UNIVERSIDADE FEDERAL DO ACRE

**EDKALLENN SILVA DE LIMA**

# Mineração de dados e aprendizagem de máquina na detecção de conflitos entre políticas

RIO BRANCO - ACRE

2020

EDKALLENN SILVA DE LIMA

Mineração de dados e aprendizagem de máquina na detecção de conflitos entre políticas

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal do Acre como requisito parcial para a obtenção do Grau de Mestre em Ciência da Computação. Área de concentração: Engenharia de Sistemas de Informação

Aprovada em <MES> de 2020.

BANCA EXAMINADORA

---

Prof. DRA. LAURA COSTA SARKIS - Orientador, UFAC

---

Prof. <NOME DO AVALIADOR>, <INSTITUIÇÃO>

---

Prof. <NOME DO AVALIADOR>, <INSTITUIÇÃO>

---

Prof. <NOME DO AVALIADOR>, <INSTITUIÇÃO>

RIO BRANCO - ACRE

2020

*Dedicatória(s): Este trabalho não seria possível sem a educação que me foi concedida, os ensinamentos e a sabedoria oriundos de você, Lucimar do Rego Albuquerque de Lima, muito mais que uma mãe, uma inspiração. <IN MEMORIAN>*

*"Assim como casas são feitas de pedras, a ciência é feita de fatos. Mas uma pilha de pedras não é uma casa e uma coleção de fatos não é, necessariamente, ciência".*

***Jules Henri Poincaré**, matemático, físico e filósofo da ciência francês*

# Agradecimentos

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis rutrum maximus fermentum. Duis id quam nibh. Aliquam cursus eget mauris at fermentum. Nam iaculis posuere sapien. Praesent facilisis nisl ipsum, at gravida est scelerisque non. In sed luctus sem. Integer odio sem, feugiat eu eros eget, lacinia lobortis mi. Donec dapibus, tellus non consequat aliquet, dui massa suscipit dolor, nec pretium tortor massa vitae lorem. Quisque non faucibus tellus. Quisque dignissim nibh quis sem imperdiet bibendum. Duis nec dictum turpis. Vestibulum auctor leo nulla, sed venenatis purus placerat et. Nam euismod mauris in justo semper laoreet.

Cras rutrum faucibus eros et feugiat. In sit amet viverra lorem. Sed euismod purus eget sodales ultricies. Praesent et blandit lorem. Pellentesque ut tempus velit. Aenean et vulputate arcu. Suspendisse finibus, tellus eu sollicitudin rutrum, augue nisi maximus lacus, ut blandit ante mi ut nulla.

Aenean metus dui, rhoncus sit amet pretium ut, laoreet quis ipsum. Sed non tempus turpis, ac eleifend lacus. Integer vel dui finibus, imperdiet neque porttitor, venenatis nunc. Vivamus egestas, turpis quis porttitor tincidunt, purus risus suscipit felis, nec consequat justo quam at purus. Pellentesque consequat sem ut nibh malesuada pulvinar. Quisque lobortis pulvinar urna, vitae luctus nulla ultricies vestibulum. Sed accumsan tortor tellus, in aliquam tellus sollicitudin eu. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia curae; Duis justo ipsum, vestibulum at magna id, ultrices lobortis turpis. Vivamus porttitor scelerisque odio sit amet posuere. Morbi fermentum ultricies sem, dictum pulvinar turpis.

# Resumo

Um dos principais componentes de segurança em sistemas computacionais é o controle de acesso, que é um mecanismo que verifica todas as solicitações de dados e recursos gerenciados pelos sistemas, determinando quando as solicitações são permitidas ou negadas. Uma política de controle de acesso descreve qual ação um sujeito pode fazer (permissão), não pode fazer (proibição) ou é obrigado a fazer (obrigação) sobre um objeto em um dado contexto. Muitas vezes as políticas de segurança apresentam conflitos. Esses conflitos podem gerar problemas, como acessos não autorizados ou negações a acessos legítimos. Diante disto, a detecção de conflitos entre políticas de controle de acesso torna-se uma atividade sensível e crítica dentro de sistemas. Este trabalho estuda se é pertinente e válido o uso de mineração de dados associada a técnicas de aprendizagem de máquina para detectar os conflitos entre estas diversas políticas. Seu objetivo é modelar uma forma de detecção automática, mediante o uso de técnicas de mineração de dados (e *aprendizagem de máquina*) dos conflitos, sejam eles, diretos ou indiretos entre as políticas de controle de acesso de um sistema.

**Palavras-chave**: Controle de Acesso. Mineração de dados. Aprendizagem de máquina. Conflitos diretos. Conflitos indiretos. Detecção de conflitos.

# Abstract

One of the main security components in computer systems is access control, which is a mechanism that checks all requests for data and resources managed by the systems, determining when requests are allowed or denied. An access control policy describes what action a subject can do (permission), cannot do (prohibition) or is obliged to do (obligation) on an object in a given context. Security policies often have conflicts. These conflicts can lead to problems, such as unauthorized access or denial of legitimate access. Given this, the detection of conflicts between access control policies becomes a sensitive and critical activity within systems. This paper studies whether the use of data mining associated with machine learning techniques to detect conflicts between these different policies is relevant and valid. Its objective is to model a form of automatic detection, using data mining techniques (and machine learning) of conflicts, whether direct or indirect between a system's access control policies.

**Keywords**: Access control. Data mining. Machine learning. Direct conflicts. Indirect conflicts. Conflict detection.

# Lista de Figuras

# Lista de Tabelas

# Lista de Abreviaturas e Siglas

<ABREVIATURA> : <SIGNIFICADO>;
<ABREVIATURA> : <SIGNIFICADO>;
<ABREVIATURA> : <SIGNIFICADO>;

# Sumário

# Capítulo 1

# Introdução

## 1.1 Contextualização

O volume de dados e informações cresce exponencialmente a cada ano [Alecrim, 2019], portanto, há uma frequente e ininterrupta demanda por mais infraestrutura de TI nas empresas, nos governos e mesmo nos usuários domésticos [Machado, 2014] e, mais ainda, por um correto tratamento, destino e interpretação à imensidão de dados gerados por pessoas, empresas e governos.

Em uma ampla variedade de campos, os dados estão sendo coletados e acumulados em um ritmo acelerado.[Fayyad et al., 1996][Lima and Pereira, 2012] e há, assim, uma crescente demanda por análise adequada destes. Neste contexto se insere a mineração de dados com suas técnicas para tratamento e extração de conhecimento desse volume crescente de dados.[Silva et al., 2017][Ferrari and Castro Silva, 2017]

Este trabalho usa diversas tarefas da mineração de dados para modelar uma hipótese que possibilite detectar conflitos em políticas de controle de acesso. Para isso, diversos algoritmos de classificação serão explorados, descritos e utilizados com ênfase nas redes neurais e outras técnicas lineares de classificação.

As políticas de proteção, confidencialidade e confiabilidade da informação, como as de controle de acesso, sendo parte da área de segurança computacional, são uma das formas de garantir, mediante o estabelecimento de regras, padrões e normas a salvaguarda e a disponibilidade das informações dos sistemas.[Sarkis, 2017][Bui et al., 2019]. Este tema, cf. [Ueda, 2012, p.1] “é um tema de pesquisa importante dentro do contexto de segurança de sistemas, pois é um dos componentes fundamentais em qualquer sistema de computação."

Segundo [Li and Tripunitara, 2006], um aspecto muito relevante e muitas vezes tra-

tado com pouca ênfase na construção de sistemas é a formulação, gerenciamento e manutenção de políticas de segurança da informação, principalmente as de controle de acesso.

Nas palavras de [Ueda, 2012, p.1],

> A definição dessas políticas é normalmente orientada por modelos que fornecem um conjunto de regras e mecanismos para o funcionamento seguro de uma representação abstrata de sistemas. Porém, a administração de tais políticas frequentemente *se torna um processo complexo*, pois deve garantir que elas sejam eficientes e que não comprometam o *desempenho* dos sistemas. *[Grifo do autor.]*

Uma política, como as de controle de acesso, descrevem qual ação um sujeito (em um sistema) pode fazer (*permissão*), não pode fazer (*proibição*) ou é obrigado a fazer (*obrigação*) sobre um objeto em um dado contexto [Sarkis, 2017]. Da mesma forma são conceituadas as *normas* e os conjuntos de normas usados para lidar com a autonomia e a diversidade de interesses entre os diferentes agentes em um sistema multiagentes como o descrito e estudado em [Silvestre, 2017]. Essas normas que regulam as ações dos agentes são semelhantes às políticas de controle e são, cf. [Silvestre, 2017] e [Sarkis et al., 2016] fatores importantes para garantir a eficácia dos sistemas.

Infelizmente, porém, em contextos reais, muitas vezes as políticas de segurança (e *normas*) apresentam conflitos entre si. Estes surgem quando, por exemplo, duas políticas regulando o mesmo comportamento de determinado objeto em um sistema estão ativas, mas uma delas obriga (ou permite) a realização de determinado comportamento ou ação enquanto a outra proíbe o mesmo. [Sarkis, 2017][Silvestre, 2017]. A detecção automatizada destes conflitos, com alta acurácia e custo computacional conveniente, é o problema de pesquisa deste trabalho. A descrição pormenorizada dos conflitos entre políticas encontra-se na seção 2.2 deste trabalho.

### 1.1.1 Problema

O problema investigado neste trabalho consiste na *detecção de conflitos de forma automatizada usando técnicas de mineração de dados e aprendizagem de máquina* que apresentem acurácias superiores a 95% e que, ao analisar simultaneamente várias políticas ao se inserir uma nova instância não leve a um custo computacional exponencial.

### 1.1.2 Justificativa

Na detecção de conflitos em políticas, geralmente, cf. revisão da literatura, usam-se abordagens semelhantes as de [Sarkis, 2017] baseadas em análise de ontologias entre os atributos que compõem uma política e em regras de propagação das mesmas ou procedimentos como os descritos em [Silvestre, 2017] que utilizam lógica deôntica[1] para encontrar os conflitos.

