Vasco Féria de Almeida Eva Ferreira

Licenciado em Engenharia Electrotécnica e de Computadores

Error and Fault detection IMS

Dissertação para obtenção do Grau de Mestre em

Engenharia Electrotécnica e de Computadores

Orientador: Pedro Miguel Figueiredo Amaral, Professor Assistente, Universidade NOVA de Lisboa

Júri

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Novembro, 2020

Error and Fault detection IMS

Este documento foi gerado utilizando o processador (pdf)LaTeX, com base no template "novathesis" [1] desenvolvido no Dep. Informática da FCT-NOVA [2].
[1] https://github.com/joaomlourenco/novathesis [2] http://www.di.fct.unl.pt

Agradecimentos

Queria em primeiro lugar agradecer ao meu orientador por toda a ajuda na revisão do documento e imprescindíveis conselhos no caminho a explorar. De seguida queria também agradecer a toda a equipa de supervisão da rede da operadora, que foram absolutamente essenciais em todo o processo de adaptação, aprendizagem e execução do trabalho desenvolvido.

Todo o trabalho desenvolvido teria sido impossível sem a ajuda e compreensão da minha família e amigos.

O apoio e espírito de entreajuda presente no meu grupo mais próximo de colegas e amigos de mestrado demonstrou-se para mim vital ao elaborar este documento. Trata-se de um grupo extremamente coeso e dedicado a ajudar os seus mais próximos que enriqueceu a minha experiência académica de uma forma extremamente marcante. Muito obrigado a todos

Por fim, devo à Margarida Cruz um enorme e impagável obrigado por todas as valiosas opiniões, ajuda, apoio inquestionável e conselhos de correcção imprescindíveis. O seu apoio e disponibilidade foram marcantes desde o início da minha vida académica, pelo qual lhe estarei para sempre grato.

Resumo

Nesta dissertação são estudados métodos de classificação de registos fraudulentos na rede IMS de uma operadora de telecomunicações, com o objectivo de detectar utilizações fraudulentas.

O registo de terminais e usurpação da password dos mesmos representa um problema de segurança para as operadoras, uma vez que os recursos dos clientes são violados e, o atacante pode usar a conta a que ganha acesso para cobrar à operadora serviços aos quais não tem direito. Para além das consequências de um ataque bem sucedido, as repetidas tentativas que um utilizador malicioso acaba por fazer para obter uma conta consomem recursos que a operadora poderia estar a alocar aos seus clientes pagos. Para evitar este processo foram desenvolvidos e testados diversos mecanismos de classificação de mensagens SIP com cabeçalho REGISTER.

Foram estudados mecanismos de classificação supervisionada e não supervisionada de forma a perceber qual o modelo que mais se adequa ao problema proposto.

Por fim, um dos modelos criados foi escolhido para implementação, tendo em consideração as necessidades da operadora.

O modelo escolhido é capaz de analisar os pedidos de registo com destino à rede da operadora durante um determinado período de tempo e, reportar casos de ataque à equipa de supervisão. Desta forma, a operadora é capaz de detectar os ataques no início de execução e bloquear as comunicações do atacante se assim o entender.

Palavras-chave: Inteligência Artificial, IMS, Cibersegurança, SIP

Abstract

In this dissertation, we will study a classification method for fraudulent SIP register traffic in a IMS network of a telecommunications operator.

The registration of an IMS terminal and usurpation of the account's password represents a security problem for telecom companies, since the resources of the real client are being violated and the malicious client might use the usurped account to charge the telecom company with services that he is not allowed to use.

Besides the consequences of a successful attack, the enormous amount of register requests sent by the malicious user to break into an account, are consuming network resources that could be used by paying clients.

In order to avoid this type of network problems, this dissertation aims to develop and test several classification mechanisms for SIP messages with a REGISTER header. Supervised and unsupervised machine learning algorithms were used as classification methods. The chosen algorithms were trained and tested in order to understand which one would be the best to satisfy the operator needs.

One of the trained algorithms was chosen to be implemented in a network scan application. The deployed mechanism is able to scan and classify all suspicious SIP register requests sent to the telecom company's IP addresses, for a given period of time. The classification result is sent to the network supervision team.

This artificial intelligence mechanism gives the telecom company the ability to detect and block malicious traffic in the early stages of an attack.

Keywords: Artificial intelligence, IMS, Cybersecurity, SIP

Índice

Lista de Figuras

Listagens

Capítulo 1

Introdução

1.1 Introdução

Neste capítulo será feita uma introdução à motivação por detrás da dissertação, assim como uma contextualização em relação ao problema a abordar e ao IP Multimedia System (IMS).
Esta dissertação tem como objectivo o estudo e desenvolvimento de métodos para a detecção de comportamentos fraudulentos em sistemas IMS. Os utilizadores fraudulentos serão detectados através da padronização de comportamentos do utilizador, tendo em conta as informações por este transmitidas a partir do protocolo de sinalização Session Initiation Protocol (SIP). A partir dos dados disponíveis, serão desenvolvidos algoritmos para detectar padrões de utilização fraudulenta.
O IMS é um sistema de controlo de serviços responsável por definir como os diversos serviços baseados em IP se interligam, comunicam e como se integram no sistema do operador. Assim sendo, o IMS é um sistema independente dos meios de acesso que, através das normas IP, consegue interligar vários serviços com conectividades distintas. A partir destas funcionalidades, o IMS oferece aos operadores uma plataforma que facilita a gestão de vários serviços [6].
O Voice-over-IP (VoIP) é uma das tecnologias de comunicação que o IMS é capaz de gerir. VoIP é um meio de comunicação IP com sinalização SIP e é, neste momento, o padrão para comunicação de voz ou video, substituindo os serviços clássicos de telefonia.
Através da tecnologia VoIP é possível estabelecer comunicações por voz, colocando o resultado da conversão do sinal analógico no payload de pacotes IP devidamente endereçados. Desta forma, o tráfego de voz passa a ser feito pelas normas e padrões IP.
O SIP é um dos protocolos utilizados nesta tecnologia e tem como função a sinalização entre as duas parte envolvidas na comunicação [38].

A conectividade IP, assim como a interconexão do SIP com a Public Switched Telephone Network (PSTN), deixa o IMS a mercê de ataques e fraudes em pontos mais frágeis como as VoIP gateways [19].
Segundo a Communications Fraud Control Association (CFCA), as fraudes por VoIP representaram uma perda de 28 mil milhões de dólares em todo o mundo no ano de 2018 [8]. As fraudes podem ter como objectivo Denial of Service (DoS) ou intuitos financeiramente lucrativos para o atacante. Nesta dissertação vai haver um maior ênfase num determinado tipo de fraude denominado Toll Fraud.
Toll Fraud é um esquema fraudulento com intuitos lucrativos do atacante. Através deste esquema o atacante é capaz de simular um terminal em software e autenticá-lo no IMS de um determinado operador. Assim que o atacante simula o terminal, este passa a ser capaz de fazer todo o tipo de chamadas, utilizando esse factor para que a taxação dessas chamadas o beneficie financeiramente.

1.2 Problema

Como foi explicado anteriormente, o problema a ser analisado nesta dissertação representa uma perda financeira bastante pesada para as operadoras de telecomunicações de todo o mundo. Qualquer operadora com serviços de telecomunicações depende do IMS como plataforma capaz de interligar todos os seus serviços com conectividades destintas através de normas IP.

O VoIP é um dos recursos que o IMS é capaz de integrar com a rede de telefone PSTN através de normas IP. As operadoras recorrem à taxação dos serviços telefónicos como fonte de rendimento quando as receitas desta operação não sofrem nenhum desvio por um utilizador fraudulento.
Sempre que um utilizador estabelece uma chamada, o terminal que a executa é taxado em benefício da operadora que formula a chamada e em benefício de outra operadora, se a chamada não se limitar à rede de origem.
A partir do momento em que há uma segunda entidade com receitas referentes à chamada, pode haver um comportamento fraudulento com o intuito de beneficiar essa entidade.
Através da rede telefónica há várias formas de custos extra para as operadoras, como números inteligentes ou despesas de roaming que, em situações normais, são pagas pela operadora e posteriormente cobradas ao utilizador. A partir do momento em que as entidades que cobram estes custos extra à operadora de origem estão associadas a um utilizador, há a possibilidade de fraude em favorecimento dessa entidade, prejudicando a operadora.

1.2.1 IMS da Operadora

O IMS é capaz de trabalhar com todo o tipo de acessos baseados na troca de pacotes. Esta flexibilidade é conseguida através da separação dos meios de acesso, transporte e controlo.

Por sua vez, o controlo está também separado em entidades destintas responsáveis por controlo de media, sessão e de aplicações [31]. O controlo de sessão é da responsabilidade do componente Common Session Control Function (CSCF) cuja função principal é a sinalização de rotas através de SIP. O CSCF é também responsável pelo registo, routing e roaming, assim como pela qualidade de serviço para cada comunicação. Para desempenhar todas estas funções o CSCF divide-se em 3 componentes: Proxy-CSCF (P- CSCF), Interrogating-CSCF (I-CSCF) e Serving-CSCF(S-CSCF).
O P-CSCF é, como o próprio nome implica, um proxy na rede de origem ou na rede visitada que representa o terminal nas operações do IMS. É o P-CSCF que regista a existência do terminal na sua rede de origem assim como os seus pedidos através de mensagens SIP enviadas para o S-CSCF.
Assim que um terminal representado pelo P-CSCF se quer registar para usufruir de um dos serviços do IMS, há primeiro uma consulta de informação ao Home Subscriber Server (HSS). O HSS é uma base de dados com os utilizadores e os respectivos serviços a que cada um tem acesso. Se depois da consulta ao HSS se verificar que o terminal pode pedir aquele serviço, é alocado um S-CSCF para estabelecer o serviço pedido. Desta forma o S-SCSF é o responsável por manter a chamada ou outro serviço enquanto este é requerido. As funções do I-CSCF serão abordadas com maior detalhe no capítulo 2.

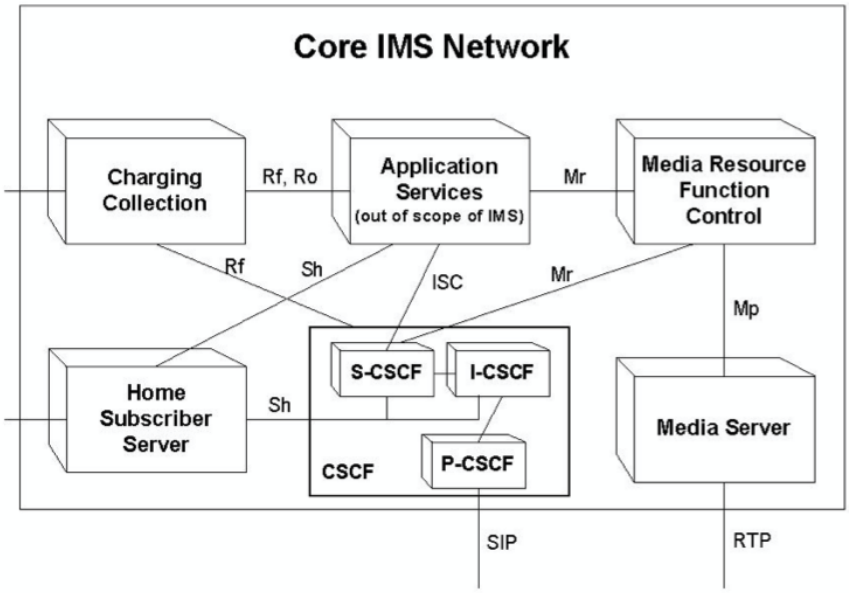


Figura 1.1: Estrutura do núcleo do IMS [31]

1.2.2 Esquema da Fraude

O sistema VoIP é responsável por estabelecer sessões de serviços telefónicos. Como já foi referido, a tecnologia VoIP utiliza o SIP como protocolo de sinalização para assegurar autenticações, predefinições de serviço e todo o tipo de serviços preliminares para estabelecer sessões. O SIP é um protocolo de pedido e resposta onde cada agente pode ter

funções de cliente e de servidor, fazendo pedidos e respondendo aos pedidos de outro agente.
Um terminal que se pretende registar na rede envia uma mensagem SIP na forma de REGISTER com as suas credenciais (nome e password) para um servidor proxy, ou seja, P-CSCF. Como resposta o terminal pode receber uma mensagem 200 OK se o nome existir e a password estiver correcta. Se não receber um 200 OK, o terminal pode receber um UNAUTHORIZED se o campo de password estiver vazio, 403 FORBIDDEN se a password estiver errada, ou 404 NOT FOUND se a conta não existir. Para além de se registar, o terminal pode também solicitar informações ao servidor com a mensagem SIP OPTIONS à qual o servidor é responsável por responder, mesmo que o terminal não esteja autenticado.
Os ataques em análise nesta dissertação têm como base autenticações fraudulentas que permitem o estabelecimento de chamadas. Desta forma, todas as chamadas feitas para o número inteligente ou para outros países, vão gerar receitas para o atacante, dono desse número inteligente ou de uma das empresas responsáveis pelo roaming. Para que estas chamadas sejam possíveis é necessário que o atacante consiga registar-se num proxy com funções de gateway.
Uma das possíveis assinaturas de ataque a um sistema IMS é descrita em [22] . O autor apresenta uma assinatura de ataque com base em dados recolhidos por uma rede secundária sem qualquer utilidade para comunicações. Esta rede secundária contem P-CSCFs com vulnerabilidades de segurança, de forma a serem vítimas de ataques deste género.
Depois de vários anos a reunir assinaturas de ataques chegaram à conclusão que um ataque a uma rede IMS é composto por 3 fases.
A primeira parte do ataque tem como objectivo obter o conjunto de endereços IP de dispositivos SIP à escuta no P-CSCFs. O atacante obtém esta informação ao enviar mensagens OPTIONS com vários endereços IP para o P-CSCF. Como foi referido anteriormente o servidor tem obrigação de responder às mensagens OPTIONS de um terminal, mesmo que este não esteja autenticado. Desta forma, depois de experimentar um conjunto de IPs, o atacante tem informações em relação a existência de dispositivos SIP representados por esses endereços na P-CSCF.
Depois de obter informações referentes ao conjunto de IPs do proxy, o atacante passa para a fase seguinte onde tenta registar-se com números de contas conhecidas ou aleatórias, sem submeter uma palavra passe. Com este método o atacante é capaz de averiguar quais as contas activas com base nas respostas que recebe do servidor (UNAUTHORIZED ou 403 FORBIDDEN se esse dispositivo existir, ou 404 NOT FOUND se não existir).
Assim que o atacante sabe quais são as contas activas e o domínio responsável pelas mesmas, começa a terceira fase do ataque, onde experimenta palavras chave com as contas que já conhece. As palavra chave que são submetidas podem ser geradas de forma aleatória ou podem fazer parte de um dicionário de palavras chave prováveis.
No fim das três fases o atacante consegue autenticar-se no P-CSCF e é autorizado a fazer chamadas cujas receitas o irão beneficiar financeiramente.

Até agora foram descritos dois métodos de chamadas fraudulentas, uma através de carriers envolvidos no ataque e outra a partir de números inteligentes.
Sempre que uma chamada telefónica tem como destino uma rede num país diferente da rede de origem esta chamada está sujeita a roaming. Para chegar à rede de destino, a informação da sessão passa por vários agentes de transporte exteriores ao país de origem a que se chama carriers . Quando uma chamada passa por um carrier é cobrado à rede de origem um valor para compensar a utilização desta rede. Se o atacante tiver acesso às receitas de um desses carriers é do seu maior interesse que o número de chamadas que esse carrier encaminha seja o maior possível.
Por fim, o segundo modo de fraude pode ser executado quando atacante é dono de um número inteligente em que, por cada chamada efectuada, uma parte do que é cobrado ao utilizador reverte para o dono do número inteligente. A partir de várias chamadas por vários terminais simulados em software o atacante é capaz de desviar grandes quantias das receitas geradas nessa chamada.

1.3 Solução Proposta

Considerando o objetivo desta dissertação, a solução proposta passa por desenvolver um método de detecção de fraude inteligente com base num algoritmo de classificação de pedidos de registo.
Como referido anteriormente, qualquer comunicação estabelecida por VoIP é precedida de trocas de mensagens SIP. Desta forma o modo de utilização de cada utilizador pode ser analisado e padronizado pelo tipo e frequência das suas mensagem SIP.
O trabalho realizado na tese teve como base dados de trocas de mensagens SIP recolhidos numa operadora durante um período de tempo, sendo que existem dados considerados não fraudulentos (não foram detectadas quaisquer fraudes relacionadas com essas mensagens) bem como dados referentes a ataques detectados.
Analisando esses dados é possível verificar que os utilizadores regulares partilham o mesmo padrão de utilização e que os utilizadores fraudulentos se destacam dos restantes pelo tempo entre mensagens, sequências de mensagens e IPs utilizados. A partir dessa análise, é proposto aplicar um algoritmo de outlier detection ou classificação com o intuito de detectar os intrusos.
Existem vários algoritmos tanto para detecção de anomalias como para classificação que, dependendo do tipo de dados, conseguem apresentar um modelo mais ou menos preciso. Como tal, os dados obtidos foram processados e aplicados a vários algoritmos, de forma a obter um algoritmo com melhores resultados para os dados disponíveis. Os dados recolhidos foram aplicados a algoritmos de classificação supervisionada como K-Nearest Neighbor, Regressão Logistica, Support Vector Machine, Decision Tree e uma rede neuronal com softmax na ultima camada. De forma a testar a possibilidade de uma abordagem não supervisionada, capaz de aprender o padrão de comportamentos normais, com o intuito de detectar outliers, os dados obtidos foram submetidos a um Autoencoder e ao

algoritmo K-Means.

1.4 Estrutura da Dissertação

A dissertação tem os seguintes próximos capítulos:

Capítulo 2 O capítulo 2 tem com objectivo a revisão do estado de arte, introduzindo o leitor à arquitectura IMS, à tecnologia VoIP, aos algoritmos mais apropriados para a detecção de comportamentos fraudulentos, assim como soluções previamente usadas com base nesses algoritmos.

Capítulo 3 O capítulo 3 apresenta a abordagem inicial, método de definição de comportamento suspeito no processo de registo e uma análise ao dataset utilizado nos testes do seguinte capitulo

Capítulo 4 O capítulo 4 contém os resultados experimentais e uma breve conclusão dos resultados obtidos em todos os modelos testados.

Capítulo 5 O capítulo 5 tem como objectivo explicar o objectivo e o método de implementação do modelo escolhido com base nos testes do capitulo anterior.

Capítulo 6 O capítulo 6 apresenta o sumario de contribuições do trabalho desenvolvido, as conclusões obtidas durante o projecto e os próximos passos do modelo implementado.

t

Capítulo 2

Estado de Arte

2.1 Estado de Arte

Neste capítulo será apresentada uma revisão do estado de arte das aplicações de algoritmos de detecção de anomalias para segurança de uma rede IMS.Considerando que esta dissertação se foca na segurança em VoIP, tambem será feita uma introdução teórica a esta tecnologia, assim como ao sistema em si.

2.2 Introdução dos Conceitos Básicos

A introdução de conceitos básicos tem como objectivo dar ao leitor os conhecimentos previamente necessários para entender as necessidades de um algoritmo aplicado à segurança no IMS e no VoIP. Para tal neste capítulo será abordado de forma genérica o IMS, a sua arquitectura e a relação do SIP com o VoIP referindo exemplos práticos da sua utilização.

2.2.1 IMS

A convergência de tecnologias de comunicações fixas e móveis foi uma necessidade óbvia, e continua a ter a sua importância uma vez que os dispositivos móveis representam um mercado em contínua expansão. Na presente era, os dispositivos móveis são cada vez mais completos e para além de câmaras fotográficas estão também equipados com vários sensores.

Através do IMS é possível interligar toda a rede telefónica fixa e móvel, enviar ficheiros como vídeo, imagem, documentos, e até estender esta conectividade a outros dispositivos, como por exemplo receber chamadas VoIP num computador. Tudo isto é possível com a interligação de meios de acesso com padrões e normas IP. De forma a comunicar através

das normas IP, as aplicações necessitam de uma tecnologia para alcançar o utilizador de destino. A rede PSTN é capaz de estabelecer ligações entre dois pontos através de uma conexão ad hoc. Esta conectividade é conseguida dentro do mesmo operador. No entanto entre diferentes operadores pode ser um processo complicado. Com o objectivo de cumprir estas exigências foi criado o IMS.

Na década de 80 a European Telecommunications Standards Institute (ETSI) definiu as normas GSM para comunicações móveis. Em 1998, foi fundada a 3GPP com o intuito de criar um novo padrão. Depois de varias versões, foi introduzido o IMS na Release 5. Como referido anteriormente o IP Multimedia System é um sistema de controlo de serviços de telecomunicações independente dos meios de acesso que, a partir das normas e padrões IP, é capaz de interligar diferentes tecnologias de comunicação. Desta forma, o IMS dá ao operador a possibilidade de ter um Open IP-Based Service que permite a convergência de varias tecnologias de transporte, multimédia e serviços e que ao mesmo tempo é capaz de as gerir [39] .

Sendo o IMS baseado em normas IP, os identificadores de cada user são alterados para as ações dentro do IMS. Isto é, cada utilizador tem um endereço público conhecido (ex:+351211111111 ou sip:351211111111@operadora.pt) no entanto para o IMS esse utilizador tem um identificador privado, com o formato utilizador@operadora.pt que representa a sua subspcrição. Um utilizador pode ter mais do que um identificador, tanto público como privado, no entanto, cada cartão SIM contem apenas um identificador privado [40] como podemos ver na figura 2.1.

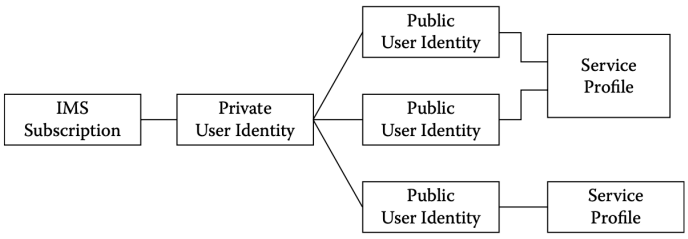


Figura 2.1: Identidades em IMS [40]

Se o registo de cada identidade pública tivesse de ser feito sequencialmente em UMTS seriam alocados recursos de rádio desnecessários. Tendo em conta o problema que este formato de registo representaria para casos em que um utilizador detém diversas identidades públicas com diferentes tipos de serviço, o 3GPP criou o Registo Implícito. O registo implícito possibilita formar um grupo de identidades públicas que são registadas de uma só vez quando uma dessas identidades é autenticada. Da mesma, forma quando uma dessas identidades anula o seu registo, todas as outras deixam também de estar registadas.

A partir de todas estas características o IMS apresenta o seu maior valor quando se pretende criar soluções com serviços inovadores e com várias funcionalidades como chats,

mensagens, voz, vídeo, TV, entre outros. Se estas funcionalidades não estivessem interligadas pelo operador, o utilizador seria o responsável pela reunião de todos os serviços.

2.2.1.1 Estrutura do IMS

Quando o IMS foi inicialmente desenvolvido o 3GPP decidiu estruturar o sistema numa arquitectura de camadas, separando o plano do utilizador, plano de controlo e plano de aplicações como se pode observar na figura 2.2. Nesta dissertação não será abordado em detalhe o plano de utilizador, uma vez que a fraude a combater se desenvolve na camada de controlo. A camada de controlo do IMS contém várias entidades, no entanto iremos apenas abordar o CSCF e a HSS, uma vez que estas são as entidades responsáveis pelo registo, autenticação, e estabelecimento de sessões VoIP. O CSCF é na realidade um servidor SIP que comunica com o HSS para localizar, taxar e autenticar os utilizadores [12].

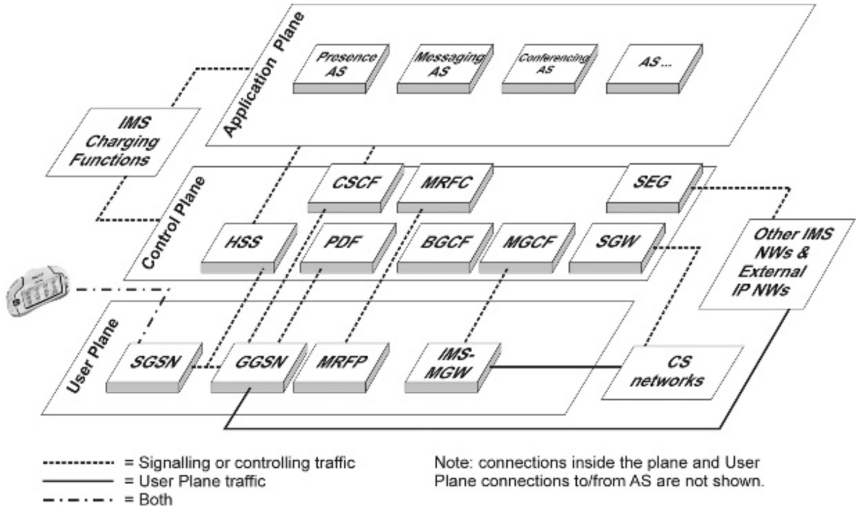


Figura 2.2: Estratificação dos Planos do IMS [39]

Como foi referido no capítulo anterior, o CSCF divide-se em 4 componentes com responsabilidades no estabelecimento de sessões comuns ,e um componente para sessões de emergência:

- Proxy Call Session Control Function (P-CSCF)
- Serving Call Session Control Function (S-CSCF)
- Interrogating Call Session Control Function (I-CSCF)
- Emergency Call Session Control Function (E-CSCF)

P-CSCF: O P-CSCF é responsável pelas interações entre a camada de controlo e a camada de transporte. Assim sendo, o P-CSCF assegura todas as comunicações entre o terminal e o IMS. Todo o tráfego de sinalização SIP enviado pelo terminal terá como destino o P-CSCF. Da mesma forma, todo o tráfego de sinalização da rede que tem como destino o terminal do utilizador, chega ao mesmo por intermédio do P-CSCF. O P-CSCF acaba por ser um representante do terminal para operações de sinalização na rede, tendo quatro funções essenciais para a comunicação, como a compressão de mensagens SIP, IPSec Security Association (IPSec SAs), interação com o Policy and Charging Rules Function (PCRF) e por fim detecção de sessões de emergência.

Sendo o SIP um protocolo de texto, este contem vários cabeçalhos, parâmetros de informações de segurança, entre outros tipos de informação que tornam os pacotes demasiado extensos em comparação com protocolos de codificação binária. Para acelerar o processo de autenticação entre o terminal e o P-CSCF, o 3GPP permitiu a compressão de mensagens SIP entre o terminal e o P-CSCF, se o terminal o solicitar. Para cumprir as exigências de um terminal, o P-CSCF necessita de ter funções de compressão das mensagens SIP. O P-CSCF é também responsável por garantir a segurança e a confidencialidade das mensagens SIP. Estas garantias são asseguradas a partir de Security Associations (SA), que são padrões de segurança estabelecidos entre entidades individuais que compõem o IMS [4]. Estes padrões de segurança são definidos na autenticação do terminal com o P-CSCF. As medidas de segurança no IMS serão explicadas com mais detalhe na secção 2.2.2.1.

O P-CSCF tem também como obrigação comunicar com o Policy and Charging Rules Function (PCRF). O PCRF é um agente importante em comunicações VoLTE, uma vez que funciona como uma mediador de recursos para o IMS, de forma a que seja possível alocar a largura de banda necessária e os devidos atributos da chamada a ser efectuada. Desta forma o operador tem capacidade de oferecer diferentes categorias de serviços de voz com base no limite de recursos a ser alocados. Para saber o tipo de recursos necessários e permitidos o PCRF recorre ao Subscriber Profile Repositories (SPR) que contem as informações referentes aos requisitos de qualidade de serviço, recursos disponíveis e atributos necessários para os serviços de cada utilizador [15].

Por fim uma das funções do P-CSCF é o estabelecimento de chamadas de emergência. Quando o utilizador utiliza um número de emergência para fazer uma chamada, o terminal detecta que o número a ser usado é um número de um serviço de emergência. Ao comunicar com o P-CSCF, o terminal informa o proxy que está a estabelecer uma sessão de emergência utilizando um ID público próprio para esse efeito, enviando também a sua localização, se possível. De seguida o P-CSCF faz a autenticação desse terminal e trata esse registo como um registo prioritário. Após a autenticação o utilizador envia a sessão para o Emergency Call Session Control Function (E-CSCF) da sua rede, que entra em contacto com as autoridades locais com base na localização do terminal [18].

I-CSCF: O I-CSCF é o ponto de acesso com a rede de origem. Desta forma, sempre que um terminal precisa de informações que estão na posse do seu operador, é utilizado o I-CSCF para fazer a comunicação entre os dois. Quando um terminal pretende estabelecer uma sessão, seja uma chamada ou um serviço disponibilizado por um Application Server (AS) , é necessário recorrer à base de dados do operador. Esta comunicação tem como objectivos assegurar que a sessão pedida pelo terminal está dentro das permissões de utilizador e saber qual sera o next hop, seja ele um S-CSCF ou um AS. O I-CSCF é responsável por fazer a ligação entre o P-CSCF e o HSS para obter as informações acima descritas. Quando um utilizador recebe um pedido SIP para estabelecer uma sessão é também o I-CSCF a entidade responsável por indicar qual deve ser o S-CSCF a ser alocado para assegurar o estabelecimento de sessão [39].

S-CSCF: O S-CSCF é a entidade com as maiores responsabilidades no IMS, sendo este responsável por estabelecer sessões, decisões de routing, manter as sessões, armazenar os perfis e os registos. Quando um terminal quer estabelecer uma sessão, esse pedido é encaminhado para o S-CSCF, que faz o download dos dados de autenticação do terminal armazenados no HSS. Com base nos dados recolhidos, responde ao pedido do terminal e espera pela sua resposta. Depois de receber e verificar a resposta do terminal o, S-CSCF aceita o pedido de sessão e supervisiona a mesma. Através do endereço público, o S-CSCF faz o download do perfil de serviço do utilizador em questão quando este se regista no IMS. Com a informação presente no perfil de serviço, o S-CSCF decide quais os ASs a utilizar quando o utilizador envia ou recebe um pedido SIP de outro terminal.

As decisões de routing estão também ao encargo do S-CSCF, visto que esta entidade recebe as sessões e os pacotes do terminal de origem e de destino. Quando o S-CSCF recebe um pedido do início de sessão vindo de um P-CSCF, é ele que decide se contacta os ASs antes de dar continuidade ao pedido. Se o utilizador utiliza um Mobile Station ISDN (MSISDN) para anunciar o destino de uma sessão, o S-CSCF converte o número num SIP Universal Resource Identifier (URI), como por exemplo sip:bob@ims.example.com. Embora o S-CSCF saiba o endereço IP do terminal a partir dos dados de registo, este envia todos os pedidos por intermédio do P-CSCF, para que este cumpra com as suas funções de compressão e segurança. Por fim, o S-CSCF é ainda capaz de enviar informações relativas a taxação para o Online Charging System (OCS) de forma a manter os registos de taxação online atualizados [39].

Base de Dados No IMS existem duas bases de dados, o Home Subscriber Server (HSS), e o Subscription Locator Function (SLF). O HSS é a base de dados central onde os utilizadores e os serviços do IMS estão contemplados. A base de dados contém utilizadores, serviços, premissas de serviço, perfis dos serviços, informações de segurança, assim como informações de routing, taxação e localizações [47]. Como já foi dito anteriormente a identidade de um utilizador pode ser dividida em identidade pública e privada. A identidade privada é da responsabilidade da operadora e é utilizada para a autenticação e

para o estabelecimento de sessões. A identificação pública, por sua vez, é a identificação que o próprio utilizador pode usar para mostrar o seu interesse de estabelecer uma sessão. Tendo em conta que o HSS armazena os perfis de utilizador, a base de dados detém também os requisitos que um utilizador exige para um S-CSCF quando este pede um determinado serviço. Como tal, quando o I-CSCF está responsável por alocar um S-CSCF a um utilizador, essa alocação é feita com auxílio da informação disponível no HSS em relação às características de S-SCCF pedidas pelo utilizador [39].

O HSS contém, por sua vez, um conjunto de bases de dados locais e de autenticação de modo a separar a informação necessária, repartindo o processamento. As bases de dados local Home Location Register e Authentication Center (HLR/AUC) são também um requisito do Packet-Switched (PS) e do Circuit-Switched (CS). O HLR é uma base de dados focada nas subscrições de utilizadores, que são identificados com o International Mobile Station Identity (IMSI) ou com o MSISDN. Por sua vez, o AUC é uma base de dados dedicada a dados de autenticação e autorizações. O AUC armazena as credenciais de segurança de um utilizador e gera um vector aleatório como palavra chave para aumentar a segurança na autenticação. Embora tenham a sua utilidade, o 3GPP não obriga a que haja uma distinção no HSS para criar o AUC e o HLR [47].

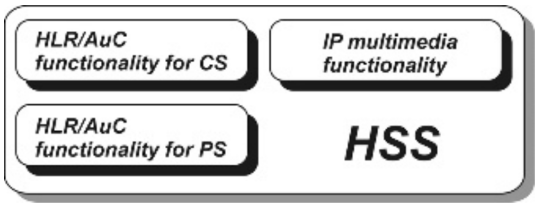


Figura 2.3: Arquitectura da Base de Dados [39]

Application Server No IMS estão definidos 3 tipos de ASs: SIP Application Servers, Open Services Architecture (OSA) application servers e Customized Applications for Mobile network Enhanced Logic (CAMEL) [28]. Os ASs são responsáveis por disponibilizar serviços multimédia no IMS. Este componente encontra-se na rede do operador ou numa localização externa a qualquer das redes. O ASs é capaz de receber e responder a pedidos SIP, assim como enviar informações de montantes a cobrar para a entidade que controla os registos financeiros [39]. Um servidor SIP pode ser usado pelo operador para ser responsável por Presence, Messaging, Push to talk Over Cellular, Multiparty Gaming, serviços de video-conferência, entre outros. Assim sendo um AS pode ter responsabilidades específicas num só serviço, como por exemplo Presence, e outro AS pode ser responsável por Messaging. Quando um utilizar pretende enviar uma mensagem por Instant Messaging vai utilizar os serviços de dois ASs. Tendo em conta o que foi descrito como sendo o problema a ser abordado pela dissertação, os SIP ASs são componentes importantes na fraude em estudo. Este tipo de ASs podem redirecionar, originar, e terminar sessões. Podem ainda agir como um proxy, assim como podem ser um back-to-back User Agent, ou

seja, um elo de ligação entre dois utilizadores em sessão [28]. Sendo o AS o responsável por redirecionar chamadas, é a este componente que o S-CSCF envia os pedidos SIP para uma sessão cujo utilizador de destino não se encontra autenticado, ou quando o mesmo não responde a esse pedido. Ao receber estas informações, o AS age como um servidor de Voice Mail ou redireciona a sessão para outro utilizador.

2.2.1.2 Session Initiation Protocol

SIP é um protocolo de controlo da application-layer capaz de estabelecer, modificar e terminar sessões multimédia como chamadas e conferências. Para além de tudo isto, o SIP é capaz de convidar participantes para sessões previamente estabelecidas, assim como consegue adicionar recursos a uma sessão. Suporta também o mapeamento de nomes e redirecciona serviços para terminais móveis [25]. O SIP torna-se assim fundamental para obter a localização, disponibilidade de um terminal, determinar os parâmetros de recursos a serem utilizados e para gerir as sessões. Quando um pedido SIP é enviado para um proxy tem no seu cabeçalho um endereço URI para o qual se destina o pedido. Para localizar o terminal de destino o proxy utiliza a informação presente no HSS. Para explicar melhor o papel do protocolo SIP no estabelecimento de chamadas que por consequência o torna vital no IMS, vai-se recorrer a um exemplo simples do estabelecimento de uma sessão entre dois terminais. Como é comum utilizaremos a Alice num softphone e o Bob que opera um SIP Phone.

A Alice pretende estabelecer uma sessão com o Bob. Para tal a Alice envia primeiro um INVITE com o seu URI e o URI destino para o seu P-CSCF que responde com um 100 Trying. Assim que o P-CSCF responde à Alice, envia também um INVITE para o P-CSCF responsável pelo Bob que, ao propagar o INVITE para o terminal do Bob responde, ao P-CSCF da Alice com um 100 Trying. A partir do momento em que o terminal do Bob recebe o INVITE, é propagada uma mensagem SIP 180 Ringing do Bob para o seu P-CSCF, e daí em diante até chegar à Alice. Quando a chamada é aceite pelo Bob, é propagado um 200 OK da mesma forma que se propagou o 180 Ringing. O 200OK enviado contém também um SDP com a descrição dos recursos a serem usados e o tipo de sessão que está a ser estabelecida. Quando a Alice recebe o 200 OK do Bob, envia um ACK a informar que tomou conhecimento do início de sessão. Depois de tudo isto, a sessão é estabelecida e são trocados os pacotes com as informações de ambas as partes. Como foi dito, o SIP é um protocolo de sinalização, ou seja, a troca de pacotes da sessão não é da responsabilidade do SIP mas sim do protocolo de media a ser usado, como por exemplo RTP. Por fim, quando a Alice quiser terminar a chamada, é enviada uma mensagem SIP com BYE, que é precedida de um ACK do Bob quando este o recebe.

