UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

DEPARTAMENTO DE ENGENHARIA DE MATERIAIS

CLASSIFICAÇÃO PREDITIVA DE FASES PARA LIGAS MULTICOMPONENTES CrCoFeMnNi UTILIZANDO MACHINE LEARNING

Kayque Rodrigues Santos

SÃO CARLOS - SP
2023

CLASSIFICAÇÃO PREDITIVA DE FASES PARA AS LIGAS MULTICOMPONENTES CrCoFeMnNi UTILIZANDO MACHINE LEARNING

Trabalho de conclusão de curso apresentado ao Departamento de Engenharia de Materiais da Universidade Federal de São Carlos, como requisito para obtenção do título de bacharel em Engenharia de Materiais.

Orientador: Francisco Gil Coury

Coorientador: Pedro Oliveira

São Carlos - SP
2023

UNIVERSIDADE FEDERAL DE SÃO CARLOS
CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA DE MATERIAIS
Telefones: 16 –3351-8244 e 3351-8246
Endereço eletrônico: demachef@ufscar.br
Rodovia Washington Luís, km 235 – Caixa Postal 676
CEP 13565-905 – São Carlos – SP - Brasil

ATA DE DEFESA DE TRABALHO DE CONCLUSÃO DE CURSO (TCC)

NOME: Kayque Rodrigues Santos

RA: 744396

TÍTULO: Classificação preditiva de fases para as ligas multicomponentes CrCoFeMnNi utilizando machine learning

ORIENTADOR(A): Prof. Dr. Francisco Gil Coury

CO-ORIENTADOR(A): Dr. Pedro Oliveira

DATA/HORÁRIO: 16/02/2022, 10h

BANCA – NOTAS:

	Monografia	**Defesa**
Prof. Dr. Francisco Gil Coury	10.0	10.0
Prof. Dr. Guilherme Zepon	10.0	10.0
Média	10.0	10.0

BANCA – ASSINATURAS:

Prof. Dr. Francisco Gil Coury _____

Prof. Dr. Guilherme Zepon _____

RESUMO

Quando nos referimos a ligas multicomponentes, ou ligas de alta entropia, como são comumente chamadas, é inevitável não se deparar com o desafio de se explorar novas composições que possam ser do interesse científico ou de engenharia. Essa dificuldade está relacionada à gama composições possíveis, dado a quantidade de diferentes elementos e seus respectivos percentuais que podem ser combinados para originar novas ligas.

Desde a descoberta das ligas multicomponente, métodos alternativos vêm sendo estudados para tentar prever algumas características dessas ligas, sem a necessidade de realizar as etapas de processamento e caracterização, ou a utilização de outros métodos empíricos. Tais métodos alternativos, além de economizar tempo, podem otimizar recursos e tornar economicamente mais viável a investigação de um maior número de composições, até se alcançar um material que justifique sua confecção para análises mais aprofundadas.

Dentre os modos existentes para fazer essa exploração, podemos destacar métodos baseados na teoria funcional da densidade, ou ainda simulações termodinâmicas, que podem utilizar de diferentes metodologias, como o CALPHAD que utilizada das funções energéticas de Gibbs. Porém, esses são métodos que ainda possuem ciclos de desenvolvimentos relativamente longos.

Partindo disso, esse trabalho destina-se a utilização da ciência orientada por big data, mais especificamente o aprendizado de máquina (do inglês *Machine learning*), onde a partir de uma base de dados, originada por meio da simulação termodinâmica, utilizando como método o CALPHAD, foram elaborados três diferentes algoritmos para realizar a classificação de fases de forma preditiva em ligas multicomponentes, formadas pelos elementos de níquel (Ni), manganês (Mn), ferro (Fe), cromo (Cr) e cobalto (Co). Essa classificação consiste em prever se uma dada composição a uma temperatura constante de 1000ºC, apresenta as fases com estrutura cúbica de face centrada, cúbica de corpo centrado ou a fase sigma, ou ainda, se não apresenta nenhuma dessas sendo agrupadas como “outros”.

Com isso, através de uma base de dados com 1000 composições diferentes, foi possível realizar o treinamento supervisionado de três diferentes tipos de algoritmos, onde, após devidamente treinados e otimizados, realizaram a classificação de cerca de 494 novas combinações não existentes na base inicial. Como resultado uma acurácia de 91,72% foi atingida para o algoritmo de árvore de decisão, 94,95% para o k-vizinhos mais próximos e 96,36% para a máquina vetor de suporte.

Palavras-chave: Aprendizado de Máquina. Classificação. Ligas Multicomponentes. Ligas de Alta Entropia. Previsão de Fases. Simulação Preditiva. Árvore de Decisão. K-vinhos mais próximos. Máquina vetor de Suporte.

ABSTRACT

When we refer to multicomponent alloys, or high entropy alloys as they are commonly called, it is inevitable to discuss the challenge of exploring new compositions that may be of scientific interest. This difficulty is related to a range of possible compositions, given the amount of different elements and their percentages that can be combined to originate new materials.

Since its discovery, alternative methods have been studied to try to predict some characteristics of these alloys without the need to experimentally produce them, or to use empirical methods, this, in addition to saving time, can optimize resources and makes it economically more feasible to investigate a greater number of combinations, until a material is reached that justifies its manufacture for further analysis.

Among the existing ways to carry out this exploration, we can highlight methods based on the functional theory of density, or even thermodynamic simulations, which can use different methods, such as CALPHAD, which uses the Gibbs energy functions. However, these are methods that still have relatively long development cycles.

Based on this, this work is intended to use science guided by big data, more specifically machine learning, where from a database, originated through thermodynamic simulation, using CALPHAD as a method, three different algorithms to predictively classify phases in multicomponent alloys, formed by the elements nickel (Ni), manganese (Mn), iron (Fe), chromium (Cr) and cobalt (Co). This classification consists of predicting whether a given composition at a constant temperature of 1000ºC presents a face-centered cubic, body-centered cubic and sigma structure as a phase, or whether none of these are present and are grouped under “others”.

With that, through a database with 1000 different compositions, it was possible to carry out supervised training of three different types of algorithms, where, after properly trained and optimized, they performed the classification of about 494 new combinations that did not exist in the initial base. As a result, an accuracy of 91.72% was reached for the decision tree algorithm, 94.95% for the k-nearest neighbors and 96.36% for the support vector machine.

Keywords: Machine Learning. Classification. Multicomponent Alloys. High Entropy Leagues. Phase Forecast. Predictive Simulation. Decision tree. Closest K-wines. Support vector machine.

LISTAS DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS

AM – Aprendizado de Máquina

CCC - Cúbico de Corpo Centrado

CFC - Cúbico da Face Centrada

Co – Cobalto

Cr – Cromo

CV - Cross-Validation

DFT - Density-functional theory

DRX – Difração de Raios X

DT - Decision Tree

Fe – Ferro

GS - Grid Search

HEA - High Entropy Alloys

ID3 - Iterative Dichotomiser 3

KNN - K-Nearest Neighbors

LAE - Ligas de Alta Entropia

LMM – Ligas Metálicas Multicomponentes

ML - Machine Learning

Mn - Manganês

nCV- nested Cross-Validation (validação cruzada aninhada)

Ni – Níquel

RBF - Gaussian Radial Basis Function

RS - Random search

S_C – Entropia configuracional

SVM - Support Vector Machine

UFSCar - Universidade Federal de São Carlos

VEC - Valence Electron Concentration

SUMÁRIO

1. INTRODUÇÃO

No mundo que tange a ciência e a engenharia dos materiais, é comum nos depararmos com estudos relacionados ao desenvolvimento e manipulação de composições e estruturas, a fim de se obter as mais adequadas propriedades de um material. Isso, com base em algum objetivo principal, ou ainda motivado por uma aplicação específica, na qual o desempenho possa ser o melhor possível, caso exista essa otimização.

Isso também pode ser visto entre os desafios relacionados ao desenvolvimento de novas Ligas Metálicas Multicomponentes (LMM) ou como também são conhecidas, Ligas de Alta Entropia (LAE), do inglês do inglês – *Higth Entropy Alloys* (HEA), os quais são uma classe de materiais metálicos que se diferem das ligas convencionais vistas ao longo da história, em virtude das LAEs não apresentarem um único elemento base em sua composição, ou seja, contrariamente a metalurgia tradicional, onde as ligas são compostas majoritariamente de um elemento principal na liga, nas LAEs não existem esse elemento metal base [1]. Um dos primeiros trabalhos no tema define HEAs como "aqueles compostos por cinco ou mais elementos principais em proporções equimolares" [2]. No mesmo artigo essa definição se expande para incluir "elementos principais com a concentração de cada elemento entre 35 e 5 % at". Assim, as HEAs não precisam ser necessariamente equimolares, aumentando significativamente o número de possibilidades. As HEAs também podem conter elementos em menores proporções para modificar as propriedades da HEA base, expandindo ainda mais o número de ligas possíveis.

Desde a descoberta dessa nova classe de material, há um grande interesse e intensificação de estudos tanto no meio acadêmico quanto industrial, devido a quantidade de possibilidades inexploradas e o potencial para se obter combinações que resultem em ligas com propriedades superiores. [3-5]

Uma das grandes dificuldades de exploração de novas composições no campo das HEAs é justamente a gama de combinações possíveis entre os elementos para formar uma nova liga. Nesse sentido, a ideologia orientada ao design e exploração de novas ligas pode ser resumida em quatro paradigmas: o primeiro paradigma é o ensaio empírico e método de erro, o segundo são as leis físicas e químicas, o terceiro é a simulação por computador e o quarto paradigma é a ciência orientada por big data. Entre eles, graças ao desenvolvimento contínuo da tecnologia de mineração de dados e inteligência artificial, o quarto paradigma pode unificar perfeitamente os outros três nos aspectos de teoria, experimento e simulação computacional e, com isso, novos

métodos baseados em big data, como aprendizado de máquina, acabaram surgindo também nessa área de estudo da ciência dos materiais.

Métodos tradicionais para descoberta de novos materiais como o método empírico padrão de tentativa e erro, e métodos como o baseado na Teoria Funcional da Densidade (*Density-functional theory* - DFT), são incapazes de acompanhar o desenvolvimento da ciência dos materiais nos dias de hoje, devido aos seus longos ciclos de desenvolvimento, grande necessidade processual atrelada e altos custos [6].

Graças principalmente ao baixo custo computacional, o Aprendizado de Máquina, do inglês, *Machine Learning* (ML) se mostra como alternativa, sendo um poderoso meio para o processamento de dados, e com alto desempenho de previsão, além de estar sendo amplamente utilizado na detecção, análise e design de materiais. O aprendizado de máquina, pode reduzir substancialmente os custos computacionais, além de encurtar o ciclo de desenvolvimento e, portanto, pode ser uma das formas mais eficientes de substituir cálculos de DFT, ou mesmo experimentos laboratoriais repetitivos. De modo geral, o aprendizado de máquina usa grandes quantidades de dados para otimizar continuamente os modelos, e realizar previsões razoáveis sob a orientação de algoritmos [7,8].

Nesse sentido, é possível aproveitar esse poder de análise e processamento para realizar previsões sobre importantes aspectos das LAEs. Quando tratamos de ligas metálicas, mais especificamente sobre suas características, um dos aspectos que mais influenciam suas propriedades são as fases que material apresenta. As formações de fase nas LAEs ligas de alta entropia são essenciais para determinar suas principais propriedades, mas a previsão eficiente delas continua sendo um desafio.[9]

Por isso, neste trabalho, o obtivo e desafio girou em torno de elaborar e avaliar o uso de métodos de aprendizado de máquina para classificação de fases, de forma preditiva, com base nas composições e em informações adicionais sobre o material a ser classificado.

Para isso, foram construídos e explorados três tipos diferentes de algoritmos para realizar a classificação preditiva de fases em ligas multicomponentes. Diferentes modelos foram treinados e testados para classificar diferentes combinações dentro do espaço composicional de 5 elementos: o Níquel (Ni), Manganês (Mn), Ferro (Fe), Cromo (Cr) e Cobalto (Co). Essa classificação consiste em determinar se uma dada composição, a temperatura de 1000ºC,apresenta-se como um material monofásico Cúbico da Face Centrada (CFC), Cúbico de Corpo Centrado (CCC), ou apresenta a fase Sigma, ou ainda se não apresenta nenhuma dessas fases de forma única, sendo agrupados em “Outros”.

Esses elementos foram selecionados devido à sua grande relevância deles dentro do campo das ligas multicomponentes, fazendo com que exista uma gama de trabalhos existentes já conhecidos, fortalecendo a confiabilidade dos resultados a obtidos. A base de dados consiste em uma série de ligas obtidas por meio da simulação termodinâmica, utilizando o método CALPHAD no software Pandat ™.

Em relação aos algoritmos de aprendizado de máquina adotados, foram utilizados a chamada árvore de decisão (do inglês, *Decision Tree* -DT), k-vizinhos mais próximos (do inglês, K-Nearest Neighbors – KNN) e a máquina vetor de suporte (do inglês, Support Vector Machine - SVM). A escolha desses algoritmos se deve à sua frequente utilização para problemas de classificação, além do seu constante aparecimento em diversos estudos similares [9-11]. Para todos esses modelos foi elaborado a otimização e validação cruzada, com o objetivo de buscar o melhor conjunto de parâmetros, para maximizar os resultados de previsão, e minimizar as margens de erro na previsão de fases.

2. REVISÃO DA LITERATURA

2.1 LIGAS MULTICOMPONENTE

Normalmente quando nos referimos a ligas metálicas, é comum pensarmos em materiais metálicos cuja composição seja dada por algum elemento principal, acompanhado de elementos de liga, como por exemplo, aços carbono, ligas de alumínio, bronze, entre outros. Desde a antiguidade, por volta de 3000 a.C, conhecida como idade do bronze, a metalurgia convencional desenvolve ligas baseadas em um ou dois metais principais, com a adição de alguns elementos de liga em baixos teores [12]. Isso deu forma a maioria dos materiais metálicos conhecidos atualmente.

Entretanto, de forma divergente à metalurgia tradicional, há poucas décadas, em 2004, surgiram os primeiros trabalhos publicados mencionando as ligas de alta entropia (LAE), os quais se referiam a esforços conduzidos um pouco antes, em 1996, que foram diferentes de tudo que se havia feito até então.

Desde o aparecimento dessa nova classe de ligas, suas definições ganharam várias vertentes com base no que se estudava acerca desses materiais. Uma das primeiras definições, e muito utilizada até hoje, define as HEAs como “aqueles compostos por cinco ou mais elementos principais em proporções equimolares” [17], porém no mesmo trabalho essa definição é expandida para a incluir "elementos principais com a concentração de cada elemento entre 35 e 5 at.%". Dessa forma as HEAs não precisam ser equimolares, e ainda podem conter elementos em menores proporções a fim de modificar as propriedades da liga. [18]. Dada essa definição,

baseada na pluralidade de elementos que compõem essas ligas, um outro nome surgiu ao longo do tempo para se referir a tais materiais, ficando conhecidas também como Ligas Metálicas Multicomponentes (LMM). Outras definições aparecem na literatura, baseadas na entropia configuracional, e até mesmo com base em especificações de LMMs monofásicas ou não, porém existe uma maior complexidade nesse tipo de definição, principalmente quando estas aparecem misturadas, o que ocasiona uma certa divergência ao se olhar diferentes trabalhos [19].