Entretanto, para que os conflitos sejam detectados nessas abordagens as políticas são analisadas em pares (e sem filtros para agrupamentos) e mesmo quando são verificadas múltiplas *normas* ou políticas, como em [Silvestre, 2017], todas elas devem ser novamente "consultadas"ou "varridas"a cada nova instância de uma política inserida no sistema (para que o conflito seja ou não detectado). O trabalho de [Shoham and Tennenholtz, 1995] provou que esta forma de analisar políticas é um problema NP-completo, ou seja, ainda não foi provado que esta classe de problemas pode ser resolvida em tempo *polinomial*, sendo assim, são computacionalmente onerosos a cada vez que uma instância nova de política é analisada (em tempo d execução sendo, normalmente exponencial). Com o crescimento orgânico, natural e temporal da quantidade de políticas em um sistema computacional este fato tende a tornar a manutenção e gerenciamento das políticas algo *computacionalmente custoso e dispendioso.*

Conforme se visualiza na seção 2.2.1, as técnicas e algoritmos de aprendizagem de máquina juntamente com as de mineração de dados foram utilizadas com resultados promissores na detecção de conflitos, principalmente em [Obaidat and Macchairolo, 1994], [Chen, 2011] bem como em [Christodoulou and Kontogeorgou, 2008] e [Jin et al., 2002] que abordam problemas variados como detecção de colisões em voos, segurança de acesso computacional, incidentes em rodovias e intrusão de sistemas — todos de alguma forma relacionados à conflitos entre normas, regras, políticas ou direção.

Em grandes organizações as *políticas de segurança*, como as de controle de acesso, pela quantidade de objetos, modalidades, sujeitos e ações inerentes a essas instituições tendem a ter grande quantidade de informações e elas crescem conforme o uso diário e constante dos sistemas computacionais. organizações grandes, com múltiplos objetos e ações possuem centenas, às vezes, milhares de políticas (entre elas *políticas de controle de acesso*, por exemplo).[Fugini and Bellettini, 2004].

Neste ambiente e considerando que as políticas analisadas em pares, como em [Sarkis, 2017],

[1] A lógica *deôntica* é um tipo de lógica usada para analisar de modo formal as normas e as proposições que tratam dessas normas [Silvestre, 2017]

tem alto custo computacional por este, como visto em [Shoham and Tennenholtz, 1995], ser um problema NP-completo e com a estratégia de minimizar a complexidade deste problema otimizando-o baseando suas soluções no conhecimento adquirido e já existente nas organizações (os *datasets* de políticas), a mineração de dados com técnicas de aprendizagem de máquina surgem como possibilidades de solução na detecção de conflitos tanto em tempo de design quanto em tempo de execução, pois, minimiza este custo computacional ao se aproveitar da "história"temporal das políticas da organização, mediante o conhecimento adquirido e "treinado"pelos algoritmos de aprendizagem de máquina.

### 1.1.3 Hipótese

Diante do contexto apresentado na seção anterior e também por [Fugini and Bellettini, 2004] tem-se *como hipótese deste trabalho*, que o problema de detectar conflitos entre políticas *pode ser convertido e transformado* em uma tarefa de classificação da mineração de dados e que o uso de algoritmos de aprendizagem de máquina associados a técnicas de *data mining* para detectar estes conflitos configure um método que apresente precisão e acurácia superiores a 95%.

## 1.2 Objetivos

### 1.2.1 Objetivo geral

O objetivo deste trabalho é propor que o problema da detecção de conflitos em políticas pode ser convertido em um problema de *data mining* (mineração de dados) resolvido pela tarefa da classificação além de modelar e sumariar uma forma de detectar estes conflitos mediante o uso de diferentes algoritmos e técnicas da aprendizagem de máquina que consigam acurácias elevadas.

### 1.2.2 Objetivos específicos

Os objetivos específicos deste trabalho são:

- Estabelecer a relação entre machine learning, técnicas de mineração de dados e o problema do conflito entre políticas;
- Determinar e comparar quais algoritmos e técnicas são mais adequados para cada tipo de conflito nas políticas (ou normas) usando as suas acurácias;

- Usar e comparar o desempenho, a precisão e taxa de acertos das principais técnicas usadas no aprendizado de máquina: redes neurais artificiais (RNA), Support Vector Machines (SVM) e redes neurais recorrentes e profundas.
- Transformar o problema da detecção de conflitos em uma tarefa de classificação da mineração de dados mantendo acurácias elevadas;
- Usar *frameworks* de aprendizado de máquina como TensorFlow, Keras, ou Torch, na construção, treinamento e teste de redes neurais, comparando-os, quando adequado, para o problema específico deste trabalho;
- Estabelecer a detecção de conflitos em políticas (ou normas) como uma classe de problemas a serem resolvidos de forma eficiente por técnicas de aprendizagem de máquina.

## 1.3 Solução Propostas

Diante da hipótese apresentada na seção 1.1.3, a solução para o problema apresentado neste trabalho na seção 1.1.1 concentra-se, prioritariamente, em mostrar que *converter* (ou *transformar*) a detecção de conflitos a um *problema de classificação* da mineração de dados associado a técnicas de aprendizagem de máquina, reestruturando os atributos do *dataset*, se necessário, se configura um método com acurácia superior a 95%.

*A primeira solução* proposta para conflitos diretos entre políticas é usar as técnicas e algoritmos de classificação (aprendizado supervisionado) para realizar a detecção automatizada de conflitos. Para isso, propõem-se:

- Usar, inicialmente, uma rede neural (um perceptron de uma camada ou com somente uma camada oculta e apenas com *forward*) como técnica algorítmica para a detecção de conflitos e
- Construir a arquitetura de uma rede neural multicamadas (com camadas ocultas), e retropropagação (*backpropagation*), comparando-a com outro classificador linear, como, por exemplo, o SVM, para estabelecer qual técnica de mineração de dados na detecção de conflitos em políticas é mais precisa.

Realizado os múltiplos experimentos, atestar a hipótese mediante os resultados apresentados.

## 1.4 Método de Pesquisa

O método de pesquisa relatando todos os passos necessários para demonstrar os objetivos descritos na seção 1.2 e de que forma eles foram atingidos serão pormenorizadamente detalhados na seção 3 deste trabalho.

## 1.5 Resultados Esperados

Ao fim deste trabalho os seguintes resultados são esperados:

- Mostrar que o problema da detecção de conflitos em políticas pode ser convertido em um problema de *data mining* (mineração de dados) resolvido pela tarefa da classificação;
- Demonstrar que o problema da detecção de conflitos é um prolema linearmente separável;
- Mostrar que a política nova (*instância inédita*) é ou não conflitante imediatamente após a criação da mesma usando como base o treinamento da rede neural no *dataset* de políticas existente;
- Modelar e resumir uma forma de detectar estes conflitos mediante o uso de diferentes algoritmos e técnicas da aprendizagem de máquina que consigam acurácias superiores a 95%;
- Estabelecer a relação entre machine learning, técnicas de mineração de dados e o problema do conflito entre políticas;
- Determinar e comparar quais algoritmos e técnicas são mais adequados para cada tipo de conflito nas políticas (ou normas) usando as suas acurácias;
- Usar e comparar o desempenho, a precisão e taxa de acertos das principais técnicas usadas no aprendizado de máquina: redes neurais artificiais (RNA), Support Vector Machines (SVM); redes neurais recorrentes e profundas;
- Estabelecer a detecção de conflitos em políticas (ou normas) como uma classe de problemas a serem resolvidos de forma eficiente por técnicas de aprendizagem de máquina.

## 1.6 Limitações do Trabalho

Não faz parte do escopo deste trabalho:

- Delinear um modelo de política com objetivos semânticos diferenciados. Para os experimentos deste trabalho será usado o modelo de políticas descrito em [Sarkis, 2017] e em [Sarkis et al., 2016]
- Analisar comparativamente os modelos de extensão de políticas em um determinado contexto;
- Usar redes neurais convolucionais profundas na detecção dos conflitos;
- Abordar a semântica em políticas;

## 1.7 Organização do trabalho

Este trabalho está organizado da seguinte forma:

- Neste capítulo 1 estão dispostas a contextualização, o problema da pesquisa, a justificativa, a hipótese de pesquisa, os objetivos, as soluções que foram propostas, os resultados esperados e as limitações do trabalho
- No Capítulo 2 apresenta-se todo o referencial teórico compreendendo uma revisão bibliográfica sobre os principais temas desta dissertação, como políticas, detecção de conflitos, mineração de dados e aprendizagem de máquina (e seus algoritmos principais)
- No Capítulo 3 são mostrados o método e os experimentos e resultados obtidos.
- No Capítulo 4 mostra-se o cronograma para a finalização da dissertação
- No Capítulo 5 apresenta-se as conclusões e propostas para trabalhos futuros relacionados a esta pesquisa.

# Capítulo 2

# Referencial teórico

Nesta seção, alguns conceitos fundamentais da pesquisa bibliográfica realizada serão explanados com o intuito de atingir os objetivos descritos na seção 1.2 e o entendimento da solução proposta neste trabalho. Serão abordados modelos de políticas, como as de controle de acesso e conflitos entre as mesmas além de trabalhos relacionados sobre o tema. Serão abordados também temas como mineração de dados, aprendizagem de máquina, algoritmos de classificação com destaque para as *Redes Neurais Artificiais* e **SVM** — *Support Vector Machines* que são a base da hipótese deste trabalho.

## 2.1 Políticas em sistemas de segurança

A política de segurança em um sistema computacional garante a proteção de suas informações. Dentre as tecnologias utilizadas para assegurar essas propriedades, temos o controle de acesso. [Sarkis et al., 2016]

O controle de acesso é o mecanismo central para atingir os requisitos de segurança em sistemas de informação. [Wang et al., 2010]. Dessa forma, trata-se de uma tecnologia indispensável para quem faz uso de qualquer tipo de sistema, podendo basear-se ou coexistir com outros serviços de segurança. [Sandhu and Samarati, 1996].

