Instituto Superior de Engenharia do Porto

Análise de Desempenho e Recomendação para Soluções IVR

Carina Alexandra Pires Araújo

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Tecnologias do Conhecimento e da Decisão

Orientador: Doutor Paulo Oliveira

Co-Orientador: Engenheiro Joaquim Azevedo

Júri:

Presidente:

João Paulo Jorge Pereira, Doutor, Instituto Superior de Engenharia do Porto

Vogais:

José Alberto da Silva Freitas, Doutor, Faculdade de Medicina da Universidade do Porto

Paulo Jorge Machado Oliveira, Doutor, Instituto Superior de Engenharia do Porto

Porto, Outubro 2014

Resumo

Interactive Voice Response (IVR) é a tecnologia que permite que um computador possa interagir com os humanos através do uso da fala ou marcação *Dual-Tone Multi-Frequency* [1](DTMF). No mundo das telecomunicações, uma solução IVR tem por base um servidor computacional, com o seu hardware de telefonia dedicado, ou sem hardware de telefonia no caso do ambiente *Session Initiation Protocol* (SIP), que assegura funcionalidades interativas como processamento avançado de áudio e de vídeo, áudio e vídeo conferência, reconhecimento e síntese de fala, reprodução de conteúdos multimédia, campanhas publicitárias para sistemas móveis, entre outros.

Todas as funcionalidades enumeradas representam uma grande parte dos serviços disponibilizados por este sector, daí a importância destas plataformas. Como tal, para manter, melhorar e aumentar a área de negócio é preciso saber o estado das plataformas e classificá-las quanto ao volume de chamadas que processam em cada instante, para definir a criticidade de cada uma. A aquisição deste tipo de conhecimento desencadeará atitudes preventivas e proactivas.

As atitudes preventivas incidem na deteção de problemas atempadamente, antecipando a abertura de um caso de suporte e evitando a desativação de plataformas, quando há diminuição do número de chamadas processadas.

As atitudes proactivas despoletam a vertente comercial do negócio. Se uma plataforma estiver constantemente com uma percentagem de ocupação dos recursos de telefonia, memória, unidade central de processamento, entre outros, muito elevada, podem ser feitas propostas de venda para ampliação de capacidade de forma a colmatar a falta de recursos iminente.

Para ajudar a este processo de tomada de decisão implementou-se um sistema que permite classificar, com base nalguns parâmetros de desempenho, o estado atual de uma solução do tipo IVR, que identifique padrões que permitam a recomendação de ações visando a otimização da solução. Toda a informação recolhida é tratada e representada sobre a forma de gráficos e submetida ao algoritmo C5.0 para classificação do estado do IVR. Este conhecimento permite a monitorização e acompanhamento da evolução da plataforma.

Os resultados obtidos com esta solução denominada de IVR Analyzer possibilitaram a deteção de problemas de uma forma expedita e da criação de um modelo de classificação por IVR permitindo a sua caracterização e disponibilizando recomendações com base no *hardware* do sistema e na classificação obtida, ajudando no processo de tomada de decisão.

Palavras-chave: IVR, *Data Mining*, Classificação, Árvores de Decisão, C5.0

[1] Tom de duas frequências gerado pelas teclas de marcação dos telefones.

Abstract

Interactive Voice Response (IVR) is a technology that allows a computer to interact with humans through the use of voice dialing or Dual-Tone Multi-Frequency [2](DTMF). In the world of telecommunications, IVR solutions are based on computational servers, with dedicated telephony hardware, or without dedicated telephony hardware in the case of using *Session Initiation Protocol* (SIP), which ensures interactive features like advanced audio and video processing, audio and video conferencing, speech recognition and synthesis, multimedia content streaming, advertising campaigns for mobile systems, among others.

All features listed above represent a significant part of the services offered by this sector, hence the importance of these platforms. To maintain, improve and enhance the business area you need to know at all times the status of these platforms and classify them according to the call volume being processed at every moment, in order to define the criticality of each. The acquisition of such knowledge, triggers preventive and proactive measures.

Preventive measures, on the one hand, enable timely detection of problems, anticipating the opening of fault reports and preventing the deactivation of platforms, in the case of reduction of the number of processed calls.

Proactive measures, on the other hand, trigger the commercial side of business. If a platform is constantly with a high percentage of telephony resources usage, such as memory, central processing unit, and others, commercial initiatives for capacity upgrade can be triggered in order to overcome imminent resource starvation.

To help this decision making process, a system was implemented that allows one to continuously assess the current state of an IVR platform, based on some performance metrics, allowing the recommendation of changes to optimize the solution. All the information collected is treated and represented on graphs and submitted to the C5.0 algorithm to classify the state of the IVR. This knowledge enables the monitoring and tracking the progress of the platform.

The results obtained with this solution called IVR Analyzer allowed the detection of problems more quickly and the creation of a classification model for each IVR allows the characterization of each one and provides recommendations based on the system hardware and the model outcome, helping in the process decision-making.

Keywords: IVR, Data Mining, Classification, Decision Trees, C5.0

[2] Two tone frequencies generated by the dialing keys phones.

Agradecimentos

Apenas se deveriam fazer trabalhos que nos picam e nos mordem, caso contrário será a estagnação.

A velocidade a que se alteram os conhecimentos é de tal forma alucinante que o que é hoje inovação amanhã poderá não o ser. É necessário, portanto, formação contínua. Esta é a razão do porquê da minha motivação impulsionada por pessoas e entidades, como:

- Os meus pais, que me ensinaram a importância da nobreza do comportamento, do saber e do querer;
- O meu marido pela compreensão, apoio e carinho;
- O meu orientador de dissertação Doutor Paulo Oliveira;
- Os engenheiros Joaquim Azevedo e Nuno Ferreira da PT Inovação;
- As empresas Shortcut, LDA e PT Inovação, SA.

