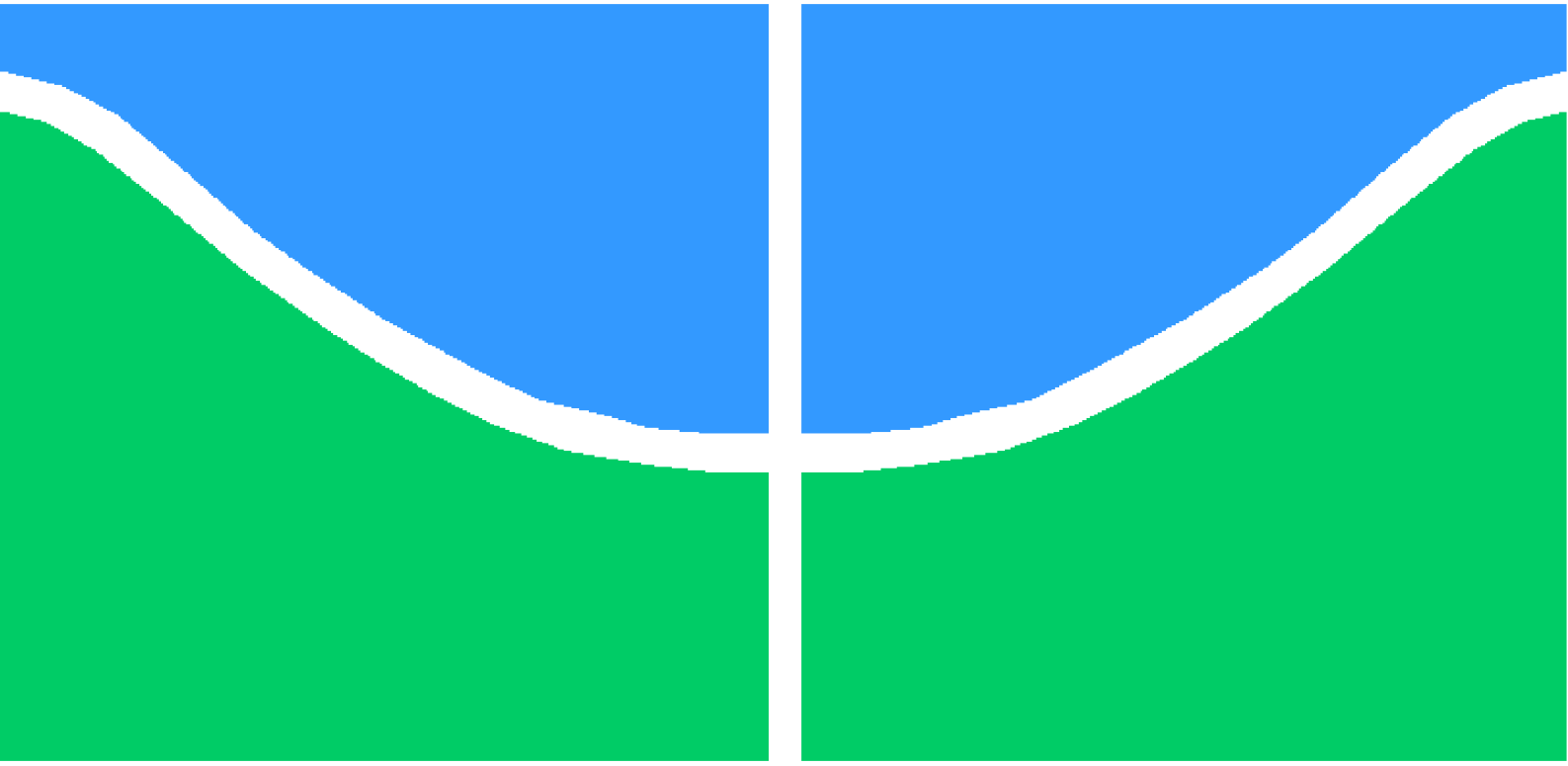

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Engenharia de Software

Pesquisa e Desenvolvimento de um Método de Aprendizagem de Máquina: Agrupamento Incremental

Autores: Gustavo Corrêia de Lima

Orientador: Prof. Dr. Nilton Correia da Silva

Brasília, DF

2015

Gustavo Corrêia de Lima

Pesquisa e Desenvolvimento de um Método de Aprendizagem de Máquina: Agrupamento Incremental

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: Prof. Dr. Nilton Correia da Silva

Brasília, DF

2015

Gustavo Corrêia de Lima

Pesquisa e Desenvolvimento de um Método de Aprendizagem de Máquina: Agrupamento Incremental/ Gustavo Corrêia de Lima. – Brasília, DF, 2015-

83 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Dr. Nilton Correia da Silva

Trabalho de Conclusão de Curso – Universidade de Brasília - UnB
Faculdade UnB Gama - FGA , 2015.

1. Aprendizagem de Máquina. 2. Aprendizagem Incremental. 3. Agrupamento - Clusterização. I. Prof. Dr. Nilton Correia da Silva. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. Pesquisa e Desenvolvimento de um Método de Aprendizagem de Máquina: Agrupamento Incremental

CDU

Gustavo Corrêia de Lima

Pesquisa e Desenvolvimento de um Método de Aprendizagem de Máquina: Agrupamento Incremental

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Trabalho aprovado. Brasília, DF, Julho de 2015:

Prof. Dr. Nilton Correia da Silva
Orientador

Prof. Dr. Fabricio Braz
Convidado 1

Prof. Dr. Luiz Augusto Fontes Laranjeira
Convidado 2

Brasília, DF
2015

Agradecimentos

Agradeço a Deus pelo privilégio de estar vivo e pela capacidade de aprender coisas novas. Sou grato a minha família que me dá todo o suporte necessário ao meu desenvolvimento humano. Aos professores que, de forma grata, passam seus conhecimentos à frente. À Universidade de Brasília e ao Brasil que são patrocinadores do conhecimento.

Resumo

Este trabalho consiste no levantamento dos métodos de Aprendizagem de Máquina e na discussão acerca da necessidade de uso de Aprendizagem Incremental. Todo o fluxo do Aprendizado de Máquina é exposto, bem como a necessidade da utilização de Aprendizagem Incremental, com ênfase em algumas de suas peculiaridades. O método não-supervisionado de agrupamento incremental TASOM é introduzido. Experimentos são realizados visando comparar a performance deste método com o SOM, seu método de origem. Um estudo de caso baseado em um problema real do Ministério da Agricultura, Pecuária e Abastecimento é apresentado, e objetivos para o desenvolvimento de uma solução são descritos. A base de dados de plantas do tipo Iris também é utilizada para validar o TASOM.

Palavras-chave: Aprendizagem de Máquina, Aprendizagem Incremental, Agrupamento - Clusterização.

Abstract

This work contains a research about Machine Learning Methods and the discussion about the need of Incremental Learning. All the Machine Learning proccess is shown, as well as the need of Incremental Learning, focusing on some of it's details. The incremental unsurpevised clustering method TASOM is introduced. Experiments are made to compare the performance of this method with the method SOM, it's parent algorithmn. A real case study based on a necessity of the Department of Agriculture, Livestock and Supply is shown. The famous dataset of Iris' plants is also used to validate the TASOM.

Palavras-chave: Machine Learning, Incremental Learning, Clusterization.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

UnB	Universidade de Brasília
UE	União Europeia
AM	Aprendizado de Máquina
MAPA	Ministério da Agricultura Pecuária e Abastecimento
MCAR	Missing completely at random
MAR	Missing at random
NMAR	Not missing at random
EC	Engenharia de Características
ROC	Receiver Operating Characteristic
SOM	Self Organizing Map
GSOM	Growing Self Organizing Map
PNCR	Plano Nacional de Controle de Resíduos e Contaminantes
SIF	Serviço de Inspeção Federal

Sumário

1 Introdução

Com o advento do crescimento do uso de Tecnologia da Informação na sociedade moderna, uma enorme quantidade de dados vem sendo gerada diariamente. Big Data é o termo em Inglês usado para descrever o fenômeno deste grande volume de informações que a sociedade gera atualmente. Citam-se como exemplos de fontes destas informações: redes sociais, registros de transações e dados de sensores. Estes conjuntos de elementos têm a característica de serem desestruturados, crescerem muito rapidamente com o tempo e terem alta complexidade.

De acordo com o relatório produzido em 2013 pela União Europeia, UE, aproximadamente 90% de todos os dados gerados pela humanidade em 2013 surgiram nos dois anos anteriores (DERVOJEDA et al., 2013). Ainda afirma-se neste relatório que o desafio moderno não está na capacidade de armazenamento dos registros, mas, sim, em usá-los de forma prática e eficiente. Estima-se que apenas um quinto desses registros sejam de origem numérica. De acordo com Hilbert e López (2011), em um trabalho que estimou a capacidade tecnológica do mundo de armazenar, comunicar e processar informação:

1. As telecomunicações foram dominadas pelas tecnologias digitais desde 1999, correspondendo a 99.9% no ano de 2007.

2. A maior parte da informação tecnológica da humanidade está em formato digital desde os anos 2000, correspondendo a 94% no ano de 2007.

3. A capacidade de armazenamento de informação tecnológica *per-capita* vem dobrando a cada 40 meses desde a década de 1980.

Considerando-se os desafios apresentados, Aprendizado de Máquina (AM), ou comumente conhecido pelo termo em Inglês, Machine Learning, é um novo campo do conhecimento que possui elementos interseccionados entre: Ciência da Computação, Estatística, Teoria da Informação, Filosofia, Biologia, Ciência Cognitiva, Complexidade Computacional e Teoria de Controle (MITCHELL, 1997). Ela tem forte relacionamento com a Inteligência Artificial, pois se assemelham em objetivos e técnicas.

Aprendizado de Máquina se preocupa em habilitar sistemas computacionais a aprender ou desempenhar determinada função sem ser diretamente programados para aquela tarefa. De acordo com Arthur Lee Samuel, cientista pioneiro do campo de Inteligência Artificial, Aprendizado de Máquina é: "Campo de estudo que habilita computadores a aprender sem ser explicitamente programados"(A.L.Samuel apud Simon (2013)).

Sistemas computacionais baseados em modelagens de AM são necessários para solucionar problemas altamente técnicos e especializados. Algumas das tarefas que demandam o uso destes métodos são: aprendizagem, raciocínio, planejamento, tomada de decisões, classificações e predições.

Geralmente sistemas computacionais são feitos para resolver tarefas de forma explícita, isto é, o programador conhece plenamente o problema que deve ser atacado e comanda o computador a executar passos que correspondem às etapas necessárias para a resolução do problema. Os programadores usam algoritmos para habilitar sistemas computacionais a realizar diversas tarefas. Um algoritmo é uma coleção bem ordenada de operações computacionais claras e efetivas que, quando executadas, produzem um resultado e param em uma quantidade finita de tempo (M.; GERSTING, 1995).

O responsável pela programação do algoritmo deve conhecer todos os relacionamentos do problema e ser capaz de identificar mentalmente todos os processos envolvidos na resolução. Isto nem sempre é possível, pois existem problemas tão complexos que tornam a abstração de sua resolução impossível para a capacidade humana. São muitos relacionamentos escondidos que tornam a tarefa de programar explicitamente inviável, pois isto requer conhecer todos os passos da resolução do problema.

Este tipo de trabalho é que o Aprendizado de Máquina nasceu para solucionar. Em vez de programar cada etapa de forma exaustiva, o objetivo é ensinar o computador a aprender sozinho as etapas de resolução do problema. Isto geralmente se dá quando é apresentado ao computador um conjunto de dados de determinado contexto. Nestes dados estão contidos todos os relacionamentos implícitos que compõem a solução.

A tarefa do computador é extrair estes relacionamentos para ser capaz de generalizar uma solução. Sistemas de AM são preparados para aprender, crescer e se modificar quando apresentados a novos conjuntos de dados, por isto muitas definições de AM estão relacionadas aos dados.

Financeiramente, Aprendizado de Máquina é um campo que vem obtendo ascendente crescimento de investimento. Citando novamente o relatório da UE, Dervojeda et al. (2013), somente na Europa estima-se que o mercado de sistemas especialistas gerou um total de 700 milhões de euros em 2013. As previsões são de que este número chegue a 27 bilhões de euros no ano de 2015. De acordo com as suposições dos autores deste relatório, os trabalhos na área de AM irão impulsionar uma grande demanda por profissionais altamente capacitados, além de incentivar pesquisas em diversas áreas do conhecimento. Isto tudo gerando expansões e investimentos, tanto nos Estados Unidos, quanto nos países que compõem o BRIC (Brasil, Rússia, Índia e China).

O objetivo deste trabalho é a pesquisa e desenvolvimento de um método de Aprendizagem de Máquina do tipo não supervisionado, de agrupamento e incremental. A pes-

quisa serve para encontrar modelos adequados ao Aprendizado Incremental e o desenvolvimento visa validar estes modelos de forma comparativa. Neste trabalho é feito um levantamento teórico sobre o processo de Aprendizagem de Máquina e sobre Aprendizagem Incremental. Dois algoritmos não supervisionados e de agrupamento são escolhidos, um clássico que não possui aprendizagem incremental e outro mais recente que incorpora este conceito incremental. Os dois algoritmos são comparados visando validar o desempenho do novo com os resultados do clássico. Um estudo de caso real é apresentado para testar o algoritmo incremental em contexto prático.

Após levantamento do estado da prática é observada a necessidade da utilização de algoritmos incrementais e não-supervisionados para aplicação em contextos de big data, onde há uma enorme quantidade de informações que precisam ser analisadas. O algoritmo SOM é escolhido por ser de agrupamento não supervisionado, e algumas de suas variações incrementais são comparadas visando encontrar um bom modelo incremental baseado do SOM. O TASOM é apontado como um algoritmo viável por ser incremental e comparável com o SOM em contexto estático. Testes são realizados visando comparar a eficácia dos dois algoritmos em contexto estático e a eficácia do TASOM em contextos dinâmicos. Estes testes realizados comparam os algoritmos do ponto de vista matemático, observando as características centrais que definem cada um dos modelos. Depois, dois casos reais são utilizados para observar a capacidade dos métodos na realização de agrupamentos para visualização das distribuições naturais dos dados.

O desenvolvimento deste documento nos próximos capítulos é dividido em 7 partes. A primeira trata de Aprendizagem de Máquina e nela é definido o que é AM e quais são as fases de seu desenvolvimento. O segundo capítulo trata de Aprendizagem Incremental. A importância desse tipo de aprendizagem é demonstrada e algumas discussões são feitas, comparando a visão de alguns pesquisadores com a visão da comunidade de AM. Na terceira parte, é discutido sobre o tipo de problema de AM da classe agrupamento, o algoritmo SOM (Self Organizing Map) é apresentado e algumas versões deste algoritmo na forma incremental são apresentadas. Logo após o algoritmo TASOM (Time Adaptative Self Organizing Map) é detalhado. Na quinta parte é apresentada a metologia de comparação entre o SOM e o TASOM, bem como os resultados dessa comparação. A sexta parte contém um estudo de caso que se baseia em um problema real do Ministério da Agricultura Pecuária e Abastecimento, MAPA e a aplicação em um conjunto de dados de plantas do tipo Iris. A última parte contém as considerações finais e propostas para a continuação do trabalho.

2 Aprendizado de Máquina

AM pode ser altamente benéfico para solução de problemas complexos. É possível observar um ciclo de trabalho bem definido que está presente na solução de problemas por uma abordagem de Aprendizado de Máquina. As etapas gerais de projetos de AM são: Aquisição de dados, construção de modelo, análise, otimização e predição.

Com estas cinco etapas, é possível sair de um conjunto de dados e chegar às respostas desejadas através de cuidadosa seleção e processamento de dados. Partindo da construção e avaliação de um método eficaz até chegar a um modelo consistente é possível solucionar problemas complexos. Embora haja uma linearidade na execução desses processos, é comum em projetos de AM revisitar estas etapas várias vezes. A figura 1 revela um ciclo de AM mais detalhado e que engloba as etapas previamente citadas, ressaltando elementos importantes:

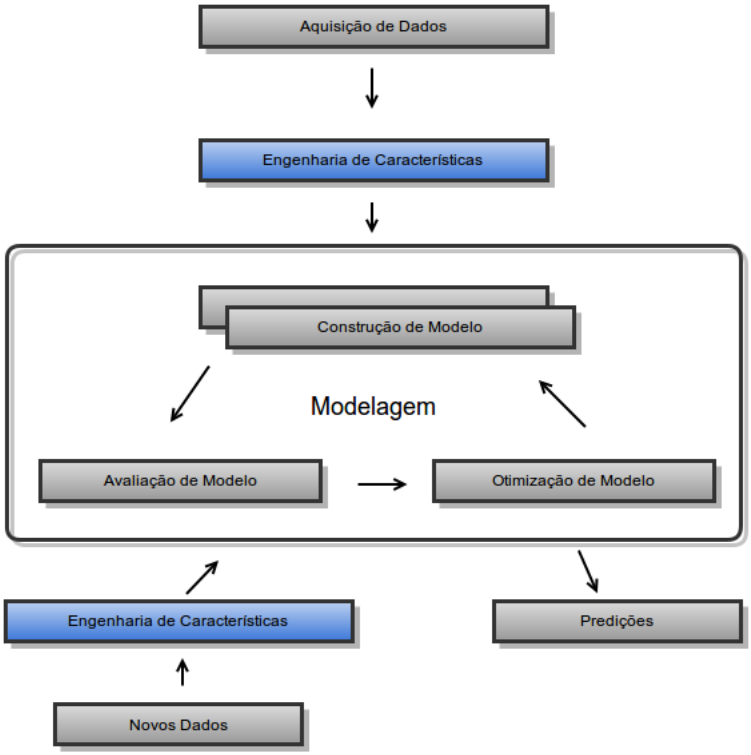

Figura 1 – Fluxo de Trabalho de AM - (BRINK; RICHARDS, 2013)

A figura 1 mostra os ciclos de trabalho que acontecem até que o modelo esteja satisfatório com as expectativas de performance. Várias etapas podem ser repetidas até que o modelo esteja refinado o suficiente. A área central que corresponde a construção do modelo pode levar várias iterações e, às vezes, é necessário rever até mesmo a primeira etapa do ciclo, Aquisição dos Dados.

2.1 Aquisição dos dados

Apesar de parecer trivial, definir a aquisição de dados como uma etapa do projeto de AM é extremamente importante e alguns cuidados especiais devem ser considerados para que as predições tenham um bom desempenho. O primeiro passo antes de começar o projeto de AM é saber qual a necessidade do mundo real que deve ser atendida. Este passo pode ser entendido como a necessidade de encontrar questões que envolvem uma variável de interesse (quem gostaria de comprar este produto, qual o significado deste áudio em linguagem natural, onde está a face humana nesta imagem,...) e o caminho para solução que envolve variáveis independentes (histórico de compras dos usuários, mapeamento de sons para texto, características de faces humanas).

Nem sempre, utilizar Aprendizagem de Máquina é a melhor resposta para solucionar os problemas do mundo real. É necessário avaliar com cuidado quais os objetivos da solução e qual o contexto do problema. Um grande indício de que o problema pode ser solucionado por AM é se ele possui algumas das seguintes características: Alta complexidade entre entradas e saídas, grande volume de dados, necessidade de generalização e necessidade de adaptabilidade a novos cenários.

Os dados que servem como entrada para os processos de modelagem geralmente estão apresentados na forma de tabelas que possuem colunas e linhas. As colunas representam as características dos dados, como se fossem meta-dados, e as linhas representam instâncias dessas características. A tabela a seguir representa os tipos de dados que podem aparecer como característica de dados na Aprendizagem de Máquina (LEBANON, 2010).

Tabela 1 – Tipos de Dados

Tipo	Exemplo
Átomo Categórico	Uma palavra na língua portuguesa
Átomo Numérico	Valor de temperatura
Átomo Ordinário	Preferência cinematográfica (Número de Estrelas)
Conjunto desordenado de números	Sinais vitais(pulso, temperatura, pressão sanguínea)
Conjunto desordenado de misturas	Informação demográfica(raça, sexo, idade, renda)
Sequência unidimensional de categóricos	Documento de texto
Sequência unidimensional de números	Série Temporal Financeira
Sequência bidimensional de números	Imagem
Sequência tridimensional de números	Filme
Grafo Arbitrário Categórico	Árvore de Conversões

A figura 2 é uma pequena seleção da base de dados que contém informações sobre os passageiros que estavam presentes no naufrágio do Titanic. Cada linha corresponde às informações de um passageiro, e cada coluna representa um tipo de informação deste passageiro. Esta base de dados foi criada com o objetivo de tentar prever, através das

informações de um passageiro, se ele sobreviveu ou não ao naufrágio. O conjunto de dados já contém a informação sobre a sobrevivência do passageiro. Muitos modelos são criados com base nas informações dele. É possível fazer predições sobre a sobrevivência dos passageiros e comparar estes resultados com a informação real que já está registrada nas tabelas. Por este motivo, o conjunto de dados vem sendo usado de forma didática para AM. Muitas pessoas utilizam-no para validar seus modelos e há competições baseadas em predições com esta base (TITANIC. . . , 2012a). A enumeração a seguir explica qual o tipo de informação de cada coluna.