Os modelos de controle de acesso fornecem um conjunto de regras e mecanismos para o funcionamento seguro dos sistemas, sendo responsáveis pela definição de políticas de controle de acesso. As políticas são diretrizes de alto nível que determinam como os acessos são controlados e decisões de acessos são estabelecidas. [De Capitani di Vimercati et al., 2005] [Sarkis, 2017]

### 2.1.1 Políticas de Controle de acesso

As políticas de controle de acesso tradicionais, inicialmente foram chamadas de autorizações e tinham a seguinte forma: sujeito, objeto, operação. Estas autorizações especificavam as operações que os sujeitos podiam executar sobre os objetos. [De Capitani di Vimercati et al., 2

Uma política de controle de acesso tem como objetivo definir ou limitar o comportamento atual ou futuro de objetos para garantir que as suas ações estejam alinhadas com os objetivos da empresa.[Dunlop et al., 2002][Sarkis, 2017]

### 2.1.2 Modelos de políticas

[Moffett and Sloman, 1994] consideram que um modelo de política deve ter pelo menos os seguintes atributos: *modalidade, sujeito, objeto e ação.* A modalidade da política envolve definir uma autorização, uma permissão ou uma proibição. O sujeito da política é a quem ela é direcionada. O objeto define o conjunto de objetos, no qual a política está direcionada. A ação é especificada como operações que podem ser executadas em objetos no sistema [Moffett and Sloman, 1994]. Os atributos da política apresentada por estes autores são considerados em outros trabalhos, com a mesma conotação aqui utilizada, para a definição de uma política.[Sarkis, 2017]

### 2.1.3 Modelo de Política utilizado

Cf. [Sarkis, 2017]:

> Definir uma política de controle de acesso não é uma tarefa simples, principalmente porque algumas vezes é necessário representar formalmente políticas complexas, tais como as que tem origem em práticas de leis e regulamentos organizacionais.

Desta forma, a definição da política deve combinar todos os diferentes regulamentos para ser executada e considerar todas as possíveis ameaças adicionais relativas ao uso de sistemas. [De Capitani di Vimercati et al., 2005]

O modelo de política utilizado neste trabalho baseia-se inteiramente no modelo proposto por [Sarkis, 2017] e [Sarkis et al., 2016].

Portanto, de acordo com [Sarkis, 2017, p.36]:

Uma política é uma tupla da forma:

$$Policy = KP \times Org \times SR \times AA \times OV \times Ac \times Dc \tag{2.1}$$

Onde $KP$ descreve o tipo de política (uma proibição (F), da palavra em inglês Forbidden; uma permissão (P); ou uma obrigação (O)). $Org.$ relata o local (ambiente) onde a política deve ser cumprida, isto é, a organização na qual os sujeitos devem cumprir a política. $SR$ descreve a quem (entidades) se destina a política (pode ser um sujeito $s \in S$ ou um papel $r \in R$, ou seja, $SR = S \cup R$. Um sujeito pode ser um usuário $u \in U$ ou uma organização $org \in Org$ representando o grupo de sujeitos que devem cumprir com a política, isto é, $S = U \cup Org$). $AA$ identifica uma ação $a \in A$ ou uma atividade $act \in Act$ (uma atividade é a união de várias ações relacionadas). $OV$ relata um objeto $o \in O$ ou uma visão $v \in V$ que está sendo manipulada pela ação/atividade (uma visão é a união de vários objetos). $Ac$ é a condição de ativação da política e $Dc$ é a condição de desativação da política. Uma condição constitui a configuração para um evento, em termos de que a política deva seguir.

Em [Sarkis, 2017] definiu-se $Ac$ e $Dc$ como datas, desta forma $Ac$ é a data de ativação da política e $Dc$ é a data de desativação.

A figura 2.1 exemplifica o modelo de políticas utilizadas neste estudo para a mineração de dados e aprendizagem de máquina.

```
Policie10-> [Permitted, Administrative_Unit, PROTOCOLIZADOR3, Close, ProcessDispatch, Sun Aug 18 20:06:14 BRT 15, Tue Dec 31 20:06:14 BRT 15]
Policie11-> [Permitted, Administrative_Unit, PROTOCOLIZADOR3, Create, ProcessDispatch, Sun Aug 18 20:06:14 BRT 15, Tue Dec 31 20:06:14 BRT 15]
Policie12-> [Permitted, Administrative_Unit, PROTOCOLIZADOR3, Record, ProcessDispatch, Sun Aug 18 20:06:14 BRT 15, Tue Dec 31 20:06:14 BRT 15]

Policie25-> [Forbidden, Institution, null, Record, Process, Sun Aug 18 20:06:14 BRT 15, Tue Dec 31 20:06:14 BRT 15]
Policie26-> [Forbidden, Institution, null, Generate, Process, Sun Aug 18 20:06:14 BRT 15, Tue Dec 31 20:06:14 BRT 15]
Policie27-> [Forbidden, Institution, PROTOCOLIZADOR3, Move, ProcNURCADesp, Sun Aug 18 20:06:14 BRT 15, Tue Dec 31 20:06:14 BRT 15]
Policie28-> [Forbidden, Institution, PROTOCOLIZADOR3, Acess, ProcNURCADesp, Sun Aug 18 20:06:14 BRT 15, Tue Dec 31 20:06:14 BRT 15]
Policie29-> [Forbidden, Institution, PROTOCOLIZADOR3, Record, ProcNURCADesp, Sun Aug 18 20:06:14 BRT 15, Tue Dec 31 20:06:14 BRT 15]

Policie43-> [Obliged, Administrative_Unit, null, Open, Process, Sun Aug 18 20:06:14 BRT 15, Tue Dec 31 20:06:14 BRT 15]
Policie44-> [Obliged, Administrative_Unit, null, Acess, Process, Sun Aug 18 20:06:14 BRT 15, Tue Dec 31 20:06:14 BRT 15]
Policie45-> [Obliged, Administrative_Unit, PROTOCOLIZADOR, Open, Process, Sun Aug 18 20:06:14 BRT 15, Tue Dec 31 20:06:14 BRT 15]
Policie46-> [Obliged, Administrative_Unit, PROTOCOLIZADOR, Acess, Process, Sun Aug 18 20:06:14 BRT 15, Tue Dec 31 20:06:14 BRT 15]
Policie47-> [Obliged, Administrative_Unit, PROTOCOLIZADOR3, Open, Process, Sun Aug 18 20:06:14 BRT 15, Tue Dec 31 20:06:14 BRT 15]
Policie48-> [Obliged, Administrative_Unit, PROTOCOLIZADOR3, Acess, Process, Sun Aug 18 20:06:14 BRT 15, Tue Dec 31 20:06:14 BRT 15]
Policie49-> [Obliged, Administrative_Unit, MARY, Open, Process, Sun Aug 18 20:06:14 BRT 15, Tue Dec 31 20:06:14 BRT 15]
```

Figura 2.1: Modelo das políticas utilizadas no estudo

Fonte: compilação do autor

## 2.2 Detecção de conflitos

Os conflitos podem ocorrer quando diferentes conjuntos de condições resultam em permitir e negar simultaneamente, ao mesmo papel, à mesma solicitação, ou proibir e obrigar o mesmo papel, à mesma solicitação.

Diz-se que duas regras estão em conflito quando o cumprimento de uma das regras viola a outra e vice-versa.

Ex:

```
{P1= Permitido, na Universidade, Paulo, acessar processos administrativos}

{P2= Proibido, na Universidade, Paulo, acessar processos administrativos}
```

A capacidade de um sistema reconhecer um estado inconsistente em andamento ou em potencial é denominada **detecção de conflitos.**

Em um conflito indireto, as políticas conflitantes regulam ações diferentes (mas relacionadas) executadas por diferentes sujeitos (porém, relacionados) sobre objetos diferentes (mas, relacionados) em organizações diferentes (mas, relacionadas).

Além disso, um conflito indireto pode ainda ocorrer, mesmo quando as políticas em conflito não têm modalidades contraditórias ou contrárias.

Ex:

```
P3 = Obrigado, Empresa E, Funcionário, receber, avaliação, mensal

P4= Permitido, Empresa E, Analista, conceder, avaliação, mensal
```

Este conflito não seria detectado diretamente, porém há um conflito se considerarmos os relacionamentos.

### 2.2.1 Trabalhos Relacionados

[Bui et al., 2019] discorre acerca de mineração de dados em políticas de controle de acesso, especificamente do modelo RBAC (*role-based access control*). [Lupu and Sloman, 1999] discorre sobre conflitos no gerenciamento de sistemas distribuídos com base em políticas de controle de acesso. [Koch et al., 2002], estudou a detecção e resolução de conflitos nas especificações das políticas de controle de acesso. [Neri et al., 2012], abordou a detecção de conflitos em políticas de segurança usando a tecnologia da Web Semântica.

[Obaidat and Macchairolo, 1994] aborda um sistema de rede neural multicamada para segurança de acesso a computadores. [Christodoulou and Kontogeorgou, 2008] aborda a detecção de conflitos sobre a ótica da prevenção de colisões no voo livre de aeronaves comerciais usando redes neurais e programação não linear. [Mukkamala et al., 2002] estuda a detecção de invasões usando redes neurais e máquinas de vetores de suporte. Neste último trabalho a ideia é descobrir padrões úteis ou características que descrevam o com-

portamento intrusivo de um usuário em um sistema e os autores usam este conjunto de características para construir classificadores que puderam reconhecer anomalias e intrusões conhecidas.

[Thenmozhi et al., 2018] apresenta uma discussão sobre a criação criação de perfis de autores multilíngues em mensagens SMS usando a abordagem de aprendizado de máquina com seleção estatística de recursos que mostra formas de tratar os hiperparâmetros de uma rede neural usando uma abordagem estatística.

