w}$ e b, devem ser zeradas. Logo

$$L_p = \frac{\partial L_p(\vec{w}, b, \vec{\alpha})}{\partial \vec{w}} = \vec{w} - \sum_{i=1}^{N} \alpha_i y_i \vec{x}_i = 0 \tag{14}$$

$$L_p = \frac{\partial L_p(\vec{w}, b, \vec{\alpha})}{\partial b} = \sum_{i=1}^{N} \alpha_i y_i = 0 \tag{15}$$

com isso obtemos:

$$\vec{w} = \sum_{i=1}^{N} \alpha_i y_i \vec{x}_i \tag{16}$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0 \tag{17}$$

As demais condições Karush-Kuhn-Tucker (KKT) são:

$$\begin{cases} \alpha_i^* \{(y_i(\vec{w}.\ \vec{x}_i + b) - 1)\} = 0 & para\ i = 1, ..., N \\ \alpha_i \geq 0, & para\ i = 1, ..., N \\ y_i(\vec{w}.\ \vec{x}_i + b) - 1 \geq 0, & para\ i = 1, ..., N \end{cases} \tag{18}$$

Substituindo as equações acima em (13), pode-se obter a formulação dual de oti-

mização, que em termos computacionais, tornará mais eficiente encontrar o ponto de sela:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i=j}^{N} \alpha_i \alpha_j y_i y_j (\vec{x}_i^T \vec{x}_j) \tag{19}$$

Podemos encontrar a solução através da minimização de L_p (problema primal) ou através da maximização de L_D (problema dual). Para cada ponto de treinamento há um multiplicador de Lagrange $\alpha_i^* > 0$ e são chamados de vetores de suporte e estão em um dos hiperplanos. O hiperplano ótimo precisa exclusivamente dos vetores de suporte, considerados os elementos críticos do conjunto de treinamento. Eles estão mais próximos do hiperplano de decisão, ou seja, estão nas fronteiras.

Assim, b^* é uma função apenas dos vetores de suporte. Pode-se mostrar que o bias do hiperplano ótimo de separação pode ser definido como:

$$b^* = \frac{1}{n_{SV}} \sum_{x_j \in SV} \left(\frac{1}{y_j} - \vec{w}^* \vec{x}_j) \right) \tag{20}$$

n_{SV} denota o número de vetores de suporte e SV é o conjunto dos vetores de suporte.

Ou ainda, substituindo $\vec{w}^*$ por sua definição em (16):

$$b^* = \frac{1}{n_{SV}} \sum_{x_j \in SV} \left(\frac{1}{y_j} - \sum_{\vec{x}_i \in SV} \alpha_i^* y_i (\vec{x}_i^T \vec{x}_j) \right) \tag{21}$$

temos a solução dada em termos dos produtos internos entre cada par de vetores de suporte.

Portanto, como resultado tem-se a seguinte regra de decisão/classificação:

$$g(\vec{x}) = sgn \left(\sum_{\vec{x}_i \in SV} y_i \alpha_i^* (\vec{x}_i \vec{x}) + b^* \right) \tag{22}$$

onde sgn denota a função sinal:
Se g($\vec{x}$) < 0 (sinal negativo): a amostra de $\vec{x}_i$ é classificada como classe 1.
Se g($\vec{x}$) > 0 (sinal positivo): a amostra de $\vec{x}_i$ é classificada como classe 2.

Os parâmetros a serem aprendidos na etapa de treinamento são $\vec{w}^*$ e b^*. Enquanto $\vec{w}^*$ controla a inclinação do hiperplano separador ótimo, b^* define sua posição em relação as margens.

2.6.1 Máquina de Vetores de Suporte não lineares

Na maioria dos problemas reais é difícil encontrarmos padrões que sejam linearmente separáveis, haverá situações em que será impossível separar um conjunto dados de treinamento por um hiperplano, e isso gerará erros de classificação. Esses conjuntos de dados são conhecidos como não-linearmente separáveis.

Considere a Figura 7, a imagem a esquerda mostra a distribuição de um pequeno conjunto de amostras apenas com base em seu tamanho de unidade e espessura do aglomerado. Podemos perceber que os pontos de dados se enquadram em duas categorias diferentes, e essas amostras representam um conjunto de dados não separáveis linearmente, porém as duas categorias podem ser separadas com curvas, mas não uma linha, como mostra a imagem a direita, ou seja, ele representa um conjunto de dados linearmente não separáveis, o caso para a maioria dos conjuntos de dados reais.

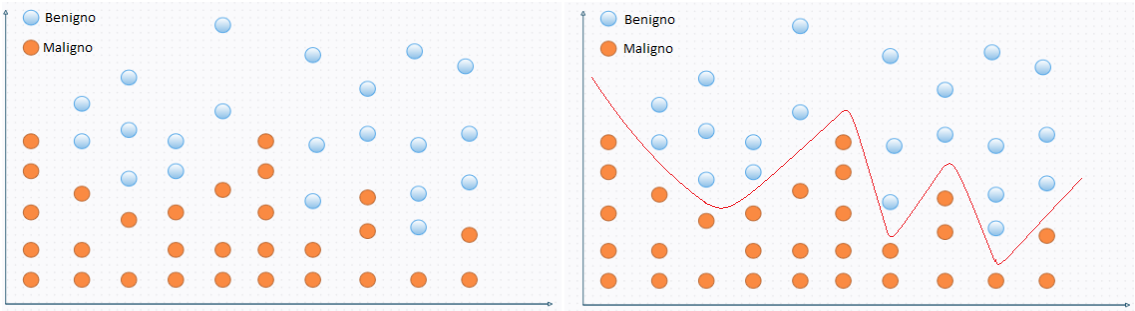

Figura 7 – Conjunto de dados linearmente não separável

Podemos transferir esses dados para um espaço de maior dimensão, por exemplo, mapeando-os para um espaço tridimensional. Após a transformação, o limite entre as duas categorias pode ser definido por um hiperplano. Como o espaço em questão é tridimensional, o separador é mostrado como um plano. Este plano pode ser usado para classificar casos novos ou desconhecidos. Portanto, o algoritmo SVM gera um hiperplano ideal que categoriza novos exemplos.

Para problemas não lineares o algoritmo usa funções de kernel para mapear o espaço de entrada, que é de alta dimensão.

