INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Algoritmo de aprendizagem semi-supervisionada

ANDRÉ FILIPE DA CRUZ FONTES
Outubro de 2023

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

Algoritmo de aprendizagem semi-supervisionada

André Filipe da Cruz Fontes

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Sistemas de informação e conhecimento

Orientador: Fátima Rodrigues

Porto, Outubro, 2023

Dedicatória

Gostaria de dedicar este trabalho aos meus pais e a toda a minha família pelo incentivo dado em concluir a dissertação.

Resumo

Os dados etiquetados são essenciais para uma aprendizagem supervisionada. No entanto, estão frequentemente disponíveis apenas em pequenas quantidades, enquanto os dados não etiquetados podem ser abundantes. A utilização de dados não etiquetados juntamente com dados etiquetados é tanto de interesse teórico como prático.

A eficiência da aprendizagem supervisionada é altamente dependente das instâncias etiquetadas. Contudo, ter um tamanho razoável de instâncias etiquetadas pode ser difícil, dispendioso e demorado de obter, uma vez que envolve necessariamente conhecimento especializado, tais como anotadores humanos.

Este é um desafio comum na investigação em diversas áreas sendo mais comum na área da saúde, mais frequentemente fundamentado em estudos em que os participantes utilizam instrumentos de *self-report*.

Nesta dissertação foi elaborado um algoritmo semi-supervisionado com recurso a dois classificadores, support vector machine e random-forest. Os resultados são promissores, tendo-se obtido um acréscimo de 5% no desempenho do algoritmo, relativamente aos algoritmos em separado e com a capacidade de etiquetar praticamente todos os dados.

Palavras-chave: Semi-supervisionado, Inteligência Artificial, Metodologia de Investigação Científica em Design, CNN, LSTM

Abstract

Labelled data are essential for supervised learning. However, they are often available only in small quantities, while unlabelled data may be abundant. Using unlabelled data together with labelled data is of both theoretical and practical interest.

The efficiency of supervised learning is highly dependent on labelled instances. However, having a reasonable size of labelled instances may be difficult, expensive and time consuming to obtain since it necessarily involves expert knowledge, such as human annotators or filling self-reported questionnaires.

This is a common challenge in health research, most often founded in studies where participants use self-report instruments.

To address this issue, we can use semi-supervised learning methods that use both labelled and unlabelled data to construct a classifier and improve the classification performance.

In this dissertation, a semi-supervised algorithm was developed using two classifiers: support vector machines and random forests. The results are promising, having achieved a 5% increase in the performance of the algorithm in comparison to the separately used algorithms and with the ability to label virtually all data.

Keywords: Semi-supervised, Artificial Intelligence, Design Science Research Methodology, support vector machine, CNN, LSTM

Agradecimentos

Gostaria de agradecer à minha orientadora pela ajuda prestada na elaboração desta dissertação assim como todo o conhecimento partilhado, também gostaria de agradecer à minha família por me apoiarem e insistirem em concluir a dissertação e também agradecer todo o apoio dado pelos meus colegas de universidade e aos meus colegas de empresa.

Índice

Lista de Figuras

Lista de Tabelas

Acrónimos e Símbolos

Lista de Acrónimos

DSRM	*Design Science Research Methodology*
SVM	*Support vector machine*
SSL	*Semi supervised learning*
TSVM	*Transductive Support Vector Machine*
IA	*Inteligência Artificial*
VSM	*Value Stream Mapping*
AM	*Aprendizagem Máquina*
CNN	*Convolutional neural network*
LSTM	*Long short-term memory*

1 Introdução

1.1 Contexto

O progresso da tecnologia tem conquistado novos métodos de coletar dados com uma velocidade rápida, armazenamento mais fácil, maior e baixo custo tornando evidente a necessidade de técnicas que usem e analisem esses dados permitindo uma melhor extração de informação.

A aprendizagem semi-supervisionada tem atraído uma atenção significativa nos últimos anos, uma vez que se tem mostrado eficaz em várias aplicações, tais como o reconhecimento da fala, a classificação de imagens, a deteção de anomalias, entre outras.

O treino semi-supervisionado também está a chamar cada vez mais a atenção na era do "big data", já que a diferença entre a abundância de dados não etiquetados, coletados automaticamente e a escassez de dados etiquetados que por sua vez são trabalhosos e caros de obter.

Algoritmos de treino supervisionado requerem uma grande quantidade de dados etiquetados no processo de treino para construir modelos com alto desempenho de previsão, enquanto os algoritmos semi-supervisionados provam ser um paradigma poderoso para resolver o problema de dados não etiquetados e ter o alto desempenho de algoritmos supervisionados.

Nesta dissertação, irão ser realizadas experiências em vários conjuntos de dados para avaliar o desempenho do algoritmo de aprendizagem semi-supervisionada proposto em comparação com as abordagens tradicionais de aprendizagem supervisionada. Os resultados destas experiências proporcionarão uma visão da eficácia da aprendizagem semi-supervisionada em cenários do mundo real.

1.2 Descrição do problema

Em aprendizagem máquina existem quatro tipos de treino, supervisionado, não supervisionado, reforçado e semi-supervisionado.
Ao contrário dos algoritmos de treino não supervisionado, os algoritmos de treino supervisionado utilizam dados etiquetados. A partir desses dados, os algoritmos desenvolvem um modelo para prever valores futuros, no caso de um problema de regressão ou atribuir dados a categorias específicas no caso de um problema de classificação.
Os algoritmos de aprendizagem supervisionada, ao contrário dos algoritmos de aprendizagem não supervisionada, não havendo dados classificados em volume

suficiente, requerem um processo inicial para etiquetar os dados de forma apropriada.
O treino semi-supervisionado ocorre quando apenas uma parte dos dados estão etiquetados.
Para conjuntos de dados muito grandes é difícil e dispendioso etiquetar corretamente todos os dados recorrendo a peritos, isto porque sendo um processo manual pode conduzir a erros, o que seria também dispendioso se esses erros fossem detetados só depois do desenvolvimento dos modelos.
O treino não supervisionado e semi-supervisionado pode ser uma alternativa, se for demorado e dispendioso confiar nos conhecimentos de especialistas do domínio para etiquetar adequadamente os dados para a aprendizagem supervisionada.

1.3 Objetivos

O objetivo final desta dissertação é o desenvolvimento de um algoritmo de classificação semi-supervisionado competitivo com os demais existentes, para isso envolve os seguintes objetivos:
- Levantamento da literatura sobre abordagens semi-supervisionadas existentes
- Desenvolvimento do algoritmo de classificação semi-supervisionado
- Avaliação do algoritmo com vários conjuntos de dados
- Comparação dos resultados com outros algoritmos de última geração

Nesta dissertação, é pretendido fornecer um estudo abrangente da aprendizagem semi-supervisionada, incluindo os seus fundamentos teóricos, várias abordagens, e aplicações práticas. Primeiro serão explorados os diferentes tipos de algoritmos de aprendizagem semi-supervisionada, incluindo modelos generativos, métodos baseados em grafos, e técnicas de autoformação. Depois, iremos investigar as vantagens e limitações de cada abordagem, juntamente com os seus casos específicos de utilização.
Globalmente, esta dissertação visa proporcionar uma compreensão profunda da aprendizagem semi-supervisionada, das suas aplicações práticas, e do impacto potencial que poderá ter no futuro da aprendizagem máquina.

1.4 Metodologia

Tem havido um interesse crescente nas áreas da engenharia para a Metodologia de Investigação Científica em Design, uma vez que esta metodologia se centra na criação de artefactos para um fim prático [1].
A metodologia usada durante esta dissertação é Metodologia de Investigação Científica em Design (DSRM), que irá orientar na resolução do problema apresentado, sugerindo vários artefactos.
Esses artefactos ajudam a estruturar a maneira de realização da pesquisa e da implementação de forma que se tenham orientações a seguir. Para a realização dos

artefactos são necessários vários passos que se deve fazer para chegar à solução pretendida.
Os passos desta metodologia são [2]:

1. Identificar bem o problema
2. Definir os objetivos para a solução
3. Criar o artefacto, isto é criar um algoritmo que classifique dados sem estarem na totalidade etiquetados.
4. Fazer desmonstração para provar que o algoritmo funciona.
5. Avaliar o algoritmo com várias medidas de avaliação comummente usadas na avaliação de algoritmos
6. Relatar os resultados e verificar a usabilidade do algoritmo

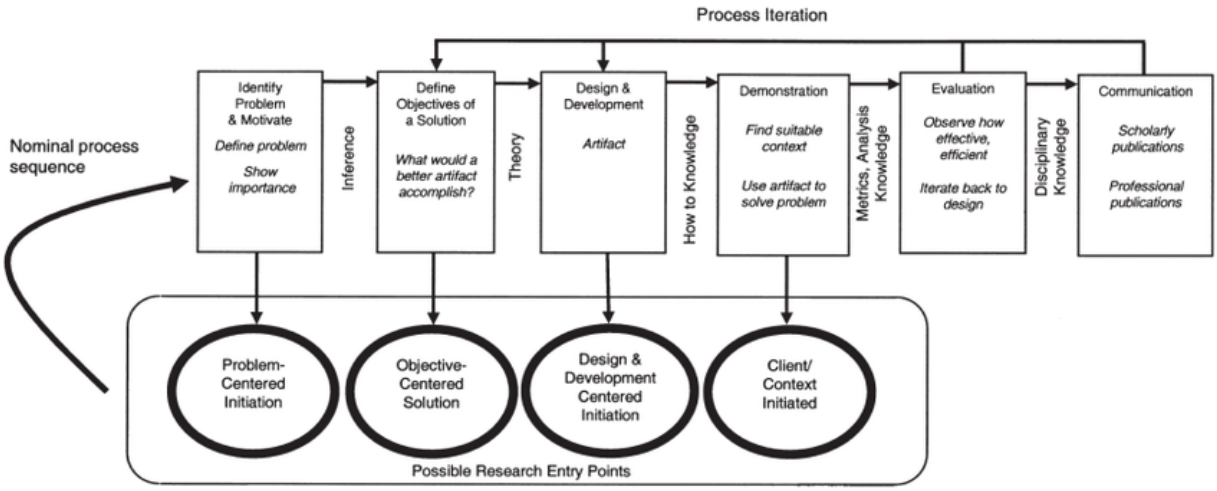

Figura 1 – Processo do DSRM [3]

Estes passos retratados no estudo de Peffers vão ser utilizados para criar a solução pretendida nesta dissertação, um algoritmo de classificação semi-supervisionado.

1.5 Estrutura do Documento

Este documento está dividido em 7 capítulos incluindo a introdução.
O capítulo 2, cujo título é **Estado da Arte**, aborda conceitos-chaves para o projeto feita através de uma revisão da literatura científica sobre os algoritmos semi-supervisionados.
O capítulo 3 corresponde à **Análise de Valor**, em que é apresentado os vários métodos utilizados para determinar a utilidade do projeto e como se deve proceder para elaborar o projeto.
O capítulo 4, corresponde à **Engenharia de Requisitos**, que apresenta os requisitos necessários para o projeto.
O capítulo 5, cujo título é **Implementação**, apresenta os pormenores técnicos da implementação da solução apresentada.

O capítulo 6, **Avaliação de solução,** apresenta as metodologias de testes empregues à solução. Também são detalhados os testes efetuados tal como o comportamento da solução perante os mesmos.
O capítulo 7, **Experimentos e resultados,** que apresenta os testes realizados e os resultados obtidos.
O último capítulo corresponde à **Conclusão** e retrata todas as conclusões retiradas neste projeto.

2 Estado de Arte

Este capítulo é referente ao estado de arte em que é feita uma revisão da literatura científica de algoritmos semi-supervisionados e sobre o seu aparecimento.
Para realizar a revisão da literatura científica será usada uma adaptação à metodologia PRISMA.

2.1 Metodologia da pesquisa

A metodologia PRISMA [4], também conhecida como *Preferred Reporting Items for Systematic Reviews and Meta-Analyses*, é uma metodologia amplamente reconhecida e recomendada para a realização de revisões sistemáticas e meta-análises em várias áreas de pesquisa. Ela fornece diretrizes e uma abordagem estruturada para garantir transparência, rigor e reprodutibilidade no processo de revisão.
A metodologia PRISMA serve como uma diretriz valiosa para garantir que o processo de revisão sistemática seja abrangente, transparente e reprodutível, aumentando assim a qualidade e confiabilidade das descobertas da revisão.
Referente à metodologia utilizada, em vez de ter perguntas mais específicas de pesquisa, vai ser usada uma abordagem mais genérica, mas completando todos os passos da metodologia para se obter uma boa revisão da literatura.

2.1.1 Perguntas para pesquisa

Uma revisão sistemática tenta coligir todas as provas empíricas que se encaixam em critérios de elegibilidade pré-especificados, a fim de responder a uma questão de investigação específica. Utiliza métodos explícitos e sistemáticos que são selecionados com vista a minimizar o enviesamento, fornecendo assim resultados mais fiáveis dos quais se podem tirar conclusões e tomar decisões".[4]

Tabela 1— Perguntas de pesquisa

Pergunta	Quantidade
P1	Que abordagens existem de algoritmos supervisionados?
P2	Quais as vantagens e desvantagens de usar um algoritmo semi-supervisionado?
P3	Qual o impacto de um algoritmo semi-supervisionado?

2.1.2 Base de dados da pesquisa

Uma vez definidas as questões de investigação, é necessário definir quais as bases de dados que iremos pesquisar. Para o efeito, foram escolhidas as bases de dados definidas no Tabela 2. Esta escolha teve em consideração a utilização de bases de dados semelhantes onde era possível aplicar os mesmos filtros, ou seja, a mesma consulta de pesquisa.

Tabela 2— Base de dados

Base de dados	URL
IEEE	https://www.ieee.org/
Web of Science	https://www.webofscience.com/wos/woscc/basic-search

2.1.3 Extração de informação

Para o processo de extração de informação começámos por extrair informação de diferentes bibliotecas digitais usando os termos de pesquisa e utilizando filtros data, apenas documentos produzidos nos últimos cinco anos. É possível observar na figura 2 os resultados obtidos na pesquisa das diferentes palavras-chave nas diferentes bases de dados.

Passos necessários para a aplicação da metodologia PRISMA:

1. Em primeiro lugar, obtém-se o diagrama do modelo PRISMA e preenche-se as informações das bases de dados utilizadas
2. Em seguida, procura-se nas bases de dados por documentos utilizando certos termos-chave e regista-se o número de documentos obtidos.
3. O terceiro passo caso haja fontes adicionais de documentos que não foram utilizadas.
4. Remover todas as duplicações usando Mendeley ou outra ferramenta.
5. Triagem de artigos através de títulos e resumos para encontrar artigos adequados com boa informação.
6. Elegibilidade, ver que documentos precisam de ser excluídos de acordo com os princípios de exclusão.
7. Revisão completa dos restantes artigos e ver que artigos são úteis para o trabalho.

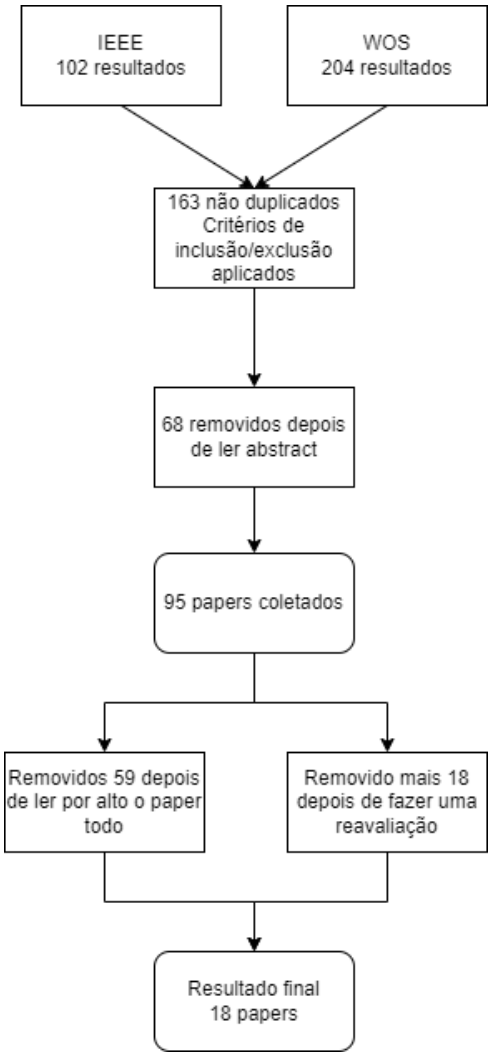

Figura 2 – Processo de eliminação de artigos

Depois de obter os artigos mais relevantes para esta revisão de literatura científica foi feita a leitura completa desses artigos para obter a informação necessária para realizar o estado de arte de algoritmos semi-supervisionados que vai ser referenciada neste capítulo.