[Jin et al., 2002] aborda o desenvolvimento e adaptação de redes neurais probabilísticas construtivas na detecção de incidentes em rodovias. [Debar et al., 1992] estuda um componente de rede neural para um sistema de detecção de intrusão. Ainda em detecção de conflitos [Chen, 2011] trabalha com a detecção e resolução de conflitos de vôo com base em redes neurais.

Todos estes trabalhos foram base para o estudo, amadurecimento bibliográfico e aprofundamento teórico sobre o problema e as soluções propostas neste trabalho, principalmente aqueles relacionados a intrusões e detecção de conflitos aéreos pois não só serviram de inspiração como provavelmente as técnicas possam ser extrapoladas para o uso na detecção de conflitos, tema deste estudo.

[Sarkis, 2017], [Silvestre, 2017] e [Sarkis et al., 2016] são os trabalhos-base no estudo da detecção de conflitos, da determinação do modelo das políticas utilizadas e analisadas além de algumas das definições empregadas neste trabalho. As hipóteses, soluções e problemas estudados por este projeto derivam do estudo inicial de [Sarkis, 2017] e são uma complementação do mesmo.

### 2.2.2 Mineração de Dados

Uma das características de nossa era é produção de dados em grande volume, velocidade e variedade de todas as formas, por dispositivos espalhados em toda parte. Entretanto, dados, mesmo em grande quantidade, são apenas dados. É preciso produzir informação e conhecimento para explorar as vantagens que essa massa pode trazer. O dado necessita ser, de alguma forma, analisado, tratado para que informações e conhecimento possa ser extraído. [Amaral, 2016] [Ferrari and Castro Silva, 2017]

Conforme [Fayyad et al., 1996]:

> [...] Os computadores permitiram que os humanos coletassem mais dados

> do que podemos digerir, é natural [,portanto] recorrer a técnicas computacionais para nos ajudar a desenterrar padrões e estruturas significativas a partir dos volumosos volumes de dados. Por isso, [a mineração de dados] é uma tentativa de resolver um problema que a era da informação digital transformou em realidade para todos nós: sobrecarga de dados.

Para [Silva et al., 2017],

> De forma simplificada, a mineração de dados pode ser definida como um processo automático ou semiautomático de explorar analiticamente grandes bases de dados, com a finalidade de descobrir padrões relevantes que ocorrem nos dados e que sejam importantes para embasar a assimilação de informação importante, suportando a geração de conhecimento.

Ainda, segundo [Fayyad et al., 1996],

> O termo mineração de dados tem sido usado principalmente por estatísticos, analistas de dados e comunidades de sistemas de informações gerenciais (MIS). Ele também ganhou popularidade no campo do banco de dados. O termo descoberta de conhecimento em bancos de dados [(KDD, da sigla em Inglês)] foi cunhada no primeiro workshop do KDD em 1989 para enfatizar que o conhecimento é o produto final de uma descoberta baseada em dados. Foi popularizado nos campos de IA e aprendizado de máquina.

Dessa forma, a mineração de dados é parte integrante de um processo mais amplo, conhecido como descoberta de conhecimento em bases de dados (*Knowledge Discovery in Databases*, ou *KDD*)[Fayyad et al., 1996]. Embora se use *mineração de dados* como sinônimo de KDD, a terminologia é empregada para a etapa de *descoberta* do processo de KDD, que inclui a *seleção* e *integração* das bases de dados, a *limpeza* da base, a *seleção e transformação* dos dados, a *mineração*(propriamente) e a *avaliação* dos dados. [Ferrari and Castro Silva, 2017][Silva et al., 2017].

Assim, a mineração de dados é definida em termos de esforços para a descoberta de padrões em bases de dados. A partir destes padrões descobertos, há condições de se gerar conhecimento útil para um processo de tomada de decisão (ou a geração de conhecimento para esta tomada).

O KDD (Knowledge Discovery in Database) é um processo de busca de conhecimento em bancos de dados e, de modo geral, consiste de uma sequência iterativa de

passos(ou **etapas**)[1]: limpeza de dados, integração dos dados, Seleção, Transformação e Mineração dos dados, Avaliação dos padrões e Apresentação e Assimilação do conhecimento. Este processo é iterativo e, em alguma etapa, pode-se voltar para uma anterior. [Silva et al., 2017]

Neste trabalho as tarefas de seleção e transformação dos dados farão parte da etapa chamada de pré-processamento (cf. [Silva et al., 2017] e serão descritas na seção 3.

O termo **modelo de conhecimento**(ou hipótese) é utilizado na literatura (e neste trabalho) para fazer referência a um padrão ou conjunto de padrões descobertos (que é, enfim, o *propósito* do processo de KDD). Estes padrões são conhecimentos representados segundo as normas sintáticas de alguma linguagem formal. Estes padrões podem ser classificados em dois tipos: preditivos e descritivos. O intuito dos preditivos é resolver um problema específico de prever os resultados ou valores de um ou mais atributos, em função dos valores de outros atributos. Os descritivos (ou informativos) tem o intuito de apresentar informações interessantes e importantes sobre os dados que um especialista de domínio possa não conhecer. Modelos de conhecimento compostos exclusivamente por padrões preditivos são chamados de modelos preditivos, enquanto que modelos descritivos são modelos de conhecimento compostos por padrões descritivos. [Goldschmidt and Passos, ][Ferrari and Castro Silva, 2017][Silva et al., 2017]

Neste contexto, este trabalho tem como objetivo modelar uma forma de detectar, mediante o uso de técnicas da mineração de dados (e aprendizagem de máquina) os conflitos entre as políticas de controle de acesso de um sistema. Vários algoritmos e técnicas serão utilizados sendo que eles serão devidamente analisados usando-se métricas específicas para cada algoritmo.

### 2.2.3 Aprendizagem de máquina

Aprendizado de máquina ou *machine learning* é um braço da Inteligência Artificial que emprega técnicas e algoritmos na criação de modelos computacionais dos quais a característica princial é a capacidade de descobrir padrões em um grande volume de dados ou de melhorar o desempenho de uma determinada tarefa através da experiência (do *reforço*).[Mohri, 2018] [Alpaydin, 2014] [Swamynathan, 2019]

Nas palavras de Arthur Lee Samuel, considerado um dos pioneiros na área de inteligência artificial [Wiederhold and McCarthy, 1992], aprendizado de máquina é "o campo

[1] O processo de KDD, segundo [Fayyad et al., 1996] é composto por: *Seleção de dados; Pré-processamento; Transformação; Mineração; Análise e assimilação de resultados*

de estudo que dá aos computadores a capacidade de aprender sem serem explicitamente programados". [Simon, 2013, p. 89].

Aprendizado de máquina tem sido aplicado na automatização de funções que para os humanos são executadas intuitivamente, mas que são difíceis de definir formalmente. [Sarkar et al., 2017]

Assim, de forma geral, a aprendizagem de máquina tem por objetivo estudar e desenvolver métodos computacionais para obter sistemas capazes de adquirir conhecimento de forma automatizada. [Lima et al., 2016]

A capacidade de determinados algoritmos tem de aprender a partir de exemplos é chamado de **aprendizado indutivo**. Estes algoritmos aprendem relacionamentos eventualmente existentes entre os dados, mostrando o resultado nos modelos de conhecimento gerado. [Goldschmidt and Passos, ][Alpaydin, 2014]

As principais abordagens de aprendizado que determinam os 3 principais tipos de aprendizagem são: a aprendizagem supervisionada, a aprendizagem não-supervisionada e a aprendizagem por reforço. [Russell and Norvig, 2013].

Na **aprendizagem não-supervisionada** o modelo/hipótese busca padrões na entrada, embora não seja fornecido nenhum feedback explícito. Portanto, na abordagem não-supervisionada não há, nos dados, uma classe, não há um rótulo prévio, ou seja, não existe a informação da saída desejada. O processo de aprendizado busca identificar regularidades entre os dados e não é necessária a divisão prévia dos dados em dados de treinamento, validação e teste. A tarefa mais comum de aprendizagem não supervisionada é o agrupamento. [Russell and Norvig, 2013] [Silva et al., 2017] [Goldschmidt and Passos, ] [Amaral, 2016]

Na **aprendizagem supervisionada** o modelo/hipótese observa alguns exemplos de pares de entrada e saída, e aprende uma função que faz o mapeamento entre a entrada e a saída. Portanto, ela compreende a abstração de um modelo a partir dos dados apresentados na forma de pares ordenados (*entrada, saída, saída desejada*). Há, assim, uma *classe*, ou um atributo especial com o qual se pode comparar e validar o resultado.

Na **aprendizagem por reforço**, aprende-se a partir de uma série de reforços — recompensas ou punições. Não está disponível, geralmente, na aprendizagem por reforço, para o algoritmo de aprendizado de máquina, um conjunto de dados para treinamento. O aprendizado se dá, então, pela interação com o ambiente que se deseja atuar por um determinado período com o objetivo de melhorar o desempenho de uma determinada

tarefa. [Russell and Norvig, 2013] [Amaral, 2016] [Silva, 2019]

### 2.2.4 Algoritmos de classificação

Conforme [Rocha et al., 2012], "o termo *classe* deve ser usado quando existe informação sobre quantas e quais são as partições presentes em um conjunto de dados, bem como qual exemplar pertence a qual partição".

Comumente denomina-se *classificação* o processo pelo qual se determina uma função de mapeamento capaz de indicar a qual classe pertence algum exemplar de um domínio sob análise, baseando-se em um conjunto já classificado. [Silva et al., 2017].