IdPassageiro	Sobreviveu	Pclasse	Nome	Sexo	Idade	IrConj	PaiFilh	Ticket	Taixa	Cabine	Embarcado
1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171	7.25		S
2	1	1	Cumings, Mrs. John Bradley	female	38	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen, Miss. Laina	female	26	0	0	STON/O2. 3101282	7.925		S
4	1	1	Futrelle, Mrs. Jacques Heath	female	35	1	0	113803	53.1	C123	S
5	0	3	Allen, Mr. William Henry	male	35	0	0	373450	8.05		S

Figura 2 – Exemplo de Dados do Titanic - (TITANIC. . . , 2012b)

1. IDPassageiro: Um átomo numérico que identifica unicamente cada passageiro.
2. Sobreviveu: Um átomo numérico que representa se a pessoa sobreviveu ou não ao naufrágio. Assume valor "1"para sobreviventes e "0"para não sobreviventes.
3. Pclasse: Átomo numérico que varia de "1"a "3", representa a classe do passageiro. "1"equivale a primeira classe, "2"a segunda classe e "3"a terceira.
4. Nome: Átomo categórico que representa o nome do passageiro.
5. Sexo: Átomo categórico que representa o sexo do passageiro.
6. Idade: Átomo numérico que representa a idade do passageiro.
7. IrConj: Átomo numérico que representa a soma da quantidade de irmãos e cônjuges do passageiro a bordo.
8. PaiFilh: Átomo numérico que representa a soma da quantidade de pais e filhos do passageiro a bordo.
9. Ticket: Átomo categórico que representa o número do ticket de embarque do passageiro.
10. Taxa: Átomo numérico que representa o preço da passagem paga pelo passageiro.
11. Cabine: Átomo categórico que representa o número da cabine do passageiro.
12. Embarcado: Átomo categórico que representa o porto de embarque do passageiro. "C"para a cidade de Cherbourg, "Q"para a cidade de Queesntown e "S"para Southampton.

É importante notar que nem todos os dados disponíveis são úteis para a resolução do problema. Qual seria a relevância do atributo **IDPassageiro** na resolução do problema? Este atributo é gerado automaticamente e está relacionado com a ordem dos registros no computador e não com uma característica real do passageiro.

A decisão de escolher que dados são úteis para a solução do problema é um trabalho não trivial, que exige conhecimento do contexto do problema e repetidas iterações de otimização. Conhecendo o problema, é possível descartar informações que não são relevantes, e, tendo um modelo pronto, é possível testar se determinada informação afeta ou não a performance da predição. Diminuir a quantidade de informações que entram no modelo pode ajudar no tempo em que as predições são feitas e também na performance do sistema em geral. Isto se deve ao fato de que, ter informações que não se relacionam com a variável de interesse, aumenta o ruído do sistema, a figura 3 demonstra este fluxo de trabalho (BRINK; RICHARDS, 2013).

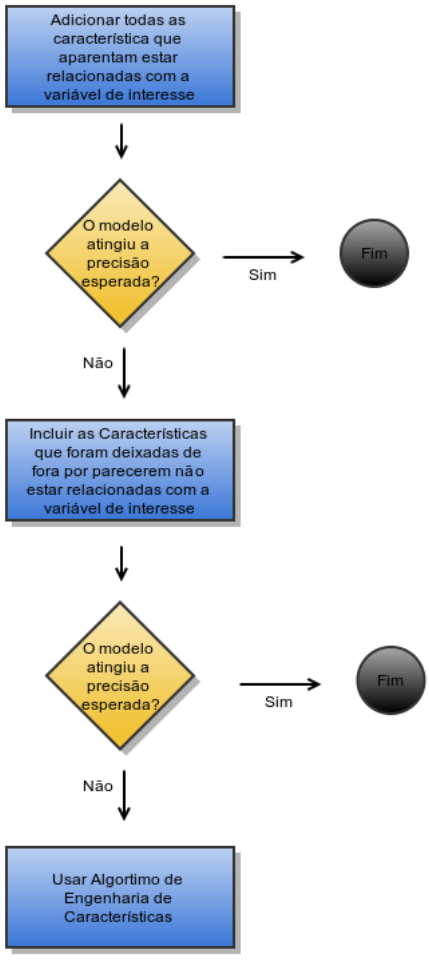

Figura 3 – Fluxo de Trabalho da Seleção de Características

Se houver dúvida sobre a relação causal entre determinada variável e a saída do sistema, é possível realizar duas execuções do modelo: Uma com a variável independente e

outra sem. Se não houver melhora na predição, este é um sinal de que a informação pode ser descartada. Caso haja queda na precisão, é um indício de que a variável independente não tem relação causal com a variável de interesse.

Outro fator relevante na Aquisição de Dados é que a maioria dos modelos só aceitam entradas numéricas ou categóricas. Portanto, é necessário transformar outros tipos de dados em entradas ideais por um processo chamado de Engenharia de Características, este processo será descrito à frente (BRINK; RICHARDS, 2013).

Como é possível observar na pequena seleção da base de dados do Titanic, coluna 11 (Cabine) da Figura 2, existem algumas linhas que não possuem seu valor preenchido. Esse fenômeno é chamado de Dados Faltantes, do inglês Missing Data. É comum em diversas bases de dados encontrar informações faltantes. Isto se deve a diversos motivos, como erro na hora da coleta ou perdas que acontecem com o tempo. Estas informações podem ser decisivas para que o modelo aprenda algum relacionamento novo. De acordo com os autores Batista e Monard (2003) (em livre tradução) "Um problema relevante na qualidade dos dados é a presença de dados faltantes"e "... o tratamento de dados faltantes deve ser feito com muito cuidado, de outra forma erro pode ser introduzido no conhecimento aprendido".

Dados faltantes podem ser encontrados por diversos motivos dentro de uma base de dados. Nem todos eles estão relacionados a erros humanos e é importante conhecer a origem de cada tipo de dado faltante para utilizar o método de tratamento correto. A enumeração a seguir demonstra alguns tipos de Dados Faltantes (LITTLE; RUBIN., 1987).

1. Missing Completely At Random (MCAR): Perda completamente ao acaso. É o maior nível de randomicidade. Acontece quando a probabilidade de perda em um caso pode não depender nem dos valores que esse atributo assume, nem do dado perdido em si. É naturalmente randômico e qualquer método de tratamento pode ser utilizado sem o risco de introduzir erro no aprendizado.

2. Missing At Random (MAR): Perda ao acaso. Quando a probabilidade de perda pode depender dos valores que o atributo pode assumir, mas não pode depender do dado perdido em si.

3. Not Missing At Random (NMAR): Perda ao não acaso. Neste caso pode existir algum motivo real para que a instância esteja perdida. A probabilidade de perda pode depender do valor que a instância do dado assume no momento.

Muitos métodos são propostos para lidar com este problema, ainda de acordo com R. J. Little e D. B. Rubin apud Batista e Monard (2003), os métodos de tratamento para dados faltantes se enquadram em uma das três categorias a seguir:

1. Ignorar e descartar o dado: Consiste em descartar o dados faltantes. A primeira forma de fazer isso é descartar toda a tupla que contém algum dado faltante. Outra forma é avaliar a quantidade de dados faltantes em cada tupla e em cada coluna, depois disso eliminar as instâncias(tupla ou coluna) de acordo com extensão dos dados faltantes. Antes de deletar qualquer tupla ou atributo é necessário analisar a importância desse dado na performance geral do modelo. Atributos que são relevantes devem ser mantidos mesmo se seu nível de perda for muito alto. Este método só deve ser usado quando os dados faltantes são do tipo MCAR.

2. Estimação de Parâmetro: Neste caso todas as instâncias parecidas são analisadas e o dado faltante tem seu valor atribuído com um valor que possui alta similaridade com seus vizinhos.

3. Imputação: Consiste em uma série de métodos que visam atribuir valores faltantes com estimações. Busca-se encontrar relacionamentos entre o sistema e os dados faltantes, pode-se dizer que um novo processo de AM é empregado para estimar estes valores.

2.2 Engenharia de Características

Características englobam informações de dados brutos que habilitam algoritmos de aprendizado de máquina a classificar um objeto desconhecido ou fazer uma estimativa nova (ANDERSON et al., 2013). Características não são simplesmente conjuntos de dados, pois elas carregam mais informações sobre o contexto do problema. São abstrações que estruturam conjuntos de informação, para que estes fiquem mais próximos a conhecimentos do mundo real.

Engenharia de Características, EC, é o processo de transformar dados nunca trabalhados em características que melhor representam o problema atacado para o modelo preditivo, resultando em uma precisão de modelo melhorada nos dados escondidos (BROWNLEE, 2014). Em uma outra definição, "... engenharia de características é projetar manualmente como os dados de entrada x devem ser"(MALISIEWICZ, 2014). Existem tipos de dados que precisam ser trabalhados antes de serem utilizados por modelos. Dados do tipo sequencial e do tipo categórico geralmente precisam ser tratados, pois há modelos que não estão preparados para recebe-los em sua forma inicial. Portanto, dois principais processos são feitos na base de informações inicial: análise dos dados, buscando encontrar características que não estão explícitas e transformações de dados explícitos em formatos que sejam compatíveis com o modelo escolhido. EC visa responder à pergunta: Qual a melhor forma de representar o problema pelos meus dados?

No fluxograma do começo deste capítulo, Engenharia de Características aparece como uma etapa anterior à fase de modelagem e como um processo que serve para me-

lhorar a acurácia da modelagem. EC está no centro do Aprendizado de Máquina, pois através dela é possível entender os dados por suas características. Ela ajuda a dominar o contexto do problema, o que agrega conhecimento no modelo de AM, aumentando assim a performance das modelagens. Por esse motivo, EC aparece em vários momentos do fluxo de Aprendizado de Máquina. Inicialmente é necessário tratar os dados para começar a modelagem, e, posteriormente, Engenharia de Características é usada para melhorar o desempenho do modelo até que se chegue a uma precisão aceitável.

O fluxo da Engenharia de Características é iterativo por si só. Ele consiste em construir novas características e testar o desempenho do modelo com elas (BROWNLEE, 2014). A figura 4 descreve este fluxo:

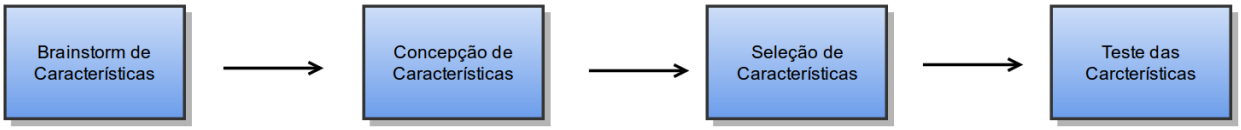

Figura 4 – Fluxo de Trabalho de Engenharia de Características

1. Brainstorm de Características: Nesta etapa um conjunto de possíveis características deve ser gerado.É necessário um aprofundamento no problema, através de meticulosa inspeção dos dados disponíveis e de experiências em casos similares.

2. Concepção de Características: Consiste em construir características. É possível utilizar métodos automatizados de extração de características ou é possível criá-las manualmente através do conhecimento obtido na etapa anterior.

3. Seleção de Características: Nesta fase é feita uma escolha de um conjunto de características para serem testadas. É possível utilizar métodos de comparação entre as características existentes ou é possível escolher o conjunto manualmente, de acordo com o que se conhece do problema.

4. Teste das Características: A última etapa é simples: basta introduzir as características no modelo e medir o desempenho.

Usar Engenharia de Características sistematicamente pode aumentar a capacidade preditiva dos modelos, pois características novas são conhecimentos que poderiam não ser aprendidos naturalmente pelos algoritmos de AM. EC possibilita: Criar características que são mais relacionadas com a variável de interesse, permite a adição de informações externas relevantes dentro do modelo, habilita o uso de informações não estruturadas dentro dos modelos de AM e seleciona as características mais relevantes para a solução do problema (BRINK; RICHARDS, 2013).

2.2.1 Decomposição de Atributos Categóricos

Atributos categóricos assumem apenas um valor dentro de um conjunto de possibilidades de cada vez. Em uma tabela convencional apenas uma coluna é capaz de descrever este comportamento. Temos do exemplo do Titanic a coluna Sexo:

IdPassageiro	Sobreviveu	Pclasse	Nome	Sexo	Idade	IrConj	PaiFilh	Ticket	Taixa	Cabine	Embarcado
1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171	7.25		S
2	1	1	Cumings, Mrs. John Bradley	female	38	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen, Miss. Laina	female	26	0	0	STON/O2. 3101282	7.925		S
4	1	1	Futrelle, Mrs. Jacques Heath	female	35	1	0	113803	53.1	C123	S
5	0	3	Allen, Mr. William Henry	male	35	0	0	373450	8.05		S

Figura 5 – Exemplo de Dados do Titanic - (TITANIC..., 2012b)

Cada pessoa pode ter como valor nesta coluna "male"para o sexo masculino e "female"para o sexo feminino. Muitos modelos não estão preparados para lidar com este tipo de dado. Em geral, modelos (algoritmos lineares, árvores de decisão...) precisam de informações numéricas atômicas para atuar de forma eficiente. A solução neste caso é criar mais categorias, uma coluna nova para cada valor de categoria que é possível assumir. No exemplo do Titanic, é necessário abolir a coluna sexo e criar duas novas colunas: "émasculino"e "éfeminino". Estas colunas assumem um valor binário e são inversamente complementares. Isto quer dizer que se o registro diz respeito a um homem ele terá o valor "1"na coluna "émasculino"e o valor "0"na coluna "éfeminino". A mesma lógica segue para quaisquer outros tipos de categoria: o valor "1"para o atributo desejado e o valor "0"para todas as outras possibilidades.

2.2.2 Decomposição de Características

Muitos bancos de dados possuem informação de hora em um formato de sequência de caracteres. Um exemplo de padrão é o ISO 8601, na forma: 2014-09-20T20:45:40Z. Existe muito conhecimento condensado em apenas uma sequência de letras. Através de EC é possível quebrar esta sequência em muitas outras colunas, tornando o conhecimento mais claro e aumentando a capacidade de aprendizagem do modelo. Este padrão pode ser quebrado em várias outras características, como: ano, mês, dia, hora, minuto, segundo e turno do dia. Este método é comumente aplicado para vídeos e imagens, que são sequências de informação. Este método consiste na decomposição de uma característica complexa em várias outras mais simples.

2.2.3 Características Derivadas

Outro tipo de conhecimento que pode ser extraído vem da precisão de átomos numéricos. Em determinado banco de dados pode haver a informação da massa de um produto. Esta massa pode ser medida em várias casas decimais de gramas ou pode estar arredondada em valores inteiros de quilogramas. Ter estas informações separadas pode

servir como catalisador da precisão do modelo, o processo de teste das características dirá se esta é uma boa característica ou não. Também é possível criar um atributo do tipo booleano para um limite de peso, isto é, se o produto possui mais do que cinco quilos recebe valor "1"nesta característica, se a massa for menor, recebe valor "0".

É necessário ainda, o cuidado com a seleção das características. Nem sempre uma grande quantidade de características é a melhor opção para o modelo, é preciso selecionar os atributos mais relevantes para aumentar a acurácia da previsão. Mais características inseridas no modelo dão a capacidade de aprender novos relacionamentos. A maioria dos modelos possui uma etapa de treino, onde o conhecimento é aprendido e absorvido pelo algoritmo, e uma etapa de teste, onde o algoritmo usa o conhecimento aprendido para realizar predições no conjunto de testes. Se muitas características do conjunto de treino forem colocadas no modelo ele poderá ser incapaz de generalizar conhecimentos e atuar de forma eficaz no conjunto de testes. O nome deste fenômeno é sobre-ajustamento, do Inglês overfitting. Sobre-ajustamento gera resultados superestimados em modelos: predições que aparecem em um modelo, que é sobre-ajustado com base no conjunto de treinamento, não existem na realidade e por isso não se replicaram no conjunto de testes (ABYAK, 2013). A figura 6 mostra um exemplo de sobre-ajuste.

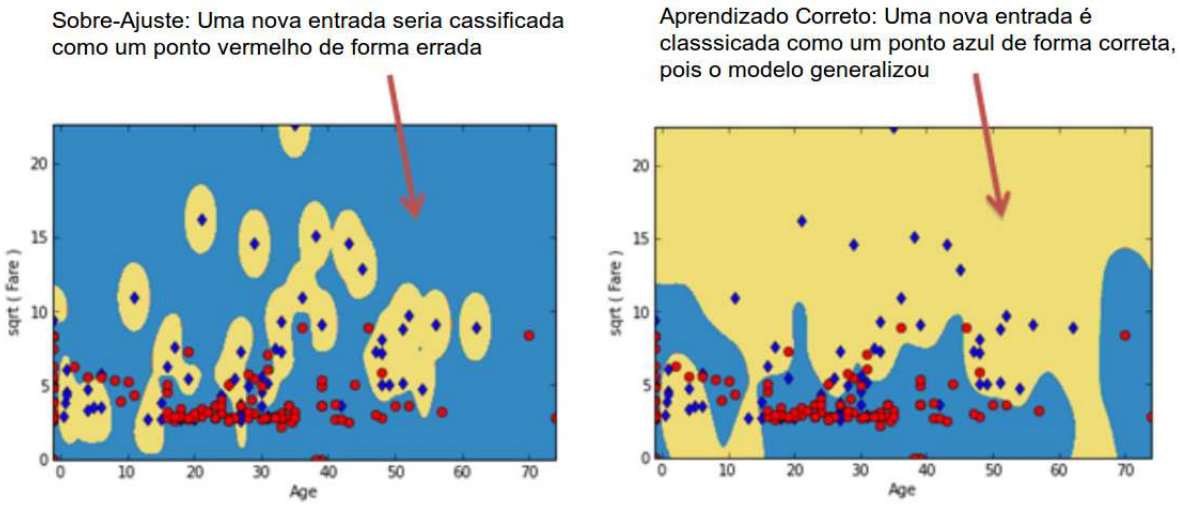

Figura 6 – Exemplo de Sobre-Ajustamento - (BRINK; RICHARDS, 2013)

No gráfico da esquerda, o modelo sobre-ajustou. Ele aprendeu características que são exclusivas do conjunto de treinamento. Provavelmente o modelo abstraiu ruídos do aprendizado. Do lado direito há um exemplo de aprendizado correto, o treinamento serviu para aprender características do mundo real e na aplicação houve generalização correta. É por este motivo que a seleção das características ideais é tão importante.

1. Força Bruta: O jeito mais simples de selecionar características é simplesmente seguir o fluxo de trabalho que foi apresentado no começo deste capítulo de forma manual.

Escolher um conjunto de características e testar a validade destas no modelo. Uma das vantagens dos algoritmos de AM é a capacidade de lidar com um grande número de características. O problema deste método é que fazer a seleção de um grande número de características manualmente torna-se rapidamente inviável.

2. Algoritmos de Seleção: Existem alguns algoritmos que são capazes de relacionar características com a variável de interesse. Estes modelos possuem a seleção de características incorporada em sua concepção e são capazes de escolher as características mais relevantes. Embora estes métodos não sejam aplicáveis a todos os casos, pode ser interessante usá-los como um norteador para realizar a seleção por outras vias.

3. Adição Iterativa: Este método consiste em partir de um conjunto vazio de características e adicionar todas as características que foram levantadas de forma sistemática, de uma a uma, e testar a precisão do modelo. A parada só acontece quando todas as características foram adicionadas, ou a precisão desejada foi alcançada ou o conjunto de características chegou a um tamanho máximo pré-determinado.

4. Remoção Iterativa: Este método parte de um conjunto que contém todas as características levantas, remove uma de cada vez e a cada iteração e testa a performance do modelo. A parada só acontece quando todas as características foram removidas, ou a precisão desejada foi alcançada ou o conjunto de características chegou a um tamanho mínimo pré-determinado.

Seleção de características pode ser usada não apenas para evitar sobre-ajustamento e deixar o modelo mais leve. É possível construir um modelo de AM apenas para chegar à etapa de seleção das características, pois nesta etapa é possível obter conhecimento valioso sobre o mundo real: Quais as características tem maior relacionamento com a variável de interesse. Apenas esse conhecimento já é suficiente para aplicar grandes mudanças no mundo real. Usando uma base que relaciona características com pacientes de câncer, é possível descobrir quais as características que são mais decisivas para o desenvolvimento de um tipo de câncer (BRINK; RICHARDS, 2013).

2.3 Construção do modelo

Depois de compreender o contexto do problema e passar um tempo definindo características na base de dados, é hora de construir um modelo de predição apropriado. AM busca encontrar relacionamentos e padrões que estão dentro do conjunto de dados. Através de métodos matemáticos e computacionais é possível realizar esta tarefa. O processo de descoberta dos conhecimentos ocultos nos dados é atingido através do uso do modelo. Nesta etapa de construção do modelo, é necessário aplicar o que foi descoberto sobre as características dos dados e sobre os objetivos da solução para escolher o modelo adequado.