Podemos modificar a equação (22) e representar matematicamente a função sinal para resolver problemas não lineares da seguinte forma:

$$g(x) = sgn\left(\sum_{\vec{x}_i \in SV} y_i \alpha_i^* \phi(\vec{x}_i)^T \phi(\vec{x}) + b^*\right) \tag{23}$$

onde $(\vec{x}_i \vec{x})$ no espaço de entrada é representado como a forma $\phi(\vec{x}_i)^T \phi(\vec{x})$ no espaço de recurso. b^* é dado pela mesma expressão da equação (21) com aplicação da função ϕ e expressa da seguinte forma:

$$b^* = \frac{1}{n_{SV}} \sum_{x_j \in SV} \left(\frac{1}{y_j} - \sum_{\vec{x}_i \in SV} \alpha_i^* y_i \phi(\vec{x}_i)^T \phi(\vec{x}) \right) \tag{24}$$

Durante a fase de treinamento, calcular $\phi(\vec{x})$ pode ser extremamente custoso, porém a função dual acima poupa tal esforço utilizando kernels.

2.6.2 Função Kernel

Uma função é dita ser uma função kernel se satisfazer as condições do Teorema de Mercer (CRISTIANINI; SHAWE-TAYLOR et al., 2000), onde essa função recebe dois pontos no espaço de entrada x_i e x_j e calcula o produto escalar no espaço de características, conforme mostra a função abaixo:

$$K(\vec{x}_i, \vec{x}_j) = \left\langle \phi(\vec{x}_i)^T \ . \ \phi(\vec{x}_j) \right\rangle \tag{25}$$

Existem diferentes tipos de função kernel, a Tabela 1 mostra alguns dos kernels mais utilizados. Cada uma dessas funções tem suas próprias características, seus prós e contras e sua equação. Para o experimento neste trabalho, será utilizado o kernel RBF (Função de Base Radial).

Tabela 1 – Principais tipos de kernels

Tipo kernel	Função K($\vec{x}_i$, $\vec{x}_j$)
Linear	$(\vec{x}_i \ . \ \vec{x}_j)$
Polinomial	$(\langle \vec{x}_i \ . \ \vec{x}_j \rangle + 1)^P$
Gaussiano ou (RBF)	$\exp\left(\frac{-\|\vec{x}_i - \vec{x}_j\|^2}{2\sigma^2}\right)$
Sigmóide	$\tanh(\beta_0 \langle \vec{x}_i \ . \ \vec{x}_j \rangle) + \beta_1$

Existem casos que mesmo após transformar o espaço com uma função kernel, continuam a surgir problemas na separação das classes. Para a solução desses casos os hiperparâmetros adicionam versatilidade ao modelo, permitindo que algumas amostras sejam classificadas de forma incorreta ou na tentativa de alcançar uma separação perfeita. Esses hiperparâmetros serão apresentados a seguir.

2.6.3 Hiperparâmetros do modelo

Os hiperparâmetros possuem uma grande importância durante o treinamento de um modelo, eles possuem a capacidade de alterar a velocidade de treinamento, a tendência ao overfit, toleram os erros de classificação, dentre outras coisas. Esses hiperparâmetros

são conhecidos como C e *Gama*, e encontrar os valores ideais para ambos pode ser uma tarefa difícil.

A presença do parâmetro C encontra-se tanto no SVM linear quanto no não-linear, e possui a responsabilidade de controlar a tolerância a erros de classificação em um modelo de treinamento, o C adiciona uma penalidade para cada ponto de dados mal classificado.

Se o C possuir um valor grande, o algoritmo SVM tentará minimizar a quantidade de exemplos classificados erroneamente, devido à alta penalidade que resultará em um limite de decisão com uma margem menor, porém essa penalidade não se aplica a todos os exemplos classificados de forma incorreta, e a demanda de tempo de treinamento é grande, por gerar fronteiras de decisão muito complexas. Se o valor de C for pequeno, flexibiliza a etapa de treinamento permitindo fronteiras de decisão com erros (ou soft margin).

A imagens da Figura 8 exemplifica o comportamento do algoritmo SVM utilizando diferentes valores para C:

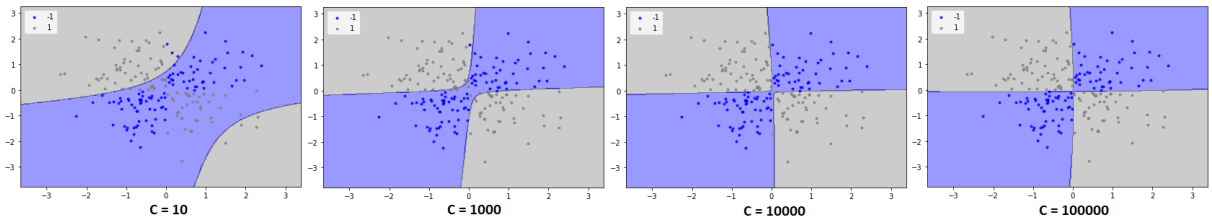

Figura 8 – Comportamento do algoritmo SVM com o uso da função de kernel RBF utilizando diferentes valores para C

O hiperparâmetro *Gama* é utilizado somente no SVM não-linear, por meio das funções de kernel. Para *Gama* com valores elevados, é necessário que os pontos estejam muito próximos uns dos outros, caso contrário fará com que os pontos mais distantes da região de separação entre as classes sejam desconsiderados, portanto, se o valor do *Gama* for muito grande há tendência do modelo se ajustar acima do ideal. Porém, se *Gama* possuir um valor baixo, indicará um grande raio de similaridade, permitindo alguns erros de classificação.

A imagens da Figura 9 exemplifica o comportamento do algoritmo SVM utilizando diferentes valores para *Gama*:

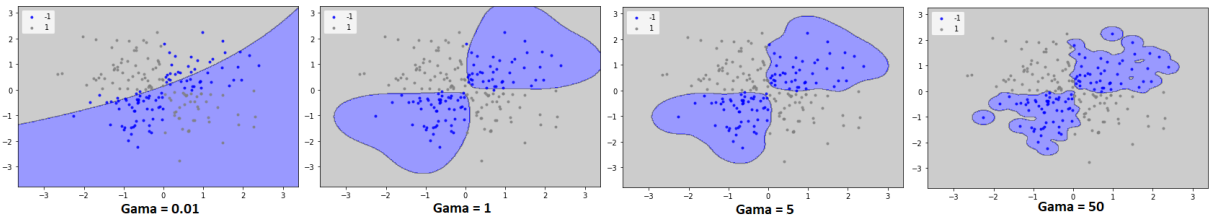

Figura 9 – Comportamento do algoritmo SVM com o uso da função de kernel RBF utilizando diferentes valores para Gama

3 METODOLOGIA

3.1 Ferramentas

3.1.1 Python

Para o experimento, o algoritmo PSO-SVM foi implementado usando a linguagem de programação python. Para o SVM, foi utilizada da biblioteca de aprendizado de máquina scikit-learn a classe SVC do módulo SVM. O algoritmo PSO foi implementado do zero.