Este exemplo mostra as mensagens SIP usadas para estabelecer uma sessão, no entanto, o IMS não é apenas responsável por estabelecer sessões telefónicas. É necessário redireccionar pedidos para ASs, autenticar e registar utilizadores, consultar o HSS entre outras operações. Tudo isto é feito com base no Session Initiation Protocol.

Uma das diferenças do SIP em relação a outros protocolos de sinalização reside no facto de um servidor ser capaz de, ao receber um pedido SIP, enviar duas ou mais mensagens SIP de resposta a esse pedido, para destinos diferentes. Estas várias respostas podem ser enviadas de uma só vez, ou sequencialmente, se um pedido anterior não tiver tido sucesso. Esta vantagem é de grande importância nos serviços telefónicos, como redirecionamento de chamadas para o voice mail, distribuição de chamadas e localização do utilizador, visto que o mesmo número pode estar autenticado em vários terminais [45].

2.2.1.3 Registo de um terminal

Como foi referido, o registo de um terminal no IMS é feito através da troca de mensagens SIP. A autenticação é um dos processos com maior importância na administração do IMS, sendo esse o processo que permite que os terminais usufruam dos serviços do IMS.

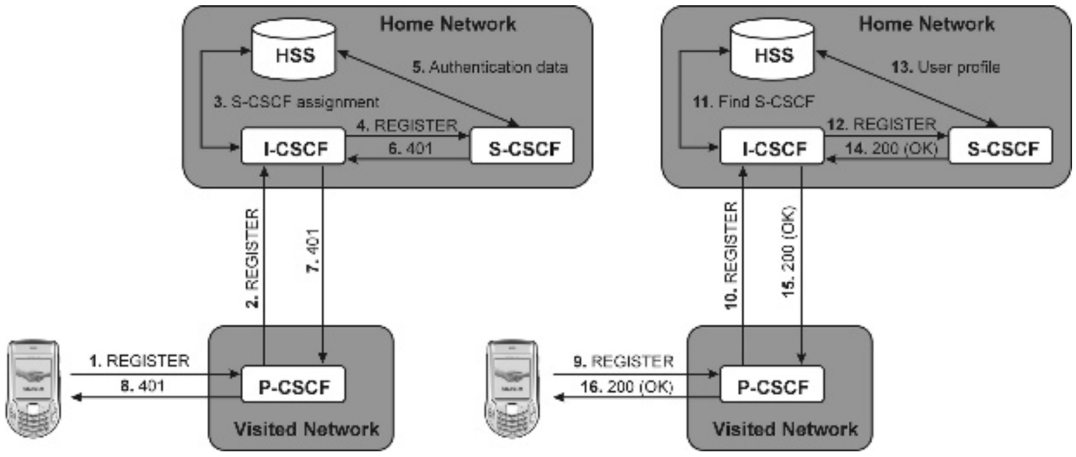


Figura 2.4: Registo em IMS [39]

O registo de um terminal tem duas fases. Na primeira fase a rede desafia o terminal com base num pedido prévio do mesmo. Na segunda fase o terminal responde ao desafio, ficando por fim registado. Para um terminal se registar este envia um pedido SIP REGISTER a fim de encontrar um P-CSCF. Neste pedido SIP, é também enviado um identificador e o domínio da Home Network, para que o P-CSCF ao verificar o pedido REGISTER consiga saber o IP do I-CSCF a alocar. Depois de encontrado o I-CSCF, este componente consulta o HSS e, com base no identificador, é capaz de saber quais são os requisitos do S-CSCF que o utilizador pretende. Ao saber qual o S-CSCF, o I-CSCF envia-lhe o pedido de autenticação. Ao receber o SIP REGISTER, o S-CSCF verifica que o terminal não está registado e recolhe as suas informações de autenticação (primeiros dados do terminal) presentes no HSS. De seguida, o S-CSCF calcula um valor que vai representar a palavra chave do registo (XRES) com base em parâmetros de segurança (RAND e AUTN) e desafia o terminal com uma resposta SIP 401 Unauthorized, que contem um campo RAND e

AUTN. Ao receber o 401 Unauthorized o terminal calcula também a palavra chave de registo (RES) com base no valor de RAND e o AUTN que recebeu do S-CSCF. De seguida o terminal responde com outra mensagem SIP REGISTER com a chave calculada (RES). Essa resposta tem como destino o P-CSCF que, por sua vez, volta a encontrar o I-CSCF e posteriormente o S-CSCF. Quando o REGISTER, com a chave calculada, chega ao S-CSCF, este compara o XRES com o RES e, se forem iguais, descarrega as informações do terminal e aceita o registo enviando uma resposta 200 OK [2]. É da responsabilidade do terminal manter o seu registo activo, através de pedidos de registo periódicos. Caso contrário, o S-CSCF vai remover os terminais que não se registaram durante um período de tempo.

2.2.2 VoIP

O protocolo IP apresenta enormes vantagens, uma vez que é o protocolo utilizado para conectar praticamente todos os dispositivos devido à sua importância na internet. Com a proliferação que o IP tem nas redes mundiais, os produtores de aplicações não necessitam de fazer várias versões consoante o protocolo em uso em redes diferentes. Sendo a versatilidade do IP uma vantagem inegável, o VoIP constitui um serviço bastante viável, transformando os serviços telefónicos analógicos num serviço com normas IP. Até ao desenvolvimento do VoIP, os serviços telefónicos eram assegurados pelo PSTN baseado numa infraestrutura analógica. Um dos problemas de transmissão de sinais analógicos reside no facto de necessitarem de um amplificador de sinal, uma vez que o sinal se deteriora com a distância entre os dois pontos. No entanto, quando o sinal é amplificado, é também amplificado o ruído do mesmo. Em transmissões digitais a amplificação do sinal não amplifica o ruído, o que representa uma enorme vantagem em relação ao PSTN [16].

A base operacional do VoIP consiste em enviar o sinal de voz através de pacotes IP. Para tal, a voz do utilizador tem de ser convertida para um sinal digital, que depois é comprimido e sequenciado em vários pacotes. Esses pacotes são colocados no payload de um pacote IP devidamente endereçado. Quando o pacote é recebido, o sinal é novamente convertido para voz, de forma a que seja audível pelo utilizador de destino. Desta forma, o VoIP permite que voz e internet sejam transmitidos em simultâneo por uma única linha telefónica.

Segundo Weiss et al , [48] o custo de uma rede pronta para VoIP, seria igual ou inferior ao custo de uma rede tradicional para PSTN. Esta redução de custos pode justificar-se com o uso mais eficiente da largura de banda disponível. Enquanto o PSTN necessita de um canal Full Duplex de 64Kbps para uma chamada, o VoIP consegue fazer o mesmo com 14Kbps devido à compressão de voz e à reutilização de largura de banda quando nada está a ser transmitido. Para além de ser mais económico para a operadora, esta tecnologia é também mais económica para uma empresa que necessite de serviços de telecomunicações. Com VoIP, a empresa deixa de pagar um custo fixo por mês para ter uma linha dedicada e deixa de ter taxação por minuto, mantendo apenas o custo da sua PBX, da infraestrutura IP e dos IP phones [30]. Como a PSTN ainda é uma realidade,

é necessário haver interligação entre VoIP e PSTN através de gateways. Como podemos observar na figura 2.5 o VoIP pode ser usado em sintonia com o PSTN e para além disso permite a comunicação entre IP phones. Pode-se também observar pela imagem que a sinalização e o transporte têm agentes diferentes.

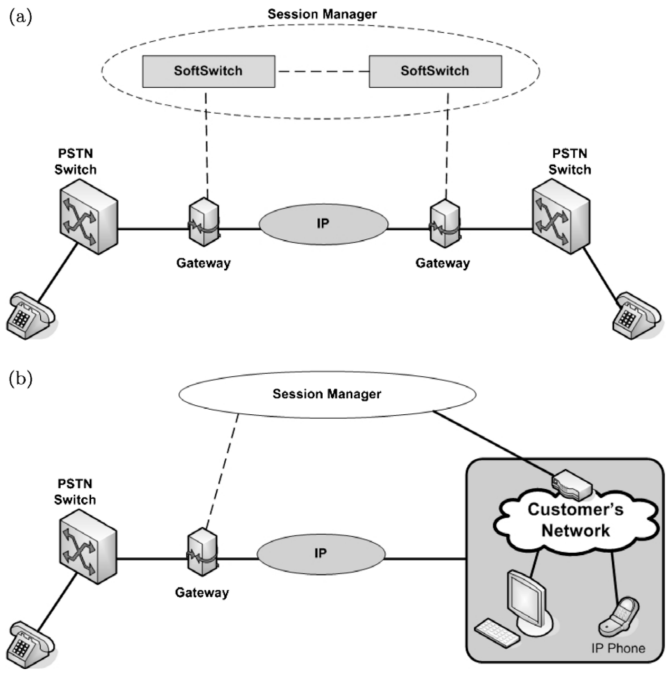


Figura 2.5: Exemplo da utilização de VoIP [26]

2.2.2.1 Segurança no IMS

Visto que o IMS suporta serviços VoIP a segurança e confidencialidade das conversações é uma prioridade. Segundo Anderlind et al, [4] na maioria dos casos, a autenticação e integridade do sistema são mais importantes do que a confidencialidade, quando se quer proteger a rede de fraudes e DoS. Como tal, são necessárias medidas de segurança. Essas medidas existem na forma de standards, que são regulados pelo 3GPP ou 3GPP2. Uma das medidas de segurança são as Security Associations (SA), que são estabelecidas entre os vários componentes do IMS que comunicam entre si. Estas SAs podem focar-se em segurança de acesso ou segurança do domínio da rede. Na figura 2.6 estão assinaladas as diferentes SAs estabelecidas entre alguns componentes do IMS. A SA 1 e 2 são de maior importância para o tema da dissertação uma vez que são responsáveis pela autenticação de um terminal no IMS. O IPsec-3GPP é o mecanismo de segurança usado pelo 3GPP e 3GPP2. O IPsec cria dois pares de IPsec SAs; um par para as trocas de informações iniciadas pelo terminal e um segundo par para comunicações iniciadas pelo P-CSCF. O IPsec é também responsável pela encriptação do payload, garantindo a integridade da conexão e autenticação de um pacote [23]. A segurança do registo de um terminal é um dos processos mais importantes, visto que a partir desse momento o terminal registado

passa a poder usufruir dos serviços do IMS. O processo de registo de um terminal é chamado IMS AKA e baseia-se numa palavra chave partilhada entre o ISIM do utilizador e o HSS.

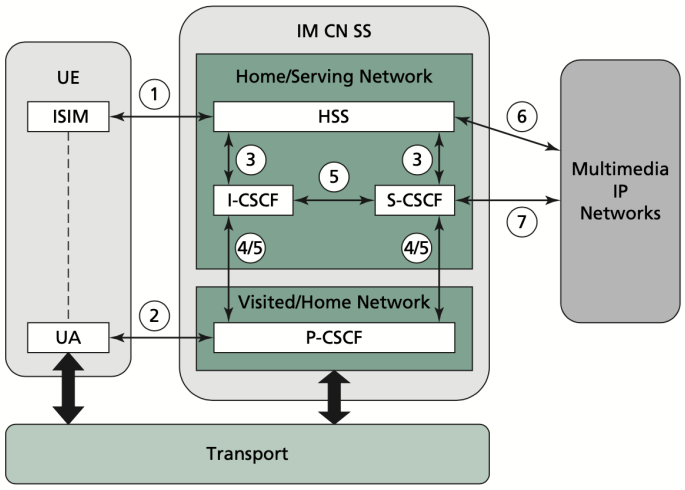


Figura 2.6: Segurança da arquitectura IMS [4]

Para garantir a segurança do IMS, o terminal necessita de fazer um pedido para se registar. O processo de registo necessita de ser criterioso de modo a impedir a usurpação de identidades no IMS. Desta forma o terminal necessita de ultrapassar um desafio apresentado pelo S-CSCF a si alocado como explicado na secção 2.2.1.3

2.3 Métodos de Detecção de Fraude

Nesta secção serão explicados os algoritmos disponíveis para a detecção de comportamentos anormais num rede de telecomunicações. Para tal ,serão abordados vários exemplos onde os autores utilizaram algoritmos de machine learning para detectar comportamentos fraudulentos, autenticações forçadas, e outros comportamentos anormais.

2.3.1 Algoritmos de Detecção de Outliers por Densidade

Os algoritmos de detecção de outliers baseados em densidade utilizam estatística para detectar acontecimentos fora do padrão da normalidade. Estes algoritmos necessitam de um conceito limite, para o qual se conclui se um acontecimento se desvia do padrão geral porque é um outlier ou simplesmente porque teve um valor um pouco fora do normal. Este conceito tem o nome de threshold. O threshold representa o desvio padrão de um acontecimento

2.3.1.1 Deteção de autenticação fraudulenta

T. Jansky et al [24] apresentam uma solução para detectar ataques no processo de autenticação numa PBX com funções de P-CSCF. Neste caso os atacantes forjavam a autenticação

a partir de dois métodos, várias tentativas de extensões e várias tentativas de palavras chave. Para combater as tentativas de autenticação, T. Jansky et al utilizaram um algoritmo de detecção de outliers. Para tal, os autores recolheram mensagens SIP do processo de autenticação de uma PBX. Os registos de autenticação eram depois enviados para um colector no formato IPIFIX e posteriormente analisados pelo algoritmo desenvolvido. Dessas mensagens, foram analisadas as respostas 401 e guardaram os IPs e extensões dos clientes.

O algoritmo proposto é composto por duas fases. Na primeira, os dados são recolhidos e guardados numa estrutura de dados. Para cada SIP Server, são guardados numa lista os IPs dos clientes e os seus nomes de utilizador. Estes dados estão todos relacionados para saber quantas vezes um cliente (IP) tentou entrar com um determinado nome de utilizador. Na segunda fase o algoritmo passa a avaliar os dados que recolheu. Se um cliente tentar autenticar-se com mais do que uma extensão (Username), o comportamento é classificado como um scan. Quando o número de tentativas ultrapassa um determinado threshold, o comportamento é classificado como um ataque e é reportado. Se depois de um ataque o servidor responder com um 200 OK o ataque foi bem sucedido, caso contrário, não havendo um 200 OK dentro de uma determinada janela temporal, a sua informação é apagada da estrutura de dados.

Neste caso o número de tentativas para que um comportamento se classifique como um ataque é um threshold. Este valor é obtido através de uma análise do comportamento de uma rede real. Ao analisar as mensagens SIP na rede CESNET2, os autores chegaram à conclusão que mais do que 99.9% dos registos bem sucedidos usam 20 mensagens ou menos. Com base nesta observação, o threshold para o número de tentativas foi definido como 20 mensagens. Para além das 20 mensagens, chegaram também à conclusão de que apenas 0.01% dos ataques têm uma janela temporal entre tentativas superior a 30 min. Com base nesta informação, o servidor espera 30 minutos até descartar as informações de uma tentativa de registo. Por fim, analisaram também o número de extensões utilizados por um cliente (IP) e verificaram que a maioria dos clientes tentam menos de 10 extensões. Assim sendo se um IP se tentar registar com mais do que 10, extensões o comportamento é considerado fraudulento. O algoritmo foi testado numa rede real com tráfego malicioso através da SIPVicious [20]. Todos os ataques foram identificados com sucesso. De seguida o algoritmo foi testado durante uma semana na rede CESNET2 e detectaram mais de 7000 eventos fraudulentos. Foi possível concluir que a maioria dos atacantes faz várias tentativas de extensões ou varias tentativas de palavras chave. No entanto, há também atacantes que optam por usar os dois processos em simultâneo. Para tal, são usadas várias extensões e 20 ou 100 palavras chave.

Embora os métodos descritos sejam rudimentares e não ofereçam uma classificação detalhada em relação à actividade do pedidos de registo, foram utilizados princípios semelhantes aos descritos, na análise dos dados da operadora que é descrita na secção 3.2.1.

Assim como os autores concluíram existe um número limite de extensões que cada

utilizador (IP) pretende registar. Embora nem todos os utilizadores que pretendem registar mais do que esse limite de extensões sejam utilizadores fraudulentos, este índice é importante para reduzir o número de casos a classificar.

2.3.1.2 Deteção de telefonemas fraudulentos

Se um utilizador fraudulento se conseguir autenticar, é capaz de fazer chamadas para seu benefício. Para efectuar uma chamada para a PSTN através do IMS é necessário enviar um INVITE com o número a que se pretende telefonar, precedido de um prefixo para chamadas na rede PSTN. Por exemplo, para enviar um INVITE para o número 211 111 111 é necessário enviar um INVITE para 77211111111@exemplo.com. O número 77 é um indicador de que o utilizador de destino se encontra na PSTN e é definido pela operadora. Deste modo, para fazer uma chamada, o atacante tem de adivinhar o prefixo, sendo executados vários pedidos com prefixos diferentes.

Para detectar essas tentativas T. Cejka et al. [7] criaram uma árvore de prefixos para cada cliente (IP). Nessa árvore, os nós representam sufixos comuns aos ramos, enquanto que os ramos representam os diferentes prefixos que o nó tem. A sequência de um nó com os seus descendentes forma um URI. Cada nó contém também o número de tentativas de chamadas, número de chamadas estabelecidas, assim como outras informações para esse URI. As árvores são analisadas periodicamente para detectar ataques. A arvore é

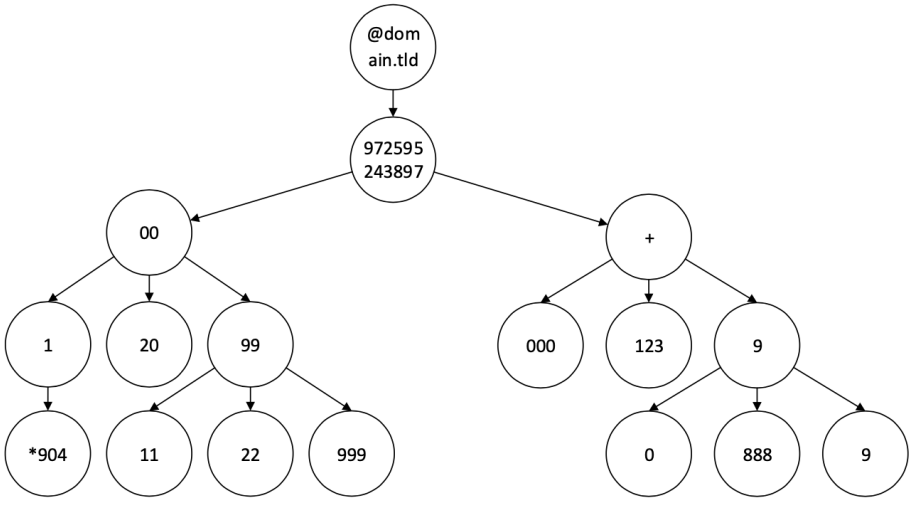


Figura 2.7: Árvore de sufixos para análise de prefixos de números de telemóveis [7]

util uma vez que, quando há um ataque, é observado um grande número de URIs para o mesmo número de telemóvel. Desta forma, um cliente que esteja a tentar um ataque vai acabar por ter uma árvore com um único nó com o número de telefone pretendido e um grande número de ramos. O algoritmo percorre primeiro as árvores de baixo para

cima. Por cada nó, avalia se os seus ascendentes ultrapassam um valor previamente estabelecido Lmax. Quando ultrapassa esse valor é muito provável que se tenha chegado ao número de telefone que o atacante pretende atigir. De seguida, são aplicados dois testes aos descendentes do nó encontrado. No primeiro teste analisa-se se o prefixo que o nó representa é inferior a Lmax e no segundo teste verifica-se se houve alguma chamada sem sucesso para o prefixo do nó. Se o nó passar nestes dois testes, faz-se uma contagem aos seus descendentes e, se foram mais do que o threshold (T) previamente definido, é reportado um ataque para aquele número.

Depois de testar o algoritmo numa rede real com tráfego malicioso através da SIPVicious, [20] o tamanho máximo de um prefixo (Lmax) foi considerado como sendo 10, assim como o threshold (T) para o número de tentativas de chamada. Nesta rede, todos os ataques com mais do que 10 tentativas foram reportados no, entanto houve chamadas bem sucedidas que precisaram de menos do que 10 tentativas. Para testar o algoritmo numa rede com ataques não fabricados, este foi submetido à CESNET2 e o resultado foi idêntico ao da rede com ataques fabricados.

O modelo apresentado parte do princípio que o utilizador fraudulento já se encontra registado no IMS de uma operadora. Assim sendo, apenas poderia ser aplicável ao futuro do projecto, no entanto a operadora em questão não utiliza prefixos para a rede PSTN, o que inviabiliza a sua aplicação. Para além da operadora não utilizar prefixos para a rede PSTN, o modelo obtido é extremamente dispendioso em recursos computacionais, uma vez que criar uma árvore de informação de prefixos usados para cada utilizador da operadora não é exequível em tempo real.

2.3.1.3 Local Outlier Factor na detecção de fraudes VoIP

O Local Outlier Factor (LOF) é um algoritmo de detecção de outliers proposto por Breunig et al. , [5] baseado na densidade local de vizinhos com o intuito de detectar anomalias em conjuntos de dados multidimensionais. O algoritmo valoriza a densidade local na medida em que o grau de outlier depende do quão isolado está um objecto em relação aos seus vizinhos. Assim sendo, para determinar o nível de outlier ,só os vizinhos do objecto em estudo são tidos em consideração. Este algoritmo foi aplicado com sucesso por D. Pokrajac et al. [34] para detectar diferenças repentinas em frames de video, para detectar trajectórias pouco comuns num ambiente controlado por câmaras de vigilância e, para detectar ataques a uma rede, através do conjunto de dados 1998 DARPA Intrusion Detection Evaluation Data.

A. Lazarevic et al. [32] demonstram que o LOF apresenta melhores resultados em relação a outros algoritmos de detecção de outliers quando aplicados à detecção de utilizadores não autorizados numa rede. O índice de isolamento de um objecto é definido por LOF e é calculado da seguinte forma:

1. Cálculo da $k-Distance(a)$ que representa a distância do objecto "a"ao seu kº vizinho mais próximo.

2. Cálculo da Reachability Distance: Para calcular este parâmetro, usa-se o $k-Distance(a)$ para calcular a Reachability Distance pela seguinte formula:

$$reach-dist(a,b) = max(k-distance(b), d(a,b)) \tag{2.1}$$

3. Cálculo da Local Reachability Density: A Local Reachability Density é o inverso da média das Reachability Distances dos vizinhos considerados:

$$Lrd(a) = \sum_{n=1}^{k} reach-dist(a,n) \tag{2.2}$$

4. Cálculo do LOF: Por fim, calcula-se o valor do LOF, que consiste na soma das Local Reachability Density dos vizinhos considerados, sobre a Local Reachability Density do objecto "a"multiplicado pelo número de vizinhos envolvidos:

$$LOF(a) = \frac{\sum_{n=1}^{k} Lrd(n)}{k \times Lrd(a)} \tag{2.3}$$

Este algoritmo foi aplicado por K. Kim et al. [29] para detectar casos de Toll Fraud em serviços VoIP. Para tal, foram usados dados de chamadas de uma operadora que necessitaram de tratamento prévio. Foram recolhidas duas amostras de dados com ataques obtidos a partir de um servidor Asterisk. A amostra 1 tem 105 chamadas fraudulentas, com 1159 mensagens recolhidas desde 29 de junho até 2 de julho de 2012. A amostra 2 tem 87, ataques com 2062 chamadas recolhidas desde 18 de julho a 20 de julho de 2012. Depois de recolhidos, os dados foram normalizados para atenuar as diferenças de escalas entre os atributos. Para tal, os atributos foram redimensionados para estarem compreendidos entre 1 e -1. Os dados originais contêm 18 colunas de informações por chamada. Os autores reduziram a informação para 6 colunas com os seguintes dados:

- Hora de início da chamada
- Tamanho da extensão de destino
- Duração da chamada a tocar
- Duração da chamada
- Desenlace da chamada (atendida, não atendida, número ocupado, incompleta)

Para além destes 6 atributos, os autores introduziram 2 novos atributos: o endereço RTP e o código do pais da chamada de origem. Foram feitos testes no MATLAB variando o número de vizinhos envolvidos, no entanto os autores não verificaram uma relação entre os números de vizinhos envolvidos. Mesmo não havendo uma relação directa, os autores concluíram que o melhor resultado se obtém com 30 vizinhos.

Existem dois critérios para avaliar a detecção de outliers, a precisão e o Recall. A precisão é a relação entre os outliers que realmente são ataques e a totalidade dos outliers detectados.

$$Precisao = \frac{n(FC|outliers)}{n(outliers)} \tag{2.4}$$

Onde n(FC|outliers) representa os ataques detectados como outliers, que realmente são outliers (verdadeiros positivos) e n(outliers) o número total de outliers detectados(verdadeiros Positivos + falsos positivos).

O Recall é a relação entre os ataques detectados sobre os ataques presentes nos dados.

$$Recall = \frac{n(FC|outliers)}{n(FC)} \tag{2.5}$$

Onde n(FC) é o número de ataques presentes nos dados em teste.

Na primeira experiência foram testados os dois conjuntos de dados referentes à amostra 1 e amostra 2 que , como já foi referido, diferem em tamanho e época de recolha. Verificou-se que, quando o número de outliers sobe de 2% para 10%, o algoritmo origina um Recall mais alto com uma precisão mais baixa. Para baixos valores de outliers (2% aos 4%), o Recall da amostra 2 foi bastante baixo enquanto a precisão da amostra 1 variou dos 10% aos 25%. Ao avaliar o modelo com base numa amostra geral, composta pelos dados da amostra 1 e 2, a primeira experiência revelou resultados muito insatisfatórios, com 6,9% de Recall e 9,2 % de percisão.

Na segunda experiência, as amostras testadas apresentam 8 atributos, ao contrario das amostras da primeira experiência que apresentavam 6 atributos. Os dois atributos adicionados são o endereço RTP e o indicativo de pais. Houve uma notória melhoria na performance do algoritmo em comparação com a primeira experiência. Em particular, a amostra 2 teve resultados superiores com poucos outliers, obtendo 60% de Recall para 4% a 10% de outliers e 70% de precisão para 2% de outliers. No entanto, ao avaliar o modelo com base numa amostra geral composta por dados da amostra 1 e 2, a segunda experiência apresentou com 32,5% de Recall e 41,3 % de percisão. Estes resultados não permitem que o modelo seja aplicado com um alto grau de confiança na detecção de ataques.

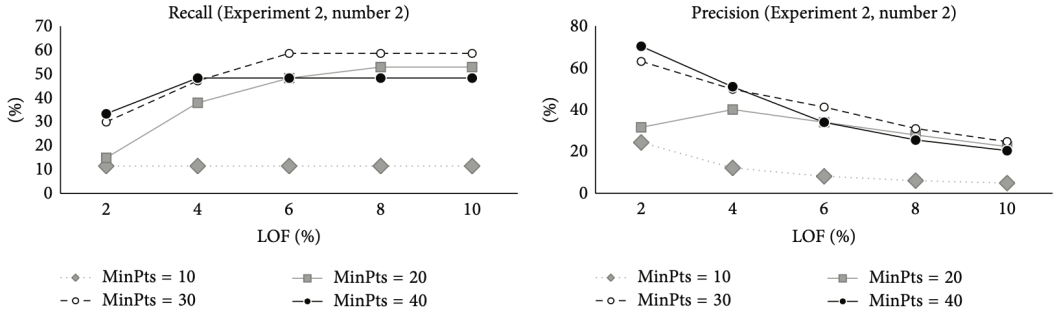


Figura 2.8: Resultados da segunda experiência com a amostra 2 [29]

2.3.2 Algoritmos de Clustering e Classificadores

Os algoritmos de clustering representam um conjunto de técnicas de tratamento de dados com o intuito de criar grupos de objectos com base nas suas semelhanças. O critério com que se atribui um objecto a um determinado conjunto distingue os vários tipos de algoritmos, dentro dos algoritmos de clustering. Os algoritmos de classificação partem de um princípio supervisionado onde existem dados previamente classificados.

2.3.2.1 Aplicação do K-Means à detecção de padrões de tráfego anormais

O algoritmo K-means [37] é um processo de clustering que agrupa objectos com base nos seus valores chave em k conjuntos, sendo k um número positivo definido à posteriori. A execução do algoritmo é composta por 5 passos. Como já foi referido o número de conjuntos pretendidos é um valor definido antes da execução do algoritmo. Após definir o valor de k, são definidos os centros dos k conjuntos. Estes centros podem ser definidos distribuindo os objectos de forma arbitrária, até se verificar que não há centros em comum, ou pelo valor de k objectos selecionados de forma aleatória. Depois de definidos os centros, o algoritmo percorre todos os objectos e calcula a sua distância em relação a cada centro. Atribui-se o objecto ao conjunto com o centro mais próximo das suas coordenadas. Os centros são recalculados e o algoritmo volta a percorrer todos os objectos até que a sua disposição permaneça inalterada.

A distância entre os dois objectos é calculada pela distância euclidiana de acordo com a seguinte função:

$$d(x,y) = \sqrt{\sum_{i=1}^{m}(x_i - y_i)^2} \tag{2.6}$$

Onde $x = (x_1,...,x_m)$ e $y = (y_1,...,y_m)$ são vectores com m dimensões. Tendo em conta que os valores associados a um objecto têm importâncias diferentes, é por vezes necessário aplicar à distância um peso relativo para influenciar a importância dessa característica nos cálculos. Para tal é aplicada a seguinte expressão para a distância euclidiana:

$$d(x,y) = \sqrt{\sum_{i=1}^{m}(\frac{x_i - y_i}{s_i})^2} \tag{2.7}$$

Onde s_i é o factor de importância que a característica tem para o cálculo da distância. Desta forma, quanto mais importante é a característica menor é o valor de s_i.

G. Münz et al. [41] propõe uma abordagem supervisionada ao utilizar o k-means para detectar anomalias de tráfego através dos padrões temporais das comunicações, pela equação descrita em 2.7. Para avaliar os padrões de uma rede, os autores utilizam factores como, o tipo de protocolo, o IP de origem, o IP de destino, o porto de origem e o porto de destino. Os autores definem três tipos de utilização para k-means: classificação, detecção de outliers e um método que combina a classificação e a detecção de outliers. Na classificação, um objecto é considerado normal se estiver mais próximo do centro do

conjunto dos objectos, que depois de uma análise detalhada foram considerados normais, do que do centro dos objectos considerados anormais. Com o método de detecção de outliers, um objecto só é considerado normal se estiver até uma distância d_{max} do centro do conjunto dos objectos normais. Por fim, os dois métodos podem ser usados em simultâneo.

Para testar o algoritmo, os autores geraram tráfego e recolheram os ficheiros tcpdump da residência de estudantes da faculdade de Twente. Na simulação foi gerado tráfego normal e tráfego de ataque de DoS. Os autores reconhecem que este teste não gera conclusões para o tráfego real, o que impede a sua implementação tendo em conta os objectivos da dissertação, no entanto foi conduzida uma segunda experiência para testar a capacidade do algoritmo na descoberta de comportamentos anormais por ataques de DoS. O algoritmo foi capaz de detectar um ataque de DoS. No teste executado com os dados da residência de estudantes da faculdade de Twente não foram detectadas anomalias.

2.3.2.2 Aplicação do k-nearest neighbour à detecção de padrões de tráfego anormais em mensagens SIP

K. Rieck et al. [43] propõe um sistema de aprendizagem contínua para detecção de anomalias em mensagens SIP. Para tal, é feita uma preparação dos dados, e definida uma função $f(x,s)$ onde s representa uma determinada String pertencente ao universo de Strings S e x uma mensagem SIP. Desta forma, $f(x,s)$ devolve a frequência da String s na mensagem x e a sua importância. Como no protocolo SIP não se pode antecipar todo o tipo de Strings possíveis, os autores optaram por extrair as Strings por sequência de n caracteres na mensagem SIP. Isto é, se n for 4 são recolhidas strings de 4 em 4 caracteres da mensagem SIP. Ao recolher a frequência de cada String numa mensagem x obtém-se a seguinte função:

$$\phi(x) \mapsto (f(x,s))_{s \in S} \tag{2.8}$$

Com base nestes resultados, é possível calcular o desvio da mensagem z em relação ao modelo de normalidade. Para tal, calcula-se a distância média de z em relação aos seus k vizinhos mais próximos, que estão representados em $x_{\pi[i]}$ ao iterar o valor de i.

$$\delta_s(z) = \frac{1}{k} \sum_{i=1}^{k} \left\| \phi(z) - \phi(x_{\pi[i]}) \right\| \tag{2.9}$$

Os pontos com um desvio maior em relação aos seus k vizinhos apresentam uma distância média mais alta do que os pontos que se assemelham mais aos seus vizinhos.

Sabendo que existem diversos tipos de utilização de uma rede, é possível criar mais do que um cluster de actividade normal com diferentes niveis de densidade. Ao avaliar cada ponto tendo apenas em consideração a distância média aos seus vizinhos, estaríamos a classificar como ataque comportamentos que pertencem a clusters normais com uma densidade inferior à média. Desta forma, e tendo em conta que um ponto será classificado a partir da distância média em relação aos seus k vizinhos mais próximos, é importante introduzir um método de normalização que torne o modelo resistente a diferenças de

densidade entre clusters. De forma a introduzir o conceito de densidade na expressão os autores utilizam a média da distância entre todos os vizinhos mais próximos como termo normalizador da média da distância do ponto a ser classificado.

$$\delta_s(z) = \frac{1}{k}\sum_{i=1}^{k}\left\|\phi(z) - \phi(x_{\pi[i]})\right\| - \frac{1}{k^2}\sum_{i=1}^{k}\sum_{j=1}^{k}\left\|\phi(x_{\pi[j]}) - \phi(x_{\pi[i]})\right\| \tag{2.10}$$

Onde o primeiro termo dá ênfase aos pontos longe dos seus vizinhos, enquanto o segundo termo é responsável por impedir que pontos em áreas normais menos densas sejam considerados anomalias por si só. Criou-se um conjunto de treino com k partições de tamanho igual. O modelo de aprendizagem foi treinado com base em $k-1$ partições e testado na partição k. Ao repetir o treino cada partição fica com um desvio definido D_i . O desvio de cada partição é utilizado para calcular o threshold do algoritmo da seguinte forma:

$$t = \frac{1}{k}\sum_{i=1}^{k} max(D_i) \tag{2.11}$$

De forma a treinar e testar o modelo, os autores geraram um conjunto de mensagens SIP com 4428 mensagens normais e 9999 mensagens anormais. As mensagens SIP normais contemplam sequências continuas de mensagens SIP referentes a diálogos de um terminal, assim como outras mensagens obtidas através de um nó onde múltiplos terminais estão conectados. As mensagens SIP anormais foram geradas através de uma ferramenta de simulação de trafego SIP. O conjunto de mensagens foi dividido em conjunto de treino e teste do modelo obtido. O processo de teste é executado através da recolha aleatória de 1000 mensagens SIP presentes no conjunto de treino onde 500 são normais e as outras 500 anormais. Estas mensagens são depois submetidas ao modelo para classificação. Este processo repete-se 50 vezes e os resultados finais são obtidos através da média.

O algoritmo foi testado para varios valores de n (sendo n o número de caracteres de uma String). A figura 2.9 apresenta a relação entre os verdadeiros positivos e os falsos negativos.

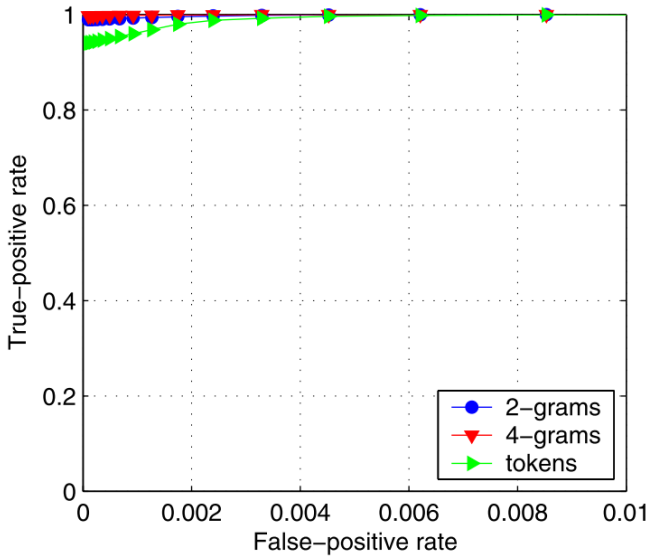


Figura 2.9: Curva de operação da característica do modelo obtido [43]

Como podemos observar na figura 2.9 é possivel obter 97% de veradeiros positivos com 0 falsos negativos para qualquer valor de n. O modelo com $n = 4$ foi inclusive capaz de detectar 99% dos ataques do conjunto de teste. É importante relembrar que os dados de teste são dados nunca antes vistos no processo de treino.