A todos os que acompanharam este projeto e que ao longo da minha carreira profissional e académica contribuíram para esta forma de pensar e de estar na vida, o meu muito obrigado.

Índice

Lista de Figuras

Lista de Tabelas

Acrónimos e Símbolos

Lista de Acrónimos

ASR	*Automatic Speech Recognition*
CPS	*Call Per Second*
CPU	*Central Processing Unit*
CSV	*Comma-Separated Values*
DTMF	*Dual-Tone Multi-Frequency*
FSK	*Frequency-Shift Keying*
FTP	*File Transfer Protocol*
GPL	*Gnu Public License*
IMS	*IP-based Multimedia Services*
IVR	*Interactive Voice Response*
KDD	*Knowledge Discovery in Databases*
MIB	*Management Information Base*
MRF	*Media Resource Function*
NRPE	*Nagios Remote Plugin Executor*
OID	*Object Identifier*
PSTN	*Public Switch Telephone Network*
RRDTool	*Round Robin Database Tool*
SCP	*Secure Copy Protocol*
SFTP	*Secure File Transfer Protocol*
SGBD	Sistemas de Gestão de Base de Dados

SIP *Session Initiation Protocol*

SLA *Service Level Agreement*

SNMP *Simple Network Management Protocol*

SSH *Secure Shell*

TTS *Text-to-Speech*

1 Introdução

Neste capítulo é destacada a importância e a motivação que impulsionou o estudo para a realização desta dissertação. Para além disso, é explicada a organização de todo o documento para orientar a sua leitura.

1.1 Enquadramento

Nos dias que correm, com o aumento da capacidade de armazenamento e processamento dos servidores, bem como a redução do seu custo, despoletou-se a recolha e armazenamento de vários tipos de dados importantes para o negócio. Toda a informação guardada, quando devidamente tratada e analisada pode permitir a identificação de padrões, podendo dar origem a conhecimento operacional e estratégico para a organização. Quando bem aproveitado pode aumentar significativamente a sua competitividade pois, possibilita a definição de novas estratégias.

O sector de telecomunicações abrange diversos produtos e serviços. Porém, numa análise global o mais importante a considerar é o processamento de chamadas. Para a realização desta funcionalidade existem plataformas designadas por *Interactive Voice Response* (IVR). Estes sistemas integram a tecnologia na área das telecomunicações, computadores e sistemas de informação, que permite o desenvolvimento de aplicações de fala interativas, reprodução de conteúdos multimédia, áudio e videoconferência, reconhecimento e síntese de fala, com acesso a diferentes sistemas de informação e incorporando princípios de gestão centralizada.

Esta plataforma produz um conjunto de dados que podem ser utilizados para monitorização do sistema, análise, estatística e previsões. Alguns dos dados estão guardados em ficheiros

Comma-Separated Values (CSV) e outros podem ser consultados através da sua *Management Information Base* (MIB) [3].

Estes dados são gerados automaticamente pela plataforma e estavam a ser guardados por um curto período de tempo, na maioria dos casos um mês. Estes registos eram referentes aos valores máximos atingidos por hora. Esta informação não indicava o estado real da solução.

Com este tipo de armazenamento de informação não se tem noção do volume de chamadas ao longo do tempo, dificultando medidas preventivas e proactivas no cliente.

Na deteção de problemas toda a análise gira em torno do número de chamadas. Se um servidor está com uma taxa de ocupação de CPU muito elevada, mas tem muitas chamadas a decorrer em simultâneo, este componente está com um funcionamento normal. Caso tenha um número reduzido de chamadas, pode ser indicação de um possível problema que esteja a afetar esse componente, exigindo à equipa de operação a sua análise.

Este tipo de elações só podem ser tiradas se a informação for recolhida em instantes de tempo constantes, até para serem estabelecidas comparações por dia no mesmo instante de tempo.

1.2 Motivação

O IVR gera um conjunto de dados sobre o estado do servidor, produto e serviços nele instalados. Esses dados são de extrema importância para a análise de performance da plataforma. De toda a informação gerada é armazenada apenas os máximos registados em cada hora. No passado, era tudo o que chegava para se ter uma visão do IVR no cliente.

Nos dias de hoje, para fazer valer o produto perante o cliente são necessários relatórios de desempenho mensais. Para tal, surgiu a necessidade de recolher nas diversas plataformas, nos diversos clientes, dados gerados no mesmo instante de tempo. Cumprindo este objectivo, a realização de relatórios estatísticos fidedignos torna-se possível, permitindo uma rápida visualização do estado dos diversos sistemas. Esta informação ajuda à tomada de decisão de ambas as partes. Do lado do cliente mostra a importância do investimento realizado e incentiva à continuação do mesmo. Do lado do fornecedor visa a identificação de problemas, possibilitando que aja de antemão e de oportunidades, se a informação retornada for positiva. Com esta informação retira-se unicamente proveito do estado atual e não do futuro, ou seja, possibilita à organização uma atitude preventiva. Para manter ou aumentar a competitividade, a organização tem de ser preventiva e proactiva.

Data Mining consiste na exploração e análise semiautomática de uma grande quantidade de dados na procura de padrões e regras úteis para o negócio [Berry e Linhoff, 1997]. A sua utilização, com o desenvolvimento de novas tecnologias e sistemas de informação, tornou-se cada vez mais importante, tanto no seu estudo como na sua aplicabilidade nas organizações.

[3] Estrutura de armazenamento hierárquico de informação.

O *Data Mining* é uma mais-valia para qualquer empresa ajudando-a a identificar padrões nas suas soluções, para melhoria dos seus recursos e qualidade de serviço, assim como a contribuir para o entendimento do meio envolvente.