Como conclusão acerca das definições, é possível afirmar que, diferentemente das ligas convencionais, nas LAEs não temos um elemento principal. Com isso, existe uma quantia grande de possibilidades, onde frequentemente é possível se deparar com resultados promissores, de um ponto de vista de propriedades. O que inspirara cada vez mais, a exploração do vasto espaço composicional existente para as ligas multicomponentes [19]. Através da Figura 1 é possível observar o crescimento exponencial de publicações cientificas que envolvem o termo "High Entropy Alloys" como tópico principal:

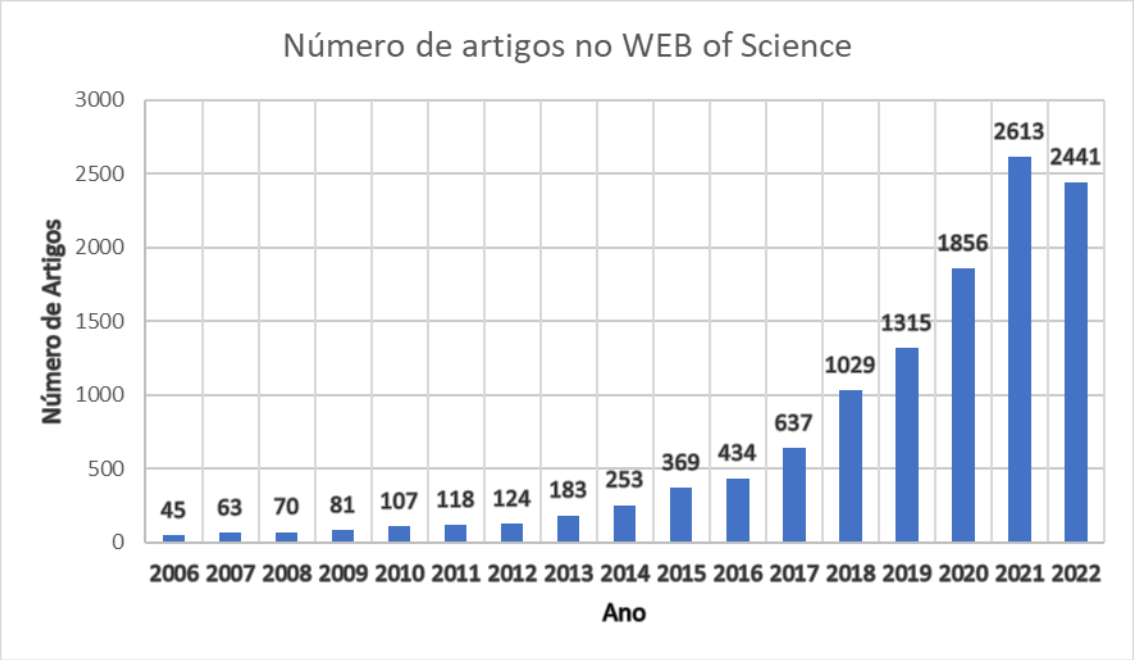


Figura 1: Número de publicações por ano, com "High Entropy Alloys" (Ligas de Alta Entropia), como tópico principal no Web of Science durante o período de 2006 a novembro de 2022. [20]

Um dos motivos das LAEs terem tanta atenção está relacionado justamente à vasta gama de combinações possíveis, e com ela a possibilidade de obter propriedades superiores àquelas de ligas convencionais, que podem ser alcançadas, a depender da composição da liga. Porém, essa quantia quase infinita de possibilidades, aumenta significativamente o número de ligas a serem caracterizadas. Mesmo em uma mesma família de ligas, o ajuste de apenas um elemento pode

ter um grande efeito na microestrutura e nas propriedades, uma vez que os elementos geralmente estão mais concentrados nas LMMs do que nas ligas convencionais. Sendo assim, o uso de métodos convencionais torna praticamente inviável a exploração rica de novos materiais através de meios convencionais. Uma das famílias de ligas mais amplamente estudadas contém pelo menos 4 dos 9 elementos a seguir: Al, Co, Cr, Cu, Fe, Mn, Ni, Ti e V [1,16,17,21,22,23,24]. Cinco desses 9 elementos juntos constituem a composição da liga conhecida como 'liga de Cantor' (CoCrFeMnNi), relatada pela primeira vez em 2004 [1]. Além de ser uma das primeiras LMM relatadas, esse também é um protótipo de liga de solução sólida desordenada monofásica, contribuindo para sua popularidade e, graças a isso, a existência de uma boa base de exploração a cerca dessa combinação, sendo uma das razões para compor o campo de estudo deste trabalho.

As ligas multicomponentes apresentam propriedades tão promissoras que vêm sendo consideradas como potenciais candidatas para um amplo grupo de aplicações. Estudos publicados relatam ultraelevada tenacidade à fratura [25], excedendo à de muitos metais e ligas metálicas, excelente resistência mecânica [26], comparável à de cerâmicos estruturais, supercondutividade [27] e resistência à corrosão significativa [28]. Porém, essas propriedades são provenientes de diferentes combinações, as quais podem aparecer conjuntamente, a depender da composição.

Contudo, o potencial para novas descobertas associadas a esses materiais mal foi riscado. Considerando os 72 elementos que não são gases tóxicos, radioativos ou nobres, o número de sistemas de 5 elementos está por volta de 13.991.544 e o número de sistemas com 3-6 elementos explode para 171.318.882. Cerca de 400 LMMs foram relatadas até o momento [19], e muitas delas são variações não equimolares dos mesmos elementos, dando apenas 112 combinações de elementos diferentes consideradas até agora. Dada a quantidade possível de sistemas e combinações, é natural que não seja viável a exploração empírica destas. Com isso, ao longo do tempo, foram surgindo e sendo refinadas formas alternativas para o design de novas ligas. Uma das principais metodologias adotadas é a simulação termodinâmica usando de métodos como o CALPHAD.

2.1.1 Simulação Termodinâmica

Os diagramas de equilíbrio são a chave para a compreensão da formação de LAEs, pois as fases de uma liga são essenciais para determinar suas principais propriedades, apesar da previsão eficiente delas ser sempre um grande desafio [9]. Normalmente, são determinados usando experimentos de equilíbrio, seguidos por análise térmica e identificação de fase, usando técnicas de difração e microscopia e, portanto, podem ser caros e demorados, especialmente

para os sistemas de ordem superior devido ao grande número possível de combinações. A mistura arbitrária de elementos da tabela periódica não resulta na formação de uma solução sólida monofásica, mas de estruturas compostas. Por exemplo, Cantor, investigou ligas de 20 e 16 componentes em proporções equimolares, e descobriu que ambas as ligas continham fases múltiplas, e eram frágeis nas formas de lingote fundido e fita fundida. Embora eles tenham fabricado uma liga FCC monofásica de CoCrFeMnNi com sucesso, sua tentativa de adicionar 1 a 4 elementos adicionais falhou [1].

Daí nasce a necessidade de desenvolver formas não empíricas para o estudo das LMM. O método CALPHAD [29,30], utilizado para simulações termodinâmicas, baseia-se na conhecida lei termodinâmica que postula que um sistema atinge seu equilíbrio quando se atinge a menor energia de Gibbs em uma determinada composição, temperatura e pressão. Desde que a energia de Gibbs (em função da pressão, temperatura e composição) seja conhecida para as fases individuais, então é possível calcular o estado de equilíbrio do sistema, por um procedimento de minimização de energia. Uma vantagem importante do método CALPHAD é permitir a previsão de diagramas de equilíbrio de ordem superior via extrapolação de seus sistemas constituintes de ordem inferior, como sistemas binários e ternários.

O método CALPHAD [29], adota uma metodologia fenomenológica (conjunto de fenômenos e como se manifestam, seja através do tempo ou do espaço) para obter uma descrição termodinâmica auto consistente de um sistema multicomponente, como mostrado na Figura 2. O termo “banco de dados termodinâmico” significa que os parâmetros para as energias de Gibbs de muitos sistemas binários e ternários importantes foram montados para as faixas de composição pretendidas.

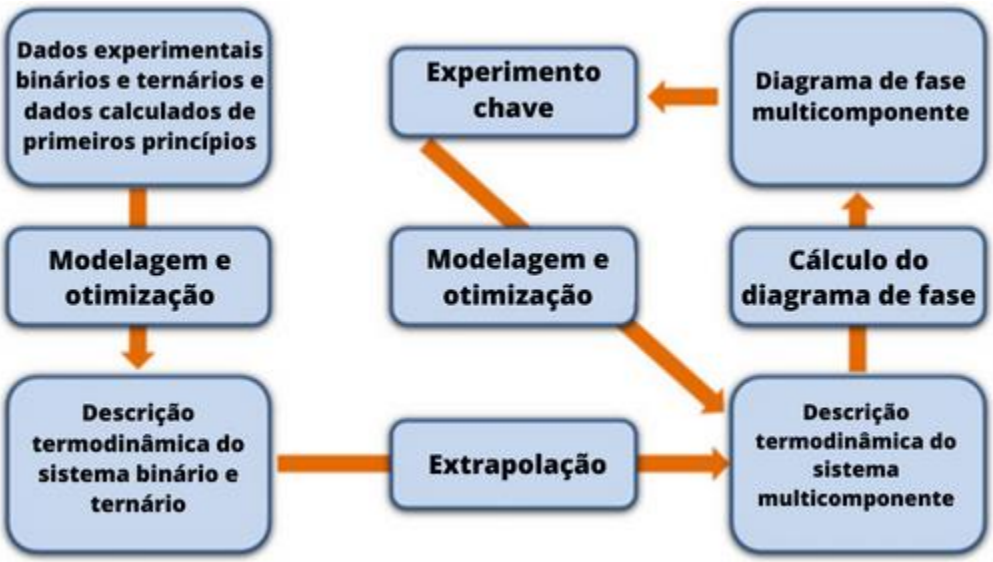


Figura 2: Esquema CALPHAD ou abordagem fenomenológica usada para obter uma descrição termodinâmica de um sistema multicomponente. Adaptado de [31].

Uma vez obtidas descrições termodinâmicas confiáveis de elementos constituintes do sistema, porém de ordem inferior, geralmente binários e ternários, o banco de dados termodinâmico de ordem superior pode ser obtido via extrapolação, como primeiro passo [32]. Um banco de dados termodinâmico, aplicável às LMMs, precisa cobrir toda a sua faixa de composição, uma vez que a composição de uma LMM está localizada no centro do espaço de composição multidimensional. Modelos termodinâmicos adequados precisam ser selecionados para descrever as energias de Gibbs de todas as fases envolvidas no sistema final da LMM, o que envolve uma série de parâmetros empíricos a serem considerados, tais como a diferença de tamanho atômico, entalpia, concentração de elétrons de valência, do inglês, Valence Electron Concentration (VEC), formalismo de energia composta, entre outros [31].

Essa série de fundamentações por trás dos modelos de simulação termodinâmica, exigem cálculos não triviais, o que normalmente pode tornar o processo computacionalmente mais complexo, resultando em ciclos relativamente longos de desenvolvimento [6]. Quando comparamos isso com novas tendências em big data, como por exemplo o aprendizado de máquina, que possui a capacidade de unificar os resultados desses métodos, com o que existe do histórico de dados gerados através de análises empíricas já feitas, e utilizamos esses inputs para realizar previsões acerca das fases das LAEs, tem-se um grande ganho de desempenho, com um processo significativamente mais rápido e menos oneroso.

2.2 APRENDIZADO DE MÁQUINA

O Aprendizado de Máquina (AM), é um vasto campo interdisciplinar que se baseia em conceitos de ciência da computação, estatística, ciência cognitiva, engenharia, teoria da

otimização e muitas outras disciplinas de matemática e ciência [33]. Existem inúmeras aplicações para o ML, mas a mineração de dados é a mais significativa entre todas [34].

O ML envolve a capacidade das máquinas de aprender, onde uma máquina é construída usando determinados algoritmos por meio dos quais ela pode tomar suas próprias decisões e fornecer o resultado ao usuário. Basicamente, é considerado o subcampo da Inteligência Artificial. Hoje, o aprendizado de máquina é usado para classificação de dados complexos e tomada de decisão [35]. Em termos simples, é o desenvolvimento de algoritmos que permitem que o sistema aprenda e tome as decisões necessárias. O ML tem fortes laços com a otimização matemática que fornece métodos teóricos para aplicação em campo, e é empregado em uma variedade de tarefas de computação onde projetar e programar algoritmos explícitos é inviável. O ML, pode ser classificado principalmente em duas grandes categorias, que incluem o aprendizado de máquina supervisionado e aprendizado de máquina não supervisionado.

O aprendizado de máquina não supervisionado é usado para tirar conclusões de conjuntos de dados que consistem em dados de entrada sem respostas rotuladas [36], ou seja, no aprendizado não supervisionado, a saída desejada não é previamente conhecida pelo usuário, mas sim definida pelo algoritmo. Por outro lado, as técnicas de aprendizado de máquina supervisionado tentam descobrir a relação entre os atributos de entrada (variáveis independentes) e um atributo de destino (variável dependente dos atributos de entrada) [37]. As técnicas supervisionadas podem ainda ser divididas em duas categorias principais: classificação e regressão. Na regressão, a variável de saída assume valores contínuos, enquanto na classificação, a variável de saída recebe rótulos de classe [38]. A fim de dar foco as técnicas abordadas nesse trabalho, exploraremos mais os conceitos ligados às técnicas supervisionadas relacionadas à classificação.

A classificação é uma abordagem de mineração de dados (aprendizado de máquina), usada para prever a associação de grupos para instâncias de dados [39]. Embora haja uma variedade de técnicas disponíveis para aprendizado de máquina, a classificação é a técnica mais amplamente utilizada [40]. Ela é categorizada como um dos problemas supremos estudados por pesquisadores dos campos de aprendizado de máquina e mineração de dados [41]. Um modelo geral de aprendizado supervisionado, acerca das técnicas de classificação, é mostrado na Figura 3.

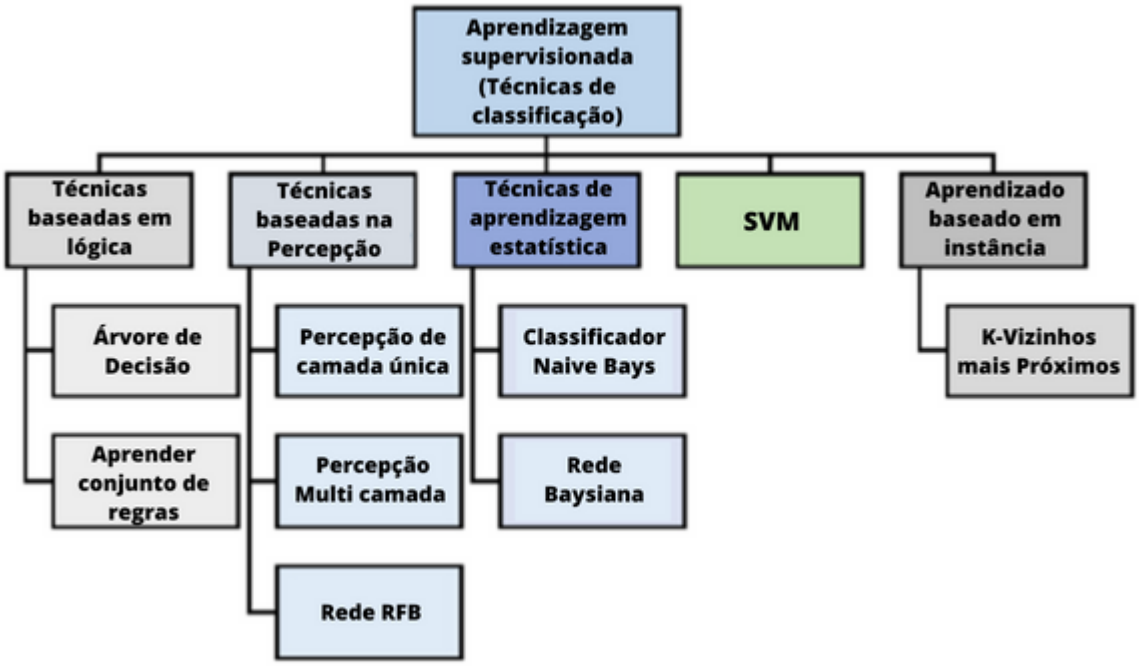


Figura 3: Diagrama esquemático ilustrando processos de aprendizagem supervisionada de técnicas de classificação. Adaptado de [42].

Nesse tipo de aprendizado, cada exemplo é um par que consiste em um objeto de entrada (basicamente um vetor, com o conjunto de características do objeto de estudo) e um valor de saída desejado (sinal de supervisão, ou rótulo). Um algoritmo de aprendizado supervisionado analisa e estuda os dados de treinamento (onde os rótulos para os objetos analisados já são conhecidos) através de uma função inferida, que pode ser usada para mapear novos exemplos. O cenário ideal, considerando uma classificação, realizada por algum dos algoritmos aplicáveis, permitirá que o mesmo determine corretamente os rótulos de classe para instâncias não vistas anteriormente. É necessário que o algoritmo de aprendizado generalize a partir dos dados fornecidos previamente para treinamento do modelo, para que, em situações não vistas, haja uma classificação "razoável" (o grau de precisão necessário a ser atingido, irá variar conforme o problema tratado). Algumas das abordagens para aprendizagem supervisionada direcionadas à classificação podem ser vistas no esquema da figura 3. Nesse trabalho foram exploradas três diferentes técnicas, que foram aplicadas ao mesmo desafio, sendo elas: Árvore de decisão , K-Vizinho mais próximo (conhecido também como, *K-nearest neighbor* - KNN), e Máquinas de Vetores de Suporte (ou, Support Vector Machine - SVM). [43]

2.2.1 Árvores de Decisão

A Árvore de Decisão, ou *Decision Tree* -DT, é uma abordagem de Aprendizado de Máquina Supervisionado para resolver problemas de classificação e regressão, dividindo dados

continuamente com base em um determinado parâmetro, ou conjunto deles. As decisões estão nas folhas e os dados são divididos através dos nós de decisão.