Embora os resultados sejam bastante promissores não nos será possível partir para uma abordagem semelhante, uma vez que o formato dos dados disponibilizados para o projecto proposto não se enquadram com os dados do artigo em questão. No entanto, tendo em conta o potencial do k-nearest neighbour para a detecção de anomalias, na secção 4.1 será testada uma adaptação deste modelo aos dados disponibilizados pela operadora.

2.3.2.3 Utilização de Markov Kernels e SVM para detectar padrões de fraude

Support Vector Machine (SVM) é um algoritmo de classificação que calcula um Hyperplane optimizado entre classes mesmo em dimensões elevadas. O Hyperplane é um plano optimizado, definido entre os dois conjuntos tendo uma distância igual em relação aos Support Vectors dos dois conjuntos. Tendo em conta que nem todos os objectos aplicados podem criar conjuntos linearmente separáveis, pode ser necessária a manipulação de dados para que, ao alterar as dimensões da amostra, seja possível criar um plano que separe os dois conjuntos. Como podemos observar na figura 2.11, em 2 dimensões não é possível optimizar um plano ideal que separe os conjuntos. Contudo, ao aplicar um kernel é possível obter a representação dos objectos em dimensões mais elevadas, possibilitando a criação de um plano que separe os objectos. Há diversos tipos de kernel. Em [10] C.Chen et al. propõe aplicar uma cadeia Markov de alta ordem como kernel, para classificar comportamentos anormais no IMS através de SVM. Uma cadeia de Markov

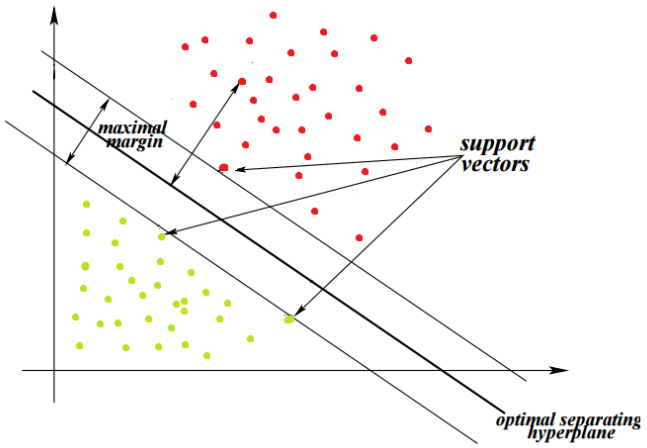


Figura 2.10: Demonstração de um Hyperplane [14]

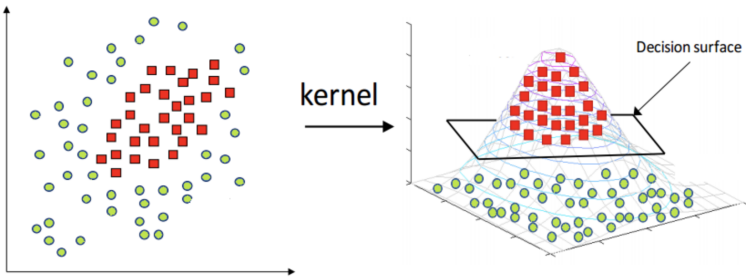


Figura 2.11: Demonstração da acção de um kernel [36]

é um modelo de análise probabilístico para detectar padrões. Antes de aplicar o kernel é necessário extrair o padrão das mensagens SIP. De seguida, os autores usam um gerador de vectores para recolher os dados das mensagens SIP de forma a poder calcular os seus padrões. Para implementar o mecanismo SVM, os autores utilizaram o libSVM, [9] onde, para além de treinarem o algoritmo com a cadeia de Markov previamente descrita, fizeram também vários treinos com outros Kernels. Na fase de treino são introduzidos dados já catalogados para o modelo conseguir aprender. Na fase seguinte, o libSVM gera um modelo para determinar os limites das diferentes classes. Por fim são introduzidos dados não catalogados para serem divididos nas devidas classes. Para obter resultados de exatidão do modelo, os autores recorreram ao svm-predict toll [9]. Com base nos testes, a exatidão do modelo foi de 97.9466% que, segundo os autores, é mais do que suficiente para prever anomalias em IMS. Comparando com outros kernels testados como o linear, polinomial e RBF, a cadeia de Markov obteve resultados superiores com uma taxa de positivos verdadeiros (Recall) a rondar os 80%, o que implica 20% de falsos negativos, com uma taxa de falsos positivos a rondar os 1% ou seja 99% de negativos verdadeiros.

O artigo pretende demonstrar que o uso de uma cadeia de Markov apresenta grandes vantagens para a classificação de comportamentos fraudulentos no IMS, como ataques de BYE e flooding de REGISTERs ou INVITEs. No entanto quando exposto a dados nunca

vistos durante o processo de treino o modelo não foi capaz de obter uma taxa de positivos verdadeiros superior a 80%. Embora seja uma melhoria significativa quando comparado com a taxa de positivos verdadeiros dos kernels mais tradicionais, que variam entre os 30% e 60%, os 80% obtidos com a cadeia de Markov implicam uma taxa de falsos negativos a rondar os 20%. Este valor não confere ao modelo um grau de confiança satisfatório para a equipa que supervisiona a rede IMS da operadora uma vez que não é capaz de detectar 20% das actividades anormais. Assim sendo a credibilidade do modelo seria posta em causa, o que levaria a que a equipa tivesse a tendência para ignorar os alarmes produzidos pelo mesmo.

Para além dos falsos positivos, a aplicação de uma cadeia de Markov como kernel influencia de forma significativa o tempo de treino e previsão do SVM. Em comparação com kernels lineares, polinomiais e RBF, a cadeia de Markov torna o processo de treino 6,5 vezes mais demorado e duplica o tempo de previsão. Na secção 4.3 foi testada a implementação de um SVM com um kernel RBF nos dados disponibilizados pela operadora. Neste teste conclui-se que o tempo de treino e previsão apresentavam uma inviabilidade à implementação do modelo nos sistemas da operadora, uma vez que os tempos de treino e previsão eram demasiado extensos. Tendo em conta que ao aplicar a cadeia de Markov o processo de treino torna-se 6,5 vezes mais extenso e o tempo de previsão duplica, é inviável aplicar este kernel ao problema proposto pela operadora.

2.3.2.4 Classificação de utilizadores com K-nearest neighbour

O K-nearest-neighbour (kNN) é um algoritmo classificador com base na proximidade dos objectos, com bastantes vantagens quando há conhecimento prévio das classes dos conjuntos de treino. Para classificar um objecto, o algoritmo verifica a sua proximidade em relação aos seus k vizinhos e, com base na classe dos seus vizinhos, classifica o objecto em estudo. O cálculo da distância pode ser feito por vários métodos, no entanto a distância euclidiana é a mais usada para estes efeitos. O valor de K é definido à priori e tem grande influência na eficácia da classificação. Ao escolher um k=3, o algoritmo vai comparar os três vizinhos mais próximos de um objecto. Se dos três vizinhos mais próximos dois forem da classe A e um for da classe B, é atribuída a classe A ao objecto em análise. Tendo em conta a importância do valor de K, este deve ser optimizado de forma a ser o valor mais indicado para o conjunto de dados em análise.

Y. Liao et al. [33] propõem utilizar o K-nearest-neighbour para distinguir utilizadores normais de intrusos numa rede a partir de um método de classificação de texto. Para tal os autores usam métodos processamento e classificação de texto de forma a que cada pedido de um cliente é tratado como se fosse uma palavra e o conjunto de pedidos um documento. Ao ordenar todos os pedidos por sessões TCP/IP entre dois utilizadores é possível obter um padrão de sessão normal. Para testar esta hipótese, os autores recorreram ao conjunto de dados 1998 DARPA e a uma rede responsável por simular o tráfego de uma LAN da força aérea, com 38 tipos de ataques. Para efeitos de classificação os autores utilizaram

apenas o nome dos pedidos de sistema que cada cliente enviou (e.g. close, open, execute, etc). Desta forma os pedidos de uma sessão entre dois computadores são guardados sobre um ID que representa a sessão em estudo, de modo a que a frequência de pedidos iguais por sessão represente a propensão de uma sessão ser ou não anormal.

Sendo a abordagem supervisionada foi primeiro necessário catalogar o comportamento de uma sessão normal. A partir do momento que os autores obtêm uma referência para a normalidade de uma sessão é possível calcular a semelhança de cossenos entre a sessão a ser classificada e o valor referência de uma sessão normal. Se o resultado do cálculo da semelhança for 1, significa que a sessão a ser classificada é em tudo semelhante ao modelo de comportamento normal, ou seja, é classificado como normal. Caso o resultado não seja 1, são escolhidos os k vizinhos mais próximos do novo objecto e é calculada a média das semelhanças entre o novo objecto e os k vizinhos. Se o valor da semelhança média for superior a um threshold previamente definido, é atribuída a classe normal ao novo objecto.

Durante as simulações de 7 dias, os autores detectaram 5 dias sem qualquer tipo de ataques. Desses 5 dias foram escolhidos 4, de forma aleatória, para treino e o quinto para teste. Depois de criado o modelo para um comportamento normal, o K-nearest-neighbour pode ser usado para detecção de objectos anormais.

Como a performance do algoritmo depende do valor de k, os autores fizeram testes com valores de k entre 5 e 25 e concluíram que a rapidez da detecção era melhor quando k=10. Com k=10 o algoritmo detectou 10 dos 35 ataques sem falsos positivos com um threshold de 1. No entanto, ao aplicar um threshold de 0.72 foi possível atingir os 100% de ataques detectados com uma taxa de falsos positivos de 0,44%(23 falsos alarmes em 5285 sessões normais)

Embora o artigo não aplique o modelo desenvolvido a mensagens SIP, é possível traçar um paralelismo entre os pedidos de cliente (e.g. close, open, execute, etc) e os cabeçalhos das mensagens SIP(e.g. BYE, INVITE, REGISTER, etc) numa sessão entre dois terminais. Assim sendo o mesmo método poderá ser usado para a detecção de comportamentos anormais no estabelecimento de chamadas VoIP. Tendo em conta que o projecto proposto parte pela detecção de pedidos REGISTER fraudulentos este método torna-se apenas aplicável nos próximos passos do projecto onde se espera expandir o modelo a pedidos de INVITE.

Os resultados obtidos pelos autores são promissores, no entanto, o alto volume de mensagens SIP da operadora pode tornar o modelo inviável para detecção em tempo real. Os autores não apresentam qualquer informação em relação ao tempo de treino, tempo de previsão e complexidade computacional do modelo. Para além das condições adversarias à implementação em tempo real, os autores admitem que uma intrusão na rede que não contemple nenhuma anomalia na frequência dos pedidos de sistema, como por exemplo o abuso de processos normais, não será detectada pelo modelo. Uma intrusão pode também não ser detectada se um atacante não finalizar a sessão.

2.3.3 Algoritmos de Detecção de Flooding

Como foi referido na secção 2.2.1 as comunicações entre o terminal e a rede IMS são feitas através de um P-CSCF, onde o terminal é livre de enviar os pedidos SIP que bem entender. Esta liberdade dá aso a ataques por flooding, onde o utilizador fraudulento pode enviar uma grande quantidade de mensagens SIP de forma a que o servidor não tenha capacidade de processamento para responder. Estes ataques são utilizados para DoS. No entanto, como já foi explicado no capítulo 1 secção (1.2), o registo de um terminal fraudulento é conseguido através de varias tentativas de autenticação. Como tal, os algoritmos de detecção de flooding podem ser bastante úteis para detectar os atacantes antes de estes terem oportunidade de começar a sua operação.

2.3.3.1 Detecção de flooding por adaptive threshold

O Adaptive Threshold é um algoritmo simples que é capaz de detectar uma variação na média de um acontecimento. Para tal, é primeiro calculada a média móvel de um acontecimento, num intervalo de tempo, da seguinte forma:

$$\mu = \beta\mu_{n-1} + (1-\beta)x_n \tag{2.12}$$

Onde x_n é o valor do acontecimento no intervalo de tempo n, os μ são as médias móveis no intervalo n e $n-1$ e, por fim, β representa o peso associado a cada uma das médias. O threshold para o desvio da média é dado por:

$$(\alpha+1)\mu_{n-1} \tag{2.13}$$

e, se for verificado k vezes, o acontecimento é considerado anormal.

M. Akbar et al [3] utilizaram o Adaptive Threshold para detectar ataques de flooding num P-CSCF, onde x_n representa o número de INVITES numa janela temporal de 10 segundos. Para tal, foi utilizado um gerador de tráfego SIP, [1] onde foram recriados três cenários. No primeiro variaram a quantidade de tráfego recebida pelo P-CSCF, no segundo cenário a intensidade dos ataques é variada e, por fim, variaram a duração dos ataques. Dentro de cada cenário foram feitas várias simulações, alterando os parâmetros que o cenário se propunha a testar. Para o primeiro cenário, foram considerados três níveis de tráfego num P-CSCF, com 500 chamadas/min, 750 chamadas/min e 1000 chamadas/min, tudo com uma duração de 60 minutos. Os ataques do segundo cenário têm todos uma duração de 1 minuto e a média de chamadas varia desde as 50 as 500 por segundo. Por fim, no ultimo cenário, os ataques têm uma duração de 10 minutos e a média de chamadas varia também desde as 50 às 500 por segundo. Os dados referentes aos ataques foram distribuídos pelos dados considerados normais e foram calculados os números de INVITE, ACK, 200 OK e BYE enviados e recebidos numa janela temporal de 10 segundos.

Depois de efectuar os testes os autores concluem que a performance do algoritmo vai piorando à medida que o tráfego normal aumenta, com base no facto de que para 2% de falsos positivos a taxa de detecção é de 77% em ataques de fraca intensidade e 100% em

ataques de grande intensidade. Assim sendo, a taxa de falsos positivos diminui com a intensidade dos ataques, sejam eles duradouros ou repentinos. No entanto, o algoritmo é mais eficaz quando os ataques são repentinos, atingindo 100% de detecção com 2% de falsos positivos para ataques durante 1 minuto, com uma média de 100 a 300 chamadas por segundo.

O modelo apresentado é útil para ataques esporádicos de flooding, no entanto existe uma grande dependência no volume de comportamento normal que quando em excesso prejudica a exactidão do modelo. Para além da dependência do volume do comportamento normal o modelo é incapaz de detectar ataques prolongados no tempo, uma vez que se baseia nas médias do número de pedidos que são facilmente influenciáveis com ataques prolongados. Por fim, o modelo é capaz de alertar que existe um ataque de flooding mas não é capaz de identificar a origem do ataque.

2.3.3.2 Detecção de flooding por Cumulative Sum

Cumulative Sums (CUSUM) é um algoritmo bastante usado na deteção de anomalias, capaz de detectar desvios da média de uma série de acontecimentos e de as acumular num somatório. O cálculo da média é feito pela equação 2.12. Tendo em conta que uma série de mensagens não é estacionária, é necessário converter a mesma para uma sequência estacionária através de:

$$\widetilde{x_n} = \frac{x_n}{\mu_{n-1}}, n \geq 1 \tag{2.14}$$

Por sua vez a (CUSUM) é dada por:

$$y_n = max(0, y_{n-1} + \widetilde{x_n} - a), y_0 = 0$$

Y. Rebahi et al. [42] propõem utilizar este algoritmo para detectar ataques de flooding num P-CSCF. Para o testar recorrem a dados SIP de uma operadora de serviços SIP com 30 minutos de tráfego, que depois de filtrados apenas têm informação referente aos pedidos INVITE. Esses 30 minutos de informação são divididos em intervalos de 10 segundos e, por cada um destes intervalos, é estudado o número de pedidos INVITE. Foram considerados dois tipos de ataque. O primeiro tipo de ataque atinge imediatamente a taxa máxima de pedidos INVITE estipulada para a simulação, que representa aproximadamente 8,8 INVITEs por 10 segundos. O segundo tipo de ataques é composto por duas fases. Na primeira fase a taxa de pedidos INVITE vai aumentado gradualmente até atingir o seu máximo. Ao atingir o máximo, começa a segunda fase do ataque responsável por manter a taxa máxima durante 5 minutos, reduzindo depois os INVITEs para 0. Cada tipo de ataque tem a duração de 10 minutos, ou seja 60 intervalos de tempo. Segundo os autores, os resultados foram favoráveis, uma vez que o algoritmo detecta os ataques do primeiro tipo em apenas 10 segundos, dando tempo ao sistema para reagir. Para ataques do segundo tipo o algoritmo demora 90 segundos até conseguir detectar a anomalia. No entanto, continua a ser um bom resultado, tendo em conta a duração total do ataque.

Este algoritmo foi também utilizado por M. Akbar et al. [3] para testar a detecção de ataques de flooding a partir da media móvel de pedidos INVITE. Os testes foram efectuados com as características experimentais referidas na secção 2.3.3.1 com um intervalo de 10 segundos como parâmetros do algoritmo. Os resultados indicam que o CUSUM apresenta melhores resultados em ataques de baixa intensidade(25 chamadas por segundo), em comparação com ataques de alta intensidade (500 chamadas por segundo), uma vez que, para ataques de baixa intensidade, atinge uma taxa de detecção de 100% em comparação com ataques de alta intensidade, onde atinge os 83% de ataques detectados. A performance do algoritmo em relação à duração dos ataques é significativamente melhor em vários ataques de curta duração (1 minuto) em relação a ataques prolongados (10 minutos). Os resultados da resposta do algoritmo a ataques de grande intensidade dos dois autores não são contraditórios uma vez que à taxa máxima de INVITEs dos testes de Y. Rebahi et al. [42] é inferior a taxa mínima dos testes de M. Ali Akbar et al. [3].

No entanto, á semelhança da detecção de Flooding por Adaptive Threshold, o modelo torna-se demasiado dependente de médias de comportamento passado que são influenciadas com ataques prolongados, e não é capaz de identificar a origem do ataque.

2.3.3.3 Detecção de flooding por Hellinger Distance

A Hellinger Distance (HD) é um algoritmo usado para medir a distância entre duas distribuições probabilísticas. Para obter HD é necessário ter como parâmetros de entrada distribuições probabilísticas de dimensões iguais. Dadas duas distribuições probabilísticas $P : (p_1, p_2, ..., p_n)$ e $Q : (q_1, q_2, ..., q_n)$, podemos calcular a Hellinger distance através de:

$$H^2(P, Q) = \frac{1}{2} \sum_{i=1}^{n} (\sqrt{p_i} - \sqrt{q_i})^2 \tag{2.16}$$

Se o resultado for 1 então as distribuições são totalmente diferentes, caso seja igual a 0 as distribuições são idênticas. É possível utilizar a Hellinger Distance para detecção de anomalias, se uma das distribuições for um modelo estatístico das condições normais.

J.Tang et al. [46] propõem usar a Hellinger Distance para detectar anomalias em mensagens SIP, mais especificamente para detectar ataques de flooding a partir do comportamento normal dos pedidos INVITE. Para tal, os autores usam uma Sketch Data Structure como uma forma de organizar dados probabilísticos. Neste modelo, cada objecto é composto por uma chave k_i e o seu valor associado v_i. Desta forma, a chave será o número de pedidos INVITE e o valor o endereço SIP. Uma Sketch Data Structure é uma tabela $H \times K$ onde as K entradas de cada linha são usadas para obter uma distribuição probabilística que servirá de parâmetro para o cálculo da HD. Uma distância grande entre a distribuição normal e uma distribuição em teste é um indicador de que existe uma anomalia, que pode ser reportada se a distância for superior a um dado valor de threshold. Como o comportamento normal está em constante mudança, os autores optaram por usar um threshold dinâmico. Partindo do princípio que os dados de treino não têm nenhum

ataque, é feito o modelo probabilístico referente ao comportamento normal. De seguida, faz-se o mesmo para o conjunto de objectos em teste e calcula-se a HD por cada linha dos dois conjuntos. Se o resultado for superior ao threshold dinâmico, o comportamento é considerado anormal.

Para testar o algoritmo, os autores usaram tráfego que consideraram normal e misturaram tráfego de ataques de flooding que recolheram de um ataque anterior. As taxas de INVITEs variaram entre os 35 por segundo e os 500 por segundo com a duração de 30 segundos. Cada teste foi executado 30 vezes e para todos as taxas de pedidos INVITE por segundo o algoritmo detectou 100% dos ataques.

2.3.4 Behaviour Profiling

Behaviour Profiling é uma técnica que permite uma análise de comportamentos diferenciada, isto é, uma comparação entre o comportamento normal ou anormal de um indivíduo, que é não apenas feita com base nos conhecimentos de todos os indivíduos, mas sim com base nos conhecimentos comportamentais do indivíduo em estudo. Nesta secção será apresentado um exemplo de como é possível analisar o comportamento individual dos utilizadores de uma rede de telecomunicações.

2.3.4.1 User Profiling para detecção de Toll Fraud

Como já foi referido, Behaviour Profiling é uma técnica que permite uma análise de comportamentos individuais para comparação com comportamentos futuros do mesmo indivíduo. A. Wiens et al. [49] propõem utilizar as características de chamadas de cada utilizador, como a duração de cada chamada e o número de chamadas por hora, para traçar o perfil de utilização de um cliente da operadora. Para tal, cada utilizador teria dois perfis, que representam o comportamento do mesmo no presente e no passado. Cada um dos perfis tem informação de chamadas efectuadas pelo utilizador numa janela temporal. Com base no número e duração das chamadas, o desvio padrão e a média destas características representam os dados para a decisão de comportamentos anormais. O perfil do passado contém a média e o desvio padrão de cada característica (duração e número de chamadas), enquanto que o perfil atual a ser analisado apenas tem a média de cada característica na última hora. Com base nos dois perfis, são calculadas as relações entre as mesmas características para ter uma noção das diferenças de utilização. Com base nas relações, é calculado um valor que irá representar a anormalidade do comportamento através de:

$$Limit = (Mean + Str \times r) \times R + a \tag{2.17}$$

Sendo a uma constante para que os utilizadores sem chamadas não sejam reconhecidos por $Limit = 0$ e r um factor de assimetria. Se o resultado ultrapassar o threshold previamente definido, o comportamento do perfil atual é considerado fraudulento, caso contrário, os dados desse perfil são introduzidos no perfil de utilização normal desse utilizador. Para além de perfis de utilizadores, o autor propõe também perfis para os destinos

que dependem das mesmas características dos perfis de utilizadores no entanto, apenas tomam em consideração chamadas recebidas. Para testar o algoritmo sugerido, os autores recorreram a tráfego real de uma operadora de telecomunicações, que corresponde a duas semanas com 3.5 milhões de chamadas. Na primeira semana foi considerado que não havia ataques, logo esses dados foram usados para iniciar o comportamento do utilizador. Foi estabelecido um threshold de 99% do quantil de chamadas do perfil de comportamento do utilizador. Os autores concluíram que o algoritmo detectou 90% dos ataques com 1.22% de falsos positivos.

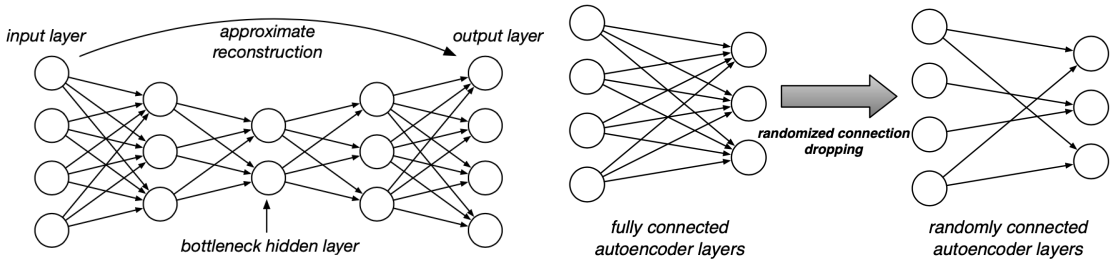
Para além da exactidão de 90% não ser capaz de garantir um alto grau de confiança, o modelo apresenta incompatibilidades com a detecção em tempo real. Sabendo que a operadora detêm milhões de usuários e sabendo que a detecção deve ser feita em tempo real, torna-se demasiado dispendioso traçar o perfil estatístico de utilização de cada utilizador e destino e, ao mesmo tempo, classificar utilizações futuras num curto espaço de tempo.

2.3.5 Redes Neuronais

As redes neuronais são sistemas de "neurónios"interligados de forma a serem capazes de fazer associações, de forma a descobrir padrões ou aprender o esquema de padrões através de dados de treino. Para oferecerem resultados eficazes, as redes neuronais são primeiro alvo de treino, de forma a que atinjam um equilíbrio no qual o mesmo objecto de input tenha sempre o mesmo output. O treino de redes neuronais é feito essencialmente por retropropagação do erro (Backpropagation), onde o erro do output é propagado pelas camadas que originaram esse output, de forma a que estas adaptem os seus pesos como objectivo de minimizar o erro.

2.3.5.1 Detecção de outliers com Autoencoder Ensembles

Um autoencoder é um tipo de rede neuronal capaz de reduzir as dimensões dos dados de entrada. Para tal, a primeira camada e a última camada necessitam de ter o mesmo número de neurónios e a arquitectura da rede tem de ser simétrica, de forma a que o número de neurónios das camadas centrais seja inferior o número das camadas periféricas, como se vê na figura 2.12 (a). O objectivo desta rede é treinar o output de forma a que



(a) Esquema de uma rede de autoencoder [11]

(b) Esquema de uma rede com conexões aleatórias [11]

Figura 2.12: Esquema de autoencoder e das redes neuronais de teste

consiga reconstruir a informação que lhe foi introduzida depois de decomposta. Primeiro a informação é introduzida na camada de input e, de seguida a rede reduz as dimensões da informação de input até que fique apenas representada pelos neurónios da camada bottleneck. O resto da rede é depois treinada para reconstruir a informação sintetizada, de forma a chegar novamente aos dados fornecidos à rede no input. No final, o input é comparado com o output e é estabelecido um erro de reconstrução. Desta forma, os autoencoders podem ser bastante úteis para deteção de anomalias, uma vez que se a rede não estiver treinada para reconstruir estes objectos, o erro de reconstrução vai ser superior ao erro de reconstrução dos objectos normais.

J. Chen et al [11] propõem utilizar autoencoders para detectar outliers, partindo do princípio que os dados de treino representam objectos normais. Para resultados mais robustos, os autores propõem usar também métodos de Ensemble Learning onde se pretende combinar os resultados de diferentes algoritmos para combater os comportamentos tendenciosos que alguns desses algoritmos podem ter em relação a certos conjuntos de dados. Os resultados dos diferentes algoritmos são tidos em consideração para formar um resultado geral, que pode representar uma média dos resultados. Como o método a ser utilizado requer uma rede totalmente ligada, e tendo em conta que para obter bons resultados os outros métodos devem ser diversificados, os autores utilizaram redes com conexões aleatórias como algoritmos de Ensemble Learning. Partindo do mesmo princípio que a diversificação de métodos é capaz de fornecer dados mais robustos, os autores utilizaram duas funções de activação na mesma rede. Foi utilizada a função sigmoid para as camadas das duas extremidades, e a Rectified Linear (ReLU) para as restantes camadas. Assim, segundo os autores, as diferentes funções de ativação em camadas diferentes são capazes de balançar as vantagens e desvantagens de cada uma. Para além das redes de ensaio serem compostas de ligações aleatórias, estas treinam também subconjuntos de dados selecionados do conjunto principal de forma aleatória. O erro de cada rede de ensaio é comparado com a rede dos autoencoders e calcula-se um valor de outlier por cada objecto estudado. A partir da média dos erros normalizada de 1 a 0, um ponto é considerado normal ou não. Os autores usaram também um método de aprendizagem adaptativo chamado RMSprop, [13] onde a ideia principal é usar um gradiente móvel em t, a partir do valor da média da magnitude em $t-1$. Para combater um problema comum no treino de redes neuronais conhecido como Local Optimum onde a rede fica optimizada com uma solução ótima para um conjunto de soluções vizinhas, os autores usaram pesos já inicializados, de forma a que o início do treino seja o mais ideal para chegar a uma optimização global. Tendo em conta que no início do treino o gradiente não necessita de ser demasiado exato, os autores aumentaram de forma exponencial os dados do conjunto de treino a cada iteração de forma a tornar o processo de optimização mais eficiente.

Para testar o algoritmo, os autores recorreram a 9 conjuntos de dados do UCI Machine Learning Repository com 2.2% a 9.6% de outliers. Esses conjuntos de dados foram também sujeitos a outros algoritmos de detecção de outliers como Hawkins [21], LOF [5],

HiCS [27] e um método espectral não linear [44]. Os autores concluíram que os autoencoders ultrapassam os restantes algoritmos na maioria dos conjuntos de dados, chegando a ter resultados superiores aos outros algoritmos em 7 dos 9 conjuntos de dados testados.

Para além dos resultados superiores quando comparado com outros modelos conhecidos, os autoencoders apresentam a vantagem de se tornarem independentes aos diferentes tipos de utilização anormal. Desta forma, se o comportamento normal não se alterar, o modelo é capaz de detectar actividades anormais novas. Por este motivo, na secção 4.6 foi treinado e testado um autoencoder com os dados disponibilizados pela operadora

2.3.5.2 Detecção de outliers com Generative Adversial Networks (GANs)

Uma GAN é composta por duas redes neuronais que realizam uma competição em min-max entre dois componentes adversários: um gerador e um descriminador. O gerador aprende o padrão do conjunto de dados ao qual é submetido e, através de ruído como parâmetro de entrada, cria variações nos dados, produzindo dados o mais semelhantes possíveis aos dados de treino. Por sua vez, o descriminador tem como papel reconhecer quais são os dados do conjunto de treino, ou seja reais, e os dados gerados pelo gerador.

Através deste princípio, Y. Liu et al. [35] sugerem aplicar este modelo de aprendizagem sem supervisão à detecção de outliers. Para tal ,o gerador vai produzir potenciais outliers com base nos dados reais, enquanto o descriminador identifica os outliers produzidos pelo gerador, de forma a aprender a fronteira que separa os outliers dos objectos normais. No entanto, os autores identificaram um problema que reside na possibilidade dos outliers estarem no meio, ou muito perto dos objectos normais. Isto levaria o gerador a produzir outliers que não se assemelham com os outliers escondidos nos objectos normais. Para solucionar este problema, os autores sugerem a criação de vários geradores de forma a espalhar o conjunto de treino e dispersar os outliers concentrados nos objectos normais. Todo este processo tem como objectivo principal criar uma noção da distribuição dos dados normais, que não estão previamente classificados. O conjunto de dados é divido em k subconjuntos e são fornecidos a cada um dos geradores com base no resultado de um teste prévio em relação à sua distribuição. Para testar o algoritmo os autores recorreram 14

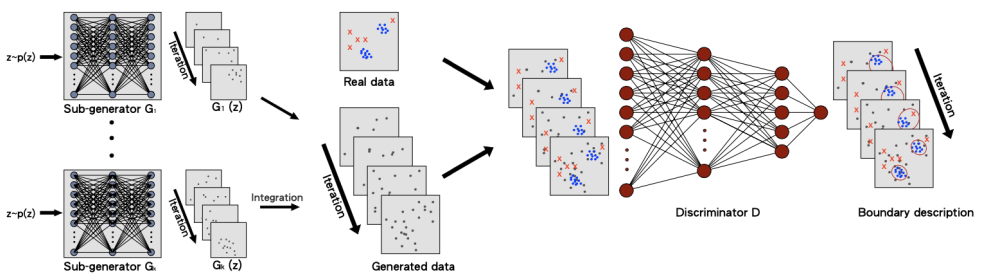


Figura 2.13: Esquema de uma rede Generative Adversarial com múltiplos geradores [35]

conjuntos de dados que foram testados em vários algoritmos de detecção de outliers como o LOF [5], kNN, K-means e, redes Generative Adversarial com um gerador, entre outros.

Nos testes realizados, o algoritmo proposto foi o algoritmo com melhores resultados, tendo taxas de detecção superiores aos restantes em 6 dos 14 conjuntos de dados, sendo o algoritmo com melhor a melhor média de detecção. Através dos testes foi também possível concluir que o algoritmo com vários geradores apresenta melhores resultados que uma GAN com apenas um gerador.

Capítulo 3

Processamento e Análise de dados

3.1 Estratégia Inicial

Como já foi referido, o objectivo da dissertação é proteger sistemas IMS na eventualidade de ataques fraudulentos. Para tal, será proposto um algoritmo que, com base em dados catalogados, será capaz de distinguir os comportamentos fraudulentos dos comportamentos considerados normais nos sistemas de uma operadora. O modelo tem como pressupostos a existência de uma grande quantidade de dados referentes às interacções SIP no sistema IMS e a necessidade de detectar os ataques antes que eles tenham sucesso. Assim é porposta a utlização de algoritmos de aprendizagem que serão treinados para classificar comportamentos anormais observados anterioremente nos dados disponíveis. Desta forma, o sistema da operadora estará constantemente protegido, sem necessidade de supervisão, para possíveis ataques já conhecidos.

3.2 Análise dos Dados

Como abordagem inicial,são analisados os dados de registo SIP nos Session Border Controllers da operadora, uma vez que esses dados representam o resultado da primeira interacção entre o utilizador e a rede IMS. A partir da análise a esses dados são definidos quais os os comportamentos de um IP suspeito, a fim de reduzir a informação a ser classificada. A filtragem dos IPs potencialmente fraudulentos é um processo de extrema importância, uma vez que a rede da operadora interage com milhões de utilizadores. Sem uma filtragem prévia (através do IP) das interacções SIP a classificar, seria necessário analisar todos os pedidos de registo. Este processo seria extremamente exaustivo e, colocaria em causa um dos objectivos da tese, a detecção em tempo real.

A partir desta análise dos dados são testados vários algoritmos para a classificação

de pedidos de registo SIP. Foram considerados vários algoritmos referidos no capítulo 2 (e.g. knn, k-means, SVM e Autoencoder), de forma a analisar os resultados de cada um em termos de precisão, exactidão e tempo de resposta. Estes resultados foram tidos em conta para eliminar algoritmos que não se adaptam ao conjunto de dados disponibilizado. O projecto foi desenvolvido com apoio da equipa responsável pelo serviço de voz da operadora, de forma a que fosse possível desenvolver o algoritmo de acordo com as necessidades da equipa.

3.2.1 Definição de Comportamento Suspeito de um IP de Origem

O IMS da operadora, onde os dados foram recolhidos, respeita o esquema tradicional do IMS descrito na secção 1.2.1. No entanto, com o intuito de limitar a exposição da rede a terceiros, a operadora dispõem de dois Session Boarder Controllers (SBC), que têm como objectivo estabelecer uma fronteira entre os serviços disponíveis no IMS e os utilizadores que, para terem acesso aos mesmos devem fazer o seu pedido de registo para os SBCs. Ao receber o registo, estes encaminham o pedido pelos meios próprios.

A operadora organiza os seus utilizadores e serviços através de diferentes domínios. Os diferentes domínios têm como objectivo separar o acesso aos diferentes serviços prestados através do IMS. Como exemplo da organização por domínios podemos pensar na separação necessária entre serviços residenciais e corporativos. Desta forma cada pedido de um utilizador é orientado para o domínio que o serviço requer através do IP de destino.

Um pedido de registo contém vários campos de interesse no entanto, na análise dos dados para identificação de comportamentos fraudulentos foram considerados os seguintes campos: IP de origem, número de utilizadores registados por IP de origem, e número de tentativas de registo por IP de origem.

O IP de origem representa a entidade que irá ser classificada como fraudulenta ou não. Assim sendo, toda a análise de fraude será feita tendo em conta o comportamento do conjunto das interacções que esse IP estabelece com o SBC.

O volume de tráfego a analisar é bastante considerável. A titulo de exemplo foram recolhidos dados referentes a uma semana de pedidos de registo no mês de Julho. Durante esta semana os SBCs da operadora foram alvo de cerca de 904,5 milhões de pedidos de registo executados por 17,9 milhões de diferentes IPs de origem. Tendo em conta que o processo de detecção de fraude deve ser o mais próximo possível do acontecimento e, considerando um intervalo de 15 minutos como atraso aceitável, a arquitectura teria de ser capaz de extrair os dados da base de dados e classificar 1,3 milhões de pedidos de registo em menos de 10 minutos. Com base nas ferramentas à disposição da operadora este processo não seria possível. Assim sendo, é necessário fazer uma análise prévia de forma a referenciar os IPs com comportamentos suspeitos, reduzindo assim a lista de IPs a classificar a cada 15 minutos.