2.2 Tipos de aprendizagem

Para entender treino semi-supervisionado é preciso entender primeiro como funcionam os algoritmos supervisionados e não supervisionados.

2.2.1 Aprendizagem supervisionada

Os métodos de aprendizagem supervisionada são métodos que usam dados etiquetados para ensinar o algoritmo a desenvolver um modelo para classificar dados e fazer previsões.

O algoritmo recebe uma entrada de dados assim como a saída pretendida desses dados para treinar o modelo. Depois de o treino acabar o modelo será capaz de identificar objetos das classes para que foi treinado.

Os algoritmos para chegarem a um bom modelo geralmente é necessário considerar várias combinações de valores para os seus parâmetros, configurando o algoritmo para identificar melhor as classes contidas nos dados.

Existem dois tipos de algoritmos supervisionados, algoritmos de classificação e de regressão:

- Os algoritmos de classificação lidam com categorias discretas. Os algoritmos de classificação comuns são classificadores lineares, máquinas vetoriais de suporte (SVM), árvores de decisão, k-vizinhos mais próximo e floresta aleatória.

- Os algoritmos de regressão são usados para fazer previsões de valores contínuos. regressão linear, regressão polinomial e árvores de regressão são algoritmos populares de regressão.

A solução que a ser desenvolvida vai se focar nos algoritmos de classificação.

Um exemplo de aprendizagem supervisionado é o estudo da classificação do nível de distração de um aluno num ambiente virtual educacional [5] que foi testado usando algoritmos supervisionados e também algoritmos não supervisionados usando CNN, LSTM e CNN-LSTM para ambos os casos.

Neste caso foi estudado os movimentos dos olhos para tentar perceber a atenção dos alunos, em que neste caso o método mais eficiente foi o supervisionado que perante um número suficiente de dados etiquetados apresentou uma boa performance.

2.2.2 Aprendizagem não supervisionada

Em contraste com o treino supervisionado existe o treino não supervisionado. Nesta abordagem, é apresentado ao algoritmo dados não etiquetados e este deteta autonomamente padrões ou semelhanças nos dados.
Na aprendizagem não supervisionada utilizando o processo de clustering, o problema reside no agrupamento de dados não etiquetados que apresentem o mesmo padrão. A sua divisão dos dados pode ser feita sem ter conhecimento propriamente sobre o assunto tratado, visto que a separação geralmente é feita por distância dos vetores.
O problema da aprendizagem não supervisionada é que até pode separar os dados em clusters, mas após esta divisão dos dados é necessário ter algum tipo de explicação ou descrição dos grupos encontrados. Sendo assim após uma aprendizagem não supervisionada, torna-se necessário recorrer a uma aprendizagem supervisionada.
Existem múltiplas utilidades para a aprendizagem não supervisionada estando bastante explorada devido ao facto da existência de grandes volumes de dados não etiquetados.
Uma das abordagens bastante explorada é na classificação de texto de modo a encontrar separações logicas no texto. Outro exemplo explorado num estudo feito por Burton R. [6]é a categorização de ataques de malware. Foi usada aprendizagem não supervisionada para registar os ataques em subconjuntos com características semelhantes. Não sabendo de antemão a natureza dos grupos ou agregados o algoritmo aprende-os a partir dos próprios dados. [6]
O objetivo era identificar agrupamentos estreitos que pudessem representar ataques criados pelo mesmo gerador de ataques. Por esta razão, não tentaram contabilizar todos os dados em falta e não se preocuparam com os valores anómalos. [6]
A característica a utilizar num agrupamento mais abrangente foi a distância de cada ataque de um conjunto de ataques de tipo de arco. Embora o agrupamento se baseasse em ataques, identificados exclusivamente por um emparelhamento de data e domínio, o agrupamento representava eficazmente os domínios, o que significa que um determinado domínio, mesmo que atacado em vários dias, era normalmente encontrado num único agrupamento. [6]
Este exemplo apresentado acima é uma de muitas aplicações dos algoritmos não supervisionados.

2.2.3 Aprendizagem por reforço

Como subcampo na aprendizagem de máquinas, a aprendizagem de reforço alcançou um tremendo desenvolvimento teórico e técnico em generalização, representação e eficiência, levando a sua aplicabilidade crescente a problemas da vida real, incluindo jogos, controlo robótico, gestão financeira e empresarial, condução autónoma, processamento de linguagem natural, visão por computador e arte criação. [7]
A aprendizagem por reforço é baseada em tentativa e erro de um agente que aprende a cada iteração.

O agente escolhe uma ação em cada período com base no seu estado atual, e recebe um feedback avaliativo e o novo estado do ambiente.
O agente de aprendizagem é o algoritmo ou modelo de aprendizagem da máquina, o objetivo do agente é aprender uma política ótima (ou seja, um mapeamento dos estados para as ações) que maximize a recompensa acumulada recebida ao longo do tempo. [7]

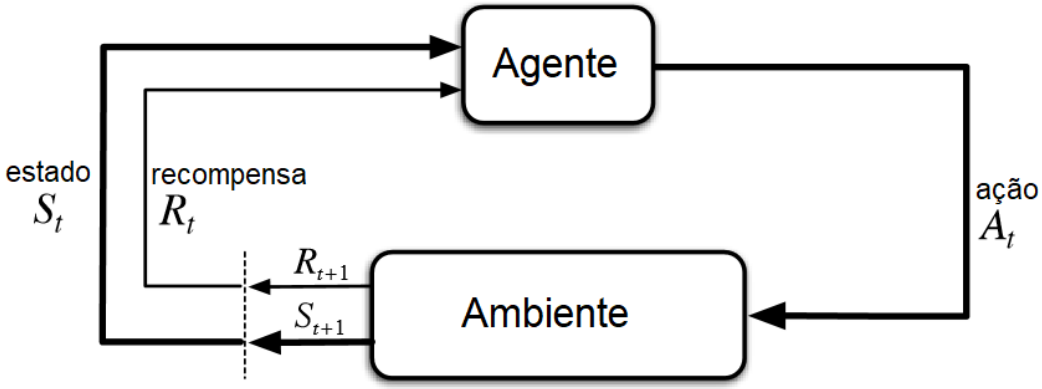

Figura 3 – Exemplo de funcionamento de aprendizagem por reforço

A aplicação da aprendizagem por reforço profundo no treino de veículos autónomos para navegarem em ambientes urbanos complexos, inspirados pelo trabalho de Smith concebido em um ambiente urbano simulado com layouts de estradas realistas, cruzamentos, semáforos, peões e outros elementos dinâmicos. [8]
O objetivo é ensinar o veículo autónomo a encontrar o caminho mais eficiente desde o seu ponto de partida até ao destino, respeitando as regras de trânsito, evitando colisões e otimizando o tempo de viagem. Utilizando um algoritmo de aprendizagem por reforço profundo, as ações do veículo são representadas como setas na vista de cima para baixo da grelha da cidade, refletindo a probabilidade de tomar ações específicas a partir de cada estado. O processo de treino envolve um equilíbrio entre exploração e aproveitamento, com o veículo a convergir gradualmente para uma rota ótima ao longo do tempo. [8]
Este exemplo apresentado por Smith mostra como o uso de aprendizagem por reforço é bastante útil e eficaz, pois permite fazer com o veículo consiga otimizar a sua rota.

2.2.4 Aprendizagem semi-supervisionada

Numa visão simplificada, o objetivo dos algoritmos de aprendizagem semi-supervisionada é aproveitar os dados não etiquetados para produzir um limite de decisão que reflita melhor a estrutura subjacente dos dados, usando os dados etiquetados como um professor para aprender e etiquetar os outros dados.

A aprendizagem semi-supervisionada pode referir-se tanto à aprendizagem transductiva como à aprendizagem indutiva. O objetivo da aprendizagem transductiva é inferir os etiquetas

corretos para os dados não etiquetados apenas e objetivo da aprendizagem indutiva é inferir o mapeamento correto.

Quando é que se deve usar algoritmos de aprendizagem semi-supervisionada? [9]

- Quando não há conjuntos de dados com etiquetas de alta qualidade de domínios semelhantes a utilizar para parametrização dos algoritmos.

- Quando os dados etiquetados são recolhidos por amostragem do conjunto dos dados não etiquetados, em vez de serem provenientes de uma distribuição ligeiramente diferente.

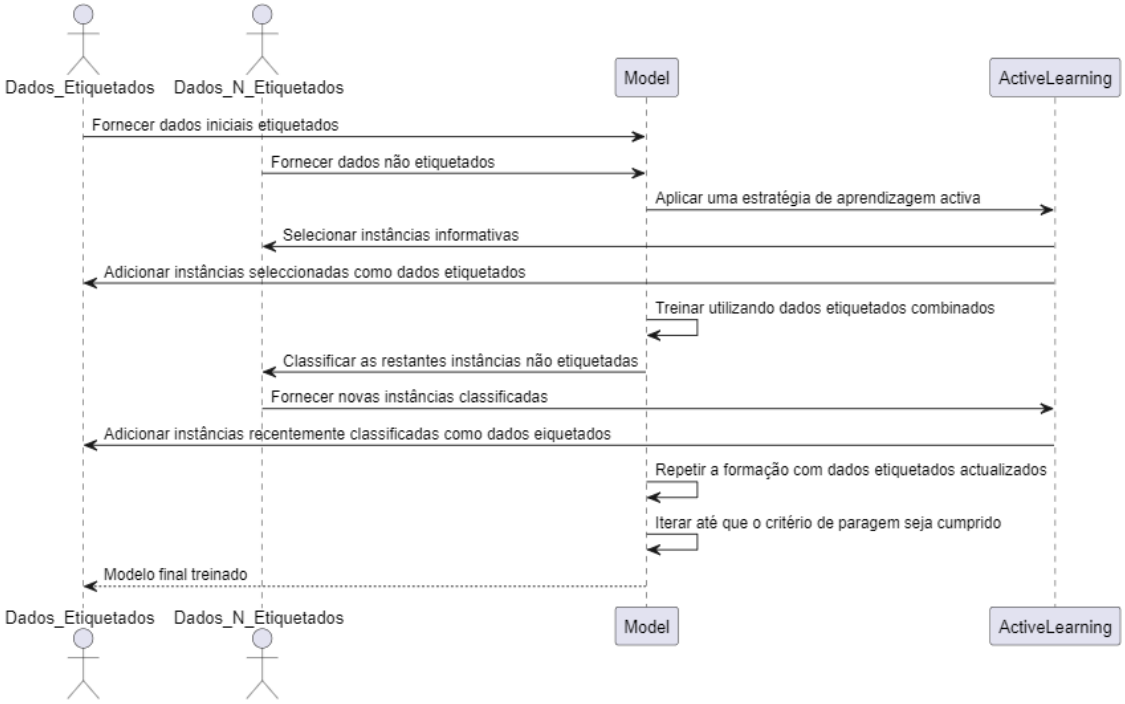

Figura 4 – Diagrama de sequência que demonstra um exemplo de um algoritmo semi-supervisionado a implementar

2.3 Suposições da aprendizagem semi-supervisionada

Um algoritmo semi-supervisionado assume o seguinte sobre os dados:
Pressuposto de continuidade - O algoritmo assume que os pontos que estão mais próximos uns dos outros são mais propensos a ter o mesmo rótulo de saída.
Suposição do cluster - Os dados podem ser divididos em clusters discretos e os pontos no mesmo cluster têm maior probabilidade de compartilhar um rótulo de saída.

Suposição múltipla - Os dados encontram-se aproximadamente em uma variedade de dimensão muito menor do que o espaço de entrada. Este pressuposto permite o uso de distâncias e densidades que são definidas em uma variedade.

2.4 Quando funciona a aprendizagem semi-supervisionada

A aprendizagem semi-supervisionada é particularmente útil quando há uma grande quantidade de dados não etiquetados disponíveis, mas é demasiado caro ou difícil de etiquetar tudo.
O custo associado ao processo de etiquetagem pode assim inviabilizar conjuntos de formação grandes e totalmente etiquetados, enquanto a aquisição de dados não etiquetados é relativamente pouco dispendiosa. Em tais situações, a aprendizagem semi-supervisionada pode ser de grande valor prático.
Dado dois conjuntos de dados idênticos, uma tarefa de aprendizagem supervisionada com um conjunto de dados totalmente etiquetado irá certamente treinar um modelo melhor do que um conjunto com uma porção de pontos não etiquetados. Mas a aprendizagem semi-supervisionada é poderosa quando as etiquetas são limitadas e os dados não etiquetados são abundantes. Neste caso, o modelo ganha exposição a instâncias que pode encontrar na implementação, sem investir tempo e dinheiro na etiquetagem de milhares e milhares de dados extra.
A aprendizagem semi-supervisionada tem sido aplicada numa variedade de contextos, incluindo processamento de linguagem natural, visão por computador e reconhecimento da fala. É especialmente útil em situações em que a obtenção de dados etiquetados pode ser difícil ou dispendiosa, mas onde grandes quantidades de dados não etiquetados estão prontamente disponíveis.

2.5 Taxonomia de métodos de aprendizagem semi-supervisionada

Existem vários métodos para construir um algoritmo semi-supervisionado como está referenciado na figura 5.

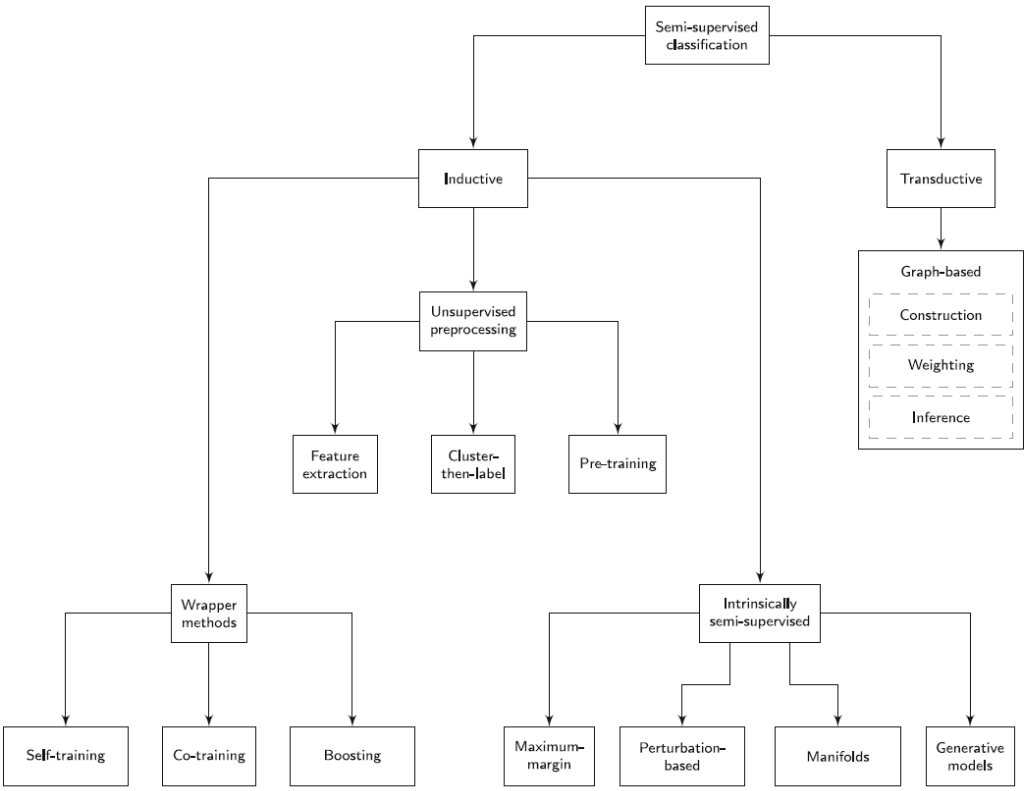

Figura 5 – Visualização da taxonomia de classificação semi-supervisionada [10]

Cada folha da taxonomia corresponde a um tipo específico de abordagem para incorporar dados não rotulados em métodos de classificação. Na folha correspondente aos métodos transdutivos baseados em grafos, as caixas tracejadas representam fases distintas do processo e cada uma delas tem uma multiplicidade de variações. [10]

Foi realizada uma pesquisa sobre estes métodos para verificar qual o mais relevante e para decidir em qual focar.