As árvores de decisão são um dos algoritmos mais úteis e poderosos na mineração de dados, sendo capazes de lidar com os diversos tipos de dados de entrada, como nominal, numérico e alfabético [44]. A árvore de decisão é um mecanismo transparente que facilita aos usuários o entendimento, ao seguir uma estrutura de árvore, para ver como a decisão é tomada. Na figura 4 a seguir temos um exemplo que mostra o funcionamento do algoritmo de árvore de decisão simples [45]:

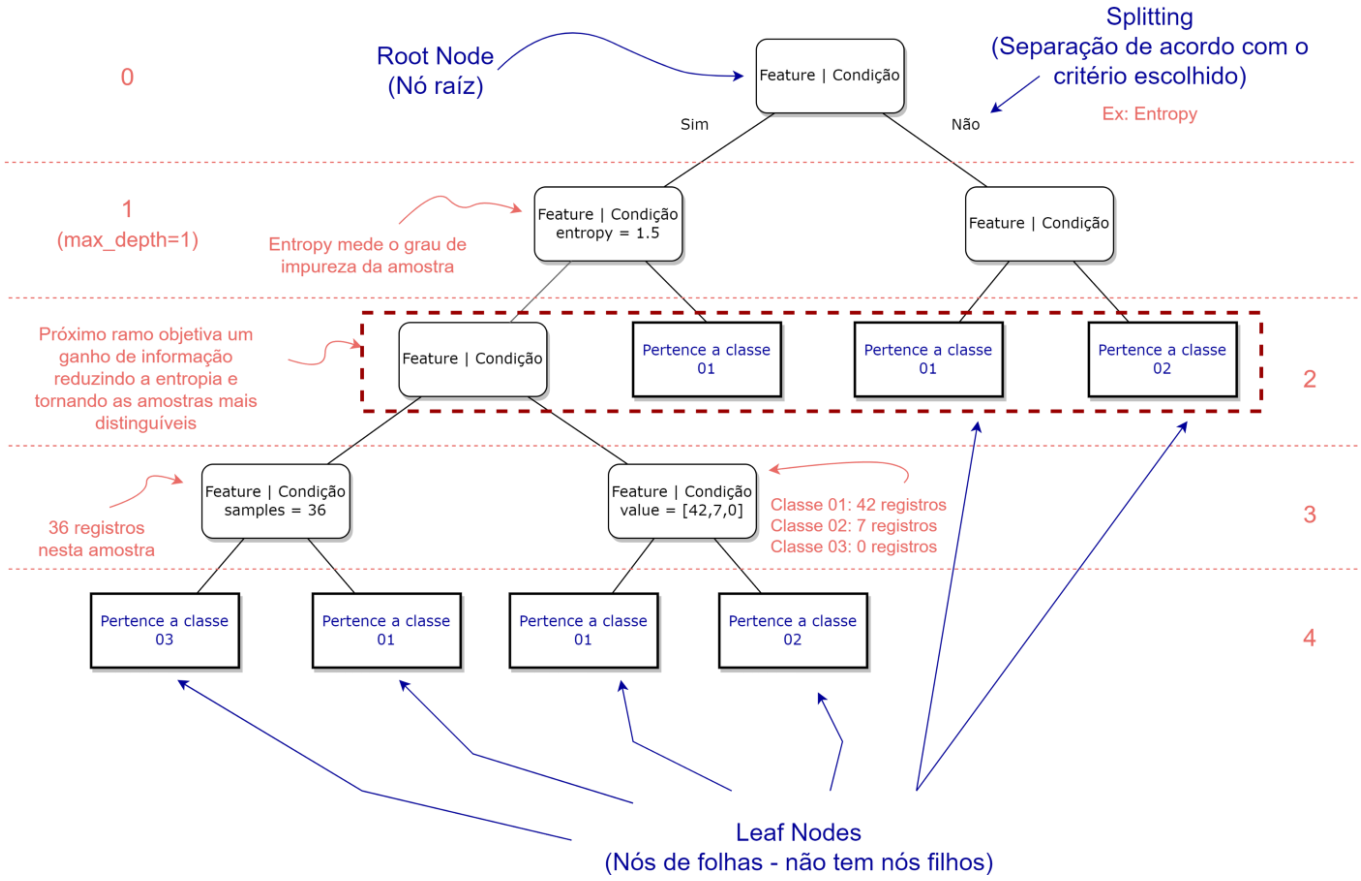


Figura 4: Estrutura de funcionamento de uma árvore de decisão. [45]

Normalmente todos os algoritmos de árvore de decisão são construídos em duas fases:

(i) Crescimento da árvore: em que o conjunto de treinamento baseado em critérios ótimos é dividido recursivamente até que a maior parte do registro pertencente à partição tenha o mesmo rótulo de classe [46]

(ii) Poda de árvore: Onde o tamanho da árvore é reduzido facilitando a compreensão [47].

Existem algumas abordagens que podem ser adotadas para a seleção dos critérios de decisão encontrados nos nós da árvore. As principais abordagens utilizadas envolver os seguintes algoritmos: o *Iterative Dichotomiser 3* - ID3 e o C4.5.

Estas abordagens estão relacionadas ao ganho de informação, para determinar a propriedade adequada dentro do conjunto de características de entrada. Assim, a partir do maior ganho de informação podemos selecionar o atributo.

O algoritmo de árvore de decisão ID3 foi introduzido em 1986 [48], e é um dos algoritmos mais amplamente utilizados na área de mineração de dados e aprendizado de máquina devido à sua eficácia e simplicidade. Alguns dos pontos fortes e fracos da árvore de decisão ID3 são apresentados em [49]. Dentre dos pontos positivos está o fácil entendimento das escolhas feitas até o resultado; já entre os pontos negativos destaca-se a pouca efetividade em de lidar com valores ausentes dentro da base e a não otimização global.

Esse tipo de algoritmo é baseado no ganho de informação para decidir o atributo de divisão. Como forma de medir esse ganho de informação é comummente utilizada da Entropia, que consiste em averiguar os dados incertos presentes no conjunto de dados, ou seja, analisa a homogeneidade das classes de acordo com um determinado atributo, à qual pode ser medido pela seguinte equação:

$$\text{Entropia(S)} = -\sum p_{(x)} log2 p_x \quad (1)$$

Onde S é o conjunto de dados para o qual a entropia é calculada, x é um conjunto de classes possíveis como rótulos para os dados em S, P(x) é a proporção/probabilidade do número de elemento ser da classe x, em relação ao número total de elementos no conjunto S. Quando a Entropia(S) = 0, então o conjunto de dados é perfeitamente classificado, ou seja, todos os elementos em S são de uma mesma classe. Em outras palavras, o quanto a incerteza em S é reduzida após a divisão de S de acordo um atributo de entrada escolhido, melhor é a separação dos dados e a realização da classificação.

O C4.5 é um algoritmo famoso para produção de árvores de decisão. É uma expansão do algoritmo ID3, que minimiza as desvantagens causadas pelo seu antecessor. Na fase de poda, C4.5 tenta eliminar os ramos desnecessários, trocando-os por nós de folha, voltando pela árvore depois de gerada [50]. Os pontos fortes do C4.5 são lidar com dados de treinamento com valores de recursos ausentes, lidar com recursos discretos e contínuos e fornecer facilidade de pré e pós remoção, reforçando assim os pontos mais frágeis da ID3 [49]. As fraquezas incluem não ser

adequado para pequenos conjuntos de dados [49] e um tempo relativamente mais alto de processamento, em comparação com outras árvores de decisão.

Os passos básicos para implementação dos algoritmos de árvore são os seguintes [44]:

1. Selecione todos os atributos dos diferentes níveis de nós da árvore de decisão;
2. Calcule o crescimento da informação para cada atributo;
3. Utilizar o ganho de informação como critério/ medida de seleção de atributos e escolher o atributo com maior ganho de informação para decidir o nó raiz da árvore de decisão.
4. Os ramos da árvore de decisão são calculados pelos diferentes valores de ganho de informação dos nós.
5. Construir os nós e ramificações da árvore de decisão recursivamente até que um determinado conjunto de dados das instâncias pertença ao mesmo grupo.

2.2.2 Máquina de Vetor de Suporte

O método de máquina de vetor de suporte, ou *Support Vector Machines (SVM),* está sendo utilizado desde 1992, quando houve a necessidade de ferramentas de classificação e regressão baseadas em algumas previsões; foi introduzido por Vapnick, Guyon e Boser em COLT-92[52]. Para separar quaisquer dados, definimos certas classes e, dependendo da complexidade dos conjuntos de dados, definimos como classificação linear ou não linear.

O SVM é considerado uma das técnicas mais proeminentes e convenientes para resolver problemas relacionados à classificação de dados [53], aprendizado e previsão [54]. Os vetores de suporte são os pontos de dados que estão mais próximos da superfície de decisão [55]. O algoritmo executa a classificação de vetores de dados por um hiperplano em um imenso espaço dimensional [56], podendo ser definido apenas como uma ferramenta de previsão em que procuramos uma linha específica ou limite de decisão, onde este é denominado hiperplano, que separa facilmente os conjuntos de dados ou classes, evitando o ajuste extra aos dados. O SVM ainda é capaz de utilizar o espaço de hipóteses de um espaço linear, em um espaço de recursos de alta dimensão. Desta maneira, também é possível de classificar os dados não lineares por meio de funções do kernel, que são as funções responsáveis por alterar a dimensionalidade dos dados para tornar viável o traçado dos vetores de decisão.

Começando pela utilização do SVM para classificação de problemas lineares, temos o surgimento de outro termo bastante conhecido dentro do SVM, que é o Classificador de Vetores de Suporte, ou *Support Vector Classifier (SVC).* Nesse ponto, salienta-se que, se o hiperplano que estamos usando para classificação estiver em condição linear, então a condição é dada como SVC. Na figura 5 podemos ver um exemplo de classificação hipotético, onde temos muitos

limites de decisão (hiperplanos) que podem ser utilizadas para separar assertivamente dois diferentes grupos (preto e vermelho):

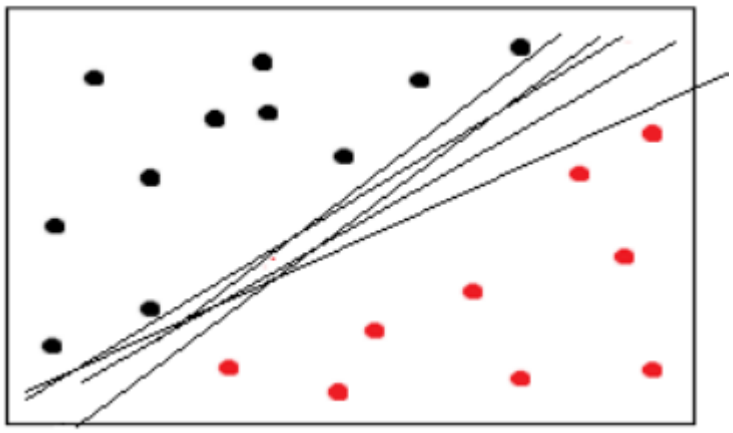


Figura 5: Representação de classificações feitas por diferentes hiperplanos [57]

Todos esses hiperplanos (linhas) representados na figura 5 são capazes de classificar adequadamente o conjunto de dados hipotéticos representados pelos pontos, uma vez que todos estão separados de acordo com as duas diferentes "classes" existentes, a preta e a vermelha, mas a questão é qual hiperplano deve ser selecionado para que a classificação seja ótima. Aqui, exigimos um hiperplano que seja justo para ambas as categorias de amostras, o que significa que, dentre todos os hiperplanos ou limites de decisão, apenas um deles deve ser selecionado. Para a seleção do hiperplano, seguimos os passos abaixo:

1. Defina uma função para gerar o hiperplano necessário, ou seja, o limite entre os diferentes conjuntos de dados.
2. O próximo passo é selecionar um hiperplano e calcular sua distância de ambos os lados dos conjuntos de dados.
    i. Se a distância calculada for máxima, é reduzido a máxima em ambos os lados em comparação com o hiperplano anterior, selecione este hiperplano como o novo limite de decisão.
    ii. Marque as amostras próximas ao hiperplano como vetores de suporte. (ajuda na seleção do limite de decisão)
3. Repita a etapa 2 até encontrar o melhor hiperplano.

Assim, notamos que para a seleção de hiperplano precisamos resolver o problema de margem máxima, que é a distância entre o limite de decisão e os vetores de suporte. a Figura 6 representa a solução para isso.

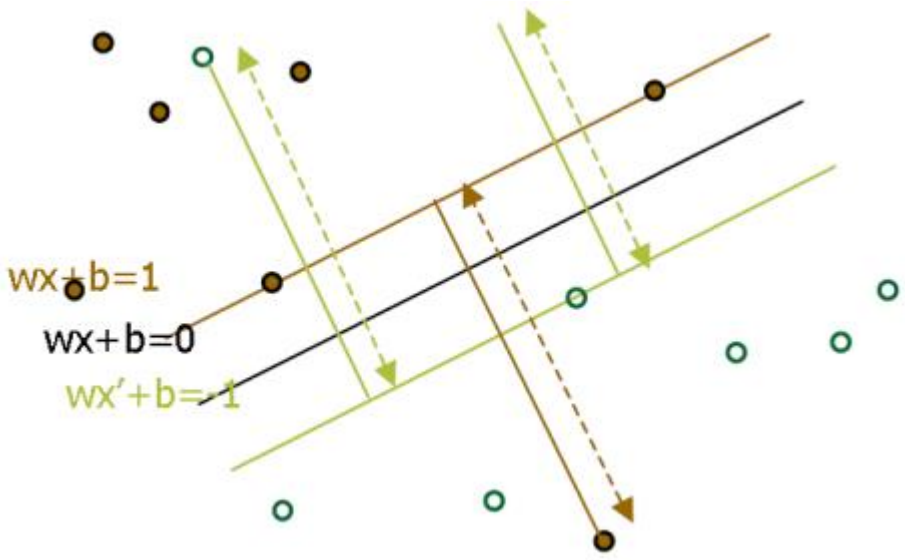


Figura 6: Traço do hiperplano ótimo e seus respectivos limites de decisão [43].

A expressão para a margem máxima é dada como:

$$\text{Margem} = \arg\frac{min_{d(x)}}{x \in D} = \arg\frac{min}{x \in D}\ \frac{|x-w+b|}{\sqrt{\sum_{i=1}^{d} w_i^2}} \quad (2)$$

Onde após resolver esse problema, a expressão para margem máxima pode ser reduzida para:

$$\text{Margem} = \frac{2}{\|w\|}$$

Os limites de decisão são margens imaginários em que estão localizados os primeiros elementos classificados de cada grupo. Pela figura 6 vemos que wx+ b=1 e wx'+b = 1 são traçados exatamente nos primeiros pontos de cada grupo de dados, limitando a margem máxima do hiperplano, formando os chamados "limites de decisão" ou margem máxima.Em alguns conjuntos de dados essa abordagem pode apresentar limitações e, assim, apresentar alguns erros. Um erro no conjunto de dados nada mais é do que o fato de que algumas das amostras podem estar presentes em outra categoria de conjunto de treinamento. A Figura 7 ilustra a representação de erros, aqui representamos duas diferentes classes no conjunto de treinamento, o vermelho e o preto.