Para definir o comportamento normal e suspeito de um IP, foi necessário analisar a relação que existe entre o número de pedidos e o número de contas que um IP pretende

registar. Como foi descrito na secção 1.2.2 o registo de um terminal é feito através de uma sequência de pedidos REGISTER. Desta forma, um registo normal requer mais do que um pedido por cada conta a registar. Por outro lado, é também comum que o mesmo IP registe várias vezes mais do que um terminal.

Ao analisar a relação contas/pedidos de diversos IPs, foi notório que os IPs fraudulentos apresentam um número de pedidos mais próximo do número de contas quando comparados com os IPs de confiança. Esta relação deve-se ao método de ataque, onde os atacantes executam mais pedidos do que o normal e percorrem um dicionário de contas sem sucesso. Desta forma os atacantes raramente executam vários pedidos de registo por cada conta, ao contrario dos IPs com comportamentos normais que tendem a registar o mesmo terminal entre 8 a 120 vezes no espaço de uma hora. Sabendo que os atacantes percorrem uma lista de números de contas por eles conhecidas fazendo um pedido por cada conta, a relação contas/pedidos aproxima-se de 1.

Para seleccionar os IPs que se desviam do comportamento normal, foi criando um índice de suspeita que representa o número de diferentes contas sobre o total de pedidos desse IP. Assim sendo, quanto mais próximo de 1 for o índice maior é a probabilidade de fraude. Para além da relação entre o número de contas e pedidos de um IP, definiu-se como comportamento anormal que um IP execute pedidos de registo para mais do que 30 contas distintas. Este método de filtragem reduz o número de IPs a analisar de forma significativa, visto que em vez de classificar os pedidos de registo referentes à totalidade dos IPs que interagem com as SBCs passa a ser apenas necessário classificar entre 10 a 30 IPs por hora.

A filtragem dos IPs normais através de uma lista de confiança, onde estariam referenciados todos os IPs que a operadora considera serem de confiança por serem da operadora ou de uma origem conhecida, foi considerada. No entanto essa hipótese foi posta de parte, uma vez que os terminais com IPs de confiança podem ser comprometidos e utilizados para actividades fraudulentas, como já aconteceu.

3.3 Catalogação dos Dados

No âmbito do estudo e treino dos classificadores, foram recolhidos dados referentes aos registos de utilizadores das SBCs em função do IP de origem. Sabendo que pretendemos solucionar o problema com um modelo de classificação, é necessário que o conjunto de treino esteja devidamente catalogado. Este conjunto de dados foi obtido através da análise do comportamento da rede durante um longo período de tempo, a fim de detectar ataques e, extrair os dados que dai resultaram. Para além de dados de ataque, foram também extraídos dados referentes a IPs com um comportamento normal e não ameaçador, de modo a formar o conjunto de dados final. Os dados foram recolhidos entre dia 20 de Abril e 16 de Junho de 2020, dando origem a 5477469 entradas fraudulentas e 5061245 normais, que origina um total de 10538714 entradas no conjunto de dados final. Cada entrada do

conjunto de dados corresponde a uma mensagem SIP com cabeçalho REGISTER e contém uma grande quantidade de campos com valores de diferentes características.

Depois de uma análise de correlação e de relevância para o problema que se pretende solucionar foram seleccionadas 7 características por cada entrada. As 7 características são: o term code de resposta ao pedido REGISTER, o porto de origem, o número de conta, o tempo de registo, o número de retransmissões associadas, o status final e, a duração entre o pedido e a resposta do servidor. A característica term code representa o tipo de cabeçalho da mensagem SIP de resposta ao pedido REGISTER. Existem várias possibilidades de resposta a um pedido REGISTER, sendo que em caso de sucesso o cabeçalho da mensagem de resposta SIP é 200 OK. Esta mensagem informa o terminal que solicita o registo que o processo foi concluído com sucesso. Caso o registo não tenha sucesso existem várias possibilidades de resposta, no entanto as mais frequentes serão UNAUTHORIZED, 403 FORBIDDEN ou 404 NOT FOUND. A característica status representa o motivo pelo qual a troca de mensagens SIP se deu como concluída. Os motivos de término da conversação podem ser o sucesso de registo, erros no processo de registo, ou término porque um dos intervenientes da conversação ultrapassou o tempo máximo de espera por uma resposta. A análise de correlação foi capaz de reduzir a quantidade de características importantes para a classificação, visto que a grande maioria das características disponíveis no conjunto de dados da operadora têm informações referentes a pedidos de INVITE (ex: porto media, RTP node, tempo de conversação, etc). Desta forma os pedidos REGISTER apresentam estes campos com o valor -1. Foram também excluídos todos os campos com valores capazes de identificar um determinado utilizador, destino final ou tipo de equipamento usado, de modo a impedir que o modelo se adaptasse a características do atacante. O objectivo do classificador é classificar comportamentos e não um determinado utilizador, assim sendo toda a informação referente ao utilizador por detrás do comportamento deve ser invisível ao modelo para evitar overfiting em relação a IPs ou modelos de equipamentos (ex: Yealink, OneAccess, etc). O IP de origem funciona apenas como identificador de um dado conjunto de pedidos REGISTER, o IP em si não deve ser uma característica com influência no processo de classificação, uma vez que é facilmente substituído.

O conjunto de dados contém mais de 1 milhão de diferentes IPs, dos quais 11 são IPs fraudulentos. Os IPs de ataque foram identificados pelo método descrito na secção 3.2.1 e confirmados como fraudulentos pela equipa de supervisão da rede. A diferença entre o número de IPs fraudulentos para IPs normais deve-se ao facto de não haver situações de ataque em abundância e também ao facto de cada ataque produzir uma grande quantidade de dados.

O conjunto de dados reunido contém pedidos de 1,7 milhões contas, sendo que cerca de 400 mil dessas contas são registadas através dos 11 IPs de origem fraudulenta. Desta forma a média de contas registadas por IP fraudulento é superior a 35 mil enquanto que, a média dos IPs normais situa-se perto de uma conta por IP.

O intervalo de tempo de cada registo é referenciado pela operadora e, está disponível no conjunto de dados como já foi referido. Em média, o tempo de registo de um pedido

proveniente de um IP normal aproxima-se dos 0,5 segundos enquanto que, para IPs fraudulentos, a média é de 0 segundos, uma vez que estes pedidos raramente têm sucesso. A discrepância temporal também é notória quando se compara o tempo de resposta entre o terminal e a SBC. Para IPs normais, este período tem uma duração média de 1 segundo, no entanto para IPs fraudulentos a média situa-se nos 15 segundos.

3.3.1 Processamento dos Dados

Como foi descrito na catalogação dos dados, a estratégia de detecção de ataques passa por analisar a distribuição dos 7 campos do conjunto de dados referentes a um IP de origem, sendo que as diferenças entre os comportamentos normais e anormais estão bastante nítidas na distribuição dos valores em relação a uma média. Tendo em conta que, o objectivo é detectar padrões de comportamento por IP a partir do comportamento da distribuição normal dos seus pedidos REGISTER, procedeu-se à normalização das variáveis quantitativas através do Z-Score. O Z-Score exprime a distância em número de desvios padrão entre o valor de um ponto de uma distribuição e a sua média. Desta forma é possível obter uma representação da variação das várias características como, a variação do número do porto, número de conta ou mesmo, a variação do tempo de registo que um IP dispõem no seu conjunto de dados.

Um exemplo prático para entender a utilidade da normalização para esta abordagem pode ser descrito através do comportamento do porto de origem de cada mensagem. Depois de analisar vários ataques é seguro assegurar que o atacante varre os portos de origem, não fazendo mais do que um certo número de pedidos por cada porto. A figura 3.1 apresenta o gráfico com o número de tentativas de registo por porto de origem por parte de um IP fraudulento. Como podemos observar na figura 3.1, o atacante percorre o número de porto de origem nunca usando mais do que 5 vezes o mesmo porto.

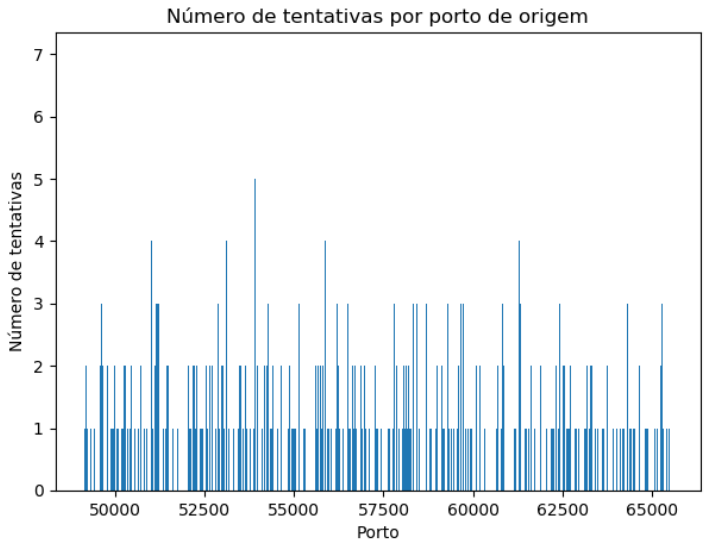


Figura 3.1: Análise do número de tentativas por porto de origem em situação de ataque

Este comportamento leva a que a dispersão do número de portos, durante um ataque, seja bastante considerável. Ao contrário de um atacante, um utilizador normal não varre portos de origem, mudando de porto uma ou duas vezes no máximo. Desta forma, ao analisar os dados por IP é notório que o valor da média dos portos de origem utilizados por um utilizador normal é muito próxima do número de porto mais utilizado. Assim sendo, o desvio padrão do número de porto dessas entradas será zero ou um valor muito próximo de zero. Para um conjunto de pedidos de registo feitos por um IP fraudulento, o desvio padrão do número de cada porto vai ser diferente de 0, uma vez que existe uma maior dispersão dos portos. A diferença entre os valores do desvio padrão de um conjunto de dados retirado de um IP de confiança quando comparado com um IP fraudulento pode ser analisada na figura 3.2. Como podemos vez, para um IP normal a variação de portos de origem é nula. Assim sendo, a média dos portos será o número do porto usado e, o z-score será 0, de forma a que nem é possível observar a área referente ao valor dos portos.

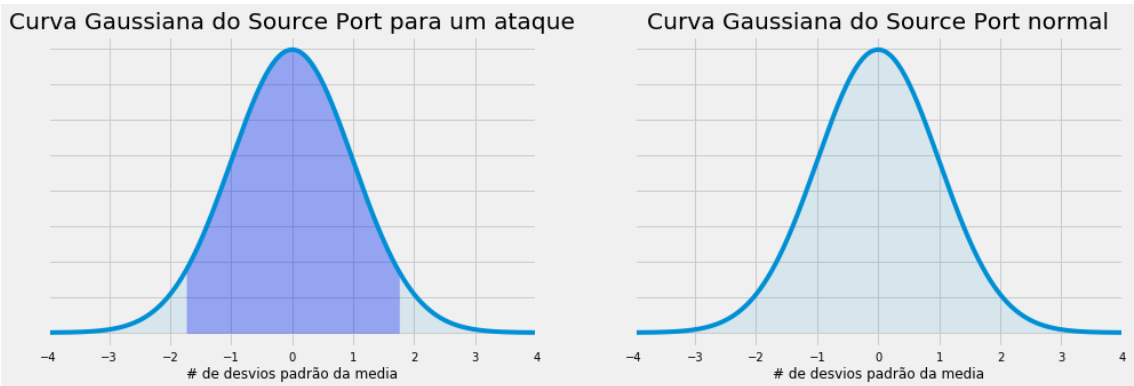


Figura 3.2: Análise de desvio padrão do porto de origem em situações normais e de ataque

O mesmo acontece com o número de conta (ID) a ser registado. Como já foi explicado, durante um ataque o número de diferentes contas a ser registadas é próximo do número de tentativas. Sabendo também que, durante um ataque os atacantes têm tendência a percorrer os números de contas (IDs) de forma sequencial, a distribuição tem tendência a ser mais concentrada em torno da média. Pelo contrário, numa utilização normal um IP está responsável por registar diversas contas que na maioria das situações têm valores com uma dispersão irregular, como é notório na figura 3.3.

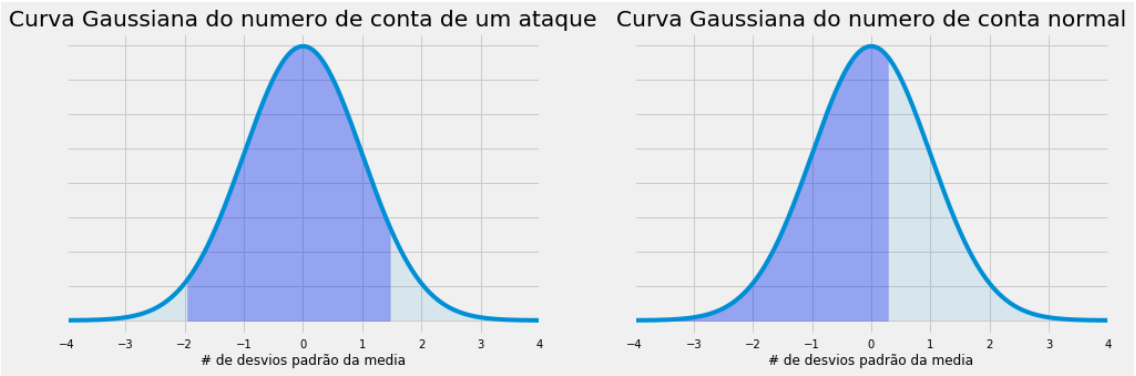


Figura 3.3: Análise de desvio padrão do número de conta em situações normais e de ataque

As retransmissões por cada pedido é também uma variável com um comportamento claramente distinto em situações normais e de ataque. Ao contrário dos comportamentos normais, numa situação de ataque não há retransmissões dos pedidos.

Com base na evidência que as variáveis numéricas devem ser avaliadas tendo em consideração a amostra onde se encontram, o cálculo do Z-Score foi feito com base no universo de cada IP. Para tal foram criados sub conjunto de dados com todos os pedidos de registo de cada IP. De seguida, foram calculados os Z-Score das variáveis numéricas desses sub conjuntos de dados. Desta forma, o cálculo do Z-Score de uma actividade fraudulenta não é influenciado pelos valores de actividades não fraudulentas. De seguida os sub conjunto de dados são reunidos para formar o conjunto de dados de treino. Este processo torna-se bastante importante para os resultados da classificação uma vez que, o modelo irá classificar conjuntos de pedidos de registo referentes a um determinado IP.

Para as variáveis categóricas foi utilizado One Hot Encoding de forma a destingir os diversos term codes e o status de término do pedido. Para definir os term codes foram criadas 27 colunas com os term codes que ocorrem com mais frequência. Para definir o status foram criadas 3 colunas, uma para término por erro, por timeout e término bem sucedido.

3.3.2 Análise da Distribuição Espacial dos Dados

Tendo em consideração que algumas das abordagens propostas introduzem o conceito de noção espacial e de dependência de variáveis, foi feita uma Principal Component Analysis

(PCA). Desta forma foi possível visualizar uma porção aleatória do conjunto de dados.

PCA é um método de análise não supervisionado capaz de transformar as variáveis originais de um conjunto de dados num novo sistema de coordenadas onde, as variáveis não representam os valores mesuráveis das amostras, mas sim a variância do conjunto de dados. Para obter o novo sistema de coordenadas, o PCA calcula quais as direcções de maior variância dentro de um dado conjunto de dados e projecta as direcções obtidas como eixos do novo sistema de coordenadas. Os eixos do novo sistema de coordenadas são referidos como componentes principais, onde o primeiro componente representa a direcção de maior variância e o ultimo a direcção de menor variância. Desta forma é possível seleccionar as duas ou três primeiras componentes (visto que se apresentam como as mais relevantes em termos de variância) de forma reduzir a dimensão do problema e obter representações gráficas dos padrões escondidos entre as 7 características originais.

A análise por PCA é extremamente sensível a diferentes escalas de unidades. Desta forma antes de proceder à análise foi necessário normalizar o conjunto de dados com 0 como média e desvio padrão de 1. De seguida, usou-se o modelo PCA presente no módulo sklearn.decomposition que recebe um conjunto de dados com n características e retorna as suas componentes principais.

Começou-se por analisar o comportamento das duas componentes que, segundo o PCA, devem ter maior influência no processo de classificação dos pedidos de registo. A figura 3.4 apresenta o resultado gráfico da simulação a duas variáveis.

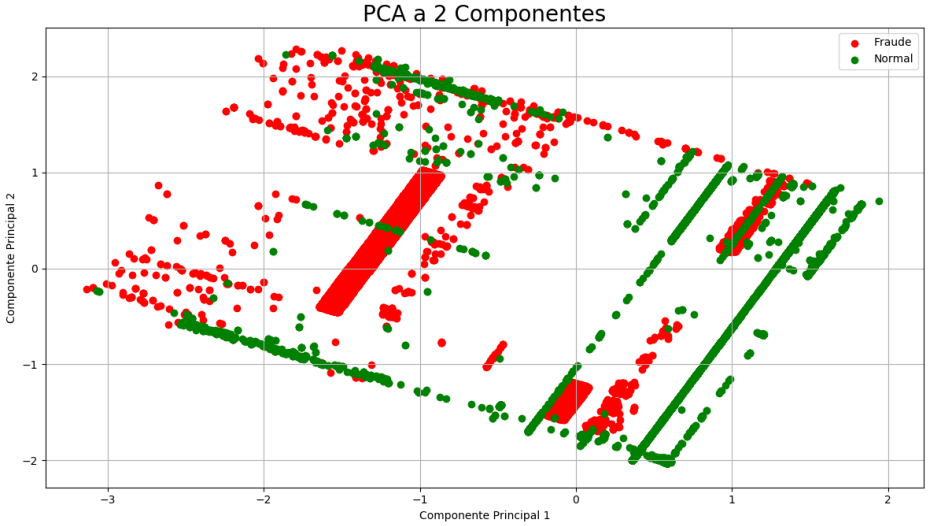


Figura 3.4: Análise PCA a duas componentes

Como podemos observar, a partir desta perspectiva não é possível traçar um plano capaz de separar os casos de fraude dos casos normais. No entanto é possível reparar que há várias sobreposições de pontos pertencentes a diferentes classes. Uma vez que os registos, que os pontos representam, não podem ser iguais, tudo indica que ao acrescentar

uma terceira componente seja possível separar os pontos sobrepostos. Para tal foi feita uma análise PCA com três dimensões. O resultado da simulação está apresentado na figura 3.5, onde é possível concluir que a terceira dimensão é essencial para separar as classes de uma forma mais eficiente.

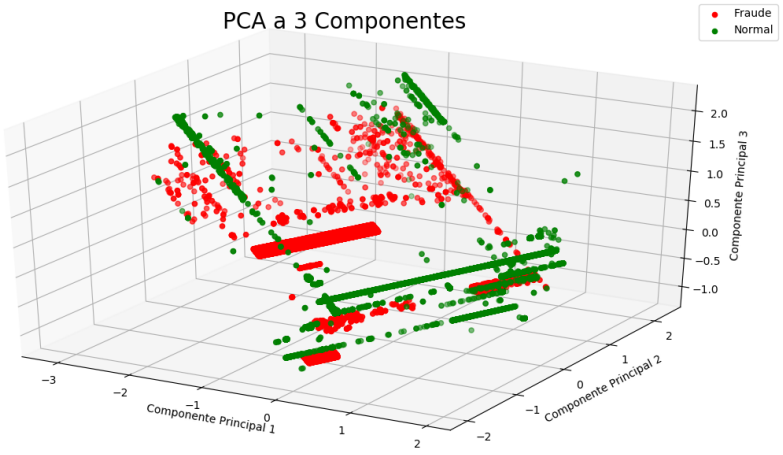


Figura 3.5: Análise PCA a três componentes

Como mostra a figura 3.5, a componente 3 acrescenta uma noção de profundidade e desta forma pode ser essencial para distinguir os dois conjuntos de pontos. A partir de uma análise da figura 3.5, e do efeito que a componente 3 introduziu na observação visual dos pedidos de registo, partiu-se para uma análise visual a duas dimensões com a componente 1 e a componente 3. A figura 3.6 confirma as suspeitas resultantes da análise da figura 3.5 uma vez que, desta forma é possível ter uma ideia mais precisa do plano capaz de separar os dois conjuntos. No entanto, podemos também concluir que, a separação dos dois conjuntos não possível através de um plano linear. Esta conclusão é vital para a decisão de certos hiperparametos na fase de treino dos vários algoritmos de classificação, descrita no capitulo 4.

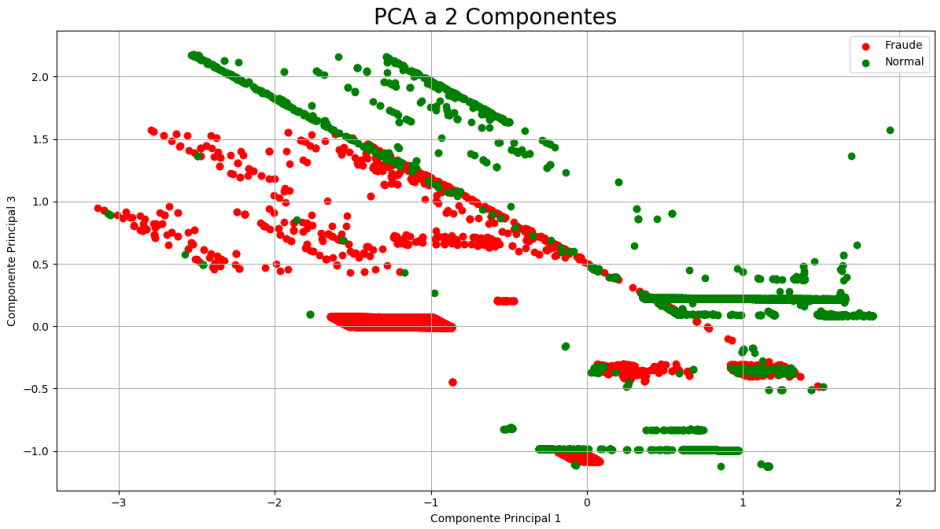


Figura 3.6: Análise PCA a partir da componente 1 e 3

Capítulo 4

Resultados Experimentais

Neste capitulo são apresentados os métodos de classificação testados assim como os seus resultados experimentais. Neste trabalho foram testados vários classificadores supervisionados, tirando partido dos dados etiquetados através da análise preliminar dos dados descrita no capitulo 3. Foi também estudado o uso de um algoritmo de classificação não supervisionada para aferir da possibilidade de detecção de ataques sem a análise prévia dos dados e etiquetagem de ataques anteriores.

Tendo em conta as diferenças de complexidade computacional dos diferentes algoritmos, o conjunto de dados original com 10 milhões de pedidos de registo demonstrou-se exaustivo para os processos de treino de algoritmos como KNN, LogReg, SVM, Decision Tree e K-Means. Assim sendo, foram recolhidos de forma aleatória 2.5 milhões de pedidos de registo pertencentes ao conjunto de dados original com 10 milhões de pedidos de registo. Este conjunto de dados reduzido representa 25% do conjunto original e apresenta o mesmo rácio de pedidos normais para pedidos fraudulentos. O processo de treino dos modelos KNN, LogReg, SVM e Decision Tree foi efectuado com 70% dos pedidos do conjunto de dados reduzido sendo que os restantes 30% foram usados para um primeiro teste ao modelo imediatamente após o treino. O processo de treino do Autoencoder e Fully connected Neural Networks foram efectuados com o conjunto de 10 milhões de pedidos, visto que o tamanho do conjunto de dados não apresentou nenhum entrave ao tempo do processo de treino.

Para o processo de treino cada pedido de registo encontra-se classificado como normal ou fraudulento tendo em conta a análise executada ao seu conjunto de pedidos de registo. Se um determinado conjunto foi considerado como fraudulento, então todos os seus pedidos são etiquetados como fraudulentos.

Todos os pedidos de registo presentes em qualquer conjunto foram pré processados tendo em conta a sua amostra (IP de origem). Desta forma é possível avaliar se um registo

isolado é ou não fraudulento. No entanto, sem uma identificação desse pedidos de registo em relação à sua origem, não é possível classificar nenhuma entidade (IP de origem) como normal ou fraudulenta. Para essa classificação é necessário que os pedidos de registo estejam agrupados em conjuntos e identificados por um IP de origem.

Uma vez que foi possível ter acesso a dados em tempo real, provenientes da rede da operadora, foi também possível testar a capacidade dos modelos em situações reais, nunca vistas durante o treino. Para tal foram recolhidos novos dados entre 23 de Maio e 23 de Julho de 2020. A escolha dos IPs a analisar foi feita de acordo com os critérios estabelecidos na secção 3.3.1. Os comportamentos de cada IP estão previamente etiquetados como fraudulentos ou normais com base no seu comportamento no dia em estudo. Desta forma é possível avaliar a exactidão dos modelos já treinados em condições reais com dados sem qualquer relação ao seu conjunto de treino. A etiquetagem dos IPs presentes no conjunto de dados recolhidos entre Maio e Junho foi feita através de uma análise detalhada do comportamento de cada IP de origem referenciado pelos métodos explicados na secção 3.3.1. Embora seja possível classificar o conjunto de pedidos de registo de um determinado IP de origem, como fraudulento ou normal, sem recorrer a um classificador, este processo é demorado e dispendioso em recursos humanos uma vez que requer que sejam analisados todos os pedidos de registo de um IP de origem. O processo de etiquetagem dos dados referentes a um IP é feito através da análise de vários parâmetros como, variância dos portos de origem, números de contas a registar, taxa de sucesso e ainda as tentativas por diferentes domínios de todos os pedidos de registo com esse IP de origem.

Este conjunto de dados, sem qualquer relação ao seu conjunto de treino, contém 469 conjuntos de pedidos de registo. Cada conjunto de pedidos de registo é identificado pelo seu IP de origem e, contém todos os pedidos de registo que esse IP apresentou às SBCs no dia em que foi referenciado pelos métodos explicados na secção 3.3.1. Desta forma se os métodos explicados na secção 3.3.1 identificarem um IP de origem como suspeito, todos os pedidos de registo executados no último dia com esse IP de origem são recolhidos e agrupados num conjunto de pedidos de registo identificado pelo seu IP de origem.

Para testar a capacidade dos modelos face a IPs com actividade normal (não fraudulenta), foram também agrupados em conjuntos, identificados pelo IP de origem, os pedidos de registos de IPs com comportamento normal. Desta forma, o conjunto de dados é compostos por 253 conjuntos de pedidos fraudulentos e 216 conjuntos de pedidos não fraudulentos. Foi ainda adicionado a este conjunto de dados 20 conjuntos de pedidos de registo, também agrupados pelo IP de origem, de dispositivos com erros no processo de registo. Os equipamentos com erros no processo de registo acabam por ter um comportamento semelhante, aos comportamentos fraudulentos. Ao introduzir estes dados espera-se testar a robustez dos modelos em relação a situações de classificação mais complexa.

Para testar os classificadores, serão classificados todos os pedidos presentes em cada conjunto. O resultado final será a percentagem de pedidos de registo desse grupo classificados como fraudulentos. Assim sendo, o comportamento de um IP de origem será

avaliado pela percentagem de pedidos de registo classificados como fraudulentos.

Todos os testes foram efectuados num equipamento fornecido pela operadora com um Intel(R) Core(TM) i5-4200U CPU de 1.60 GHz a 2.30 GHz e 8GB de RAM.

Cada algoritmo terá a sua própria secção e no final será feita uma pequena análise e comparação dos resultados experimentais.

4.1 Experiência 1 - K-Nearest Neighbors

Como já foi explicado, o KNN é um algoritmo de classificação supervisionada capaz de criar relações entre acontecimentos através da sua proximidade espacial. Este algoritmo representa todas as entradas do conjunto de dados num referencial e baseia a sua previsão nos acontecimentos mais próximos do objecto a ser classificado. Estes acontecimentos mais próximos são denominados de vizinhos do objecto a ser classificado. A classe da maioria dos seus vizinhos é atribuída ao objecto. Como o voto tem de ser restrito aos vizinhos mais próximos, o número de vizinhos com direito de voto é escolhido como um hiperparâmetro do modelo a treinar.

4.1.1 Processo de Treino

Tendo em conta que por cada objecto analisado é necessário criar uma distribuição espacial dos dados, este processo é altamente exaustivo em termos computacionais. Por este motivo, foi necessário reduzir o tamanho dos dados de treino utilizando o conjunto de dados reduzido. Com o conjunto de dados original, de 10 milhões de pedidos de registo, o algoritmo não foi capaz de acabar o treino, tendo sido interrompido depois de um período de treino superior a 24 horas. Ao treinar com o conjunto de dados reduzido, que corresponde a 25% do conjunto de dados de 10 milhões de pedidos, a fase de treino demorou um pouco mais de 12 minutos.

O conjunto de dados foi separado em duas secções, uma secção de treino e uma secção de teste sendo a última secção um terço do conjunto de dados reduzido. Depois do treino o modelo foi submetido aos dados de teste de forma a comprovar a sua precisão em situações não contempladas na fase de treino. O classificador foi testado em 754594 pedidos que corresponde a 30% do conjunto de treino. Os resultados foram bastante favoráveis, como podemos ver na tabela 4.1 com valores arredondados a duas casas decimais, uma vez que o modelo foi capaz de obter uma precisão de 99,9998%, com 48 falsos positivos e 82 falsos negativos.

	Precisão	Recall	F1	Ocorrências
Normal	1.00	1.00	1.00	375325
Fraude	1.00	1.00	1.00	379269
Cross Validation	1.00	1.00		754594

Tabela 4.1: Relatório de Classificação de KNN.

Tendo em conta que o conjunto de dados de teste pode estar influenciado pela divisão entre conjunto de teste e de treino, foi também executada uma cross validation com 5 secções para testar a capacidade de generalização do modelo. A precisão do teste foi de 99.9842% com um recall de 99.9983%.

Embora os resultados iniciais sejam muito favoráveis, foi necessário abordar outras hipóteses de classificação, visto que o modelo não é capaz de generalizar as suas previsões para equipamentos com erros de registo, como podemos observar na secção 4.1.2.

4.1.2 Classificação no Conjunto de Teste

Como explicado na introdução do capitulo 4, os modelos treinados com o conjunto de dados recolhidos foram utilizados para classificar dados de um conjunto de teste separado, de modo a validar o modelos em dados nunca vistos, e sem qualquer relação ao conjunto de treino. O output do modelo é a percentagem de registos considerados fraudulentos, de um dado grupo de pedidos. Assim, é necessário definir um threshold que estabeleça o percentagem a partir da qual um grupo de pedidos de registo de um IP de origem é classificado como fraudulento.

Foram recolhidas as percentagens de pedidos classificados como fraudulentos por cada um dos 469 grupos de pedidos. Cada um destes grupos de pedidos representa o comportamento um IP de origem durante um dia. Na figura 4.1 podemos observar que a esmagadora maioria dos comportamentos fraudulentos apresentam uma percentagem superior a 90% de pedidos classificados como fraudulentos, enquanto que os conjuntos de pedidos com comportamentos normais entre 0% a 10% de registos fraudulentos.

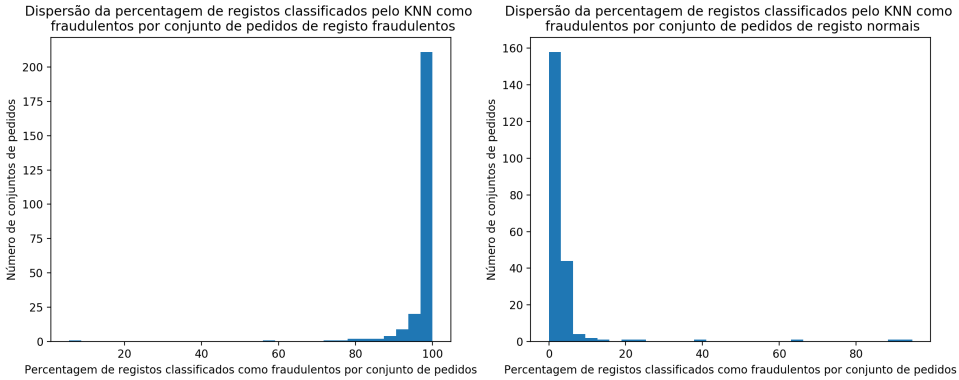


Figura 4.1: Dispersão das percentagens de registos classificados como fraudulentos pelo KNN em conjuntos de pedidos de registo fraudulentos e normais

A partir de uma abordagem mais detalhada foi possível concluir que, dos 216 conjuntos de pedidos normais, 207 obtiveram menos de 10% de registos classificados como fraude e apenas 3 obtiveram percentagens acima dos 50%. Ao sobrepor as duas simulações, é notório que os 50% representam uma fronteira solida entre casos normais e casos de fraude como podemos observar na figura 4.2

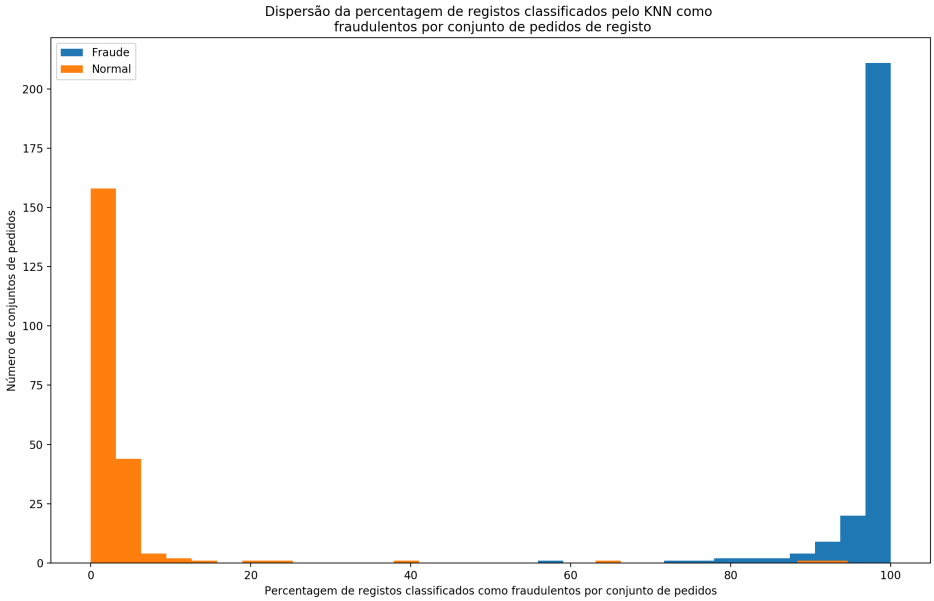


Figura 4.2: Dispersão das percentagens de registos classificados como fraudulentos pelo KNN em conjuntos de pedidos fraudulentos e normais

No entanto, foram conduzidas simulações com valores de threshold compreendidos entre 20% e 80% de forma a obter o valor exacto de falsos negativos e falsos positivos que cada threshold origina. A tabela 4.2 apresenta os resultados das simulações.

Threshold	20-22	23-39	40-57	58-64	65-74	75-76	77-78	79-80
FN	0	0	0	1	1	2	3	4
FP	5	4	3	3	2	2	2	2

Tabela 4.2: Falsos Positivos e Falsos Negativos do KNN em relação ao Threhold.

Ao analisar os resultados é perceptível que o valor mais apropriado para uma boa classificação se situa entre os 40% e 57%, visto que estes valores de threshold detectam todos os casos de ataque com prejuízo de 3 casos normais. Tendo em conta que a operadora pretende o mínimo possível de falsos negativos, mesmo em detrimento de um maior número de falsos positivos, escolheu-se 40% como threshold. Este valor foi escolhido pelo facto de ser o mínimo dentro da gama de valores com melhores resultados.

Assim sendo, podemos definir que a partir dos 40% de registos classificados como fraudulentos, um comportamento pode ser classificado como fraude. Com base neste threshold partiu-se para a análise do modelo perante conjuntos de pedidos de equipamentos com erros de registo. Foram recolhidas as percentagens que o modelo retornou ao classificar os 20 conjuntos de pedidos de equipamentos com erro. Os resultados do teste estão expostos na figura 4.3 onde é possível observar que uma grande parte dos equipamentos têm uma percentagem de classificação de registos fraudulentos entre os 65% e os

95% chegando mesmo a valores muito próximos de 100%. Ao aplicar o threshold de 40% estaríamos a classificar como fraude o comportamento da totalidade dos equipamentos com erro.