2.5.1 Métodos indutivos

Os métodos indutivos visam construir um classificador que possa gerar previsões para qualquer objeto no espaço de entrada. Podem ser utilizados dados não rotulados para treinar este classificador. Isto corresponde ao objetivo dos métodos de aprendizagem supervisionada: um modelo é construído na fase de treino e pode depois ser utilizado para prever as etiquetas de novos pontos de dados [10].

Dentro dos métodos existem 3 tipos de abordagem para realizar o algoritmo semi-supervisionado.

2.5.1.1 Métodos Wrapper

Utilizam um ou mais aprendizes de base supervisionados e treinam-nos iterativamente com os dados originais etiquetados, bem como com dados previamente não etiquetados que são aumentados com previsões de iterações anteriores dos aprendizes. O procedimento consiste normalmente em duas etapas alternadas as de treino e pseudo-labeling. Na fase de treino, um ou mais classificadores supervisionados são treinados com os dados etiquetados e, possivelmente, com os dados pseudo-etiquetados de iterações anteriores. Na etapa de pseudo-labeling, os classificadores resultantes são utilizados para inferir etiquetas para os objetos. [10]

2.5.1.2 Pré-processamento não supervisionado

Em segundo lugar, consideramos métodos de pré-processamento não supervisionados, que extraem características úteis dos dados não etiquetados, pré-agrupam os dados ou determinam os parâmetros iniciais de um processo de aprendizagem supervisionado de forma não supervisionada. Tal como os métodos wrapper, podem ser utilizados com qualquer classificador supervisionado. No entanto, ao contrário dos métodos wrapper, o classificador é fornecido apenas com pontos de dados originalmente etiquetados. [10]

2.5.1.3 Métodos intrinsecamente semi-supervisionados

A última classe de métodos indutivos que consideramos incorpora diretamente dados não etiquetados na função objetivo ou no procedimento de otimização do método de aprendizagem. Muitos destes métodos são extensões diretas de métodos de aprendizagem supervisionada para o cenário semi-supervisionado: eles estendem a função objetivo do classificador supervisionado para incluir dados não etiquetados. As máquinas de vetores de suporte semi-supervisionadas, por exemplo, alargam as SVMs supervisionadas maximizando a margem não só em dados etiquetados, mas também em dados não etiquetados.[10]

2.5.2 Métodos Transdutivos

Ao contrário dos métodos indutivos, os métodos transdutivos não constroem um classificador para todo o espaço de entrada. Em vez disso, o seu poder de previsão é limitado exatamente aos objetos que encontra durante a fase de treino. Por conseguinte, os métodos transdutivos não têm fases distintas de treino e de teste. Uma vez que os métodos de aprendizagem supervisionada não são, por definição, fornecidos com dados não etiquetados até à fase de teste, pelo que não existem analogias claras dos algoritmos transdutivos na aprendizagem semi-supervisionada. [10]

2.5.3 Self-training

Self-training é um método que se encaixa nos algoritmos de aprendizagem semi-supervisionada e a ideia é bastante simples de se perceber.

1. Primeiro treina-se o algoritmo só com os dados etiquetados e com os melhores hiper parâmetros de forma a obter um modelo estável.
2. De seguida usa-se o modelo para fazer previsões nos dados não etiquetados, usando só as previsões com uma confiança elevada, criando "pseudo-labels" para os dados não etiquetados.
3. Estes novos dados etiquetados são adicionados ao conjunto de treino, para se obter um novo modelo para fazer previsões nos restantes dados não etiquetados.
4. Por fim repete-se este processo até os dados ficarem todos etiquetados ou acabar as iterações pretendidas.

Apesar de ser um método simples em certos problemas consegue ter uma boa capacidade de aprendizagem.

2.5.4 Co-training

Este método é parecido com o método self-training, só que consiste em ter dois ou mais modelos ao nível da classificação das labels.
No co-training, dois ou mais classificadores supervisionados são treinados iterativamente nos dados etiquetados, adicionando em cada iteração as suas previsões com maior confiança no conjunto de dados etiquetados dos outros classificadores.

Para que o co-training seja bem-sucedido é importante que as previsões dos classificadores não sejam muito correlacionadas, pois se forem, diminui o potencial de fornecer aos outros classificadores informações úteis.

2.5.5 Tri-training

Tri-training é uma extensão do método co-training, só que em vez de serem dois algoritmos classificadores são três.
Começa-se por treinar o primeiro modelo com os dados etiquetados e faz -se a previsão dos dados não etiquetados. De seguida treina-se o segundo modelo de classificação com os dados etiquetados e com os dados que o primeiro modelo previu a etiqueta fazendo a previsão para os restantes dados não etiquetados. Depois treina-se o último modelo com os dados etiquetados e com os "pseudo-labels" dos outros modelos e faz-se uma nova previsão nos restantes dados não etiquetados.
Por fim usa-se a previsão dos três modelos para fazer uma previsão final nos dados não etiquetados baixando os erros de classificação e melhorando a sua taxa de acerto.
Quando dois dos três classificadores concordam na sua previsão para um dado ponto, esse ponto de dados é passado para o outro classificador juntamente com a respetiva

etiqueta. Crucialmente, a triformação não se baseia em previsões probabilísticas de classificadores individuais, e pode assim ser aplicado a uma gama muito mais vasta de algoritmos de aprendizagem supervisionada. [10]
Dentro dos algoritmos de tri-training é normal usar outro método em cima do mesmo, ou seja, na fase de atribuição do pseudo-label é normal elaborar um ensemble dos classificadores para obter uma melhor escolha na atribuição das etiquetas.
Os classificadores de conjunto consistem em múltiplos classificadores de base, que são treinados e depois utilizados para formar previsões combinadas [10]. A forma mais simples de aprendizagem em ensemble é treinar k classificadores de base independentemente e agrega as suas previsões.

2.5.6 Multi-view learning

Multi-view é uma técnica de separação de dados para melhorar a previsão do modelo, em que consiste em treinar diferentes modelos em diferentes vistas dos dados etiquetados. O que faz com que aprenda diferentes aspetos do dataset.
Após cada etapa de treino, as previsões mais confiantes para cada vista são adicionadas ao conjunto de dados etiquetados para a outra vista. [10]
Com os dados a fluir de diferentes vistas ambos os classificadores que estão a ser treinados conseguem aprender a partir de dados fornecidos pelas duas vistas ficando um algoritmo mais genérico.

2.5.7 Transductive support vector machines

SVM é um método de treino que consiste na aprendizagem através de estatística e tem muitas vantagens como otimização global, generalização, etc.
Transductive support vector machine (TSVM) é uma variante semi-supervisionada de SVM e é utilizada em muitas aplicações, tais como classificações de cancro, classificação de anomalias mamográficas, classificação de glaucoma, recuperação de imagem, reconhecimento de categoria de navio, etc. [11]
O modelo TSVM é um modelo que só funciona bem em alguns conjuntos de dados, ou seja, é um pouco mais específico para o tipo de situação.
Os TSVM fazem previsões apenas sobre os pontos de dados que fazem parte do conjunto de treino, isto é não fazem previsões sobre dados novos e não vistos. Em vez disso, concentram-se na otimização do limite de decisão com base nos pontos de dados etiquetados e não etiquetados disponíveis.

2.5.8 Métodos baseados em Grafos

Nos últimos anos têm sido propostas redes de grafos convulsionais (GCN) para generalizar as redes neuronais convulsionais em domínios irregulares ou não-Euclidianos, em particular as redes GCN tentam lidar com grafos gerais (irregulares) onde a convolução computacional não é simples como para a modelação das relações de pixéis dos grafos de grelha [12].

Os métodos baseados em grafos, por exemplo, dependem tipicamente de uma medida de semelhança local para construir um grafo sobre todos os pontos de dados. Para aplicar tais métodos com sucesso, é importante que uma medida de semelhança local significativa possa ser concebida. Em dados de alta dimensão, tais como imagens, onde a distância Euclidiana raramente é uma boa característica do indicador da semelhança entre os pontos de dados, isto é muitas vezes difícil. [10]
Os GCN são bem adequados para este tipo de algoritmos de aprendizagem porque podem efetivamente propagar informação e etiquetas desde os nós etiquetados até aos nós não etiquetados no grafo, permitindo um melhor desempenho na tarefa de classificação.
Ao aproveitar a estrutura do grafo, os GCN são capazes de utilizar eficazmente os dados etiquetados limitados e fazer previsões para os nós não etiquetados, levando a um melhor desempenho de classificação em comparação com os modelos tradicionais de deep learning.[12]
Existem mais modelos e técnicas uteis para a realização do algoritmo visto que existe um vasto leque de opções para elaborar o algoritmo semi-supervisionado, mas o que influenciará mais será o tipo de dados em que é proposto usar o algoritmo.

2.5.9 Métodos para melhorar a eficácia da classificação das etiquetas

Existem vários métodos para melhorar a eficácia do algoritmo tais como:

- SSMBoost
- ASSEMBLE
- SemiBoost

2.5.9.1 SSMBoost

Crucialmente, o SSMBoost não atribui pseudo-etiquetas aos pontos de dados não etiquetados.
Como resultado, requer aprendizes de base semi-supervisionados para fazer uso dos dados não etiquetados e é intrinsecamente semi-supervisionado, em contraste com a maioria dos outros algoritmos que são métodos Wrapper. No entanto, o SSMBoost é incluído aqui porque constitui a base para todas as outras formas de algoritmos de boosting semi-supervisionado, que não requerem aprendizes de base semi-supervisionados. [12]

2.5.9.2 ASSEMBLE

O algoritmo ASSEMBLE, abreviatura de Adaptive Supervised Ensemble, atribui pseudo-etiquetas aos pontos de dados não etiquetados após cada iteração, e usa esses pontos de dados pseudo-etiquetados na construção do classificador seguinte, aliviando assim a necessidade de aprendizes de base semi-supervisionados.
O ASSEMBLE maximiza efetivamente a margem de classificação no espaço de funções.
O ensemble combina as previsões dos classificadores de base utilizando um esquema de votação por maioria ponderada. Os pesos são atribuídos com base na precisão de

cada classificador de base num conjunto de validação. Esta previsão do conjunto é considerada como a previsão final do algoritmo ASSEMBLE. [13]

2.5.9.3 SemiBoost

O SemiBoost utiliza o modelo de classificação boosting padrão, expressando a previsão da etiqueta final como uma combinação linear das previsões dos classificadores individuais. A sua função de custo, no entanto, é muito diferente dos métodos de boosting semi-supervisionados anteriormente descritos. [10]
O SemiBoost alarga a abordagem de boosting para incorporar dados não etiquetados no processo de aprendizagem. Começa por treinar um classificador fraco nos dados iniciais etiquetados. Em seguida, usa esse classificador para classificar os dados não etiquetados. As instâncias nos dados não etiquetados que o classificador prevê com confiança são adicionadas ao conjunto de dados etiquetados. A estas instâncias recém-etiquetadas são atribuídos pesos mais elevados nas iterações subsequentes para garantir que têm uma influência mais forte no modelo. [13]
O SemiBoost aproveita as informações adicionais presentes nos dados não etiquetados para melhorar o desempenho do modelo. Ao selecionar e etiquetar ativamente instâncias confiantes, adapta o classificador para melhor e captar os padrões subjacentes nos dados. Esta combinação de boosting e aprendizagem semi-supervisionada leva a uma maior precisão em comparação com os métodos tradicionais de aprendizagem supervisionada que dependem apenas de dados etiquetados. [13]
Existem outros métodos para melhorar a eficácia da classificação dos dados não etiquetados, sendo referido acima os três mais usados.

2.6 Tecnologias e ferramentas

Nesta secção, aprofundamos as tecnologias e ferramentas utilizadas na implementação da metodologia de aprendizagem automática proposta. A escolha das tecnologias desempenha um papel fundamental na execução bem-sucedida de qualquer projeto baseado em dados. Descrevemos as tecnologias escolhidas para a manipulação de dados, formação de modelos, avaliação e visualização, fornecendo informações sobre a sua importância e compatibilidade com os objetivos da investigação.

2.6.1 Manipulação e análise de dados

Para o pré-processamento e manipulação de dados, baseamo-nos principalmente na linguagem de programação Python devido ao seu rico ecossistema de bibliotecas adaptadas à análise de dados e à aprendizagem automática. A versatilidade, a facilidade de utilização e as extensas bibliotecas do Python, como o NumPy e o Pandas, permitem o carregamento, a limpeza e a transformação eficientes dos dados. Estas bibliotecas facilitam o manuseamento perfeito de diversos conjuntos de dados,

garantindo que os dados são preparados adequadamente para a modelação subsequente.

2.6.2 Estruturas de aprendizagem automática

O foco da nossa implementação reside na utilização de estruturas de aprendizagem automática bem estabelecidas. Especificamente, aproveitamos o poder do scikit-learn, uma biblioteca de aprendizagem automática amplamente adotada na comunidade Python. O scikit-learn fornece uma série de ferramentas para pré-processamento de dados, seleção de características, formação de modelos e avaliação. As suas APIs fáceis de utilizar e a sua extensa documentação fazem dela a escolha ideal para a implementação de algoritmos de aprendizagem automática.

2.6.2.1 Máquinas de suporte vetorial (SVM)

Na nossa metodologia, utilizamos o algoritmo máquina de suporte vetorial (SVM) devido ao seu historial comprovado em tarefas de classificação. A implementação do SVM no scikit-learn oferece flexibilidade na escolha das funções de kernel, e a sua capacidade de lidar com dados de elevada dimensão aumenta a sua aplicabilidade ao nosso problema de investigação.

2.6.2.2 Random Forest

O algoritmo Random Forest é outro componente-chave da nossa metodologia, utilizado pela sua abordagem baseada em conjuntos que melhora a robustez e o desempenho do modelo. A implementação do Random Forest do Scikit-learn permite-nos criar e treinar uma floresta de árvores de decisão de forma eficiente, aproveitando tanto o ensacamento como a aleatoriedade das características.

2.6.3 Visualização e relatórios

A visualização dos resultados e dos conhecimentos é crucial para uma comunicação e compreensão efetivas. Para isso, utilizamos a biblioteca Matplotlib, que oferece recursos gráficos versáteis. A integração da Matplotlib na nossa implementação permite-nos criar gráficos informativos, diagramas e gráficos que mostram o desempenho do modelo, as distribuições de dados e as tendências.

2.6.4 Controlo de versões e colaboração

Para garantir uma colaboração e um controlo de versões contínuos ao longo do projeto, utilizamos o Git e o GitHub. Estas ferramentas permitem aos membros da equipa, ao autor deste projeto e orientador trabalhar em simultâneo, acompanhar as alterações e gerir a base de código do projeto de forma eficiente. A utilização do controlo de versões promove a transparência, reduz o risco de erros e facilita a integração das contribuições.

2.6.5 Justificação de escolha

A seleção destas tecnologias baseia-se na sua aceitação pela indústria, na disponibilidade de documentação e no seu alinhamento com os objetivos da investigação. O ecossistema de bibliotecas Python abrange todas as fases do fluxo de trabalho de aprendizagem automática, desde o pré-processamento de dados até à implementação do modelo. A fiabilidade do Scikit-learn e a extensa coleção de algoritmos permitem-nos aplicar e comparar vários modelos de forma eficaz. Além disso, as ferramentas de visualização, o controlo de versões e a colaboração fornecidos pelo Matplotlib, Git e GitHub asseguram a robustez e a capacidade de manutenção do projeto.
Ao adotar estas tecnologias, aproveitamos o poder de um conjunto de ferramentas completo que apoia a implementação prática da metodologia proposta. A sua integração sinérgica aumenta a nossa capacidade de obter conhecimentos significativos e de promover uma compreensão mais profunda do problema de investigação em causa.