Se um IVR num cliente processa muitas chamadas por dia e está a atingir o máximo dos recursos disponíveis, pode ser feita uma proposta para aumento de capacidade. Se, pelo contrário, está a processar poucas chamadas, tem de se realizar um *brainstorming* para a recolha de ideias para despoletar o aumento de chamadas, ou seja, aumentar a taxa de utilização da plataforma.

Em suma, o objetivo inerente à recolha de informação dos IVRs é de em qualquer local e momento poder saber o estado em que se encontram, melhorando todo o trabalho de operação, manutenção e crescimento desta plataforma.

1.3 Objetivos

Neste trabalho realizou-se um estudo de ferramentas de monitorização e análise de performance de plataformas e serviços. Analisadas as vantagens e desvantagens de cada uma delas, apresentou-se uma solução para dar resposta às necessidades sentidas - uma solução que tem como objetivos a seleção, recolha e transformação dos dados de desempenho de uma plataforma IVR a fim de obter conhecimento sobre o seu estado atual e tendência de evolução. Esta solução vem dar resposta a uma necessidade existente na empresa e dando-lhe conhecimento para responder às necessidades do ambiente que a envolve.

Todas as etapas para a aquisição de conhecimento são desempenhadas de uma forma autónoma e independente, prescindindo de qualquer ação do utilizador. Este processo é moroso e para ser feito manualmente exigiria muito tempo e alocação de recursos. A solução proposta executa automaticamente todas as fases de descoberta de conhecimento de tal forma que o utilizador só será necessário na reta final para dar o seu parecer, ou seja, para fazer a avaliação e interpretação dos resultados obtidos, para organizá-los e apresentá-los à empresa e aos clientes. A tarefa de elaboração de um relatório de desempenho torna-se mais fácil e quase que automática, possibilitando uma maior rapidez de ação, tanto nas tarefas de operação e manutenção de plataformas como nas proposta evolutivas das mesmas.

Todos os dados recolhidos são armazenados em base de dados o que permite colmatar a falta de histórico que existia pois, os dados eram guardados em ficheiros com uma duração máxima de um mês e com registo apenas dos máximos atingidos em cada hora.

Para que facilmente um utilizador consulte a informação ou gere relatórios de desempenho sem precisar de formação, a interface gráfica da solução tem uma estrutura simples, ergonómica e intuitiva.

1.4 Organização do Documento

Esta dissertação está organizada em seis capítulos. É pretendido realizar uma introdução ao problema e motivações relativos à análise do estado de desempenho dos IVRs, identificar, descrever e comparar ferramentas existentes, modelos de classificação *Data Mining* e algoritmos de árvores de decisão para justificar a solução desenvolvida. No final, é apresentada a solução desenvolvida e as conclusões do trabalho realizado.

No primeiro capítulo é descrito o âmbito do projeto, conceitos envolvidos e a área de negócio em que se insere. Após o enquadramento, é abordada a motivação e objetivos para o desenvolvimento da solução.

No segundo capítulo é feito o levantamento do estado de arte com a descrição de múltiplas ferramentas possíveis de serem usadas na resolução do problema em mãos. São abordadas as mais usadas e descritas, para cada uma delas, as suas principais funcionalidades, vantagens e desvantagens, concluindo-se com uma comparação entre elas. Segue-se a descrição do conceito de *Data Mining* e abordagem de alguns dos métodos utilizados. Por fim, são descritos os algoritmos de geração de árvores de decisão mais conhecidos.

O terceiro capítulo descreve o estudo da solução. São definidos os requisitos funcionais e não funcionais, o tipo de utilizador, a arquitetura, indicando quais os módulos necessários e que funcionalidades albergam.

No quarto capítulo é explicada a implementação de cada um dos módulos da solução. Para além disso, são descritas as tecnologias e protocolos utilizados, justificando cada uma das escolhas.

No quinto capítulo são caracterizados alguns dos testes realizados e apresentados os resultados obtidos pela solução, acompanhados da explicação e análise de cada uma das situações.

Por fim, é examinado todo o trabalho realizado, enaltecendo os objetivos atingidos, melhorias a desenvolver e dificuldades sentidas ao longo do desenvolvimento da solução. O capítulo termina com a sugestão de possíveis melhorias a incorporar na solução no âmbito de trabalhos futuros a realizar.

2 Estado de Arte

Este capítulo apresenta o levantamento do estado de arte, realizado antes do início de desenvolvimento da solução. São apresentadas aplicações de monitorização e recolha de informação de sistemas, selecionadas durante a pesquisa como as mais interessantes e mais utilizadas. As ferramentas descritas foram alvo de estudo e experimentação, permitindo a criação de uma tabela de comparação.

É feita uma breve apresentação do que é o *Data Mining*, são descritos os modelos mais conhecidos, como classificação, regras de associação e *clustering*. São referenciados e caracterizados os algoritmos de classificação mais utilizados.

2.1 Ferramentas de Monitorização

Ferramentas de monitorização são aplicações que permitem uma gestão centralizada dos diversos terminais que compõem uma rede. Fazem a recolha do estado de diversas componentes que compõem um determinado nó, reportando em que estado se encontra [Andreolini, Colajanni e Pietri,2012].

Embora recolham dados que possibilitem a análise de performance de sistemas, a sua utilização inclina-se mais para a operação e manutenção de plataformas, são encaradas como ferramentas de alarmística.

Este tipo de solução representa a informação sob a forma de gráficos e listas de eventos assinalados com cores. As cores são muito utilizadas para definir a criticidade de um estado de uma dada componente, tornando rápida a identificação de problemas.

2.1.1 Nagios

O Nagios [Nagios,2013] é uma ferramenta web que permite a monitorização de sistemas e redes, permitindo observar o estado de servidores e serviços, enviando alertas e notificações sempre que o seu estado sofre alterações. Desenvolvido para sistemas Linux, é gratuito e de fácil instalação.