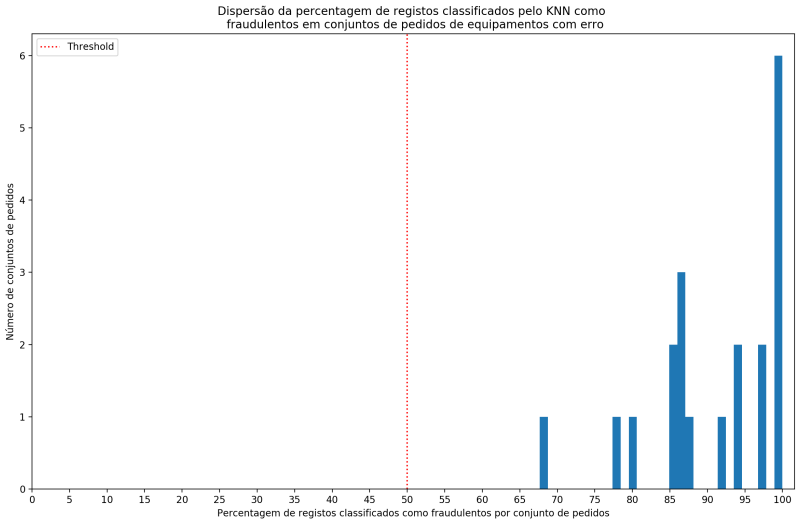


Figura 4.3: Dispersão da percentagem de registos classificados pelo KNN como fraudulentos em conjuntos de pedidos de equipamentos com erro

4.2 Experiência 2 - Regressão Logística

A Regressão Logística é um método estatístico tradicional capaz de ser aplicado como classificador na área de Machine Learning. Ao contrário da regressão linear, que tem como objectivo encontrar a recta que melhor representa os pontos em estudo através do R^2, a regressão logística pretende encontrar a melhor curva de probabilidades entre dois acontecimentos. Assim sendo, a regressão logística obtém resultados binomiais como verdadeiro ou falso em vez de obter previsões em relação a uma variável contínua. A curva de probabilidades obtida tem a forma de uma sigmoid com valores compreendidos entre 1 e 0 como podemos ver na figura 4.4.

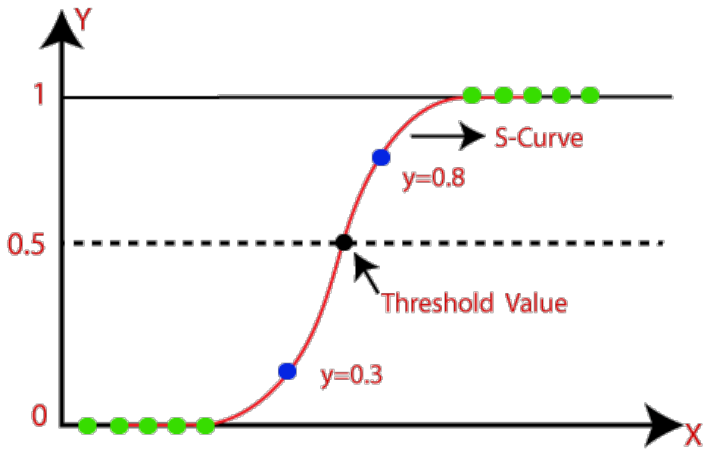


Figura 4.4: Curva de regressão logística

Sendo por exemplo $Y = 0$ o identificador de um comportamento normal e $Y = 1$ o resultado de um comportamento fraudulento, o modelo obtido iria traçar as curvas probabilísticas em relação aos vários parâmetros de análise. Desta forma, para cada interacção com a rede da operadora os valores dos 7 parâmetros serão aplicados às curvas probabilísticas. Por consequência, o resultado final será um valor compreendido entre 0 e 1, referente à probabilidade de fraude.

4.2.1 Processo de Treino

Para o treino do modelo foi criado um vector de 15 números igualmente espaçados numa escala logarítmica com começo em 10^{-5} até 10^{8}. Este vector foi utilizado numa Grid Search com 5 secções para o hiperparâmetro C, que representa a força de regularização do modelo. Uma Grid Search é um processo utilizado na regulação dos parâmetros dos modelos de Machine Learning e inteligência artificial. Através da Grid Search é possível criar vários modelos com diferentes valores nos seus parâmetros e, recolher os resultados de precisão e exactidão para cada modelo criado. Desta forma é possível obter os melhores parâmetros para o modelo a desenvolver através da selecção do modelo que obteve a maior precisão. A força de regularização é usada para penalizar a magnitude dos parâmetros de forma a reduzir o overfitting. Depois de executada a Grid Search, o valor escolhido para a força de regularização foi de 3737 visto que com este valor de C foi possível obter um modelo com uma precisão de 93.8514%. O alto valor de C obtido com a Grid Search é justificável pela grande quantidade de dados disponíveis, o que permite que a força de regularização seja menor.

O conjunto de dados reduzido foi separado em duas secções, uma secção de treino e uma secção de teste sendo a última secção um terço do conjunto original. Através dos dados da secção de treino, criou-se um modelo de regressão logística com $C = 3737$. De seguida o modelo foi testado com os dados da secção de teste de forma a obter a curva característica de operação do receptor apresentada na figura 4.5. A curva de operação

do receptor ou *Receiver Operating Characteristic (ROC)* é a curva que representa o desempenho de um classificador binário à medida que o valor de referencia ou threshold de classificação varia. A variação do valor de threshold afecta a taxa de falsos positivos e verdadeiros positivos. Ao analisar a curva ROC pretende-se encontrar o ponto da curva que apresenta a melhor relação possível entre falsos positivos e positivos verdadeiros.

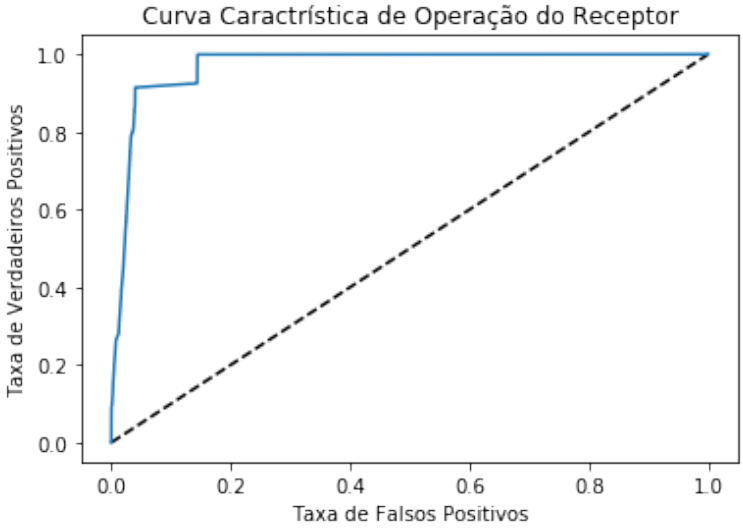


Figura 4.5: Curva Característica de Operação do Receptor

Pela curva ROC é possível observar que a taxa de falsos positivos está bastante próxima de zero para 90% de positivos verdadeiros. Estes valores aparentam ser bastante promissores tendo em conta as necessidades da operadora.

Uma vez que a regressão logística apresenta como resultado um valor de 0 a 1 correspondente à probabilidade de um acontecimento ser verdadeiro, é necessário definir um valor limite a partir do qual é seguro classificar um elemento como verdadeiro ou falso. Este conceito está ilustrado na figura 4.4 como threshold. Na figura 4.4 o threshold foi definido a 0.5 uma vez que, segundo os autores da imagem, era seguro classificar um objecto se ele tiver mais do que 50% de probabilidades de pertencer a uma dada classe. Este valor pode variar de problema para problema consoante o grau de certeza necessário para o problema em questão. Sendo o objectivo a classificação de comportamentos fraudulentos, é do interesse da operadora que a taxa de falsos negativos seja o mais baixa possível, mesmo que tal implique o aumento dos falsos positivos. No entanto, mesmo sabendo que a operadora pretende uma baixa taxa de falsos negativos, é também importante que a taxa de falsos positivos não ultrapasse o limite do razoável, de modo a garantir a credibilidade do modelo. Assim sendo, foi necessário analisar o comportamento da precisão e Recall do modelo em relação à variação do threshold. Para tal foram executadas várias classificações dos dados de teste com diferentes valores de threshold e recolhidas as matrizes de confusão com o objectivo de traçar o comportamento da precisão e recall em função do threshold. O resultado das simulações está ilustrado na figura 4.6. Uma matriz de

confusão é uma tabela de duas colunas por duas linhas capaz de resumir o desempenho do classificador ao apresentar o número de falsos positivos , falsos negativos , verdadeiros positivos e verdadeiros negativos.

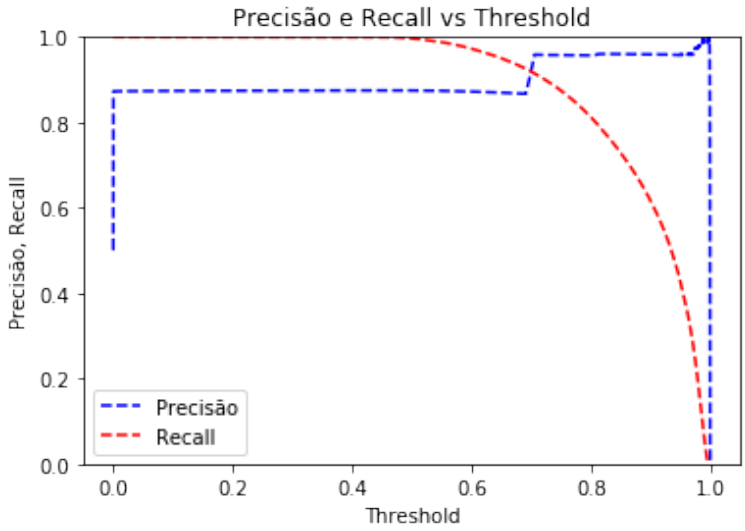


Figura 4.6: Curva da Precisão e Recall em função do Threshold

Como podemos ver na figura 4.6, o Recall só começa a ser afectado para um threshold superior a 0.5 e a precisão mantêm-se estável para valores de threshold intermédios. À medida que o threshold se aproxima de 0.8 a precisão aumenta e o Recal entra numa tendência descendente. Assim sendo, a escolha do threshold não foi um problema, uma vez que existe um largo espectro de hipóteses que satisfazem as necessidades do problema. Como tal foi escolhido o valor 0.5, visto que para a operadora o Recall é mais importante do que a precisão.

Com o threshold definido como 0.5 foram efectuados testes com os dados da secção de teste. O classificador foi testado em 754 mil amostras. Os resultados foram favoráveis, como podemos ver na tabela 4.3 uma vez que, o modelo foi capaz de obter uma precisão de 92,6382%. No entanto o modelo obteve 54183 falsos positivos e 1369 falsos negativos. Embora estejamos a falar de 54183 pedidos de registo e não de 54183 grupos de pedidos (ataques), o alto valor de falsos positivos pode ser preocupante. Embora seja um valor relativamente alto, correspondendo a 7% do conjunto de dados, este dado não afecta a capacidade do modelo em classificar conjuntos de pedidos como poderemos ver na secção 4.2.2

	Precisão	Recall	F1	Ocorrências
Normal	1.00	0.86	0.92	375325
Fraude	0.87	1.00	0.93	379269
Cross Validation	0.98	1.00		754594

Tabela 4.3: Relatório de Classificação da Regressão Logística.

Tendo em conta que o conjunto de dados de teste pode estar influenciado pela divisão do conjunto original, foi também executada uma cross validation com 5 secções para testar a capacidade de generalização do modelo. A precisão do teste foi de 98.4273% com recall de 99.9957%

4.2.2 Classificação no Conjunto de Teste

O modelo foi testado ao classificar o conjunto de dados que representam os 469 grupos de pedidos de registo. A classificação final do comportamento de um IP é obtida a partir da sua percentagem de pedidos classificados como fraudulentos. Foram recolhidas 469 percentagens que o modelo retornou ao analisar cada um dos 253 grupos de pedidos fraudulentos e 216 conjuntos de pedidos normais.

Como mostra a figura 4.7, todos os conjuntos de pedidos fraudulentos têm mais de 90% de registos classificados como fraudulentos. Os resultados para os conjuntos de pedidos normais encontram-se concentrados nos 3,28%, o que aparenta ser bastante promissor.

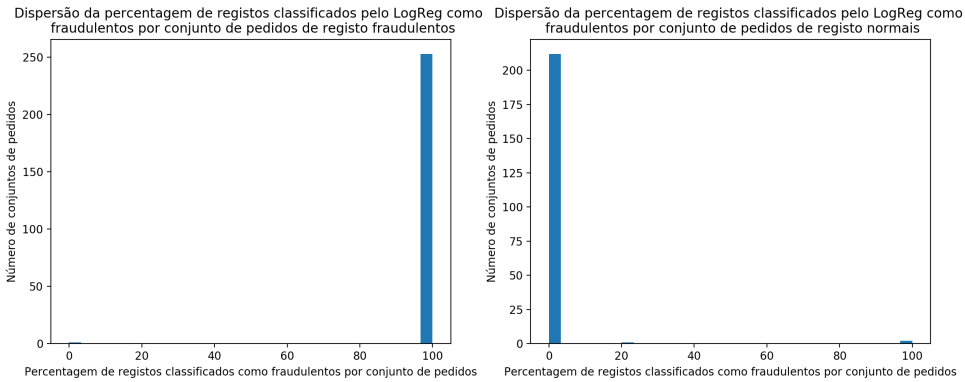


Figura 4.7: Dispersão das percentagens de registos classificados como fraudulentos pelo LogReg em conjuntos de pedidos fraudulentos e normais

A figura 4.8 sobrepõe as duas distribuições. A partir da análise desta figura é mais uma vez notório que existe um plano separador entre as duas classes.

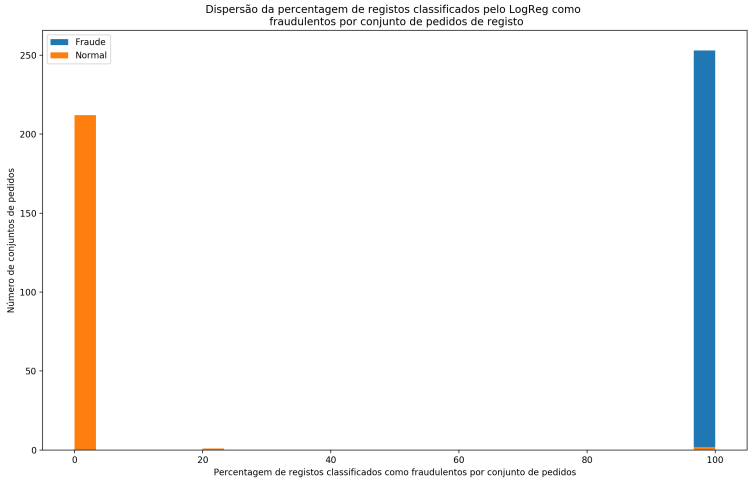


Figura 4.8: Dispersão das percentagens de registos classificados como fraudulentos pelo LogReg em conjuntos de pedidos fraudulentos e normais

Foram feitas simulações com valores de threshold entre 20% e 80% de forma a obter o número de falsos positivos e falsos negativos. A tabela 4.4 apresenta os resultados da simulação.

Threshold	20-21	23-80
FN	0	0
FP	4	3

Tabela 4.4: Falsos Positivos e Falsos Negativos do LogReg em relação ao Threhold.

A partir da análise da tabela 4.4 é possível observar que qualquer valor de threshold entre 20% e 80% apresenta bons resultados. Este acontecimento pode ser explicado pelas médias dos resultados obtidos. As percentagens de pedidos normais classificados como fraudulentos concentram-se muito próximas de zero, tendo uma média de 3,28%. Por outro lado, as percentagens de pedidos fraudulentos classificados como tal concentram-se muito próximas de 100% com um recall de 97,90%. Assim sendo, os resultados estão muito concentrados nos extremos do referencial o que torna possível escolher qualquer valor de threshold compreendido entre 10% e 90%. Desta forma, foi escolhido 50% como threshold, uma vez que o modelo mostrou ser capaz de ter um alto grau de precisão.

Com base neste threshold partiu-se para a análise do modelo perante comportamentos de equipamentos com erros de registo. Foram recolhidas as percentagens que o modelo retornou quando sujeito aos 20 conjuntos de pedidos provenientes de equipamentos com erro. Os resultados estão expostos na figura 4.9

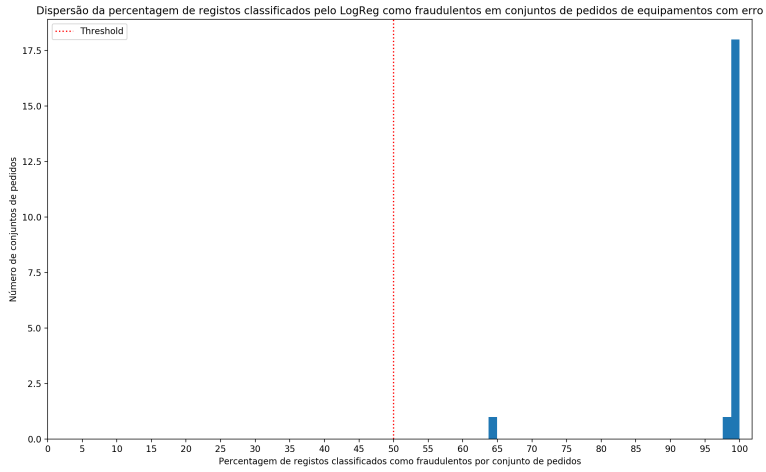


Figura 4.9: Dispersão da percentagem de registos classificados pelo LogReg como fraudulentos em conjuntos de pedidos de equipamentos com erro

Tendo em consideração que o threshold escolhido foi de 50%, ao observar a figura 4.9 concluímos que o modelo não consegue distinguir entre casos de fraude e erros de equipamentos, uma vez que classifica 100% das actividades de equipamentos com erro como sendo fraudulentas. Em suma, a regressão logística apresenta resultados favoráveis mesmo sem ter sido capaz de distinguir entre casos de fraude e de erro. Para além destes resultados, este modelo apresenta uma segunda vantagem. O intervalo de tempo para a classificação de grandes volumes de dados é bastante inferior aos modelos até agora testados (KNN). Sendo a detecção em tempo real uma condição para a utilidade da ferramenta proposta, o modelo por regressão logística apresenta uma forte vantagem em relação aos restantes.

4.3 Experiência 3 - Support Vector Machine

Como explicado na secção 2.3.2.1, o Support Vector Machine (SVM) é um algoritmo de classificação capaz de calcular um Hyperplane entre classes, mesmo a partir de elevadas dimensões. O Hyperplane é um plano optimizado, definido entre os dois conjuntos de dados tendo uma distância igual em relação aos Support Vectors dos dois conjuntos. Tendo em conta que nem todos os conjuntos de dados são linearmente separáveis, pode ser necessário alterar as dimensões da amostra, de modo a que seja possível criar um plano que separe os dois conjuntos.

4.3.1 Processo de Treino

Para o treino do modelo foi usado o conjunto de dados reduzido por motivos de eficiência e rapidez de treino. A escolha do tipo de kernel para o modelo a treinar foi feita tendo em conta a conclusão apresentada na secção 3.4.2. Nesta secção demonstrou-se que não é

possível separar as duas classes por um método linear. Assim sendo, usou-se um kernel Gaussiano. O conjunto de dados foi separado em duas secções, uma secção de treino e uma secção de teste, sendo a última secção um terço do conjunto original. Depois do treino, o modelo foi submetido aos dados de teste de forma a comprovar a sua precisão e Recall em situações não contempladas na fase de treino. O classificador foi testado em mais de 750 mil amostras e obteve resultados favoráveis com 99,8508% de exactidão, 1143 falsos positivos e 36 falsos negativos. Os restantes indicadores do processo de treino, arredondados a duas casas decimais, estão presentes na tabela 4.5.

	Precisão	Recall	F1	Ocorrências
Normal	1.00	1.00	1.00	410564
Fraude	1.00	1.00	1.00	379840
Cross Validation	1.00	1.00		754594

Tabela 4.5: Relatório de Classificação de SVM.

Tendo em conta que o conjunto de dados de teste pode estar influenciado pela divisão do conjunto original, foi também executada uma cross validation com 5 secções para testara capacidade de generalização do modelo. A precisão do teste foi de 99.9939% com recall de 99.9982%

Mesmo usando o conjunto de dados reduzido, o treino teve a duração de 10 horas e o processo de classificação dos dados de teste foi concluído em 6 minutos. A demora no processo de classificação pode representar um entrave para o projecto, uma vez que a operadora pretende um mecanismo em tempo real.

4.3.2 Classificação no Conjunto de Teste

À semelhança das outras experiências, o modelo foi testado ao classificar o conjunto de dados que representam os 469 grupos de pedidos de registo. A classificação final do comportamento de um IP é obtida a partir da sua percentagem de pedidos classificados como fraudulentos. Foram recolhidas 469 percentagens que o modelo retornou ao analisar cada um dos 253 grupos de pedidos fraudulentos e 216 conjuntos de pedidos normais. A figura 4.10 apresenta os resultados obtidos.

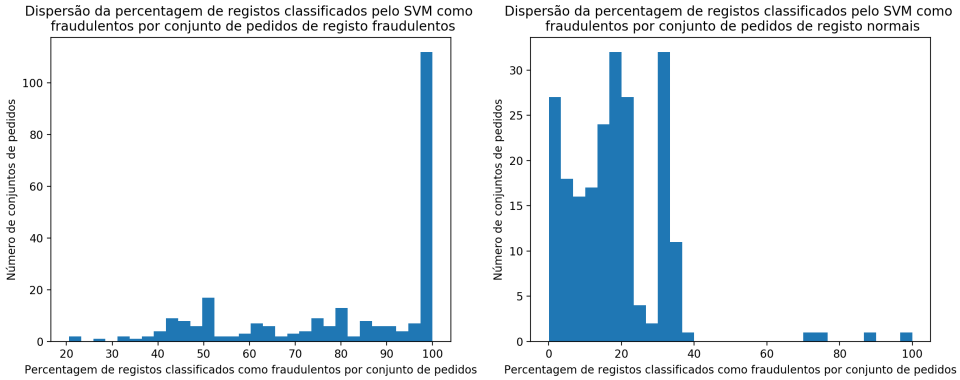


Figura 4.10: Dispersão das percentagens de registos classificados como fraudulentos pelo SVM em conjuntos de pedidos fraudulentos e normais

Como podemos observar na figura 4.10, a esmagadora maioria dos conjuntos de pedidos não fraudulentos obtiveram resultados inferiores a 40% de registos classificados como fraude, sendo a média de percentagens igual a 22,7195%. A percentagem de pedidos de registo de conjuntos de pedidos normais classificados como fraudulentos, têm uma dispersão em nada semelhante à das experiências anteriores, visto que os resultados se encontram muito mais distribuídos e não tão concentrados nos 0%.

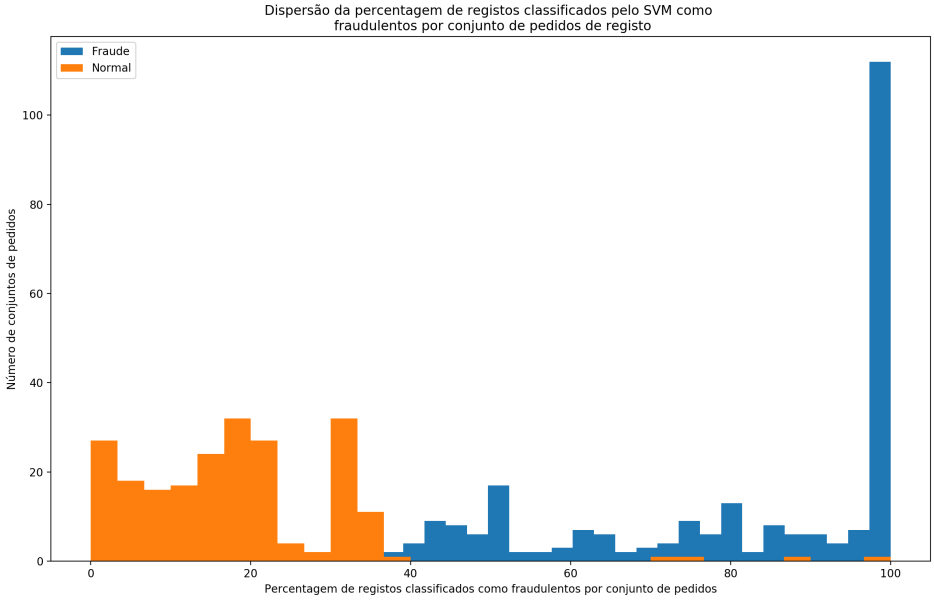


Figura 4.11: Dispersão das percentagens de registos classificados como fraudulentos pelo SVM em conjuntos de pedidos fraudulentos e normais

A partir da sobreposição dos resultados referentes aos dois tipos de actividades, presente na figura 4.11, podemos observar que o melhor threshold para o problema se situa entre os 35% e os 45%. Com o intuito de definir melhor o valor de threshold, foram feitas

simulações de forma a obter o número de falsos negativos e falsos positivos em função do valor de threshold. A tabela 4.6 apresenta os resultados da simulação.

Threshold	35	36	37	38	39	40-43	44	45
FN	2	2	2	3	4	4	6	9
FP	45	33	23	16	8	5	5	5

Tabela 4.6: Falsos Positivos e Falsos Negativos do modelo SVM em relação ao Threhold.

Tendo em conta os objectivos que a operadora traçou para o projecto, o threshold mais apropriado encontra-se em qualquer valor entre 40% e 43%. Usou-se 40% como threshold uma vez este se torna mais abrangente para casos de fraude visto que , quanto menor a percentagem menos provável é a existência de falsos negativos. De seguida foi feito um teste ao modelo a partir de dados referentes a equipamentos com erro, de forma a averiguar a sua robustez na distinção entre ataques e aparelhos com erro. Os resultados da classificação estão exibidos na figura 4.12 e mostram claramente que, com o threshold escolhido, todas as actividades que correspondem a equipamentos com erro são classificadas como fraudulentas.

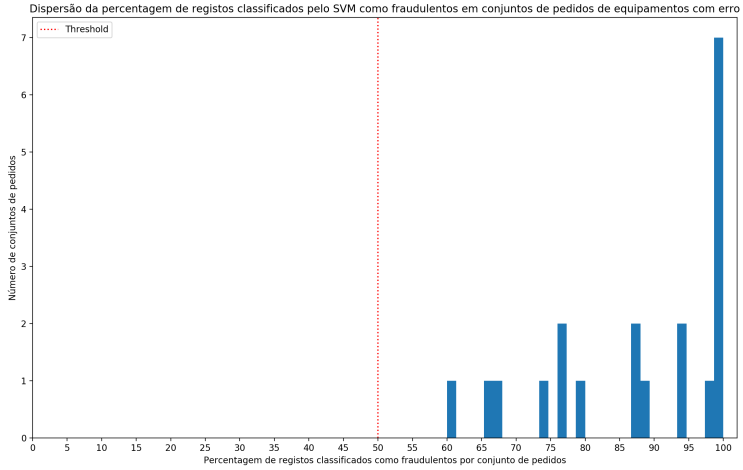


Figura 4.12: Dispersão da percentagem de registos classificados pelo SVM como fraudulentos em conjuntos de pedidos de equipamentos com erro

4.4 Experiência 4 - Decision tree

Uma árvore de decisão é um algoritmo capaz de classificar objectos com base em uma determinada sequência de condições. A melhor sequência de condições é obtida ao treinar o algoritmo com o tipo de dados que pretendemos que o mesmo seja capaz de classificar. A cadeia de condições pode ser vista como uma árvore, uma vez que esta é composta por vários nós que divergem para criar novos nós. Cada nó representa uma condição de verdadeiro ou falso com base em uma das características dos dados de treino. Consoante

o resultado da condição criam-se dois conjuntos de dados, os que respeitam a condição e os que não respeitam a condição. Estes conjuntos de dados serão o input dos dois nós seguintes. No caso de um dos conjuntos obtidos contemplar dados de apenas uma classe, diz-se que esse conjunto dá origem a um nó perfeitamente equilibrado, uma que vez não existe incerteza em relação à classe. Se o outro nó não for perfeitamente equilibrado, o seu conjunto de dados é submetido a uma nova condição até que se encontre um nó perfeitamente equilibrado. A escolha das condições é feita em função do cálculo da incerteza por impuridade, ou entropia que essa condição cria. Por esta lógica, ao fim de algumas condições o modelo é capaz de separar as classes presentes no conjunto de treino.

4.4.1 Processo de Treino

Tendo em conta que o modelo de árvores de decisão suporta vários parâmetros de aprendizagem, foi primeiro feita uma Grid Search de forma a apurar os melhores parâmetros para os dados à nossa disposição.

Através da Grid Search é possível criar vários modelos com diferentes valores nos seus parâmetros de forma a recolher os resultados de precisão e exactidão para cada modelo criado. Desta forma é possível obter os melhores parâmetros para o modelo a desenvolver através da selecção do modelo que obteve a maior precisão.

Para tal foram criados 4 vectores que definem os valores dos parâmetros a serem testados. O primeiro vector representa o critério de incerteza que pode ser Gini ou por entropia.

O segundo vector apresenta 6 valores de 5 a 30, igualmente espaçados. Este vector representa os valores para a máxima profundidade da árvore. Sem este parâmetro o processo de treino terminava assim que todos os nós fossem perfeitamente equilibrados. Ao impor uma altura máxima é possível evitar que a árvore se torne exaustiva e, combater questões de overfitting.

O terceiro vector é responsável pela escolha da condição de divisão dos nós. A escolha da condição pode ser feita através do melhor valor obtido pelo critério ou, de forma aleatória entre os melhores valores obtidos pelo critério.

Por fim, o último vector contém 3 valores que representam o mínimo de objectos que um nó deve ter para ser dividido.

O resultado da Grid Search indica que o Gini é o melhor modelo para o critério de escolha da condição quando comparado com a entropia. A altura máxima da árvore deve ser de 5 nós, o mínimo de amostras para uma divisão obtido foi 100 e o método de escolha deve ser aleatório.

O conjunto de dados reduzido foi separado em duas secções, uma secção de treino e uma secção de teste, sendo a última secção um terço do dataset original. Através dos dados da secção de treino criou-se uma árvore de decisão com os parâmetros obtidos na Grid Search. De seguida o modelo foi testado com os dados da secção de teste. Os resultados obtidos com o conjunto de dados de teste foram perfeitos, com uma precisão

e Recall igual a 100%, 0 falsos positivos e 0 falsos negativos. A tabela 4.7 apresenta os resultados obtidos com o conjunto de teste.

	Precisão	Recall	F1	Ocorrências
Normal	1.00	1.00	1.00	410564
Fraude	1.00	1.00	1.00	379840
Cross Validation	1.00	1.00		754594

Tabela 4.7: Relatório de Classificação de Decision Tree.

Tendo em conta que o conjunto de dados de teste pode estar influenciado pela divisão do conjunto original, foi também executada uma cross validation com 5 secções para testar a capacidade de generalização do modelo. A precisão e recall do teste foram de 100%

Para entender os motivos pelos quais a árvore apresenta resultados muito melhores do que o esperado foi necessário analisar a estrutura da árvore de decisão que o modelo criou. A figura 4.13 mostra a árvore criada pelo algoritmo. Como podemos observar, a árvore é demasiado simplista e cria decisões com base em apenas dois campos do conjunto de dados. O parâmetro do conjunto de treino term_code_200:OK representa a ocorrência de um 200:OK como term code do pedido a ser analisado. O parâmetro de treino register_setup_time representa o tempo de registo associado a um pedido. Como foi referido anteriormente, o conjunto de dados usado não contém casos de tentativas de registo fraudulento com sucesso, isto é, todos os exemplos de pedidos de registo fraudulento presentes no conjunto de dados têm como term code algo diferente de 200:OK. Sendo o objectivo do projecto detectar comportamentos fraudulentos antes que o atacante tenha sucesso o conjunto de dados obtidos apresenta uma grande concentração de pedidos de registo fraudulento sem sucesso. Assim sendo, a árvore tem uma forte inclinação para classificar qualquer pedido sem sucesso como sendo fraude.

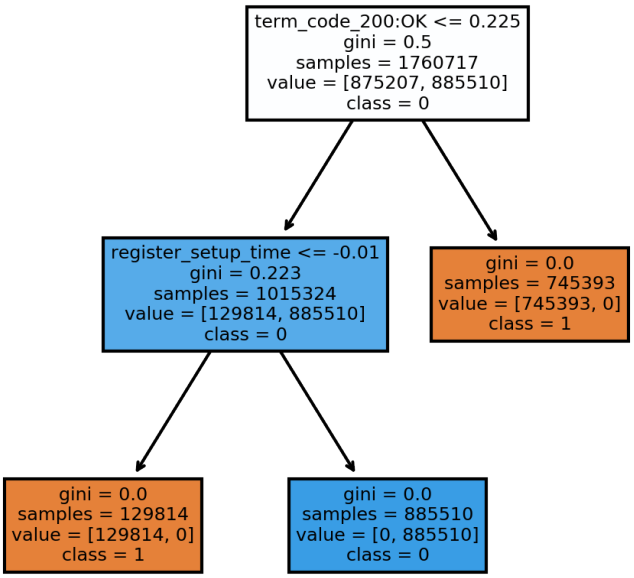


Figura 4.13: Esquema da arvore de decisão criada pelo Decision Tree

O modelo obtido evidencia um claro overfitting em relação aos dados de treino. Tendo em conta que o term code pode ser a raiz do problema de simplificação excessiva por parte da árvore de decisão, foi treinado um segundo modelo de árvore de decisão sem a característica term code. Embora a árvore obtida seja mais extensa, o modelo apresenta os mesmos resultados com apenas um registo mal classificado, visto que se foca apenas no tempo de registo.

4.4.2 Classificação do Conjunto de Teste

À semelhança das outras experiências, o modelo com a característica term code foi testado ao classificar o conjunto de dados que representam os 469 grupos de pedidos de registo. A classificação final do comportamento de um IP é obtida a partir da sua percentagem de pedidos classificados como fraudulentos. Foram recolhidas 469 percentagens que o modelo retornou ao analisar cada um dos 253 grupos de pedidos fraudulentos e 216 conjuntos de pedidos normais. A figura 4.14 apresenta a os resultados obtidos.

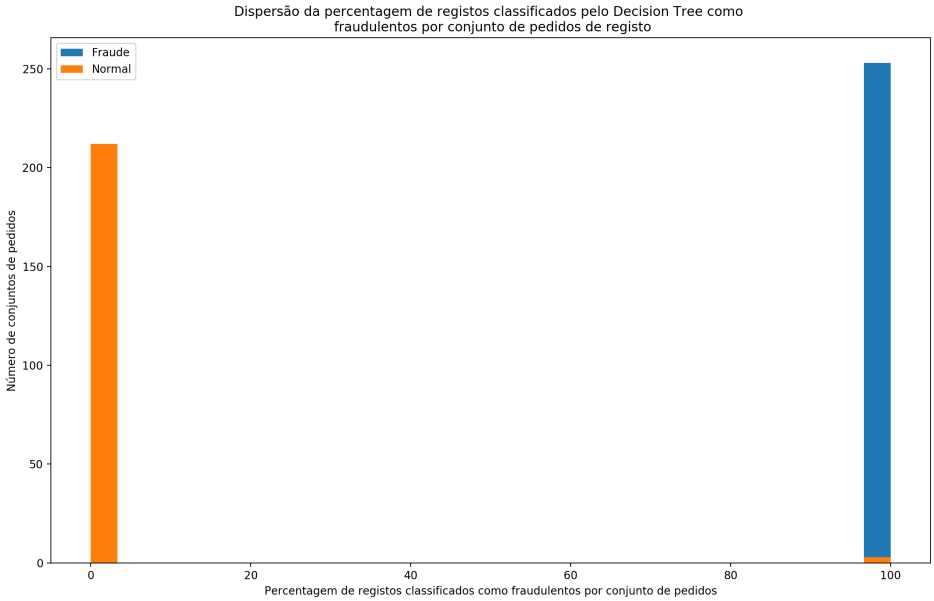


Figura 4.14: Dispersão das percentagens de registos classificados como fraudulentos pelo Decision Tree em conjuntos de pedidos fraudulentos e normais

Como foi explicado na secção de treino, era espectável que o modelo tivesse uma performance acima da média, uma vez que as suas condições estão bastante adaptadas ao conjunto de treino. Assim sendo todos os conjuntos de pedidos fraudulentos obtiveram 100% dos pedidos classificados como fraude e dos 216 conjuntos de pedidos normais 213 obtiveram 0% de pedidos classificados como fraude. Tendo em consideração que os pedidos fraudulentos estão concentrados nos 100% e os pedidos normais nos 0%, à excepção de 3 conjuntos de pedidos normais que obtiveram 100%, é seguro considerar 50% como threshold do problema.