3 Análise de valor

O valor e a inovação são alguns dos aspetos mais importantes ao avaliar se um o negócio é bem ou malsucedido.
O processo de tomada de decisão antes de investir num novo produto pode ser muito difícil para a maioria empresas e para ajudar neste processo, pode ser feita uma análise de valor.
A análise de valor examina as vantagens e os custos para reduzir as despesas sem diminuir a qualidade do produto final. Por outras palavras, o seu objetivo é atingir o máximo possível valor de uma forma sustentável.
Este capítulo descreve a análise de valor deste projeto para avaliar o seu valor potencial tomando em consideração o contexto do problema a ser resolvido.
Com a execução deste projeto é pretendido elaborar uma plataforma com um algoritmo semi-supervisionado que receba um conjunto de dados e retorne um resultado satisfatório, isto é, que apresente resultados de pelo menos 2% nas previsões do algoritmo.

3.1 Processo de inovação

O processo de inovação divide-se em três fases: front end, new product development e commercialization. A primeira fase baseia-se num trabalho de pesquisa de oportunidades, seguindo-se uma fase de análise, onde se fazem algumas avaliações à potencialidade do projeto a nível técnico e de marketing. Por fim, a terceira fase, foca-se num âmbito mais comercial. [14]

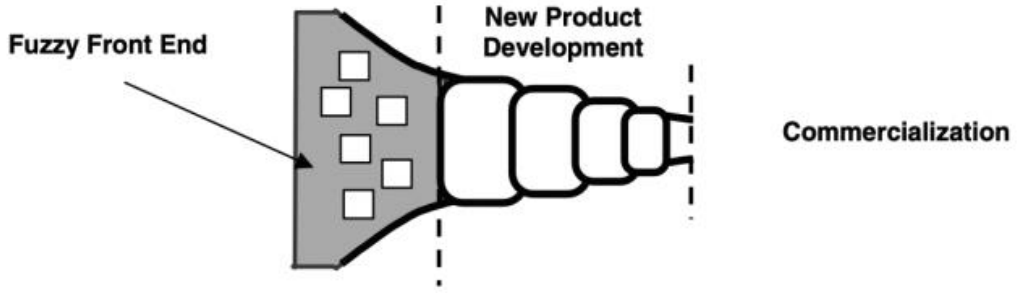

Figura 6 – Fases do processo de inovação

As atenções estão cada vez mais concentradas em as atividades de front-end que precedem este processo formal e estruturado em ordem para aumentar o valor, montante e probabilidade de sucesso de conceitos com elevado lucro entrar no desenvolvimento e comercialização de produtos. [15]

3.2 Desenvolvimento do Novo Conceito (New Concept Development)

O Desenvolvimento do novo conceito refere-se ao processo de trazer um novo produto ou ideia para um mercado seguindo uma metodologia para chegar a uma solução mais elaborada.

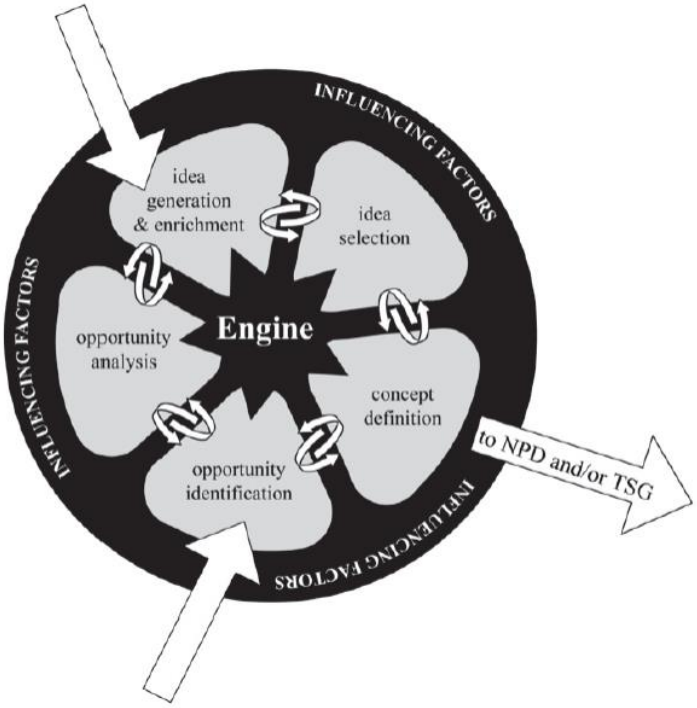

Figura 7 - Modelo de New Concept Development [14]

Este modelo baseia-se em 5 conceitos fundamentais: ideia, seleção da ideia, definição de conceitos, identificação de oportunidade e análise de oportunidade.Este modelo, que se encontra ilustrado na figura 4, consiste em três partes cruciais [15]:

• Motor: representa a liderança, cultura e estratégias de negócio da organização que estimula os cinco elementos-chave controláveis pela organização;

• Elementos-Chave: Cinto elementos de atividade controláveis do FFE, nomeadamente, identificação da oportunidade, análise da oportunidade, geração de ideias, seleções de ideias e por último a definição do conceito;

• Fatores de Influência: Estes fatores podem ser as capacidades da organização, o mundo exterior (leis, política governamental, concorrentes, políticas económicas, etc.) e as ciências capacitadoras (internas ou externas) que possam estar envolvidas.

Estes fatores influenciam todo o processo de inovação e não podem ser controlados pelas organizações.

3.2.1 Identificação de oportunidade

A identificação de oportunidades como objetivo de identificar o que a empresa quer perseguir. É normalmente impulsionada por objetivos empresariais que podem levar a uma completa nova direção ou uma atualização para um produto existente.
"A identificação global de oportunidades define o mercado ou a arena tecnológica em que a empresa pode querer participar" [14].
O projeto desta tese vai ao encontro do desenvolvimento de um algoritmo semi-supervisionado que aproveita o facto de os dados etiquetados sejam dispendiosos e difíceis de obter, usando os dados etiquetados para achar etiquetas para os outros dados não etiquetados. Garante também em grande parte dos casos uma melhoria de precisão do algoritmo.

3.2.2 Análise de oportunidade

A análise de oportunidade tem o objetivo verificar se esta solução deve ser concebida dependendo de vários fatores.
Para esta análise foi elaborado uma análise SWOT. A analise SWOT tem 4 quadrantes importantes em que foca a sua análise [16]:

- **S(Forças) Pontos Fortes** – Atributos positivos tangíveis e internos que ajudem a ultrapassar algum tipo de ameaça.
- **W (Fraquezas) Pontos Fracos** – Atributos internos que estão sobre o controlo da organização, mas que prejudicam a sua capacidade de atingir a meta.
- **O (Oportunidades)** - Fatores atrativos e externos que são a razão para que uma organização exista e se desenvolva.
- **A (Ameaças)** – Fatores externos, fora do controle de uma organização que poderiam colocar em risco a missão ou operação da organização.

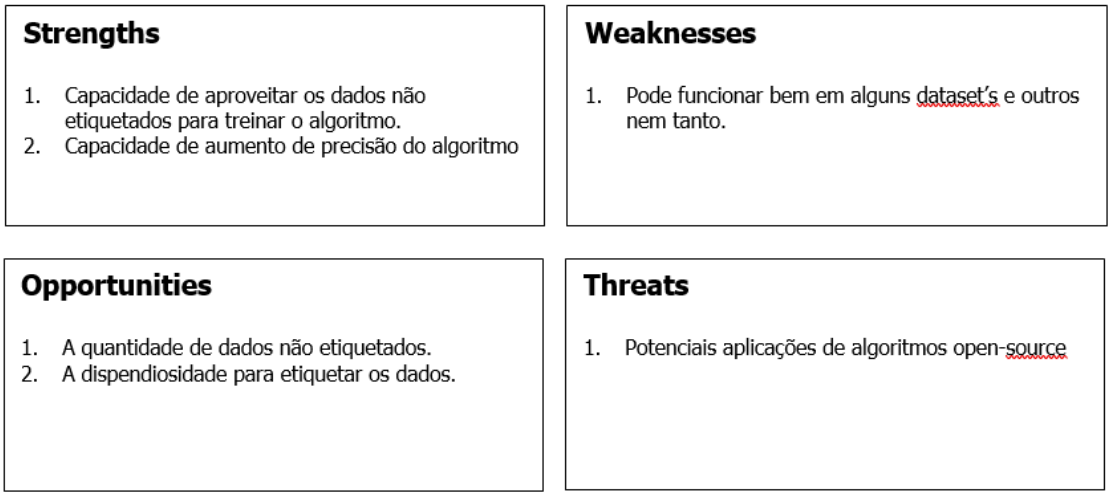

Figura 8 – Análise SWOT

3.2.3 Geração de ideias

A geração de ideias é um passo importante que faz com que a ideia original amadureça para um projeto mais concreto dependendo das necessidades do cliente.
Isto representa um processo evolutivo em que as ideias são construídas, demolidos, combinados, remodelados, modificados e atualizados. A ideia pode passar por muitas iterações e alterações à medida que é examinado, estudado, discutido e desenvolvido [17].
A geração da ideia apareceu primeiro com a identificação do problema em que pensado e com brainstorming foi originar a ideia mais concreta para resolver o problema. Progrediu ainda mais depois ao fazer este tipo de análises de mercado, de oportunidade e der verificar soluções mais existentes.
Relativamente ao projeto surgiram várias ideias, mas apresento as duas mais impactantes:

- Criação do algoritmo do zero inovando a sua forma de implementação e criando uma plataforma para o cliente.
- Começar com técnicas já existentes para o algoritmo e adaptá-lo à solução pretendida.

3.3 Valor da solução

Nesta secção é apresentada que tipo de valor poderá trazer para um cliente, utilizador final ou entidade. Para isso são usadas ferramentas para analisar a solução apresentada.

3.3.1 Valor

Normalmente valorizam o valor como significados diferentes dependendo se se refere ao produtor ou ao cliente. O produtor pode obter valor de clientes leais, uma vez que estes são benéficos para o negócio de várias maneiras. O valor dos clientes, por outro lado, tende a decorrer de condições como a qualidade de um produto/serviço que compraram, o esforço que fizeram para o adquirir, as interações sociais vividas com o produtor, etc.
Neste projeto considera-se o maior benefício a taxa de acerto potencial do algoritmo sem precisar de ter os dados todos etiquetados, podendo tirar vantagem de existir muitos dados não etiquetados.

3.3.2 Valor para o cliente

O valor do cliente pode ser visto como o entendimento geral do que os clientes realmente a procura e o valor de um produto ou serviço, e o que pode ser feito pelo produtor para satisfazer as necessidades dos consumidores para alcançar uma vantagem competitiva sobre outras empresas. [18]

Assim, o valor para o cliente, surge da comparação que o mesmo faz com outros produtos semelhantes, no sentido de adquirir aquele com mais valor para as suas necessidades. Se o valor que as organizações atribuírem àquilo que oferecem, não corresponder ao valor que efetivamente transmite para o cliente, a expetativa dos clientes não é concretizada, levando uma desilusão da parte do mesmo.

3.3.3 Valor Percecionado

De acordo com Graf e Maas (2008), o valor percecionado pelo cliente "é conceptualizado como um tradeoff entre benefícios e sacrifícios, com foco nas características de desempenho concreto dos produtos/serviços".
De acordo com o presente projeto pretende-se criar uma solução que traga valor para o cliente, desenvolvendo o algoritmo como boa taxa de acerto e aproveitando os dados não etiquetados pode ser uma mais-valia para o cliente por não ter de gastar recursos em etiquetar dos dados e obter um bom resultado.

3.3.4 Proposta de valor

A proposta de valor pode ser definida como os benefícios que são oferecidos por uma empresa aos seus clientes.
Uma proposta de valor serve um papel integral na criação e eventual venda de qualquer novo produto e/ou serviço. Resume-se essencialmente a um referendo sobre a razão pela qual um cliente deve comprar um produto oferecido por uma organização, que benefícios traz, que dores alivia e o que pode o cliente esperar obter com o pacote completo.
Para melhor avaliar a proposta de valor de uma empresa para o seu segmento de clientes, um gráfico ferramenta denominada Value Proposition Canvas, que foi proposta por Osterwalder, pode ser utilizada. [19]

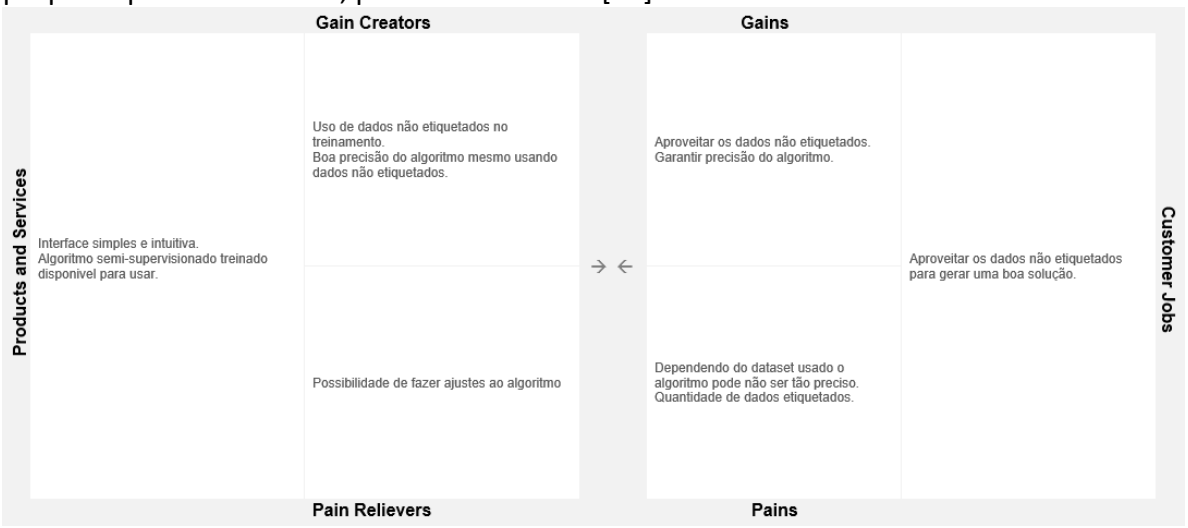

Figura 9 – Canvas – Proposta de valor

3.3.5 Método Multicritério Analytic Hierarchy Process

Para se efetuar uma boa seleção de ideias é um passo importante para assegurar o sucesso do projeto e para isso o autor recorreu o método de apoio à decisão Analytic Hierarchy Process (AHP).
Este método, desenvolvido por Saaty, tem como objetivo quantificar prioridades relativas para um determinado conjunto de alternativas numa dada escala. Esta escala tem por base a opinião e a intuição do decisor [20].
Os critérios decisivos que vão ser usados são:

- **Tempo de desenvolvimento**: Este critério diz respeito ao tempo de desenvolvimento do sistema. Visto que projeto está a ser realizado num contexto académico, existem prazos que devem ser cumpridos. Para este efeito, considera-se uma alternativa superior a outra se o tempo despendido na fase de implementação for menor;
- **Desempenho:** Este critério foca-se em questões de desempenho do algoritmo, visto que é proposto que o algoritmo tenha um certo nível de taxa de acerto este critério é muito importante.
- **Complexidade**: Este critério compreende a dificuldade associada à implementação dos componentes de cada uma da alternativa como também os custos ligados aos processos de manutenção.