Existem vários métodos que podem ser utilizados para revelar os relacionamentos ocultos nos dados. Os métodos podem ser simples ou complexos, indo desde regressões lineares até redes neurais que possuem centenas de neurônios. O objetivo desta seção é descrever características de diversos tipos de modelos, a relação entre necessidades do problema e características dos modelos é que indica os possíveis métodos adequados para atingir a solução.

A seguinte equação serve muito bem para ilustrar o objetivo de um modelo de AM: $\mathbf{Y} = \mathbf{f(X)} + \mathbf{e}$. $\mathbf{Y}$ corresponde às predições, que é o resultado final do uso do modelo. $\mathbf{X}$ corresponde a todas as entradas que são usadas no modelo, estas informações vem da base de dados e já passaram pela etapa de pré-processamento e Engenharia de Características nas fases anteriores. O conjunto $\mathbf{X}$ são os registros reais, os métodos de AM são capazes de utilizar estas informações e extrair conhecimentos relevantes para a estimação de $\mathbf{Y}$. Estes métodos são representados por $\mathbf{f(X)}$, onde $\mathbf{f()}$ é a técnica de modelagem que gera as predições baseadas nas entradas do sistema. O último elemento da equação é o erro $\mathbf{e}$. Este erro determina a precisão da predição, de forma que, se o erro chegar a 0 a precisão é total. Este erro é advindo de várias fontes, algumas delas são: Medições imperfeitas, dados faltantes e super ajustamento (BRINK; RICHARDS, 2013).

Depois de ter uma boa estimativa para a função $\mathbf{f(X)}$, é possível realizar duas tarefas distintas: predição e inferência. Predições são o resultado direto da aplicação do modelo. Quando coloca-se um novo valor de $\mathbf{X}$, que nunca foi visto antes, como entrada do modelo, a resposta da equação é equivalente a uma predição da realidade. Inferência é algo mais profundo, é entender os motivos dos relacionamentos entre as entradas e as saídas do modelo. Como citado anteriormente, Engenharia de Características pode ser usada para realizar inferências, pois uma inferência diz o quanto uma variável relaciona-se com o resultado final (BRINK; RICHARDS, 2013).

Existem dois tipos de modelos de Aprendizado de Máquina, paramétrico e não-paramétrico. A diferença entre eles é que os modelos paramétricos assumem que a solução possui uma forma pré-definida, isto é, estes modelos tentam descobrir o formato da função $\mathbf{f(X)}$ assumindo que ela possui um formato de curva previamente conhecido. Um modelo caracteristicamente paramétrico é a regressão linear, pois este método assume que a função $\mathbf{f(X)}$ é uma combinação linear do vetor de entrada, $\mathbf{X}$, com parâmetros numéricos. Métodos não-paramétricos não fazem nenhuma pré suposição com relação ao formato geométrico da função de estimativa, estes métodos mapeiam os parâmetros de entrada às saídas. Métodos paramétricos geralmente são mais fáceis de interpretar, mas possuem performance inferior a métodos não paramétricos. Métodos não-paramétricos são difíceis de interpretar, mas são uma forma de solução mais direta e não fazem pré-suposições, que podem estar erradas (NG; JORDAN, 2014).

Problemas de Aprendizado de Máquina podem cair em duas categorias: Super-

visionado e Não-Supervisionado. No caso supervisionado, o modelo de AM recebe um conjunto de dados de treinamento rotulado e faz predições para pontos que não possuem rótulo. Neste caso, o algoritmo utiliza o rótulo de cada entrada para refinar o aprendizado, fazendo com que as características de cada entrada sejam relacionadas com o rótulo que ela possui. No caso de aprendizado não-supervisionado, o modelo recebe apenas dados que não possuem rótulo. Neste caso, o modelo é capaz de agrupar os dados de entrada em categorias, mas ele não possui conhecimento para afirmar o que estes grupos são no mundo real (MOHR; ROSTAMIZADEH; TALWALKAR, 2012). A vantagem da utilização de aprendizagem supervisionada é que há conhecimento extra inserido por especialistas, isto dá mais sentido aos modelos deixando claro o que cada previsão representa no mundo real. As vezes os rótulos que acompanham os dados podem estar errados, e isto aumenta o ruído do sistema. Uma das vantagens da abordagem não supervisionada é não depender de rótulos, diminuindo o erro que vem de entradas classificadas de maneira errônea.

Existem duas principais abordagens para aprendizado não-supervisionado: agrupamento e redução de dimensionalidade. O agrupamento consiste em descobrir qual é a maneira natural que os dados de entrada se agrupam, basicamente o que o modelo faz é unir as entradas em grupos distintos de acordo com a semelhança que eles possuem entre si. A redução de dimensionalidade consiste em diminuir a complexidade das entradas. Isto é atingido quando características mais relevantes são selecionadas de acordo com a variabilidade dos dados.

Os modelos de AM realizam predições de duas formas distintas: classificação e regressão. Classificação é quando o resultado final do modelo de AM é um conjunto de saídas categóricas. Neste caso as entradas são analisadas pelo modelo e classificadas de acordo com o que foi aprendido. Um exemplo deste tipo de objetivo é o que se pode fazer com a base de dados do Titanic que foi previamente apresentada. É possível analisar as características de cada passageiro e classificá-los entre sobreviventes e mortos. Quando regressão é o objetivo final da previsão, então o modelo irá resultar em um valor contínuo. Nestes casos o aprendizado do modelo se deu na forma de uma função, e o que ele faz é computar a entrada X para gerar uma saída Y, na forma $Y = f(X) + e$.

A figura 7 lista os tipos mais comuns de algoritmos de Aprendizado de Máquina e suas características.

2.3.1 Redes Neurais

As redes neurais merecem um detalhamento a mais em comparação aos outros métodos, pois são tipos de algoritmos largamente utilizados pela comunidade e possuem grande robustez de processamento. A criação das redes neurais artificias é fortemente baseada no fato de que sistemas de aprendizagem biológicos, isto é, presente nos seres vivos, são construídos através de enormes teias de neurônios inter-conectados. Basicamente, re-

Nome	Tipo	Parametrização
Regressão Linear	Regressão	Paramétrico
Regressão Logística	Classificação	Paramétrico
SVM	Classificação/Regressão	Paramétrico
SVM com Kernel	Classificação/Regressão	Não Paramétrico
K Vizinhos mais próximos	Classificação/Regressão	Não Paramétrico
Árvores de decisão	Classificação/Regressão	Não Paramétrico
Floresta Randômica	Classificação/Regressão	Não Paramétrico
Boosting	Classificação/Regressão	Não Paramétrico
Rede Neural	Classificação/Regressão	Não Paramétrico

Figura 7 – Algoritmos de AM - (BRINK; RICHARDS, 2013)

des neurais artificiais são um conjunto de pequenas unidades de processamento, que são apresentadas a entradas e geram uma saída, inter-conectadas. Isto dá capacidade a estas redes de capturar pequenas especificidades, que são relevantes para a solução como um todo, além de possuírem alta capacidade de processamento paralelo (MITCHELL, 1997).

2.4 Avaliação e Otimização de Modelo

Encontrar uma solução para um problema através de Aprendizado de Máquina demanda tempo e algumas iterações para ajustes no modelo. É necessário construir um modelo e testar sua performance antes de começar a usá-lo no mundo real. Este capítulo consiste em métodos de avaliação de acurácia de modelos. É necessário medir a performance dos modelos de AM para saber se a precisão atingiu níveis aceitáveis, caso não, é necessário continuar o refinamento da modelagem até que o objetivo seja alcançado.

O primeiro passo para resolver o problema da avaliação de performance é a definição de uma métrica que seja relevante ao método aplicado. Para controlar algo é necessário medir primeiro (HARRINGTON, 2014). Em uma abordagem supervisionada o objetivo desta métrica deve ser o de comparar o resultado da previsão com o valor especificado no rótulo. Em abordagens não supervisionadas, o objetivo desta métrica pode ser o de avaliar a coesão dentro e fora dos grupos criados (LEARN, 2014).

Para abordagens supervisionadas é possível medir a eficácia de um modelo, considerando apenas as informações que entraram no sistema, isto é, avaliar se o modelo foi capaz de aprender os relacionamentos que foram apresentados. Neste caso, não há

preocupação com a generalização do modelo, a avaliação é feita apenas com dados que o modelo já viu e não com um conjunto novo. Este método é chamado de Residuais. Ele consiste em utilizar o modelo para fazer previsões com os dados que foram usados para o treino, às previsões são comparadas com os rótulos que já estão associados as entradas. Se o modelo for classificador, registra-se a quantidade de entradas classificadas de forma errada e a precisão é dada pela porcentagem de erros totais. Se o modelo for um regressor, cada previsão é comparada quantitativamente com o rótulo criando-se um erro numérico. Nestes casos, a métrica de qualidade é a média de todos os erros (SCHNEIDER, 1997b).

Como já foi citado anteriormente, um problema que pode acontecer com algoritmos de AM é o sobre-ajustamento. Quando isto acontece, o modelo aprende as peculiaridades inerentes ao conjunto de treinamento tão bem, que captura alguns conhecimentos que são exclusivos do conjunto de treinamento. Logo, o modelo absorve em sua morfologia características que não existem no mundo real e por isso terá uma precisão ruim quando apresentado a entradas que nunca foram vistas.

Modelos de AM são treinados em um conjunto especial de dados, geralmente este conjunto é uma pequena amostra do que se encontra no mundo real. Os algoritmos de AM são capazes de aprender conhecimentos através de um conjunto de dados, o desafio é garantir que este aprendizado seja relevante para todos os dados do mundo real. Avaliar a performance de um modelo de Aprendizado de Máquina é descobrir qual a capacidade desse modelo de generalizar, ou seja, ser útil em fazer previsões quando apresentado a um conjunto de dados que nunca foi visto.

Para modelos supervisionados, buscando uma solução para o problema da medição de precisão para conjunto de dados que nunca foram vistos pelo sistema, existe o método chamado de Validação Cruzada. Este método consiste em não usar todo o conjunto de treino para a construção do modelo, uma parte do conjunto é separada exclusivamente para testar a performance do modelo. Basicamente, utiliza-se a técnica de Residuais com o conjunto que foi separado especificamente para o teste. Isto evita os erros de sobre-ajustamento que seriam encontrados no método de Residuais (SCHNEIDER, 1997a). Existem dois tipos de Validação Cruzada, Holdout e K-partes.

O método de Validação Cruzada por Holdout, também conhecido como estimação por conjunto de teste, consiste em dividir a base de dados em dois conjuntos mutuamente excludentes chamados conjunto de treinamento e conjunto de testes, ou conjunto de holdout. É comum separar 2/3 dos dados para o conjunto de testes e 1/3 para o treinamento. O conjunto de treino é usado pelo modelo de AM para aprendizado, é deste conjunto que todo o conhecimento do modelo será retirado. Após o treinamento, o conjunto de testes é utilizado para realizar o teste do tipo Residual sobre o sistema. Quanto maior for o número de dados que são deixados para o teste, menor será a capacidade de generalização do modelo, mas um conjunto de testes pequeno diminui a confiança na capacidade preditiva

do modelo (KOHAVI, 1995). A figura 8 demonstra o processo de holdout.

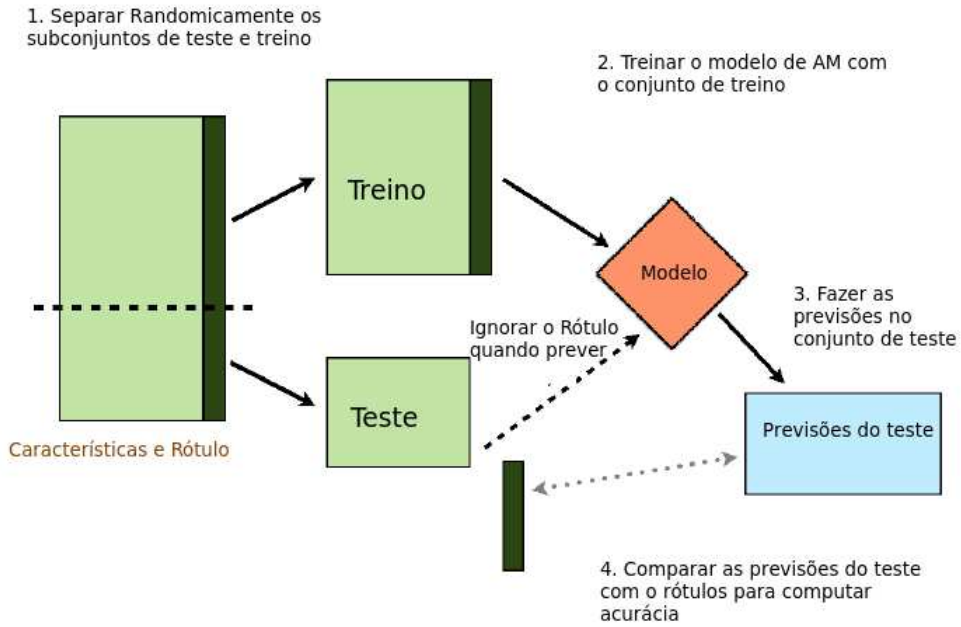

Figura 8 – Crossover - Holdout - (BRINK; RICHARDS, 2013)

Uma outra forma de fazer a validação cruzada é através do método de K-partes, também conhecido como estimação rotativa. O conjunto de dados é dividido em K partes mutuamente excludentes de tamanho aproximadamente igual. O modelo é treinado e testado k vezes. O treino é realizado com todos os sub-conjuntos menos um de cada vez. O sub-conjunto que foi deixado de fora é então utilizado como conjunto de teste. Na próxima iteração, o modelo será treinado novamente e o próximo sub-conjunto será deixado de fora do treino e utilizado para a realização do teste. Todas as previsões dos testes são guardadas em um vetor e no fim de todas as iterações os rótulos são comparadas com o vetor que possui as previsões (KOHAVI, 1995). Este método é melhor para estimar o erro no conjunto real, porém é computacionalmente mais pesado e nem sempre viável. A figura 9 demostra o processo da validação cruzada utilizando k-partes.

Um outro método eficaz para medição de performance é a Matriz de Confusão. Este método serve para mostrar se um algoritmo está confundindo a classificação das entradas, portanto é um método aplicável a aprendizagem supervisionada. A matriz contém informações sobre os rótulos e sobre as predições realizadas pelo modelo. A performance é medida pelas informações que estão contidas na matriz e ela pode ser usada para qualquer número de classes. A tabela a seguir mostra como funciona uma matriz de confusão para um sistema que possui apenas duas classes, negativo e positivo (HAMILTON, 2012a).

O significado de cada elemento dentro da matriz é:

1. *a* corresponde a quantidade de predições **corretas** da classe **negativa**

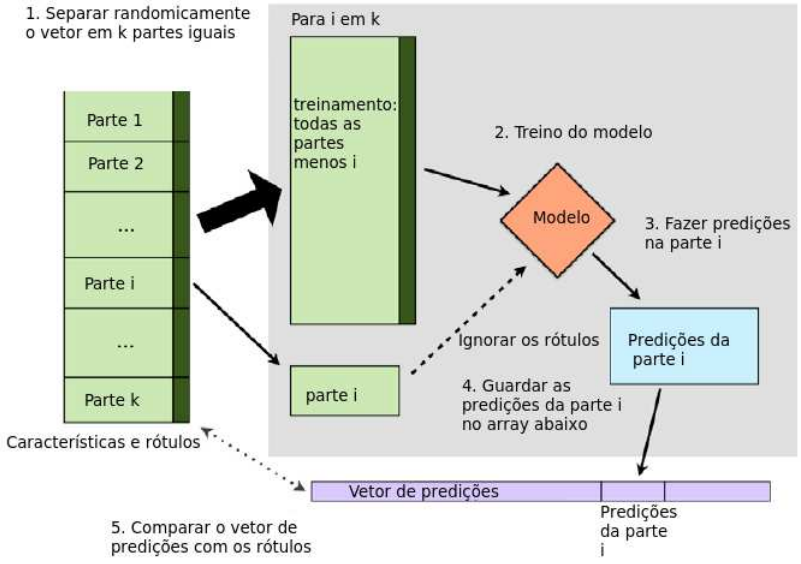

Figura 9 – Crossover - K partes - (BRINK; RICHARDS, 2013)

		Previsão	
		Negativo	Positivo
Rótulo	Negativo	a	b
	Positivo	c	d

Tabela 2 – Matriz de Confusão

2. b corresponde a quantidade de predições **incorretas** da classe **positiva**
3. c corresponde a quantidade de predições **incorretas** da classe **negativa**
4. d corresponde a quantidade de predições **corretas** da classe **positiva**

É possível retirar várias métricas desta matriz, tais como:

1. A acurácia (AC) é a quantidade de classificações corretas dividida por todas as possibilidades, pode ser medida através da seguinte fórmula:

$$AC = \frac{a+d}{a+b+c+d} \tag{2.1}$$

2. A taxa de verdadeiro positivos (TP) é a proporção de verdadeiros positivos que foram corretamente identificados, pode ser medida por:

$$TP = \frac{d}{c+d} \tag{2.2}$$

3. A taxa de falsos positivos (FP) é a proporção de falsos positivos que foram incorretamente identificados, pode ser medida por:

$$FP = \frac{b}{a+b} \tag{2.3}$$

4. A taxa de verdadeiros negativos (VN) é a proporção de verdadeiros negativos que foram corretamente identificados, pode ser medida por:

$$VN = \frac{a}{a+b} \tag{2.4}$$

5. A taxa de falsos negativos (FN) é a proporção de falsos negativos que foram incorretamente identificados, pode ser medida por:

$$FN = \frac{c}{c+d} \tag{2.5}$$

6. A precisão (P) é a proporção de casos positivos que foram corretamente classificados, pode ser calculada por:

$$P = \frac{d}{d+b} \tag{2.6}$$

É importante notar que a acurácia pode não ser uma boa métrica quando o número de casos negativos é muito maior do que o os casos positivos ou vice e versa. Neste caso, o sistema pode classificar todos os casos como pertencente a apenas uma classe, como a quantidade de casos verdadeiros desta classe é alto a acurácia será alta, porém o sistema falhou em classificar uma classe inteira! Por isso é importante observar as outras métricas em conjunto com a acurácia.

Muitos modelos de AM possuem suas saídas dadas em uma probabilidade, isto é, o modelo afirma com uma porcentagem de certeza que a entrada pertence a uma determinada classe. Existe um outro método de medir a acurácia de um modelo que encapsula todas as informações da matriz de confusão de forma gráfica e toma proveito das probabilidades das saídas. Este método é a Característica Operativa do Receptor, do Inglês, Receiver Operating Characteristic (ROC). Um gráfico ROC é uma plotagem com a taxa de falsos positivos no eixo X e taxa de verdadeiros positivos no eixo Y. Os eixos representam probabilidades e vão de zero a um. O ponto (0,1) é o classificador perfeito, que é capaz de classificar corretamente todos os casos, positivos ou negativos.Uma curva ROC é capaz de mostrar a troca entre a habilidade de classificar corretamente os casos positivos e os casos negativos que são incorretamente classificados. É possível analisar o nível de confiança do modelo de acordo com a curva. A área a baixo da curva ROC pode ser usada para medir a acurácia de vários modelos. A figura 10 é um exemplo de gráfico ROC.

Assim como na matriz de confusão, é possível analisar modelos que produzem como saída mais de uma classe. Para isto, basta plotar cada par características no mesmo espaço e avaliar qual a confiança geral de acordo com a média das curvas ROC.