A configuração é realizada sob a forma de ficheiros disponíveis na diretoria principal da aplicação (/etc/nagios):

- nagios.cfg: definição de quais os ficheiros de configuração a serem carregados pelo Nagios e localização dos *plugins* instalados.

As restantes configurações podem ser feitas utilizando um único ficheiro ou vários, tem como funcionalidades a definição de servidores (dados como ip e nome do servidor), serviços, comandos e tipos de validações a realizar. As validações podem ser classificadas como intrusivas (*active check*) ou não intrusivas (*passive check*). No primeiro caso, o Nagios executa comandos para verificar se o sistema que está a monitorizar está operacional, enquanto no segundo, fica à escuta, ou seja, à espera que o sistema lhe envie uma *trap SNMP* [4] a indicar o seu estado.

Tabela 1 – Nagios Parâmetros de configuração

Tipo	**Configuração**

Servidor	define host{
	use generic-host
	host_name IVR1
	alias IVR1
	address 127.0.0.1
	check_command check-host-alive
	}

[4] São pacotes de rede que contém dados sobre o estado de um componente ou serviço

Comando	define command{
	command_name check_ping
	command_line $USER1$/check_ping -H $HOSTADDRESS$ -w $ARG1$ -c $ARG2$ -p 5
	}

Active Check	define service{
	use local-service ; Name of service template to use
	host_name IVR1
	service_description INFRA-PING
	check_command check_ping!100.0,20%!500.0,60%
	}

Passive Check	define service{
	use passive_service
	service_description IVR1 Check Passivo
	host_name IVR1
	check_freshness 1
	freshness_threshold 3600
	check_command no_passive_check
	}

Para a monitorização de cada componente têm de ser geradas as entradas definidas na Tabela 1, o que torna muito complicado a gestão das configurações em ficheiros quando se tem um sistema de alarmística centralizado, isto é, que monitoriza diversas plataformas e serviços em simultâneo, gerando inúmeras entradas de configuração.

O Nagios *Core* gere todas as tarefas de monitorização, limites de tempo de execução de uma tarefa, seleção dos utilizadores a serem notificados quando surge um alarme, periodicidade das verificações do sistema (*checks*). A sua responsabilidade é apenas de coordenar.

Assim sendo, para que possa monitorizar tem de ter disponíveis os seguintes agentes:

- SNMP (*Simple Network Management Protocol*) - para notificação do estado da solução, podendo ser feito através de um *check_snmp* a um porto de monitorização de um serviço;
- NRPE - para monitorização de servidores Linux;
- NSClient++ - para monitorização de terminais Windows.

Existem quatro estados possíveis *Unknown*, *OK*, *Warning* e *Critical*. Os limiares de cada um são definidos pelo utilizador. Por exemplo, a definição de alarmística para o espaçamento em disco:

  - *Unknown* - não conseguiu obter informação do componente;
  - *OK* -taxa de ocupação de disco < 75%;
  - *Warning* - taxa de ocupação de disco >= 75% e < 90%;
  - *Critical* - taxa de ocupação de disco >= 90%.

Cada tipo de alarme é caracterizado por uma cor laranja, verde, amarelo e vermelho, respetivamente, como se verifica na Figura 1.

Host	Service	Status	Last Check	Duration	Attempt	Status Information
IVR1	INFRA-Current Load	OK	11-25-2014 22:57:34	738d 7h 18m 45s	1/3	Load OK - 19 Load Average [5 Minutes]
	INFRA-Current Users	OK	11-25-2014 22:57:36	738d 7h 18m 24s	1/3	Users OK - 0 Logged On
	INFRA-Free Disk Space [/]	OK	11-25-2014 22:57:34	738d 7h 18m 26s	1/3	Partition [/] OK - 10964432 Kb free space
	INFRA-Free Disk Space [/var]	OK	11-25-2014 22:54:54	336d 3h 29m 44s	1/3	Partition [/var] OK - 4598944 Kb free space
	INFRA-Free Memory	OK	11-25-2014 22:56:23	738d 7h 19m 17s	1/3	Memory OK - 858132 Kb free memory
	INFRA-Free Swap	OK	11-25-2014 22:57:34	738d 7h 18m 26s	1/3	Swap OK - 794752 Kb free swap
	INFRA-PING	OK	11-25-2014 22:56:47	738d 7h 19m 3s	1/3	PING OK - Packet loss = 0%, RTA = 0.10 ms

Figura 1 – *Nagios Service Status*

Os dados capturados são armazenados em ficheiros de log mas, podem ser guardados em base de dados MySQL.

O Nagios tem como principais vantagens [Ruivo, 2010]:

- Permite a monitorização de terminais Windows e Linux, serviços e outros equipamentos, desde que tenham conectividade;
- Desenvolvimento de *plugins* em diversas linguagens;

- Definição hierárquica de alarmística, se um servidor for desligado será enviada uma notificação a alertar para o seu novo estado, assumindo todos os serviços dele dependentes;

- A monitorização dos diversos sistemas é realizada de forma paralela;

- As notificações são enviadas quando o alarme crítico se mantém durante três verificações seguidas e quando este recupera, passando do estado *Critical* a *OK*, podem ser feitas por email, SMS ou por outro meio criado;

- Alarmística intrusiva (*Active Checks*) e não intrusiva (*Passive Checks*);

- Consulta de histórico e log de eventos;

- Relatórios de análise de tendências e de disponibilidade - Figura 2.