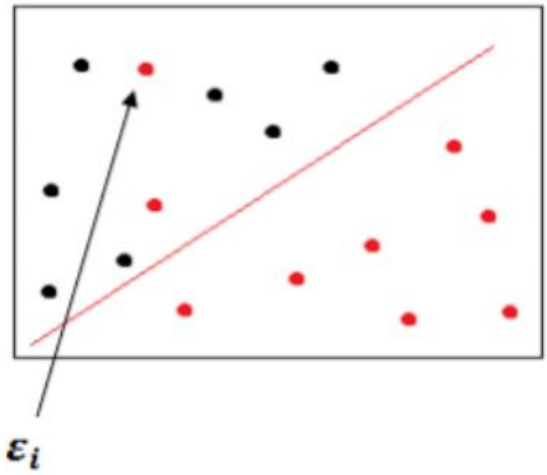


Figura 7: Representação de erro de classificação por um hiperplano [57]

Aqui podemos ver que temos um certo erro na classificação elaborada pelo hiperplano (linha vermelha), onde os conjuntos de dados do tipo vermelhos estão presentes fora do limite de decisão, ou seja, no conjunto de dados preto, portanto, isso é denominado como um erro e o limite de decisão falha aqui. Logo, não podemos desenhar um hiperplano nesses dados dispersos para separar os pontos de dados entre as classes (para a classificação) por uma linha reta.

A limitação do SVC é compensada pelo SVM de forma não linear. E essa é a diferença entre SVM e SVC. Se o hiperplano classifica o conjunto de dados linearmente, então o chamamos de algoritmo SVC e se o algoritmo separar o conjunto de dados por alguma abordagem não linear, o chamamos de SVM [58]. A figura 8 mostra a diferença de uma separação linear e não linear:

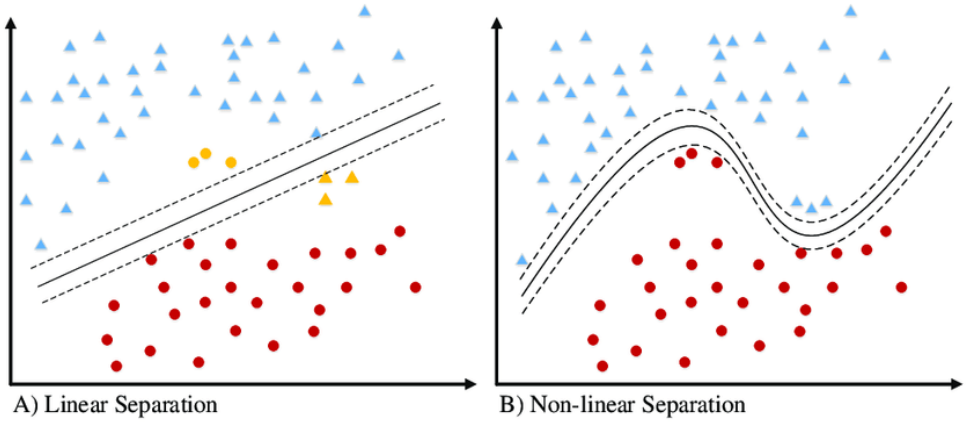


Figura 8: Comparação entre uma separação feita com hiperplanos lineares e não lineares.[58]

Para dados lineares, um hiperplano de separação pode ser usado para classificá-lo. No entanto, nem sempre é possível ter dados linearmente separáveis o tempo todo. Às vezes, um

conjunto de dados não lineares precisa ser classificados onde o hiperplano de separação não funcionará facilmente. Portanto, precisamos de uma função especial conhecida como função kernel para mapear os dados não lineares para o espaço de recursos de alta dimensão, ou seja, funções que pegam um espaço de entrada de baixa dimensão e o transformam em um espaço de dimensão superior, convertendo um problema não separável em um problema separável [43,58]. A figura 9 ilustra esse tipo de transformação:

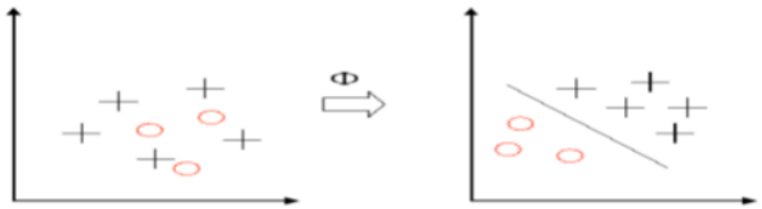


Figura 9: Ilustração do resultado de uso de Kernels. [57]

Há várias funções do kernel que realmente nos ajudam a classificar os dados não lineares, e a ideia básica por trás disso são as operações a serem executadas no espaço de entrada em vez do espaço de recursos de alta dimensão.

Um dos kernels mais importante é a função de base radial gaussiana, ou *Gaussian Radial Basis Function* (*RBF*), essa função pode ser expressa da seguinte maneira [59]:

$$K\,(x, x') = \exp\left(-\frac{\|x-x'\|^2}{2\sigma^2}\right) \quad (3)$$

Onde σ é a variância, ou desvio padrão, enquanto *x e x'* são pontos vetoriais em qualquer espaço dimensional fixo. O motivo principal do kernel é fazer cálculos em qualquer espaço d-dimensional onde d > 1, para que possamos obter uma equação quadrática, cúbica ou de qualquer grau polinomial para obter a linha de classificação/regressão. Como o kernel de base radial usa expoente e, a expansão de e^x fornece uma equação polinomial de potência infinita, então, usando esse kernel, tornamos nossa linha de regressão/classificação infinitamente poderosa também.

Outros kernels importantes e bastante comuns são o polinomial SVM, e o núcleo Sigmoide. O kernel adequado, pode variar conforme o problema abordado, e a otimização do modelo pode refinar quais são os parâmetros mais adequados ao problema.

Entre as principais vantagens do SVM, há uma maior eficácia para classificações de alta dimensão, bom funcionamento para dados não estruturados e semiestruturados, como texto, imagens e árvores. Além disso, o SVM lida bem com dados não lineares através do uso de

kernels que transformam o problema para SVC. Entre as desvantagens do SVM, destaca-se um desempenho não ótimo para conjunto de dados extensos, sensibilidade a outliers, hiperparâmetros de difícil ajuste como o custo (C) e gama, e possui um entendimento e compreensão mais complexos quando comparado a modelos como árvores de decisão.

2.2.3 K-Vizinhos Mais Próximos

O classificador k-vizinho mais próximo, ou *k-nearest neighbor (KNN),* usado para classificar observações não rotuladas, atribuindo-as à classe dos exemplos rotulados mais semelhantes, onde k define quantos vizinhos mais próximos devem ser considerados. As características das observações são coletadas para o conjunto de dados de treinamento e teste.

Dessa forma, para que um registro de dados t seja classificado, seus k vizinhos mais próximos são considerados, e isso forma uma vizinhança de t. A comparação de determinada amostra com a maioria "k" dos registros de dados na vizinhança é geralmente usada para decidir a classificação de t, com ou sem consideração de ponderação baseada na distância. No entanto, para aplicar KNN, precisamos escolher um valor apropriado para k (o qual podemos considerá-lo como um hiperparâmetros), na qual o sucesso da classificação depende muito desse valor. Em certo sentido, o método KNN é enviesado por k. Existem muitas maneiras de escolher o valor de k, mas um simples é executar o algoritmo muitas vezes com diferentes k e escolher aquele com o melhor desempenho [60].

Por exemplo, podemos pegar a figura 10 abaixo, onde tentamos exibir a distinção de frutas, vegetais e grãos baseado em duas características: doçura e crocância. Para facilitar o entendimento estão sendo usadas apenas duas dimensões, mas na realidade pode haver qualquer número de características para determinar uma classe, podendo assim estender esse exemplo para diversos problemas com uma maior complexidade dimensional.

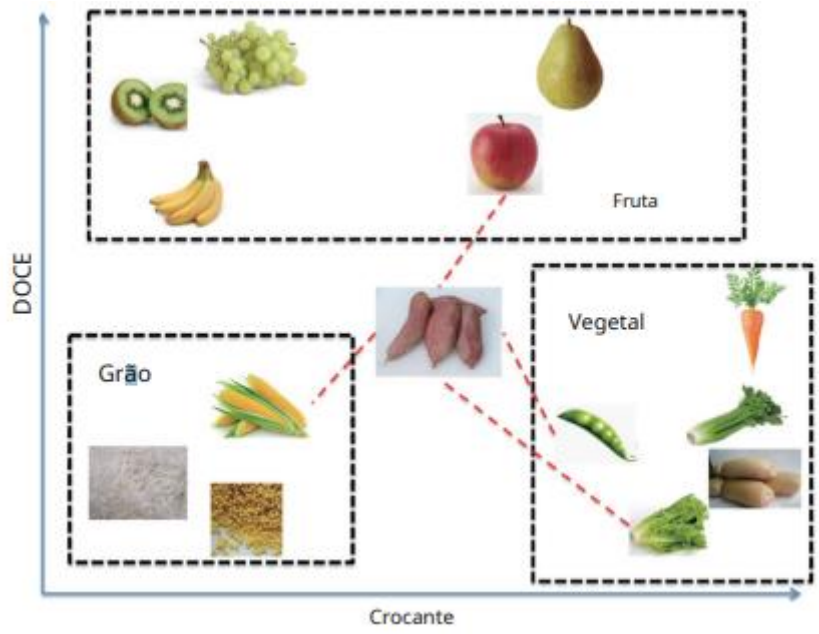


Figura 10: Ilustração de como funciona o algoritmo dos k vizinhos mais próximos. [61]

Em geral, as frutas são mais doces que os vegetais. Já os grãos, não são crocantes nem doces. Neste exemplo escolhemos quatro alimentos mais próximos, ou seja, k=4, assim são considerados a maçã, vagem, alface e milho. Como os vegetais ganha mais votos, a batata-doce é designada para a classe dos vegetais. Você pode ver que o conceito-chave do KNN é de fácil entendimento.

Existem dois conceitos importantes acima, um deles já bastante destacado, que é o parâmetro k que decide quantos vizinhos serão escolhidos para o algoritmo kNN. A escolha apropriada de k tem sempre um impacto significativo no desempenho de diagnóstico do algoritmo kNN. Um k grande pode reduzir o impacto da variância causada pelo erro aleatório, mas corre o risco de ignorar um padrão pequeno, mas importante. A chave para escolher um valor k apropriado é encontrar um equilíbrio entre *overfitting* e *underfitting* [62]. Alguns autores sugerem definir k igual à raiz quadrada do número de observações no conjunto de dados de treinamento [63].

O outro conceito é o método para calcular a distância entre a batata doce e outros alimentos. Por padrão, a função knn () emprega distância euclidiana que pode ser calculada com a seguinte equação (64,65).

$$D(p,q) = \sqrt{(p_1 - q_1)^2 - (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2}$$

Onde p e q são objetos a serem comparados com n características. Existem também outros métodos de distância que podem ser utilizados, como a distância de Manhattan, semelhança de cosseno, Hamming e outras.[66]

As vantagens do algoritmo KNN incluem ser uma técnica simples, que é facilmente implementada, a construção do modelo é barata, e é um esquema de classificação extremamente flexível e adequado para classes multimodais. Além disso, também é um modelo de fácil entendimento. Como desvantagens pode ser destacado uma complexidade computacional relativamente maior comparado a alguns outros algoritmos de classificação, características ruidosas/irrelevantes resultarão na degradação da precisão.

2.3 VALIDAÇÃO E OTIMIZAÇÃO DE MODELOS

Classificação e seleção de características são operações fundamentais e complementares em mineração de dados e aprendizado de máquina. A qualidade dos recursos/características selecionadas afeta a qualidade do modelo de classificação e seu desempenho nos dados de validação (dados não usados para treinar e testar o modelo). Especificamente, incorporar muitos recursos irrelevantes no modelo de treinamento pode levar a previsões que não generalizam bem para dados de validação porque a compensação de variação de viés é inclinada para alta variância (overfitting). Por outro lado, excluir características importantes do modelo de treinamento pode levar a previsões com baixa precisão porque a compensação de viés-variância tende a ter um viés alto, ou seja, um subajuste.

Por isso, existem várias maneiras de usar a seleção de recursos na classificação, para lidar com a compensação de variação de viés. Uma das mais usadas, e que será discutida aqui, é a utilização da validação cruzada, ou *cross-validation (CV)*. Outra coisa importante para a otimização adequada dos modelos com as quais se trabalha, é o ajuste do modelo, e isso envolve a seleção dos chamados hiperparâmetros. A seleção da melhor configuração de hiperparâmetros para modelos de aprendizado de máquina tem um impacto direto no desempenho deles. Muitas vezes, tal seleção requer conhecimento profundo de algoritmos de aprendizado de máquina e técnicas apropriadas de otimização de hiperparâmetros.

2.3.1 Hiperparâmetros e Validação Cruzada

Em geral, construir um modelo de aprendizado de máquina eficaz é um processo complexo e demorado que envolve determinar o algoritmo apropriado e obter uma arquitetura de modelo ideal ajustando seus hiperparâmetros [67].

Existem dois tipos de parâmetros em modelos de aprendizado de máquina: um que pode ser inicializado e atualizado por meio do processo de aprendizado de dados (por exemplo, os pesos dos neurônios em redes neurais), denominados parâmetros do modelo; enquanto o outro, denominado hiperparâmetros, que não podem ser estimados diretamente a partir do aprendizado de dados e deve ser definido antes do treinamento de um modelo de ML, porque define a

arquitetura do modelo [68]. Hiperparâmetros são aqueles usados para configurar um modelo de ML (por exemplo, o parâmetro de Kernel em um modelo de máquina de vetores de suporte, ou o número de vizinhos k, no modelo kNN).

Para construir um modelo de ML ideal, uma gama de possibilidades deve ser explorada. O processo de projetar a arquitetura do modelo ideal, com uma configuração de hiperparâmetros ideal, é chamado de ajuste de hiperparâmetros. O ajuste de hiperparâmetros é considerado um componente chave da construção de um modelo eficaz [69]. O processo de ajuste de hiperparâmetros é diferente entre os diferentes algoritmos de ML, devido aos seus diferentes componentes de modelagem, incluindo tipos categóricos, discretos e contínuos [70].

É crucial selecionar uma técnica de otimização apropriada para detectar hiperparâmetros ideais. Alguns dos métodos mais comuns para otimização de hiperparâmetros são baseados em gradiente descendente, um tipo comum de algoritmo de otimização que pode ser usado para ajustar hiperparâmetros contínuos calculando seus gradientes [71].

Em comparação com os métodos tradicionais de otimização, como gradiente descendente, muitas outras técnicas de otimização são também adequadas para problemas de otimização de hiperparâmetros, incluindo abordagens teóricas de decisão, modelos de otimização bayesiana, técnicas de otimização multifidelidade e algoritmos metaheurísticos [70]. Além de detectar hiperparâmetros contínuos, muitos desses algoritmos também têm a capacidade de identificar efetivamente hiperparâmetros discretos, categóricos e condicionais.

Desses métodos, vale destacar o que utiliza a teoria da decisão, já que esse foi o principal meio utilizado neste trabalho para uma otimização dos hiperparâmetros. Nela, o conceito gira em torno de definir um espaço de busca de hiperparâmetros e, em seguida, detectar as combinações de hiperparâmetros dentro desse espaço de busca, selecionando finalmente a combinação de hiperparâmetros de melhor desempenho.

A busca em grade, ou *Grid search* (GS) [72] é uma abordagem teórica de decisão que envolve a procura exaustiva de um domínio fixo de valores de hiperparâmetros. A busca aleatória, ou *Random search* (RS) [73] é outro método teórico de decisão que, seleciona aleatoriamente combinações de hiperparâmetros no espaço de busca, garantindo muitas vezes, a experimentação de uma maior macro exploração, principalmente quando se possui um dado tempo de execução e recursos limitados. Em GS e RS, cada configuração de hiperparâmetro é tratada de forma independente. A vantagem do RS é que, por explorar o espaço de hiperparâmetros de forma aleatória, seu processamento normalmente ocorre de forma a

necessitar de uma menor capacidade computacional, explorando muitas vezes maiores espaços em menos tempo, porém não de tão rica como com GR, perdendo um pouco da precisão fina.

Essas varreduras de hiperparâmetros são normalmente executadas em conjunto com a validação cruzada e, desse modo, para cada conjunto de hiperparâmetros selecionados já fica possível verificar seu desempenho. Por isso, a validação cruzada é importante para o ajuste de hiperparâmetros, pois é através dela que se tem a certificação de um bom desempenho.

Dessa forma, o CV é outra operação fundamental no aprendizado de máquina, onde ela divide os dados em conjuntos de treinamento e teste para estimar a precisão da generalização de um classificador para um determinado conjunto de dados [74,75]. Algumas das formas como o CV foi implementado incluem o CV deixar um de fora, ou leave-one-out [76], CV k-fold [77] e CV aninhado, ou nested CV (nCV).