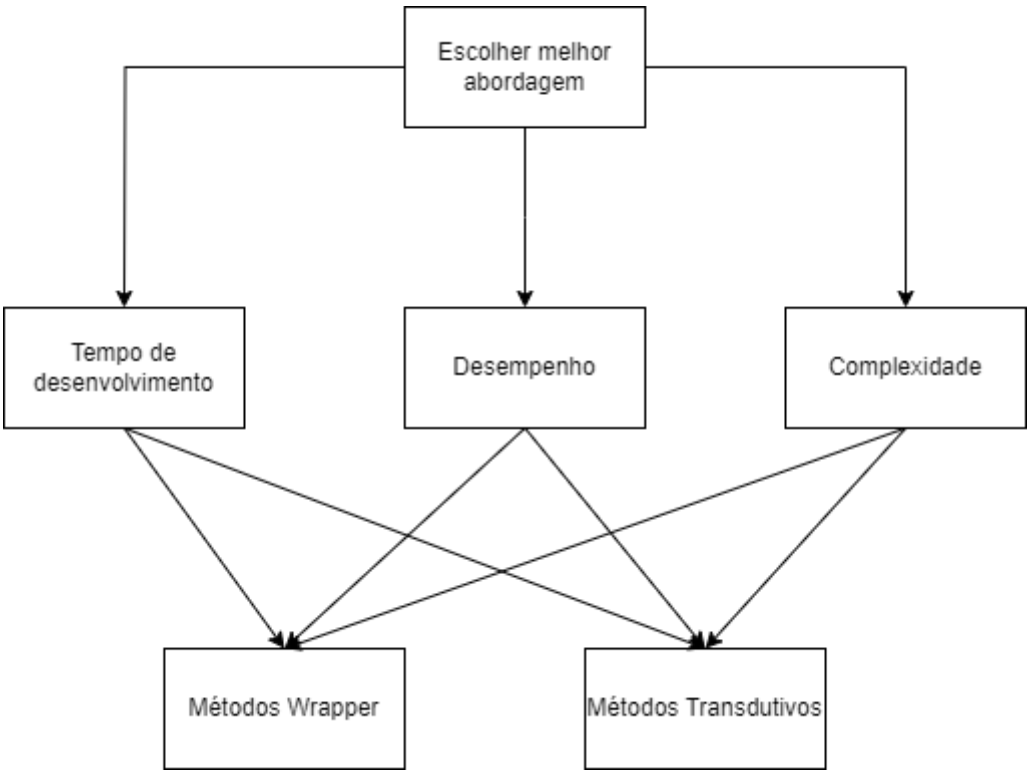

Figura 10 – Estrutura hierárquica do AHP

De seguida é necessário definir as prioridades para cada critério definido previamente.

Tabela 3 - Tabela Comparativa de Critérios do Método AHP

Valor da entrada	Interpretação
1	j e k são igualmente importantes
3	j é ligeiramente mais importante que k
5	j é mais importante que k
7	j é fortemente mais importante que k
9	j é absolutamente mais importante que k
2,4,6,8	valores intermédios

Tabela 4 – Prioridade relativa de cada critério

	Tempo	**Desempenho**	**Complexidade**	**Prioridade Relativa**
Tempo	*1*	4	5	0,67
Desempenho	1/4	1	3	0,23
Complexidade	1/5	1/3	1	0,1

Na tabela 4 está representado a importância dos vários critérios usados em que o que tem mais importância é o tempo limitado do projeto, de seguida é necessário garantir um certo nível de desempenho e por fim a complexidade do algoritmo.
Depois foram calculadas as várias prioridades relativas das alternativas para cada critério, que estão referidas nas tabelas 5,6 e 7.

Tabela 5 - Prioridade relativa das alternativas segundo o critério - Tempo de desenvolvimento

	Inovação do processo	**Usar técnicas existentes**	**Prioridade Relativa**
Métodos Transdutivos	*1*	1/7	0,125
Métodos wrapper	7	1	0,875

Tabela 6 - Prioridade relativa das alternativas segundo o critério - Desempenho

	Inovação do processo	Usar técnicas existentes	Prioridade Relativa
Métodos Transdutivos	*1*	1/2	0,37
Métodos wrapper	2	1	0,67

Tabela 7 - Prioridade relativa das alternativas segundo o critério - Complexidade

	Inovação do processo	Usar técnicas existentes	Prioridade Relativa
Métodos Transdutivos	*1*	3	0,75
Métodos wrapper	1/3	1	0,25

Tabela 8 – Prioridades relativas das soluções

	Tempo de desenvovimento	Desempenho	Complexidade	Prioridade Relativa
Métodos Transdutivos	*0,125*	0,37	0,75	0,415
Métodos wrapper	0,875	0,67	0,25	0,59

A partir da tabela 8 consegue-se perceber a partir da prioridade relativa que usar a metodologia wrapper de algoritmos semi-supervisionado é uma melhor escolha pelo o tempo limitado do projeto e pelo facto de se conseguir bons resultados, enquanto criar um algoritmo transdutivo seria mais complexo e iria demorar mais tempo e não garante melhores resultados.

4 Engenharia de requisitos

O presente capítulo tem como objetivo descrever tecnicamente a solução proposta no sentido de contribuir para a resolução do problema descrito.
A disciplina de engenharia de requisitos permite identificar, analisar as necessidades e restrições de determinado software a desenvolver, sendo um importante artefacto de análise, para a implementação de qualquer software.

4.1 Requisitos funcionais

Os requisitos funcionais descrevem quais são as funcionalidades do sistema e o seu comportamento. Eles podem ser representados através de um diagrama de caso de uso e diagramas de sequência do sistema usando Unified Modeling Language (UML).
A implementação deste algoritmo tem como requisitos funcionais:

- RQ1: Carregar os dados
- RQ2: Dividir os dados em conjunto de treino, teste e de validação
- RQ3: Treinar o algoritmo com os dados de treino
- RQ4: Avaliar o modelo gerado pelo algoritmo com os dados de teste
- RQ5: Otimizar os hyper-parâmetros do algoritmo com os dados de validação
- RQ6: Transferir os dados etiquetados
- RQ7: Usar o modelo com novos dados.

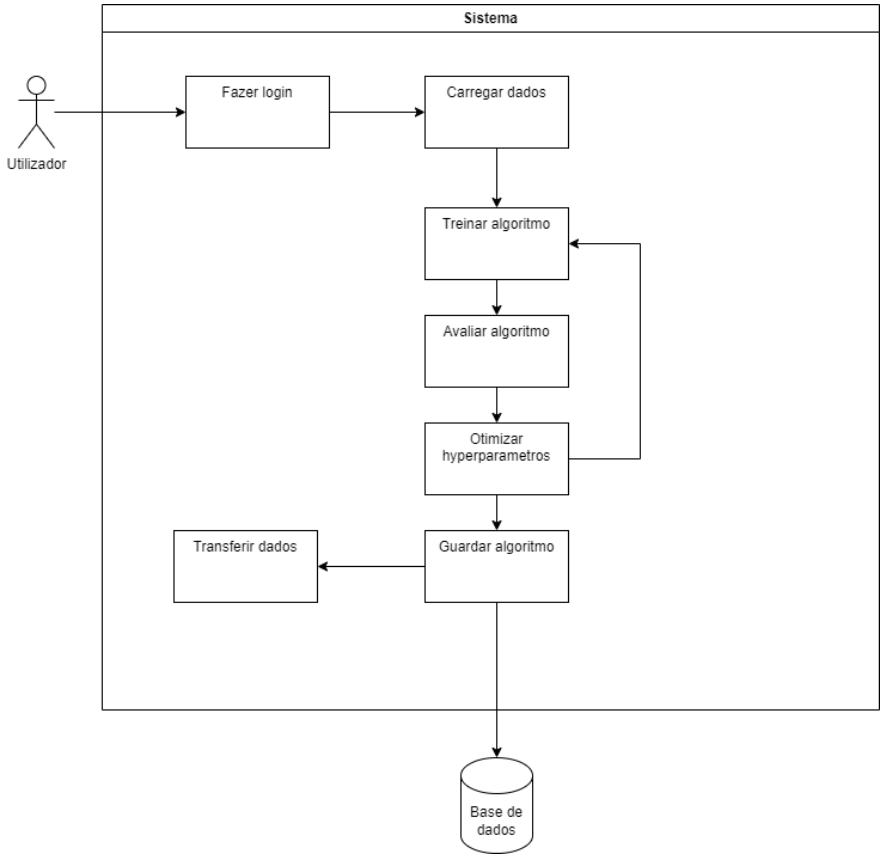

Figura 11 – Diagrama de casos de uso

O diagrama de casos de uso da figura 13 demonstra os requisitos funcionais necessários para a implementação do software.

RQ1: Carregar os dados
Este requisito envolve a etapa inicial de carregamento do conjunto de dados no sistema. O sistema tem de ser capaz de carregar os dados relevantes a partir de uma localização especificada, que servirá de base para a análise e processamento subsequentes.

RQ2: Dividir os dados em conjunto de treino, teste e de validação
Para garantir um processo de treino eficaz, o sistema deve dividir o conjunto de dados carregado em dois subconjuntos distintos: o conjunto de treino e o conjunto de validação. Esta separação permite treinar o modelo num subconjunto e validar o seu desempenho no outro, ajudando a evitar o sobre-ajustamento do modelo aos dados de treino e a avaliar as capacidades de generalização do modelo.

RQ3: Treinar o algoritmo com os dados de treino
Este requisito implica o treino do algoritmo de aprendizagem automática escolhido com o conjunto de dados de treino. O sistema deve facilitar o processo de treino do

modelo, para aprender com os dados de treino fornecidos e gerar um modelo preditivo.

RQ4: Avaliar o modelo gerado pelo algoritmo com os dados de teste
Depois de treinar o modelo, é crucial avaliar o seu desempenho utilizando o conjunto de dados de teste que foi previamente reservado. Esta etapa fornece informações sobre a capacidade do modelo de generalizar para dados novos e não vistos. As métricas de desempenho como a exatidão, a precisão, a recuperação e a pontuação F1 podem ser utilizadas para avaliar a qualidade do modelo.

RQ5: Otimizar os hyper-parâmetros do algoritmo com os dados de validação
Para afinar o processo de treino e otimizar o desempenho do modelo, este requisito envolve a exploração e o ajuste dos hiperparâmetros do algoritmo escolhido. Os hiperparâmetros são definições de configuração que não são aprendidas durante o treino, mas que afetam o comportamento do modelo. A experimentação de diferentes valores para estes hiperparâmetros pode levar a melhores resultados do modelo.

RQ6: Transferir os dados etiquetados
O sistema permite o download do csv que foi usado para treinar o algoritmo com os dados que foram etiquetados durante o treino do mesmo, para que possa usar esses dados para outro uso.

RQ7: Usar dados novos
Este requisito representa o carregamento de novos dados para o modelo que foi usado num treino prévio.
Estes requisitos funcionais formam coletivamente a base do trabalho da dissertação, demonstrando as várias fases envolvidas na construção, formação, avaliação e otimização de modelos de aprendizagem automática utilizando um conjunto de dados. Ao abordar cada requisito, o sistema mostra a abordagem sistemática que se optou para atingir os seus objetivos de investigação.

4.1.1 Diagramas de funcionamento

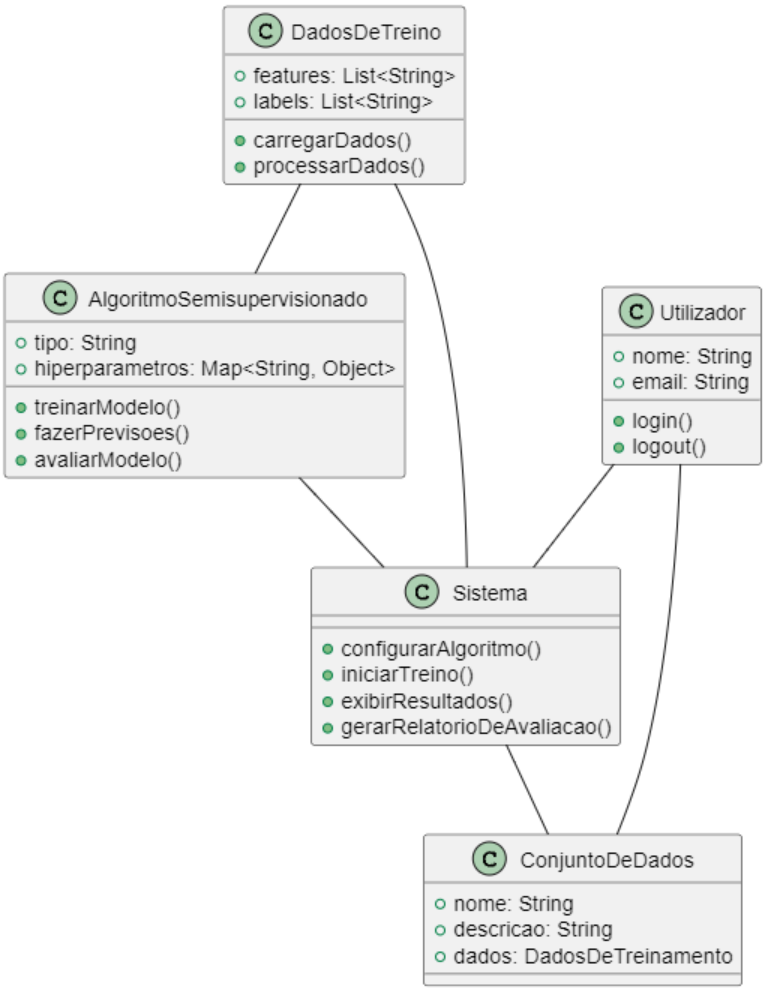

Figura 12 – Diagrama de classes

O diagrama de classe apresenta as funcionalidades implementadas na construção do algoritmo e suas dependências como demonstrado na figura 16.
O sistema recebe do utilizador pedidos para executar o algoritmo com o conjunto de dados que o utilizador pretende enviar, em que neste caso é um “csv”. De seguida requisita o processamento dos dados na classe “DadosDeTreino”, onde é feito o pré-processamento dos dados e alguns ajustes para que se possa correr o algoritmo.
Com os dados de treino ajustados, o sistema chama o algoritmo para iniciar o treino, que por sua vez começa pela parametrização, de seguida o treino do algoritmo e for fim o resultado dele é envidado para o front-end do sistema.

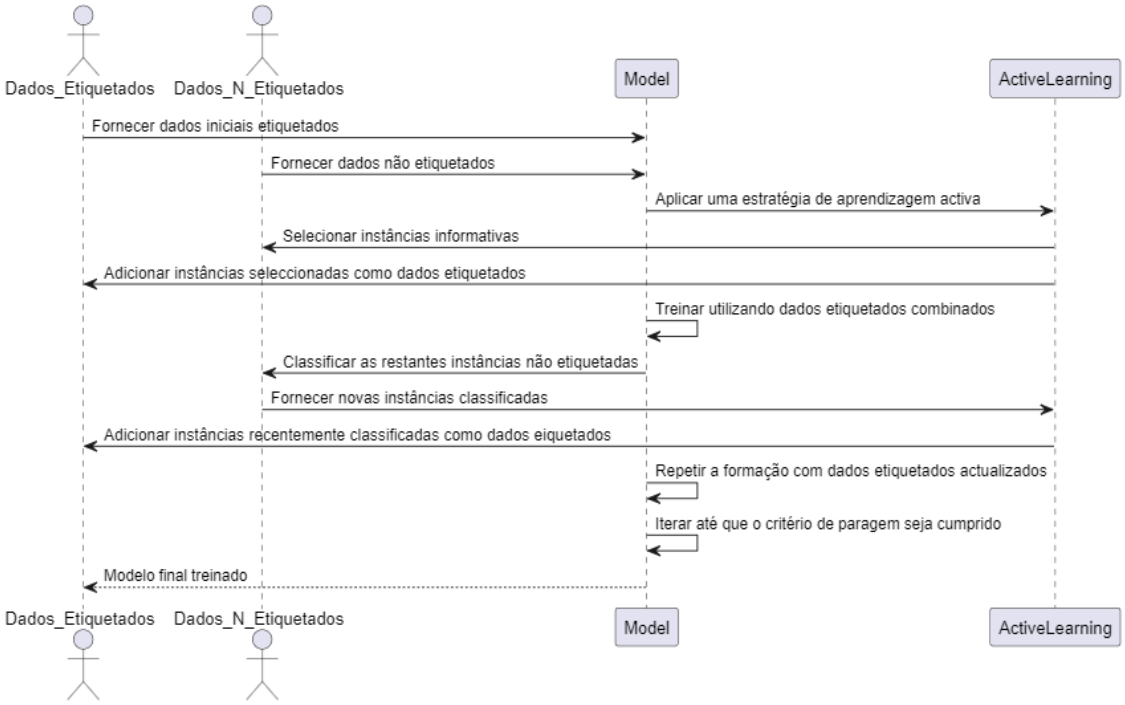

Figura 13 – Diagrama de sequência representando o treino do algoritmo semi-supervisionado

O diagrama de sequência, representado na figura 13, demonstra todas as etapas que o algoritmo realiza para chegar à solução e as estratégias usadas no algoritmo, como aprendizagem ativa, como é feito o treino e o retorno do algoritmo.

4.2 Requisitos não funcionais

Esta subsecção apresenta os requisitos não-funcionais do sistema, que normalmente consistem em algumas restrições que devem ser tomadas em consideração na conceção e implementação de uma solução de software.
A classificação Funcionalidade, Usabilidade, Fiabilidade, Desempenho, Suportabilidade (FURPS+) é utilizado para documentar os requisitos não funcionais do projeto.