Para modelos que são capazes de realizar regressões, não existe o conceito de predição errada ou predição certa. Para estes casos é possível analisar o quão distantes

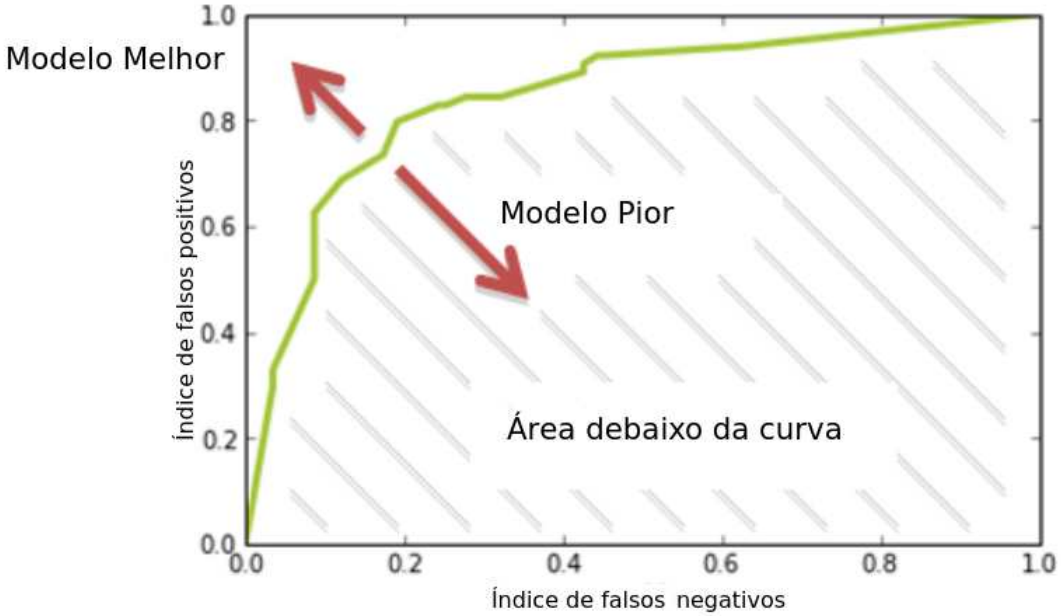

Figura 10 – Curva ROC - (HAMILTON, 2012b)

dos rótulos os valores gerados estão. Portanto, utiliza-se métodos de medição de distâncias entre pontos no espaço geométrico. O método mais simples é o cálculo da média das raízes do erro ao quadrado. Este método calcula o erro para cada uma das predições e faz uma média normalizada através da seguinte equação, yi como rótulo e f(xi) como previsão (BRINK; RICHARDS, 2013):

$$\frac{1}{\sqrt{n}}\sqrt{\sum \left[yi - f\left(xi\right)\right]^2} \tag{2.7}$$

Existem muitos modelos de AM que possuem um número de parâmetros que podem ser alterados, estes parâmetros são capazes de configurar diferentes comportamentos nos algoritmos e são únicos para cada contexto de uso. Existem métodos para escolher o parâmetros de forma ótima, sendo o caso mais comum, a utilização de força bruta.

3 Aprendizado Incremental

Um outro desafio que emerge da grande quantidade de informações que são geradas é, o tempo de validade dos modelos de AM. Atualmente, a maioria dos modelos de aprendizagem possuem uma etapa de treino que acontece antes da etapa de uso efetivo dele. Esta etapa serve para que o sistema aprenda as características do espaço do problema, para depois ser capaz de atuar nos dados que vem sendo gerados em tempo real. O processo de construção de um sistema especialista efetivo geralmente demanda muito tempo e trabalho. É possível que estes sistemas se tornem menos precisos ou ineficazes em pouco tempo, pois a geração de dados no contexto do problema pode ser muito rápida e novas características podem emergir dos registros mais recentes. Se estes modelos de AM só conseguem aprender na etapa de treino, há um problema, pois eles não serão capazes de lidar com conceitos de aprendizagem que aparecerão somente nas informações mais novas.

Para aumentar o tempo de validade dos sistemas de Aprendizagem de Máquina, é possível usar Aprendizagem Incremental. Isto consiste em capacitar modelos de AM a aprender continuamente, conforme novas entradas chegam ao sistema. Desta forma, mesmo que uma característica surja somente nos registros mais recentes, o sistema será capaz de aprender novamente e atuar de forma efetiva neste novo contexto. É importante ressaltar que o agente especialista deve ser capaz de guardar conhecimento na forma de inteligência no sistema, isto é, ele não tem acesso aos dados que já passaram por ele, mas tem em sua morfologia o conhecimento necessário que foi aprendido quando estes dados passaram por ele. Essa necessidade vem do fato de que é muito custoso armazenar todos os dados que já passaram pelo sistema, pois o volume de informação neste contexto é enorme (HE et al., 2011).

A tendência da necessidade de Aprendizado Incremental é clara, como foi dito na introdução deste trabalho, o volume de informações gerado pela Humanidade tem crescido de forma acelerada e faz-se necessário processar toda esta informação de forma eficaz.As fontes modernas de dados não só são altamente dinâmicas como produzem informação em uma velocidade acelerada. As principais características de um contexto que requer aprendizagem incremental comparado às abordagens tradicionais são (READ et al., 2013):

1. Necessidade de realizar previsões a qualquer momento.
2. A base de dados evolui constantemente com o tempo.
3. É esperado, uma entrada infinita de dados, porém os recursos computacionais são finitos

Para entender aprendizado incremental, é necessário compreender a diferença entre duas formas distintas que modelos de AM podem ter, aprendizado online e por lotes. Aprendizado por lotes é o tipo mais comum de algoritmos que são usados atualmente, eles possuem duas fases claramente distintas, treino e uso. Na fase de treino, o modelo é capaz de aprender os relacionamentos que estão ocultos nos dados e na fase de uso não há aprendizado, há somente predições. Os algoritmos que possuem aprendizado online não possuem essa distinção em suas etapas, eles são capazes de aprender novos conceitos a todo o tempo e o fazem junto com as predições usuais. É esperado que depois da fase de treinamento os algoritmos que possuem aprendizado por lotes tenham criado em suas estruturas internas uma função hipótese que seja capaz de generalizar corretamente em qualquer conjunto de dados daquele contexto. No aprendizado online, não se espera que uma hipótese seja criada para aquele contexto de informações, o algoritmo trata todas as entradas de maneira única, uma por uma, sempre aglomerando os conhecimentos desta entrada (DEKEL, 2009).

Existe um terceiro tipo de aprendizagem que deriva da aprendizagem por lotes. Ela é capaz de resolver os problemas acima citados comparando-se com a aprendizagem online, é a abordagem lote-incremental. Este método consiste em aplicar a abordagem por lotes de tempos em tempos no conjunto de testes. Este método é uma simples renovação de conhecimentos do modelo. Depois de um determinado período de tempo ou volume de dados acumulado, o modelo é novamente treinado com um novo conjunto de dados que incorpora todos os dados anteriores. Esta abordagem não altera os métodos do algoritmo em si, é uma solução de engenharia em que uma revalidação do modelo é feita rotineiramente para incorporar novos conhecimentos. Este método possui algumas desvantagens, como: Ter que desconsiderar todo o conhecimento que já foi aprendido para aprender novamente e não poder aprender dos dados mais recentes até que um novo lote esteja pronto para entrar no sistema (READ et al., 2013).

A aprendizagem lote-incremental não é um algoritmo de AM em si, pois ela é uma estratégia de uso de modelos. Portanto, qualquer algoritmo que possua fase de treino e fase de uso pode estar sujeito a este tipo de aprendizagem. A figura 11 ilustra como este método pode ser empregado.

Os algoritmos online são compostos por uma sequência de iterações. Cada iteração é composta por três etapas: Recebimento da entrada, predição da saída correspondente e recebimento do verdadeiro rótulo da instância. Com a presença do verdadeiro rótulo, o algoritmo é capaz de ajustar suas hipóteses de forma a ajustar sua morfologia de acordo com a diferença entre a predição e o rótulo. O objetivo do algoritmo é minimizar o erro entre a previsão e o rótulo (LITTLESTONE, 1987). O problema desse tipo de algoritmo é que nem sempre é possível obter os rótulos com a mesma velocidade que as entradas chegam, porém em alguns casos isto é possível. Em problemas que envolvem séries temporais

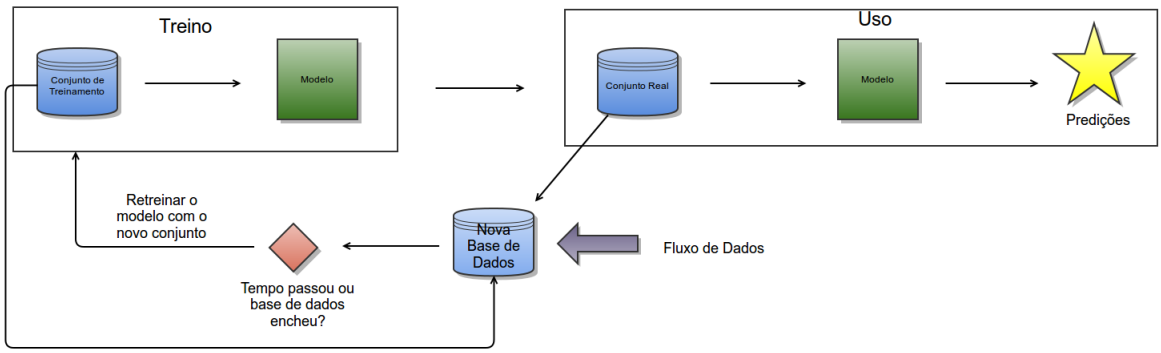

Figura 11 – Fluxo do Aprendizado por Lotes

basta esperar o tempo para verificar qual o resultado aconteceu no mundo real, este é o rótulo. No caso de abordagens não supervisionadas outras métricas de coesão devem ser constantemente verificadas para gerar os ajustes no modelo.

A aprendizagem lote-incremental não se encaixa na definição de aprendizagem incremental defendida por He et al. (2011), pois ele necessita de dados anteriores para aprender novamente. A defesa dos autores do artigo acima citado é que um modelo só pode ser considerado como sendo de aprendizagem incremental se ele não depender dos dados históricos, baseando-se somente nas hipóteses que foram aprendidas anteriormente. Esta defesa é interessante, pois isso dá maior robustez ao modelo. Considerando-se que o contexto para o uso de aprendizagem incremental é uma entrada infinita de dados com recursos computacionais finitos, os algoritmos que dependem de todos os dados históricos incorrem em grande falha, pois o recurso computacional finito neste caso é a capacidade de armazenamento.

A afirmação do parágrafo anterior vem do conhecido dilema de estabilidade/ plasticidade: Um classificador totalmente estável é capaz de reter conhecimento, mas não consegue aprender nova informação, enquanto um classificador totalmente plástico pode aprender novos conhecimentos instantaneamente, mas não consegue reter conhecimentos passados (MUHLBAIER; TOPALIS; POLIKAR, 2004).

Porém, é importante ressaltar que algumas experiências da comunidade de Aprendizado de Máquina indicam que modelos com aprendizagem online muitas vezes possuem performance inferior e complexidade maior quando compara-se com lote-incremental. Muitos problemas podem simplesmente ser resolvidos até mesmo por método de lotes. É importante avaliar o contexto com cuidado para escolher o método apropriado a solução do problema.

Pelos motivos citados anteriormente, outra característica que deve ser avaliada na hora de escolher algoritmos para solucionar problemas no contexto incremental é a supervisão. Há casos onde é possível encontrar alguma verdade que classifica os dados, neste

cenário é possível utilizar soluções da categoria supervisionada. Algoritmos supervisionados utilizam a verdade imposta nos rótulos das amostras para ajustar seus parâmetros de aprendizado na hora do treinamento. É necessário que alguém gaste um tempo para gerar estas classificações para o conjunto de treinamento.

Em um contexto onde os dados de entrada são infinitos e os recursos computacionais finitos, muitas vezes não é possível utilizar algoritmos supervisionados pelo simples motivo de não se ter rótulos nas amostras. Esta dificuldade em gerar um conjunto de treinamento com rótulos se deve pela grande velocidade de geração dos dados e pelo contexto dinâmico dos problemas. Por este motivo as abordagens não supervisionadas são interessantes. Não é necessário ter uma verdade explícita para que estes modelos gerem algum tipo de conhecimento.

Muitos dos algoritmos não supervisionados são do tipo agrupamento. Estes algoritmos utilizam a natureza dos dados para gerar agrupamentos diversos. O próximo capítulo aborda com mais profundidade esta classe de problemas e mostra algoritmos deste tipo.

4 Agrupamento

Agrupamento é usar as características de entrada para descobrir aglomerações naturais nos dados e para dividir os dados nestes grupos (BRINK; RICHARDS, 2013). Ou ainda, particionar itens em regiões homogêneas. Um exemplo aplicado a redes sociais é: encontrar comunidades dentro de um grande grupo de pessoas, Mohr, Rostamizadeh e Talwalkar (2012).

Dentre os algoritmos mais conhecidos de agrupamento estão: K-médias, modelos de mistura gaussianas e clusterização hierárquicas. Um dos métodos mais interessantes para realizar agrupamento é a rede neural SOM (Self Organizing Map, Mapa auto-organizável) desenvolvida por Teuvo Kohonen em 1982 (KOHONEN, 1982). Além de realizar o agrupamento com robustez, o SOM é capaz de operar uma redução de dimensionalidade nos dados, dessa forma não somente a morfologia dos agrupamentos é interessante, mas também o mapa gerado pelo algoritmo. Entretanto este algoritmo ainda apresenta o modelo de lote, possuindo fase de treino e uso bem distintas.

4.1 SOM

É possível pensar o algoritmo SOM como uma combinação de dois sistemas menores. Uma parte funciona como uma rede neural competitiva do tipo o vencedor leva tudo. Nesta etapa um dos neurônios da rede é selecionado de acordo com sua semelhança a amostra de entrada, este neurônio é o único vencedor daquela amostra. O segundo sistema acontece depois que o neurônio é escolhido, agora a rede irá se adaptar para incorporar esse novo aprendizado, o peso do neurônio vencedor e de seus vizinhos é alterado, isto é o que dá plasticidade à rede. De acordo com Teuvo em, Kohonen (1982), o processo auto-organizável do algoritmo pode ser simplificado em quatro etapas:

1. Um vetor de unidades de processamento que recebem estímulos coerentes de um espaço de eventos e formam funções discriminantes simples com base nas entradas.

2. Um mecanismo que compara as funções discriminantes e seleciona a unidade que possui o maior valor.

3. Algum tipo de interação local, que, simultaneamente, ativa a unidade vencedora e seus vizinhos.

4. Um processo adaptativo que faz os parâmetros das unidades ativadas aumentarem seus valores de função discriminante com base na atual entrada.

Na figura 12 pode-se observar a arquitetura de mapa auto-organizado. As variáveis *epsilon* representam as entradas do sistema, elas são comparadas com todas as unidades de processamento *eta*, e o neurônio que for mais parecido é o vencedor daquela amostra. As unidades de processamento também possuem o conceito de vizinhança, quanto mais próximos uma unidade for da outra, mais parecidas elas serão. A redução de dimensionalidade é alcançada pois, ao fim do treinamento, cada unidade de processamento será um represente do conjunto de entradas que ele foi capaz de selecionar. A figura 13 demonstra a saída de uma rede SOM aplicada a um conjunto de cores, é possível observar que a rede foi capaz de reduzir a dimensionalidade e as características semelhantes ficaram próximas umas das outras.

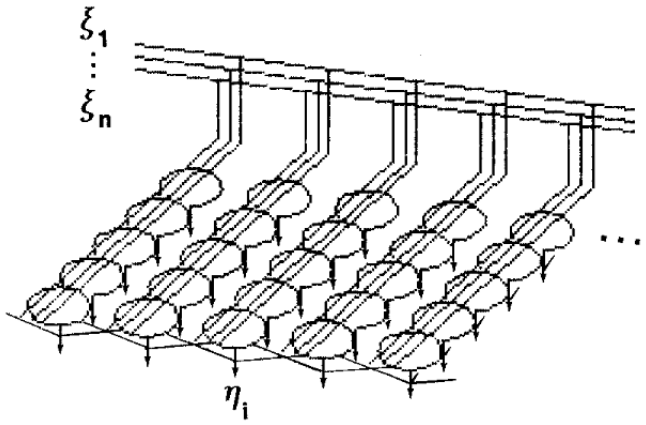

Figura 12 – Sistema que Apresenta um Mapa Organizado - (KOHONEN, 1982)

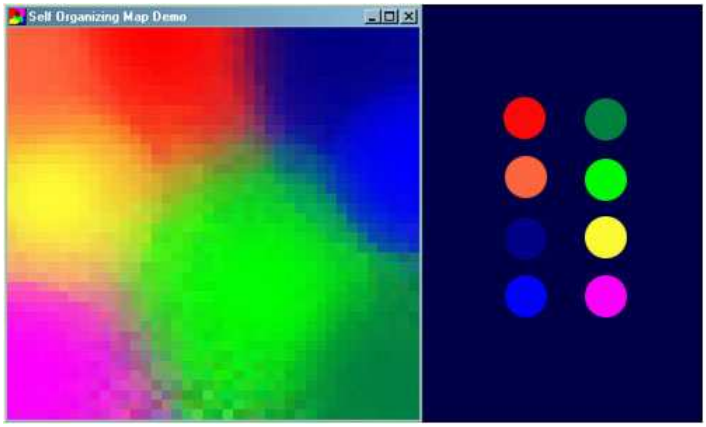

Figura 13 – Exemplo de Redução de Dimensionalidade - (AI. . . , 2015)

A aplicação SOM pode ser separada em duas etapas, treino e uso. Na etapa de treino as entradas são apresentadas, selecionadas e os parâmetros do mapa são ajustados para incorporar o aprendizado. O processo de treino ocorre até que uma de três condições sejam alcançadas. Ou a quantidade de épocas de treino é atingida, ou o taxa de aprendizado chegue a zero ou a rede atinja algum critério de erro pré-estabelecido que seja

aceitável. A etapa de uso é uma variação do treino, as amostras de entrada são apresentadas ao sistema e alguma unidade de processamento será capaz de selecioná-la, a diferença desta etapa para o treino é que não há aprendizado ou ajuste de parâmetro. O SOM é incapaz de aprender continuamente, pois seus parâmetros internos decaem com tempo, chegando a estabilidade total, onde não há mais aprendizado.

Para compreender cada etapa do algoritmo SOM uma rede bidimensional de neurônios será usada como o mapa auto-organizável. Neste caso cada neurônio possui um conjunto de coordenadas (x, y) que servem para posicionar os neurônios em um espaço. Em um caso computacional, pode-se pensar nesse mapa como uma matriz quadrada que possui coordenadas (i, j), i para linha e j para coluna. O conjunto de pesos de cada neurônio possui a mesma dimensionalidade dos dados de entrada. Abaixo serão descritas as etapas do algoritmo em modo de treinamento.

4.1.1 Criação do Mapa e Inicialização dos Pesos

A primeira etapa do algoritmo é a criação do mapa. O tamanho da matriz a ser escolhida é uma parametrização do sistema e diferentes tamanhos podem ser úteis para diferentes soluções. Independentemente do tamanho que for escolhido, todos os neurônios da rede irão ajustar seus pesos para representarem as entradas. É necessário então inciar todos os pesos da rede, de acordo com Teuvo em Kohonen (1982), uma boa forma de realizar esta etapa é atribuindo pesos randomicamente pequenos a todos os conjuntos de pesos dos neurônios da rede.

4.1.2 Escolha do Neurônio Vencedor

Para o cálculo do neurônio vencedor, é necessário comparar o dado de entrada com os pesos de cada um dos neurônios do mapa. Aquele neurônio que for mais parecido com a entrada será escolhido como neurônio vencedor. Para realizar esta comparação é necessário estabelecer um método de medir a distância entre a entrada e os neurônios, o neurônio mais parecido será aquele que apresentar menor distância com a entrada. A equação 4.1 mostra a equação de comparação utilizando a distância Euclidiana onde, $\mathbf{V}$ representa o vetor de entrada e $\mathbf{W}$ é o vetor de pesos do neurônio.

$$Dist = \sqrt{\sum_{i=0}^{n}(V - W)^2} \tag{4.1}$$

4.1.3 Cálculo da Vizinhança

Depois de cada iteração é necessário atualizar os pesos de cada neurônio que estão dentro do raio de vizinhança do neurônio vencedor. Este raio é um atributo do mapa todo,

e é o mesmo para todos os neurônios. De acordo com o número de iterações este raio vai diminuindo, este efeito diminui a plasticidade do mapa de acordo com o tempo, fazendo com o sistema seja capaz de convergir a um estado ideal. A equação de cálculo do raio é definida na equação 4.2 onde, $\sigma 0$ representa a largura inicial do raio que é parametrizado no sistema. λ é uma constante de tempo que determina a velocidade do encolhimento do raio, quanto maior ela for, menos o raio diminuirá com o tempo. E t é número da atual iteração.

$$\sigma(t) = \sigma_0 \exp\Big(-\frac{t}{\lambda}\Big) \qquad t = 1, 2, 3... \tag{4.2}$$

4.1.4 Ajuste dos Pesos

Todos os neurônios dentro do raio de vizinhança, incluindo o neurônio vencedor, terão seus pesos atualizados para se parecerem mais com a amostra que acabou de ser apresentada. O grau dessa atualização diminui de acordo com a distância entre o neurônio vencedor e seu vizinho, quanto mais distante, menor o efeito da atualização. A equação 4.3 torna isso possível onde, t representa o número da iteração atual, $L(t)$ é o fator de aprendizagem, $\mathbf{V}$ representa o vetor de entrada, $\mathbf{W}$ é o vetor de pesos do neurônio atual e $\Theta(t)$ o grau de atualização de acordo com distância do neurônio vencedor. A equação 4.4 mostra os detalhes da equação $L(t)$ e a equação 4.5 os detalhes da equação $\Theta(t)$. É possível observar que a equação $L(t)$ decai exponencialmente um fator de aprendizagem que foi parametrizado inicialmente. Em $\Theta(t)$, $\sigma(t)$ é a equação que calcula o tamanho da vizinhança, representada na equação 4.2.

$$W(t+1) = W(t) + \theta(t)L(t)(V(t) - W(t)) \tag{4.3}$$

$$L(t) = L_0 exp\Big(-\frac{t}{\lambda}\Big) \qquad t = 1, 2, 3... \tag{4.4}$$

$$\theta(t) = exp\Big(-\frac{dist^2}{2\sigma^2(t)}\Big) \qquad t = 1, 2, 3... \tag{4.5}$$

4.2 SOM Incremental

Embora o método SOM seja robusto e amplamente utilizado, ele não possui aplicação em contextos de aprendizagem incremental. O sistema SOM precisa de uma quantidade finita de tempo para receber amostras e absorver as características do espaço de entrada em seu mapa. Em um contexto incremental, não é possível reservar um tempo de treinamento, pois a quantidade de informações que entra no sistema é contínua e as

características do espaço de entrada podem mudar com o tempo. Pensando nisso foram desenvolvidos modelos baseados no SOM que apresentam aprendizado incremental.