O nCV é uma maneira eficaz de incorporar a seleção de recursos e o ajuste de parâmetros de aprendizado de máquina para treinar um modelo de previsão ideal. Na abordagem nCV padrão, os dados são divididos em k partes externas e as partes internas são criadas em cada conjunto de treinamento externo para selecionar recursos, ajustar parâmetros e treinar modelos.

Na figura 11 podemos ver um esquema que ilustra o funcionamento do nCV padrão, onde o conjunto de treinamento é dividido em n partes internas para uma série de treinamentos, e testes com diferentes divisões do conjunto de dados. Normalmente, o nCV escolhe o modelo por meio da parta externa que combina os parâmetros que minimiza o erro de teste interno. Para restringir o overfitting, escolhe-se o modelo onde a parte externa e os hiperparâmetros apresentam a menor diferença entre treinamento e precisão de teste. O modelo e os hiperparâmetros com menor overfitting nas dobras internas são escolhidos como o modelo de loop externo de treinamento, e testados na dobra de teste do loop externo.

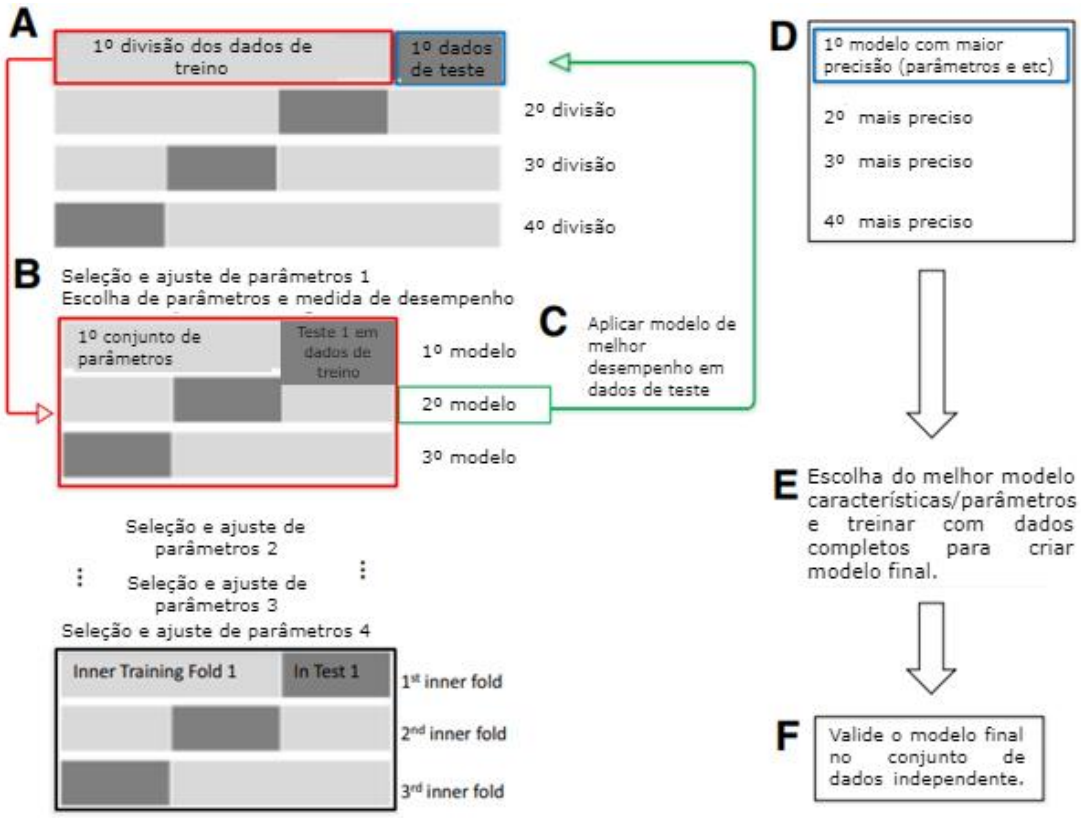


Figura 11 - Validação cruzada aninhada padrão (nCV) - Adaptado de [78].

(A) Divida os dados em partes externas, em pares de dados de treinamento e teste (quatro dobras externas nesta ilustração). Em seguida, para cada dobra externa de treinamento é realizado o treinamento com diferentes conjuntos de hiperparâmetros {na ilustração começando com a dobra externa de treinamento 1 [caixa vermelha (A)]}. (B) Divida a dobra de treinamento externa em dobras internas para seleção de recursos e possível ajuste de hiperparâmetros por pesquisa de grade GS ou pesquisa aleatória RS. (C) Use o melhor modelo de treinamento interno, incluindo recursos e parâmetros (segundo modelo interno, caixa verde, foi aquele com melhor desempenho interno para ilustração) com base no overfitting mínimo (diferença entre precisão de treinamento e teste) nas dobras internas para então testar na dobra de teste externa (seta verde para a caixa azul, Dobra de teste 1). (D) Salve o melhor modelo para esta dobra externa, incluindo os recursos e as precisões de teste. Repita (B)–(D) para as dobras externas restantes. (E) Escolha o melhor modelo externo com suas características baseadas no overfitting mínimo, trazendo o melhor desempenho ao modelo. Treine com os dados completos para criar o modelo final. (F) Valide o modelo final em dados independentes

Normalmente, o nCV escolhe o modelo de dobra externa que minimiza o erro de teste interno, mas restringindo o overfitting escolhendo o modelo de dobra externa e os hiperparâmetros com a menor diferença entre treinamento e precisão de teste nas dobras

internas (menor overfitting). Por isso é essencial a execução de métodos que avaliem o desempenho dos diferentes conjuntos de hiperparâmetros, para que haja uma ótima otimização dos parâmetros de um modelo de aprendizado de máquina.

2.4 APRENDIZADO DE MÁQUINA NA CIÊNCIA DOS MATERIAIS

Métodos tradicionais para descoberta de novos materiais, como o método empírico padrão de tentativa e erro, e métodos baseados na teoria funcional da densidade (DFT) muitas vezes tem dificuldade em acompanhar o desenvolvimento da ciência dos materiais nos dias de hoje, devido aos seus longos ciclos de desenvolvimento, baixa eficiência e altos custos relacionados. [6,80]

Graças principalmente ao baixo custo computacional comparado a outros métodos, o ML se mostra como alternativa, sendo um poderoso meio para o processamento de dados com alto desempenho de previsão, e está sendo amplamente utilizado na detecção de materiais, análise de materiais, determinação de diagramas de equilíbrio, previsões de propriedades e design de materiais. [9,10,79,81]

Dentro da área de ciência dos materiais, a utilização de métodos de ML vem se mostrando cada vez mais atrativos para aplicação em diversas áreas, inclusive para previsão de fases em ligas de alta entropia, como mostrado em alguns exemplos neste tópico.

2.4.1 Aprendizado de Máquina Para Prever Formações de Fase de Ligas de Alta Entropia

Até agora, uma fração extremamente pequena do espaço de composição das LAEs foi estudada, principalmente por meio de abordagens de tentativa e erro e apenas algumas centenas de composições de ligas metálicas multicomponentes foram descobertas. Com mais de 80 elementos metálicos na tabela periódica, é um grande desafio reconhecer as fases das LAEs e encontrar todas as composições possíveis de forma eficiente.[9]

Originalmente, a alta entropia configuracional (S_C) foi pensada como um critério de formação de soluções sólidas monofásicas. Posteriormente, verificou-se que as LAEs não podem ser identificados apenas por SC e muitos outros parâmetros e critérios teóricos foram propostos [82]. Por meio da evolução na pesquisa cientifica nesse campo da ciência e engenharia de materiais, foi mostrado que características como entalpia de mistura (ΔH) e concentração de elétrons de valência (VEC) são fatores que podem ser usados para diferenciar as fases cúbica de corpo centrado (CCC) e cúbica de face centrada (CFC) [83,84].

Graças ao ML conter uma ampla gama de algoritmos orientados a dados, usados para fazer inferências e classificações a partir de dados observados anteriormente e predeterminados, seus

algoritmos podem melhorar iterativamente seu desempenho com cada nova amostra de dados e descobrir insights ocultos de dados complexos heterogêneos e de alta dimensão sem serem explicitamente programados [11,85,86]. Assim, com dados adequados de experimentos anteriores, modelos de ML podem ser construídos para descobrir novas tendências, prever características de materiais, como candidatos de liga promissores, microestruturas, propriedades físicas e químicas.

Isso possibilitou a construção desse trabalho, onde a partir de dados originados de uma base de dados, construída por meio de cálculos termodinâmicos (utilizando CALPHAD), construímos modelos diferentes de ML para fazer previsões acerca das combinações provenientes dos elementos considerados para constituição de ligas multicomponentes.

3. MATERIAIS E MÉTODOS

Como já mencionado, um dos principais objetivos desse trabalho foi a construção de algoritmos de aprendizado de máquina capazes de realizar a classificação de forma preditiva de fases para ligas multicomponentes.

Para isso, definiu-se quais as fases de interesse para serem detectadas e previstas pelos modelos, onde foram estabelecidas 3 principais fases., São elas: a fase cúbica de corpo centrado (CCC), cúbica de fase centrada (CFC) e Sigma. Ligas que apresentaram fases destintas (inclusive combinações entre as fases de interesse) são agrupadas em uma única classe, que foi definida como “outros”.

Dessa forma, para construção dos algoritmos de classificação, foram escolhidos 3 métodos de aprendizado de máquina supervisionados, onde foi feita a modelagem e otimização dos modelos, para a partir das experiencias passadas aos algoritmos como treino, houvesse a adaptação deles para aplicação em casos desconhecidos.

Para o treinamento dos modelos, foi necessária a construção de uma base de dados, que foi gerada por meio da unificação de diversos cálculos de ponto de diagrama de equilíbrio em software CALPHAD (cerca de 1000 pontos, onde cada um trata-se de uma variação de composição entre o conjunto de elementos trabalhados), considerando temperatura constante de 1000ºC, onde variou-se a composição dos elementos selecionados que formam as ligas multicomponentes, sendo eles o Níquel (Ni), Manganês (Mn), Ferro (Fe), Cromo (Cr) e Cobalto (Co). Além do mais, foi adotado um determinado passo, ou *Step*, que define a variação percentual entre uma composição e outra. A variação definida para a base de treino foi de passo 10, ou seja, a cada novo diagrama de ponto tinha-se a variação de 10% de algum dos 5 elementos, onde a soma de todos sempre equivalesse aos 100% da liga. Uma base para teste

também foi criada, com cerca de 495 ligas para teste, onde estas são inicialmente desconhecidas pelos modelos de aprendizado de máquina, destinadas apenas para o teste dos mesmos. Nessa base, o passo escolhido foi de 12,5%.

Para o cálculo desses diagramas de equilíbrio pontuais, contendo já as fases e outras informações como entalpia, entropia etc., foi utilizado o software para de design de materiais Pandat™ 2022 64 bits, onde para as simulações termodinâmicas utiliza da metodologia o CALPHAD, e para este trabalho, o software atuou com a base panHEA2022_TH+MB, que contém as interações binárias e terciárias entre os elementos selecionados bem definidas, necessário para aplicação do CALPHAD. Para utilização do software foi necessário o acesso a uma máquina específica, com o devido licenciamento para uso do software e realização dos cálculos, que é utilizada por um conjunto de estudantes e pesquisadores da Universidade Federal de São Carlos (UFSCar). Como meio facilitador para acesso a máquina, utilizou-se de outro software, o TeamViwer, que possibilita o acesso à computadores e dispositivos de forma remota, viabilizando assim, a utilização a distância do equipamento necessário para utilização do Pandat™.

Uma vez calculados os diagramas de ponto e, unificados em um único arquivo (de formato .xlsm), foi necessário realizar um tratamento inicial a esses dados, como retirar linhas em branco e acrescentar informações relevantes, como determinadas características para classificação. informação implementada às bases iniciais extraídas do Pandat foi a concentração dos elétrons de valência (VEC), e assim, após a inclusão dessa, foram consideradas bases de dados aptas para o processo. Esse tratamento inicial dos dados foi feito utilizando o Microsoft Excel (Versão 2211, 64 bits).

Uma vez criadas as bases de dados, o próximo passo consistiu na construção e otimização dos modelos de aprendizado de máquina, que foram constituídos utilizando a linguagem de programação Python, em sua versão 3.10.4. O principal ambiente de desenvolvimento utilizado foi o Jupyter Notebook, um ambiente computacional web para a internet, rica para criação de documentos. Alguns passos também foram elaborados utilizando o Visual Studio Code. Neles, além do Python como linguagem de programação, é importante destacar algumas bibliotecas que foram importantes para a confecção dos 3 modelos de machine learning: a Scikit-Learn, versão 1.2 (uma biblioteca de aprendizado de máquina de código aberto para a linguagem de programação Python, a qual vários algoritmos de classificação, regressão e agrupamento incluindo máquinas de vetores de suporte, florestas aleatórias, gradient boosting, k-means e DBSCAN, e é projetada para interagir com as bibliotecas Python numéricas e científicas NumPy e SciPy [87]) e pandas, versão 1.4.3 (uma biblioteca de software criada para a

linguagem Python para manipulação e análise de dados. Em particular, oferece estruturas e operações para manipular tabelas numéricas e séries temporais [88]).

Alguns outros recursos de bibliotecas foram utilizados dentro do código, os quais podem ser vistos na íntegra dentro do código, armazenado no GitHub, uma plataforma de hospedagem de código-fonte e arquivos com controle de versão usando o Git. Ele permite que programadores, utilitários ou qualquer usuário cadastrado na plataforma contribuam em projetos privados e/ou Open Source de qualquer lugar do mundo. Essa plataforma foi utilizada como repositório, a fim de manter os arquivos desse trabalho salvos para consultas. Em alguns momentos ao longo do texto pode haver trechos ou imagens tiradas do algoritmo as quais poderão ser vistas também dentro do GitHub, caso necessário. O endereço para acesso aos arquivos desse trabalho é " https://github.com/kkayquek/Classific-o-preditiva-de-fases-para-ligas-de-alta-entropia-Ni-Mn-Fe-Cr-e-Co-.git", e poderá ser encontrado também no APÊNDICE A deste trabalho.

4. RESULTADOS E DISCUSSÃO

A partir dos diferentes algoritmos escolhidos para aplicação no desafio de prever as fases para ligas multicomponentes, compostas por variações dos elementos Ni, Mn, Fe, Cr e Co, a temperatura constante de 1000ºC, obteve-se diferentes resultados, onde todos podem ser considerados satisfatórios, dado o percentual de acerto na faixa de acurácia de 88,48% até 96,36%.

O conjunto de características/entradas das ligas, utilizadas pelos modelos para realizar a previsão e classificação das fases, podem ser divididas em dois principais grupos. O primeiro levou em consideração apenas o percentual atômico de cada elemento que compõe o material, já o segundo grupo de entradas, considerou além dos percentuais atômicos de cada elemento, a concentração dos elétrons de valência (VEC).

A otimização dos modelos se deu através da execução da validação cruzada, em conjunto com a análise dos melhores hiperparâmetros de cada algoritmo. Para a exploração dos hiperparâmetros, utilizou-se da varredura aleatória (utilizando funções da biblioteca Scikit-Learn), que possibilita procurar e combinar um espaço de parâmetros grande sem necessitar de um alto poder de processamento. Para cada busca aleatória realizada foi definida uma procura por 100 diferentes combinações de hiperparâmetros, totalizando 600 modelos diferentes calculados (cada tipo de algoritmo totaliza 200 variações, onde 100 são referente a busca da melhor otimização para o grupo que possui como entrada apenas os percentuais atômicos da liga, e mais 100 para o grupo que considera também o VEC).

Partindo do algoritmo de árvore de decisão, tem-se que este obteve uma menor acurácia geral, quando comparado com os demais modelos de aprendizado de máquina utilizados. Em sua otimização estavam envolvidos parâmetros como a profundidade de árvore (*max_depth*), o critério utilizado como medida para o ganho de informação (*criterion*), o mínimo de amostras necessárias para realizar uma nova tomada de decisão, ou seja, o mínimo de amostras em um nó para dividi-lo (min_samples_split) e por fim o mínimo de amostras em um nó final, ou folha que classifica as amostras da base (min_samples_leaf). Abaixo, na figura 12, temos o espaço de parâmetros que foi utilizado para o algoritmo de árvore de decisão para ambos os grupos de entradas trabalhados.