3.1.2 Colab

Os testes e a implementação do algoritmo foram feitas utilizando a plataforma do *Google Colaboratory* incluindo nas configurações do tipo de ambiente de execução o acelerador de hardware GPU.

3.2 Bases de dados

3.2.1 Breast Cancer Wisconsin (Diagnostic)

Neste estudo, foi utilizado a base de dados Breast Cancer Wisconsin (Diagnostic) retirado do Repositório de Aprendizagem de Máquinas UC Irvine. O conjunto de dados contém 569 instâncias, dos quais 357 casos pertencem à classe benigna e 212 casos pertencem à classe maligna.

As características dessa base de dados são computadas a partir de uma imagem digitalizada de uma agulha fina aspirada (FNA) de uma massa mamária. Descrevem características dos núcleos celulares.

Dez recursos de valor real são calculados para cada núcleo da célula:

1. raio (média das distâncias do centro aos pontos do perímetro)
2. textura (desvio padrão dos valores da escala de cinza)
3. perímetro
4. área
5. suavidade (variação local nos comprimentos dos raios)
6. compactação ($permetro^2$ / área - 1,0)
7. concavidade (severidade das porções côncavas do contorno)

8. pontos côncavos (número de porções côncavas do contorno)
9. simetria
10. dimensão fractal

3.2.2 Early stage diabetes risk prediction

Neste estudo, também foi utilizado o conjunto de dados de previsão de risco de diabetes em estágio inicial do Repositório de Aprendizagem de Máquina UC Irvine. O conjunto de dados contém um total de 520 amostras, desse total há 328 homens e 192 mulheres, ou 63,08% e 36,92% respectivamente. Dos 320 casos positivos, 45,94% são do sexo masculino e 54,06% do sexo feminino.

O conjunto de dados é dividido em 17 atributos, como mostra a Tabela 2.

Tabela 2 – Descrição dos atributos do conjunto de dados Early stage diabetes risk prediction

Atributos	Valores
Idade	16-90
Sexo	1.Masculino, 2.Feminino
Poliúria	1.Sim, 0.Não
Polidipsia	1.Sim, 0.Não
Perda de peso repentina	1.Sim, 0.Não
Fraqueza	1.Sim, 0.Não
Polifagia	1.Sim, 0.Não
Tordo genital	1.Sim, 0.Não
Embaçamento visual	1.Sim, 0.Não
Coceira	1.Sim, 0.Não
Irritabilidade	1.Sim, 0.Não
Cura retardada	1.Sim, 0.Não
Paresia parcial	1.Sim, 0.Não
Rigidez muscular	1.Sim, 0.Não
Alopecia	1.Sim, 0.Não
Obesidade	1.Sim, 0.Não
Classe	1.Positivo, 0.Negativo

3.2.3 Sonar, Mines vs. Rock

Neste estudo, também foi utilizado o conjunto de dados Sonar, obtido através do repositório datahub.io. O conjunto de dados foi contribuído para a coleção de referência por Terry Sejnowski, agora no Salk Institute e na Universidade da Califórnia em San Deigo. O conjunto de dados foi desenvolvido em colaboração com R. Paul Gorman, do Allied-Signal Aerospace Technology Center.

O conjunto contém 208 casos padrão de sinais de sonar. O sinal de sonar transmitido é um chiado de frequência modulada, aumentando em frequência. O conjunto de dados contém sinais obtidos de uma variedade de ângulos de aspecto diferentes, abrangendo 90 graus para o cilindro e 180 graus para a rocha.

Cada padrão é um conjunto de 60 números no intervalo de 0,0 a 1,0. Cada número representa a energia dentro de uma determinada banda de frequência, integrada ao longo de um determinado período de tempo. O rótulo associado a cada registro contém a letra "R" se o objeto for uma rocha e "M" se for uma mina (cilindro de metal).

3.3 Validação cruzada

Pertencente à família dos métodos de Monte Carlo, a validação cruzada ou Cross-validation, é um método estatístico que avalia a capacidade de generalização de modelos de machine learning, com base em um conjunto de dados. O uso dessa técnica é amplamente fundamental em problemas cujo objetivo da modelagem é estimar o erro de predição, ou seja, o quão preciso um modelo é na prática.

O uso da validação cruzada pode ser com diferentes tipos de métodos, o método empregado neste trabalho é o K-fold (BURMAN, 1989). A utilização do método K-fold, resulta em um modelo menos tendencioso em comparação com outros métodos.

No método K-fold, os dados de entrada são divididos em subconjuntos de dados K (ou folds), o modelo é treinado usando as dobras como dados de treinamento, menos em um $(K-1)$ dos conjuntos de dados, seguidamente, o modelo resultante é avaliado no conjunto de dados restante, ou seja, é usado como um conjunto de testes com a finalidade de calcular o erro de predição. Todo o processo é repetido K vezes, e a cada repetição tal processo utiliza um conjunto diferente reservado para avaliação (e excluído do treinamento). A precisão com a validação cruzada se dá pela média de todas as precisões obtidas nos conjuntos de validação.

A Figura 10 exemplifica a esquematização da validação cruzada com $K = 3$.