4.2.1 SOM Crescente

O GSOM (*Growing Self Organizing Map*, Mapa Auto Organizável Crescente), introduz o conceito de um fator de espalhamento. Este fator de espalhamento é responsável por guiar a direção em que o mapa auto organizável cresce. O aprendizado incremental é incorporado neste modelo pelo fato de o mapa poder crescer livremente, no SOM tradicional o mapa tem tamanho fixo. Quando um erro muito alto é alcançado, um novo conjunto de neurônios é criado a partir do neurônio mais parecido que já existe na rede. Outra vantagem do fator de espalhamento é a hierarquização. Através desse fator o mapa é capaz de se sub-espalhar dentro de uma ramificação, significando que o antigo ramo está em uma hierarquia superior ao ramo novo (ALAHAKOON; HALGAMUGE; SRINIVASAN, 2000). A figura 14 mostra o resultado final de um mapa GSOM após ser apresentado a um conjunto de dados com informações sobre os animais de um zoológico.

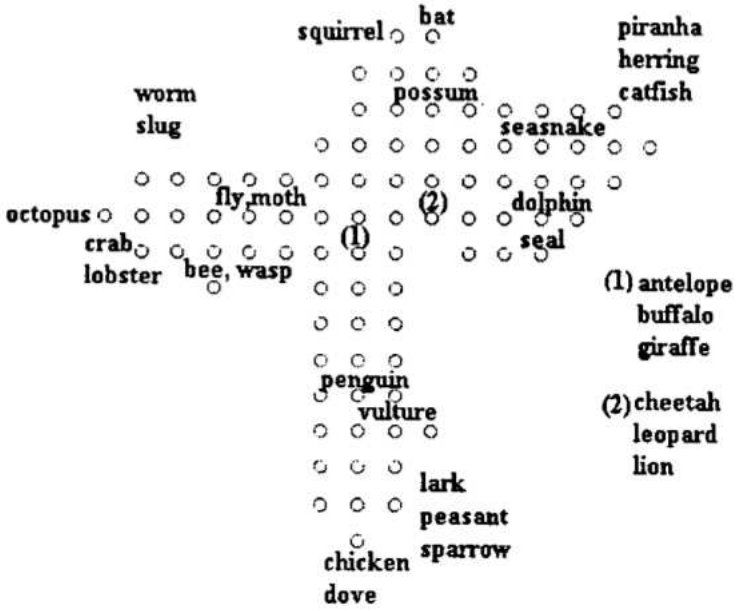

Figura 14 – Resultado Final do GSOM - Alahakoon, Halgamuge e Srinivasan (2000)

4.2.2 Estruturas Celulares Crescentes

O Algoritmo de Estruturas Celulares Crescentes é uma variação do SOM que é capaz de automaticamente encontrar uma morfologia e tamanho de mapa. Ao contrário do SOM, que possui seu tamanho e morfologia pré-estabelecidos, este método é capaz de criar e deletar neurônios conforme os conceitos aprendidos pelo sistema. Sua topologia

inicial é no formato de um triângulo, que vai crescendo e se adaptando de acordo com os padrões de entrada. É capaz de mesclar várias dimensionalidades na criação dos mapas por ser capaz de criar novos neurônios de maneira empilhada. A figura 15 mostra o resultado final do mapa deste algoritmo, é possível observar que os agrupamentos estão fisicamente separados, pois o sistema foi capaz de deletar os neurônios que estavam no caminho (FRITZKE, 1994).

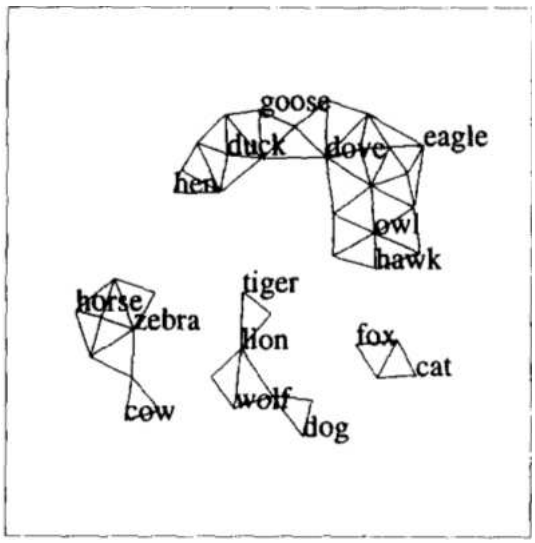

Figura 15 – Mapa do Estruturas Celulares Crescentes - Fritzke (1994)

4.2.3 Mapa Auto Organizável Adaptável com o Tempo

O TASOM (*Time Adaptative Organizing Map* ,Mapa Auto Organizável Adaptável com o Tempo) é o que mais se aproxima do SOM original. A criação do mapa é idêntica, uma morfologia e tamanho devem ser definidas antes do começo do uso da rede. A rede SOM possui uma taxa de aprendizado e raio de vizinhança que são universais para todos os neurônios da rede. Além dessa universalidade, o aprendizado e a vizinhança no SOM são dependentes do tempo. O TASOM introduz o conceito de que cada neurônio possui seu própio raio de vizinhança e taxa de aprendizado.

No TASOM a taxa de aprendizado de cada neurônio é determinada por uma função de distância entre o vetor de entrada e os pesos do neurônio. O raio de vizinhança é dependente da distância entre os pesos do neurônio com os pesos de seus vizinhos. Estas duas características tornam o TASOM livre da variável de iteração, portanto ele não necessita de um período de treinamento e é capaz de se ajustar conforme a precisão de seus agrupamentos e da coesão entre um neurônio e seus vizinhos. Este algoritmo é capaz de aprender novos conceitos não por criar ou deletar novos neurônios, mas por esquecer informações antes aprendidas e dar um novo aprendizado para os neurônios (HOSSEINI; SAFABAKHSH, 2001).

O TASOM é o método SOM incremental mais simples em seu conceito dos que aqui foram apresentados, além de criar um mapa idêntico ao do SOM tradicional. Por ser capaz de criar um mapa equivalente ao do SOM, é possível comparar o desempenho dos dois algoritmos em um ambiente não incremental. Por estes motivos o algoritmo TASOM foi escolhido como objeto de estudo de um método não supervisionado de agrupamento incremental.

5 TASOM

O TASOM foi criado com o intuito de modificar o SOM para retirar a dependência temporal e adicionar mecanismos com capacidade de detectar mudança. O TASOM automaticamente ajusta seus parâmetros de aprendizagem para incorporar as mudanças do conjunto de entrada nos pesos dos neurônios. O algoritmo será resumido nas próximas sub-seções deste capítulo. Os parâmetros iniciais do sistema são:

1. Altura do Mapa: O número de linhas do mapa.
2. Largura do Mapa: O número de colunas do mapa.
3. Raio de Vizinhança Inicial: Tamanho do raio de vizinhança inicial.
4. Taxa de Aprendizagem Inicial: Valor da taxa de aprendizagem incial.
5. Coeficiente de Aprendizagem: Constante que altera o cálculo da taxa de aprendizagem, quanto maior ele for maior será a plasticidade da rede.
6. Coeficiente de Vizinhança: Constante que altera o cálculo do raio de vizinhança, quanto maior ele for mais lentamente será a diminuição do raio de vizinhança.
7. SG: Constante que altera o cálculo do raio de vizinhança, possui efeito similar ao coeficiente de vizinhança, mas seu efeito é menor.
8. SF: Constante que altera o cálculo da taxa de aprendizado, possui efeito similar ao coeficiente de aprendizagem, mas seu efeito é menor.
9. Alfa:Constante que atua no fator de escalabilidade, quanto maior ela for maior será a plasticidade da rede.
10. Beta:Constante que atua no fator de escalabilidade, quanto maior ela for maior será a convergência da rede.

5.1 Inicialização

A primeira etapa do TASOM é a criação do mapa. A dimensionalidade do mapa pode ser de qualquer tamanho, para os estudos aqui realizados foi escolhido a criação de um mapa bi-dimensional. Para cada neurônio é necessário a criação de um conjunto de pesos, variável de aprendizado e raio de vizinhança, estes atributos são individuais de cada neurônio. A variável de aprendizado e o raio de vizinhança são parâmetros iniciais da rede e o vetor de pesos deve ser inicializado com valores randômicos e pequenos.

Existem algumas restrições que devem ser observadas nos parâmetros iniciais: A taxa de aprendizado inicial deve ser inicializada com um valor próximo de 1. As constantes Alfa, Beta, Coeficiente de Aprendizagem e Coeficiente de Vizinhança devem ser valores entre 0 e 1. O raio de vizinhança inicial deve ser maior do que 0. O fator de escalabilidade inicial dever ser positivo, e, preferencialmente, igual a 1. Os valores das funções internas do fator de escalabilidade, Ek e E2k, devem ser inicializados com valores randomicamente pequenos.

5.2 Regra de Micro-Coesão

Para o cálculo do raio de vizinhança de cada neurônio o TASOM leva em consideração o quão coeso o neurônio está de seus vizinhos. Quanto maior for esta coesão, menor será a atualização no tamanho novo do raio de vizinhança. Para aferir a micro-coesão é necessário determinar qual a regra de vizinhança do mapa. Esta regra varia de acordo com a aplicação da rede e a dimensionalidade do mapa. A figura 16 mostra uma regra de vizinhança para mapas quadrados, onde: NH é o conjunto de vizinhos do neurônio i e M é a dimensionalidade do mapa.

$$
\begin{aligned}
NH_{(i1,i2)} &= \{(i1-1,i2),(i1+1,i2),(i1,i2-1),(i1,i2+1)\} \\
NH_{(1,1)} &= \{(1,2),(2,1)\} \\
NH_{(1,M)} &= \{(1,M-1),(2,M)\} \\
NH_{(M,1)} &= \{(M,2),(M-1,1)\} \\
NH_{(M,M)} &= \{(M,M-1),(M-1,M)\}
\end{aligned}
$$

Figura 16 – Regra de Vizinhança - Hosseini e Safabakhsh (2001)

5.3 Escolha do Neurônio Vencedor

Para o cálculo do neurônio vencedor, é necessário comparar o dado de entrada com os pesos de cada um dos neurônios do mapa. Aquele neurônio que for mais parecido com a entrada será escolhido como neurônio vencedor. Para realizar esta comparação é necessário estabelecer um método de medir a distância entre a entrada e os neurônios, o neurônio mais parecido será aquele que apresentar menor distância com a entrada. A equação 5.1 mostra este cálculo utilizando a distância euclidiana, onde: *X(n)* é o vetor de entrada e *W(n)* o vetor de pesos dos neurônios. Esta etapa também é encontrada no

SOM.

$$\begin{aligned} i(x) &= arg_j min \parallel X(n) - W_j(n) \parallel, j = 1, 2, ..., N \quad (1) \\ &\quad onde \\ \parallel X(n) - W_j(n) \parallel &= \sqrt{\left(\sum_k \left(X_k(n) - W_{jk}(n) \right)^2 \right)} \quad (2) \end{aligned} \tag{5.1}$$

5.4 Atualização do Raio de Vizinhança

Após a descoberta do neurônio vencedor, é necessário atualizar seu raio de vizinhança de acordo com a equação 5.2, onde: σi é o raio de vizinhança do neurônio *i*. β é o Coeficiente de Vizinhança. *g* é uma função denota por *(M* $\sqrt{2}$ *- 1)(1 -* $\frac{1}{1+z}$*)*, *M* é a dimensionalidade do mapa. *SG* é a constante inicialmente parametrizada. *sl(n)* é o fator de escalabilidade. E, *#(NHi)* é a cardinalidade do conjunto de vizinhos. Esta cardinalidade dividida pelo somatório das distâncias entre o neurônio vencedor e seus vizinhos é a micro-coesão.

$$\begin{aligned} \sigma_i(n+1) &= \sigma_i(n) + \\ \beta\left(g\left(\frac{1}{sg.sl(n).\#(NH_i)} \sum_{j \exists NH_i} \parallel W_i(n) - W_j(n) \parallel \right) - \sigma_i(n) \right) \end{aligned} \tag{5.2}$$

5.5 Atualização da Taxa de Aprendizagem

A taxa de aprendizagem de todos os neurônios deve ser atualizada de acordo com a equação 5.3, onde: ηj(n) é a taxa de aprendizagem do neurônio atual. α é o Coeficente de Aprendizagem. *f* é uma função denotada por 1 - $\frac{1}{1+z}$. *SF* é a constante inicialmente parametrizada. *sl(n)* é o fator de escalabilidade. E, $\|X(n) - Wj(n)\|$ é a norma da distância entre os pesos do neurônio e a entrada.

$$\eta_j(n+1) = \eta_j(n) + \alpha\left(f\left(\frac{1}{S_f.sl(n)} \parallel X(n) - W_j(n) \parallel \right) - \eta_j(n) \right) \tag{5.3}$$

5.6 Atualização dos Pesos

Nesta etapa é necessário atualizar o peso de todos os neurônios seguindo a equação 5.4, onde: *Wj(n)* é o vetor de pesos do neurônio *n*. ηj(n+1) é a taxa de aprendizado daquele neurônio já atualizada. $\mathbf{h}_i j(x)$*(n+1)* é a mesma equação que determina a influência da

atualização de acordo com distância do neurônio vencedor do SOM, pode ser encontrada na equação 4.5. X(n) - $\mathbf{W}_j$ é a diferença entre a entrada e o peso do neurônio atual.

$$W_j(n+1) = W_j(n) + \eta_j(n+1)h_{i,j(x)}(n+1)[X(n) - W_j(n)] \tag{5.4}$$

5.7 Fator de Escalabilidade

A atualização do fator de escalabilidade segue a equação 5.5, que por sua vez segue as equações das figuras 5.6 e 5.7, onde: α_s é a constante Alfa parametrizada inicialmente. β_s é a constante Beta parametrizada inicialmente. E, $(z)^+ = \text{z se z} > 0$, ou z = 0 caso contrário.

$$sl(n+1) = \sqrt{\sum_k sk(n+1)} \tag{5.5}$$

$$\begin{aligned} sk(n+1) &= \Big(E2_k(n+1) - E_k(n+1)^2\Big)^+ \\ E2_k(n+1) &= E2_k(n) + \alpha_s\Big(X_k^2(n) - E2_k(n)\Big) \end{aligned} \tag{5.6}$$

$$E_k(n+1) = E_k(n) + \beta_s\Big(X_k(n) - E_k(n)\Big) \tag{5.7}$$

6 SOM x TASOM

Para verificar a eficácia do TASOM foram escolhidos dois cenários de teste. A primeira forma de comparação é em um ambiente estático, ou seja, não incremental. Assim é possível comparar o desempenho dos mapas gerados pelos dois algoritmos. Neste caso o mesmo conjunto conjunto de dados foi apresentado aos dois algoritmos. O critério de comparação nesse caso é o cálculo da distância média depois que todos os pesos da rede foram ajustados.

O segundo teste visa testar a eficácia do TASOM em ambientes com alta mutabilidade. Neste caso o SOM não foi considerado por ser incapaz de funcionar nesse tipo de cenário. O mesmo conjunto de dados é apresentado ao TASOM, só que dessa vez cada agrupamento é apresentado em sequência, e, não em amostragem randômica, como no primeiro cenário. Como os dados de entrada apresentam mudanças bruscas é possível testar a eficácia do TASOM em aprender novos conceitos. O método de aferição de erro neste caso leva em conta somente o erro momentâneo para o cálculo do erro médio da rede, ou seja, a distância entre a entrada e o neurônio vencedor no momento da apresentação.

O espaço escolhido para realizar estas análises foi o bi-dimensional, esta escolha foi feita para facilitar a visualização dos agrupamentos. Neste conjunto existem 15 agrupamentos de diferentes tamanhos e formas, dispersos em uma janela que varia de 0 a 100 000 em ambos os eixos. A figura 17 mostra a plotagem destes dados.

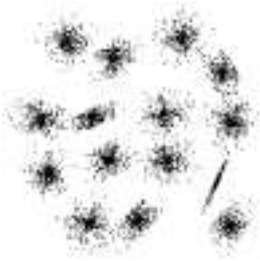

Figura 17 – Conjunto de Dados Sintético - University... (2015)

6.1 Protótipo

Para realizar os testes comparativos um protótipo foi desenvolvido em C++ utilizando a plataforma *C++ Builder XE2* da Embarcadero. O software carrega um arquivo csv do disco rígido que possui as informações dos dados a serem agrupados. É possível parametrizar ambos algoritmos através da interface, as janelas de parametrização podem ser vistas nas figuras 19 e 20. No centro do protótipo existem duas tabelas, uma para cada

algoritmo. Nesta tabela são registrados os valores da entrada, a posição do neurônio vencedor que classificou as amostras e o erro desssa classificação. A visão geral do protótipo está na figura 18.

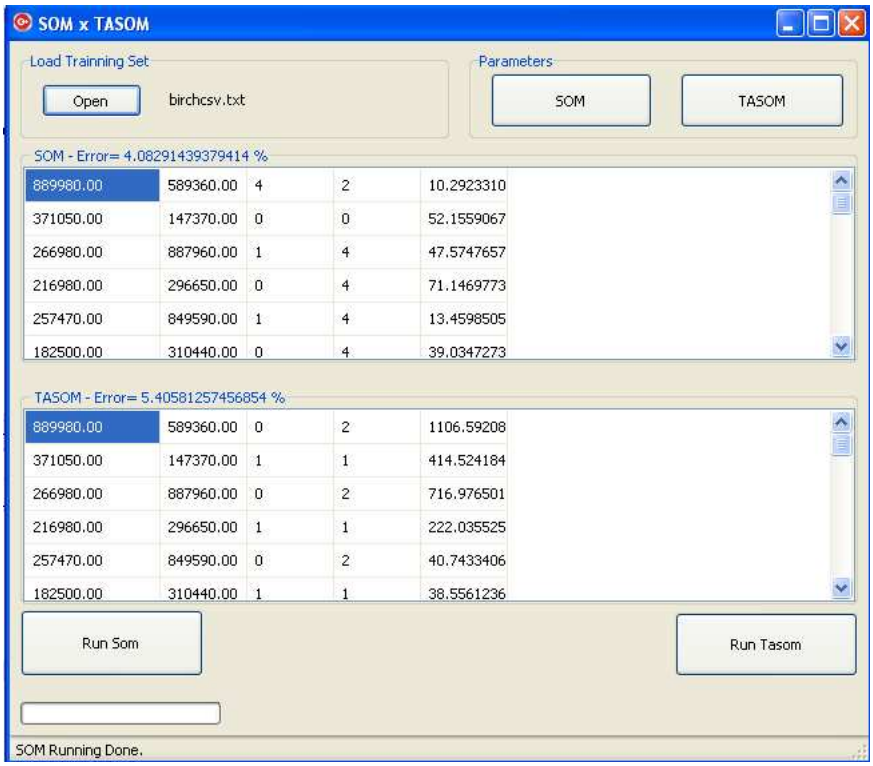

Figura 18 – Protótipo

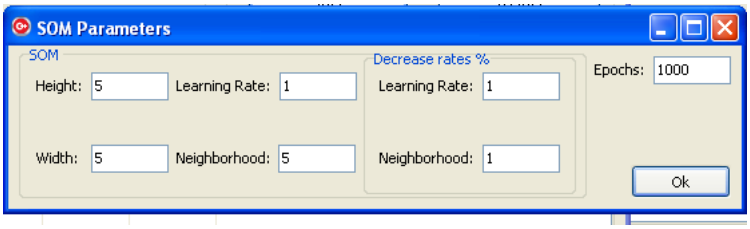

Figura 19 – Parâmetros do SOM

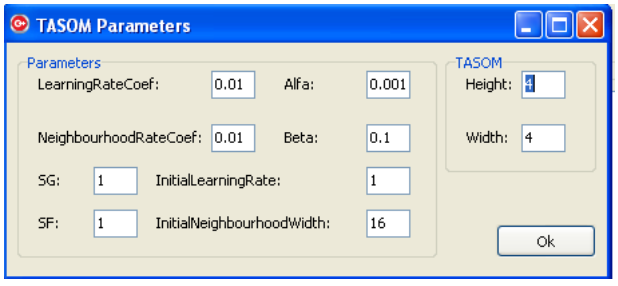

Figura 20 – Parâmetros do TASOM

6.2 SOM x TASOM

Para comparar o SOM com o TASOM é necessário utilizar o mesmo contexto de aplicação. O algoritmo SOM não funciona em ambientes incrementais, o objetivo desta comparação é verificar se o desempenho do TASOM é aceitável também em ambiente não incremental. O conjunto de dados sintéticos apresentado anteriormente foi utilizado nas duas redes. As entradas foram randomicamente embaralhadas para que todos os agrupamentos sejam considerados como estando no mesmo espaço vetorial. Quando os agrupamentos são apresentados um de cada vez existem vários espaços vetoriais que mudam rapidamente e não um espaço com vários agrupamentos. Dentro dos dois algoritmos foi feita uma normalização, onde todas as entradas de todas as dimensões variam entre 0 e 1000. Esse intervalo de normalização foi escolhido pelo fato de o TASOM não se comportar bem com o intervalo padrão entre 0 e 1.