Assim, classificação é uma técnica de mineração de dados (aprendizado de máquina) usada para prever a associação ao grupo para instâncias de dados.[Kesavaraj and Sukumaran, 2013]. É, segundo [Amaral, 2016] e [Shazmeen et al., 2013], a tarefa mais utilizada em mineração de dados. Além de ser a mais complexa e a que possui a maior quantidade de algoritmos disponíveis.[Kesavaraj and Sukumaran, 2013]

A classificação é uma das tarefas preditivas de Mineração de Dados e aprendizado de máquina. Tarefas de predição consistem na análise de um dataset (conjunto de dados), descritos por atributos e rótulos associados com o objetivo de descobrir um modelo capaz de mapear corretamente cada um dos dados a seus rótulos. Esse objetivo é alcançado por meio de técnicas chamadas de supervisionadas. A análise preditiva é dividida em categórica, também chamada de classificação ou em numérica, também chamada de regressão. [Silva et al., 2017] [Kesavaraj and Sukumaran, 2013] [Ferrari and Castro Silva, 2017] [Goldschmidt a

Formalmente, a tarefa de classificação pode ser descrita como a busca por uma função de mapeamento para um conjunto $X$ de vetores de entrada (ou, exemplares — os dados) $\vec{x_i} \in E^d$ para um conjunto finito de rótulos $C$ de cardinalidade $c$. A função $F$ é, então, definida como $F : E^d \times W \rightarrow C$, em que $d$ é a dimensão do espaço $E$, ou seja, a quantidade de coordenadas do vetor $\vec{x_i}$, e $W$ é um espaço de parâmetros ajustáveis por meio do algoritmo de indução supervisionada. [Silva et al., 2017]

Pode ser dividida em, ao menos, duas categorias: classificação binária e classificação multiclasse. Na binária, a cardinalidade $c$ é 2. Para o caso em que $c > 2$, o problema é considerado de múltiplas classes.[Silva et al., 2017]

Os textos de [Kesavaraj and Sukumaran, 2013], [Shazmeen et al., 2013], [Wolpert, 1996], [Kumar, 2012] e [Al-Radaideh and Nagi, 2012] trazem reflexões, técnicas, comparações e

explicações detalhadas de muitos algoritmos de classificação, entre eles, árvores de decisão, k-vizinhos mais próximos, Naive Bayes e Redes Bayesianas, Redes Neurais Artificiais, Máquinas de Vetores de Suporte (SVM) entre outros.

Sobre teoria da aprendizagem e algoritmos de classificação há em [Russell and Norvig, 2013] uma discussão sobre qual seria, em relação às hipóteses de modelos de aprendizagem, aquela (ou aquelas) que melhor se ajuste aos dados futuros. O autor cita a **suposição de estacionaridade**, ou seja, que há uma distribuição de probabilidade sobre os dados que permanece estacionária ao longo do tempo. Supõe-se, portanto que cada exemplo de ponto de dados (antes de conhecê-lo) é uma variável aleatória $E_j$ cujo valor observado $e_j = (x_j, y_j)$ é amostrado da distribuição e é independente dos exemplos anteriores. Assim:

$$P(E_j|E_{j-1}, E_{j-2}, ...) = P(E_j), \tag{2.2}$$

e cada exemplo tem uma distribuição de probabilidade anterior idêntica:

$$P(E_j) = P(E_{j-1}) = P(E_{j-2}) = ... \tag{2.3}$$

Estes exemplos são chamados de *independentes e identicamente distribuídos* ou **i.i.d**. Esta suposição é necessária para tentar a previsão sobre o futuro dos dados. Há claro, ainda em [Russell and Norvig, 2013], um alerta sobre o fato de ser possível a aprendizagem ocorrer caso haja pequenas alterações (lentas) na distribuição.

Outro fato importante para a definição e avaliação da escolha da melhor hipótese (modelo) de um algoritmo de classificação é definir o "melhor ajuste". [Russell and Norvig, 2013] define a **taxa de erro** de uma hipótese como uma métrica importante para definir o "melhor ajuste"de um modelo/hipótese.

A taxa de erro é, assim, a proporção de erros que o algoritmo classificador comete—a proporção de vezes que $h(x) \neq y$ para o exemplo $(x, y)$ — sendo $h(x)$ a função que mapeia uma hipótese/modelo $h$ com a previsão/valor conhecido $y$. Nem sempre, como alerta, [Russell and Norvig, 2013], uma hipótese/modelo(algoritmo) $h$ que tenha uma taxa de erro baixa no conjunto de treinamento generaliza bem. A forma de testar o algoritmo é importante. Para isso há, na literatura, algumas técnicas que são utilizadas como estratégia de treinamento, validação e teste.

Autores como [Silva et al., 2017], [Amaral, 2016], [Grus and Nascimento, 2016] e[Ferrari and Cas citam, como estratégia de treinamento, validação e teste as seguintes técnicas:

- Resubstituição;

- Holdout;
- Validação cruzada;
- Bootstrap;

Na Resubstituição, segundo [Silva et al., 2017], as medidas de avaliação dos classificadores são aplicadas no próprio conjunto de dados usados para indução do modelo. Essa técnica, embora tenha alguns vantagens discutidas em [Ferrari and Castro Silva, 2017] e [Silva et al., 2017], pode levar ao sobreajuste (*overfitting*) discutido em [Grus and Nascimento, 2016], [Amaral, 2016] e [Russell and Norvig, 2013]. Basicamente, o sobreajuste é quando se produz um modelo de bom desempenho com os dados de treinamento, mas que não lida bem com novos dados.

Na técnica de **Holdout**, pressupõem-se uma divisão, ou criação de dois subconjuntos de dados distintos, a partir do conjunto de dados disponível pra uso na indução do modelo/hipótese. Um desses subconjuntos será usado para treinamento (indução) do modelo de previsão e o segundo, para teste após o término do treinamento e, consequentemente, na aplicação das medidas de avaliação do modelo/hipótese.[Silva et al., 2017]

A imagem 2.2 mostra o funcionamento da técnica de holdout de forma mais detalhada

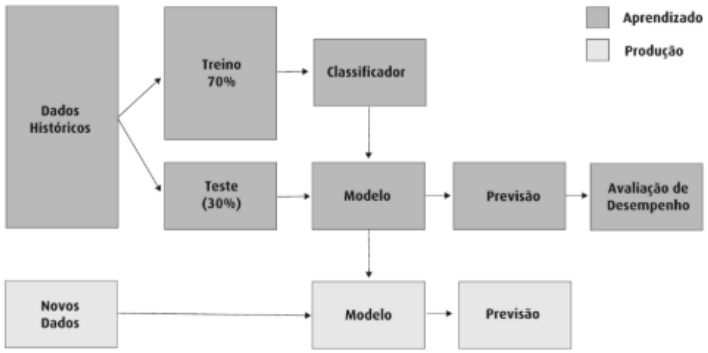



Figura 2.2: Funcionamento da técnica holdout.
Fonte:[Amaral, 2016]

Na estratégia de validação cruzada, todos os dados farão parte, em algum momento, do conjunto de dados usado no teste do modelo/hipótese. A ideia é que cada exemplo sirva duplamente — como dados de treinamento e dados de teste. Primeiro divide-se o conjunto em $k$ subconjuntos iguais. Em seguida realiza-se $k$ rodadas de aprendizagem; em cada iteração $\frac{1}{k}$ dos dados é retido como conjunto de teste e os exemplos restantes são usados como

treinamento. Valores populares de $k$ são 5 e 10 — o suficiente para uma estimativa estatisticamente provável que seja precisa a um custo 5-10 vezes maior no tempo de computação. Há também o extremo do $k = n$, também conhecido como **validação cruzada com omissão de um**. O método de validação cruzada permite que o modelo/hipótese seja avaliado uma série de vezes, cada série sendo conhecida como partição (ou *fold*). Ao final, a avaliação pode ser realizada aplicando medidas estatísticas como média, desvio-padrão e intervalo de confiança ao conjunto de $k$ avaliações obtidas ou somando-se os desempenhos obtidos pelos $k$ modelos gerados e dividindo essa soma pelo número de exemplares original. [Russell and Norvig, 2013][Silva et al., 2017][Ferrari and Castro Silva, 2017] [Amaral, 2016]

A imagem 2.3 mostra um exemplo didático de como funciona a validação cruzada:


Figura 2.3: Funcionamento da técnica *cross validation*
Fonte:[Gufosowa, 2019]

Já a técnica de *Bootstrap* funciona de forma parecida à estratégia *holdout*. Ela também usa dois conjuntos, um de treinamento e outro para teste, porém durante o processo de formação dos subconjuntos, exemplares que já foram sorteados podem novamente serem contemplados, com probabilidade igual. É uma estratégia que permite, portanto, a reposição.

Neste trabalho, todos os algoritmos de classificação usados foram testados usando as técnicas de resubstituição, *holdout* (com taxas de 70-30 e 60-40), além de *cross-validation* com 3, 5 e 10 folds. Como explicado em [Wolpert and Macready, 1995] e [Wolpert, 1996] não existe um algoritmo de aprendizado superior a todos os demais quando considerados todos os problemas de classificação possíveis (teorema **NFL**, ou *No Free Lunch*), portanto, variações foram executadas nos experimentos em todas técnicas avaliadas, alterando-se os padrões para chegar a métricas e medidas de avaliação mais eficientes.

Há, conforme [Silva et al., 2017], [Amaral, 2016], [Kesavaraj and Sukumaran, 2013] e [Kumar, 2012] diversas medidas usadas na avaliação de classificadores. Uma delas (a que

será usada neste trabalho) é a acurácia ou taxa de classificações corretas. A acurácia é dada, portanto, por:

$$\text{Acurácia} = |y - f(\aleph) = 0|, \tag{2.4}$$

em que $|\cdot|$ representa a contagem de vezes em que $\cdot$ é verdadeiro, $f$ é o modelo preditivo, $\aleph$ é o subconjunto de dados sob o qual o modelo está sendo avaliado, $f(\cdot)$ é a classificação fornecida pelo modelo preditivo para cada um dos exemplares (dos dados), e $y$ é a classe esperada como resposta. [Silva et al., 2017]

A acurácia de um classificador também pode ser descrita em termos do **erro de generalização** $\xi_g$, e uma função de perda binária e, portanto, ser interpretada como a probabilidade de ocorrer uma classificação correta. Dessa forma:

$$\text{Acurácia}_g = 1 - \xi_g \tag{2.5}$$

Ou seja, a acurácia é, basicamente o número de acertos (positivos) divido pelo número total de exemplos. Será a métrica mais usada para avaliar os classificadores neste trabalho.