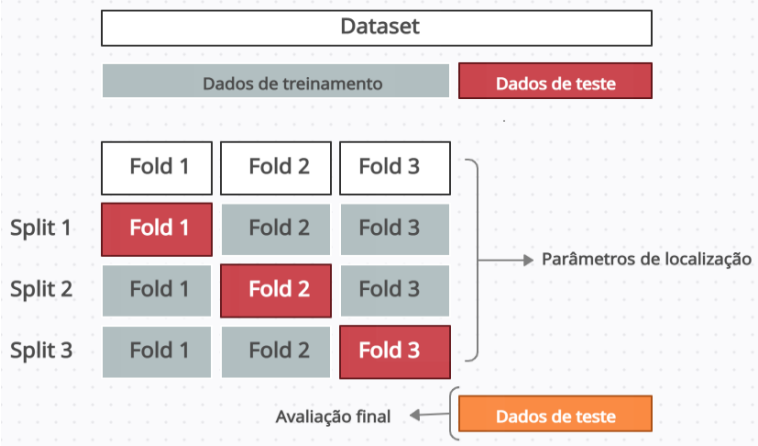

Figura 10 – Validação cruzada de um único modelo com $K = 3$

A estimativa de validação cruzada do erro de predição é da seguinte forma:

$$CV(\hat{f}) = \frac{1}{N}\sum_{i=1}^{N} L\left(y_i, \hat{f}^{-k(i)}(x_i)\right) \tag{26}$$

onde $\hat{f}^{-k}(x_i)$ é a função ajustada, calculada coma k-ésima parte dos dados.

3.4 Área sob a curva ROC (AUC)

ROC (características operacionais do receptor) é uma curva de probabilidade que mostra o desempenho de um modelo de classificação e plota dois parâmetros, taxa verdadeiro positivo e taxa falso positivo. Ela mostra o número de vezes que um classificador acertou contra o número de vezes que tal classificador errou em uma predição.

Os parâmetros da curva ROC são dados da seguinte forma:

- Taxa de verdadeiro positivo (True Positive Rate), é dado por verdadeiro positivo / (verdadeiro positivo + falso negativo)
- Taxa de falso positivo (False Positive Rate), é dado por falso positivo / (falso positivo + verdadeiro negativo)

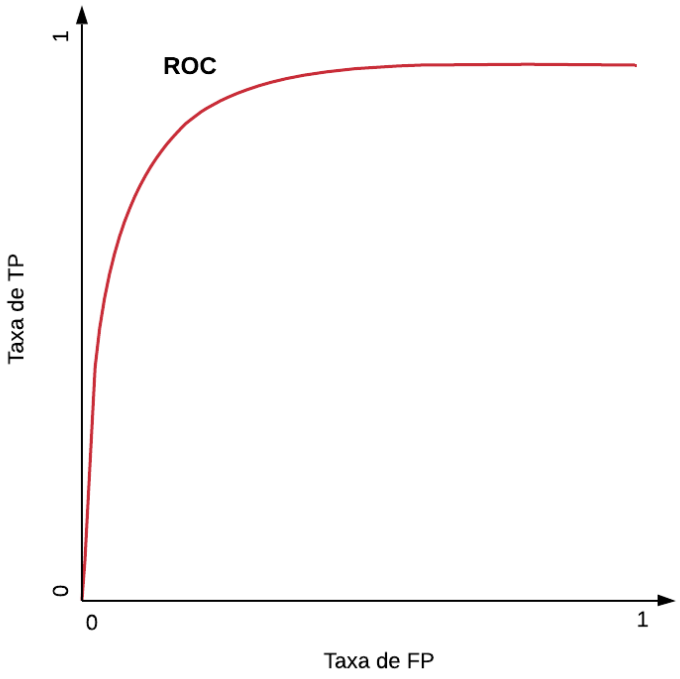

Figura 11 – Curva ROC

Uma curva ROC traça "Taxa de verdadeiro positivo vs. taxa de falso positivo" em diferentes limiares de classificação. A figura 11 mostra uma curva ROC.

Com o objetivo de simplificar a curva ROC, foi criada a área sob a curva, abreviado AUC (BRADLEY, 1997) (HANLEY; MCNEIL, 1982). A AUC possui a função de medir toda a área bidimensional sob toda a curva ROC e o seu valor sempre estará entre 0 e 1,0.

Modelos de predição cujo o valor da AUC esteja próximo de 1, possui uma boa medida de separabilidade, já os modelos cujo valor da AUC esteja próximo de 0, possui uma pior medida de desempenho, significa que o modelo esteja prevendo 1s como 0s e 0s como 1s. Quando o modelo não consegue fazer uma separação de classes, significa que o valor da AUC é 0,5. Quanto maior o valor da AUC, melhor será seu desempenho em predições.

A maneira mais simples de calcular a área sob a curva ROC é usando a integração trapezoidal,

$$AUC = \sum_i \left\{ (1 - \beta_i \cdot \Delta\alpha) + \frac{1}{2}[\Delta(1-\beta) \cdot \Delta\alpha] \right\}, \tag{27}$$

onde

$$\Delta(1-\beta) = (1-\beta_i) - (1-\beta_{i-1}), \tag{28}$$

$$\Delta\alpha = \alpha_i - \alpha_{i-1} \tag{29}$$

A AUC do classificador é equivalente à probabilidade de que o classificador classifique instâncias positivas selecionadas aleatoriamente mais alto do que instâncias negativas selecionadas aleatoriamente. Em comparação com classificadores de AUC baixa, classificadores de AUC alta podem ter pior desempenho em regiões específicas do espaço ROC.

AUC é desejável por dois motivos:

- AUC é a escala inalterada. Ele mede as classificações previstas, não seu valor absoluto.
- AUC é a classificação invariante de limiar. Meça a qualidade das previsões do modelo, independentemente do limite de classificação selecionado.

No entanto, esses dois motivos são acompanhados por avisos, que podem limitar a utilidade da AUC em certos casos de uso:

- A invariância de escala nem sempre é desejável. Por exemplo, às vezes precisamos de uma saída de probabilidade bem calibrada e a AUC não nos mostra isso.

- A invariância do limite de classificação nem sempre é desejável. Em situações em que há uma grande diferença no custo de falsos negativos e falsos positivos, minimizar um tipo de classificação incorreta pode ser crucial.

3.5 Otimização dos parâmetros SVM com base no algoritmo PSO

Dois fatores são essenciais para determinar os parâmetros otimizados através do algoritmo enxame de partículas (PSO): o primeiro fator é definir a posição da partícula que irá representar esses parâmetros, e o segundo fator é o uso da função de aptidão (função fitness), que retornará o melhor valor de scores e consequentemente avaliará o quão bom é uma partícula. Neste trabalho, empregamos a validação cruzada, como método da função fitness, além da função de kernel RBF (Função de Base Radial), pois tal função possui a capacidade de analisar dados não-linear e possibilita o uso dos parâmetros C e $Gama$, no qual C é um parâmetro de regularização usado para definir a tolerância do modelo para permitir a má classificação dos pontos de dados, a fim de alcançar um erro de generalização mais baixo, e $Gama$ é um parâmetro que define até onde a influência de um único exemplo de treinamento chega, com valores baixos que significam longe e altos valores que significam perto.