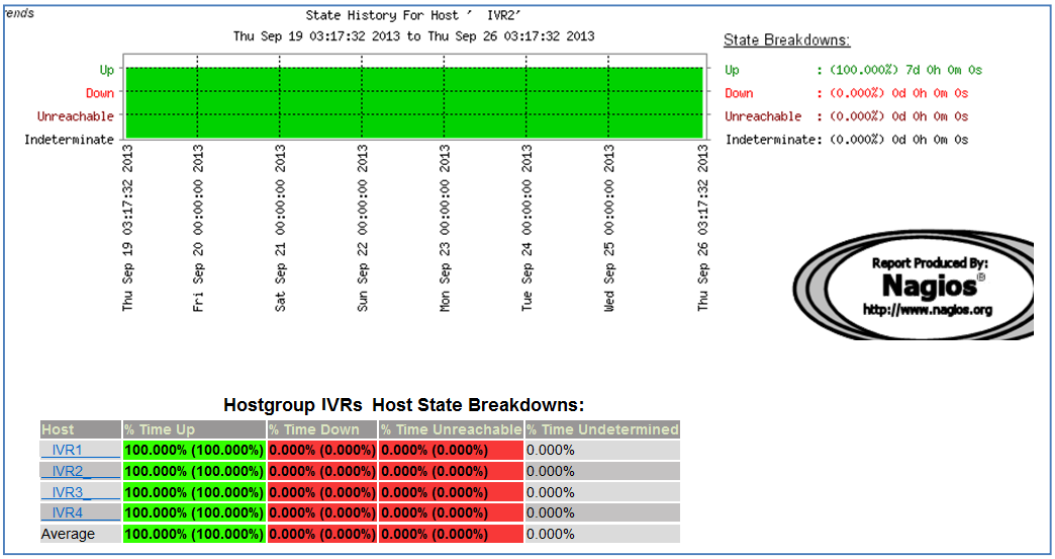

Figura 2 – Nagios - Relatório de Análise de Tendência e disponibilidade

Durante a utilização do sistema foram sentidas as seguintes dificuldades:

- O sistema não possui ferramenta gráfica de configuração, todo este processo é realizado de forma manual, tornando a sua gestão complicada;

- Subvalorização da informação, ou seja, com os dados recolhidos podia disponibilizar um maior conjunto de gráficos para permitir uma observação mais abrangente da plataforma monitorizada.

2.1.2 Cacti

O Cacti [Cacti, 2004] é uma ferramenta web que permite a análise do estado de plataformas e serviços através de gráficos. Desenvolvida para plataformas Linux e de licenciamento gratuito, é de fácil instalação e usabilidade. Utiliza o *Round Robin Database Tool* (RRDTool), sistema criado por Tobias Oetiker, para o armazenamento de séries de dados numéricos para a criação de gráficos (Figura 4) e para guardar em base de dados MySQL [Oetiker,2013]. Os parâmetros dos gráficos gerados são configuráveis, dando ao utilizador a possibilidade de mudar o conteúdo, tamanho e cor.

Utiliza o protocolo SNMP para a recolha de informação da MIB dos sistemas observados. Porém pode utilizar *scripts*, ou *plugins* criados para essas funções.

O modo de monitorização que dispõe é do tipo intrusiva, ou seja, o Cacti solicita a informação às plataformas que nele estão definidas.

Toda a configuração da aplicação é realizada de forma intuitiva, através de uma página acessível na ferramenta web (Figura 3), dispondo dos seguintes parâmetros:

- Periodicidade de recolha de informação;
- Modo de visualização de omissão:
- Estrutura em árvore;
- Lista;
- Pré-Visualização;
- Definição dos valores de omissão de criação dos gráficos;
- Configuração da estrutura em árvore;
- Número de gráficos por página;
- Separador a utilizar na exportação dos dados para CSV;
- Tipo de gráficos - barra, linear, correlação e diagrama circular.