### 2.2.5 Redes Neurais artificiais

As redes neurais instituem um campo da ciência da computação, parte da área da inteligência artificial, que busca efetivar modelos matemáticos que se assemelhem às redes neurais biológicas. Elas apresentam capacidade de adaptar seus parâmetros como resultado da interação com o meio externo. [Ferneda, 2006][Russell and Norvig, 2013]

De acordo com [Lima et al., 2016, p. 47], "redes neurais podem ser caracterizadas como modelos computacionais com capacidades de adaptar, aprender, generalizar, agrupar ou organizar dados".

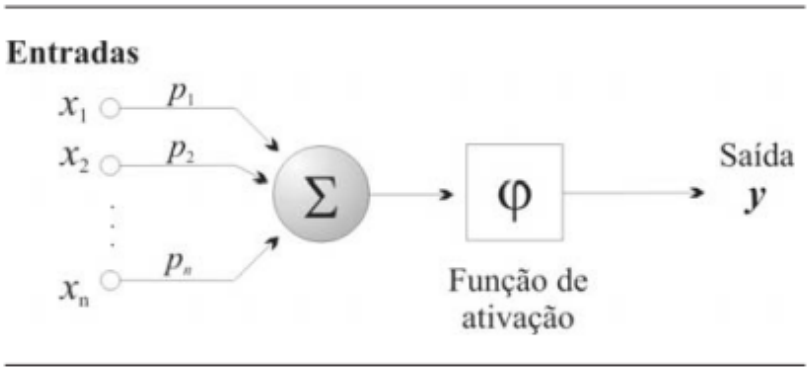
Inicialmente, portanto, se desenvolveram como uma estratégia de simular os processos mentais humanos, como reconhecimento de imagens e sons, e após, como instrumento tecnológico e eficienten para muitas tarefas. [Jin et al., 2002]

Para [Obaidat and Macchairolo, 1994], as redes neurais artificiais podem ser usadas efetivamente para prover soluções para um amplo espectro de aplicações, incluindo mapeamento de padrões e classificação, análise e codificação de imagens, processamento de sinais, otimização, manipulação de grafos, reconhecimento de caracteres, reconheci-

mento automático de alvo, fusão de dados, processamento de conhecimento, controle de qualidade, mercado de ações, processamento de hipotecas, triagem de créditos para empréstimos entre muitos outros problemas.

Desde a década de 1940 com o trabalho de [Mcculloch and Pitts, 1943] que se busca um modelo computacional que simule o cérebro humano e suas conexões. O interesse pela pesquisa nesta área cresceu e se desenvolveu durante os anos 50 e 60. É dessa época que [Rosenblatt, 1958] sugeriu um método de aprendizagem para as redes neurais artificiais chamado *percepton*.

Até o final da década de 1960 muitos trabalhos foram feitos usando o percepton como modelo, mas ao final desta década, [Minsky and Papert, 1969] apresentaram significativas limitações do perceptron. A pesquisa diminui consideravelmente nos anos seguintes, porém durante os anos 80, a excitação ressurge mediante os avanços metodológicos importantes e, também, ao aumento dos recursos computacionais disponíveis. O modelo de neurônio artificial da figura 2.4 é uma simplificação do apresentado por [Haykin, 2001, p. 36]




Fonte:[Haykin, 2001, p. 36]

Figura 2.4: Modelo matemático de um neurônio

Este modelo acima (da figura 2.4) é composto por três elementos:

- um conjunto de $n$ conexões de entrada $(x_1, x_2, ..., x_n)$, caracterizadas por pesos $(p_1, p_2, ..., p_n)$;
- um somador $(\sum)$ para acumular os sinais de entrada;
- uma função de ativação $(\varphi)$ que, no caso do neurônio de McCullock-Pitts [Mcculloch and Pitts, 1
é uma função de limiar. [Ferneda, 2006] [Lima et al., 2016]

O comportamento das conexões entre os neurônios é simulado através de seus pesos $(p_1, p_2, ..., p_n)$. Os valores podem ser positivos ou negativos (dependendo se a conexão é inibitiva ou excitativa. O efeito de um sinal proveniente de um neurônio é determinado pela multiplicação do valor do sinal recebido pelo peso da conexão correspondente $(x_i \times p_i)$. Então é efetuada a soma dos valores $x_i \times p_i$ de todas as conexões e o valor resultante é enviado para a função de ativação que define a saída $(y)$ do neurônio.[Russell and Norvig, 2013][Mcculloch and Pitts, 1943][Minsky and Papert, 1969][Ferneda,

As redes neurais artificiais (**RNA**) se formam quando diversos neurônios se combinam. De forma resumida, "uma rede neural artificial (RNA) pode ser vista como um grafo onde os nós são os neurônios e as ligações fazem a função das sinapses". Isto está demonstrado na figura 2.5

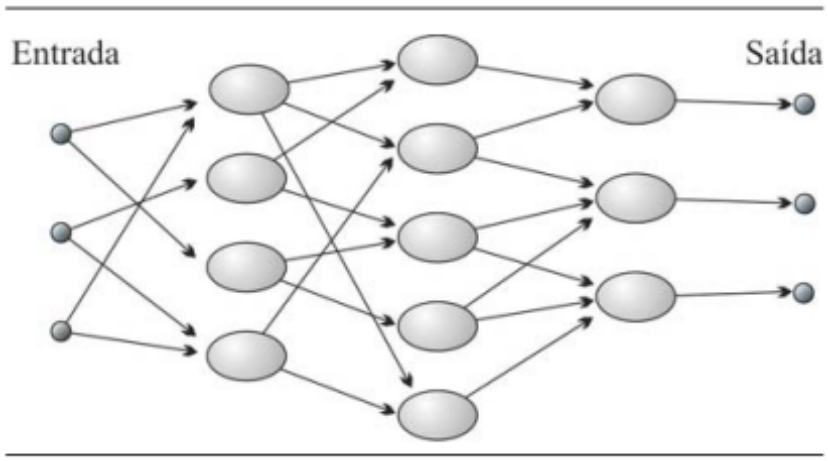



Figura 2.5: Representação simplificada de uma RNA

Fonte: [Ferneda, 2006, p.26]

As redes neurais artificiais se diferem pelas suas arquiteturas e pela forma como os pesos associados às conexões são ajustados durante o processo de aprendizado. A arquitetura de uma rede neural restringe o tipo de problema no qual a rede poderá ser utilizada, e é definida pelo número de camadas (camada única ou múltiplas camadas), pelo número de nós em cada camada, pelo tipo de conexão entre os nós (*feedforward* ou *feedback*) e por sua topologia. [Haykin, 2001, p. 46-49]

O desenvolvimento de uma rede neural artificial consiste em determinar sua arquitetura, ou seja, os números de camadas e de neurônios em cada camada, bem como o ajuste dos pesos na fase conhecida como treinamento.[Hagan et al., ] [Haykin, 2001]

Uma das características mais importantes de uma rede neural artificial é a habilidade de aprender através de exemplos e fazer inferências sobre o que aprendeu, melho-

rando, assim, o seu desempenho. As RNA's utilizam um algoritmo de aprendizagem que serve, basicamente, para ajustar os pesos de suas conexões. [Haykin, 2001] [Ferneda, 2006] [Lima et al., 2016] [Russell and Norvig, 2013]. Aqui também há, cf. explicitado na seção 2.2.3, duas formas básicas de aprendizado, o supervisionado e o não-supervisionado.

### 2.2.6 SVM - Support Vector Machines

Segundo [Cortes and Vapnik, 1995], o algoritmo SVM (*Support Vector Machines*) é um dos mais efetivos para a tarefa de classificação.

Cf. [Goldschmidt and Passos, ],

> No algoritmo SVM, o conjunto de dados de entrada é utilizado para construir uma *função de decisão* $f(x)$, tal que:
>
> $$\begin{array}{lll} Se\ f(x_i) \geq 0, & \text{então} & y_i = 1 \\ Se\ f(x_i) < 0, & \text{então} & y_i = -1 \end{array}$$
>
> O algoritmo SVM constrói os denominados classificadores lineares, que separam o conjunto de dados por meio de um hiperplano que é a generalização do conceito de *plano* para dimensões maiores que três.

Assim, SVM, cf. [Amaral, 2016, p. 45] "são um algoritmo de classificação que maximizam as margens entre instâncias mais próximas, dessa forma, é criado um vetor otimizado que é então utilizado para classificar novas instâncias".

Conforme se vê na figura 2.6, os dois vetores *não pontilhados* são as margens otimizadas. As instâncias por onde as margens otimizadas passam são os vetores de suporte. O vetor pontilhado é a referência para classificar novas instâncias. Assim, a nova instância, na figura 2.6 é classificada como triângulo.

Seguindo o estudo de [Mukkamala et al., 2002] há duas razões principais que levaram os autores do artigo citado de usarem SVMs para detecção de intrusão:o primeiro é a velocidade já que a performance é prioritariamente uma das características mais importantes para sistemas de detecção de intrusos. A segunda razão é a escalabilidade, pois, cf. os autores, SVMs são relativamente indiferentes ao número de *data points* e a complexidade da classificação não depende da dimensionalidade do espaço de características. Dependendo da aplicação, ainda conforme os autores, uma vez que os dados estão classificados em duas classes, um algoritmo de otimização adequado pode ser usado, se necessário, para identificação de mais características.