O algoritmo PSO é implementado para obter subconjuntos de soluções, com o objetivo de reduzir o grande número de possíveis soluções a serem posteriormente classificadas. O uso do classificador SVM é sempre requisitado quando a função de aptidão (função fitness) é necessário.

Neste trabalho a formulação do algoritmo PSO foi utilizado de duas maneiras diferentes, a primeira formulação é a proposta por (SHI; EBERHART, 1998), e a segunda foi a junção do algoritmo proposto por (SHI; EBERHART, 1998) com o conceito de Coeficientes de Aceleração Variados no Tempo (TVAC) introduzidos em (RATNAWEERA; HALGAMUGE; WATSON, 2004) juntamente com o Peso de Inércia Variável no Tempo (TVIW) (EBERHART; SHI, 2000).

No algoritmo de otimização dos parâmetros C e $Gama$ do SVM com base em PSO, cada partícula possui duas dimensões. O algoritmo de otimização é descrito da seguinte forma:

1. Processo de inicialização:

    - Para o conjunto de dados Breast Cancer Wisconsin defina os valores para para as constantes c_{1f}, c_{1i}, c_{2f} e c_{2i}, defina a quantidade de partículas e a quantidade de interações e os valores de w_{max} e w_{min}. Para a execução do algoritmo com os conjuntos de dados Early stage diabetes risk prediction e Sonar, definir os valores para as constantes de aceleração do melhor individual c_1 e global c_2 e para a constante de peso de inércia w.

- Estabeleça uma população PSO de duas dimensões e para cada partícula inicie a localização e velocidade.
- Inicie o *Pbest* de cada partícula por sua localização atual, faça o cálculo do valor de aptidão de cada partícula e defina o *Pbest* com o melhor valor de aptidão (scores) dentre todas as partículas, da mesma forma defina para o *gbest*.

2. Repita esse processo até que a solução não tenha alteração no valor ou até que alcance a quantidade máxima de iterações.

   - Para o conjunto de dados Breast Cancer Wisconsin faça o cálculo dos valores para as constantes de aceleração do melhor individual c_1 e global c_2 e para a constante de peso de inércia w. Para a execução do algoritmo com os conjuntos de dados Early stage diabetes risk prediction e Sonar não será necessário esta etapa, pois os valore de tais variável já foram pré-definidas no inicio.
   - Altere a localização e a velocidade de cada partícula.
   - Faça o cálculo do valor de aptidão de cada partícula, se o resultado desse cálculo do novo local for melhor do que o valor de aptidão (scores) do *Pbest*, *Pbest* recebe a nova localização.
   - Se a melhor partícula da população for melhor do que o *gbest*, *gbest* recebe essa melhor partícula.

3. Retorne os valores ótimos de C e $Gama$.

4. Obtenha a melhor estimativa de desempenho do classificador SVM.

   - Após a otimização, os parâmetros ótimos C e $Gama$ são trazidos para o modelo SVM para treinar o conjunto de treinamento, e o melhor desempenho do classificador (scores) é obtido.

A Figura 12 demonstra o processo de otimização dos parâmetros (C, $Gama$) do algoritmo SVM, com o uso do algoritmo PSO.

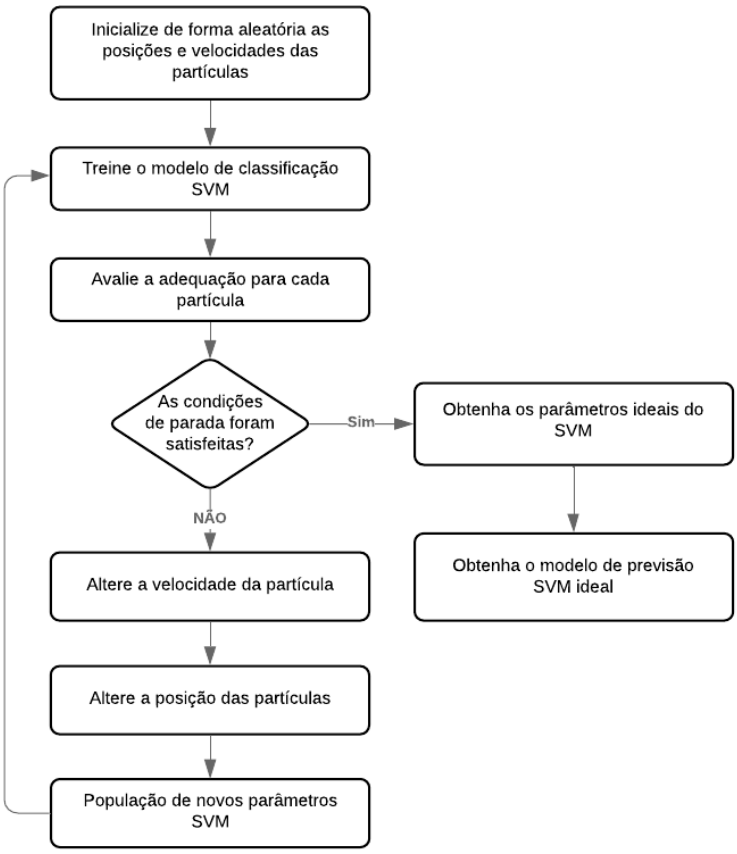

Figura 12 – Fluxograma representando o processo de otimização dos parâmetros do SVM com o PSO

4 RESULTADOS

O experimento empírico foi executado em uma CPU Intel(R) Core(TM) i5-3337U (1.80GHz) com 6GB de RAM. Os dados foram normalizados para evitar que os valores das características em faixas numéricas maiores dominassem aquelas em faixas numéricas menores.

Para garantir os resultados válidos, a análise de validação cruzada k-fold foi utilizada para avaliar a precisão da classificação. De modo a obter o melhor valor de k, os parâmetros ideais e a melhor estimativa de desempenho do classificador SVM, a cada rodada de testes foi definido para a validação cruzada um valor para k de 2 a 10.

Para todas as bases de dados foi utilizado a função de kernel RBF, pois tal função além de possibilitar o uso dos parâmetros C e $Gama$ e de poder analisar dados não-lineares é a função de kernel mais utilizada no algoritmo PSO-SVM.