Com base no threshold escolhido, o modelo classificou os conjuntos de pedidos de teste e 20 conjuntos de pedidos de equipamentos com erro de autenticação. Este teste pretende averiguar a robustez do modelo em distinguir comportamentos fraudulentos e comportamentos de aparelhos com erro. Como nas simulações anteriores, o modelo demonstrou uma grande facilidade em classificar todos os pedidos de um conjunto de pedidos com o mesmo resultado. Assim sendo, o modelo prova não ser robusto para situações de equipamentos com erro de registo visto que, classificou todos os seus conjuntos de pedidos de registo com 100% de pedidos fraudulentos.

Embora os resultados indiquem que o modelo é demasiado simplista na sua análise, este apresenta uma vantagem em relação aos restantes modelos já treinados, uma vez que é bastante rápido a treinar e a classificar.

4.5 Experiência 5 - K-Means

Como já foi explicado na secção 2.3.1.4, o K-Means é um processo de clustering que agrupa objectos em k conjuntos, sendo k um número positivo definido à posteriori. O algoritmo começa por atribuir k centros de forma aleatória. Depois de definidos os centros, o algoritmo percorre todos os objectos e calcula a sua distância em relação a cada centro. Atribui-se o objecto ao conjunto com o centro mais próximo das suas coordenadas. Os centros são recalculados a cada iteração e o algoritmo volta a percorrer todos os objectos até que a sua disposição permaneça inalterada.

O K-Means é um modelo de classificação não supervisionado, ou seja os dados fornecidos ao modelo para o processo de treino não são previamente etiquetados. Ao aplicar um modelo não supervisionado pretende-se obter um método capaz de distinguir os diferentes comportamentos apenas com base em conhecimento adquirido pelo próprio modelo e sem qualquer indicação humana relativamente a pedidos com comportamento normal ou fraudulento. Neste caso o K-Means pretende encontrar e agrupar em k grupos os pedidos de registo mais semelhantes, com o objectivo de encontrar o padrão de um pedido normal e fraudulento.

4.5.1 Processo de Treino

O primeiro passo para iniciar o processo de treino do K-Means passa por obter o número de clusters que acreditamos que existam no nosso conjunto de dados. Segundo a lógica do problema, seria de esperar que existissem dois (pedidos normais e pedidos fraudulentos) ou três clusters, se tivermos em consideração os que os pedidos de equipamentos com erro. No entanto, de forma a obter os melhores resultados possíveis, foi feito um teste de inércia para valores de k entre 1 e 10. A inércia representa a soma do quadrado do erro para cada cluster. Desta forma, quanto menor a inércia mais denso os clusters são. Um método para determinar o melhor número de clusters é identificar qual o valor de k a partir do qual o decréscimo da inércia começa a ser residual. Como podemos ver na figura 4.15, a inércia tem um comportamento decrescente com o aumento do número de clusters. No entanto, a partir dos 3 clusters o valor da inércia decresce com diferenças mais pequenas. Tendo em conta que a escolha não é imediata, foram treinados dois modelos de K-Means, um com 3 e outro com 4 clusters. Embora o gráfico da inércia mostre que obter um modelo de K-Means com 2 clusters não é o mais adequado, foi também criado um modelo com k igual a 2, uma vez que em teoria pretendemos encontrar dois tipos de utilização.

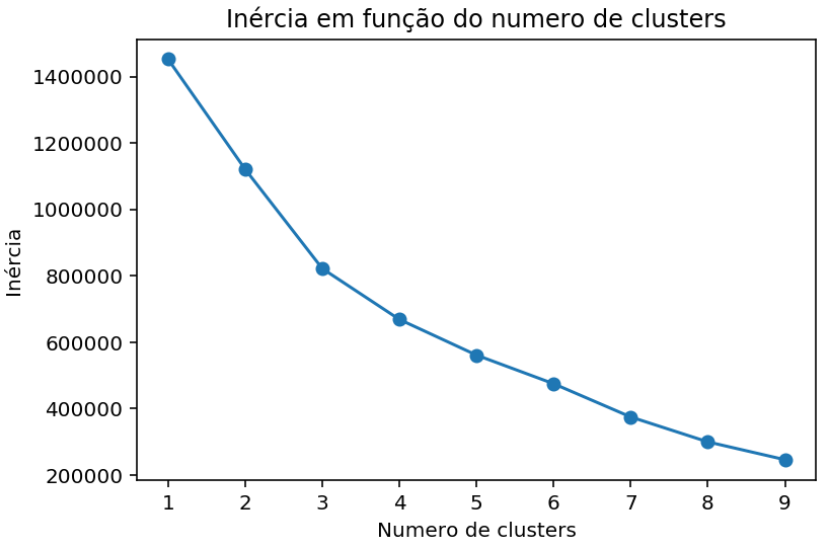
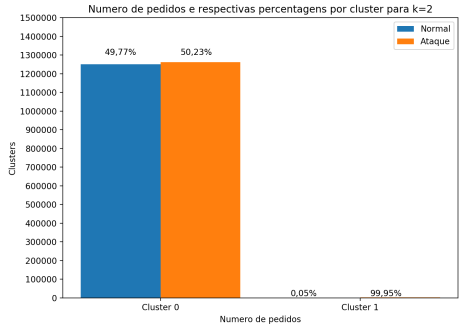
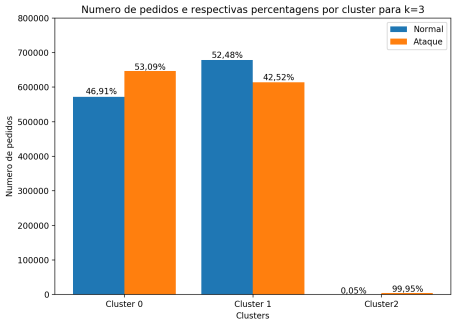


Figura 4.15: Gráfico da inércia para diferentes números de clusters

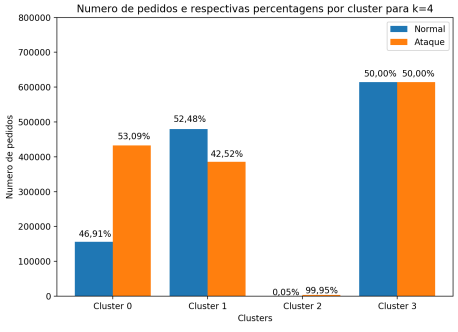
De forma a averiguar a exactidão do algoritmo, os modelos obtidos classificaram todo o conjunto de dados reduzido . Desta forma é possível observar por que clusters os modelos distribuíram os pedidos assinalados como fraude e os pedidos normais.



(a) K-Means com k=2



(b) K-Means com k=3



(c) K-Means com k=4

Figura 4.16: Distribuição dos pedidos normais e fraudulentos presentes no dataset2 pelos diferentes clusters criados pelos modelos de K-Means com k igual a 2, 3 e 4

A figura 4.16 apresenta os resultados dos testes feitos aos modelos com diferentes valores de k. Como podemos observar, nenhum dos modelos propostos é capaz de diferenciar dados normais de dados de ataque. Assim sendo, não é necessário testar o algoritmo em condições reais, uma vez que este não é capaz de aferir os resultados mínimos para ser sequer considerado.

4.6 Experiência 6 - Autoencoder

Como foi explicado na secção 2.3.5.1, um autoencoder é um tipo de rede neuronal capaz de reduzir as dimensões dos dados de entrada. Para tal, a primeira camada e a última camada necessitam de ter o mesmo número de neurónios e a arquitectura da rede deve ser simétrica, de forma a que o número de neurónios das camadas centrais seja inferior o número das camadas periféricas. O objectivo da rede neuronal é obter como output uma reconstrução dos dados de input. Para tal a rede treina de forma a minimizar a média do quadrado do erro entre o input e o output. Um autoencoder bem treinado é capaz de sintetizar as entradas em dimensões inferiores às iniciais, e de seguida reconstruir as informações a partir dos dados sintetizados. Quanto mais semelhante for o output da rede em relação ao input, melhor a sua performance. Este tipo de redes pode ser utilizado para a detecção de outliers se for treinada para reconstruir dados tidos como normais. Desta forma ao analisar outliers a rede não será capaz de reconstruir o input, uma vez que não foi treinada para reconstruir esse tipo de dados. Assim, será possível destingir os outliers, visto que o erro da sua reconstrução será superior ao erro de reconstrução de dados normais.

Esta abordagem pode ser utilizada para a detecção de ataques se a rede em questão for treinada para reconstruir pedidos de registo não fraudulentos. Desta forma, o erro de reconstrução será um indicador de fraude.

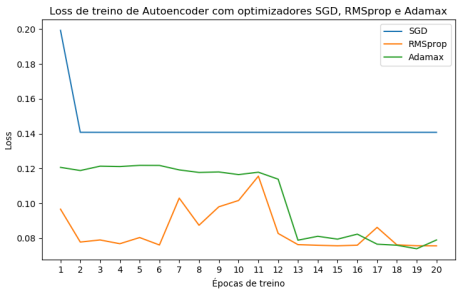
4.6.1 Processo de Treino

4.6.1.1 Escolha do otimizador

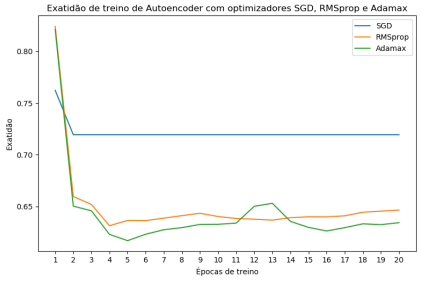
Para explorar esta abordagem foi primeiro feito um teste em relação ao otimizador a usar. Para tal criou-se um autoencoder com 35 dimensões de input, duas hidden layers de 25 e 10 neurónios de cada lado e uma camada central de 3 neurónios. Foram executados 7 treinos de 15 épocas com os otimizadores, Stochastic Gradient Descent, Adam, Adamax, RMSprop, Adadelta, Adagrad e Nadam. Tendo em conta que estamos a lidar com um modelo que pretende reconstruir um input, a loss function usada foi a Mean Squared Error. Foram retirados os valores de exactidão e erro de reconstrução por cada época dos 7 treinos. Ao treinar o modelo com o erro de reconstrução como loss function, o modelo vai evoluir no sentido de minimizar esta métrica. Quanto melhor for a reconstrução de dados normais, pior será a reconstrução de pedidos fraudulentos e consequentemente maior o erro de reconstrução. Tendo em conta que o erro de reconstrução será o valor capaz de

distinguir comportamentos normais e comportamentos fraudulentos, esta métrica torna-se a mais relevante nos processos de treino.

A figura 4.17 apresenta as curvas, do valor da loss function e exactidão, obtidas durante os treinos conduzidos com os otimizadores Stochastic Gradient Descent,RMSprop e Adamax. Como podemos observar, estes otimizadores penalizam o modelo uma vez que a partir da segunda época a curva do erro apresenta grandes dificuldades a encontrar o seu comportamento decrescente. O SGD apresenta os piores resultados, visto que a loss function estabiliza a partir da segunda época. O optimizdor Adamax necessita de 12 épocas até iniciar a minimização do erro, enquanto que o RMSprop chega mesmo a ter um comportamento ascendente. Esta dificuldade em minimizar a loss function tem como consequência a redução e estabilização da exactidão durante as épocas seguintes como podemos também observar na figura 4.17.



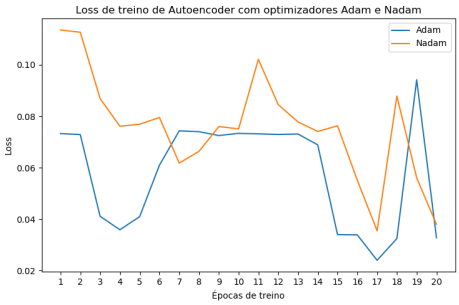
(a) Loss de treino



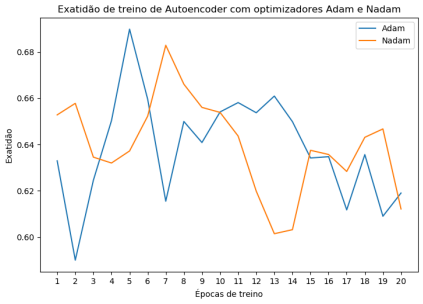
(b) Exatidão de treino

Figura 4.17: Exatidão e Loss por época de treino de Stochastic Gradient Descent,RMSprop e Adamax

A figura 4.18 apresenta a exactidão e o erro de cada época de treino para os otimizadores Adam e Nadam. Como é perceptível pela figura 4.18, embora durante as primeiras 4 épocas a função de erro tenha um comportamento decrescente, é notório que nas épocas seguintes o modelo apresenta bastantes dificuldades a encontrar o mínimo da função. O mesmo acontece com a exactidão. Durante as primeiras épocas o modelo aumenta a sua exactidão até que esta rapidamente tende a cair durante o resto do treino. Os dois otimizadores apresentam comportamentos semelhantes. Embora o Adam tenha obtido o menor valor de erro entre os 5 otimizadores já estudados, não é possível considerar a sua implementação uma vez que o comportamento do erro é totalmente irregular.



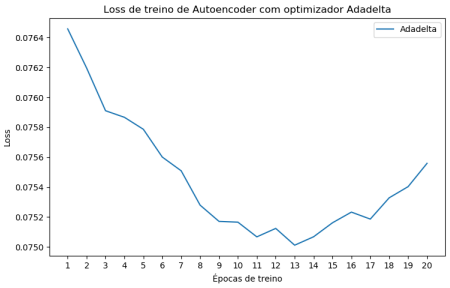
(a) Loss de treino



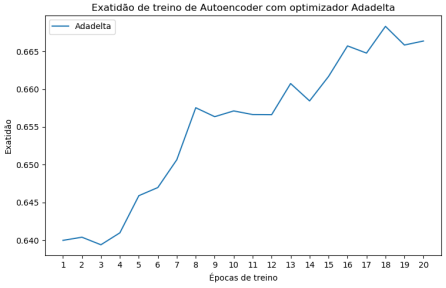
(b) Exatidão de treino

Figura 4.18: Exatidão e Loss por época de treino de Adam e Nadam

Restam por fim os dois otimizadores com melhores resultados. O Adadelta e o Adagrad. A figura 4.19 apresenta o gráfico do erro e da exactidão do Adadelta. O comportamento do erro e da exactidão são consistentes e coincidem com a trajectória esperada para as métricas. O erro apresenta um comportamento decrescente até à 13ª época, onde se encontra o valor mínimo da função que corresponde a 0.075. Embora este não seja o melhor valor de erro, o comportamento da sua curva garante um maior grau de confiança em relação aos outros otimizadores. O gráfico da exactidão tem um comportamento crescente apresentando um máximo na 18ª época de 66,83% de exactidão.



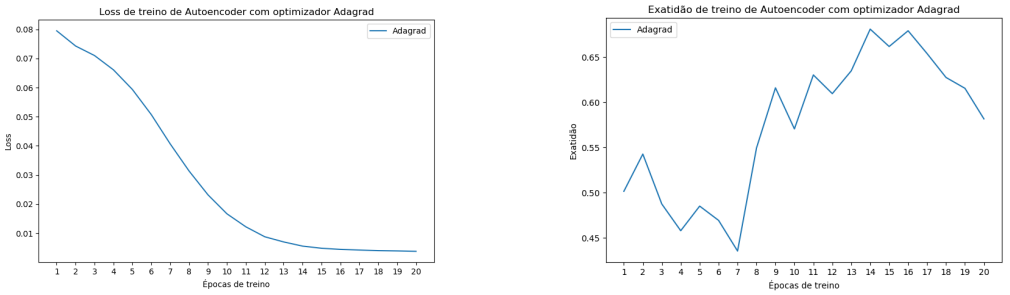
(a) Loss de treino



(b) Exatidão de treino

Figura 4.19: Exatidão e Loss por época de treino de Adadelta

Por último, o otimizador Adagrad apresentou os melhores valores de erro. Embora ao início o modelo tenha apresentado alguma dificuldade a obter melhores valores de exactidão, este foi capaz aprender de modo ao melhorar a exactidão época após época. Desta forma o modelo foi capaz de obter 66,81% de exactidão na 14ª época. No entanto, o ponto forte do otimizador reside no comportamento da curva de erro sendo este o modelo que obteve o valor mais próximo de 0 (0.003791) à 20ª época. Tendo em conta que os treinos tiveram uma duração máxima de 20 épocas, tudo leva a querer que o erro se aproximaria ainda mais de 0 ao aumentar o número de épocas de treino. Desta forma, o otimizador que melhor se adequa ao problema é o Adagrad, assim sendo as restantes simulações foram conduzidas com este otimizador.



(a) Loss de treino

(b) Exatidão de treino

Figura 4.20: Exatidão e Loss por época de treino de Adagrad

4.6.1.2 Escolha da função de activação

O próximo parâmetro do autoencoder a ser avaliado foi a função de activação de cada camada. Os testes iniciais foram feitos com a ReLu por ser uma função simples sem grandes exigências computacionais e por ser resistente a problemas de vanishing gradient. A função é facilmente descrita como:

$$f(x) = max(0, x) \tag{4.1}$$

Como mostra a equação 4.1, a função retorna 0 para valores inferiores a 0 e o próprio valor para valores superiores a 0. Esta função de activação tem-se tornado bastante popular por ser resistente a problemas de vanishing gradient e simples em termos de recursos computacionais. No entanto, esta função de activação tem também os seus defeitos. Um dos problemas desta função, conhecido como "Dying ReLu problem", reside no facto de esta não contemplar um output diferente de 0 para valores negativos. Desta forma, durante o treino, e em consequência da actualização dos pesos por um valor muito elevado do gradiente, os neurónios podem perder a possibilidade de activação, uma vez que todos os valores negativos são representados como 0. Desta forma, o neurónio morre e todos os seus outputs serão 0. Tendo em conta as vantagens e as possíveis consequências de usar a ReLu em todas as camadas da rede, foram feitas simulações com diferentes funções de activação para cada camada.

Com base no trabalho publicado por J.Chen et al [11], foram usadas duas funções de activação para a primeira simulação. Foi utilizada a função Sigmoid para as hidden layers das extremidades e a ReLu para as restantes. A função sigmoid pode ser descrita por:

$$\sigma(x) = \frac{1}{1 - e^{-x}} \tag{4.2}$$

Esta função é capaz de receber qualquer valor como input e retornar um valor entre 0 e 1. No entanto, a função sigmoid, é bastante sensível a problemas de vanishing gradient e não está centrada em 0 o que influencia o sinal do gradiente dos pesos. Embora a função sigmoid tenha as suas desvantagens os autores consideraram que os modelos obtiveram

uma melhor performance com as duas funções. Sabendo que todos os neuronios com ReLu podem "morrer", ao usar duas camadas com sigmoid os autores acreditam que no pior cenário existem pelo menos duas camadas a funcionar. No entanto, a verdadeira razão para o uso da sigmoid com ReLu parte do princípio que o "Dying ReLu problem"acontece quando os gradientes são demasiado altos. Assim sendo, ao usar sigmoid nas camadas exteriores os autores estão a introduzir um problema de vanishing gradient de forma propositada, para a garantir que o gradiente não se torne demasiado elevado, protegendo assim os neurónios com ReLu.

De seguida foram também executadas simulações com duas variantes da função ReLU. A função Elu e a função LeakyReLU.

A função Elu é bastante semelhante à ReLu, sendo que ambas apresentam um função identidade para valores acima de 0. A diferença entre as duas reside nos valores de output na aproximação de 0 por valores negativos. Enquanto para a ReLu qualquer valor negativo origina um output nulo, para a Elu existe um valor mínimo de output para valores de input inferiores a 0, chamado alfa, a partir do qual a função cresce gradualmente até atingir o seu 0.

$$f(x) = \begin{cases} x, x > 0 \\ \alpha(e^x - 1), x <= 0 \end{cases} \tag{4.3}$$

Para a simulação com Elu foi usado $\alpha = 1.0$ para observar se a aproximação que a ReLu tem de 0 tem efeito no processo de treino. O valor $\alpha = 1.0$ foi escolhido tendo em consideração as recomendações dos autores da biblioteca usada para o desenvolvimento do autoencoder (Keras). A função Leaky ReLu parte do mesmo princípio da função ReLu no entanto para valores inferiores a 0 a função apresenta uma recta que passa na origem com declive positivo. O declive é um parâmetro da função e é descrito como alfa.

$$f(x) = \begin{cases} x, x > 0 \\ \alpha x, x <= 0 \end{cases} \tag{4.4}$$

Para a simulação com Leaky ReLu foi usado $\alpha = 0.3$ por este valor ser aconselhado pela literatura da biblioteca usada (Keras)

A evolução da média dos quadrados do erro dos treinos com as diferentes funções pode ser observada na figura 4.22.

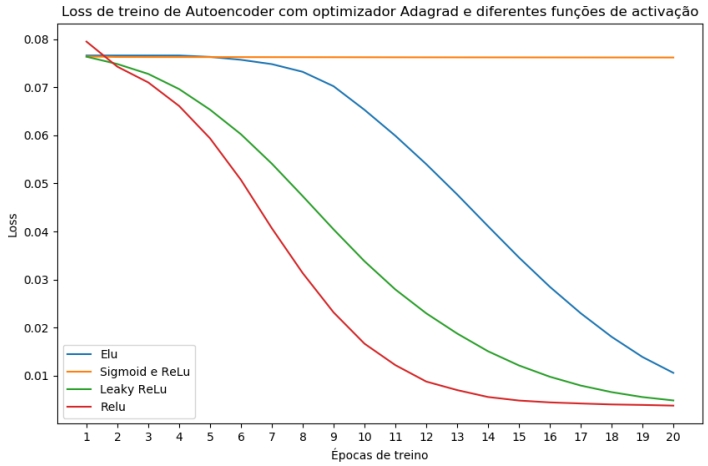


Figura 4.21: Loss de treino do Autoencoder para diferentes funções de activação

Embora a junção da função sigmoid com a função ReLu tenha sido utilizada por J. Chen [11] para reduzir o efeito do "Dying ReLu problem", na simulação efectuada foi, como podemos observar na figura 4.21, altamente prejudicial para o modelo. O seu gráfico da evolução do erro durante o treino é um exemplo perfeito de um problema de vanishing gradients, uma vez que o valor da loss function está a decrescer mas a um ritmo tão baixo que em comparação com as outras funções é praticamente impossível observar a sua evolução. As restantes funções apresentam um comportamento semelhante, no entanto a função com menor valor de erro é a função ReLu com 0.003791. A função Leaky ReLu apresenta resultados muito próximos da ReLu com um mínimo de 0.004863. Desta forma estas duas funções aparentam ser a melhor escolha para o modelo.

De forma a obter a função de activação com melhores resultados fora do processo de treino, os 4 autoencoderes com diferentes funções de activação (Sigmoid e ReLu, Elu, ReLu, Leaky ReLu) foram testados a classificar os 469 conjuntos de pedidos de registo acima descritos como conjunto de teste. Este teste tem como objectivo observar a influencia das funções de activação nos falsos positivos e falsos negativos. Os 4 autoencoders, com diferentes funções de activação, foram sujeitos a 253 conjuntos de pedidos fraudulentos e 216 conjuntos de pedidos normais. O resultado final de cada conjunto de registos é a média do erro de reconstrução dos vários pedidos do seu conjunto. A tabela 4.8 apresenta os melhores resultados de threshold da média do erro de reconstrução por cada um dos 4 autoencoders, assim como os falsos positivos e falsos negativos que resultam da aplicação do threshold escolhido.

Modelo	Sigmoid e ReLu	Elu	ReLu	Leaky ReLu
FP	115	91	36	67
FN	238	91	34	81
Threshold	0,280	0,236	0.2741	0,2322

Tabela 4.8: Tabela de falsos positivos e falsos negativos em função do melhor threshold para os modelos com as diferentes funções de activação

Tendo em conta os resultados das simulações, onde o modelo com a função ReLu apresenta melhores resultados de classificação e, tendo também em conta os valores da media dos quadrados do erro durante o processo de treino. Podemos concluir que com a função ReLu o modelo atinge o mínimo da loss function em menos épocas do que as outras funções e obtém melhores resultados de classificação.

De seguida passou-se para a análise do último parâmetro do modelo, a sua estrutura em número de neurónios e número de camadas.

4.6.1.3 Escolha da arquitectura

De forma a obter o número de neurónios por camada capaz de apresentar melhores resultados, foi necessário treinar 4 tipos arquitecturas diferentes. Cada arquitectura tem duas estruturas diferentes, uma com duas hidden layers e uma com três hidden layers. Desta forma, é possível testar a resposta do modelo aos diferentes números de neurónios e camadas. A primeira arquitectura começa por reduzir os 35 campos de informação de 25 variáveis até 3 variáveis. Para observar o comportamento do modelo ao diferente número de camadas, foram treinados dois autoencoders com esta arquitectura. O primeiro modelo tem uma camada de 25 neurónios em cada extremidade e uma camada de 3 neurónios no centro. O segundo modelo apresenta uma camada de 25 neurónios assim como uma camada de 10 em cada extremidade e uma camada central de 3 neurónios.

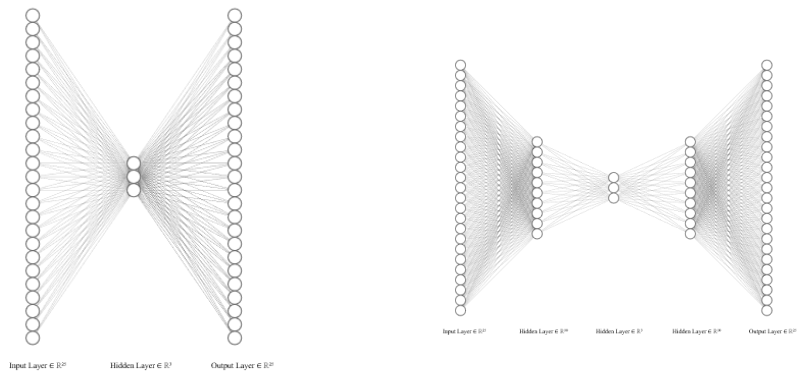


Figura 4.22: Esquema do número de neurónios e camadas dos modelos da arquitectura 1

A segunda arquitectura reduz os 35 campos de informação de 35 variáveis até 10 variáveis. Foram treinados dois autoencoders com esta arquitectura. O primeiro com uma

camada de 35 neurónios em cada extremidade e uma camada de 10 neurónios no centro. O segundo apresenta uma camada de 35 neurónios assim como uma camada de 25 em cada extremidade com 10 neurónios no centro.

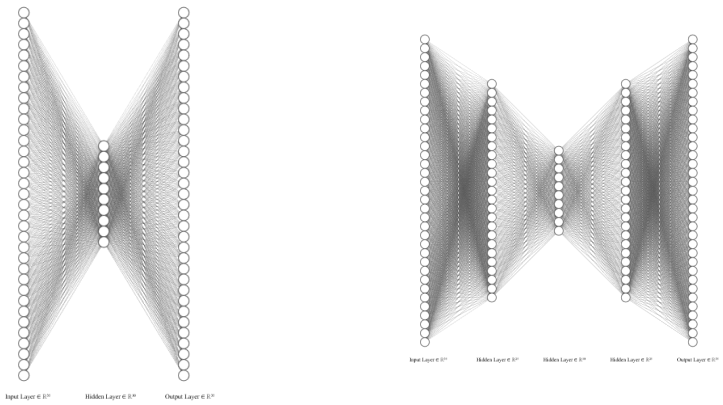


Figura 4.23: Esquema do número de neurónios e camadas dos modelos da arquitectura 2

A terceira arquitectura é em tudo semelhante à segunda arquitectura com excepção do número de neurónios centrais que em vez de 10 passam a ser 15.

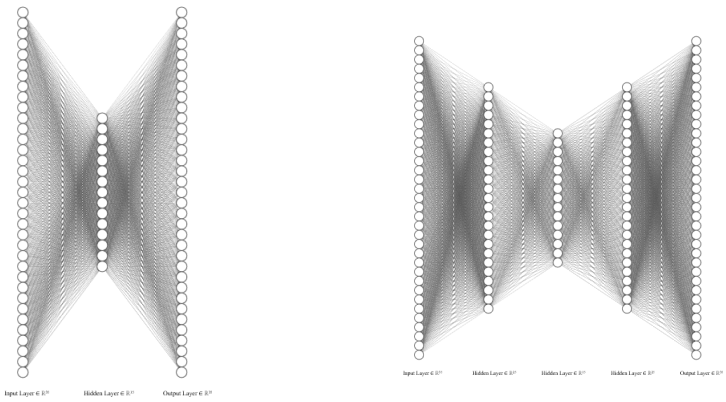


Figura 4.24: Esquema do número de neurónios e camadas dos modelos da arquitectura 3

A quarta arquitectura reduz os 35 campos de informação de 18 variáveis até 9 variáveis. Foram treinadas dois autoencoders com esta arquitectura. O primeiro com uma camada de 18 neurónios em cada extremidade e uma camada de 9 neurónios no centro. O segundo apresenta uma camada de 18 neurónios assim como uma camada de 14 em cada extremidade com 9 neurónios no centro.

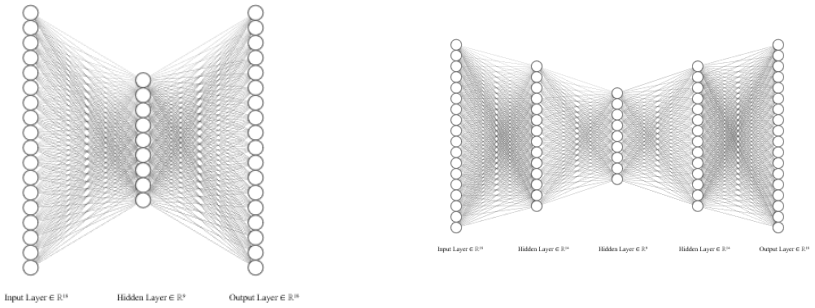


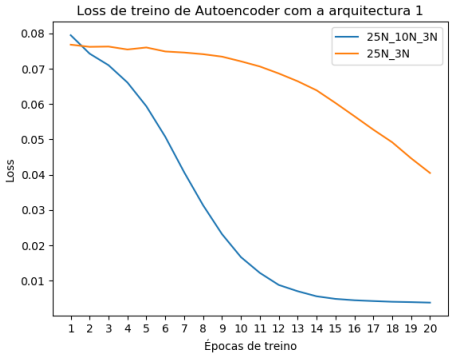
Figura 4.25: Esquema do número de neurónios e camadas dos modelos da arquitectura 4

Cada autoencoder foi treinado durante 20 épocas com o conjunto de dados original. Durante o treino foram recolhidos os valores da média dos quadrados dos erros que foi utilizada como loss function da rede. A figura 4.26 apresenta a curva da loss function de treino das 4 arquitecturas. Ao observar o comportamento da loss function por arquitectura foi possível concluir que a camada extra apenas apresenta uma grande influencia no processo de aprendizagem da arquitectura 1 e 4. Para as arquitecturas 2 e 3 o decréscimo do erro de treino apresenta um comportamento muito semelhante mesmo com a camada extra. Este fenómeno pode ser explicado através da diferença entre o número de neurónios centrais de cada arquitectura. Embora haja diferenças no processo de aprendizagem dos dois modelos da arquitectura 4, estas não são tão significativas quando comparadas com as do processo de treino da arquitectura 1. As arquitecturas com mais neurónios na camada central apresentam uma maior facilidade na reconstrução dos dados de input uma vez que a dimensão de reconstrução é maior e consequentemente a perda de informação menor. Assim sendo as arquitecturas com menos neurónios na camada central necessitam de processos adicionais de reconstrução oferecidos pela camada extra.

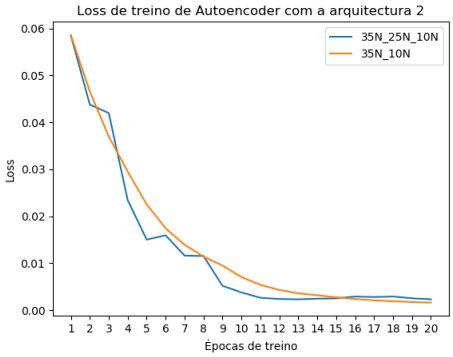
As arquitecturas número 2 e 3 apresentam os melhores valores de erro. Embora o valor final do erro depois de 20 épocas de treino seja semelhante, a arquitectura número 3 apresenta o processo de aprendizagem mais rápido do que as restantes.

De forma a observar qual a arquitectura capaz de obter menores valores de erro para conjuntos de pedidos normais e maiores valores de erro de reconstrução para os conjuntos de pedidos fraudulentos, foram feitos testes às 4 arquitecturas com os 253 conjuntos de pedidos fraudulentos e 216 conjuntos de pedidos normais. A tabela 4.9 apresenta os melhores resultados para threshold e Area under ROC curve (AUC) por modelo, assim como os falsos positivos e falsos negativos que resultam da aplicação do threshold escolhido.

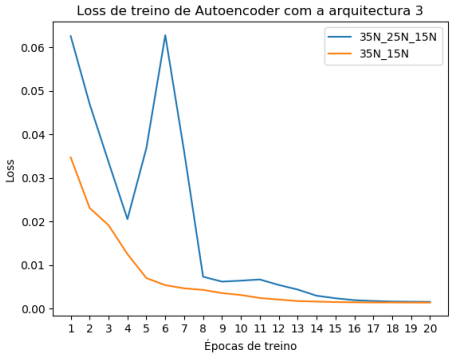
Como explicado na secção 4.2.1 a curva de operação do receptor ou Receiver Operating Characteristic (ROC) é a curva que representa o desempenho de um classificador binário em relação aos falsos positivos e positivos verdadeiros. A Area under ROC curve (AUC) é como o nome indica a área da curva ROC com máximo de 1 e mínimo de 0. Assim sendo quanto menor o número de falsos positivos e maior o número de positivos verdadeiros maior será a área debaixo da curva ROC. Desta forma quanto maior o valor de AUC,



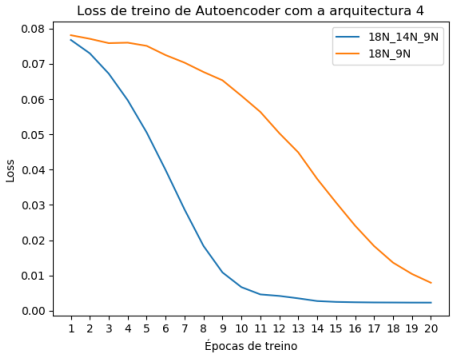
(a) Arquitectura 1



(b) Arquitectura 2



(c) Arquitectura 3



(d) Arquitectura 4

Figura 4.26: Média do quadrado dos erros dos processos de treino dos autoencoders das 4 arquitecturas

melhor é a capacidade de classificação do modelo em estudo.

Arquitectura	**1.1**	**1.2**	**2.1**	**2.2**	**3.1**	**3.2**	**4.1**	**4.2**
AUC	0.715	0.857	0.583	0.708	0.677	0.605	0.898	0.413
FP	92	36	113	85	92	107	25	125
FN	93	34	120	87	93	108	26	129
Threshold	0.238	0.274	0.178	0.174	0.152	0.199	0.239	0.211

Tabela 4.9: Tabela de falsos positivos, falsos negativos e AUC em função do melhor threshold para os modelos com as diferentes arquitecturas

Tendo em conta que a arquitectura 4.1 apresentou o valor mais alto de AUC, assim como a menor relação de falsos positivos e falsos negativos, conclui-se que a melhor arquitectura para o problema passa por uma camada de 18 neurónios em cada extremidade e uma camada de 9 neurónios no centro do autoencoder.

4.6.2 Classificação do Conjunto de Teste

Com base nas simulações efectuadas podemos concluir que o modelo com a arquitectura 4.1, com ReLu como função de activação em todas as camadas, com Adagrad como otimizador apresenta os melhores resultados para o problema em questão. Assim sendo, o modelo eleito foi submetido ao teste de detecção de equipamentos com erro de modo a aferir a sua capacidade de distinção entre fraude e equipamentos com erro. Os resultados obtidos foram expostos na figura 4.27 onde podemos observar que todas os conjuntos de pedidos de equipamentos com erro de registo foram classificadas como fraude.