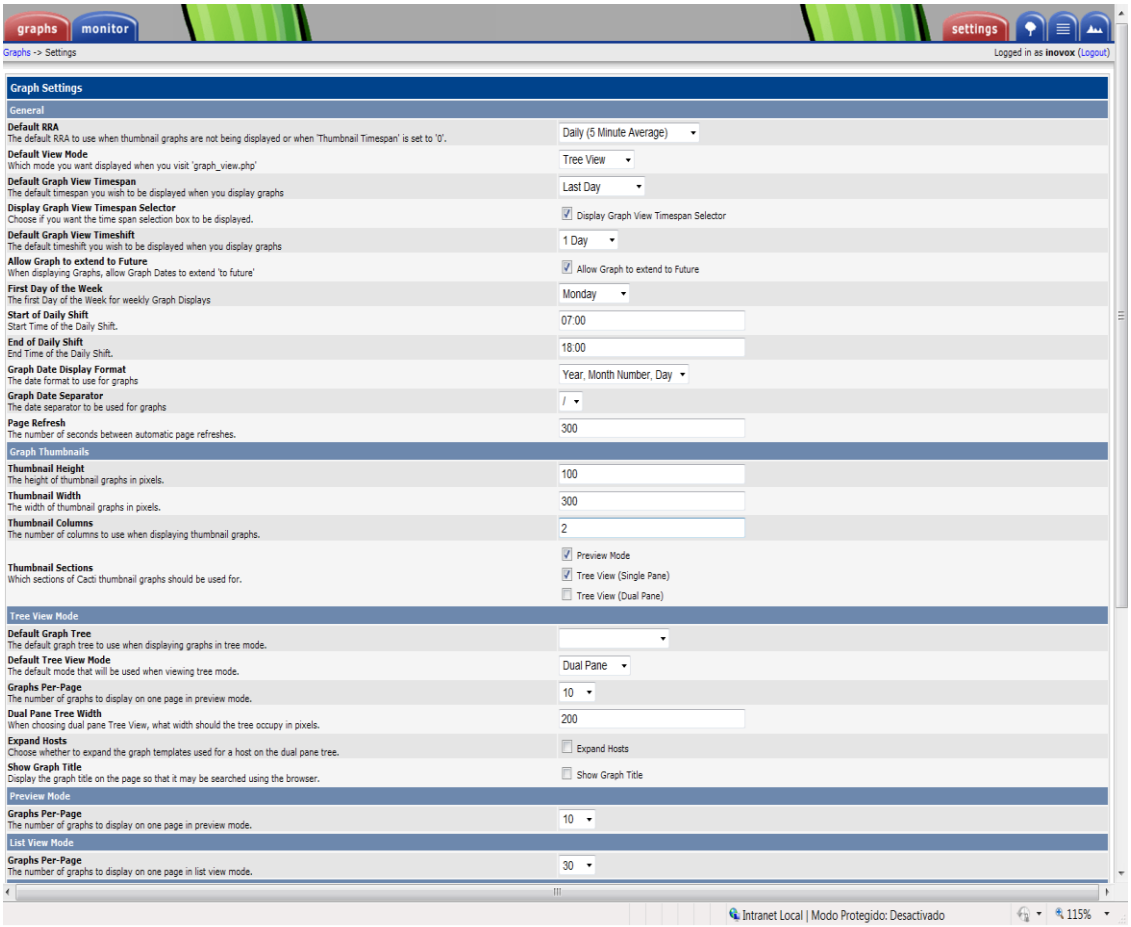

Figura 3 – Cacti - Página de configuração

Aquando da instalação, a aplicação já dispõe de alguns modelos de monitorização definidos para alguns tipos de dispositivos, como terminais Linux, Netware, Windows e *routers* Cisco. No entanto, em qualquer momento pode ser definido um novo tipo de modelo, indicando as características da plataforma, a estrutura de MIB que usa e quais os dados a utilizar [Cacti, 2004].

Para aceder à aplicação é preciso estar registado. É multiutilizador e podem ser criados diferentes perfis, restringindo os conteúdos visíveis.

O sistema possibilita ao utilizador com poucos conhecimentos de rede o domínio da informação gerada, pois toda a interação realizada é no modo gráfico. No entanto, tem limitações ao nível de fornecimento de comandos de controlo, o esquema de eventos é limitado, *trap* sem reconhecimento, estando assim desprotegida para entidades externas ao negócio.

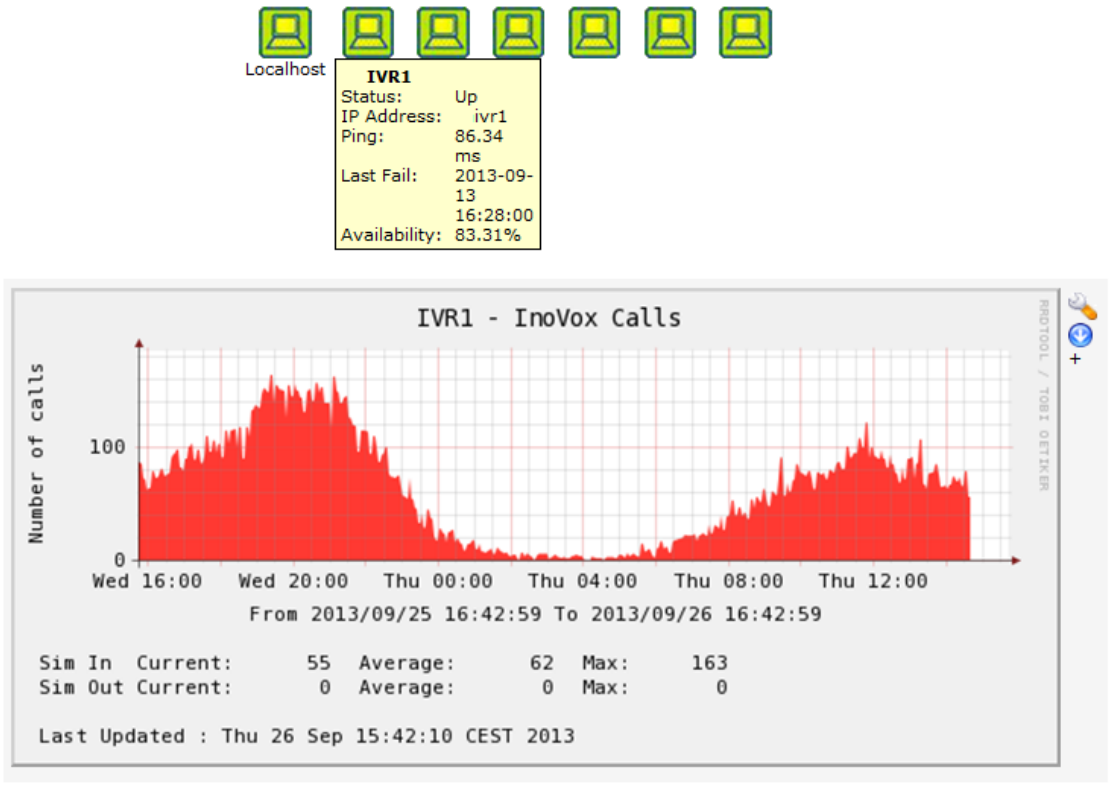

Figura 4 – Cacti - Gráfico e Informação gerada de um IVR

2.1.3 Zabbix

O Zabbix [Zabbix, 2001] das ferramentas referidas é a mais recente no mercado e a mais utilizada.

A maioria das aplicações deste tipo são orientados para alarmística ou para geração de gráficos para análise de sistemas e serviços. Neste caso, aplica-se os dois modos de operação.

Foi desenvolvido para Linux sobre licenciamento *Gnu Public License* (GPL). A sua arquitetura é do tipo servidor-agente, ou seja, os servidores de cliente têm de ter os agentes desta componente instalados para poderem ser monitorizados. Existem agentes para os terminais Linux, Unix, MacOS X, FreeBSD, Netware, Solaris, Windows e dispositivos que têm o protocolo SNMP a funcionar [Santos, 2008].