Os dados de teste usados no estágio de testes são isolados dos dados de treinamento, o melhor par de parâmetros para o SVM e o subconjunto de recursos são obtidos do conjunto de dados de treinamento, e o conjunto de dados de teste é usado para obter o valor médio da precisão da validação cruzada na fase de testes, evitando a superestimativa da precisão.

A área sob a Curva Característica de Operação do Receptor (AUC) foi utilizada no experimento para testar o desempenho do modelo PSO-SVM proposto, e apresentar uma exibição gráfica que fornece a medida da precisão preditiva dos modelos, conforme utilizado em (CHEN et al., 2012). A curva exibe a taxa de verdadeiro positivo e a taxa de falso positivo. AUC é a área sob a curva ROC, que é uma das métricas de avaliação mais importantes para verificar o desempenho de modelos de classificação.

Para o primeiro experimento utilizando o conjunto de dados Breast Cancer Wisconsin foi adotado o conceito de Coeficientes de Aceleração Variados no Tempo (TVAC) juntamente com a equação conhecida como o peso de inércia variável no tempo (TVIW). Os detalhes da configuração dos parâmetros para o algoritmo PSO-SVM utilizando o conjunto de dados Breast Cancer Wisconsin é definido da seguinte forma: o número de iterações e partículas é definido como 200 e 30, respectivamente, w_{max} e w_{min} são definidos como 0,9 e 0,4 respectivamente, como sugerido em (CHEN et al., 2012), c_{1i} = 2,5, c_{1f} = 0,5, c_{2i} = 0,5 e c_{2f} = 2,5 conforme sugerido em (RATNAWEERA; HALGAMUGE; WATSON, 2004). A Tabela 3 demonstra os valores de partida para os parâmetros c_1, c_2 e w, bem como os melhores valores para taís parâmetros, obtidos através do conceito TVAC juntamente com a equação TVIW.

A Tabela 4 mostra as taxas de score da classificação e os valores ideais de (C, Gama) para cada fold, e tais valores são os resultados ótimos obtidos através do algoritmo PSO-SVM. Pode-se observar que, a taxa média de score alcançada pelo algoritmo desenvolvido

Tabela 3 – Valores para os parâmetros c_1, c_2 e w do PSO

Parâmetros	Valores de partida	Melhores valores
c_1	2.5	2.21
c_2	0.5	0.79
w	0.9	0.8275

é de 99,524%, e o melhor score alcançado foi de 99,776% para k = 4. A Figura 13 mostra a curva de convergência do algoritmo PSO, a cor azul refere-se ao parâmetro C e a cor vermelha ao parâmetro *Gama*. A Figura 14 nos mostra um valor de AUC próximo de 1, provando assim, que o modelo de predição possui uma boa medida de desempenho.

Tabela 4 – Taxas de score e configurações dos parâmetros C e Gama (valores ótimos) ideais para o banco de dados WBDC usando PSO-SVM

Fold	C	Gama	Score(%)
2	2.5100674541374413	1.3221464025165792	99,620
3	2.613514883397373	1.6055898091684373	99,525
4	2.4868489895598387	1.1504216358795274	99,776
5	2.9726013262071413	1.3034107908267454	99,431
6	2.196125722065326	1.2492355592812228	99,697
7	1.5902077898826357	0.660011174732572	99,212
8	1.2870285530385468	1.5289659029197948	99,764
9	1.9842103574840644	1.262619444721061	99,287
10	1.6054900652924682	3.129520037363293	99,408

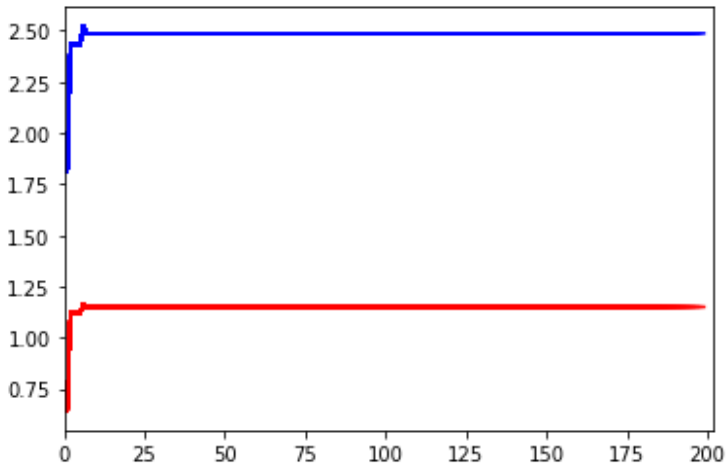

Figura 13 – Curva de convergência dos parâmetros C e Gama do algoritmo PSO para base de dados WBDC com o valor de k = 4

Os detalhes da configuração dos parâmetros para o algoritmo PSO-SVM utilizando o conjunto de dados Early stage diabetes risk prediction é definido da seguinte forma: o número de iterações e partículas é definido como 130 e 45, respectivamente, w = 0.8, c_1 = 0.5, c_2 = 0.5, como sugerido em (TANG et al., 2021).

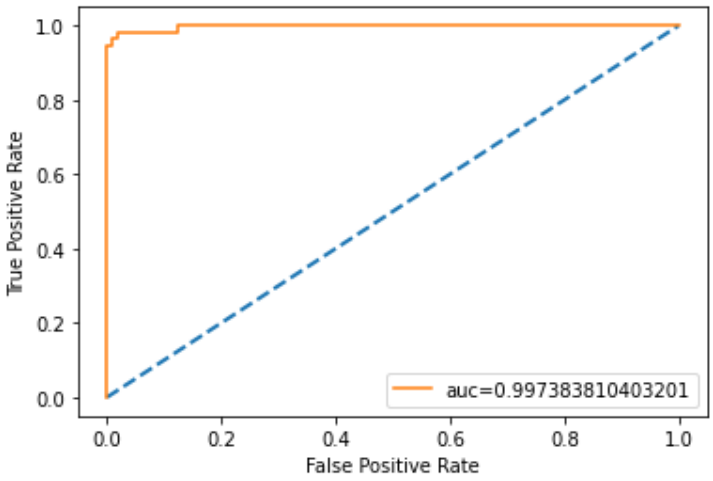

Figura 14 – Área sob a curva ROC (AUC) referente ao melhor resultado obtido da base de dados WBDC

A Tabela 5 mostra as taxas de score da classificação e os pares ideais de (C, Gama) para cada fold. Pode-se observar que, a taxa média de score alcançada pelo algoritmo desenvolvido é de 99,476%, e o melhor score alcançado foi de 99,673% para k = 8. A Figura 15 mostra a curva de convergência do algoritmo PSO, a cor azul refere-se ao parâmetro C e a cor vermelha ao parâmetro $Gama$. A Figura 16 nos mostra um valor de AUC próximo de 1, provando assim, que o modelo de predição possui uma boa medida de desempenho.