- **Desempenho –** A aplicação para dados pré-treinados oferece um desempenho rápido comparado com a parte de treinar o algoritmo que pode demorar mais tempo.
- **Segurança -** Irá ter as regras de segurança para criação de utilizadores, os procedimentos exigidos para a utilização de palavras-passe, a necessidade de criptografia e questões relacionadas para garantir a proteção dos dados.
- **Usabilidade –** A aplicação será relativamente fácil de usar com tradução de linguagem para ser usado por todos os tipos de utilizadores.
- **Disponibilidade -** Com este requisito pretende-se que sejam implementadas metodologias que garantam a qualidade de performance da solução mesmo nos casos em que a sua taxa de utilização seja elevada.

4.3 Atores

Os atores nesta aplicação serão o utilizador e possivelmente um administrador caso ocorra algum problema para dar suporte.

4.4 Armazenamento

Depois do algoritmo treinado é necessário manter um registo de versões do algoritmo assim como guardar o seu modelo para ser reutilizado, para isso, uma boa opção é usar o MLflow.
O MLflow é uma plataforma de gestão do ciclo de vida de Machine Learning que permite o rastreamento e gestão de experiências, registo de modelos, controle de versão e implantação de modelos. Fornece uma estrutura para armazenar e organizar dados de treino, bem como meta dados associados, como hiperparâmetros, métricas de desempenho e artefactos do modelo.

5 Implementação

Neste capítulo, o foco passa dos conceitos teóricos para a aplicação prática. Com base nos fundamentos estabelecidos nos capítulos anteriores, esta secção detalha o processo passo-a-passo de tradução dos objetivos e métodos de investigação em resultados tangíveis.
O processo de implementação envolve uma série de tarefas, cada uma concebida para responder a questões de investigação específicas e alcançar resultados práticos. Estas tarefas englobam o carregamento de dados, o pré-processamento, a utilização de algoritmos de Machine Learning, a avaliação de modelos, a afinação de hiperparâmetros e o aproveitamento de modelos pré-treinados. Ao abordar sistematicamente cada requisito funcional, este capítulo demonstra como a estrutura teórica se transforma em conhecimentos acionáveis através da codificação prática.
Ao longo deste capítulo, os trechos de código e as explicações fornecem uma compreensão clara do processo de codificação, ajudando a entender particularidades da implementação da aprendizagem automática no mundo real. No final deste capítulo, ter-se-á uma compreensão abrangente das etapas práticas necessárias para transformar os conceitos de aprendizagem automática semi-supervisionada em ferramentas funcionais para análise de dados e tomada de decisões.

5.1 Carregamento e preparação de dados

Nesta secção, será explicado como carregar o conjunto de dados para o ambiente de programação escolhido e executar os passos iniciais de pré-processamento, tais como o tratamento de valores em falta e a codificação de variáveis categóricas.

5.1.1 Fontes e aquisição de dados

O primeiro passo fundamental é a aquisição do conjunto de dados que constitui a base da nossa análise. Para este efeito, selecionamos cinco conjuntos de dados disponíveis em formato CSV. Estes conjuntos de dados captam um conjunto abrangente de atributos relacionados com doenças, outro relacionado com deteção de fraude de cartões de crédito, classificação de vinho e fruta, o que se tornam candidatos ideais para o estudo.
Não foram usados conjuntos de dados com imagens devido ao tempo de processamento que o algoritmo precisaria para classificar as imagens, mas foi feito um teste para demonstrar que o algoritmo funciona com imagens.

5.1.2 Carregamento de dados e inspeção inicial

Para facilitar a manipulação e análise de dados, aproveitamos o poder da biblioteca Pandas em Python. A Pandas fornece uma interface eficiente e intuitiva para carregar dados de várias fontes, incluindo ficheiros CSV. Utilizamos a função pd.read_csv() para carregar o conjunto de dados num Pandas DataFrame, uma estrutura de dados versátil que nos permite efetuar uma vasta gama de operações sobre dados.

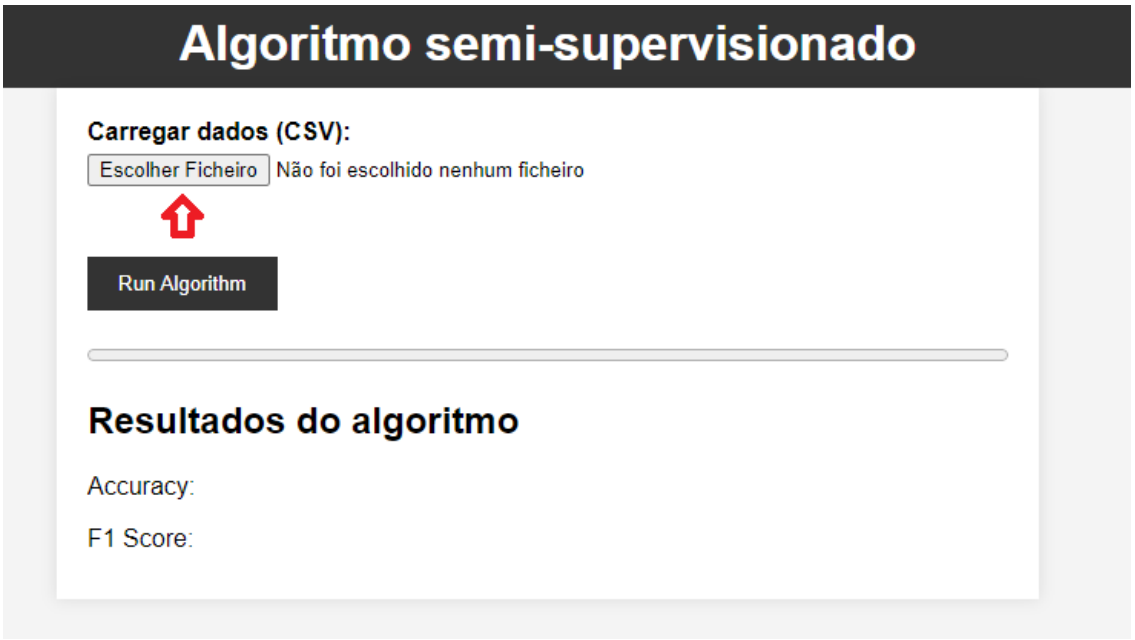

Figura 14 – Carregamento de dados na aplicação web

O carregamento do CSV na aplicação web assume que o atributo objectivo que se pretende prever estará na última coluna, sendo uma estrutura necessária para conseguir correr o algoritmo sem problemas.
Após o carregamento do conjunto de dados, é efetuada uma inspeção inicial para obter uma compreensão preliminar da sua estrutura e conteúdo. Avaliamos as dimensões do DataFrame utilizando o atributo.shape e examinamos os nomes das colunas com .columns. Este passo permite-nos verificar a exatidão do carregamento de dados e garantir a consistência com os atributos esperados.

5.1.3 Pré-processamento e limpeza de dados

Os conjuntos de dados do mundo real contêm muitas vezes imperfeições, valores em falta e valores atípicos que podem dificultar o treino do modelo. Para enfrentar estes desafios, embarcamos num processo abrangente de pré-processamento e limpeza de dados. Identificamos e tratamos os valores em falta, removemos os duplicados e resolvemos quaisquer inconsistências nos dados. Por vezes como em algumas linhas faltava um só valor numa coluna, este foi preenchido com a média, não influenciando o resultado esperado.

Por exemplo, utilizou-se a função Pandas .isna() para detetar valores em falta e a função .drop_duplicates() para eliminar linhas duplicadas. Além disso, utilizei a classe SimpleImputer do scikit-learn para preencher os valores em falta com estimativas adequadas, garantindo a integridade dos dados sem comprometer a qualidade geral do conjunto de dados.

5.1.3.1 Codificação de características (Feature Encoding)

Os nossos conjuntos de dados podem incluir variáveis categóricas que requerem codificação para a forma numérica antes de serem introduzidas nos modelos de aprendizagem automática. Foi utilizado técnicas como a codificação de uma só vez utilizando a função pd.get_dummies() para converter atributos categóricos numa representação binária adequada ao consumo do algoritmo.

5.1.3.2 Escalonamento de dados

Os atributos numéricos podem apresentar escalas variáveis, o que pode afetar o desempenho de determinados algoritmos. Resolvemos essa preocupação empregando técnicas de escalonamento de características. Uma abordagem comum é usar a classe StandardScaler do scikit-learn para padronizar as características, garantindo que elas tenham uma média de 0 e um desvio padrão de 1.

5.1.3.3 Divisão de dados

Um passo fundamental na aprendizagem automática envolve a divisão do conjunto de dados em subconjuntos de treino e de teste. Esta separação garante que os modelos são avaliados em dados não vistos, fornecendo uma avaliação realista das suas capacidades de generalização. Utilizamos a função train_test_split() do scikit-learn para efetuar esta divisão, mantendo a distribuição das classes para evitar enviesamentos nas análises subsequentes.

5.2 Seleção de dados para desenvolvimento de modelos

Nesta secção vai ser aprofundado o processo crítico de partição de dados no contexto do estudo de algoritmos semi-supervisionados. Ao contrário das abordagens convencionais, este estudo envolve uma divisão única do conjunto de dados - 10% de instâncias etiquetadas e o restante não etiquetado, isto para efeitos de teste tirando partido da etiquetagem simulando casos reais. Este particionamento não convencional desempenha um papel fundamental na implementação do algoritmo de co-treino, permitindo o seu paradigma de aprendizagem único. Vamos aprofundar as motivações por detrás deste particionamento e o seu alinhamento com este estudo de co-treino. Também foram realizados outros testes com o uso de 20% e 30% dos dados com etiquetas para verificar como o algoritmo se comportava.

5.2.1 Estratégias de partição de dados

A estratégia de partição de dados que foi utilizada está adaptada à estrutura de co-treino que se está a investigar. Por definição, atribuiu apenas 10% do conjunto de dados como instâncias marcadas. Esta escassez de dados etiquetados é fundamental para a filosofia de co-treino, uma vez que simula cenários em que os dados etiquetados são limitados e procura ativamente aproveitar um conjunto maior de instâncias não rotuladas. Esta abordagem promove um processo de aprendizagem mais robusto e incentiva o modelo a adaptar-se e a melhorar iterativamente.

5.2.1.1 Divisão aleatória

A divisão aleatória consiste em baralhar aleatoriamente o conjunto de dados e atribuir uma determinada percentagem de instâncias a cada subconjunto. Por exemplo, uma prática comum é uma divisão 70-20-10, em que 70% dos dados são utilizados para treino, 20% para teste e 10% para validação. Esta abordagem é simples e minimiza o enviesamento, tornando-a adequada para vários cenários.

```
# Randomly split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figura 15 – Divisão dos dados entre treino e teste

5.2.1.2 Validação cruzada

A validação cruzada envolve a partição do conjunto de dados em vários subconjuntos ou "dobras". O modelo é treinado numa combinação destas dobras e avaliado na dobra restante iterativamente. Esta técnica ajuda a utilizar melhor os dados disponíveis, fornecendo uma estimativa mais robusta do desempenho do modelo.

```
# Perform k-fold cross-validation
scores = cross_val_score(model, X_train, y_train, cv=5)
average_accuracy = scores.mean()
```

Figura 16 – Código da Validação cruzada

5.2.2 O uso dos dados não etiquetados

A maior parte do conjunto de dados consiste em instâncias não etiquetadas, atuando efetivamente como um recurso "inexplorado". Estas instâncias não etiquetadas são vitais no treino do conjunto de dados, uma vez que representam o potencial do algoritmo para descobrir padrões e relações ocultas, melhorando o seu desempenho. Esta abordagem de seleção alinha-se com o princípio central da metodologia de co-treino: aproveitar múltiplos pontos de vista ou classificadores para melhorar o processo de aprendizagem, particularmente em cenários com dados etiquetados limitados.

5.2.3 O papel da divisão de dados neste estudo

Para este estudo, foi adotado uma estratégia de divisão aleatória para dividir o conjunto de dados em conjuntos de treino e de teste. Assegurando uma semente aleatória fixa (random_state) para garantir a reprodutibilidade. Além disso, incorporou-se técnicas de validação cruzada durante a seleção de modelos e a afinação de hiperparâmetros para garantir que os modelos são fiáveis e capazes de generalizar para dados não vistos.
Ao reservar apenas 10% do conjunto de dados para instâncias etiquetadas para o teste do algoritmo, está-se a orientar o estudo para a exploração da aprendizagem semi-supervisionada, aprendizagem ativa e melhoria iterativa. Esta estratégia de particionamento não convencional alinha-se harmoniosamente com os objetivos principais do estudo, permitindo que o algoritmo de co-treino prospere em cenários com dados etiquetados limitados.

5.3 Afinação de hiperparâmetros

Nesta secção, é discutido o processo de afinação de hiperparâmetros para otimizar o desempenho do nosso algoritmo de co-treino. Os hiperparâmetros são parâmetros definidos antes do treino e que afetam o processo de aprendizagem e a complexidade do modelo. O ajuste dos hiperparâmetros ajuda a melhorar a capacidade do modelo de generalizar para dados não vistos e produzir melhores resultados.

5.3.1 Importância dos hiperparâmetros

Os hiperparâmetros desempenham um papel crucial no desempenho dos algoritmos de aprendizagem automática. Diferentes valores de hiperparâmetros podem levar a diferentes comportamentos do modelo, afetando fatores como a velocidade de convergência, a complexidade do modelo e a capacidade de generalização. Por conseguinte, identificar os hiperparâmetros ideais é essencial para obter os melhores resultados possíveis.

5.3.2 Pesquisa em grelha para afinação de hiperparâmetros

A pesquisa em grelha é uma técnica comum para a afinação de hiperparâmetros. Envolve a especificação de uma gama de valores possíveis para cada hiperparâmetro e a avaliação exaustiva do desempenho do modelo utilizando diferentes combinações desses valores. A combinação que produz o melhor desempenho num conjunto de validação é escolhida como o conjunto ótimo de hiperparâmetros.

```python
param_space = {
    'svm': {
        'C': [0.1, 1, 10, 100],  # Adjust as needed
        'kernel': ['linear','rbf'],
    },
    'random_forest': {
        'n_estimators': [10, 50, 100, 200],
        'max_depth': [None, 10, 20, 30, 40],
    },
}
```

Figura 17 – Parâmetros usados na pesquisa de hyperparâmetros dos dois classificadores

Na figura 17, foi definido uma grelha de possíveis valores de hiperparâmetros para o classificador SVM (C, kernel e gamma) e Ramdom Forest (n_estimators, max_deph).
Em seguida, foi efetuado uma pesquisa de grelha utilizando a validação cruzada (cv=5) para encontrar a melhor combinação de hiperparâmetros que maximiza o desempenho do modelo nos dados de treino.

```python
# Create the cross-validation object
cv = get_cross_validation_strategy(X_train, y_train, num_folds)
# Create a GridSearchCV instance for the specified classifier with simplified hyperparameters
optimizer = GridSearchCV(
    SVC(probability=True) if classifier_name == 'svm' else RandomForestClassifier(),
    param_grid=param_space[classifier_name],
    cv=cv,
    n_jobs=1,
    verbose=1,
    refit=True,
)
```

Figura 18 - Afinação dos hiperparametros do classificador ramdom forest

Através destas técnicas de avaliação e comparações, obtemos informações valiosas sobre os pontos fortes e as limitações de cada modelo, o que ajuda a tomar decisões informadas sobre a seleção de modelos e potenciais melhorias.
Em resumo, o algoritmo utiliza princípios de co-treino para aproveitar a discordância entre classificadores para identificar instâncias informativas, que são então adicionadas aos dados etiquetados. Este processo iterativo, associado à aprendizagem em conjunto, não só melhora o desempenho do modelo, como também fornece informações sobre o seu comportamento e eficácia na utilização de dados etiquetados limitados.

5.4 Treino do algoritmo

Esta secção descreve os passos e a lógica subjacente a cada fase do processo de formação. Ao combinar o poder preditivo dos classificadores Support Vetor Machines

(SVM) e Random Forest através do treino conjunto, pretende-se extrair o melhor dos dois algoritmos e melhorar as capacidades preditivas do modelo final.

5.4.1 Treino de classificadores individuais

O estudo começa com o passo crucial que estabelece as bases: treinar os classificadores individuais. Começamos por alimentar o classificador SVM e o classificador Random Forest com 10% de instâncias etiquetadas, cuidadosamente escolhidas para formar a base do nosso esforço de co-treino.
Esta formação inicial prepara os algoritmos para as iterações de co-treino subsequentes. O SVM, com a sua capacidade de criar limites de decisão intrincados, aprende a discernir relações complexas. O Random Forest, constituído por um conjunto de árvores de decisão, capta interações multifacetadas nos dados. Ao treiná-los de forma independente, estamos a criar uma boa base para previsões.