Possui uma arquitetura distribuída (Figura 5), em que a informação não está centralizada no gestor de monitorização, podendo ser armazenada em bases de dados, utilizando um dos seguintes Sistemas de Gestão de Base de Dados (SGDB) MySQL, PostgresSQL, Oracle e SQLite. O MySQL e PostgreSQL são ideais quando se tem uma grande quantidade de servidores configurados para um único servidor de base de dados. O Oracle recomenda-se quando existe um elevado número de terminais em recolha de informação para um *cluster* de base de dados. O SQLite sugestivo para cargas de dados baixas.

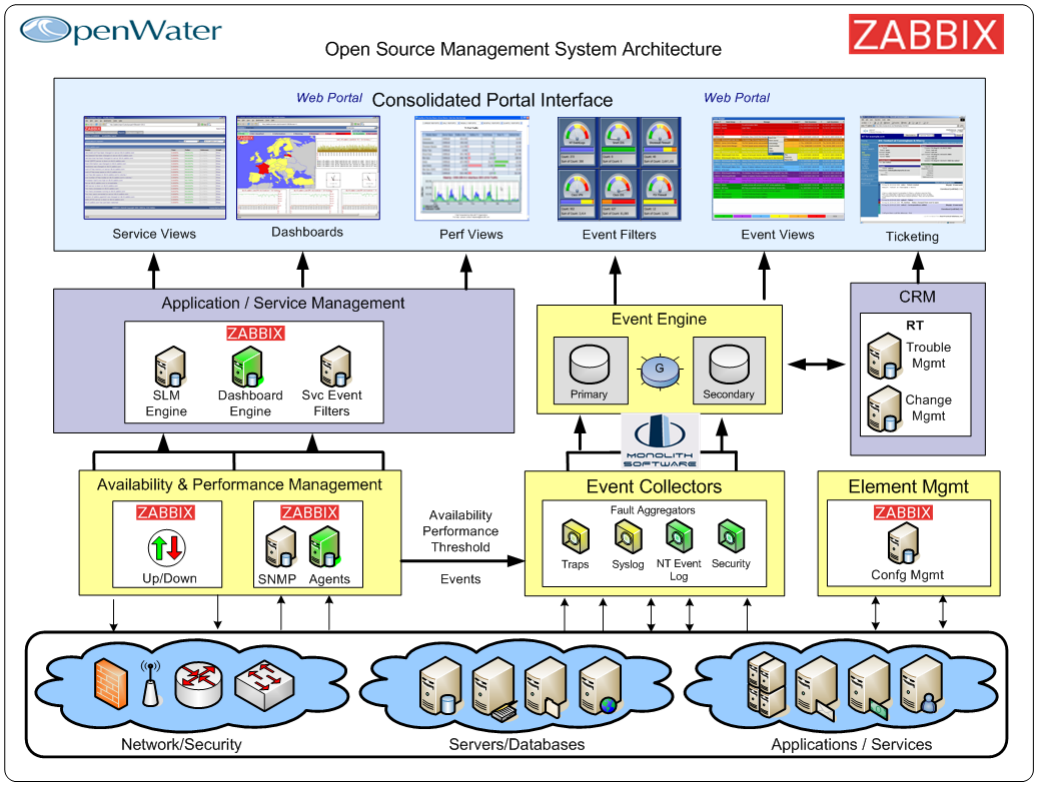

Figura 5 – Zabbix - Modo de Funcionamento [Gonçalves, 2010]

O Zabbix divide-se em três grandes áreas que dispõem as funcionalidades de resposta necessária a cada uma:

- Monitorização em tempo real, onde os sistemas podem ter os seguintes estados *Disaster*, *High*, *Average*, *Warning*, *Information*, *Not Classifie*d, ordenados do mais grave e mais importante para o menos grave e importante;
- *Dashboard* - são apresentados os estados no formato de tabelas estando agrupadas por grupos de servidores e últimas ocorrências de alarmística;
- *Overview* - matriz de servidores por tipo de alarme, onde os estados são assinalados por cores;
- *Latest Data* - últimos dados recebidos por terminal;
- *Triggers* - lista dos últimos alarmes despoletados;
- *Events* - últimos eventos detetados;
- *Graphs* - criação de gráficos por servidor ou grupos de servidores e por componente, no intervalo de tempo selecionado, desde 1 hora até 6 meses;

- *Screens* - possibilita a configuração de vários tipos de informação, para ficar disponível numa única página (Figura 6);
- *Maps* - criação de mapas de rede;
- *IT Services* - listagem de serviços, indicando estado, em caso de problemas a razão e respetivo *Service Level Agreement* (SLA);
- *Inventory* - indica quantos servidores estão registados e quais;
- Relatórios de disponibilidade dos processos com o maior número de alterações de estado.

A configuração é demorada devido à quantidade de componentes disponibilizadas. Atraso no retorno de eventos, tornando muitas vezes a informação obsoleta pois, a data e hora do evento corresponde ao momento em que foi entregue ao Zabbix e, não ao momento da ocorrência.