Tabela 5 – Taxas de score e configurações de parâmetros ideais para o banco de dados Early stage diabetes risk prediction usando PSO-SVM

Fold	C	Gama	Score(%)
2	8.014304890600307	3.1324777051532684	99,151
3	7.951995724411524	3.0881407637771505	99,444
4	7.766101682650427	3.029149650647459	99,563
5	7.802763438880954	3.1303080469080506	99,473
6	8.11019288138817	3.0270463299756853	99,453
7	7.958441366432409	3.134349482788844	99,511
8	7.229323546767429	3.0434553149187815	99,673
9	7.920612551061479	3.1317352086974424	99,460
10	8.274927251704433	3.1275135481628777	99,557

Para o terceiro experimento foi utilizado a base de dados Sonar. Os detalhes da configuração dos parâmetros para o algoritmo PSO-SVM é definido da seguinte forma: o número de iterações é 100 e a quantidade de partículas é 20, $w = 0.9$, $c_1 = c_2 = 2$, como sugerido em (GAO; PENG; LI, 2010).

A Tabela 6 mostra as taxas de score da classificação e os pares ideais de (C, Gama) para cada fold. Pode-se observar que, a taxa média de score alcançada pelo algoritmo desenvolvido é de 94,681%, e o melhor score alcançado foi de 96,329% para k = 5. A

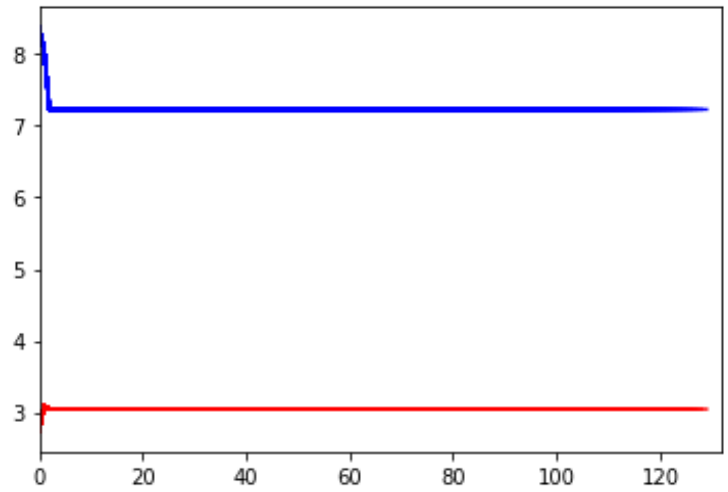

Figura 15 – Curva de convergência dos parâmetros C e Gama do algoritmo PSO para base de dados Early stage diabetes risk prediction com o valor de k = 8

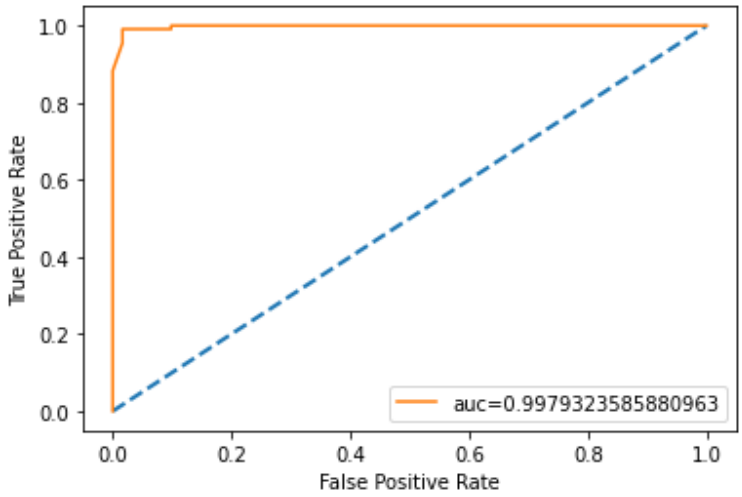

Figura 16 – Área sob a curva ROC (AUC) referente ao melhor resultado obtido da base de dados Early stage diabetes risk prediction

Figura 17 mostra a curva de convergência do algoritmo PSO, a cor azul refere-se ao parâmetro C e a cor vermelha ao parâmetro $Gama$. A Figura 18 nos mostra um valor de AUC próximo de 1, provando assim, que o modelo de predição possui uma boa medida de desempenho.

Tabela 6 – Taxas de score e configurações de parâmetros ideais para o banco de dados Sonar usando PSO-SVM

Fold	C	Gama	Score(%)
2	9.455253876595856	1.192721921699805	94,167
3	4.421457835017401	2.303494112244448	93,110
4	7.7316344641054835	1.424845779239078	93,633
5	3.7654986980202994	1.203183456507725	96,329
6	4.663588008418969	1.7913716398159045	95,792
7	8.122237878384905	1.4091309298593697	93,172
8	4.097208218917757	1.8289700496859462	96,077
9	2.167193361335451	1.0003476546785406	95,180
10	4.025995600012402	1.9263639396594878	94,669

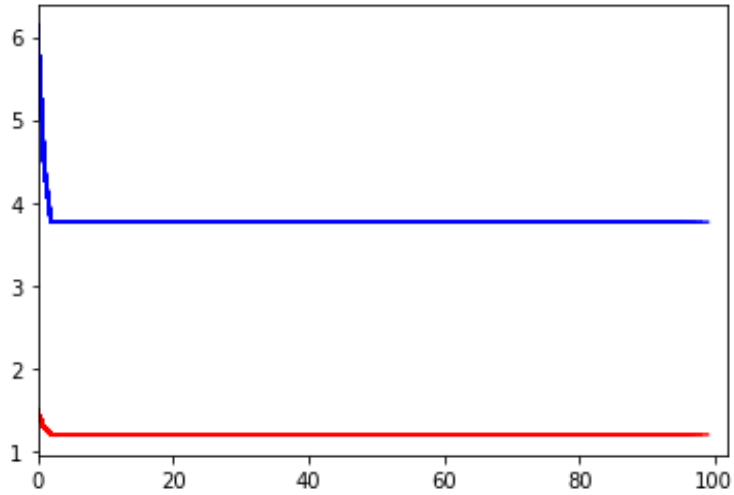

Figura 17 – Curva de convergência dos parâmetros C e Gama do algoritmo PSO para base de dados Sonar com o valor de k = 5