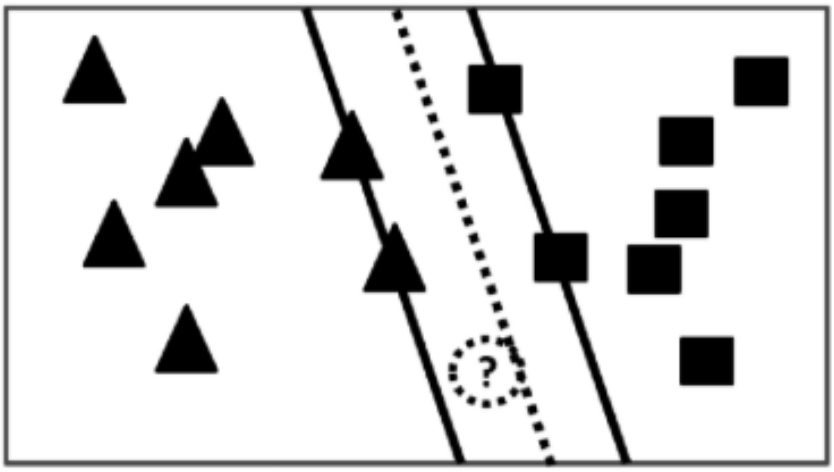



Figura 2.6: Vetores de Suporte

Fonte: [Amaral, 2016, p. 45]

Neste trabalho, também foi usado, com boa eficácia (cf. se vê na seção 3) o algoritmo SVM.

# Capítulo 3

# Experimentos/Resultados

Para os experimentos, um arquivo de políticas foi gerado a partir do proposto em [Sarkis, 2017] e, de acordo com o exposto na seção 2.1.3. O arquivo gerado possui cerca de 68 políticas nomeadas (constituindo a *fase de seleção*[1] da Mineração de Dados).

Este arquivo foi usado nos testes preliminares da hipótese: *Converter a detecção de conflitos a um problema de classificação reestruturando os atributos* (colunas). Para este problema da detecção de conflitos diretos serão usadas técnicas de aprendizagem supervisionada. Para tanto, ao arquivo com as políticas, no pré-processamento foi acrescentada uma coluna rotulando os conflitos da seguinte forma: **1**: *conflito direto* e **0**: *sem conflito*.

A figura 3.1 demonstra o aspecto do arquivo das políticas geradas paraos experimentos deste trabalho. Na imagem, pode-se notar a classe (coluna) criada para guiar o aprendizado supervisionado dos algoritmos utilizados no estudo.

Dois **ambientes computacionais** foram utilizados para as tarefas de mineração: um **notebook** Intel Core i5 vPro-8350U (8ª Geração de 64 bits com 1.70GHz e 8 GB de RAM, com SSD de 256 GB rodando Windows 10 Pro. O outro ambiente foi um **Desktop** Intel Core i7 vPro-6700 de 8ª geração de 64 bits com 3.40 Ghz e 20 GB de RAM, com HD de 1 TB rodando o Windows 10 Pro.

Ainda na fase de *pré-processamento*, a coluna 9 (Conflito) foi transformada do tipo de dado *Numérico para Nominal*. Para isso foi usada o softwate WEKA (descrito em [Witten et al., 2016]) aplicado o filtro *NumericToNominal* do software.

Logo após, mais de 30 experimentos foram realizados de forma preliminar no *dataset* envolvendo os diversos algoritmos e muitos parâmetros alterados (a maioria com pequena

[1] cf. seção 2.2.2 deste trabalho.


Figura 3.1: Aspecto do arquivo das políticas geradas para os experimentos

Fonte: compilação do autor

ou nenhuma variação) para se chegar às técnicas finais que foram utilizadas nos posteriores experimentos e que serão explicitadas a seguir.

Utilizando-se a ferramenta WEKA ([Witten et al., 2016]) para as últimas fases do KDD (Mineração de Dados), foram utilizados alguns algoritmos de classificação que segundo [Wu et al., 2007] são alguns dos mais utilizados na Mineração de Dados. Para avaliar o desempenho definiu-se o método cross-validation com 10 folds. Em seguida suas acurácias foram comparadas.

A tabela 3.1 mostra o resultado destes experimentos:

Tabela 3.1: Acurácia dos classificadores

| Classificador/Algoritmo | Acurácia |
|---|---|
| Multi Layer Perceptron | 0.9705 |
| Random Forest | 0.9558 |
| J48 | 0.9411 |
| K* (K-star) | 0.9411 |
| Trees LMT | 0.9117 |
| IBk (KNN, com k =1) | 0.8970 |
| JRip | 0.8970 |
| SVM kernel linear | 0.8676 |
| Nayve Bayes | 0.8674 |
| Random Tree | 0.7794 |

Fonte: Elaborada pelo autor mediante experimentos

As figuras 3.2 e 3.3 mostram os resultados das classificações do arquivo de políticas usando, respectivamente, os classificadores/algoritmos: *SVM* e o *MultiLayer Perceptron*

(que foram os principais citados nos trabalhos relacionados, cf. descrito na seção 2.2.1). Importante ressaltar que outros classificadores, como o Random Forest, o J48, o K* e o KNN tiveram resultados superiores (em termos de acurácia e precisão) ao SVM, cf. mostrado na tabela 3.1. Entretanto, no referncial teórico, o SVM foi citado diversas vezes na detecção de alguns tipos de conflitos e em outras tarefas de classificação de diversos conjuntos de dados.

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          59               86.7647 %
Incorrectly Classified Instances         9               13.2353 %
Kappa statistic                          0.7012
Mean absolute error                      0.1324
Root mean squared error                  0.3638
Relative absolute error                 30.0527 %
Root relative squared error             77.6279 %
Total Number of Instances               68

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,891    0,182    0,911      0,891    0,901      0,702    0,855     0,886     0
                 0,818    0,109    0,783      0,818    0,800      0,702    0,855     0,699     1
Weighted Avg.    0,868    0,158    0,870      0,868    0,868      0,702    0,855     0,825

=== Confusion Matrix ===

  a  b   <-- classified as
 41  5 |  a = 0
  4 18 |  b = 1
```

Figura 3.2: Saída do software WEKA. Classificador: SVM

Fonte: compilação do autor

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          66               97.0588 %
Incorrectly Classified Instances         2                2.9412 %
Kappa statistic                          0.9344
Mean absolute error                      0.0574
Root mean squared error                  0.1664
Relative absolute error                 13.029  %
Root relative squared error             35.5106 %
Total Number of Instances               68

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,957    0,000    1,000      0,957    0,978      0,936    0,996     0,998     0
                 1,000    0,043    0,917      1,000    0,957      0,936    0,996     0,992     1
Weighted Avg.    0,971    0,014    0,973      0,971    0,971      0,936    0,996     0,996

=== Confusion Matrix ===

  a  b   <-- classified as
 44  2 |  a = 0
  0 22 |  b = 1
```

Figura 3.3: Saída do software WEKA. Classificador: *MultiLayer Perceptron*

Fonte: compilação do autor

Assim, com uma acurácia de 97,05% na classificação dos conflitos diretos, o algoritmo Multilayer Perceptron (que implementa uma rede neural sigmoide multicamadas) foi o que teve a maior acurácia, com 95,7% de *TP rate*(taxa de *True Positives* ou verdadeiros positivos) para a classe 0 (não há conflito) e, somente, 4,3% de *FP rate*(taxa de Falsos

Positivos) para a classe 1 (quando há conflito direto). Nos experimentos realizados (assim como se esperava inicialmente na hipótese deste trabalho — baseado em evidências da literatura), este modelo algorítmico foi o mais eficiente para a detecção de conflitos diretos.

# Capítulo 4

# Propostas para a dissertação

Para a pesquisa que resultará na dissertação de mestrado os seguintes pontos serão levantados, estudados e melhor definidos em termos dos objetivos do trabalho:

- Pesquisar sobre o relacionamento entre entidades, ações e definições sobre políticas (para entendimento da propagação de políticas);
- Construção (teoria) e Programação (prática) do Perceptron (com *backpropagation* para ajuste de pesos e atributos — treinamento da rede neural);
- Análise da função de ativação no classificador;
- Análise teórica e construção da Função soma (e funções sigmóides de ativação do perceptron);
- Análise teórica e construção das múltiplas layers do perceptron;
- Usar Reinforcement Learning e Deep Learning para a detecção dos conflitos indiretos;
- Comparação com outros classificadores (preferencialmente, geométricos, como o KNN e o SVM, avaliando suas acurácias e eficiência.

# Capítulo 5

# Conclusões

- Esta pesquisa mostrou que é possível *converter a detecção de conflitos a um problema de classificação* conforme demonstrado neste trabalho, especificamente, para os conflitos ***diretos***;
- O classificador mais acurado, nos experimentos, foi, como se imaginava pela hipótese, o *MultiLayer Perceptron* que é um classificador que usa *backpropagation* para aprender usando perceptron de várias camadas para classificar instâncias desconhecidas [Witten et al., 2016];
- Este será um dos classificadores usados para detectar conflitos indiretos. O outro será o SVM (e outros classificadores geométricos). Suas acurácias serão devidamente comparadas juntamente com a eficiência das soluções propostas.

# Referências


[Al-Radaideh and Nagi, 2012] Al-Radaideh, Q. A. and Nagi, E. A. (2012). Using Data Mining Techniques to Build a Classification Model for Predicting Employees Performance.

[Alecrim, 2019] Alecrim, E. (2019). *O que é Big Data?* INFOWESTER. Disponível em: https://www.infowester.com/big-data.php. Acesso em 12/12/2019.

[Alpaydin, 2014] Alpaydin, E. (2014). *Introduction to Machine Learning, Third Edition*. Adaptive Computation and Machine Learning. The MIT Press, 3ª edition.

[Amaral, 2016] Amaral, F. (2016). *Aprenda Mineração de Dados: Teoria e prática*. Autoria Nacional. ALTA BOOKS.

[Bui et al., 2019] Bui, T., Stoller, S. D., and Le, H. (2019). Efficient and Extensible Policy Mining for Relationship-Based Access Control. In *Proceedings of the 24th ACM Symposium on Access Control Models and Technologies*, SACMAT '19, pages 161–172, New York, NY, USA. ACM. event-place: Toronto ON, Canada.

[Chen, 2011] Chen, M. Q. (2011). Flight Conflict Detection and Resolution Based on Neural Network. In *2011 International Conference on Computational and Information Sciences*, pages 860–862. ISSN: null.