A

```
espaco_de_parametros= {
    'criterion': ["gini", "entropy"],
    'max_depth': randint(3,25),
    'min_samples_split': randint(10,200),
    'min_samples_leaf': randint (10,200)
}
```

B

```
espaco_de_parametros2= {
    'criterion': ["gini", "entropy"],
    'max_depth': randint(3,25),
    'min_samples_split': randint(10,300),
    'min_samples_leaf': randint (10,300)
}
```

Figura 12: Espaço de hiperparâmetros varrido para otimização do melhor modelo de árvore de decisão. A) Espaço de hiperparâmetros para o primeiro grupo entradas (apenas percentuais atômicos). B) Espaço de hiperparâmetros para o segundo grupo de entradas (inclusão de VEC).

Como resultado da busca, para cada grupo de entradas, foi obtido um modelo mais otimizado, onde estes foram treinados com toda a base de dados de passo 10 (base onde as ligas apresentam uma variação percentual de cada elemento de 10 em 10%, totalizando 1000 composições diferentes). A base completa pode ser vista no arquivo base_10_steps.xlsm contida no repositório do GitHub, apresentado no apêndice A. Após treinados, esses algoritmos foram aplicados em uma nova base, onde o passo configurado foi de 12,5% (a qual também pode ser vista no GitHub), nesta havia cerca de 495 ligas desconhecidas pelo algoritmo.

Como resultado, para o modelo direcionado ao grupo onde as entradas consistiam apenas nos percentuais atômicos de cada elemento na liga, obteve-se uma acurácia de 88,48%, ou seja, das 495 ligas com determinadas fases desconhecidas pelo algoritmo, ele realizou corretamente a previsão de 438 fases presentes nas ligas (isso com base nos resultados de referência encontrados pelo Pandat). Para o algoritmo onde existe a consideração de VEC, a classificação feita atingiu uma acurácia de 91,72%, ou seja, a previsão correta de 454 fases das ligas. Assim, nota-se um ganho considerável em relação ao número total de previsões feitas.

Uma forma de apurar as previsões feitas por classe é visualizar os resultados na chamada “Matriz de Confusão”, ela exibe a distribuição dos registros em termos de suas classes reais e previstas. Como estamos trabalhando com 4 diferentes classes, temos uma matriz de 4x4, onde

na diagonal principal temos o que foi classificado corretamente pelo algoritmo, nos espaços fora dessa diagonal temos, a previsão vs a classificação real.

As classes trabalhadas são CFC (armazenadas em 0), CCC (armazenadas em 1), Sigma (armazenados em 2) e outros (armazenados em 3). Abaixo temos a Tabela 1 que contém a disposição das ligas da base de teste (com passo de 12,5%) de acordo suas fases obtidas pela simulação termodinâmica feita utilizando o software Pandat, utilizado como referência "real" para comparar com os resultados de cada algoritmo.

	CFC (0)	CCC (1)	Sigma (2)	Outros (3)
	315	36	14	130
Total	**495**			

Tabela 1: Agrupamento das ligas de acordo suas fases "reais", conforme calculado no Pandat.

Uma vez conhecido a distribuição "real" das ligas conforme suas fases, é mais fácil visualizar as matrizes de confusão antes descritas. Na figura 13, temos as matrizes de confusão para ambos os grupos de entrada do algoritmo do tipo árvore de decisão.

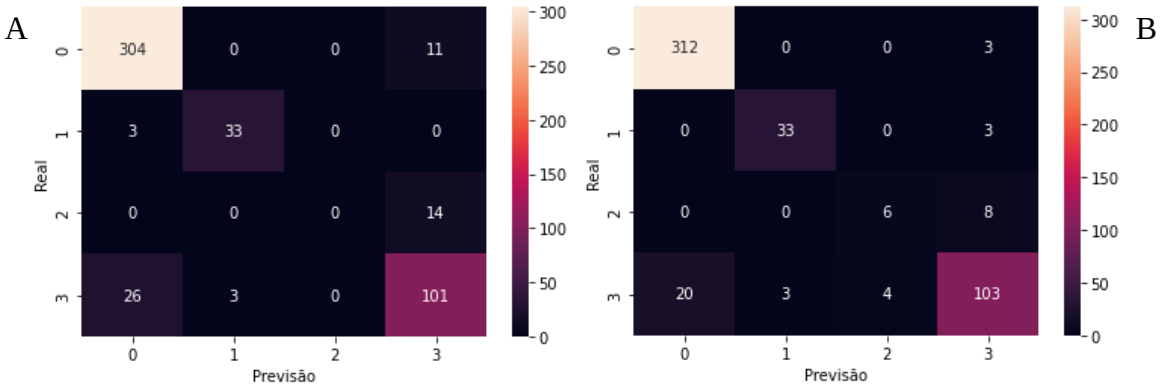


Figura 13: Matrizes de confusão do algoritmo de árvore de confusão. A) Primeiro grupo entradas (apenas percentuais atômicos). B) Segundo grupo de entradas (inclusão de VEC).

Como pode ser visto pela figura 13, ao adicionarmos o VEC como uma característica de entrada para determinar as fases da liga, há uma melhora na previsão das fases CFC, Sigma e uma ligeira melhora na previsão da classe "outros", porém não houve alteração no resultado da classe CCC, dessa forma a depender das fases de interesse a serem previstas pelo usuário, um modelo pode ser preferível em relação ao outro.

Para exemplificar a interpretação dessas matrizes, podemos olhar para primeira linha da matriz A, nela há 304 amostras alocadas na posição (0,0), ou seja, o modelo previu de forma correta 304 ligas que contém como fase apenas a CFC, porém ao compararmos com o "real", retirado

dos diagramas de ponto calculados pelo Pandat e mostrado na tabela 1, vemos que o total de ligas CFC são 315, ou seja, existem 11 ligas de fase CFC classificadas de forma incorreta pelo algoritmo, e olhando novamente para a matriz vemos na posição (0,3) as 11 ligas CFC, que o modelo classificou como sendo da classe 3, ou seja, classificadas como "outros". De forma similar podemos fazer essa análise para as demais linhas da matriz, onde as fases classificadas corretamente sempre se encontram na diagonal principal, e as classificações equivocadas ficam nas outras posições.

Além das matrizes de confusão, outra coisa que vale a pena ser observada são os modelos em si, ou seja, o conjunto de hiperparâmetros do algoritmo de árvore de decisão que atingiram esses resultados. Na figura 14 é mostrado o melhor modelo encontrado tanto para o primeiro grupo de entradas, como para o segundo.

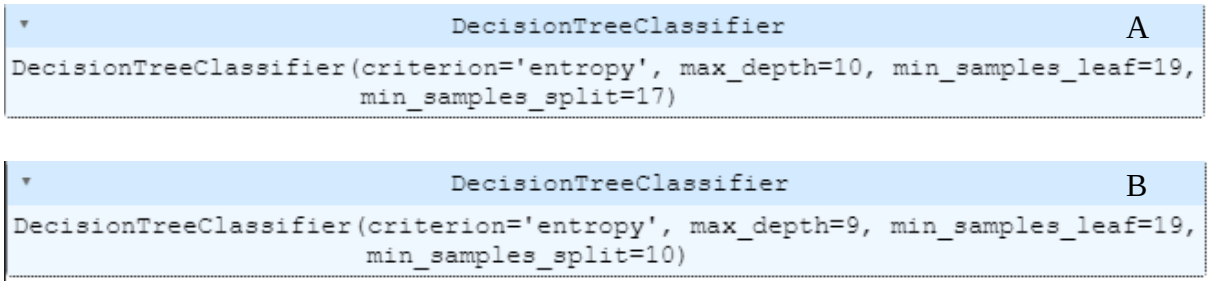
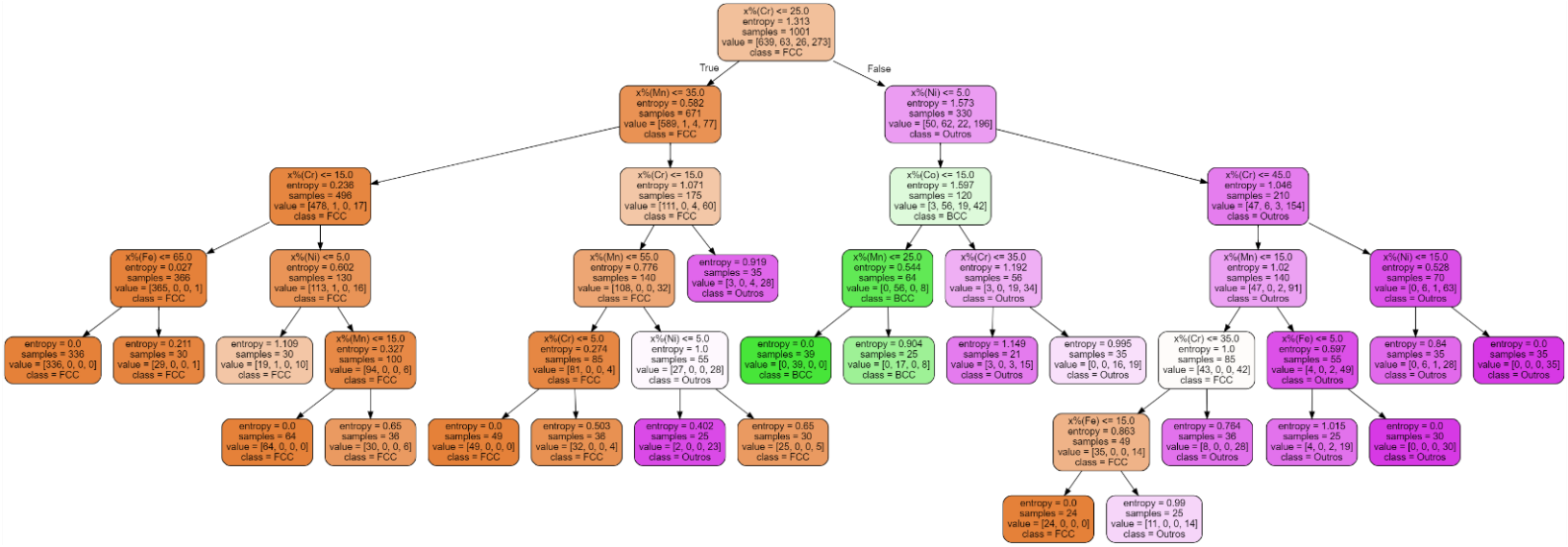


Figura 14: Conjunto de hiperparâmetros para o algoritmo de árvore de decisão. A) Primeiro grupo entradas (apenas percentuais atômicos). B) Segundo grupo de entradas (inclusão de VEC).

O algoritmo de árvore apresenta um diferencial único em relação aos demais modelos, a sua facilidade de representação visual, graças ao esquema de árvore, que permite enxergarmos o caminho de escolhas feitas com base nos critérios de decisão, que são escolhidos de acordo seu critério de ganho de informação, e que para ambos os modelos de árvore encontrados para os diferentes grupos, este é medido pela entropia. Apesar de diferentes hiperparâmetros serem necessários para cada grupo de entradas, é possível representar qualquer árvore de decisão, como mostrado na figura 15 . Dado o tamanho da árvore, graças a sua profundidade e quantidade de decisões, as imagens também se encontram no repositório para melhores visualizações

A



B

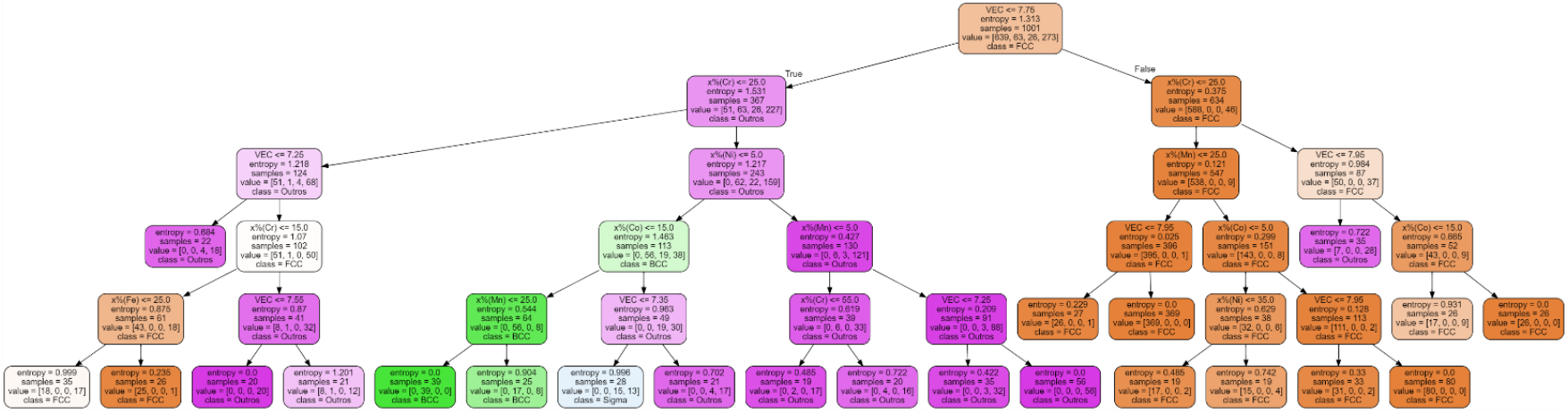


Figura 15: Árvores de decisão originadas por cada algoritmo. A) Primeiro grupo entradas (apenas percentuais atômicos). B) Segundo grupo de entradas (inclusão de VEC).

Depois de construído e otimizado o modelo de árvore de decisão, partiu-se para estruturação do algoritmo de aprendizado de máquina KNN. Nele, os procedimentos feitos para construção do modelo são semelhantes, onde foi utilizada a base de dados de passo 10 para realizar o treinamento e otimização, e a base de passo 8, com variações de 12,5% para um teste de previsão.

No KNN existe uma etapa adicional quando comparado ao algoritmo anterior, que é a fase de normalização dos dados. Como seu procedimento envolve medidas de distâncias para avaliar os k vizinhos mais próximo, os valores existentes na escala influenciam seu desempenho, dessa forma é necessário modificar a escala das características de entrada. Assim, a normalização é uma técnica geralmente aplicada como parte da preparação de dados para o aprendizado de máquina, na qual o objetivo é mudar os valores das colunas numéricas no conjunto de dados, para usar uma única escala comum, sem distorcer as diferenças nos intervalos de valores nem perder informações. Essa padronização dentro do código é feita utilizando uma função chamada StandartScaler(), a qual realiza a seguinte operação:

$$Z = \frac{x-\mu_x}{\sigma_x} \quad (4)$$

Onde Z seria o novo valor padronizado, x é o elemento que está sendo padronizado, μ_x é a média dos valores da característica de x de todas as amostras, e σ_x é o desvio padrão delas. Desse modo todos os valores ficam dentro de uma escala de -1 e 1, possibilitando a aplicação de modelos nos quais as grandezas numéricas interferem no aprendizado, como é o caso do método KNN e SVC que veremos adiante.

Uma vez normalizada a escala das características, temos novamente dois grupos, onde para cada uma foi necessário um processo de normalização. A partir disso, teve-se o treinamento do modelo, que envolve também a validação cruzada e otimização de hiperparâmetros como parte do processo. Para o KNN, os parâmetros relevantes são quantos vizinhos mais próximos considerar (valore de k, onde no algoritmo é definido *por n_neighbors*), o peso que a distância terá nessa classificação (onde por exemplo, todos os pontos em cada vizinhança podem ser ponderados igualmente ou não, definido por *weights*), a medida de distância a ser utilizada (como a euclidiana, definida por *metric*) e o valor aproximado das vizinhanças, onde o valor ideal varia bastante a depender da natureza do problema (definido em *leaf_size*). Abaixo, na figura 16, temos o espaço de hiperparâmetros explorados de maneira aleatória para ambos os

grupos de características, onde para cada espaço foram varridas e comparadas 100 diferentes combinações.

A

```
espaco_de_parametros_knn= {
    'n_neighbors': randint(5,15),
    'weights': ['uniform','distance'],
    'metric': ['minkowski','euclidean','manhattan'],
    'leaf_size': randint(30,120)
}
```

B

```
espaco_de_parametros_knn2= {
    'n_neighbors': randint(5,15),
    'weights': ['uniform','distance'],
    'metric': ['minkowski','euclidean','manhattan'],
    'leaf_size': randint(30,120)
}
```

Figura 16: Espaço de hiperparâmetros varrido para otimização do melhor modelo de k-vizinhos mais próximos. A) Espaço de hiperparâmetros para o primeiro grupo entradas (apenas percentuais atômicos). B) Segundo grupo de entradas (inclusão de VEC).