```
# Initialize classifiers
svm = SVC(probability=True)
random_forest = RandomForestClassifier()

# Train SVM classifier
svm.fit(X_labeled, y_labeled)

# Train Random Forest classifier
random_forest.fit(X_labeled, y_labeled)
```

Figura 19 – Treino individual de classificadores

A figura 19 acima mostra a inicialização dos classificadores sem efetuar algum tipo de alteração de hiperparâmetros, para verificar como se comportam os classificadores sem grandes alterações.

5.4.2 O algoritmo

5.4.2.1 Escolha das instâncias

A estratégia de partição de dados prepara o terreno para as iterações de co-treino. Durante cada iteração, o algoritmo identifica as instâncias com elevada discordância entre os dois classificadores (SVM e Random Forest) treinados nos dados etiquetados limitados. Estas instâncias de elevada discordância são subsequentemente etiquetadas e incorporadas no conjunto de dados etiquetados para a iteração seguinte. Este processo iterativo permite que o modelo aperfeiçoe gradualmente as suas previsões e se adapte à distribuição de dados subjacente.

```
# Make predictions on unlabeled data using both classifiers
y_pred_svm = svm.predict_proba(X_unlabeled)
y_pred_rf = random_forest.predict_proba(X_unlabeled)

# Compute disagreement between the classifiers
disagreement = np.sum(np.abs(y_pred_svm - y_pred_rf), axis=1)
```

Figura 20 – Cálculo da discordância

Este cálculo de discordância envolve fazer previsões sobre os dados não etiquetados utilizando ambos os classificadores e calcular a discordância entre as suas previsões.
Aqui é usado o método predict_proba para obter as probabilidades de classe previstas para cada classe para cada instância nos dados não etiquetados. Em seguida, calculamos a discordância encontrando a diferença absoluta entre as probabilidades previstas pelo SVM e pelo Random Forest para cada instância.

```
# Compute disagreement between the classifiers
disagreement = np.sum(np.abs(y_pred_svm - y_pred_forest), axis=1)

# Select instances with high disagreement
high_disagreement_idx = np.argsort(disagreement)[-num_new_labeled_instances:]

# Get the newly labeled instances and their corresponding labels
X_new_labeled = X_unlabeled[high_disagreement_idx]
y_new_labeled = y_unlabeled[high_disagreement_idx]  # No .iloc needed here

# Remove the newly labeled instances from the unlabeled set
X_unlabeled = np.delete(X_unlabeled, high_disagreement_idx, axis=0)
y_unlabeled = np.delete(y_unlabeled, high_disagreement_idx)
```

Figura 21 – Seleção das instâncias com maior discordância

Para medir a discordância entre os classificadores, calcula-se a diferença absoluta entre as distribuições de probabilidade previstas pelos dois classificadores para cada ponto de dados não etiquetado. Soma-se essas diferenças absolutas para obter um único valor que representa o nível de discordância. Essencialmente, está a quantificar o quanto os classificadores discordam sobre a probabilidade de cada classe para cada ponto de dados.
Em seguida, ordenam-se os pontos de dados não etiquetados com base nessa pontuação de discordância em ordem decrescente. Os pontos de dados com as pontuações de discordância mais elevadas são considerados os mais incertos ou ambíguos porque os classificadores discordam mais das suas etiquetas.
Os principais pontos de dados com as pontuações de discordância mais altas são selecionados para etiquetar. Esses são os pontos em que os classificadores discordam

mais. Ao etiquetar estas instâncias, o objetivo é reduzir a incerteza e melhorar o desempenho do modelo.

5.4.2.2 Refinar através da iteração

Depois de selecionar estes pontos de dados para etiquetar, remove-os do conjunto não etiquetado (uma vez que estão agora etiquetados) e adiciona-os ao conjunto etiquetado (uma vez que foram etiquetados com a verdade fundamental). Este processo é iterativo e, a cada iteração, o modelo torna-se mais confiante e preciso.
A opção de escolher as instâncias com maior desacordo é uma estratégia inteligente para a aprendizagem ativa. Dá prioridade à etiquetagem de pontos de dados em que os modelos são mais incertos, conforme indicado pela discordância significativa entre os diferentes classificadores. Esta abordagem pode levar a uma convergência mais rápida do modelo e a custos de etiquetagem reduzidos em comparação com a seleção aleatória de pontos de dados para etiquetagem.

5.4.3 Ensemble Classifier

Um classificador de ensemble é um modelo de aprendizagem automática de conjunto que combina vários classificadores de base e utiliza as suas previsões para fazer uma previsão final. Esta abordagem de ensemble é utilizada para melhorar o desempenho global da previsão e reduzir o overfitting.

```
# Train the ensemble classifier
ensemble_classifier = VotingClassifier(estimators=[('svm', svm), ('random_forest', random_forest)], voting='soft')
ensemble_classifier.fit(X_train, y_train)

# Evaluate the ensemble classifier on the test set
y_test_pred = ensemble_classifier.predict(X_test)
```

Figura 22 – Ensemble dos classificadores

Foi criado um classificador de ensemble com o SVM e o Random Forest que são utilizados como estimadores. O classificador de ensemble é então treinado com os dados etiquetados atualizados. Após o treino, o classificador de conjunto é usado para fazer previsões no conjunto de teste (X_test) e calcular a precisão das previsões.
Depois das previsões de todos os dados é retornado na aplicação o CSV original com uma nova coluna com os dados previstos pelo algoritmo e os dados etiquetados usados no treino e em teste.

5.4.4 Vantagens e implicações da implementação

Embora não seja convencional, esta abordagem de partição de dados traz vários benefícios ao estudo de co-treino:

- Aprendizagem semi-supervisionada: Ao designar um pequeno subconjunto como instâncias marcadas, o estudo simula um cenário de aprendizagem semi-

supervisionada. Isto é particularmente vantajoso quando os dados etiquetados são escassos, uma vez que o modelo pode explorar eficazmente os dados não etiquetados.

- Aprendizagem ativa: Esta abordagem imita os princípios da aprendizagem ativa. O algoritmo seleciona ativamente as instâncias para etiquetagem, concentrando-se naquelas que contribuem para reduzir o desacordo entre os classificadores. Esta abordagem direcionada maximiza o valor dos recursos limitados de etiquetagem.

- Robustez do modelo: Aproveitar instâncias não etiquetadas e refinar progressivamente o modelo através de iterações aumenta a sua robustez, adaptabilidade e capacidade de generalização.

6 Avaliação da solução

Este capítulo é composto por três secções. Em primeiro lugar, são enumeradas as hipóteses de investigação importantes e, em seguida, alguns indicadores e fontes de informação. Depois, é apresentada a Metodologia de Avaliação conduzida para este projeto.

6.1 Hipóteses de investigação

As hipóteses de investigação específicas para o nosso estudo são formuladas com base nas características do conjunto de dados em análise, uma vez que diferentes domínios de conjuntos de dados podem requerer abordagens métricas distintas para uma avaliação mais precisa. Nesse contexto, as hipóteses de investigação formuladas para o nosso caso são as seguintes:

- RH_N1: O algoritmo semi-supervisionado deve ser capaz de obter 0,75 ou mais da métrica F1 Score.
- RH_N2: O algoritmo semi-supervisionado deve ser capaz de terminar toda a execução em menos de 1 hora.
- RH_N3: O algoritmo semi-supervisionado deve apresentar melhorias de precisão em relação aos classificadores separados em pelo menos 3 de 5 conjuntos de dados.

6.2 Indicadores e fontes de informação

Relativamente às métricas para avaliar a solução, a exatidão e a taxa de erro são adequadas para cenários de avaliação de algoritmo para permitir perceber como o algoritmo semi-supervisionado se comporta e para poder comparar com o algoritmo supervisionado e perceber se a utilização dos dados não etiquetados ajudou na exatidão do algoritmo.
Existem medidas de classe única mais adequadas, como a precisão, a recuperação, a medida F1 e a média G, e algumas métricas globais, como a área sob a curva (AUC), o coeficiente de correlação de Matthews (MCC) e o Kappa. Todas estas métricas são calculadas a partir da matriz de confusão. Para cada conjunto de dados, se o objetivo for prever uma etiqueta de classe, pode ser dada prioridade à medida F1 ou, se o objetivo for prever probabilidades de uma etiqueta de classe, pode ser dada prioridade à AUC precisão-recall.
Relativamente aos conjuntos de dados vão ser escolhidos cinco conjuntos de dados para avaliar o algoritmo semi-supervisionado e comparar o algoritmo ao supervisionado e avaliar os resultados.

6.3 Metodologia

Foi executado o algoritmo de aprendizagem ativa com um número fixo de iterações, permitindo-lhe selecionar iterativamente instâncias informativas para etiquetagem. Em cada iteração, o algoritmo treinou um conjunto de classificadores Support Vetor Machine (SVM) e Random Forest nos dados etiquetados. A discordância entre os classificadores foi calculada em instâncias não etiquetadas, ajudando a selecionar as instâncias com maior incerteza para etiquetagem.
O desempenho do algoritmo foi avaliado utilizando várias métricas, incluindo a exatidão, a precisão, a recuperação e a pontuação F1. Além disso, foram construídas matrizes de confusão para visualizar as previsões de verdadeiros positivos, verdadeiros negativos, falsos positivos e falsos negativos efetuados pelo algoritmo.

6.4 Avaliação do algoritmo

Nesta secção, aprofundamos a tarefa crítica de avaliar e comparar o desempenho do algoritmo de co-treino com os classificadores SVM e Random Forest otimizados. A avaliação dos modelos utilizando várias métricas e técnicas ajuda-nos a obter informações significativas sobre os seus pontos fortes e fracos.

6.4.1 Motivação para a seleção de dados

A principal motivação para a escolha dos dados reside na necessidade de criar subconjuntos separados de dados que cumpram funções distintas no ciclo de vida do desenvolvimento do modelo. Esses subconjuntos incluem:

- Dados de treino: Este subconjunto constitui a maior parte do nosso conjunto de dados e serve de base para o treino dos nossos modelos de aprendizagem automática. Os modelos aprendem padrões, relações e correlações presentes nos dados de treino para fazer previsões exatas.
- Dados de validação: Também conhecido como conjunto de desenvolvimento, este subconjunto ajuda na afinação de hiperparâmetros e na seleção de modelos. Permite-nos avaliar o grau de generalização dos nossos modelos para dados não vistos e tomar decisões informadas sobre a arquitetura e as definições do modelo.
- Dados de teste: Os dados de teste permanecem intocados durante o desenvolvimento do modelo e o ajuste dos hiperparâmetros. São utilizados para avaliar rigorosamente o desempenho do modelo em instâncias totalmente novas e não vistas. Os resultados obtidos nos testes fornecem uma estimativa realista da eficácia do modelo em cenários do mundo real.

6.4.2 Métricas de avaliação

Para avaliar o desempenho dos modelos, utilizamos um conjunto de métricas de avaliação. Algumas métricas comuns incluem a exatidão, a precisão, a recuperação e a pontuação F1.

- Exatidão: Mede a correção geral das previsões.
- Precisão: Indica a proporção de instâncias positivas corretamente previstas de entre todas as instâncias previstas como positivas.
- Recuperação: Mede a proporção de instâncias positivas corretamente previstas de todas as instâncias positivas reais.
- F1-Score: Média harmónica da precisão e da recuperação, fornecendo uma avaliação equilibrada.

```
# Calculate evaluation metrics for co-training algorithm
accuracy_co_train = accuracy_score(y_test, y_test_pred_co_train)
precision_co_train = precision_score(y_test, y_test_pred_co_train)
recall_co_train = recall_score(y_test, y_test_pred_co_train)
f1_co_train = f1_score(y_test, y_test_pred_co_train)

# Print the metrics
print("Co-Training Algorithm Metrics:")
print("Accuracy:", accuracy_co_train)
print("Precision:", precision_co_train)
print("Recall:", recall_co_train)
print("F1 Score:", f1_co_train)
```

Figura 23 – Métricas de avaliação do algoritmo

A métrica usada para avaliar o algoritmo é F1 score porque fornece uma avaliação equilibrada e abrangente do desempenho do modelo em tarefas de classificação. Ao contrário da exatidão, que pode ser enganadora em conjuntos de dados desequilibrados, a pontuação F1 tem em conta tanto a precisão como a recuperação, tornando-a robusta a situações em que os falsos positivos e os falsos negativos têm implicações diferentes. Ao calcular a média harmónica da precisão e da recuperação, a pontuação F1 oferece uma métrica única e fácil de interpretar que quantifica eficazmente a capacidade do modelo para classificar corretamente as instâncias, tendo em conta o compromisso entre a precisão e a recuperação.

6.4.3 Comparação de desempenho

Para comparar o desempenho dos classificadores individuais e do algoritmo de forma abrangente, podemos visualizar e analisar as suas métricas de avaliação lado a lado.

```
# Create a bar chart to compare evaluation metrics
metrics = ['Accuracy', 'Precision', 'Recall', 'F1 Score']
co_train_metrics = [accuracy_co_train, precision_co_train, recall_co_train,
svm_metrics = [accuracy_svm, precision_svm, recall_svm, f1_svm]
rf_metrics = [accuracy_rf, precision_rf, recall_rf, f1_rf]

x = np.arange(len(metrics))
width = 0.25

plt.bar(x - width, co_train_metrics, width, label='Co-Training')
plt.bar(x, svm_metrics, width, label='SVM')
plt.bar(x + width, rf_metrics, width, label='Random Forest')