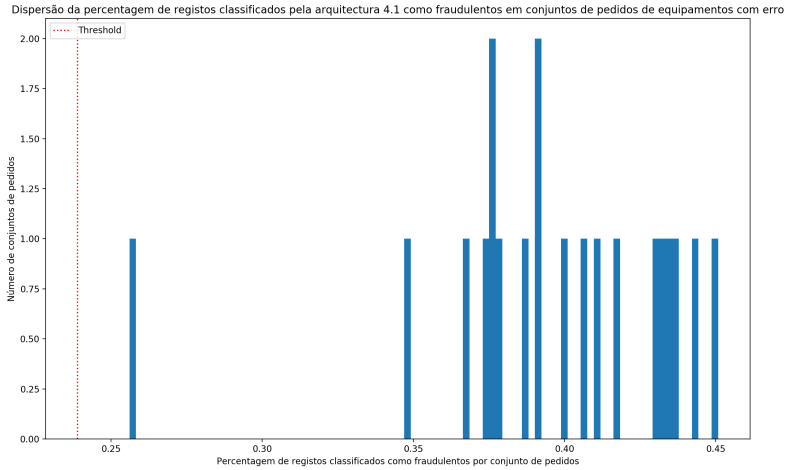


Figura 4.27: Dispersão da percentagem de registos classificados pela arquitectura 4.1 como fraudulentos em conjuntos de pedidos de equipamentos com erro

A falta de capacidade do modelo em destingir comportamentos fraudulentos de comportamentos de equipamentos com erros de registo é explicável pelo facto do modelo ser capaz de detectar anomalias e não casos de fraude. Isto é, o modelo foi treinado para reconstruir comportamentos normais, a partir do momento em que o modelo não é capaz de reconstruir um comportamento com um erro inferior ao threshold esse comportamento é anormal mas não necessariamente fraudulento. Assim sendo, o modelo é incapaz de distinguir o motivo da anomalia, apenas conclui que algo não é normal. Esta abordagem apresenta a desvantagem de não ser capaz de classificar conjuntos de pedidos de equipamentos com erro de registo no entanto, ao contrário dos restantes modelos propostos, o autoencoder é teoricamente capaz de detectar tipos de fraude nunca antes vistos, uma vez que apenas conhece o padrão de um comportamento normal.

4.7 Experiência 7 - Fully Connected Neural Networks

As redes neuronais podem ter as mais variadas aplicações dependendo dos seus parâmetros como, estrutura ou funções de activação. Desta forma as redes neuronais podem

ser usadas em processos de classificação supervisionada quando construídas com esse propósito.

A função de activação softmax recebe um vector com n números reais e normaliza o seu input numa distribuição de probabilidades proporcionais às exponenciais dos números de entrada. Ao contrário das outras funções de activação, a softmax toma em consideração todos os inputs de uma camada e não apenas o input de neurónio em neurónio. A função apresenta como resultado final uma distribuição de probabilidades para a camada em questão, assim sendo o processamento do resultado de cada nó depende dos inputs dos restantes. A função devolve valores entre 0 e 1, desta forma a soma dos n valores que a função devolve é sempre 1. Tendo em conta que a função softmax é capaz de criar uma distribuição de probabilidades para n elementos, podemos aplicar esta função na última camada de uma rede neuronal com o número de neurónios igual ao número de classes do problema. Desta forma, a distribuição de probabilidades nos neurónios da última camada será na realidade uma distribuição de probabilidades em relação às classes, sendo que cada neurónio representará uma classe.

Este tipo de rede neuronal pode ser usada como solução ao problema apresentado pela operadora. Para tal a última camada apresenta dois neurónios, um neurónio para a classe de comportamento normal e um neurónio para a classe de comportamento fraudulento. Sabendo que será usada uma função softmax na última camada, os resultados do modelo serão probabilidades de um pedido de registo pertencer a uma das classes.

Desta forma, última camada irá retornar dois valores entre 0 e 1 por pedido. Todos os pedidos de registo, do dataset, estão previamente classificados por um vector que contém um 0 e um 1, sendo que a posição do valor 1 indica a classe do pedido de registo. Assim sendo podemos usar o valor da entropia cruzada para aferir a performance e evolução da rede durante o seu processo de treino. Se S(y) representar os resultados da rede para um pedido de registo, e L(y) a sua classificação prévia. Podemos calcular a entropia cruzada D(S,L), da seguinte forma:

$$S(y) = \begin{bmatrix} 0.8 \\ 0.2 \end{bmatrix} L(y) = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{4.5}$$

$$D(S,L) = -\sum_{i} L_i log(S_I) \tag{4.6}$$

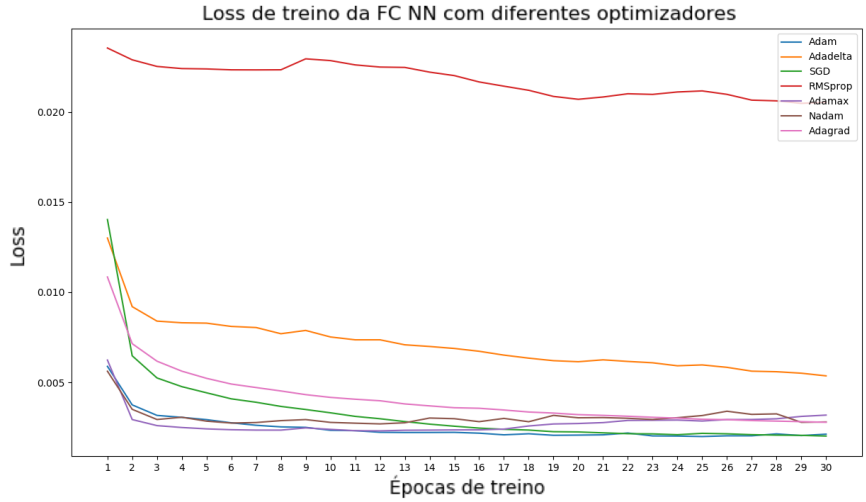
Tendo em conta que a softmax dá um resultado possível de ser comparado com a classificação prévia através da entropia cruzada, usou-se a este método de comparação como loss function para o treino da rede neuronal.

4.7.1 Processo de Treino

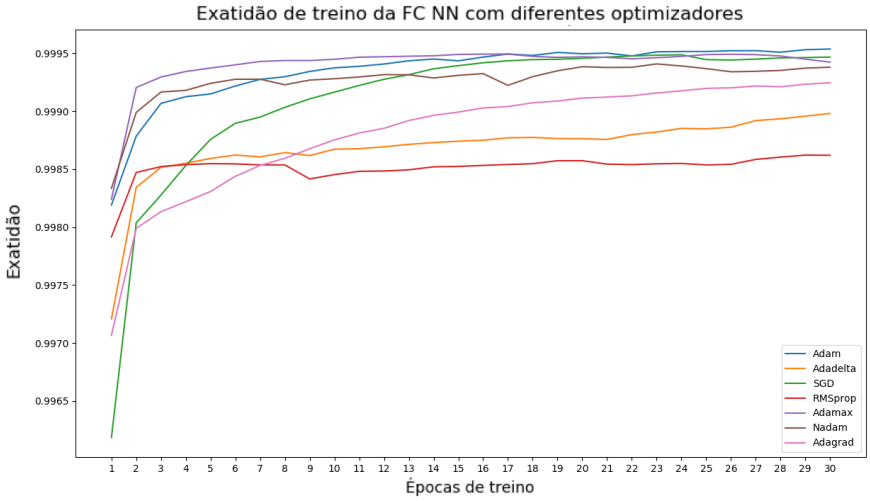
4.7.1.1 Escolha do otimizador

De seguida foram feitos 7 treinos de 30 épocas com os otimizadores, Stochastic Gradient Descent, Adam, Adamax, RMSprop, Adadelta, Adagrad e Nadam com o objectivo de aferir

qual o melhor otimizador para o modelo em questão. Para o teste de otimizadores, foi considerada uma estrutura decrescente com duas hidden layers, uma com 35 neurónios e outra com 19 neurónios, ambas com ReLu como função de activação. Foi também introduzida uma dropout layer para reduzir a probabilidade de overfitting, uma vez que o processo de treino envolve uma grande quantidade de dados bastante semelhantes. Os resultados de loss e exatidão obtidos durante o processo de treino dos diferentes otimizadores pode ser analisado na figura 4.28.



(a) Loss de treino



(b) Exatidão de treino

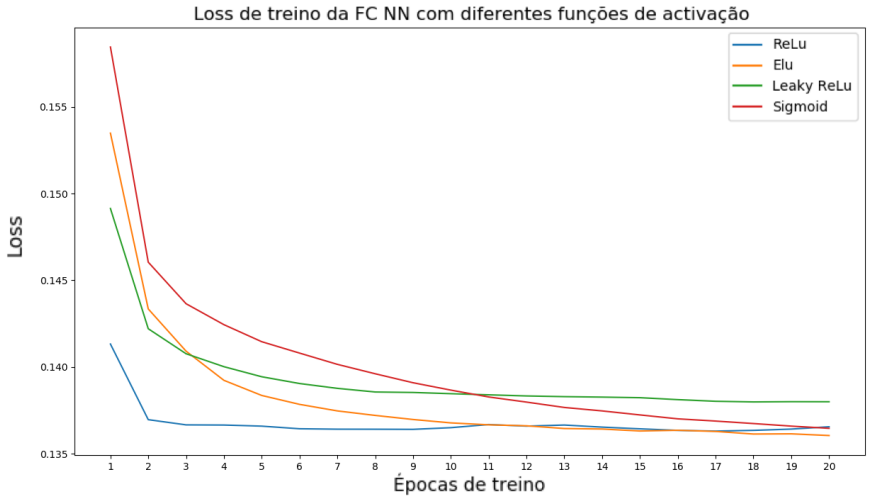
Figura 4.28: Gráfico da evolução da loss e exatidão durante os processos de treino com diferentes otimizadores

Como podemos concluir com o auxílio da figura 4.28, o otimizador Adamax apresenta resultados muitos semelhantes ao Adam. No entanto, o Adamax atinge os valores óptimos

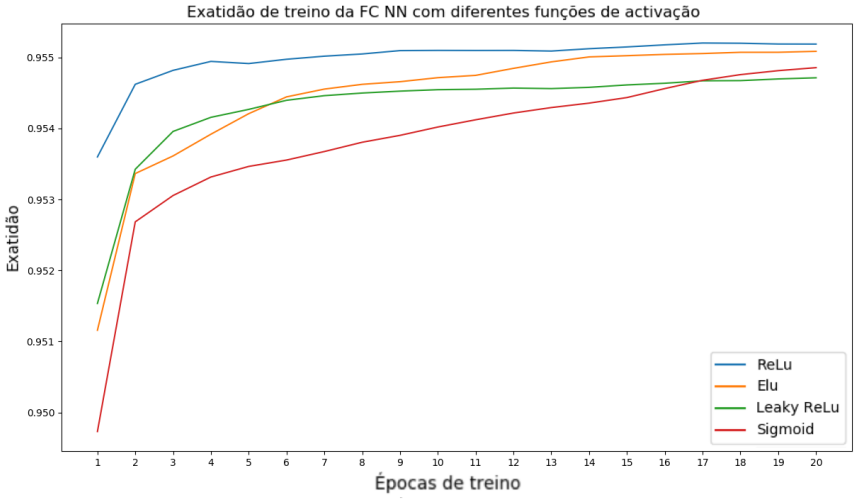
com menos de 10 épocas de treino o que apresenta uma enorme vantagem, uma vez que torna o processo de treino mais rápido. Os restantes otimizadores apresentam resultados que se aproximam dos resultados obtidos pelo Adam e Adamax no entanto necessitam de mais épocas de treino para resultados inferiores. O otimizador RMSprop demarcou-se como o pior otimizador para o problema apresentado uma evolução da loss function muito lenta o que impediu o modelo de ter valores próximos dos obtidos pelos restantes otimizadores em 30 épocas de treino.

4.7.1.2 Escolha da função de activação

De seguida foram efectuadas 4 simulações com o objectivo de testar qual a melhor função de activação para as hidden layers do modelo. Para tal usou-se o Adamax como otimizador e criou-se 4 modelos com a mesma arquitectura usada na escolha do otimizador. As 4 funções de activação testadas foram: ReLu, Leaky ReLu, Elu e Sigmoid. A figura 4.29 apresenta a evolução da exatidão e da entropia cruzada (loss funtion) durante as 20 épocas de treino dos 4 modelos. Como podemos observar, todos os modelos apresentam uma evolução decrescente da entropia cruzada, no entanto o modelo com ReLu obtém os menores valores de entropia e destaca-se dos restantes por uma aprendizagem mais rápida. Ao analisar a evolução da exatidão durante o processo de treino, podemos também concluir que o modelo com ReLu optem melhores resultados a uma velocidade muito superior aos restantes modelos. Na realidade, para os modelos com Elu e ReLu, os valores de exatidão e entropia cruzada da 30ª época são bastante semelhantes, no entanto escolheu-se o modelo com a função ReLu uma vez que este atinge os melhores valores de exatidão e entropia em menos de 10 épocas.



(a) Exatidão de treino



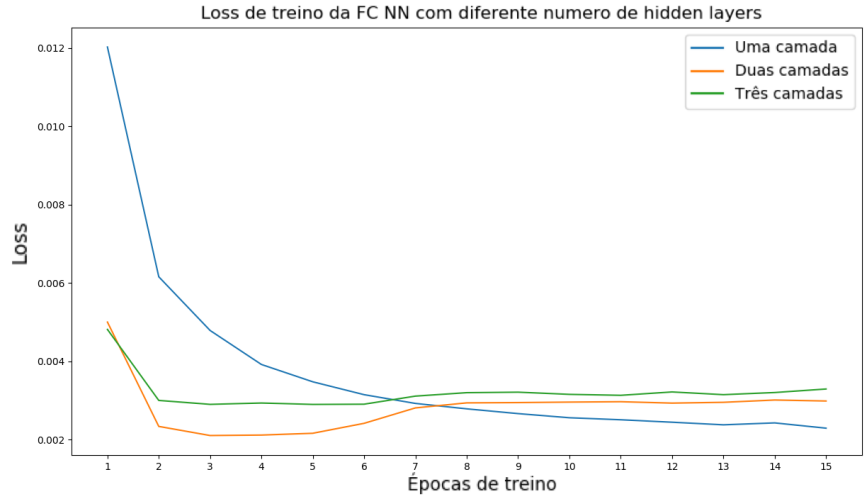
(b) Exatidão de treino

Figura 4.29: Gráfico da evolução da loss e exatidão durante os processos de treino com diferentes funções de activação

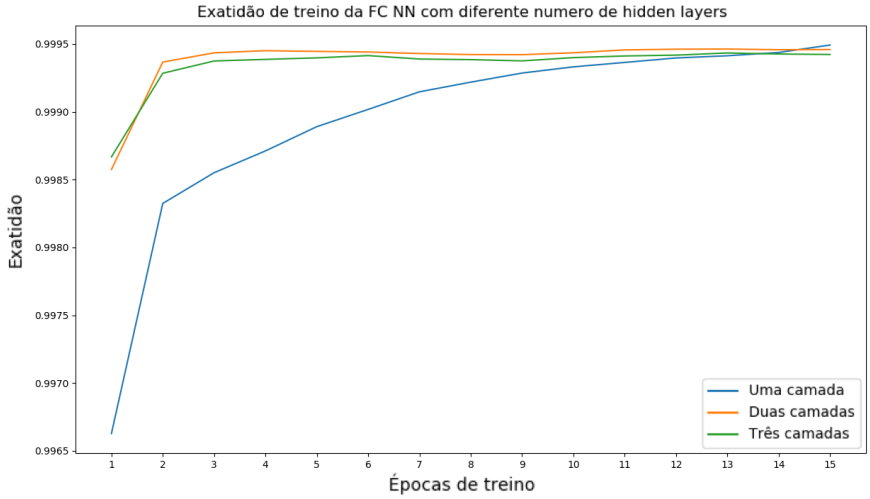
4.7.1.3 Escolha da arquitectura

Por fim foram feitas simulações de modo a encontrar a arquitectura de camadas e número de neurónios por camada capaz de apresentar os melhores resultados. Tendo em conta que todas as simulações anteriores foram feitas com uma arquitectura de duas hidden layers, uma com 35 neurónios e uma com 19 neurónios, começou-se por analisar as consequências da segunda camada de 19 neurónios nos resultados da simulação. Para tal foram criados 3 modelos com 35 neurónios por cada camada. Cada modelo apresenta uma camada a mais do que o anterior. Desta forma será possível aferir as consequências da introdução de cada camada. Foram retirados os valores de exatidão e entropia cruzada

dos 3 treinos durante 15 épocas. A figura 4.30 apresenta a evolução dos valores de treino para as 3 arquitecturas de uma, duas e três camadas.



(a) Loss de treino



(b) Exatidão de treino

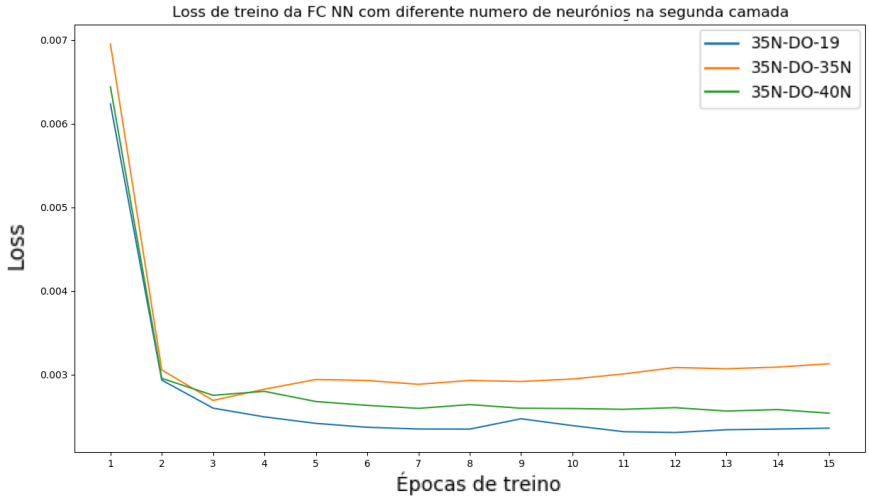
Figura 4.30: Gráfico da evolução da loss e exatidão durante os processos de treino com diferentes funções de activação

Como podemos observar a introdução de uma segunda e terceira camada acabam por apresentar os mesmos resultados finais, no entanto com mais do que uma camada o processo de treino é acelerado, uma vez que os valores óptimos são atingidos de forma mais imediata em comparação com o modelo de uma camada. Desta forma podemos concluir que o modelo beneficia da introdução de uma segunda ou terceira camada.

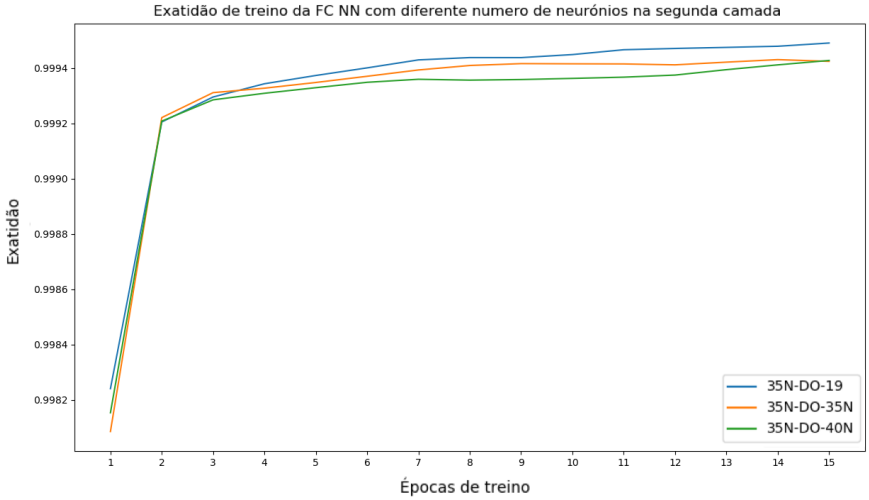
Entre os modelos de duas e três camadas as diferenças são residuais, no entanto por cada camada acrescentada acresce as exigências computacionais de treino. Assim sendo

a introdução da terceira camada acaba por ser desnecessária. Partindo do princípio que a terceira camada era desnecessária, partiu-se para o estudo do número de neurónios da segunda camada.

Foram efectuadas 3 simulações com diferentes números de neurónios da segunda camada. O primeiro modelo apresenta 19 neurónios na segunda camada, o segundo modelo apresenta 35 neurónios e por fim o terceiro apresenta 40 neurónios na segunda camada. Foi introduzida uma camada Dropout entre a primeira e a segunda camada dos três modelos, para defender os resultados de um possível problema de overfitting causado pela grande quantidade de dados semelhantes no dataset. A figura 4.31 apresenta a evolução dos valores de exatidão e entropia cruzada dos três processos de treino.



(a) Loss de treino



(b) Exatidão de treino

Figura 4.31: Gráfico da evolução da loss e exatidão durante os processos de treino com diferentes números de camadas

Como podemos observar na figura 4.31, entre as três arquitecturas, com diferentes números de neurónios, a arquitectura com 19 neurónios na segunda camada apresenta os melhores resultados de treino.

Todos os modelos testados apresentam resultados um pouco semelhantes, no entanto o modelo com duas camadas de 35 neurónios apresenta os melhores resultados de treino para apenas 5 épocas treino.

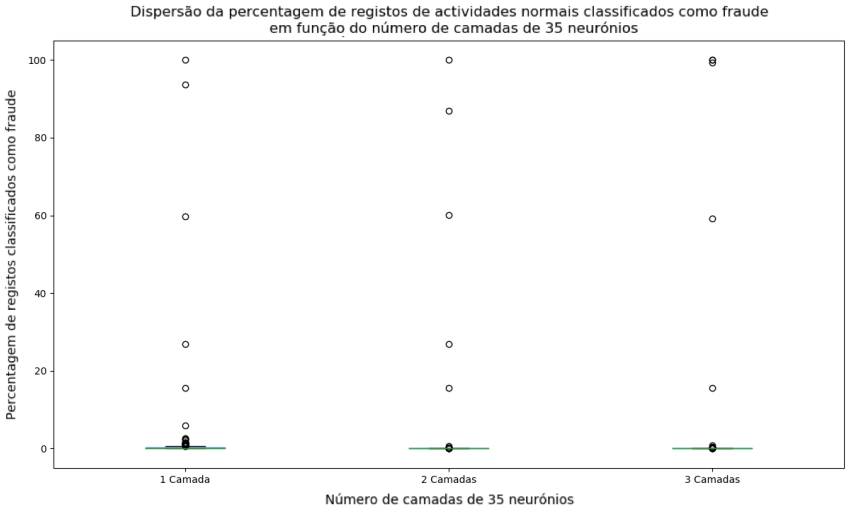
4.7.2 Classificação do Conjunto de Teste

À semelhança das outras experiências, o modelo foi testado ao classificar o conjunto de dados que representam os 469 grupos de pedidos de registo. A classificação final do comportamento de um IP é obtida a partir da sua percentagem de pedidos classificados como fraudulentos. Foram recolhidas 469 percentagens que o modelo retornou ao analisar cada um dos 253 grupos de pedidos fraudulentos e 216 conjuntos de pedidos normais e 20 conjuntos de pedidos de equipamentos com erro. A tabela 4.10 apresenta os melhores resultados dos testes efectuados para as arquitecturas de 35 neurónios por camada sem camada dropout.

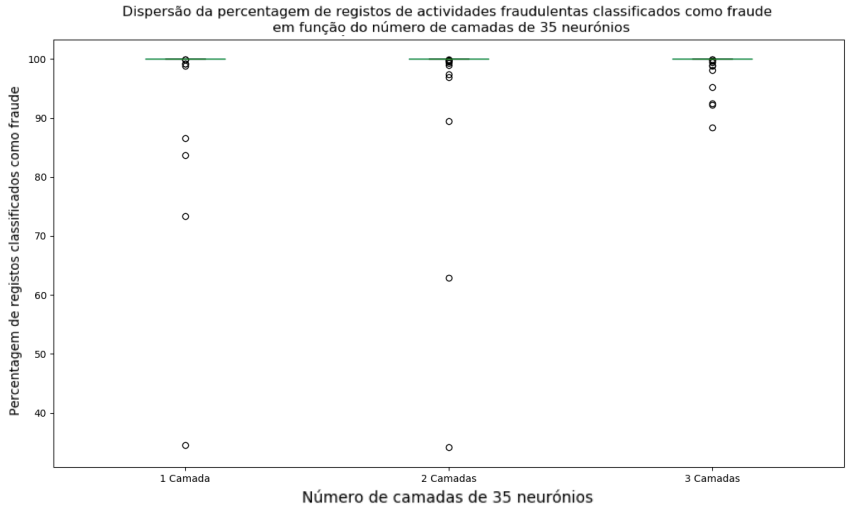
número de camadas	**FN**	**FP**	**AUC**	**Threshold**	**Equipamentos com erro mal classificados**
1	0	3	0.9975	34%	100%
2	0	3	0.997	34%	100%
3	0	3	0.995	88%	100%

Tabela 4.10: Tabela de falsos positivos, falsos negativos, AUC e erro de classificação de equipamentos com erro em função do melhor threshold para os modelos com os diferentes números de camadas

Embora os modelos não sejam capazes de classificar devidamente os conjuntos de pedidos de equipamentos com erro, os resultados de exactidão são bastante positivos. A precisão também se encontra bastante alta como podemos observar no gráfico da figura 4.32 que apresenta a distribuição dos resultados finais dos modelos ao classificar os 253 conjuntos de pedidos fraudulentos e 216 conjuntos de pedidos normais.



(a) Dispersão de percentagens de pedidos classificados como fraudulentos para conjuntos de pedidos normais



(b) Dispersão de percentagens de pedidos classificados como fraudulentos para conjuntos de pedidos fraudulentos

Figura 4.32: Gráficos da dispersão da percentagens de registos classificados como fraudulentos em conjuntos de pedidos fraudulentos e conjuntos de pedidos normais em função do número de camadas com 35 neurónios.

Como podemos observar na figura 4.32 a), a grande maioria dos conjuntos de pedidos normais obtêm um resultado final próximo dos 0% de registos classificados como fraude. O modelo com três camadas apresenta uma precisão superior aos restantes, no entanto este modelo classifica três conjuntos de pedidos normais com uma percentagem de registos fraudulentos superior a 88%. Desta forma o seu threshold máximo torna-se bastante superior aos restantes para o mesmo número de falsos positivos. Ao observar o gráfico

da figura 4.32 b) podemos também concluir que a grande maioria dos conjuntos de pedidos fraudulentos obtêm um resultado final próximo dos 100% de registos classificados como fraude. Mais uma vez o modelo de três camadas apresenta uma precisão superior aos restantes, no entanto nenhum dos modelos apresenta uma percentagem de registos fraudulentos por conjunto de pedidos fraudulentos inferior a 34%.

O mesmo processo de teste foi efectuado para os modelos com camada dropout e diferentes números de neurónios na segunda camada. A tabela 4.11 apresenta os melhores resultados dos testes efectuados para as arquitecturas de duas camadas, com camada dropout e diferentes números de neurónios na segunda camada.

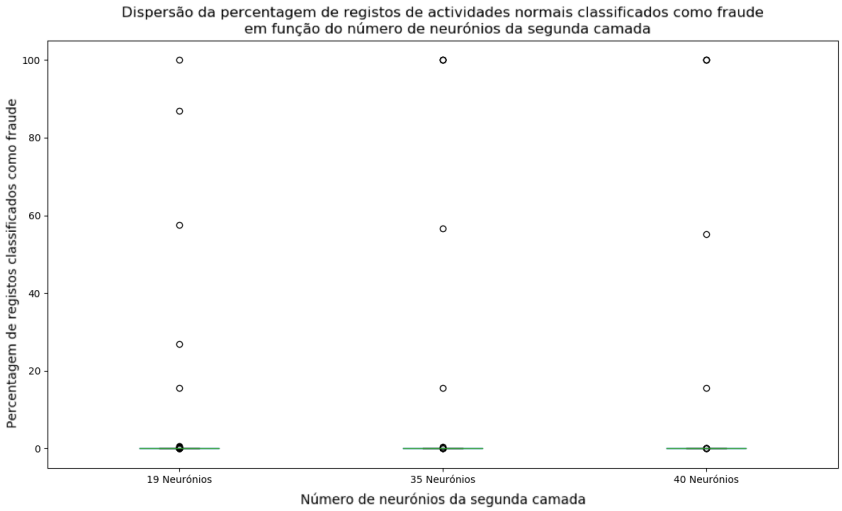
número de neurónios da segunda camada	**FN**	**FP**	**AUC**	**Threshold**	**Equipamentos com erro mal classificados**
19	0	2	0.998	67%	100%
35	0	3	0.993	83%	75%
40	0	3	0.993	99%	70%

Tabela 4.11: Tabela de falsos positivos, falsos negativos, AUC e erro de classificação de equipamentos com erro em função do melhor threshold para os modelos com os diferentes números de neurónios na segunda camada

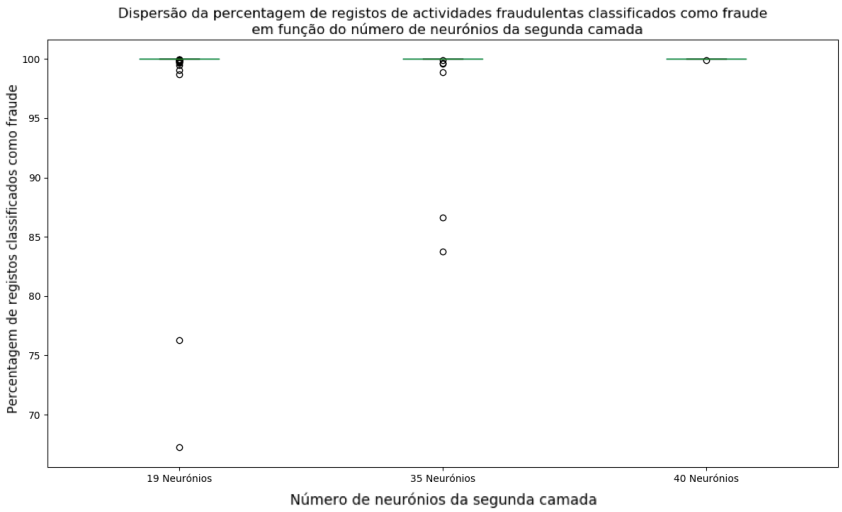
Com a introdução da camada de dropout, torna-se possível classificar 25% dos conjuntos de pedidos de equipamentos com erro como não fraudulentas. Os resultados da exactidão e AUC são bastante positivos para o problema.

A figura 4.33 apresenta as percentagens de pedidos classificados como fraudulentos por conjunto de pedidos de registo obtida pelos modelos ao classificar os 253 conjuntos de pedidos fraudulentos e 216 conjuntos de pedidos normais.

Mais uma vez todos os modelos apresentam percentagens de pedidos de registo normais classificados como fraudulentos bastante próximos de 0%, com excepção de alguns outliers. O mesmo acontece para os pedidos de conjuntos fraudulentos, cuja esmagadora maioria foram devidamente classificados. Dentro da precisão e exatidão, o modelo com 40 neurónios na segunda camada destaca-se dos restantes, uma vez que a média da percentagem de pedidos devidamente classificados como fraudulentos por conjunto fraudulento é de 99.9996%. Dos 253 conjuntos de pedidos fraudulentos classificados pelo modelo de 40 neurónios, apenas um obteve menos do que 100% de pedidos classificados como tal. Mesmo o único conjunto de pedidos que não obteve os 100% de exactidão apresentou 99.91% de exactidão.



(a) Dispersão de percentagens de pedidos classificados como fraudulentos para conjuntos de pedidos normais



(b) Dispersão de percentagens de pedidos classificados como fraudulentos para conjuntos de pedidos fraudulentos

Figura 4.33: Gráficos da dispersão da percentagens de registos classificados como fraudulentos em conjuntos de pedidos fraudulentos e conjuntos de pedidos normais em função do número de neurónios da segunda camada.

Para os conjuntos de pedidos normais o modelo de 40 neurónios apenas classificou cinco com percentagens superiores a 0%. É importante destacar que três dos cinco conjuntos de pedidos normais com percentagens superiores a 0% também foram assim classificados pelos restantes modelos. A precisão do modelo de 40 neurónios permite estabelecer o threshold a 100% de pedidos classificados como fraude. Com o threshold nos 100% o modelo obteve 1 falso negativo com 0 falsos positivos e foi ainda capaz de classificar devidamente todos os conjuntos de pedidos de equipamentos com erros.

4.8 Conclusões Experimentais

Ao fim das 7 experiências é seguro concluir que pelos métodos apresentados é possível classificar com exatidão e precisão a esmagadora maioria dos pedidos fraudulentos presentes nos dados de teste e treino.

A abordagem supervisionada por K-Nearest Neighbors demonstrou-se extremamente exacta tendo obtido 0 falsos negativos e 3 falsos positivos em todo o conjunto de 469 grupos de pedidos de registo de treino. Embora seja bastante exacto, o modelo não foi capaz de classificar devidamente os equipamentos com erro presentes no conjunto de dados de teste. Ainda para mais, o tempo de treino e de execução das previsões não é de todo favorável para um modelo que se pretende em tempo real.

A segunda abordagem por regressão logística também apresentou excelentes resultados de exactidão, precisão e tempo de execução. No entanto, o modelo não foi capaz de classificar um único aparelho com erro como não fraudulento.

O terceiro modelo criado através de uma Support Vector Machine, apresentou excelentes resultados de treino, no entanto as capacidades apresentadas pelo modelo durante a fase de treino não se mantiveram quando sujeito ao conjunto de dados de teste. Quando deparado com o conjunto de teste o modelo obteve 4 falsos negativos e 5 falsos positivos. É importante relembrar que para a operadora a existência de falsos negativos representa uma grande perda na funcionalidade do projecto. Para além dos 4 falsos negativos, o modelo foi incapaz de classificar devidamente comportamentos de registo provenientes de equipamentos com erro. Por fim, o tempo de treino de 10 horas e o tempo de execução das previsões impedem o modelo de ser operacionalizado para o projecto em questão.

A quarta experiência recorreu a uma árvore de decisão para classificar de forma supervisionada pedidos normais e fraudulentos. A exactidão e precisão do modelo demonstraram-se acima da média dos restantes modelos, visto que os conjuntos de pedidos não fraudulentos tiveram percentagens de classificação de pedidos fraudulentos concentradas nos 0% e os conjuntos de pedidos fraudulentos nos 100%. Para além da precisão, o modelo apresentou 3 falsos positivos e 0 falsos negativos. Por fim, o modelo foi capaz de classificar como normal um conjunto de pedidos de um equipamento com erros e, tanto o treino como a execução das previsões do modelo foram bastante rápidas. No entanto, a estrutura de decisão do modelo é bastante simplista dependendo apenas de duas condições. Embora os resultados sejam superiores aos restantes modelos, a forma como a decisão é tomada pode ser demasiado simplista para comportamentos de registo que no futuro possam ter um comportamento ligeiramente diferente do então estudado.

A quinta experiência teve como intuito estudar os pedidos de registo de uma forma não supervisionada, de forma a garantir uma possível actualização do modelo em tempo real na operadora. Para tal foi utilizado o K-Means. Depois de três simulações com o objectivo de dividir os pedidos de registo em grupos de pedidos normais e fraudulentos não foram obtidos resultados satisfatórios. O modelo mostrou-se incapaz de criar clusters onde a maioria relevante dos pedidos pertencesse a uma só classe.

A sexta abordagem pretende utilizar um autoencoder para reconhecer e reconstruir pedidos de registo normais. Os resultados obtidos pelo modelo escolhido apresentaram-se abaixo da média dos restantes modelos, uma vez que este obteve 25 falsos positivos, 26 falsos negativos e 0 equipamentos com erro classificadas como normais. Embora os resultados sejam muito inferiores aos restantes, a abordagem apresenta outro tipo de vantagens como a capacidade de detectar tipos de ataque nunca antes vistos durante o processo de treino e, o tempo de execução.

Por fim, na sétima experiência foi treinada uma Fully connected neural network com duas camadas e uma camada dropout para evitar questões de overfitting. O modelo com 35 neurónios na primeira camada e 40 neurónios na segunda apresentou os melhores resultados entre todos os modelos anteriormente referidos. Este modelo obteve 0 falsos negativos, 3 falsos positivos e classificou devidamente 30% dos conjuntos de pedidos de equipamentos com erro. No entanto, a sua precisão é de tal forma alta que ao estabelecer o threshold como 100% é possível obter 1 falso positivo, 0 falsos negativos e classificar devidamente todos os conjuntos de pedidos de equipamentos com erro. Para além da sua alta exactidão e precisão, o modelo pode ser treinado em menos de uma hora e a execução das previsões é instantânea.

Os resultados dos diferentes modelos estão resumidos na tabela 4.12 onde podemos observar os falsos positivos, falsos negativos, capacidade de generalização, tempos de execução e a percentagens de equipamentos com erro devidamente classificados.

Modelo	FN	FP	Exctidão em equipamentos com erro	Tempo de treino	Tempo de previsão
KNN	0	3	0%	12 min	1 min
LogReg	0	3	0%	6 min	23 seg
SVM	4	4	0%	10 horas	10 min
Decision Tree	0	0	0%	4 seg	1 seg
Autoencoder	25	26	0%	29 min	14 seg
FCNN	0	3	30%	27 min	12 seg

Tabela 4.12: Tabela com análise comparativa dos diferentes algoritmos testados

Embora o modelo FCNN apresente melhores resultados com o threshold a 100%, para efeitos de comparação foi tida em conta a performance do modelo com o threshold a 99%, uma vez que desta forma o modelo apresenta 0 falsos negativos. Como já foi referido a operadora acredita que o modelo deve apresentar o menor valor possível de falsos negativos. Com base em todos os resultados obtidos e, tendo em conta as exigências do projecto, foi escolhido o modelo da sétima experiência (FCNN) para ser implementado no sistema alarmística da operadora.