Com o resultado da busca, selecionou-se os melhores modelos para cada grupo de características de entrada, e estes foram treinados com toda a base de passo 10, e depois aplicados ao conjunto de ligas desconhecido da base de passo 8. Como resultado foram alcançadas melhores acurácias quando comparado ao modelo de árvore, onde o KNN para o primeiro grupo de entradas atingiu um percentual de 93,33%, já o algoritmo que considera o VEC também como uma informação de entrada, atingiu uma acurácia de 94,95%. Abaixo, nas figuras 17 e 18, podemos ver o conjunto de hiperparâmetros para cada modelo e suas respectivas matrizes de confusão para cada algoritmo.

A

```
KNeighborsClassifier
KNeighborsClassifier(leaf_size=117, metric='manhattan', n_neighbors=12,
                     weights='distance')
```

B

```
KNeighborsClassifier
KNeighborsClassifier(leaf_size=96, metric='manhattan', n_neighbors=8,
                     weights='distance')
```

Figura 17: Conjunto de hiperparâmetros para o algoritmo k-vizinhos mais próximos. A) Primeiro grupo entradas (apenas percentuais atômicos). B) Segundo grupo de entradas (inclusão de VEC).

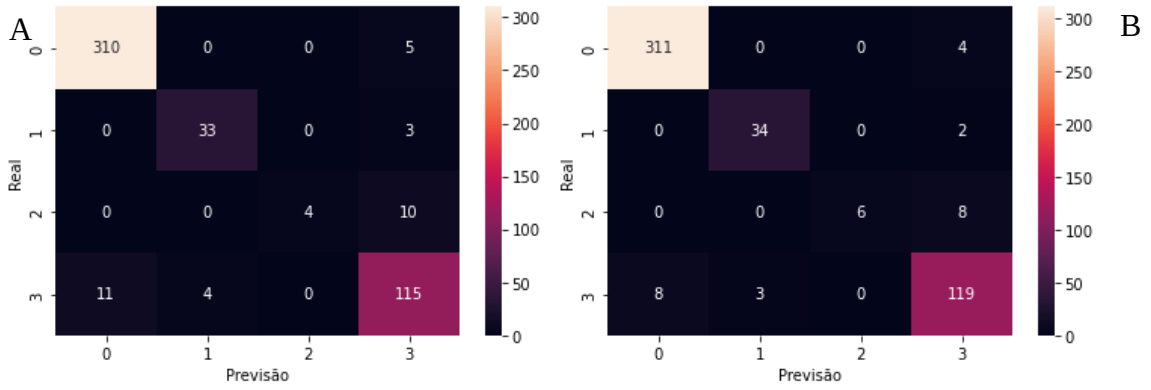


Figura 18: Matrizes de confusão do algoritmo k-vizinhos mais próximo. A) Primeiro grupo entradas (apenas percentuais atômicos). B) Segundo grupo de entradas (inclusão de VEC).

Como é possível ver pelas matrizes de confusão, para o modelo de aprendizado de máquina do tipo KNN, a inclusão da informação da concentração dos elétrons de valência foi benéfica para a classificação de todos os tipos de fases a serem identificadas consideradas no problema, como é possível observar, para as fases CFC temos um acerto de 311 no modelo B das 315 de referência, além disso, as fases CCC e sigma também possuem excelentes desempenhos, conde na CCC temos uma boa coincidência entre o algoritmo e a Pandat de 34 das 36 ligas com essa fase e para a sigma 6 das 14. A classificação do grupo “outros” também se mostra muito assertiva, batendo 119 das 130 ligas .

Por fim, temos os resultados obtidos do algoritmo do tipo SVM, onde trabalhamos com funções de Kernel, o que permite realizarmos uma separação linear do problema e assim trabalhar com uma condição SVC.

Para o SVC, o treinamento e teste ocorreu com os mesmos dados dos modelos anteriores, a mudança está na forma como esse algoritmo realiza a classificação, e por isso sua modelagem se difere quanto a otimização, já que este também possui seus hiperparâmetros específicos, entre eles temos o parâmetro de regularização (responsável pela tolerância da classificação incorreta, influenciando principalmente as margens envolta do hiperplano traçado, definido por C), a função responsável por transformar os dados de entrada no formato necessário (definidas pelo o *kernel*), o grau da função (sendo definido por *degree*, esse hiperparâmetro só é válido quando o kernel utilizado pelo modelo é do tipo *polinomial*, caso contrário ele é desconsiderado) e o parâmetro que decide os limites de curvatura do hiperplano (definido por *gamma*). Desse modo, assim como os demais modelos, foram vasculhados de modo aleatório dentro dos espaços composicionais ilustrados na figura 19 abaixo, 100 diferentes combinações para cada grupo de entradas.

```
espaco_de_parametros_svc= {
    'C': randint(1,500),
    'kernel':['rbf','poly','sigmoid'],
    'degree': randint(2,8),
    'gamma': ['scale','auto']
}
```
A

```
espaco_de_parametros_svc2= {
    'C': randint(1,500),
    'kernel':['rbf','poly','sigmoid'],
    'degree': randint(2,8),
    'gamma': ['scale','auto']
}
```
B

Figura 19: Espaço de hiperparâmetros varrido para otimização do melhor modelo SVC. A) Espaço de hiperparâmetros para o primeiro grupo entradas (apenas percentuais atômicos). B) Segundo grupo de entradas (inclusão de VEC).

Após a identificação dos melhores algoritmos para cada grupo, foi executado o treinamento com a totalidade dos dados da base de passo 10 e em seguida testados em uma base desconhecida pelo modelo (base de testes, de passo 8). Como resultado atingiu-se uma acurácia de 96,36% para o primeiro conjunto de entradas e 96,36% para o segundo. Nas matrizes de confusão ilustradas na figura 20 é possível ver o desempenho desse modelo para cada fase identificada corretamente ou não, enquanto na figura 21 pode ser identificado o conjunto de parâmetros de cada grupo de entradas.

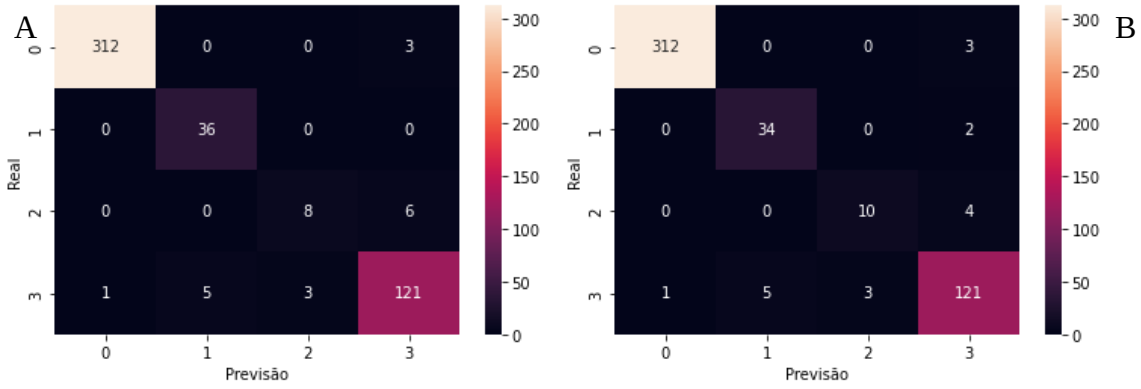


Figura 20: Matrizes de confusão do algoritmo SVC. A) Primeiro grupo entradas (apenas percentuais atômicos). B) Segundo grupo de entradas (inclusão de VEC).

```
{'C': 86, 'degree': 4, 'gamma': 'scale', 'kernel': 'rbf'}
```
A

```
{'C': 86, 'degree': 4, 'gamma': 'scale', 'kernel': 'rbf'}
```
B

Figura 21: Conjunto de hiperparâmetros para o algoritmo SVC. A) Primeiro grupo entradas (apenas percentuais atômicos). B) Segundo grupo de entradas (inclusão de VEC).

Ao contrário dos demais modelos, ao considerar a concentração dos elétrons de valência como uma das características de entrada não se notou uma melhora no desempenho global do algoritmo. Além do disso, há ainda uma pequena deterioração para a classe CCC, porém uma melhora na classificação de Sigma. Tais resultados podem ser decorrentes da alta faixa de

precisão já atingida pelo SVC, o que pode justificar também o mesmo conjunto de hiperparâmetros para ambos os casos, dessa forma é possível identificar que a partir de elevadas faixas de acurácia é cada vez mais complexo otimizar e aprimorar o modelo.

Com isso, foi possível ver e analisar o potencial da ciência orientada para dados, como uma poderosa ferramenta a ser explorada dentro do campo da ciência e engenharia de materiais. Toda a estruturação, otimização e cálculos elaborados pelos algoritmos foram executados com um baixo poder de processamento, podendo ser elaborados em uma máquina de escritório comum, e ainda levando um tempo bastante inferior quando comparado a métodos mais complexos, como simulações termodinâmicas ou cálculos de DFT. Para fins de comparação, enquanto o modelo de ML com maior tempo de execução levou cerca 90 segundos para realizar a previsão de fases das ligas presentes na base de testes, a simulação termodinâmica feia pelo Pandat levou aproximadamente 4 horas, vale também ressaltar que uma vez treinado o modelo, novas previsões são extremamente rápidas, o que não é verdade para os cálculos termodinâmicos. Além disso, foi atingida uma ótima faixa de acurácia para a previsão de fases em ligas multicomponentes, formadas pela combinação dos elementos selecionados. Na tabela 2 podemos ver um resumo dos resultados atingidos pelos diferentes algoritmos, entre eles temos a acurácia média, que pode ser esperada por cada um, desvio padrão e tempo decorrido para execução de 100 combinações de hiperparâmetros.

Modelo	Grupo de entradas[1]	Média de acurácia[2]	Desvio padrão	Resultado alcançado	Tempo de execução[3]
Árvore de decisão	A	88,51%	± 2,66	88,48%	9,36 s
Árvore de decisão	B	89,91%	± 4,18	91,72%	9,36s
KNN	A	91,31%	± 1,61	93,33%	7,52s
KNN	B	94,31%	± 1,65	94,95%	7,48 s
SVC	A	96,11%	± 1,51	96,36%	93,87 s
SVC	B	96,00%	±1,95	96,36%	74,16 s

Tabela 2: Resumo de resultados dos algoritmos de aprendizado de máquina. 1- Grupo A considera apenas os percentuais atômicos de cada elemento, grupo B considera também o VEC. 2- Acurácia média que pode ser esperada ao executar o modelo otimizado. 3- Tempo total para execução de 100 diferentes combinações de hiperparâmetros e validações cruzadas.

5. CONCLUSÃO E CONSIDERAÇÕES FINAIS

A partir do contexto técnico apresentado neste trabalho, é possível compreender a importância da modelagem de novas ligas para a evolução da ciência e engenharia de materiais, uma vez que, novos materiais podem trazer um novo arranjo de propriedades especificas, e com isso novas e diferentes aplicações, ou ainda a otimização de algumas já existentes. Porém, quando falamos na confecção e estudo de novas ligas multicomponentes tratamos de um desafio complexo, dado a infinidade de possibilidades existentes e ainda não exploradas.

A produção e caracterização tradicional já se mostra como um caminho inviável, visto a quantidade de recursos naturais, tempo e dinheiro que seriam gastos na tentativa de uma modelagem de sucesso. A partir daí, tem-se a origem da modelagem auxiliada por cálculos teóricos e computação, que possibilitam boas estimativas e são capazes de executar simulações termodinâmicas coerentes como visto ao longo deste trabalho. Contudo, muitos desses métodos teóricos evolvem cálculos complexos, onde suas reproduções podem exigir muito processualmente, além de possuírem ciclos de desenvolvimento relativamente longos que não acompanham a real necessidade do mundo moderno.

Levando isso em consideração, e olhando para a grande evolução da ciência orientada para big data nos últimos anos, este trabalhou tratou de analisar a possibilidade de combinar o aprendizado de máquina supervisionado com a modelagem de novas ligas multicomponentes, onde o foco foi a previsão de fases CFC, CCC, Sigma ou outras, contidas em diferentes ligas, todas compostas por um mesmo conjunto de elementos, os mesmos da liga de Cantor (contendo diferentes percentuais atômicos de Ni, Mn, Fe, Cr e Co).

Como resultado, pode-se observar que o objetivo principal foi atingido, onde para todos os algoritmos treinados e testados foram alcançados desempenhos muito promissores para a previsão de fases. Entre os diferentes modelos construídos, o SVC foi o que obteve o melhor desempenho, atingindo uma maior acurácia global, de 96,36 ± 1,95 % quando comparado com a base de referência, extraída do Pandat (o qual utilizou do processo de simulação termodinâmica, a partir da metodologia CALPHAD), além disso, foi observado que, a inclusão de mais características de entrada como, a concentração dos elétrons de valência, contribuiu positivamente para o desempenho de alguns modelos. Além de maior acurácia, o SVC também é o algoritmo com maior necessidade processual entre os modelos de aprendizado de máquina construídos, porém, todos eles comparados a métodos teóricos mais complexos como DFT ou a própria simulação termodinâmica feita pelo Pandat são muito mais rápidos para realizar uma classificação e realizar a previsão de fases. Como exemplo dessa diferença. temos o tempo para previsão das fases da base de teste, onde modelo de ML com maior ciclo para o cálculo ainda

foi cerda de 160 vezes mais rápido que a simulação feita pelo Pandat com a simulação termodinâmica.

Portanto, de modo geral, é possível concluir que o machine learning é uma ferramenta muito poderosa e que poder ser tranquilamente aliada da engenharia de materiais quando tratamos do desenvolvimento de nova ligas multicomponentes, onde o aprendizado de máquina, além desta aplicação, aparece na literatura também na predição de propriedades e em outros temas. Também é possível dizer que, para cada conjunto de entradas é possível alcançar um modelo ótimo para aplicação, onde este vai sempre variar conforme as entradas declaradas e o objetivo a ser alcançado, onde quanto maior é o refinamento e seleção de características de entradas e hiperparâmetros selecionados, melhor é o desempenho do modelo.

Por fim, é preciso levar em consideração que um modelo que acerte 100% dos casos é praticamente impossível, dado que atingir essa taxa de acerto de forma consistente seria muito difícil, mas que, com os resultados atingidos já é possível obter uma boa orientação quanto as fases a serem esperadas por uma liga composta pelos elementos estudados, e com isso economizar muito tempo e recursos na confecção de novos materiais.

Como próximos passos, um caminho possível a ser explorado, pode ser a expansão das bases de treino, juntamente com a seleção de mais características de entrada, como por exemplo a entalpia, fração molar, raio atômico, ponto de fusão, a fim de refinar ainda mais os resultados e acurácia atingida, para a partir de dados resultantes da simulação, realizar a comprovação por meio do caminho experimental, onde a partir destas, pode ser possível comparar as previsões do algoritmo e a realidade observada experimentalmente. Até mesmo para casos divergentes entre algoritmos de aprendizado de máquina e outros métodos teóricos, como o próprio CALPHAD, o caminho experimental pode comprovar quais dos modelos teóricos realmente possui uma melhor previsão quanto a fase a ser obtida.

BIBLIOGRAFIA

[1] – B. Cantor, I.T.H. Chang, P. Knight, A.J.B. Vincent, Microstructural development in equiatomic multicomponent alloys, Mater. Sci. Eng. A 375e377 (2004) 213-218.

[2] – J.-W. Yeh, S.-K. Chen, S.-J. Lin, J.-Y. Gan, T.-S. Chin, T.-T. Shun, C.-H. Tsau, S.-Y. Chang, Nanostructured high-entropy alloys with multiple principal elements: Novel alloy design concepts and outcomes, Adv. Eng. Mater. 6 (2004) 299-303.

[3] – B. Gludovatz, A. Hohenwarter, D. Catoor, E. H. Chang, E. P. George, and R. O. Ritchie, Science 345, 1153 (2014).

[4] – D. Li, C. Li, T. Feng, Y. Zhang, G. Sha, J. J. Lewandowski, P. K. Liaw, and Y. Zhang, Acta Mater. 123, 285 (2017).

[5] – C. Liu, H. Wang, S. Zhang, H. Tang, and A. Zhang, J. Alloys Compd. 583, 162 (2014).

[6] – Wei J, Chu X, Sun X-Y, et al. Machine learning in materials science. InfoMat. 2019; 1:338–358. https://doi.org/10.1002/inf2.12028

[7]- Wu W, Sun Q. Applying machine learning to accelerate new materials development. Sci Sin Phys Mech Astron. 2018;48: 107001.