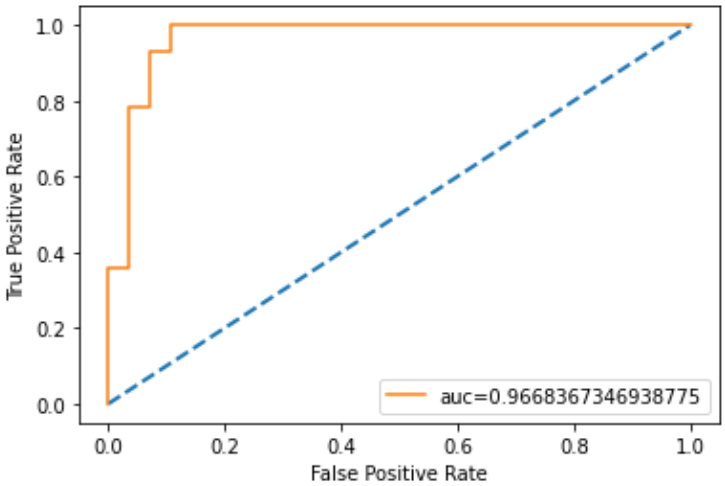

Figura 18 – Área sob a curva ROC (AUC) referente ao melhor resultado obtido da base de dados Sonar

5 CONCLUSÃO

Os parâmetros da máquina de vetores de suporte (SVM) são de grande importância para o sucesso dos SVMs. Para otimizar os parâmetros adequadamente, este trabalho apresenta um método baseado na otimização por enxame de partículas (PSO), que retorna valores ótimos dos parâmetros do algoritmo SVM para obter um subconjunto de características. O subconjunto é, portanto, utilizado tanto no treinamento quanto no teste para obtenção dos resultados ideais na classificação. Os resultados apresentados neste trabalho mostram que a abordagem PSO-SVM desenvolvida possui uma boa precisão de classificação.

Os resultados são obtidos por meio da análise de validação cruzada, utilizada na avaliação da precisão da classificação, da área sob a curva ROC (AUC), utilizada para testar o desempenho do modelo PSO-SVM e da função de kernel RBF. No entanto, outras funções de kernel e outros métodos de avaliação de precisão também podem ser utilizados na abordagem PSO-SVM. Os resultados experimentais obtidos a partir de conjuntos de dados UCI, outros conjuntos de dados públicos e problemas reais podem ser testados no futuro para verificar e estender este método.

REFERÊNCIAS

BRADLEY, A. P. The use of the area under the roc curve in the evaluation of machine learning algorithms. **Pattern recognition**, Elsevier, v. 30, n. 7, p. 1145–1159, 1997.

BURMAN, P. A comparative study of ordinary cross-validation, v-fold cross-validation and the repeated learning-testing methods. **Biometrika**, Oxford University Press, v. 76, n. 3, p. 503–514, 1989.

CHEN, H.-L. et al. Support vector machine based diagnostic system for breast cancer using swarm intelligence. **Journal of medical systems**, Springer, v. 36, n. 4, p. 2505–2519, 2012.

CRISTIANINI, N.; SHAWE-TAYLOR, J. et al. **An introduction to support vector machines and other kernel-based learning methods**. [S.l.]: Cambridge university press, 2000.

EBERHART, R. C.; SHI, Y. Comparing inertia weights and constriction factors in particle swarm optimization. In: IEEE. **Proceedings of the 2000 congress on evolutionary computation. CEC00 (Cat. No. 00TH8512)**. [S.l.], 2000. v. 1, p. 84–88.

GAO, J.; PENG, J.-Y.; LI, Z. Application of improved pso-svm approach in image classification. In: IEEE. **2010 Symposium on Photonics and Optoelectronics**. [S.l.], 2010. p. 1–4.

GLOVER, F. Future paths for integer programming and links to artificial intelligence. **Computers & operations research**, Elsevier, v. 13, n. 5, p. 533–549, 1986.

HANLEY, J. A.; MCNEIL, B. J. The meaning and use of the area under a receiver operating characteristic (roc) curve. **Radiology**, v. 143, n. 1, p. 29–36, 1982.

HAYKIN, S. **Redes neurais: princípios e prática**. [S.l.]: Bookman Editora, 1999.

KENNEDY, J.; EBERHART, R. Particle swarm optimization. In: IEEE. **Proceedings of ICNN'95-international conference on neural networks**. [S.l.], 1995. v. 4, p. 1942–1948.

OSMAN, I. H.; KELLY, J. P. Meta-heuristics theory and applications. **Journal of the Operational Research Society**, Taylor & Francis, v. 48, n. 6, p. 657–657, 1997.

RATNAWEERA, A.; HALGAMUGE, S. K.; WATSON, H. C. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. **IEEE Transactions on evolutionary computation**, IEEE, v. 8, n. 3, p. 240–255, 2004.

REYNOLDS, C. W. Flocks, herds and schools: A distributed behavioral model. In: **Proceedings of the 14th annual conference on Computer graphics and interactive techniques**. [S.l.: s.n.], 1987. p. 25–34.

SHI, Y.; EBERHART, R. A modified particle swarm optimizer. In: IEEE. **1998 IEEE international conference on evolutionary computation proceedings. IEEE world congress on computational intelligence (Cat. No. 98TH8360)**. [S.l.], 1998. p. 69–73.

SÖRENSEN, K.; GLOVER, F. Metaheuristics. **Encyclopedia of operations research and management science**, Springer New York, v. 62, p. 960–970, 2013.

TANG, H. et al. Risk prediction of early diabetes mellitus based on combination model. In: EDP SCIENCES. **MATEC Web of Conferences**. [S.l.], 2021. v. 336, p. 07018.

VAPINK, V. The nature of statistical learning theory. **M. Springer, Verlag**, 1995.

ZHU, Z.; ZHANG, W. B.; LIU, W. Speech recognition based on fuzzy support vector machine. **Computer Engineering**, v. 2, p. 186–188, 2006.