O algoritmo SOM foi treinado com 1000 épocas. Os critérios de parada do treinamento foram: A conclusão das mil épocas de treino ou, a taxa de aprendizado alcançar o valor zero ou o erro imediato da rede for menor que 10%. Em todos os testes realizados o primeiro critério de parada alcançado foram as 1000 épocas, a taxa de aprendizado sempre ficou próxima de $5x10^{-5}$ em todos os testes. O conjunto de parâmetros escolhido para o SOM foram os que apresentaram menor erro final, eles foram configurados da seguinte forma:

1. Altura do Mapa: 4.
2. Largura do Mapa: 4.
3. Raio de Vizinhança Inicial: 4.
4. Taxa de Aprendizagem Inicial: 1.
5. Decaimento da Taxa de Aprendizagem: 1%.
6. Decaimento do Raio de Vizinhança: 1%.

As figuras 21, 22 e 23 mostram o estado da rede SOM em três momentos diferentes: Começo, metade e final do treino. Os pontos representam as amostras de entrada e os quadrados representam os neurônios da rede. É possível observar o desenvolvimento da aprendizagem ao decorrer do tempo, pois os neurônios se aproximam dos agrupamentos. Na figura 23 os pontos são coloridos de acordo com a cor do neurônio que os classifica.

Por ser um algoritmo incremental o TASOM não possui fase de treinamento. O mapa se ajusta de acordo com as entradas que chegam ao sistema. O conjunto de parâmetros escolhidos para o TASOM foram os que fizeram com o que mapa apresentasse menor erro final, eles foram configurados da seguinte forma:

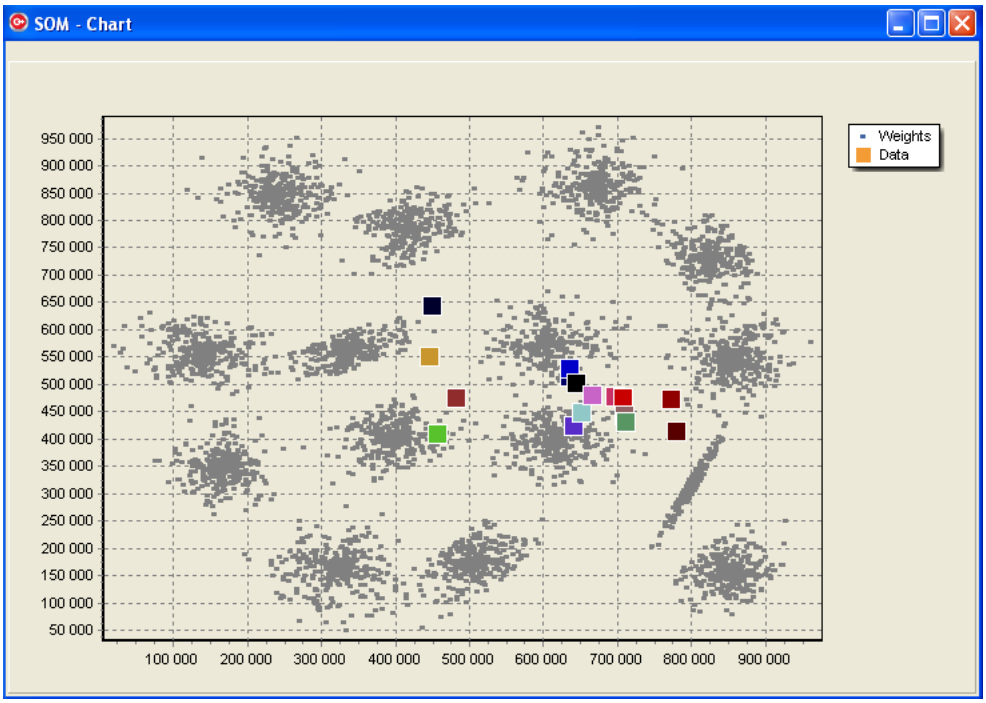

Figura 21 – Começo do Treinamento

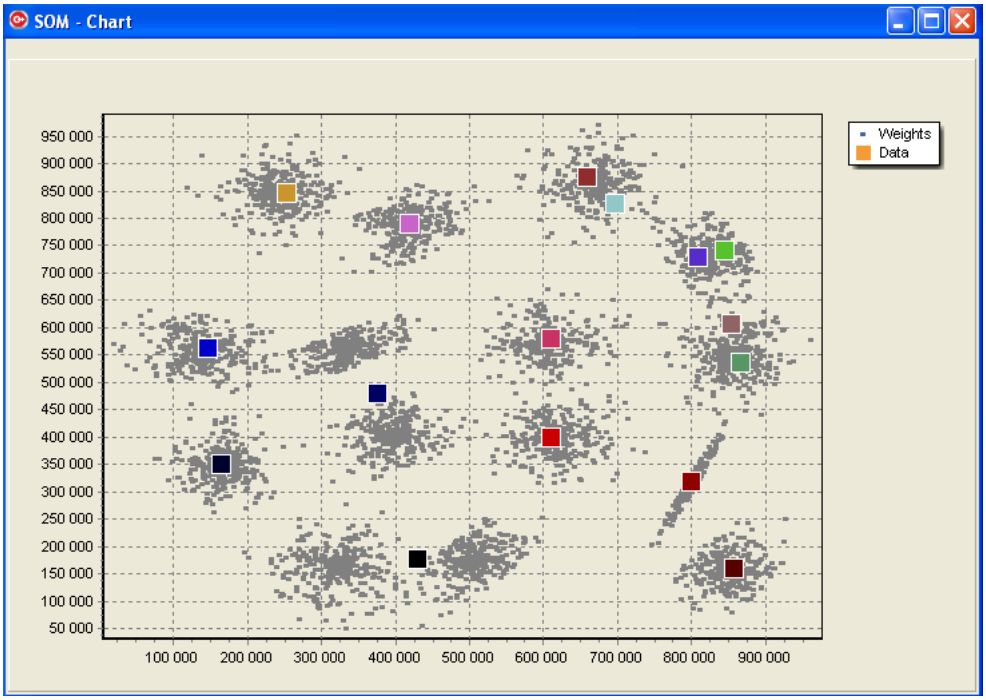

Figura 22 – Meio do Treinamento

1. Altura do Mapa: 4.
2. Largura do Mapa: 4.
3. Raio de Vizinhança Inicial: 4.
4. Taxa de Aprendizagem Inicial: 1.

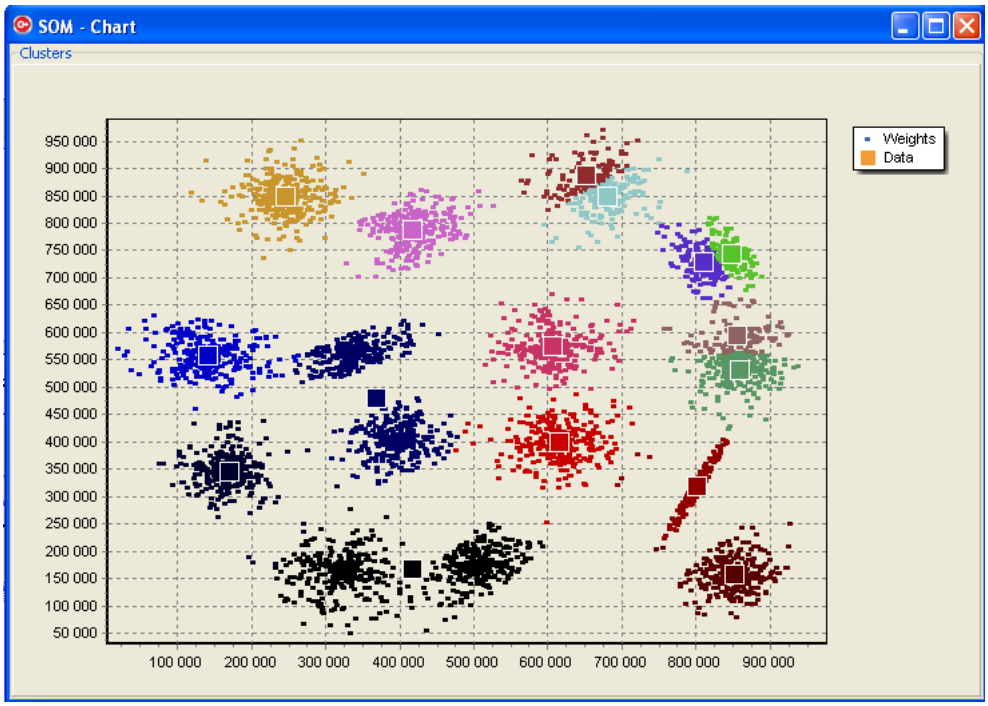

Figura 23 – Fim do Treinamento

5. Coeficiente de Aprendizagem: 0.01.
6. Coeficiente de Vizinhança: 0.01.
7. SG: 1.
8. SF: 1.
9. Alfa: 0.001.
10. Beta: 0.1.

As figuras 24, 25 e 26 mostram o estado da rede TASOM em três momentos diferentes: Começo, metade e final da apresentação. Os pontos representam as amostras de entrada e os quadrados representam os neurônios da rede. É possível observar o desenvolvimento da aprendizagem ao decorrer do tempo, pois os neurônios se aproximam dos agrupamentos. Em todas as etapas as entradas já são coloridas de acordo com os neurônios que as classificam. A execução do TASOM é mais rápida do que a do SOM. Considerando somente o número de vezes que o conjunto de entrada é apresentado, o TASOM se mostrou 1000 vezes mais rápido, pois os dados de entrada são apresentados apenas uma vez.

Para comparar os dois algoritmos é necessário estabelecer uma métrica que seja aplicável aos dois mapas. Para verificar a capacidade que cada algoritmo tem em corretamente agrupar os conjuntos corretos, foi utilizado o erro médio final. O erro final é a

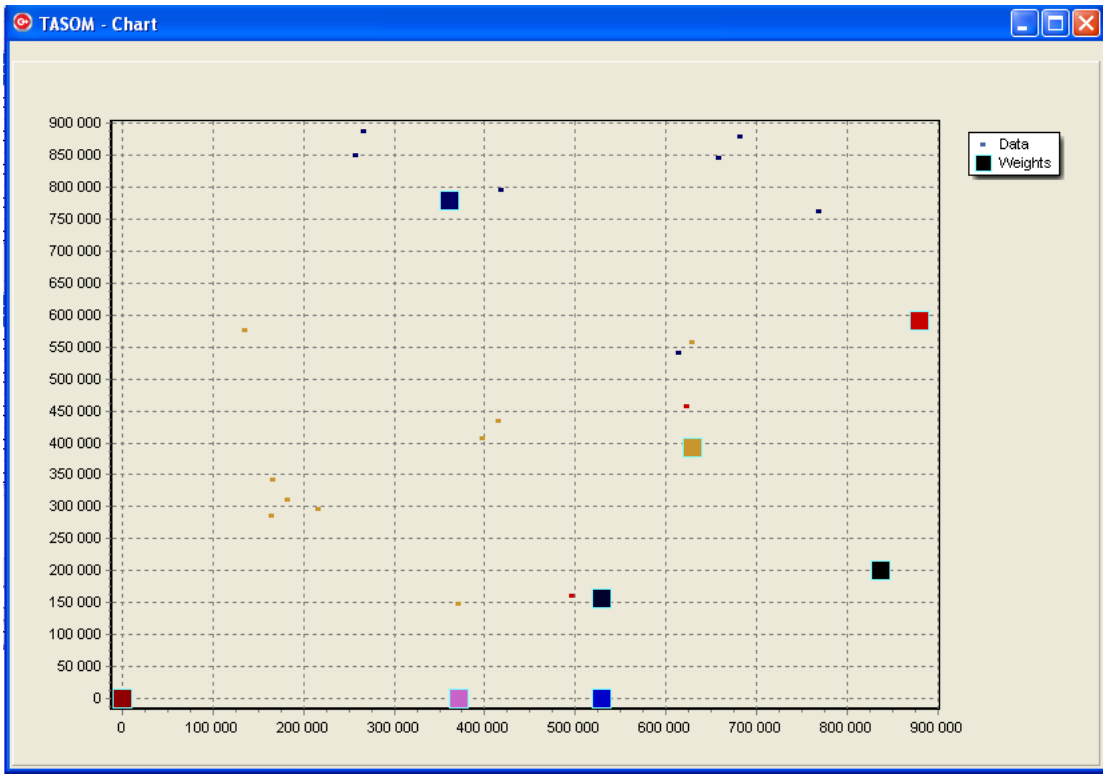

Figura 24 – Começo - Estático

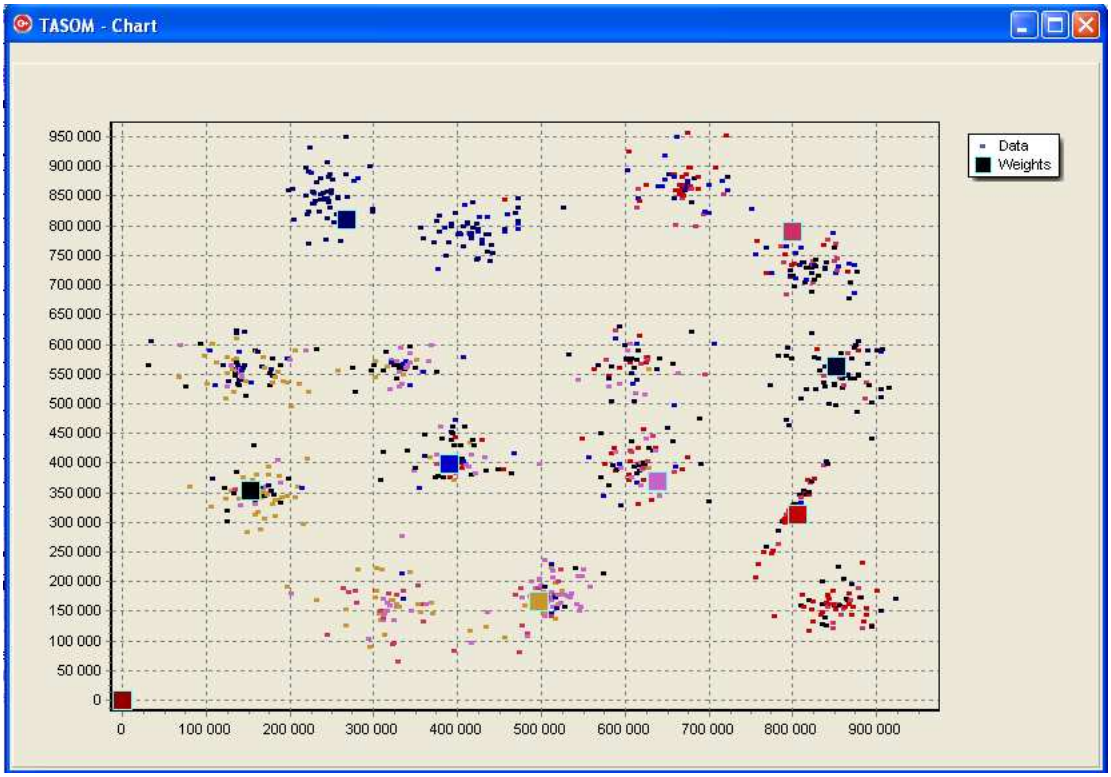

Figura 25 – Meio - Estático

distância entre a entrada e o neurônio que a classificou depois da estabilização da rede. Após todas as épocas de treinamento do SOM e depois de todas as amostras passarem

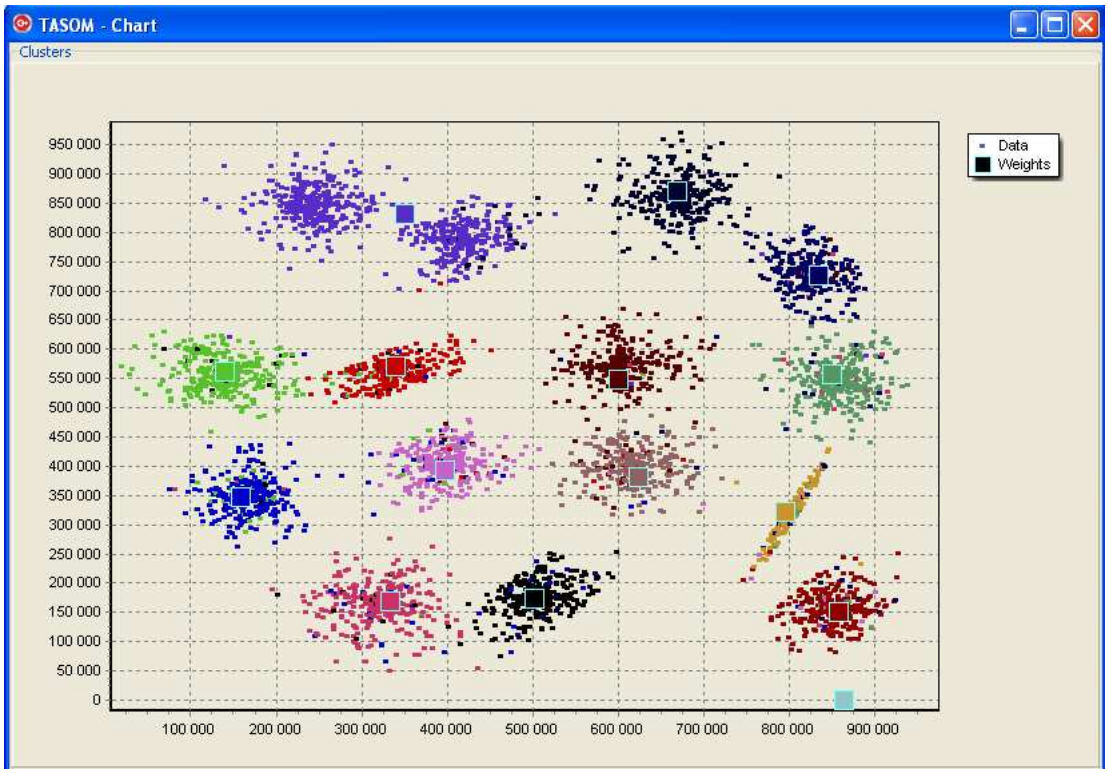

Figura 26 – Fim - Estático

pelo TASOM, as entradas foram novamente apresentadas e foi possível calcular o erro médio em um momento em que ambos os mapas estão estabilizados. Foram feitas 10 execuções, pois ambas as redes possuem uma inicialização de pesos randômica que pode influenciar no resultado final. A tabela 3 mostra o erro médio de cada um dos algoritmos.

Tabela 3 – SOM x TASOM

	SOM	TASOM
1	5.134363%	4.675699%
2	5.134363%	7.970087%
3	5.134363%	4.657833%
4	5.134363%	4.636273%
5	5.134363%	6.346224%
6	5.134363%	3.632400%
7	5.134363%	3.755198%
8	5.134363%	4.669887%
9	5.134363%	3.695024%
10	5.134363%	4.780292%
Média	5.134363%	4.881891%

6.3 TASOM em Ambiente Incremental

Para medir a eficiência do TASOM em um ambiente incremental de maneira sintética é necessário utilizar um conjunto de entradas que muda bruscamente. Dessa forma é possível verificar se o TASOM é capaz de aprender novos conceitos com rapidez e eficiência. O cálculo de erro neste caso só deve levar em consideração a distância entre o neurônio e a entrada na hora em que o dado é apresentado, pois amostras subsequentes podem introduzir novos conceitos que serão absorvidos por aquele neurônio. O mesmo conjunto de entradas foi utilizado, mas seus agrupamentos são apresentados de maneira sequencial. Para realizar este teste todos os parâmetros da rede TASOM previamente apresentados foram mantidos. As figuras 27, 28 e 29 são representações no começo, meio e fim da apresentação do dados de entrada.