[Christodoulou and Kontogeorgou, 2008] Christodoulou, M. A. and Kontogeorgou, C. (2008). Collision avoidance in commercial aircraft Free Flight via neural networks and non-linear programming. *International Journal of Neural Systems*, 18(5):371–387.

[Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). Support-vector networks. 20(3):273–297.

[De Capitani di Vimercati et al., 2005] De Capitani di Vimercati, S., Samarati, P., and Jajodia, S. (2005). Policies, models, and languages for access control. In Bhalla, S., editor, *Databases in Networked Information Systems*, Lecture Notes in Computer Science, pages 225–237. Springer.

[Debar et al., 1992] Debar, H., Becker, M., and Siboni, D. (1992). A neural network component for an intrusion detection system. In *Proceedings 1992 IEEE Computer Society Symposium on Research in Security and Privacy*, pages 240–250. ISSN: null.

[Dunlop et al., 2002] Dunlop, N., Indulska, J., and Raymond, K. (2002). Dynamic conflict detection in policy-based management systems. In *Proceedings. Sixth International Enterprise Distributed Object Computing*, pages 15–26. ISSN: null.

[Fayyad et al., 1996] Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. (1996). From data mining to knowledge discovery in databases. 17(3):37–37.

[Ferneda, 2006] Ferneda, E. (2006). Redes neurais e sua aplicação em sistemas de recuperação de informação. 35(1).

[Ferrari and Castro Silva, 2017] Ferrari, D. G. and Castro Silva, L. N. d. (2017). *Introdução a mineração de dados*. Editora Saraiva.

[Fugini and Bellettini, 2004] Fugini, M. and Bellettini, C. (2004). *Information Security Policies and Actions in Modern Integrated Systems*. Idea Group Pub.

[Goldschmidt and Passos, ] Goldschmidt, R. and Passos, E. *Data mining: um guia Prático*. Elsevier Editora.

[Grus and Nascimento, 2016] Grus, J. and Nascimento, W. (2016). *Data Science Do Zero*. ALTA BOOKS.

[Gufosowa, 2019] Gufosowa (2019). *K-fold cross validation*. Wikipedia. Disponível em: https://en.wikipedia.org/wiki/File:K-fold_cross_validation_EN.svg.

[Hagan et al., ] Hagan, M. T., Demuth, H. B., and Beale, M. H. *Neural Network Design*. Brooks/Cole. Google-Books-ID: cUNJAAAACAAJ.

[Haykin, 2001] Haykin, S. (2001). *Redes Neurais - 2ed.* Bookman.

[Jin et al., 2002] Jin, X., Cheu, R. L., and Srinivasan, D. (2002). Development and adaptation of constructive probabilistic neural network in freeway incident detection. *Transportation Research Part C: Emerging Technologies*, 10(2):121–147.

[Kesavaraj and Sukumaran, 2013] Kesavaraj, G. and Sukumaran, S. (2013). A study on classification techniques in data mining. In *2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, pages 1–7. ISSN: null.

[Koch et al., 2002] Koch, M., Mancini, L. V., and Parisi-Presicce, F. (2002). Conflict Detection and Resolution in Access Control Policy Specifications. In Nielsen, M. and Engberg, U., editors, *Foundations of Software Science and Computation Structures*, Lecture Notes in Computer Science, pages 223–238, Berlin, Heidelberg. Springer.

[Kumar, 2012] Kumar, R. (2012). Classification Algorithms for Data Mining: A Survey. *International Journal of Innovations in Engineering and Technology*, 1(2):8.

[Li and Tripunitara, 2006] Li, N. and Tripunitara, M. V. (2006). Security analysis in role-based access control. page 28.

[Lima et al., 2016] Lima, I., Pinheiro, C., and Santos, F. (2016). *Inteligência Artificial*. Elsevier Brasil.

[Lima and Pereira, 2012] Lima, R. A. F. and Pereira, A. C. M. (2012). Fraud detection in web transactions. page 273.

[Lupu and Sloman, 1999] Lupu, E. and Sloman, M. (1999). Conflicts in policy-based distributed systems management. *IEEE Transactions on Software Engineering*, 25(6):852–869.

[Machado, 2014] Machado, A. (2014). Estudo da EMC prevê que volume de dados virtuais armazenados será seis vezes maior em 2020.

[Mcculloch and Pitts, 1943] Mcculloch, W. S. and Pitts, W. (1943). A LOGICAL CALCULUS OF THE IDEAS IMMANENT IN NERVOUS ACTIVITY. 52(1):17.

[Minsky and Papert, 1969] Minsky, M. and Papert, S. (1969). *Perceptrons: An Introduction to Computational Geometry*. MIT Press.

[Moffett and Sloman, 1994] Moffett, J. D. and Sloman, M. S. (1994). Policy conflict analysis in distributed system management. 4(1):1–22.

[Mohri, 2018] Mohri, Mehryar e Rostamizadeh, A. e. T. A. (2018). *Foundations of Machine Learning, Second Edition*. Adaptive Computation and Machine Learning. The MIT Press, 2ª edition.

[Mukkamala et al., 2002] Mukkamala, S., Janoski, G., and Sung, A. (2002). Intrusion detection using neural networks and support vector machines. In *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No.02CH37290)*, volume 2, pages 1702–1707 vol.2. ISSN: 1098-7576.

[Neri et al., 2012] Neri, M. A., Guarnieri, M., Magri, E., Mutti, S., and Paraboschi, S. (2012). Conflict detection in security policies using Semantic Web technology. In *2012 IEEE First AESS European Conference on Satellite Telecommunications (ES)*, pages 1–6. ISSN: 2375-8554.

[Obaidat and Macchairolo, 1994] Obaidat, M. and Macchairolo, D. (1994). A multilayer neural network system for computer access security. 24(5):806–813.

[Rocha et al., 2012] Rocha, T., Peres, S. M., Bíscaro, H. H., Madeo, R. C. B., and Boscarioli, C. (2012). *Tutorial sobre Fuzzy-c-Means e Fuzzy Learning Vector Quantization: abordagens híbridas para tarefas de agrupamento e classificação*.

[Rosenblatt, 1958] Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. 65(6):386–408.

[Russell and Norvig, 2013] Russell, S. and Norvig, P. (2013). *Inteligêngia Artificial - Tradução da3ª Edição*. Elsevier, Rio de Janeiro, Brasil.

[Sandhu and Samarati, 1996] Sandhu, R. and Samarati, P. (1996). Authentication, access control, and audit. *ACM Comput. Surv.*, 28(1):241–243.

[Sarkar et al., 2017] Sarkar, D., Bali, R., and Sharma, T. (2017). *Practical Machine Learning with Python: A Problem-Solver's Guide to Building Real-World Intelligent Systems*. Apress.

[Sarkis, 2017] Sarkis, L. C. (2017). Uma abordagem para detecção de conflitos indiretos entre políticas de controle de acesso.

[Sarkis et al., 2016] Sarkis, L. C., da Silva, V. T., and Braga, C. (2016). Detecting indirect conflicts between access control policies. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, SAC '16, pages 1570–1572, New York, NY, USA. ACM.

[Shazmeen et al., 2013] Shazmeen, S. F., Mustafa, M. A., and Baig, A. (2013). Performance Evaluation of Different Data Mining Classification Algorithm and Predictive Analysis.

[Shoham and Tennenholtz, 1995] Shoham, Y. and Tennenholtz, M. (1995). On social laws for artificial agent societies: off-line design. 73(1):231–252.

[Silva et al., 2017] Silva, L. A. d., Peres, S. M., and Boscarioli, C. (2017). *Introdução à Mineração de Dados: Com Aplicações em R.* Elsevier Brasil.

[Silva, 2019] Silva, L. C. e. (2019). Aprendizado de máquina com treinamento continuado aplicado à previsão de demanda de curto prazo: o caso do restaurante universitário da universidade federal de uberlandia.

[Silvestre, 2017] Silvestre, E. A. (2017). Verificação de conflitos entre múltiplas normas em sistemas multiagentes.

[Simon, 2013] Simon, P. (2013). *Too Big to Ignore: The Business Case for Big Data.* Wiley and SAS Business Series. Wiley.

[Swamynathan, 2019] Swamynathan, M. (2019). *Mastering Machine Learning with Python in Six Steps: A Practical Implementation Guide to Predictive Data Analytics Using Python.* Apress, 2ª edition.

[Thenmozhi et al., 2018] Thenmozhi, D., Kalaivani, A., and Aravindan, C. (2018). Multilingual author profiling on SMS messages using machine learning approach with statistical feature selection. page 9.

[Ueda, 2012] Ueda, E. T. (2012). Análise de políticas de controle de acesso baseado em papéis com rede de petri colorida.

[Wang et al., 2010] Wang, Y., Zhang, H., Dai, X., and Liu, J. (2010). Conflicts analysis and resolution for access control policies. In *2010 IEEE International Conference on Information Theory and Information Security*, pages 264–267. ISSN: null.

[Wiederhold and McCarthy, 1992] Wiederhold, G. and McCarthy, J. (1992). Arthur samuel: Pioneer in machine learning. 36(3):329–331.

[Witten et al., 2016] Witten, I., Eibe, F., Hall, M., and Pal, C. (2016). *Data Mining: Practical Machine Learning Tools and Techniques.* The Morgan Kaufmann Series in Data Management Systems. Elsevier Science.

[Wolpert, 1996] Wolpert, D. H. (1996). The lack of a priori distinctions between learning algorithms. *Neural Comput.*, 8(7):1341–1390.

[Wolpert and Macready, 1995] Wolpert, D. H. and Macready, W. G. (1995). *No Free Lunch Theorems for Search.* Santa Fe Institute.

[Wu et al., 2007] Wu, X., Kumar, V., Ross Quinlan, J., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G. J., Ng, A., Liu, B., Yu, P. S., Zhou, Z.-H., Steinbach, M., Hand, D. J., and Steinberg, D. (2007). Top 10 algorithms in data mining. *Knowl. Inf. Syst.*, 14(1):1–37.