Capítulo 5

Operacionalização do Algoritmo

Neste capitulo será explicada a necessidade de supervisão inteligente da rede IMS, assim como todo o processo de implementação do modelo na infraestrutura de alarmística da operadora. De seguida, será feita uma análise de vulnerabilidade partindo do princípio que o atacante conhece todos os detalhes do modelo. Por fim será também explicado o processo de reporte que torna o modelo útil e prático para o dia-a-dia da equipa de supervisão da rede IMS da operadora.

5.1 Objectivo

Como já foi explicado na secção 3.3.1, a operadora separa os seus serviços IMS por domínios a que se refere como Realms. Alguns dos domínios estão expostos à Internet através das SBCs, que pretendem estabelecer uma fronteira de protecção da rede. Tendo em conta que alguns domínios estão expostos à Internet, estes são por vezes vítimas de ataques com o intuito de registar uma conta que não pertence ao atacante. Sendo os ataques rápidos e sem aviso, a operadora pretende proteger-se dos mesmos através de mecanismos igualmente rápidos e inteligentes. Assim sendo, foi proposta a criação de um modelo de inteligente capaz de destingir e classificar os pedidos de registo que chegam às SBCs.

O modelo proposto não pretende classificar cada pedido de registo sem conhecimento prévio do comportamento do autor do pedido. Tendo em conta que a sequência dos valores de cada pedido representam a assinatura que permite distinguir um utilizador fraudulento de um utilizador normal, o objectivo do modelo é classificar toda a actividade de pedidos de registo de um IP de origem com base no conjunto dos seus pedidos de registo. Depois de analisado o comportamento de cada IP suspeito a equipa de supervisão da rede será notificada com o resultado da classificação. A partir desse momento a equipa terá a informação necessária para actualizar a Access Control List das SBCs bloqueando

ou não, o tráfego dos IPs classificados pelo modelo.

5.2 Arquitectura de Implementação

A operadora é capaz de armazenar todo o tipo de informação proveniente dos SBCs e de outros serviços externos aos objectivos da dissertação. Assim sendo, todos os dados de pedidos de registo dos SBCs encontram-se disponíveis numa base de dados da operadora para supervisão das diferentes redes. Para além da base de dados, a operadora possui também clusters de processamento de dados com conexão directa ás informações dos SBCs. Assim sendo, o modelo de classificação estará em execução nos clusters da operadora. Todos os dados que são fornecidos ao modelo serão extraídos através de Queries SQL à base de dados de supervisão das diferentes redes. Este processo não é de todo o mais aconselhado, uma vez que a base de dados não é actualizada ao minuto e, o tempo de processamento é superior em comparação ao tempo de processamento caso os dados fossem retirados directamente do SBC. No entanto, como se trata da primeira implementação de um modelo capaz de classificar tráfego fraudulento, optou-se por obter os dados por Queries à base de dados. Depois de extraídos, os dados são processados num cluster como explicado na secção 3.4.1. De seguida o conjunto de pedidos de registo de cada IP é classificado numa escala de 0% a 100% de pedidos considerados como fraudulentos pelo modelo escolhido.

O processo de recolha dos IPs suspeitos, processamentos dos dados desses IPs e consequente classificação foi incorporado num script em python com uma conexão JDBC à base de dados da operadora.

Como já foi referido, o modelo irá ser executado num cluster de processamento de dados da operadora. Sendo os clusters partilhados por diversos serviços, a operadora pretende que as rotinas implementadas nos clusters sejam, no futuro executadas a partir de containers. Para tal foi desenvolvido um container em Docker [17] para a rotina desenvolvida, com uma imagem de python 3 e as bibliotecas necessárias para a classificação e processamento dos dados. O desenvolvimento do modelo em um container apresenta vantagens de processamento para os clusters e garante que o modelo não se altera com futuras actualizações da suas bibliotecas ou sistema operativo. Ao criar um container estamos a criar uma "pequena caixa impermeável"com um ambiente python dedicado apenas para o script desenvolvido. Desta forma não é necessário alocar uma maquina virtual, recursos, e instalar um sistema operativo cada vez que se pretende analisar o estado da rede. Embora a organização por containers apresente grandes vantagens para a administração dos diversos clusters da operadora, a implementação do projecto em containers só será consumada quando o sistema da operadora estiver pronto para tal. Quando o processo de administração dos clusters em containers estiver concluído, o docker criado será executado por um temporizador de hora em hora. O resultado final da execução será uma tabela, guardada na base de dados da operadora, com os IPs suspeitos, o domínio onde

estes estão a efectuar os pedidos de registo, o número de pedidos, o número de contas, o número de tentativas com sucesso e a classificação que o modelo deu a cada IP.

Tendo em conta que este processo ainda não está disponível, o script foi implementado numa cadeia de acções em Apache Airflow, executada de hora em hora através de um temporizador. O resultado final de cada execução é actualizado numa tabela como previsto na implementação por containers. O Apache Airflow é capaz de gerir e seccionar um processo em diferentes acções com ou sem dependência que podem ser executadas em paralelo. Esta ferramenta é também capaz de fornecer um ambiente para gestão e controlo dos diferentes processos em execução num cluster de processamento.

Para extrair os dados do cluster, o serviço de reporte da equipa de supervisão fará um pedido a uma API que a operadora tem disponível para interligar os diversos dashboards operacionais aos clusters. Por fim, o cluster responsável por administrar a base de dados da operadora enviará uma resposta em JSON ao pedido do sistema de Dashboards contendo a tabela de output do modelo.

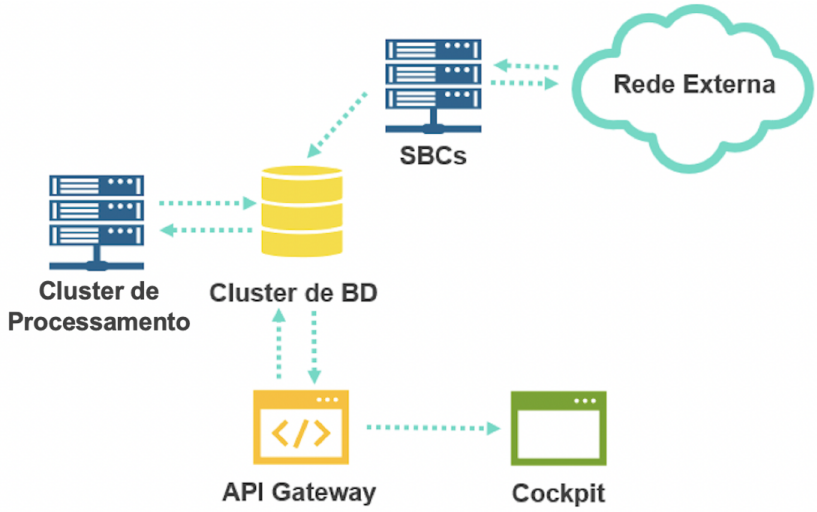


Figura 5.1: Esquema de implementação

5.3 Processo de Detecção

Como já foi referia na secção dos 5.1, o objectivo é classificar o comportamento e de um determinado IP suspeito a partir dos seus pedidos REGISTER dirigidos aos SBCs. Para que tal seja possível é necessário obter uma listas dos IPs que na última hora apresentem um comportamento suspeito. Embora a abordagem inicial contemplasse uma vistoria ao comportamento da rede com intervalos de 15 minutos, tal não será possível para a primeira implementação do modelo. Como foi explicado na secção 5.2, a arquitectura do modelo recolhe os dados de pedidos de registo através de Queries a uma base de dados. O processo de recolha dos dados demonstrou-se demasiado demorado para ser executado em 15 minutos. Assim sendo, partiu-se para uma abordagem de hora em hora. No entanto,

no futuro será possível uma segunda implementação com análises de 15 minutos, uma vez que os dados serão recolhidos directamente dos SBCs.

A análise de comportamentos suspeitos é feita a partir da relação entre o número de pedidos de registo e o número de diferentes contas como explicado na secção 3.3.1. No entanto, sendo a análise feita de hora em hora, os valores médios da relação entre contas e tentativas flutuam bastante consoante a altura do dia. Por este motivo optou-se por ordenar a lista de IPs suspeitos por ordem decrescente em função do valor de contas sobre o número de tentativas de um IP.

A pior consequência de um ataque é um eventual sucesso de autenticação por parte do atacante. Tendo em conta que existe a possibilidade de um ataque ser bem sucedido, é imperativo que o modelo seja capaz de priorizar a análise desses ataques. Com base nesta necessidade foi criada uma segunda lista de IPs suspeitos. Esta segunda lista contempla todos os IPs que tentaram autenticar mais do que 65 contas com uma taxa de sucesso diferente de zero. A escolha do número 65 para o número de contas foi feita tendo em conta os conhecimentos dos técnicos da operadora. Segundo os mesmos, a autenticação de mais do que 65 contas por um só IP, apresenta por si só um comportamento anormal. Na realidade, sendo esta lista de IPs uma lista de IPs com sucessos de autenticação, é muito pouco provável que um atacante obtenha um pedido com sucesso com menos de 65 tentativas de contas. Desta forma, todos os IPs com mais do que 65 contas e taxa de sucesso acima de 0, são altamente suspeitos e de análise prioritária. A lista de IPs suspeitos com sucesso é ordenada em função do número de sucessos de cada IP, sendo que os IPs com maior número de pedidos com sucesso são analisados primeiro.

Tentativas	Contas	IP	Index
368749	362842	[redacted]	0.9839809735077247
1490	1426	[redacted]	0.9570469798657718
2926	2347	[redacted]	0.8021189336978811
113	84	[redacted]	0.7433628318584071
61677	28029	[redacted]	0.4544481735492971
8813	2933	[redacted]	0.3328038125496426
3279	638	[redacted]	0.19457151570600792
68314	13268	[redacted]	0.19422080393477179
463	31	[redacted]	0.06695464362850972
479	32	[redacted]	0.06680584551148225
4947	50	[redacted]	0.01010713563776026
3068	31	[redacted]	0.010104302477183833

Figura 5.2: Tabela de IPs suspeitos ordenado em função do índice

Depois de ordenar as listas como mostra a figura 5.2, são retirados e classificados os pedidos de registo dos 10 primeiros IPs suspeitos de cada lista. Se o décimo IP for

classificado como fraudulento, inicia-se uma segunda ronda de classificação, desta vez com os dez IPs seguintes. O processo repete-se até que o último IP de uma ronda seja classificado como não fraudulento ou até atingir 30 IPs classificados. Na esmagadora maioria das situações normais a rede nunca sofre mais do que dez ataques de diferentes IPs no espaço de uma hora. No entanto a funcionalidade de classificação por rondas de 10 IPs demonstrou-se necessária na eventualidade de um atacante utilizar mais do que 10 IPs diferentes. A utilização de vários IPs para o mesmo ataque torna-se perfeitamente possível com o auxilio de VPNs ou se o ataque for executado por um grupo organizado.

5.4 Análise de Vulnerabilidades

Tendo em consideração todos os mecanismos do modelo para a detecção de comportamentos fraudulentos podemos fazer uma análise de vulnerabilidades do mesmo. Como foi previamente referido, o índice de cada IP na lista de IPs suspeitos tem uma enorme influência na selecção dos IPs a serem classificados. Uma das possíveis formas de evitar que um determinado IP de origem seja eleito para classificação por parte do modelo, seria colocar o comportamento do IP de ataque numa posição de pouco destaque na lista de suspeitos. Se um comportamento fraudulento apresentar um índice próximo de 0, o seu IP de origem estará no fim da lista de IPs suspeitos. Para tal o atacante teria de fazer um enorme número de tentativas para cada conta, de forma a obter um índice o mais próximo de 0 possível. Embora seja possível, este nunca é o comportamento de um atacante, uma vez que estes tendem a varrer um conjunto de números e um dicionário de passwords. O uso de dicionários justifica-se, uma vez que as passwords da operadora não podem ser descobertas com facilidade através do varrimento dos caracteres possíveis. Segundo a operadora, os atacantes não procuram descobrir a password de uma conta por exaustão de combinações, mas sim descobrir contas com fraca protecção. Desta forma o varrimento de contas é o único método que permite uma hipótese de sucesso. Embora pareça lógico ter em conta o alto nível de tentativas e não o número de diferentes contas, esta abordagem não é de todo a mais aconselhada, uma vez com o auxílio de VPNs o atacante pode mudar o seu IP depois de executar um número reduzidos de tentativas. Desta forma, a equipa nunca teria conhecimento do ataque.

O único método de ataque ao qual o modelo se encontra vulnerável, consiste em separar o ataque em vários IPs de origem, sendo que cada IP não poderá registar mais do que 30 contas. Tendo em conta que um ataque sem sucesso pode chegar alcançar as 30 milhões de contas (como já foi experienciado na operadora), seria necessário usar 1 milhão de IPs para que nenhum dos IPs fraudulentos estivesse presentes na lista de IPs suspeitos. Na eventualidade de um dos IPs de ataque ultrapassar as 30 contas este IP estará presente na lista de suspeitos. Tendo em conta esta possibilidade, foi criado um novo sistema de alarmista capaz de contar o número de IPs , presentes na lista de suspeitos, com uma taxa de sucesso igual a 0. Desta forma, se o atacante alterar o seu IP de origem com bastante frequência, o número de IPs com uma taxa de sucesso igual a 0 irá também aumentar de

forma clara. A partir do momento que a maioria dos IPs da lista de suspeitos apresentam uma taxa de sucesso nula a equipa de supervisão será alertada para analisar toda a lista de IPs suspeitos.

Capítulo 6

Conclusões

Neste capítulo será feita conclusão da pesquisa e trabalho desenvolvido. Para além das conclusões de pesquisa e abordagem, será feita uma reflexão sobre os resultados obtidos, métodos utilizados assim como os próximos passos para aperfeiçoamento do modelo e, possibilidades de expansão.

6.1 Sumário de Contribuições

O projecto desenvolvido apresenta três contribuições:

- O método de análise dos dados através do Z-Score por conjuntos de pedidos apresenta uma grande mais valia para o processo de classificação, uma vez que cria nos pedidos de registo uma dependência em relação aos pedidos envolventes. Esta dependência permite classificar os comportamentos de cada IP com base em conjuntos de pedidos.
- O estudo comparativo entre os diferentes tipos de classificadores, supervisionados e não supervisionados permite uma análise geral sobre a capacidade de adaptação de cada modelo aos dados retirados da operadora. Desta forma o projecto desenvolvido apresenta uma forte contribuição como benchmarking de modelos de Inteligência Artificial e Machine Learning para classificação de pedidos de registo SIP.
- Por fim foi executada a implementação de um script Python nos clusters de processamento e a integração do processo de detecção de comportamentos de registo fraudulentos no sistema de alarmística da operadora. A implementação do modelo num sistema de controlo de rede de uma operadora exige uma arquitectura bem definida e estruturada visto que o processo implementado deve coexistir com os demais processos a serem executados nos clusters da operadora. Assim sendo, durante

o processo de implementação houve especial atenção às exigências computacionais e de memoria do modelo, de forma a obter a melhor optimização para o processo de classificação.

6.2 Conclusões

Com base nas experiências e resultados experimentais presentes no capitulo 4, é viável afirmar com certezas que é possível distinguir tráfego de registos SIP fraudulentos e normais. A grande maioria das soluções propostas apresentaram resultados de classificação com uma enorme exactidão. Desta forma, foi possível escolher entre os melhores classificadores qual o que se adaptava melhor as necessidades da operadora. Embora os resultados de classificação sejam bastante favoráveis para os modelos estudados, existe ainda uma dificuldade em classificar o comportamento de equipamentos com erro como sendo normal. Na realidade, a classificação de comportamentos de equipamentos com erro como comportamento anormal está correcta, no entanto não é correcto classificar esses comportamentos como fraudulentos.

O maior factor para o sucesso dos classificadores estudados para análise de tráfego de registo SIP, reside no método de pré processamento dos dados a serem classificados.

Como já foi explicado, um pedido de registo não pode ser avaliado sem o contexto onde este se encontra. Para ter uma noção da verdadeira motivação do pedido de registo, é necessário ter em conta o historial de acções do agente que se pretende registar. Um pedido de registo não pode ser considerado fraudulento porque utilizou um determinado porto de origem, ou porque pretende registar um determinado número de conta (ID), no entanto um pedido de registo pode ser classificado como fraudulento quando os seus valores representam a variação dos argumentos de vários pedidos de registo que em conjunto criam um comportamento fraudulento. Desta forma a análise e normalização dos conjuntos de pedidos de um IP de origem representam o maior factor de sucesso dos processos de classificação.

Tendo a noção da importância que reside no agrupamento dos pedidos de registo por IP de origem, foi necessário que os pedidos de registo presentes no conjunto de dados fossem normalizados separadamente em função do IP de origem. A normalização do conjunto de dados como um todo, iria influenciar os valores de grupos de pedidos normais e fraudulentos, uma vez que a média e desvio padrão do conjunto de dados iria ser obtida através de dados de duas classes diferentes. O pré processamento dos dados, como explicado na secção 3.3.1, é portanto um dos grande contributos da dissertação. A figura 6.1 mostra as diferenças na precisão do modelo escolhido para a implementação (Fully Connected Neural Network), ao classificar grupos de pedidos fraudulentos com a normalização seccionada e, normalização de todo o conjunto de dados.

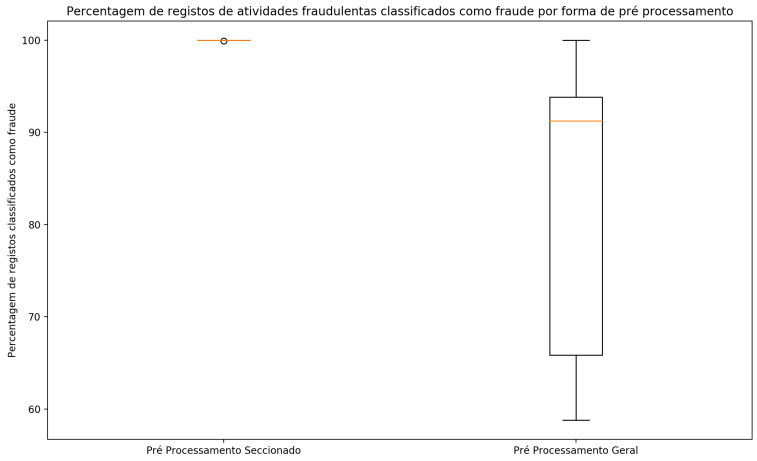


Figura 6.1: Dispersão das percentagens de registos de atividades fraudulentas classificados como fraude por forma de pré processamento

Como podemos observar na figura 6.1, ao aplicar um pré processamento seccionado, o modelo torna-se extremamente preciso, no entanto o mesmo não se verifica com o pré processamento geral do conjunto de dados. Embora todos os conjuntos de pedidos fraudulentos tenham obtido valores superiores a 50% de classificação de registos fraudulentos, quando classificados com o modelo de pré processamento geral, esta dispersão não afecta apenas a precisão do modelo. Com base na simulação da figura 6.1 o modelo com pré processamento geral necessitaria de um threshold de 58%. Ao analisar a figura 6.2, que apresenta dispersão de registos classificados como fraudulentos, provenientes de equipamentos com erro, podemos observar que para o threshold de 58% todos os equipamentos com erro são considerados fraudulentos

Figura 6.2: Dispersão das percentagens de registos de conjuntos de pedidos de equipamentos com erro classificados como fraude por forma de pré processamento

Desta forma, a diminuição da precisão implica um threshold mais abrangente, que neste caso, implica uma diminuição da exactidão.

Embora a ideia inicial fosse executar o projecto nas instalações da operadora, a pandemia que vivemos não permitiu que tal fosse possível. Desta forma, todo o trabalho executado foi levado a cabo com interacções virtuais que, de uma certa forma, limitaram a velocidade do projecto. Ainda assim, a operadora, a equipa de transformação, a equipa de administração dos clusters e a equipa de supervisão da rede IMS facilitaram e asseguram todas as necessidades do projecto da melhor forma possível. Tendo em conta que o momento que vivemos obrigou grande parte da população a alterar os seus diversos hábitos, essas mudanças podem reflectir-se na forma como usamos a rede de voz, que foi objecto de estudo. Assim sendo, e tendo em conta que os hábitos dos utilizadores não são estáticos, a operadora necessitará de avaliar a necessidade de actualizar o treino do modelo com base nas informações que recolhe da sua rede.

A necessidade de um modelo capaz de analisar tráfego de forma inteligente é clara para a operadora que, quando confrontada com as capacidades do modelo, decidiu partir para a sua implementação em tempo real. Mesmo tendo em conta que a maioria dos ataques não obtêm qualquer resposta com sucesso, a possibilidade de bloquear um ataque no inicio de execução apresenta uma vantagem, uma vez que são recursos da rede de voz que a operadora pode passar a conduzir para os seus utilizadores e não para atacantes.

6.3 Próximos Passos

Embora o modelo proposto apresente bons resultados, existem detalhes que podem ser melhorados. Como já foi referido o modelo obtém os dados a classificar através de Queries SQL à uma base de dados disponível para acções de supervisão. A base de dados não é actualizada de hora em hora e o acesso à informação é demorado, o que obriga o modelo a analisar a rede com intervalos de uma hora. Em uma hora bastante pode acontecer na rede IMS da operadora, assim sendo é importante que a recolha dos dados seja agilizada de forma a que seja possível uma análise mais frequente, como de 15 em 15 minutos. Estas necessidades podem ser cumpridas se o modelo tiver uma ligação as SBCs por streams de dados com o software Apache Kafka, por exemplo. Desta forma, o modelo é capaz de obter dados em tempo real e com menor latência.

Assim que os servidores da operadora estiverem disponíveis para a administração e execução de containers, o projecto já executado em docker irá ser capaz de substituir a sequência de acções de Apache Airflow em execução. No entanto, enquanto a execução de containers ainda não é possível, existe espaço para melhorias na implementação em Airflow. O Apache Airflow é capaz de gerir e seccionar um processo em diferentes acções com ou sem dependência que podem ser executadas em paralelo. Até ao momento de implementação inicial, a cadeia de execução de Airflow está bastante simplista com apenas 3 acções. A primeira acção é responsável por iniciar o processo em Airflow, a segunda

acção executa todo o script desenvolvido e, por fim a terceira termina o processo. Este esquema de implementação não é capaz de retirar total proveito das capacidades do Apache Airflow uma vez que o script é executado sequencialmente sem actividades em paralelo. Numa futura implementação os processos de recolha de informação à base de dados, pré processamento dos dados recolhidos e classificação dos conjuntos de pedidos de registo podem ser processos isolados com dependências, de forma a que sejam executados em paralelo. Desta forma o tempo de espera para a recolha dos dados a serem analisados pode ser aproveitado para processar e classificar os dados previamente recolhidos.

Todo o trabalho desenvolvido foi conduzido no sentido de classificar pedidos REGISTER, no entanto existem diversos tipos de actividade maliciosa através de mensagens SIP de diferentes cabeçalhos. Assim sendo, é ainda possível analisar comportamentos fraudulentos com modelos capazes de inspeccionar diferentes tipos de mensagens. A existência de chamadas fraudulentas é uma evidencia no nosso dia-a-dia. Para que a operadora seja capaz de combater essas actividades, o modelo necessita de inspeccionar também os comportamentos e respostas das mensagens com cabeçalho INVITE. A análise do comportamento dos INVITEs seria feito tendo em conta o número de conta (ID) que origina os pedidos, de forma a que fosse possível traçar um plano capaz de separar um utilizador normal, de um utilizador malicioso. Para além das mensagens SIP, a operadora sofre também tentativas de acesso a ficheiros de configuração com o objectivo de edição dos mesmos. Todo o tráfego, dos diversos serviços, é guardado e pode ser analisado de forma a que seja possível criar mecanismos inteligentes e rápidos, capazes de evitar uma falha de segurança.

A arquitectura de implementação prevê que os resultados do modelo sejam uma fonte de informação para a equipa de supervisão da rede de voz. No entanto, caso o modelo se demonstre altamente preciso, existe a possibilidade de contornar a decisão humana, de forma a que o output do modelo deixe de ser apenas uma tabela informativa. Desta forma, o modelo obteria permissões de edição na Access Control List das SBCs e seria capaz de bloquear o tráfego de determinados IPs que classificou como fraudulentos.

Bibliografia

[1] *SIPp*. URL: http://sipp.sourceforge.net/ (acedido em 2007).

[2] M. Abid, S. Song, H. Moustafa e H. Afifi. "Efficient identity-based authentication for IMS based services access". Em: *MoMM2009 - The 7th International Conference on Advances in Mobile Computing and Multimedia* (2009), pp. 260–266. DOI: 10.1145/1821748.1821798.

[3] M. A. Akbar, Z. Tariq e M. Farooq. "A comparative study of anomaly detection algorithms for detection of sip flooding in IMS". Em: *IMSAA'08 - 2nd International Conference on Internet Multimedia Services Architecture and Application* (2008). DOI: 10.1109/IMSAA.2008.4753934.

[4] E. E. Anderlind, D. W. Faucher, E. H. Grosse, D. N. Heer, A. R. McGee, D. P. Strand e R. J. T. Jr. "IMS security". Em: *Bell Labs Technical Journal* 11.1 (2006), pp. 37–58. ISSN: 1538-7305. DOI: 10.1002/bltj.20143.

[5] M. M. Breunig, H.-P. Kriegel, R. T. Ng e J. Sander. *LOF: Identifying Density-Based Local Outliers*. Rel. téc.

[6] G. Camarillo, T. Kauppinen, M. Kuparinen e I. M. Ivars. "Towards an innovation oriented IP multimedia subsystem". Em: *IEEE Communications Magazine* 45.3 (2007), pp. 130–136. ISSN: 01636804. DOI: 10.1109/MCOM.2007.344594.

[7] T. Cejka, V. Bartos, L. Truxa, H. Kubatova, T. Cejka, V. Bartos, L. Truxa, H. Kubatova e U. A.-a. Flow. "Using Application-Aware Flow Monitoring for SIP Fraud Detection To cite this version : HAL Id : hal-01410154 Using Application-Aware Flow Monitoring for SIP Fraud Detection". Em: (2016).

[8] *CFCA*. URL: https://www.cfca.org/ (acedido em 07/02/2020).

[9] C.-C. Chang e C.-J. Lin. *LIBSVM: A Library for Support Vector Machines*. Rel. téc. 2001. URL: www.csie.ntu.edu.tw/.

[10] C. Y. Chen, K. D. Chang e H. C. Chao. "Transaction-pattern-based anomaly detection algorithm for IP multimedia subsystem". Em: *IEEE Transactions on Information Forensics and Security* 6.1 (2011), pp. 152–161. ISSN: 15566013. DOI: 10.1109/TIFS.2010.2095845.

[11] J. Chen, S. Sathe, C. Aggarwal e D. Turaga. "Outlier detection with autoencoder ensembles". Em: *Proceedings of the 17th SIAM International Conference on Data Mining, SDM 2017* (2017), pp. 90–98. DOI: 10.1137/1.9781611974973.11.

[12] E. Christian e A. Roos. "A review of policy-based resource and admission control functions in evolving access and next generation networks". Em: *Journal of Network and Systems Management* 16.1 (2008), pp. 14–45. ISSN: 10647570. DOI: 10.1007/s10922-007-9096-3.

[13] *climin: optimization, straight-forward — climin 0.1 documentation*. URL: https://climin.readthedocs.io/en/latest/ (acedido em 02/02/2020).

[14] *Comparative Study on Classic Machine learning Algorithms , Part-2*. URL: https://medium.com/@dannymvarghese/comparative-study-on-classic-machine-learning-algorithms-part-2-5ab58b683ec0 (acedido em 06/02/2020).

[15] C. Cox. "An Introduction to LTE: LTE, LTE-Advanced, SAE, VoLTE and 4G Mobile Communications: Second Edition". Em: *An Introduction to LTE: LTE, LTE-Advanced, SAE, VoLTE and 4G Mobile Communications: Second Edition* 9781118818039 (2014), pp. 1–449. DOI: 10.1002/9781118818046.

[16] J. Davidson, M. B. J. Peters e S. M. S. Kalidindi. "VoiceOver IP Fundamentals, second ed". Em: (2006).

[17] *Docker*. URL: https://www.docker.com/ (acedido em 24/08/2020).

[18] M. El Barachi, R. Glitho e R. Dssouli. "Enhancing the QoS and resource management aspects of the 3GPP IMS emergency service architecture". Em: *2008 5th IEEE Consumer Communications and Networking Conference, CCNC 2008* (2008), pp. 112–116. DOI: 10.1109/ccnc08.2007.32.

[19] S. El-Sawda e P. Urien. "SIP security attacks and solutions: A state-of-the-art review". Em: *Proceedings - 2006 International Conference on Information and Communication Technologies: From Theory to Applications, ICTTA 2006* 2 (2006), pp. 3187–3191. DOI: 10.1109/ICTTA.2006.1684926.

[20] *GitHub - EnableSecurity/sipvicious: SIPVicious suite is a set of security tools that can be used to audit SIP based VoIP systems*. URL: https://github.com/EnableSecurity/sipvicious (acedido em 24/01/2020).

[21] S. Hawkins, H. He, G. Williams e R. Baxter. "Outlier detection using replicator neural networks". Em: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 2454 LNCS. 2002, pp. 170–180. ISBN: 3540441239. DOI: 10.1007/3-540-46145-0_17.

[22] D. Hoffstadt, A. Marold e E. P. Rathgeb. "Analysis of SIP-based threats using a VoIP Honeynet System". Em: *Proc. of the 11th IEEE Int. Conference on Trust, Security and Privacy in Computing and Communications, TrustCom-2012 - 11th IEEE Int. Conference on Ubiquitous Computing and Communications, IUCC-2012* (2012), pp. 541–548. DOI: 10.1109/TrustCom.2012.90.

[23] M. T. Hunter, R. J. Clark e F. S. Park. "Security issues with the IP Multimedia Subsystem (IMS)". Em: *Middleware Conference - Proceedings of the 2007 Workshop on Middleware for Next-generation Converged Networks and Applications 2007, MNCNA'07* (2007). DOI: 10.1145/1376878.1376887.

[24] T. Jansky, T. Cejka e V. Bartoš. "Security of Networks and Services in an All-Connected World". Em: 10356 (2017), pp. 125–130. DOI: 10.1007/978-3-319-60774-0. URL: http://link.springer.com/10.1007/978-3-319-60774-0.

[25] J.Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks e E. Schooler. *hjp: doc: RFC 3261: SIP: Session Initiation Protocol. RFC 3261*. 2002. URL: http://www.hjp.at/doc/rfc/rfc3261.html (acedido em 21/01/2020).

[26] S. Karapantazis e F.-N. Pavlidou. "VoIP: A comprehensive survey on a promising technology". Em: (2009). DOI: 10.1016/j.comnet.2009.03.010.

[27] F. Keller, E. Müller e K. Böhm. "HiCS: High contrast subspaces for density-based outlier ranking". Em: *Proceedings - International Conference on Data Engineering*. 2012, pp. 1037–1048. DOI: 10.1109/ICDE.2012.88.

[28] H. Khlifi e J.-C. Grégoire. "IMS Application Servers". Em: *IEEE Computer Society* (2008).

[29] K. I. Kim, T. Kim, N. W. Cho e M. Kim. "Toll Fraud Detection of VoIP Service Networks in Ubiquitous Computing Environments". Em: *International Journal of Distributed Sensor Networks* 2015 (2015). ISSN: 15501477. DOI: 10.1155/2015/276408.

[30] M. Kolon. "Voice over IP". Em: *The CRC Handbook of Modern Telecommunications* 45.1 (2010), pp. 1–30–1–38. DOI: 10.1201/9780849333378.

[31] M. Koukal e R. Bestak. "Architecture of IP Multimedia Subsystem". Em: June (2006), pp. 7–9.

[32] A. Lazarevic, L. Ertoz, V. Kumar, A. Ozgur e J. Srivastava. *A Comparative Study of Anomaly Detection Schemes in Network Intrusion Detection †*. Rel. téc. URL: http://www.siam.org/journals/ojsa.php.

[33] Y. Liao e V. R. Vemuri. "Classifier for Intrusion". Em: *Computers & Security* 21.5 (2002), pp. 439–448.

[34] R. P. Lippmann, D. J. Fried, I. Graf, J. W. Haines, K. R. Kendall, D. McClung, D. Weber, S. E. Webster, D. Wyschogrod, R. K. Cunningham e M. A. Zissman. "Evaluating intrusion detection systems: The 1998 DARPA off-line intrusion detection evaluation". Em: *Proceedings - DARPA Information Survivability Conference and Exposition, DISCEX 2000*. Vol. 2. Institute of Electrical e Electronics Engineers Inc., 2000, pp. 12–26. ISBN: 0769504906. DOI: 10.1109/DISCEX.2000.821506.

[35] Y. Liu, Z. Li, C. Zhou, Y. Jiang, J. Sun, M. Wang e X. He. "Generative Adversarial Active Learning for Unsupervised Outlier Detection". Em: *IEEE Transactions on Knowledge and Data Engineering* (2019), pp. 1–1. ISSN: 1041-4347. DOI: 10.1109/tkde.2019.2905606. arXiv: 1809.10816.

[36] *Machine Learning Basics: Support Vector Machines - Data Driven Investor - Medium*. URL: https://medium.com/datadriveninvestor/machine-learning-basics-support-vector-machines-358235afb523 (acedido em 06/02/2020).

[37] J Macqueen. *SOME METHODS FOR CLASSIFICATION AND ANALYSIS OF MULTIVARIATE OBSERVATIONS*. Rel. téc.

[38] L. Martin. "Security of Networks and Services in an All-Connected World". Em: 10356 (2017), pp. 173–178. DOI: 10.1007/978-3-319-60774-0. URL: http://link.springer.com/10.1007/978-3-319-60774-0.

[39] G. M. Miikka Poikselkä. *The IMS-IP Multimedia Concepts and Services*. Vol. 53. 2013, pp. 1689–1699.

[40] S. A. A. Mohammad Ilyas. "IP Multimedia System (IMS) Handbook". Em: 53 (2013), pp. 1689–1699.

[41] G. Münz, S. Li e G. Carle. "Traffic Anomaly Detection Using K-Means Clustering". Em: *Leistungs-, Zuverlassigkeits- und Verlasslichkeitsbewertung von Kommunikationsnetzen und Verteilten Systemen* (2007), pp. 1–8. URL: http://www.decom.ufop.br/menotti/rp122/sem/sem3-luciano-art.pdf.

[42] Y. Rebahi, M. Sher e T. Magedanz. "Detecting flooding attacks against IP multimedia subsystem (IMS) networks". Em: *AICCSA 08 - 6th IEEE/ACS International Conference on Computer Systems and Applications* (2008), pp. 848–851. DOI: 10.1109/AICCSA.2008.4493627.

[43] K. Rieck, S. Wahl, P. Laskov, P. Domschitz e K. R. Müller. "A self-learning system for detection of anomalous SIP messages". Em: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 5310 LNCS (2008), pp. 90–106. ISSN: 03029743. DOI: 10.1007/978-3-540-89054-6-5.

[44] S. Sathe e C. Aggarwal. "LODES: Local density meets spectral outlier detection". Em: *16th SIAM International Conference on Data Mining 2016, SDM 2016*. Society for Industrial e Applied Mathematics Publications, 2016, pp. 171–179. ISBN: 9781510828117. DOI: 10.1137/1.9781611974348.20.

[45] H. Schulzrinne e J. Rosenberg. "ADVANCED SIGNALING AND CONTROL IN The Session Initiation Protocol :" em: March 1999 (2000), pp. 134–141.

[46] J. Tang, Y. Cheng e C. Zhou. "Sketch-based SIP flooding detection using Hellinger distance". Em: *GLOBECOM - IEEE Global Telecommunications Conference* (2009). DOI: 10.1109/GLOCOM.2009.5426267.

[47] A. Vrábel, R. Vargic e I. Kotuliak. "Subscriber databases and their evolution in mobile networks from GSM to IMS". Em: *Proceedings Elmar - International Symposium Electronics in Marine* September (2007), pp. 115–117. ISSN: 13342630. DOI: 10.1109/ELMAR.2007.4418811.

[48] M. Weiss e J. Hwang. " Internet Telephony or Circuit-Switched Tele-phony: Which is Cheaper? " Em: Sept. 1999 (2006).

[49] Wiens,Anton and Kübler, Sandra and Wiens, Torsten and Massoth, Michael. "Improvement of User Profiling, Call Destination Profiling and Behavior Pattern Recognition Approaches for Telephony Toll Fraud Detection". Em: *International Journal on Advances in Security* 8 (2015), pp. 1–15.