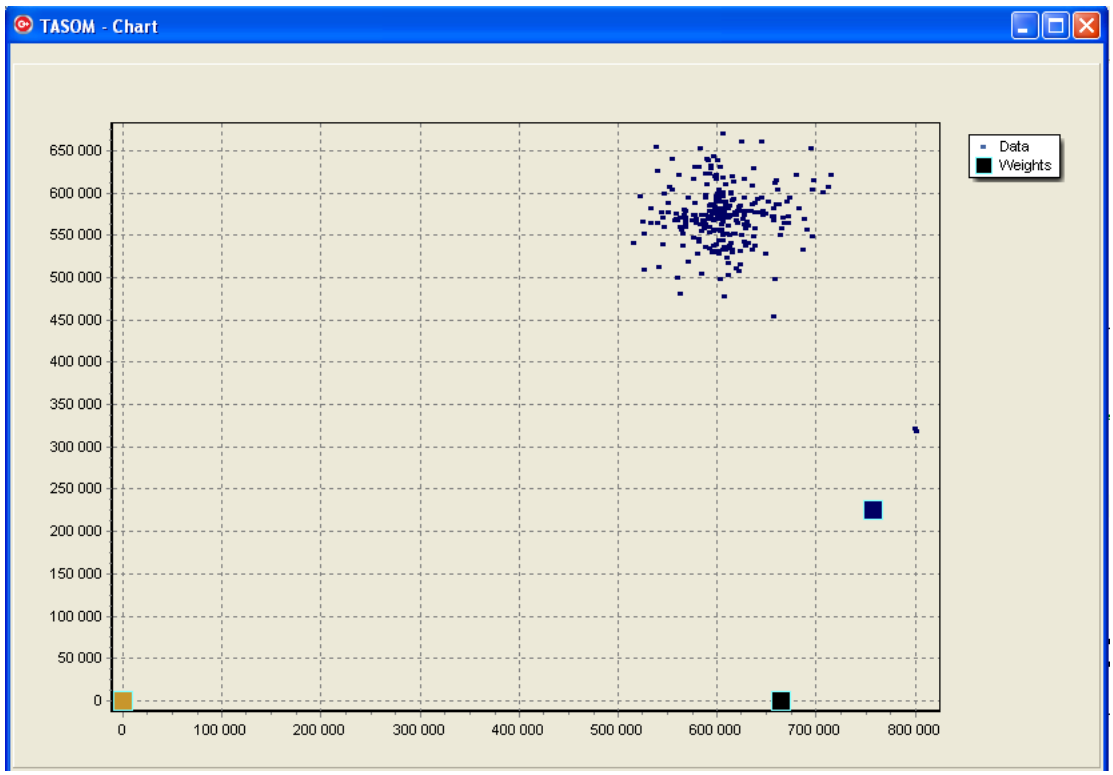

Figura 27 – Começo - Dinâmico

É possível observar que cada um dos agrupamentos é marcado por apenas um neurônio. Isso acontece porque o neurônio mais próximo a primeira amostra apresentada de cada conjunto se move até a posição daquele agrupamento. Depois disso ele sempre é o neurônio que está mais próximo daquele grupo. O TASOM é altamente sensível aos pesos randômicos de inicialização, é possível observar esse fenômeno quando a quantidade de neurônios agrupadores muda de execução para execução e na variação de erro que foi observada na comparação entre o SOM e o TASOM. O erro encontrado para este teste pode ser observado na tabela 4.

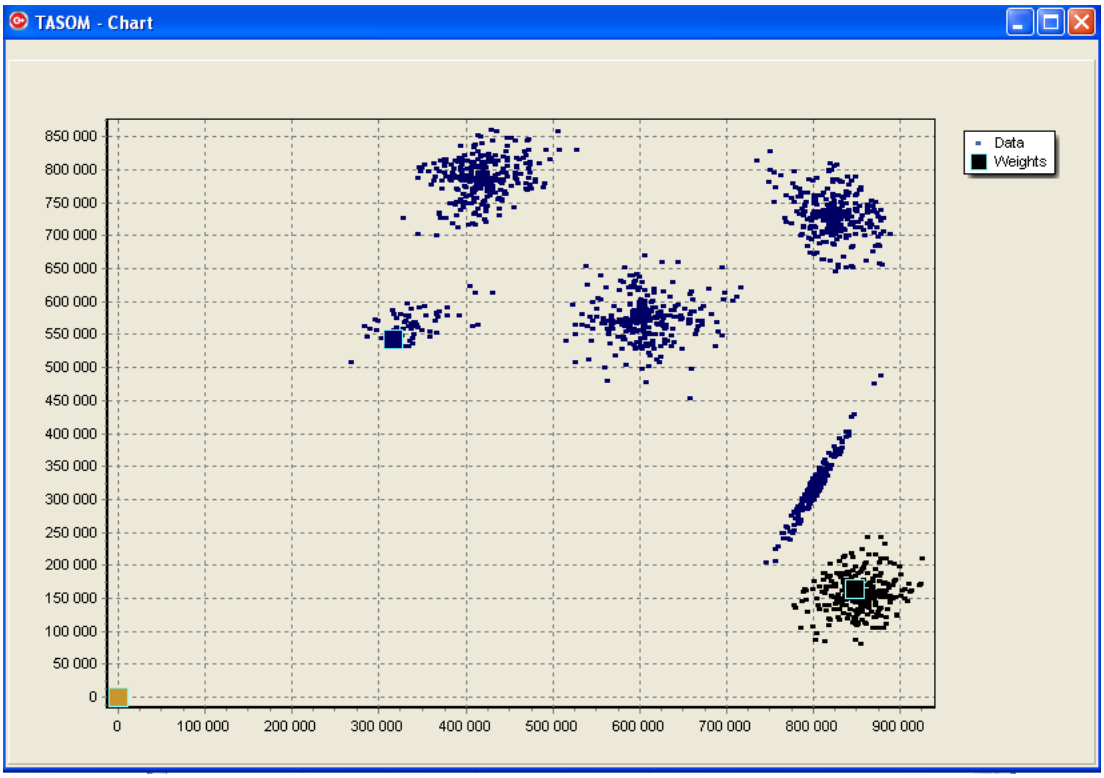

Figura 28 – Meio - Dinâmico

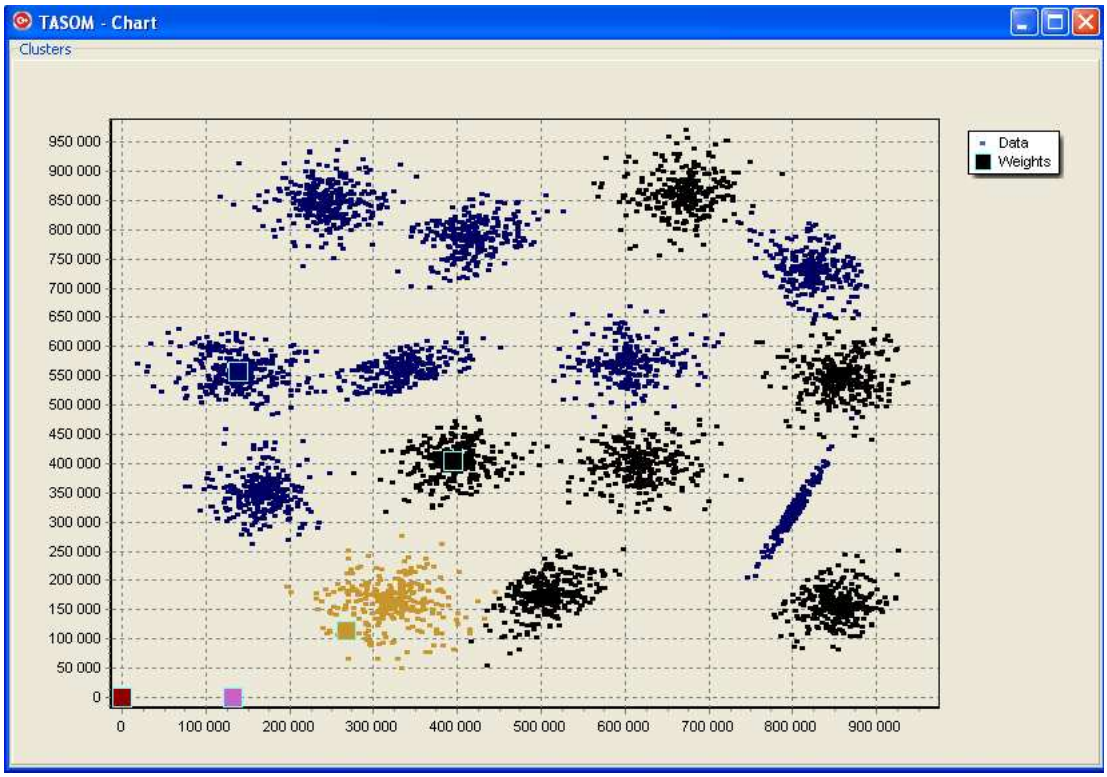

Figura 29 – Fim - Dinâmico

Tabela 4 – TASOM Incremental

	TASOM
1	11.454104%
2	11.122445%
3	6.6950870%
4	5.5194768%
5	11.454104%
6	11.563024%
7	5.1994760%
8	4.6800620%
9	4.6800621%
10	4.9833902%
Média	7.7351230%

6.4 Resultado da Comparação

O TASOM foi capaz de superar o SOM em ambiente estático, apresentando um erro médio de 4.881891% contra 5.134363% do SOM. Em um contexto incremental o erro médio foi de 7.7351230%. Os resultados destes experimentos indicam que o TASOM é uma boa alternativa ao SOM, pois é capaz de gerar um mapa que possui as mesmas características do SOM e é capaz de ser usado em um contexto incremental. Alguns cuidados devem ser tomados no uso do TASOM, a escolha dos parâmetros inicias pode ser complicada e ele é muito sensível à semente inicial dos pesos.

7 Validação com Dados Reais

Dois problemas reais são abordados visando testar os algoritmos em ambiente não simulado. O primeiro caso é aplicado à necessidade do Ministério da Agricultura, Pecuária e Abastecimento (MAPA). O segundo é aplicado a uma base de dados que possui informações sobre as plantas do tipo Iris. O objetivo deste experimentos é utilizar os métodos não supervisionados de agrupamentos de maneira comparativa entre si. A importância da utilização de um método de agrupamento não supervisionado em bases de dados que possuem rótulo é a observação de características ocultas nos dados. Estas características podem validar a integridade dos dados e dar informações sobre presença de ruídos e separabilidade das classes no espaço de análise.

7.1 MAPA

O primeiro estudo que será usado para validar os métodos de Aprendizagem de Máquina com Aprendizagem Incremental é baseado em um problema real do MAPA. Este ministério é responsável "pela gestão das políticas públicas de estímulo à agropecuária, pelo fomento do agronegócio e pela regulação e normatização de serviços vinculados ao setor"(MAPA,). Um dos objetivos do MAPA é garantir a segurança alimentar do povo brasileiro além de suportar a produção de exportação, garantindo o sucesso dos produtos brasileiros no mercado internacional.

No Brasil existe um plano nacional que tem como objetivo garantir a qualidade e segurança dos produtos animais e vegetais que são consumidos no país. O Plano Nacional de Controle de Resíduos e Contaminantes (PNCR), é um programa de inspeção e fiscalização das cadeias produtivas de alimentos. Este plano é dividido em duas grandes áreas, PNCR/Animal e PNCR/Vegetal. Cada um possui regras de monitoramento específicas. Os tipos de animais que são contemplados pela parte animal são: Bovinos, aves, suínos, equinos, avestruz, caprinos e ovinos. Também são verificados no setor animal os seguintes produtos: Leite, mel, ovos e pescados.

O execução do PNCR ocorre através de medições e análises estatísticas de amostras coletadas em Serviços de Inspeção Federais (SIF's). Os produtores de cada região são cadastrados em um SIF específico, este SIF é responsável por coletar e analisar amostras de alimentos produzidos nestes estabelecimentos. As coletas são feitas de maneira randômica de tempos em tempos, e o objetivo das coletas é verificar a presença de resíduos contaminantes ou, substâncias acima de limiar adequado a saúde, nos produtos alimentícios.

Após a análise feita, e uma vez identificada uma violação no limite máximo de resíduo, ou a presença de contaminante, o MAPA adota medidas regulatórias imediatas visando proteger a saúde da população brasileira. O objetivo da utilização de métodos de agrupamento nas informações referentes a cada produtor é verificar a presença de agrupamentos naturais que sejam classificados como prováveis amostras irregulares. Desta forma seria possível ao MAPA tomar medidas que melhorem a eficácia da fiscalização.

Para realizar esta análise foram disponibilizadas duas planilhas com informações sobre amostras coletas de aves e suínos do período de 2002 a 2014. As informações presentes na base de dados de aves e suínos são:

1. Ano da Análise: Ano no qual a amostra foi coletada.
2. Código do Tipo da Análise: Código que identifica o tipo de análise realizada na amostra, isto é, a procura pela presença de determinada substância na amostra.
3. Resultado da Análise:Quantidade da substância analisada na amostra.
4. SIF: Código que identifica o SIF que coletou a amostra.
5. Semana da Analise: Semana do ano em que a amostra foi coletada.
6. Quantidade de Animais:Quantidade de animais amostrados.
7. Latitude: Latitude do Município de origem da amostra.
8. Longitude: Longitude do Município de origem da amostra.
9. Código de Status: Código que indica o resultado da análise laboratorial. 5 para substância não presente, 6 para substância presente mas no limite aceitável e 7 para substância proibida ou acima do limite aceitável.
10. Tipo de Tecido(Somente na base de suínos):Código que indica o tipo de tecido animal coletado.

Ambos algoritmos, SOM e TASOM, foram parametrizados da mesma forma que foram parametrizados no capítulo SOM X TASOM. A única diferença é que os tamanhos dos mapas foram de 2 x 2. Todas as amostras já possuem o rótulo que classifica o resultado da análise, Código de Status. Para verificar a eficácia dos métodos em agrupar as amostrar de acordo com este rótulo, foram registradas a porcentagem da presença de cada classe em cada um dos neurônios das redes. O resultado ideal é que um neurônio possua alta densidade de uma determinada classe e baixa densidade das outras, isto significa que o método de agrupamento é eficaz em determinar o rótulo das amostras. Para um primeiro teste, os algoritmos foram alimentados com todas as informações, excluindo o próprio

rótulo. A tabela 5 mostra as porcentagens de cada classe em cada um dos neurônios para os dois mapas na base de aves, e a tabela 6 para suínos.

Tabela 5 – Agrupamentos feitos com todas as informações - Aves - %

Neurônio	SOM			TASOM		
	Classe 5	Classe 6	Classe7	Classe 5	Classe 6	Classe 7
1	99.54	0.23	0.21	0	0	0
2	99.45	0.40	0.13	99.40	0.41	0.18
3	99.67	0.16	0.16	99.54	0.29	0.16
4	99.46	0.37	0.15	99.52	0.26	0.21

Tabela 6 – Agrupamentos feitos com todas as informações - Suínos - %

Neurônio	TASOM			SOM		
	Classe 5	Classe 6	Classe7	Classe 5	Classe 6	Classe 7
1	0	0	0	97.70	1.91	0.38
2	98.60	1.09	0.30	99.25	0.35	0.38
3	98.59	0.92	0.48	96.37	3.28	0.24
4	97.29	2.48	0.22	97.47	2.32	0.21

É possível constatar pelos dados das tabelas 5 e 6 que a classe predominante nos neurônios dos dois algoritmos foi a 5, que representa a normalidade. Este fato é também observado nos dados, as classes 6 e 7 são responsáveis por aproximadamente 10% do total das amostras de cada um de seus conjuntos. Os métodos não foram capazes de separar os dados em relação ao resultado da análise neste contexto. Visando alcançar melhores resultados, um processamento foi realizado nos dados para omitir todas as amostras do tipo 5, desta forma o objetivo agora é verificar se os algoritmos conseguirão agrupar os dados separando as classes 6 e 7. As tabelas 7 e 8 contém estes resultados.

Tabela 7 – Agrupamentos feitos sem a classe 5 - Aves- %

Neurônio	SOM		TASOM	
	Classe 6	Classe 7	Classe 6-	Classe 7
1	54.09	45.90	0	0
2	74.28	25.71	43.93	56.06
3	62.24	37.75	76.68	23.31
4	60.20	39.79	65.64	34.35

Pelos resultados das tabelas 7 e 8 é possível observar que a retirada da classe 5 teve impacto na forma como as classes 6 e 7 são agrupadas. A quantidade de dados que foram analisados para os suínos foi de 4004 amostras e para as aves 490. Os resultados dos agrupamentos de aves são muito mais misturados do que os de suínos, que possuem alta densidade de uma só classe em seus neurônios. Estes resultados não podem ser chamados de satisfatórios, pois os agrupamentos de aves estão misturados entre as classes 6 e 7. E os de suínos, embora possuam alta densidade da classe 6 em seus neurônios, não possuem

Tabela 8 – Agrupamentos feitos sem a classe 5 - Suínos- %

Neurônio	SOM		TASOM	
	Classe 6	Classe 7	Classe 6	Classe 7
1	65.60	24.29	0	0
2	86.74	13.25	62.42	37.57
3	91.79	8.2	88.16	11.83
4	91.88	39.79	95.66	4.33

algum neurônio com alta densidade de classe 7. Os dados de entrada destes experimentos ainda possuem um erro quanto se pensa em uma aplicação prática. O atributo Resultado da Análise, que contém a quantidade da substância analisada na amostra, está presente. Este atributo é o fator determinante para o Código de Status, para uso real, seria ideal que os agrupamentos fossem observados sem a presença desta característica nos dados. As tabelas 9 e 10 mostram os resultados dos mapas sem este atributo presente nas entradas.

Tabela 9 – Agrupamentos feitos sem a classe 5 e sem o resultado da análise - Aves - %

Neurônio	SOM		TASOM	
	Classe 6	Classe 7	Classe 6	Classe 7
1	63.46	36.53	0	0
2	0	0	69.28	30.73
3	0	0	39.29	40.70
4	0	8.11	0	0

Tabela 10 – Agrupamentos feitos sem a classe 5 e sem o resultado da análise - Suínos - %

Neurônio	SOM		TASOM	
	Classe 6	Classe 7	Classe 6	Classe 7
1	91.79	8.2	0	0
2	91.96	8.03	100	0
3	65.60	34.39	85.46	14.53
4	89.24	11.03	87.73	12.26

Os resultados das tabelas 8 e 9 apresentam os mesmos problemas acima citados. Um fato curioso é que na análise de suínos o neurônio 2 do TASOM foi capaz de agrupar 100% de amostras da classe 6, porém, apenas três entradas. O resultado deste experimento aponta que não há informações o suficiente no conjunto de dados para realizar a separabilidade, o espaço de características não foi capaz de realizar separação entre as classes.

Uma outra análise que pode ser feita é a observação da dispersão deste conjunto de dados em mapas maiores. Com a informação da posição de agrupamento de cada classe, é possível observar a região do mapa em que os dados foram agrupados. É possível observar a separabilidade por região. As tabelas 11 e 12 mostram o agrupamento das classes 6 e 7 para um mapa SOM e as tabelas 13 e 14 para um mapa TASOM. Ambos os mapas

Tabela 11 – Classe 6 - SOM - 5x5

Linha / *Coluna*	1	2	3	4	5
1	7	1	30	13	20
2	39	14	8	12	14
3	34	14	15	6	9
4	5	16	32	1	0
5	9	4	4	0	0

Tabela 12 – Classe 7 - SOM - 5x5

Linha / *Coluna*	1	2	3	4	5
1	4	21	1	5	16
2	24	17	1	0	9
3	12	1	1	10	8
4	10	2	0	0	0
5	12	0	0	0	25

Tabela 13 – Classe 6 - TASOM - 5x5

Linha / *Coluna*	1	2	3	4	5
1	0	0	0	0	0
2	0	43	0	0	0
3	0	93	36	56	22
4	7	12	0	3	0
5	0	27	0	5	0

tem dimensionalidade 5x5, cada neurônio possui o número de amostras daquele classe que foram agrupadas por aquele neurônio. A distribuição dos dados em um mapa 9x9 pode ser observada nas tabelas 15 e 16, para o SOM, e 17 e 18, para o TASOM.

Como pode-se observar nas tabelas 11 a 18, este conjunto de dados apresenta grande mesclagem. Isto pode significar duas coisas, ou há grande inseparabilidade linear nos dados ou há grande presença de ruído. Não há características suficientes na base de dados para gerar agrupamentos que separam os dados pelas classes 5, 6 ou 7. Como também foi constatado em (CARVALHO, 2014).