[8]- Mantaras RL, Armengol E. Machine learning from examples: inductive and lazy method. Data Knowl Eng. 1998; 99:99-123.

[9] – Y.Li, W.Guo, Machine-learning model for predicting phase formations of high-entropy alloys. Physical Review Materials 3, (2019)

[10] – R. Machaka , Machine learning-based prediction of phases in high-entropy alloys, Computational Materials Science 188 (2021). https://doi.org/10.1016/j.commatsci.2020.110244

[11] – W. Huang, P. Martin, H.L. Zhuang, Machine-learning phase prediction of high-entropy alloys, Acta Materialia, https://doi.org/10.1016/j.actamat.2019.03.012.

[12] – JIEN-WEI YEH et al. High-Entropy Alloys – A New Era of Exploitation. Materials Science Forum Vol. 560, pp 1-9, 2007.

[13]- T.K. Chen, T.T. Shun, J.-W. Yeh, M.S. Wong, Nanostructured nitride films of multi-element high-entropy alloys by reactive DC sputtering, Surf. Coat. Technol. 188e189 (2004) 193e200.

[14] C.-Y. Hsu, J.-W. Yeh, S.-K. Chen, T.-T. Shun, Wear resistance and hightemperature compression strength of FCC CuCoNiCrAl0.5Fe alloy with boron addition, Metall. Mater. Trans. A 35A (2004) 1465e1469.

[15] P.-K. Huang, J.-W. Yeh, T.-T. Shun, S.-K. Chen, Multi-principal-element alloys with improved oxidation and wear resistance for thermal spray coating, Adv. Eng. Mater. 6 (2004) 74e78.

[16] J.-W. Yeh, S.-K. Chen, J.-W. Gan, S.-J. Lin, T.-S. Chin, T.-T. Shun, C.-H. Tsau, S.-Y. Chang, Formation of simple crystal structures in Cu-Co-Ni-Cr-Al-Fe-Ti-V alloys with multiprincipal metallic elements, Metall. Mater. Trans. A 35A (2004) 2533e2536.

[17] J.-W. Yeh, S.-K. Chen, S.-J. Lin, J.-Y. Gan, T.-S. Chin, T.-T. Shun, C.-H. Tsau, S.-Y. Chang, Nanostructured high-entropy alloys with multiple principal elements: Novel alloy design concepts and outcomes, Adv. Eng. Mater. 6 (2004) 299e303.

[18] J.-W. Yeh, Recent Progress in High Entropy Alloys, Ann. Chim. Sci. Mat. 31 (2006) 633e648.

[19] – D.B MIRACLE, O.N SENKOV. A critical review of high entropy alloys and related concepts. Acta Materialia, v. 122, p. 448-511, 2017. http://dx.doi.org/10.1016/j.actamat.2016.08.081

[20]- Web of Science. Disponível em: < https://www.webofscience.com/wos/woscc/summary/97ª8b5ec-542e-4ed5-90e0-50f3566ab9f8-5f6aa447/relevance/1 >. Accesso em 19/11/2022.

[21] – Y. Zhang, T.T. Zuo, Z. Tang, M.C. Gao, K.A. Dahmen, P.K. Liaw, Z.P. Lu, Microstructures and properties of high-entropy alloys, Prog. Mat. Sci. 61 (2014).

[22] -B.S. Murty, J.-W. Yeh, S. Ranganathan, High-entropy Alloys, 2014.

[23] – J.-W. Yeh, Recent Progress in High Entropy Alloys, Ann. Chim. Sci. Mat. 31 (2006) 633e648.

[24]- M.-H. Tsai, J.-W. Yeh, High-entropy alloys: a critical review, Mater. Res. Lett. 2 (2014) 107e123.

[25] – GLUDOVATZ, B. et al. A fracture-resistant high-entropy alloy for cryogenic applications. Science, v. 345, n. 6201, p. 1153, 2014. Disponível em: < http://science.sciencemag.org/40ontente/345/6201/1153.abstract >.

[26] - YOUSSEF, K.; ROBERTO, S. R. Applications of salt solutions before and after harvest

affect the quality and incidence of postharvest gray mold of 'Italia' table grapes. Postharvest Biology and Technology, v. 87, p. 95-102, 1// 2014. ISSN 0925-5214. Disponível em: < http://www.sciencedirect.com/science/article/pii/S0925521413002573 >.

[27] - KOŽELJ, P. et al. Discovery of a Superconducting High-Entropy Alloy. Physical Review Letters, v. 113, n. 10, p. 107001, 09/02/ 2014. Disponível em: < http://link.aps.org/doi/10.1103/PhysRevLett.113.107001 >.

[28] - LEE, C. P. et al. The Effect of Boron on the Corrosion Resistance of the High Entropy Alloys Al0.5CoCrCuFeNiB x. Journal of The Electrochemical Society, v. 154, n. 8, p. C424-C430, 2007. Disponível em: < http://jes.ecsdl.org/content/154/8/C424.abstract >.

[29] - Kaufman L, Bernstein H (1970) Computer calculation of phase diagrams. Academic, New York

[30] - Hillert M (1968) Phase transformations. ASM, Cleveland

[31] - Zhang, C., Gao, M.C. (2016). CALPHAD Modeling of High-Entropy Alloys. In: Gao, M., Yeh, JW., Liaw, P., Zhang, Y. (eds) High-Entropy Alloys. Springer, Cham. https://doi.org/10.1007/978-3-319-27013-5_12

[32] - Chou KC, Chang YA (1989) A study of ternary geometrical models. Berichte der Bunsengesellschaft fu¨r physikalische Chemie 93(6):735–741. doi:10.1002/bbpc.19890930615

[33] - Ghahramani Z. "Unsupervised learning," in Advanced lectures on machine learning, ed: Springer, 2004; pp. 72- 112. https://doi.org/10.1007/978-3-540-28650-9_5

[34] - Kotsiantis SB, Zaharakis I, Pintelas P. Supervised machine learning: A review of classification techniques. ed, 2007.

[35] – U.V Kulkarni, S.V Shinde, "Neuro –fuzzy classifier based on the Gaussian membership function",4th ICCCNT 2013,July 4-6,2013,Tiruchengode,India.

[36] - Zhang D, Nunamaker JF. Powering e-learning in the new millennium: an overview of e-learning and enabling technology. Information Systems Frontiers 2003; 5: 207-218. https://doi.org/10.1023/A:1022609809036.

[37] - Maimon O, Rokach L. Introduction to supervised methods, in Data Mining and Knowledge Discovery Handbook, ed: Springer, 2005 pp. 149-164.

[38] - Ng A. "CS229 Lecture notes."

[39] - Kesavaraj G, Sukumaran S. A study on classification techniques in data mining. in Computing, Communications and Networking Technologies (ICCCNT), 2013 Fourth International Conference on, 2013; pp. 1-7.

[40] - Singh M, Sharma S, Kaur A. Performance Analysis of Decision Trees. International Journal of Computer Applications 2013; 71.

[41] - Zhang L, Ji Q. A Bayesian network model for automatic and interactive image segmentation, Image Processing, IEEE Transactions on, 2011; 20: 2582-2593. https://doi.org/10.1016/j.trc.2006.11.001

[42] - Soofi A A, Awan A. Classification Techniques in Machine Learning: Applications and Issues. Journal of Basic & Applied Sciences, 2017, 13, 459-465

[43] - Vikramaditya Jakkula, “Tutorial on Support Vector Machine” ,2013

[44] - Lior Rokach and Oded Maimon,IEEE Transaction On System, Man and Cybernetics Part C, Vol 1, No. 11, November Top Down Induction Of Decision Tree Classifier-A Survey,2002

[45] – Freitas, Tales. Entendendo as árvores de decisão em Machine Learning, Sigmoidal, 2022. Disponivel em: < https://sigmoidal.ai/entendendo-as-arvores-de-decisao-em-machine-learning/ >. Acesso em: 03/12/2022

[46] - Rutkowski L, Pietruczuk L, Duda P, Jaworski M. Decision trees for mining data streams based on the McDiarmid's bound. Knowledge and Data Engineering, IEEE Transactions on, 2013; 25: 1272-1279. https://doi.org/10.1109/TKDE.2012.66

[47] - Patil DD, Wadhai V, Gokhale J. Evaluation of decision tree pruning algorithms for complexity and classification accuracy, 2010.

[48] - Quinlan JR. Induction of decision trees. Machine learning 1986; 1: 81-106. https://doi.org/10.1007/BF00116251

[49] - Sharma S, Agrawal J, Agarwal S. Machine learning techniques for data mining: A survey, in Computational Intelligence and Computing Research (ICCIC), 2013 IEEE International Conference on, 2013; pp. 1-6.

[50] - Bhukya DP, Ramachandram S. Decision tree induction: an approach for data classification using AVL-tree. International Journal of Computer and Electrical Engineering 2010; 2: 660. https://doi.org/10.7763/IJCEE.2010.V2.208

[51]- 1Mr. Brijain R Patel, 2Mr. Kushik K Rana Department of computer engineering, GEC Modasa, India Assistant Professor, Department of computer engineering, GEC Modasa, India, Presentation Slides "A Survey on Decision Tree Algorithm for Classification". IJEDR, 2014

[52] - Vapnik VN. The Nature of Statistical Learning Theory, 1995.

[53] - Nizar A, Dong Z, Wang Y. Power utility nontechnical loss analysis with extreme learning machine method. Power Systems, IEEE Transactions on, 2008; 23: 946-955. https://doi.org/10.1109/TPWRS.2008.926431

[54] - Xiao H, Peng F, Wang L, Li H. Ad hoc-based feature selection and support vector machine classifier for intrusion detection, in 2007 IEEE International Conference on Grey Systems and Intelligent Services, 2007; pp. 1117-1121. https://doi.org/10.1109/GSIS.2007.4443446

[55] - Berwick R. An Idiot's guide to Support vector machines (SVMs)

[56]- Ahmad I, Abdulah AB, Alghamdi AS. Towards the designing of a robust intrusion detection system through an optimized advancement of neural networks, in Advances in Computer Science and Information Technology, ed: Springer, 2010; pp. 597-602

[57] – Somvanshi, Madan. Tambade, Shital. Chavan, Pranjali. S.V. Shinde. Department of Information Technology, Pimpri Chinchwad College of engineering, Pune,India. "A Review of Machine Learning Techniques using Decision Tree and Support Vector Machine".

[58] - Premanand S. The A-Z guide to Support Vector Machine, analytics Vidhya, 2022. Disponivel em : < https://www.analyticsvidhya.com/blog/2021/06/support-vector-machine-better-understanding/#:~:text=The%20limitation%20of%20SVC%20is,we%20call%20it%20as%20SVM.> Acesso em: 03/12/2022.

[59] - Tom Mitchell, Machine Learning, McGrawHill Computer science series, 1997

[60]- D. Hand, H. Mannila, P. Smyth.: Principles of Data Mining. The MIT Press. (2001)

[61] - Zhang Z. Introduction to machine learning: k-nearest neighbors. Ann Transl Med 2016;4(11):218. doi: 10.21037/atm.2016.03.37

[62] - Zhang Z. Too much covariates in a multivariable model may cause the problem of overfitting. J Thorac Dis 2014;6:E196-7.

[63] - Lantz B. Machine learning with R. 2nd ed. Birmingham: Packt Publishing; 2015:1.

[64] - Short RD, Fukunaga K. The optimal distance measure for nearest neighbor classification. IEEE Transactions on Information Theory 1981; 27:622-7.

[65] - Weinberger KQ, Saul LK. Distance metric learning for large margin nearest neighbor classification. The Journal of Machine Learning Research 2009; 10:207-44.

[66] - Cost S, Salzberg S. A weighted nearest neighbor algorithm for learning with symbolic features. Machine Learning 1993; 10:57-78.

[67] - R. E. Shawi, M. Maher, S. Sakr, Automated machine learning: Stateof-the-art and open challenges, arXiv preprint arXiv:1906.02287, (2019). http://arxiv.org/abs/1906.02287.

[68] - M. Kuhn and K. Johnson, Applied Predictive Modeling., Springer (2013) ISBN: 9781461468493.

[69] - F. Hutter, L. Kotthoff, and J. Vanschoren, Eds., Automatic Machine Learning: Methods, Systems, Challenges, Springer (2019) ISBN: 9783030053185.

[70]- N. Decastro-Garc´ıa, A. L. Mu˜noz Casta˜neda, D. Escudero Garc´ıa, and ´ M. V. Carriegos, Effect of the Sampling of a Dataset in the Hyperparameter Optimization Phase over the Efficiency of a Machine Learning Algorithm, Complexity 2019 (2019). https://doi.org/10.1155/2019/6278908.

[71] - D. Maclaurin, D. Duvenaud, R.P. Adams, Gradient-based Hyperparameter Optimization through Reversible Learning, arXiv preprint arXiv:1502.03492, (2015). http://arxiv.org/abs/1502.03492.

[72] – J. Bergstra, R. Bardenet, Y. Bengio, and B. K´egl, Algorithms for hyperparameter optimization, Proc. Adv. Neural Inf. Process. Syst., (2011) 2546–2554.

[73] - B. James and B. Yoshua, Random Search for Hyper-Parameter Optimization, J. Mach. Learn. Res. 13 (1) (2012) 281–305.

[74] – Kohavi,R. (1995) A study of cross-validation and bootstrap for accuracy estimation and model selection. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence, Montreal, Quebec, Canada. Vol. 2, pp. 1137–1143.

[75] – Molinaro,A.M. et al. (2005) Prediction error estimation: a comparison of resampling methods. Bioinformatics, 21, 3301–3307.

[76] - Stone,M. (1974) Cross-validatory choice and assessment of statistical predictions. J. R. Stat. Soc. Series B Methodol., 36, 111–147.

[77] - Bengio,Y. et al. (2003) No unbiased estimator of the variance of K-fold cross-validation. J. Mach. Learn. Res., 5, 1089–1105

[78] - Saeid Parvandeh, Hung-Wen Yeh, Martin P Paulus, Brett A McKinney, Consensus features nested cross-validation, Bioinformatics, Volume 36, Issue 10, 15 May 2020, Pages 3093–3098, https://doi.org/10.1093/bioinformatics/btaa046.

[79] - Wenjiang Huang, Pedro Martin, Houlong L. Zhuang, Machine-learning phase prediction of high-entropy alloys, Acta Materialia, Volume 169, 2019, Pages 225-236, ISSN 1359-6454, https://doi.org/10.1016/j.actamat.2019.03.012.

[80] - Kusne, A. , Mueller, T. and Ramprasad, R. (2016), Machine learning in materials science: Recent progress and emerging applications, Reviews in Computational Chemistry, [online], https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=915933 (Accessed December 16, 2022).

[81] - M. L. Green, I. Takeuchi, and J. R. Hattrick-Simpers, J. Appl. Phys., 113, 231101 (2013). Applications of High Throughput (Combinatorial) Methodologies to Electronic, Magnetic, Optical, and Energy-Related Materials.

[82] - D. B. Miracle and O. N. Senkov, Acta Mater. 122, 448 (2017).

[83] -Y. Zhang, Y. J. Zhou, J. P. Lin, G. L. Chen, and P. K. Liaw, Adv. Eng. Mater. 10, 534 (2008).

[84] - S. Guo, C. Ng, J. Lu, and C. T. Liu, J. Appl. Phys. 109, 103505 (2011).

[85] - M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L.D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, K. Zieba, End to End Learning for Self-Driving Cars, 2016

[86] - L. Zhou, S. Pan, J. Wang, A.V. Vasilakos, Machine learning on big data: opportunities and challenges, Neurocomputing 237 (2017) 350–361, https://doi.org/10.1016/j.neucom.2017.01.026

[87] – Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos e David Cournapeau (2011). «Scikit-learn: Machine Learning in Python». Journal of Machine Learning Research. 12: 2825–2830

[88] - pandas (software). CONTEÚDO aberto. Em: WIKIPÉDIA: a enciclopédia livre. Disponível em: <https://pt.wikipedia.org/wiki/Pandas_(software)>. Acesso em: 21 dez. 2022.

APÊNDICE A

Link para Códigos e bases dados construída neste trabalho: < https://github.com/kkayquek/Classific-o-preditiva-de-fases-para-ligas-de-alta-entropia-Ni-Mn-Fe-Cr-e-Co-.git >.