plt.xlabel('Metrics')
plt.ylabel('Scores')
plt.title('Model Performance Comparison')
plt.xticks(x, metrics)
plt.legend()
plt.show()
```

Figura 24 – Comparação de métricas usando os classificadores individuais e o algoritmo de co-treino

Este código cria um gráfico de desempenho dos classificadores individuais e do algoritmo para se comparar a performance de cada um.

7 Experimentos e resultados

Neste capítulo, apresentamos a configuração experimental, a metodologia e os resultados obtidos para avaliar o desempenho do algoritmo proposto. As experiências são realizadas em conjuntos de dados do mundo real para demonstrar a eficácia do algoritmo numa variedade de cenários.

7.1 Configuração experimental

A configuração experimental engloba a seleção do conjunto de dados, as etapas de pré-processamento e a configuração do algoritmo de aprendizagem ativa. Foram escolhidos cinco conjuntos de dados diversos, nomeadamente os conjuntos de dados "Doenças cardíacas", "deteção de fraude", "Qualidade do vinho", “Deteção de cancro da mama” e classificação de movimento humano, para avaliar a adaptabilidade do algoritmo em diferentes domínios. Os valores em falta foram tratados utilizando técnicas de imputação adequadas, garantindo que os conjuntos de dados estavam limpos e eram adequados para a formação.

7.1.1 Conjuntos de dados

D1 -> Cancro da mama

Com este conjunto de dados o principal desafio para a deteção é a classificação dos tumores em malignos (cancerosos) e benignos (não cancerosos).

D2 -> Doenças cardíacas

O conjunto de dados de doença cardíaca tem como objetivo detetar através dos atributos se o paciente tem uma doença cárdica.

D3 -> Movimento do corpo humano

O conjunto de dados consiste em dados recolhidos de 36 utilizadores diferentes que realizam seis tipos de atividades humanas (subir e descer escadas, sentar-se, caminhar, correr e estar de pé) durante períodos específicos.
O objetivo neste conjunto de dados é conseguir prever a atividade da pessoa através dos atributos que se encontram no conjunto de dados.

D4 -> Deteção de fraúde

O conjunto de dados contém transações efetuadas com cartões de crédito em setembro de 2013 por titulares de cartões europeus. Este conjunto de dados apresenta transações que ocorreram em dois dias, onde temos 492 fraudes em 284 807 transações.

Devido a questões de confidencialidade, não foi possível obter as características originais e mais informações de base sobre os dados. A caraterística "Class" é a variável de resposta e assume o valor 1 em caso de fraude e 0 no caso contrário.

D5 -> Qualidade de vinho
Este conjunto de dados está relacionado com as variantes do vinho verde português. O conjunto de dados descreve a quantidade de vários químicos presentes no vinho e o seu efeito na sua qualidade. O objetivo é prever a qualidade do vinho.

Tabela 9 – Tabela com informação dos conjuntos de dados

ID	Conjunto de dados	N linhas x colunas	Atrib Obj.
D1	Cancro da mama	1021 x 32	2 classes
D2	Doenças cardíacas	1027 x 14	2 classes
D3	Movimento do corpo humano	7353 X 563	6 classes
D4	Deteção de fraúde	284808 X 31	2 classes
D5	Qualidade de vinho	1145 X 14	7 classes

Começou-se por testar os classificadores isoladamente para ver como se comportam e para ter uma ideia de quanto o algoritmo semi-supervisionado pode melhorar relativamente aos classificadores base.
De seguida, foi testado o algoritmo semi-supervisionado com vários testes variando a quantidade de dados fornecida para verificar qual a quantidade de dados mais adequada para o algoritmo poder obter resultados satisfatórios relativamente à classificação das etiquetas.
Por fim é analisado todo o algoritmo relativamente á sua eficácia, tempo e desempenho.

7.2 Experimento 1 (avaliar os classificadores isoladamente)

7.2.1 SVM

A Máquina de Suporte de Vetores (SVM) é uma técnica de aprendizagem automática que encontra a melhor linha ou plano para separar os diferentes grupos de pontos de dados. O seu objetivo é criar uma fronteira clara entre estes grupos, maximizando a distância entre a fronteira e os pontos de dados mais próximos de cada grupo. Em

termos mais simples, o SVM ajuda a classificar as instâncias desenhando uma linha que as mantém o mais afastadas possível.

Tabela 10 – Resultados do classificador SVM isoladamente com 10% dos dados etiquetados

	D1	D2	D3	D4	D5
F1 score	0.337	0.709	0.87	0.635	0.479

O resultado nos vários conjuntos de dados foi satisfatório nos conjuntos de dados D2, D3, D4 apesar de nos conjuntos de dados D1 e D5 os resultados serem inferiores a 50%. Significa que o algoritmo semi-supervisionado tem margens para melhorias aprendendo com o outro classificador nas instâncias mais difíceis para a SVM.

7.2.2 Ramdom Forest

O Random Forest é uma poderosa abordagem de aprendizagem automática que combina várias árvores de decisão para fazer melhores previsões. O Random Forest cria muitas árvores, cada uma utilizando dados e variáveis diferentes. Em seguida, combina as previsões individuais para tomar uma decisão final. Tal como um grupo de especialistas a colaborar num problema, o Random Forest aproveita a diversidade destas árvores para fornecer uma previsão mais fiável e precisa.

Tabela 11 - Resultados do classificador Ramdom forest isoladamente com 10% dos dados etiquetados

	D1	D2	D3	D4	D5
F1 score	0.742	0.857	0.88	0.932	0.69

O classificador random forest conseguiu obter resultados bons em todos os conjuntos de dados sendo por isso um classificador adequado para ajudar o classificador SVM a aprender e a melhorar a sua previsão, obtendo novos dados e expandindo o seu poder de previsão.

7.3 Experimento 2 (avaliar o algoritmo co-train)

Depois de testar os algoritmos separadamente vamos verificar se o algoritmo semi-supervisionado consegue etiquetar novos dados e com estes adicionados aos dados etiquetados melhora a taxa de acerto dos algoritmos base.
O algoritmo primeiro vai ser treinado com 10% dos dados etiquetados e depois com 20% para verificar também se existe melhoria de previsão dos dados.

7.3.1 Treino com 10% dos dados etiquetados

Tabela 12 – Resultados dos classificadores isolados e o algoritmo com 10% dos dados

F1-score	D1	D2	D3	D4	D5
SVM	0.337	0.709	0.87	0.635	0.379
Random Forest	0.742	0.857	0.88	0.932	0.69
Co-train	0.971	0.97	0.991	0.981	0.75

O algoritmo só com 10% dos dados etiquetados conseguiu obter melhores resultados nos cinco conjuntos de dados.

7.3.2 Treino com 20% dos dados etiquetados

Tabela 13 - Resultados dos classificadores isolados e o algoritmo com 20% dos dados

F1-score	D1	D2	D3	D4	D5
SVM	0.492	0.784	0.882	0.73	0.66
Random Forest	0.793	0.833	0.901	0.941	0.72
Co-train	0.982	0.987	0.99	0.997	0.81

O treino com 20% dos dados etiquetados oferece um aumento na performance do algoritmo co-train cerca de 2% a 3%, sendo que adicionando mais dados melhor seria.

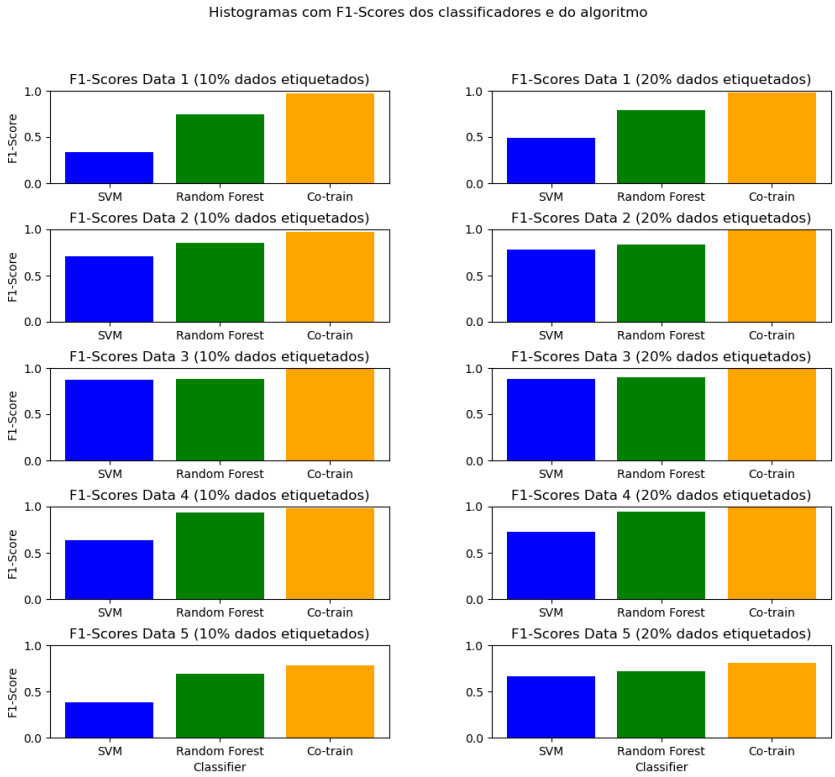

Figura 25 – Demonstração dos resultados com histogramas

Nestes histogramas consegue-se ver os resultados dos classificadores isoladamente com 10% e 20% dos dados etiquetados e também do algoritmo semi-supervisionado. Como se pode ver nos 5 conjuntos de dados o algoritmo obtém sempre melhores resultados melhorando a previsão com a etiqueta dos dados.

7.4 Discussão e interpretação

Os resultados obtidos a partir das experiências destacam as capacidades do algoritmo em aproveitar dados etiquetados limitados para melhorar o desempenho da classificação. Em todos os cinco conjuntos de dados, o algoritmo superou consistentemente os classificadores SVM e Random Forest autónomos. Isto demonstra a eficácia do algoritmo em aproveitar a força dos princípios de aprendizagem em conjunto e de co-treino.

As métricas de exatidão, precisão, recuperação e pontuação F1 revelaram melhorias notáveis ao longo das iterações. À medida que o algoritmo selecionava iterativamente e etiquetava as instâncias informativas, o desempenho dos classificadores apresentava um aumento constante. Além disso, as matrizes de confusão forneceram informações sobre o comportamento do classificador na distinção entre diferentes classes, revelando a sua robustez e potencial para aplicações no mundo real.

Os resultados experimentais sublinham o potencial do algoritmo para várias aplicações, incluindo a deteção de fraudes, o diagnóstico de doenças e a previsão sobre a aprovação de crédito. O processo iterativo do algoritmo de etiquetagem das instâncias mais incertas contribui para a sua capacidade de adaptação e melhoria ao longo do tempo, tornando-o uma escolha atrativa para cenários em que a etiquetagem manual consome muitos recursos.
Foram também realizados testes em conjuntos de imagens, mas devido ao tempo de execução, não foi implementado, apesar dos resultados positivos obtidos com o algoritmo de co-train.

8 Conclusão

Nesta dissertação, pretendeu-se explorar e analisar vários aspetos da aprendizagem automática e das técnicas de aprendizagem ativa para a classificação de dados. O objetivo principal era desenvolver e avaliar algoritmos que pudessem lidar eficazmente com dados etiquetados e não etiquetados, utilizando eficientemente os recursos disponíveis e mantendo ou melhorando o desempenho da classificação.

Inicialmente, foi realizada uma análise deste projeto através do levantamento do estado da arte da literatura sobre conceitos teóricos fundamentais relacionados com os algoritmos semi-supervisionados e sobre as suas capacidades de aproveitamento dos dados não etiquetados.

Além disso, foram também estudadas algumas técnicas que são usadas para construir os algoritmos de classificação semi-supervisionada.

A seguir, foi feita uma análise de valor do projeto, para além da arquitetura inicial da solução que é fundamental para a implementação futura deste projeto.

Na parte da implementação, começou-se por discutir a importância da etiquetagem de dados na aprendizagem supervisionada e destaca-se os desafios colocados por dados etiquetados limitados. Para resolver este problema, explorámos o conceito de aprendizagem ativa, que visa selecionar e etiquetar de forma inteligente as instâncias mais informativas. Isto não só reduz a carga de etiquetagem como também melhora o desempenho do modelo.

Esta investigação envolveu o desenvolvimento e a implementação de algoritmos semi-supervisionados de forma a provar que estes algoritmos conseguem ter bom desempenho em alguns conjuntos de dados. Foi explicado meticulosamente o funcionamento destes métodos e demonstrou-se a sua eficácia através de experiências. Além disso, foram exploradas técnicas para lidar com conjuntos de dados desequilibrados, um problema comum em cenários do mundo real.

Relativamente a uns dos objetivos de comparar aos outros algoritmos de semi-supervisionados, não foi elaborada uma comparação de entre algoritmos semi-supervisionados recentes.

Em conclusão, as experiências e os resultados apresentados nesta dissertação demonstram a eficácia do algoritmo no aproveitamento de dados etiquetados limitados através de uma combinação de princípios de aprendizagem em conjunto e de co-treino. Os resultados obtidos fornecem provas empíricas do potencial do algoritmo para melhorar o desempenho da classificação em vários domínios, estabelecendo a sua aplicabilidade em cenários do mundo real.

Embora os resultados sejam prometedores, é importante reconhecer certas limitações. O desempenho do algoritmo depende fortemente das instâncias iniciais etiquetadas e da qualidade dos dados não etiquetados. Além disso, a convergência do algoritmo pode variar consoante o conjunto de dados e a natureza do problema. As direções futuras incluem a exploração de estratégias híbridas que integram outras técnicas de aprendizagem que possam melhorar a aptidão do algoritmo.

Referências

[1] Design science research applied to difficulties of teaching and learning, initial programming 2022, Figueiredo, J e Garcia-Penalvo, F, Universal Access in the Information Society, view 27 Janeiro 2023, https://doi.org/10.1007/s10209-022-00941-4

[2] Analysis of Design Science Research Methodology and Entrepreneurship Connections, José S. Dias, F, Instituto Superior Técnico, Universidade Técnica de Lisboa, Summary of dissertation for the degree of Master in Information Systems and Computer Engineering, view 30 Janeiro 2023, https://doi.org/10.1007/s10209-022-00941-4

[3] A Design Science Research Methodology for Information Systems Research, Ken Peffers, Journal of Management Information Systems , https://doi.org/10.2753/MIS0742-1222240302

[4] Canito, A., Corchado, J. & Marreiros, G. A systematic review on time-constrained ontology evolution in predictive maintenance. Artif Intell Rev 55, 3183–3211 (2022). https://doi.org/10.1007/s10462-021-10079-z

[5] Sarker Monojit Asish, Ekram Hossain, Arun K. Kulshreshth and Christoph W. Borst. 2021. Deep Learning on Eye Gaze Data to Classify Student Distraction Levelinan Educational VR Environment. In ICAT-EGVE2021-International Conference on Artificial Reality and Telexistence and Eurographics Symposium on Virtual Environments, JasonOrlosky, DirkReiners, and Benjamin Weyers (Eds.). The Eurographics Association. https://doi.org/10.2312/egve.20211326

[6] Renée Burton. 2020. Unsupervised Learning Techniques for Malware Characterization: Understanding Certain DNS-based DDoSAttacks. Digit.Threat.:Res. Pract. 1,3, Artigo14 (Agosto 2020) ,26pages. https://doi.org/10.1145/3377869

[7] Chao Yu, Jiming Liu, Shamim Nemati, and Guosheng Yin. 2021. Reinforcement Learning in Healthcare: A Survey. ACM Comput. Surv. 55, 1, Article 5 (November 2021), 36 pages. https://doi.org/10.1145/3477600

[8] Deep Reinforcement Learning for Autonomous Vehicle Navigation in Urban Environments, 2022

[9] Oliver, A., Odena, A., Raffel, C. A., Cubuk, E. D., & Goodfellow, I. (2018). Realistic evaluation of deep semi-supervised learning algorithms. Advances in neural information processing systems, 31.

[10] van Engelen, J.E., Hoos, H.H. A survey on semi-supervised learning. Mach Learn 109, 373–440 (2020). https://doi.org/10.1007/s10994-019-05855-6

[11] Singla, M., Ghosh, D. & Shukla, K.K. TSVM: A Robust Transductive Support Vector Machine and its Application to the Detection of COVID-19 Infected Patients. Neural Process Lett 53, 3981–4010 (2021). https://doi.org/10.1007/s11063-021-10578-8

[12] A. Fabijanska, "Graph Convolutional Networks for Semi-Supervised Image Segmentation," in IEEE Access, vol. 10, pp. 104144-104155, 2022, doi: 10.1109/ACCESS.2022.3210533.

[13] Mohamed Farouk Abdel Hady and Friedhelm Schwenke, Chapter 7 Semi-supervised Learning, Handbook on Neural Information Processing, DOI: 10.1007/978-3-642-36657-4_7

[14] Koen P., Bertels H. and Kleinschmidt E., Managing the Front End of Innovation-Part I: Results from a Three-Year Study: Senior Management Commitment, Vision, Strategy, Resource Commitment, and Culture Are the Keys to Front-End Success. Em: Research-Technology Management

[15] Koen P., Ajamian G., Boyce S., Clamen A., Fisher E., Fountoulakis S., Johnson A., Puri P., and Rebecca Seibert, Fuzzy Front End: Effective Methods, Tools, and Techniques

[16] Markus Hofrichter, Simplissimo Livros Ltda, 17/01/2017

[17] Koen P., Ajamian .G, Burkart R., Clamen A., Davidson J., D'Amore R, Elkins C., Herald K., Incorvia M., Johnson A., Karol R., Seibert R., Slavejkov A., and Wagner K.,PROVIDING CLARITY AND A COMMON LANGUAGE TO THE "FUZZY FRONT END", Eight companies collectively determined a theoretical construct for the Fuzzy Front End of innovation in order to provide a common framework and language; they found that highly innovative companies have a more proficient FFE.

[18] Graf, Albert & Peter Maas (2008). "Customer value from a customer perspective: a comprehensive review". In: Journal für Betriebswirtschaft 58.1, pp. 1–20.

[19] Osterwalder, Alexander, Yves Pigneur, et al. (2015). Value proposition design: How to create products and services customers want. Vol. 2. John Wiley & Sons.

[20] Thomas L. Saaty. «What is the Analytic Hierarchy Process?» Em: Mathematical Models for Decision Support. Ed. por Gautam Mitra et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 1988, pp. 109–121. isbn: 978-3-642-83555-1.

[21] Zhou, Z. H. (2012). Ensemble methods: Foundations and algorithms. Boca Raton: CRC Press.