Tabela 14 – Classe 7 - TASOM - 5x5

Linha / *Coluna*	1	2	3	4	5
1	0	0	0	0	0
2	0	24	0	0	0
3	0	56	12	40	26
4	12	0	0	3	0
5	0	0	0	5	0

Tabela 15 – Classe 6 - SOM - 9x9

Linha / Coluna	1	2	3	4	5	6	7	8	9
1	14	13	1	7	11	12	6	6	6
2	15	20	9	8	2	7	5	1	8
3	10	16	5	0	2	2	1	2	0
4	20	11	3	4	5	1	1	1	0
5	7	3	2	2	0	2	2	3	0
6	4	1	0	0	1	1	0	4	0
7	0	3	2	0	1	1	2	0	0
8	4	1	1	2	0	0	0	0	0
9	4	0	1	1	2	2	0	0	0

Tabela 16 – Classe 7 - SOM - 9x9

Linha / Coluna	1	2	3	4	5	6	7	8	9
1	18	2	25	0	0	12	0	0	0
2	9	5	1	0	0	0	0	0	0
3	5	1	4	0	12	0	0	0	0
4	16	1	1	0	0	12	0	0	0
5	0	0	0	0	0	0	0	0	0
6	0	0	1	0	0	0	0	0	0
7	0	0	0	9	0	0	0	0	0
8	0	21	0	0	0	4	0	0	0
9	12	0	0	0	0	0	0	0	8

Tabela 17 – Classe 6 - TASOM - 9x9

Linha / Coluna	1	2	3	4	5	6	7	8	9
1	0	5	56	12	30	12	23	9	38
2	7	0	4	0	0	0	0	25	1
3	1	0	0	1	0	1	1	0	0
4	1	5	7	0	0	6	25	0	0
5	0	0	0	0	0	0	0	0	0
6	0	1	0	0	0	40	0	0	0
7	0	0	0	9	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0

Tabela 18 – Classe 7 - TASOM - 9x9

Linha / *Coluna*	1	2	3	4	5	6	7	8	9
1	0	0	16	31	12	1	22	5	2
2	0	0	0	0	0	0	0	8	0
3	25	0	0	0	0	0	0	0	0
4	0	10	12	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	24	0	0	0
7	0	0	0	9	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0

7.2 Iris

A última análise feita usa o conjunto e dados de plantas do tipo Iris, estes dados são muito famosos, sendo citados em vários artigos de aprendizado de máquina. Ele é composto por 3 classes de 50 amostras cada, cada classe refere-se a um tipo de planta Iris. A primeira classe é linearmente separável das outras, as classes 2 e 3 são linearmente inseparáveis (FISHER, 1936). O objetivo do experimento é observar se o SOM ou o TASOM são capazes de realizar o agrupamento destes dados separando as classes em diferentes grupos. A descrição do conjunto de dados é:

1. Comprimento da cepa.
2. Largura da cepa.
3. Comprimento da pétala.
4. Largura da pétala.
5. Classe da Iris.0 para Iris Setosa, 1 para Iris Versicolour e 2 para Iris Virginica.

Dez execuções de cada algoritmo foram realizadas, elas não apresentaram grande variação entre si. A tabela 19 mostra os melhores resultados da porcentagem de cada classe nos neurônios do SOM e do TASOM e a quantidade de amostras que foram agrupadas por cada neurônio. O tamanho dos mapas foi de 2x2, as parametrizações do TASOM foram as mesmas já utilizadas, com exceção da variável beta que foi configurada para 0.001. O SOM foi executado com 450 épocas. Estas foram as parametrizações que apresentaram melhor resultado.

Pela tabela 19 é observável que este experimento obteve êxito. As redes foram capazes de separar completamente a classe 0, que é linearmente separável, e quase foram capazes de separar totalmente as classes 1 e 2. O TASOM obteve melhor desempenho

Tabela 19 – Agrupamentos de Iris- %

Neurônio	TASOM				SOM			
	Setosa	Versicolour	Virginica	Nº Amostras	Setosa	Versicolour	Virginica	Nº Amostras
1	100	0	0	2	0	93.10	6.89	29
2	0	70.42	29.71	71	0	54.76	45.23	42
3	100	0	0	48	100	0	0	50
4	0	0	100	29	0	0	100	29

na separação das classes 1 e 2. Se uma fusão dos neurônios 1 e 3 do TASOM for feita, é possível dizer que as amostras do tipo Setosa foram perfeitamente separadas das outras.

Os algoritmos de agrupamento não foram feitos para gerar aglomerações baseadas em um rótulo, antes, eles agrupam os dados unicamente por critérios matemáticos. Entretanto este tipo de método pode ser usado como um auxiliador dos métodos supervisionados. Observar o agrupamento natural dos dados pode ajudar os analistas de dados a tomar decisões importantes, pois é possível ter indicações a respeito da separabilidade das informações e sobre o nível de ruído presente nas amostras. Se os agrupamentos naturais separam os dados de acordo com os rótulos, isso quer dizer que o espaço de entrada é linearmente separável e possui ruído baixo. Se os agrupamentos matemáticos não seguem as separações das classes, ou os dados são linearmente inseparáveis, ou há ruído no espaço de entrada, ou ambos.

As próximas tabelas mostram a distribuição natural dos dados de plantas do tipo Iris com diferentes dimensões de mapas, e uma classe de cada vez. Cada tabela possui as informações de apenas uma classe. Cada número representa a quantidade de amostras que foram agrupadas naquele neurônio. A tabela 20, 21 e 22 possui a distribuição dos dados no mapa 5x5 da rede SOM, e as tabelas 23, 24, e 25 para mapas do mesmo tamanho do tipo TASOM.

Nas tabelas 26, 27 e 28, são apresentados os dados de plantas do tipo Iris espalhados por um mapa SOM de dimensões 9x9. As tabelas 30, 31 e 32 apresentam mapas TASOM com a mesma dimensionalidade.

Pelos resultados das tabelas de distribuição dos dados em mapas de dimensionalidade 5x5 e 9x9, é possível observar regiões para a classe Setosa separadas das outras classes. As classes Virginica e Versivolour dividem espaço nos mapas. Estes agrupamentos ajudam a compreender a natureza dos dados e podem servir como instrumento na escolha de métodos supervisionados para serem aplicados neste conjunto de dados. O TASOM tem a características de ativar apenas alguns neurônios para cada agrupamentos, enquanto o SOM distribui seus neurônios por todo o conjunto de dados. É possível observar esta característica nos mapas de distribuição, pois o SOM sempre ativa uma região maior de neurônios para cada classe.

Tabela 20 – Setosa - SOM - 5x5

Linha / Coluna	1	2	3	4	5
1	0	4	9	6	7
2	0	1	5	4	2
3	0	0	3	1	3
4	0	0	0	1	2
5	0	0	0	1	1

Tabela 21 – Versicolour - SOM - 5x5

Linha / Coluna	1	2	3	4	5
1	0	0	0	0	0
2	0	0	0	0	0
3	3	1	0	0	0
4	16	4	0	0	0
5	13	10	3	0	1

Tabela 22 – Virginica - SOM - 5x5

Linha / Coluna	1	2	3	4	5
1	18	0	0	0	0
2	11	0	0	0	0
3	10	1	0	0	0
4	0	1	0	0	0
5	1	1	0	0	0

Tabela 23 – Setosa - TASOM - 5x5

Linha / Coluna	1	2	3	4	5
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	12	1	0	37

Tabela 24 – Versicolour - TASOM - 5x5

Linha / Coluna	1	2	3	4	5
1	0	0	0	3	0
2	0	0	0	0	0
3	0	0	0	0	0
4	46	0	0	1	0
5	0	0	0	0	0

Tabela 25 – Virginica - TASOM - 5x5

Linha / Coluna	1	2	3	4	5
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	29	0	0
4	21	0	0	0	0
5	0	0	0	0	0

Tabela 26 – Setosa - SOM - 9x9

Linha / *Coluna*	1	2	3	4	5	6	7	8	9
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0
5	3	1	0	0	0	0	0	0	0
6	5	3	1	0	0	0	0	0	0
7	4	2	4	3	0	0	0	0	0
8	6	4	1	0	0	0	0	0	0
9	7	2	2	1	1	0	0	0	0

Tabela 27 – Versicolour - SOM - 9x9

Linha / *Coluna*	1	2	3	4	5	6	7	8	9
1	3	0	0	0	0	0	0	0	0
2	2	2	1	0	1	2	0	0	0
3	2	2	0	0	2	2	3	0	0
4	0	1	1	0	0	3	2	1	0
5	0	0	2	2	0	2	1	1	0
6	0	0	0	0	1	0	1	0	1
7	0	0	0	0	0	1	2	1	1
8	0	0	0	0	0	0	1	0	0
9	0	0	0	0	0	1	1	0	1

Tabela 28 – Virginica - SOM - 9x9

Linha / *Coluna*	1	2	3	4	5	6	7	8	9
1	0	1	5	3	4	9	2	2	3
2	0	0	3	2	3	0	1	2	2
3	0	0	0	4	0	0	0	0	1
4	0	0	0	1	0	0	0	0	1
5	0	0	0	0	0	0	0	0	1
6	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0

Tabela 29 – Setosa - TASOM - 9x9

Linha / *Coluna*	1	2	3	4	5	6	7	8	9
1	0	0	0	0	1	1	0	0	0
2	0	0	8	0	0	0	0	0	9
3	0	4	0	13	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	13	0	0
6	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	1	0	0	0	0	0	0

Tabela 30 – Versicolour - TASOM - 9x9

Linha / *Coluna*	1	2	3	4	5	6	7	8	9
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	45	0	0	0	0
4	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	3	0
6	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	20	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0

Tabela 31 – Virginica - TASOM - 9x9

Linha / *Coluna*	1	2	3	4	5	6	7	8	9
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	21	0	0	0	0
4	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	29	0	0	0
9	0	0	0	0	0	0	0	0	0

8 Considerações Finais

Na introdução deste documento foram levantados os seguintes objetivos: Pesquisa e desenvolvimento de um método de Aprendizagem de Máquina do tipo não supervisionado, de agrupamento e incremental. Um levantamento teórico foi realizado visando compreender os processos de aprendizagem de máquina, os conceitos de aprendizagem incremental e o conhecimento de métodos de agrupamento incremental. As pesquisas levaram a escolha do método TASOM como um algoritmo não-supervisionado de agrupamento incremental. Este método foi escolhido por ser uma variação do SOM que atende aos requisitos do trabalho e por ter um mapa de saída comparável com o SOM. Desta forma é possível avaliar o desempenho dos dois em contexto de aplicação similar.

Os dois algoritmos foram implementados e um protótipo foi criado para a realização dos experimentos. Utilizando um conjunto de dados sintéticos, que incorporava diferentes possibilidades de agrupamentos, e considerando-se uma utilização em ambiente estático, o TASOM obteve desempenho similar ao SOM. Em um contexto dinâmico, onde o SOM clássico não funciona, o TASOM também obteve êxito, tendo erro muito baixo. Os parâmetros de análise nestes experimentos foram puramente matemáticos, extraindo assim a essência da ideologia de criação dos métodos.

Para validar os métodos em um contexto de utilização real, dois conjuntos de dados foram utilizados. Em todos eles tentou-se observar se os algoritmos seriam capazes de separar seus agrupamentos tendo em vista rótulos pré-estabelecidos. Está não é a aplicação para a qual estes sistemas foram criados, pois eles partem do pré-suposto não supervisionado e agrupam os dados com base em similaridade matemática. Mesmo assim, foi observado no experimento com dados de plantas Iris, que os métodos geraram agrupamentos interessantes tendo em vista a classe das amostras. Neste caso, o TASOM apresentou melhor desempenho que o SOM.

A utilização de métodos de agrupamento foi feita em problemas que possuem rótulo para servir como um processo auxiliador aos métodos supervisionados. Através da observação do agrupamento natural das entradas é possível observar comportamentos importantes para entender os dados. Se os agrupamentos separam o espaço da mesma forma que as classes separam as amostras, este é um sistema que possui separabilidade linear e baixo ruído, como foi demonstrado no experimento com a base de dados de Iris. O resultado contrário indica a presença de ruído ou a existência de dados linearmente inseparáveis, como foi demonstrado do experimento do MAPA. Estas informações podem guiar novas etapas no ciclo de aprendizagem de máquina, como um reavaliação dos processo de coleta ou uma melhor engenharia de características. Ainda é possível tomar decisões

a respeito dos métodos a serem utilizados, pois a morfologia do espaço a ser analisado se torna conhecida.

Os resultados encontrados neste trabalho apontam para o TASOM como uma boa alternativa ao SOM em ambientes estáticos, semi-estáticos e dinâmicos. Os objetivos propostos foram alcançados, tendo-se um método não-supervisionado de agrupamento incremental implementado e testado.

8.1 Futuros Trabalhos

Como pode-se observar nos experimentos do capítulo 6, o resultado do TASOM varia muito por causa da inicialização randômica dos pesos. Esta sensibilidade pode influenciar no desempenho. Portanto, é necessário pesquisar novos métodos de inicialização dos pesos iniciais da rede que não introduzam viés nos agrupamentos e que otimizem a performance do algoritmo.

Como pôde ser visto também, os parâmetros livres do TASOM não são de trivial inicialização. Uma possibilidade de melhoria é criar mecanismos de parametrização automática que otimizem a performance da rede.

Referências

ABYAK, M. What you see may not be what you get: A brief, nontechnical introduction to overfitting in regression-type models. 2013. Disponível em: <http://www.cs.vu.nl/~eliens/sg/local/theory/overfitting.pdf>. Citado na página 31.

AI Junkie. 2015. <http://www.ai-junkie.com/ann/som/som1.html>. Acessado: 2015-25-6. Citado 2 vezes nas páginas 11 e 46.

ALAHAKOON, D.; HALGAMUGE, S. K.; SRINIVASAN, B. Dynamic self-organizing maps with controlled growth for knowledge discovery. 2000. Disponível em: <http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=846732&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D846732>. Citado 2 vezes nas páginas 11 e 49.

ANDERSON, M. et al. Brainwash: A data system for feature engineering. 2013. Disponível em: <http://www.cs.stanford.edu/people/chrismre/papers/mythical_man.pdf>. Citado na página 28.

BATISTA, G. E. A. P. A.; MONARD, M. C. An analysis of four missing data treatment methods for supervised learning. 2003. Disponível em: <http://www.icmc.usp.br/pessoas/gbatista/files/aai2003.pdf>. Citado na página 27.

BRINK, H.; RICHARDS, J. W. *Real-World Machine Learning*. [s.n.], 2013. Disponível em: <http://www.manning.com/brink/RWML_MEAP_CH01.pdf>. Citado 13 vezes nas páginas 11, 23, 26, 27, 29, 31, 32, 33, 35, 37, 38, 40 e 45.

BROWNLEE, J. *Discover Feature Engineering, How to Engineer Features and How to Get Good at It*. 2014. <http://machinelearningmastery.com/discover-feature-engineering-how-to-engineer-features-and-how-to-get-good-at-it/>. Acessado: 2014-10-26. Citado 2 vezes nas páginas 28 e 29.

CARVALHO, H. M. Aprendizado de maquina voltado para mineração de dados: Arvores de decisao. 2014. Disponível em: <http://fga.unb.br/articles/0000/5556/TCC_Hialo_Muniz.pdf>. Citado na página 71.

DEKEL, O. From online to batch learning with cutoff-averaging. 2009. Disponível em: <http://msr-waypoint.com/en-us/um/people/oferd/papers/Dekel09.pdf>. Citado na página 42.

DERVOJEDA, K. et al. Case study - big data - artificial intelligence. 2013. Disponível em: <http://ec.europa.eu/enterprise/policies/innovation/policy/business-innovation-observatory/files/case-studies/09-bid-artificial-intelligence_en.pdf>. Citado 2 vezes nas páginas 19 e 20.

FISHER, R. *Iris Data Set*. 1936. <https://archive.ics.uci.edu/ml/datasets/Iris>. Citado na página 73.

FRITZKE, B. Growing cell structures—a self-organizing network for unsupervised and supervised learning. 1994. Disponível em: <http://www.sciencedirect.com/science/article/pii/0893608094900914>. Citado 2 vezes nas páginas 11 e 50.

HAMILTON, H. J. *Confusion Matrix*. 2012. <http://www2.cs.uregina.ca/~dbd/cs831/notes/confusion_matrix/confusion_matrix.html>. Acessado: 2014-11-1. Citado na página 37.

HAMILTON, H. J. *Confusion Matrix*. 2012. <http://www2.cs.uregina.ca/~dbd/cs831/notes/ROC/ROC.html>. Acessado: 2014-11-1. Citado 2 vezes nas páginas 11 e 40.

HARRINGTON, H. J. *Quotes About Measurement*. 2014. <http://www.goodreads.com/quotes/tag/measurement>. Acessado: 2014-10-8. Citado na página 35.

HE, H. et al. Incremental learning from stream data. 2011. Disponível em: <http://www.ele.uri.edu/faculty/he/PDFfiles/incrementaldata.pdf>. Citado 2 vezes nas páginas 41 e 43.

HILBERT, M.; LóPEZ, P. The world's technological capacity to store, communicate, and compute information. 2011. Disponível em: <http://www.sciencemag.org/content/332/6025/60>. Citado na página 19.

HOSSEINI, H. S.; SAFABAKHSH, R. Tasom: the time adaptive self-organizing map. 2001. Disponível em: <http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=844265&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D844265>. Citado 3 vezes nas páginas 11, 50 e 54.

KOHAVI, R. A study of cross-validation and bootstrap for accuracy estimation and model selection. 1995. Disponível em: <http://www.cs.iastate.edu/~jtian/cs573/Papers/Kohavi-IJCAI-95.pdf>. Citado na página 37.

KOHONEN, T. Self-organized formation of topologically correct feature maps. 1982. Disponível em: <http://www.cioslab.vcu.edu/alg/Visualize/kohonen-82.pdf>. Citado 4 vezes nas páginas 11, 45, 46 e 47.

LEARN, S. *Clustering*. 2014. <http://scikit-learn.org/stable/modules/clustering.html>. Acessado: 2014-11-1. Citado na página 35.

LEBANON, G. *A Taxonomy of Data Types*. 2010. <http://smlv.cc.gatech.edu/2010/03/23/a-taxonomy-of-data-types/>. Acessado: 2014-10-21. Citado na página 24.

LITTLE, R. J.; RUBIN., D. B. *Statistical Analysis with Missing Data*. [S.l.: s.n.], 1987. Citado na página 27.

LITTLESTONE, N. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. 1987. Disponível em: <http://www.cs.utsa.edu/~bylander/cs6243/littlestone1988.pdf>. Citado na página 42.

M., S.; GERSTING, J. *An Invitation to Computer Science*. [S.l.: s.n.], 1995. Citado na página 20.

MALISIEWICZ, T. *What is feature engineering?* 2014. <http://www.quora.com/What-is-feature-engineering>. Acessado: 2014-10-26. Citado na página 28.

MAPA. <http://www.agricultura.gov.br/ministerio>. Acessado: 2014-11-1. Citado na página 67.

MITCHELL, T. M. *Machine Learning*. [S.l.: s.n.], 1997. Citado 2 vezes nas páginas 19 e 35.

MOHR, M.; ROSTAMIZADEH, A.; TALWALKAR, A. *Foundations of Machine Learning*. [S.l.: s.n.], 2012. Citado 2 vezes nas páginas 34 e 45.

MUHLBAIER, M.; TOPALIS, A.; POLIKAR, R. Learn++.mt: A new approach to incremental learning. 2004. Disponível em: <http://users.rowan.edu/~polikar/RESEARCH/PUBLICATIONS/mcs04.pdf>. Citado na página 43.

NG, A. Y.; JORDAN, M. I. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. 2014. Disponível em: <http://papers.nips.cc/paper/2020-on-discriminative-vs-generative-classifiers-a-comparison-of-logistic-regression-and-naive-bayes.pdf>. Citado na página 33.

READ, J. et al. Batch-incremental versus instance-incremental learning in dynamic and evolving data. 2013. Disponível em: <http://albertbifet.com/wp-content/uploads/2013/10/IDA2012.pdf>. Citado 2 vezes nas páginas 41 e 42.

SCHNEIDER, J. *Cross Validation*. 1997. <http://www.cs.cmu.edu/~schneide/tut5/node42.html>. Acessado: 2014-11-1. Citado na página 36.

SCHNEIDER, J. *Judging Model Quality by Residuals*. 1997. <http://www.cs.cmu.edu/~schneide/tut5/node41.html>. Acessado: 2014-11-1. Citado na página 36.

SIMON, P. *Too Big to Ignore: The Business Case for Big Data*. [S.l.: s.n.], 2013. Citado na página 19.

TITANIC, Machine Learning from Disaster. 2012. <https://www.kaggle.com/c/titanic-gettingStarted>. Acessado: 2014-10-21. Citado na página 25.

TITANIC, Machine Learning from Disaster. 2012. <https://www.kaggle.com/c/titanic-gettingStarted/data>. Acessado: 2014-10-21. Citado 3 vezes nas páginas 11, 25 e 30.

UNIVERSITY of Eastern Finland. 2015. <http://cs.joensuu.fi/sipu/datasets/>. Acessado: 2015-25-6. Citado 2 vezes nas páginas 11 e 57.