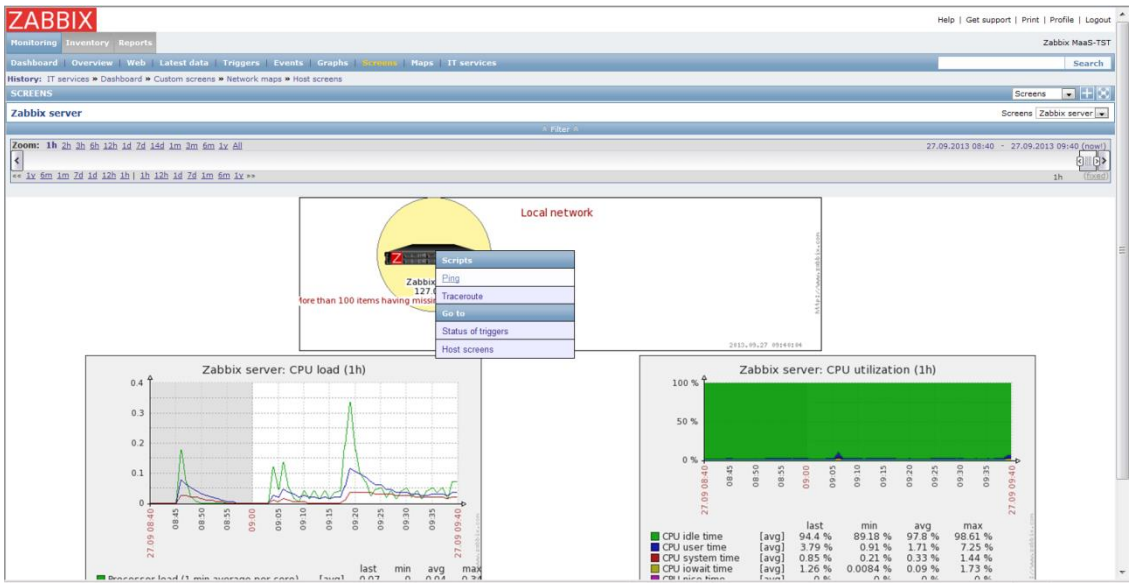

Figura 6 – Zabbix - *Monitoring* - *Screens*

2.1.4 Comparação

No estudo das três ferramentas verificou-se que todas elas ajudam ao controlo e operação de sistemas, permitindo a observação em tempo real de servidores e serviços. Todas elas são de fácil instalação. Quanto à configuração, o Cacti e o Zabbix possuem componente gráfica para o fazer, enquanto o Nagios recorre a ficheiros.

Este último foca mais a componente de alarmística, ou seja, consulta de alarmes, embora disponha de alguns gráficos. O Cacti contrariamente ao anterior, concentra-se na criação de gráficos, permitindo gerar gráficos de diversos tipos e diferentes parametrizações. O Zabbix é o melhor dos dois anteriores, é composto por uma forte componente de alarmística e de geração de gráficos.

Todos os sistemas tornam-se um pouco lentos a gerar gráficos e por vezes, há atraso na entrega de alarmes e eventos.

A Tabela 2 foi adaptada de um estudo comparativo realizado [Black, 2008] e completada com informação obtida na experimentação das ferramentas.

O Nagios, Cacti e Zabbix permitem a utilização de scripts externos, *plugins* e criação de gráficos. Todas elas fazem monitorização distribuída de plataformas e serviços e possuem sistemas de alertas. Podem recorrer ao protocolo SNMP e aos logs gerados pelo syslog para recolha de dados mas, no caso do Nagios é preciso a instalação de *plugins* para o fazer. A criação de mapas está disponível em todas elas, no caso do Cacti recorrendo a um *plugin*. No caso da geração de inventários e eventos, para o primeiro são necessários *plugins* para o Nagios e Cacti e no segundo para o Cacti. Todas elas possibilitam o armazenamento de dados em base de dados MySQL, no caso do Cacti também pode armazenar em RRDTool e no Zabbix dispõe de opções como PostgreSQL, Oracle e SQLite. São de licenciamento GPL e de fácil instalação. O sistema mais robusto é o Zabbix, fazendo com que dos três seja o mais difícil de operar. No entanto, numa vertente comercial existe um mais robusto ainda, denominado SCOM 2012 da Microsoft mas, devido ao seu elevado preço e requisitos de *hardware*, as pequenas e médias empresas acabam por optar pelo Zabbix[Marik e Zitta,2014].

Tabela 2 – Comparação das ferramentas estudadas

	Nagios	**Cacti**	**Zabbix**
Processo de Instalação	Fácil	Fácil	Fácil
Configuração	Através de Ficheiros	Através de componente gráfica	Através de componente gráfica
Armazenamento de Dados	MySQL	RRDTool, MySQL	MySQL, PostgreSQL, Oracle, SQLite

Desenvolvido na Linguagem	Perl	PHP	C e PHP
Agentes	Sim	Não	Sim
SNMP	Plugin	Sim	Sim
Syslog	Plugin	Sim	Sim
Utilização de Scripts Externos	Sim	Sim	Sim
Plugins	Sim	Sim	Sim
Sistema de Alertas (Alarmística)	Sim	Sim	Sim
Monitorização Distribuída	Sim	Sim	Sim
Criação de Gráficos	Sim	Sim	Sim
Criação de Mapas	Sim	Plugin	Sim
Inventário	Plugin	Plugin	Sim
Eventos	Sim	Plugin	Sim
Grau de Dificuldade de Operação	Baixo	Baixo	Médio

Licenciamento	GPL	GPL	GPL

2.2 *Data Mining*

Todos os dias é armazenada uma quantidade enorme de dados, que devidamente tratados podem fornecer conhecimento vital para uma organização.

Para o processamento desses dados a fim da aquisição de conhecimento é preciso aplicar as cinco fases do processo *Knowledge Discovery in Databases* (KDD) (Figura 7):

- Seleção - de todos os dados disponíveis são apenas recolhidos os considerados importantes para o negócio;
- Pré-Processamento - limpeza dos dados de forma a garantir a sua qualidade e veracidade;
- Transformação - os dados são transformados para garantir que estão no formato adequado para aplicação do algoritmo de *Data Mining* escolhido;
- *Data Mining* - aplicação de algoritmos para extração de padrões e conhecimento;
- Interpretação e Avaliação - a informação retornada é analisada, face ao objetivo de negócio definido.

O *Data Mining* é a transformação e análise de uma grande quantidade de dados procurando padrões, regras e associações que possibilitem a extração de novo conhecimento [Berry e Linhoff, 1997].

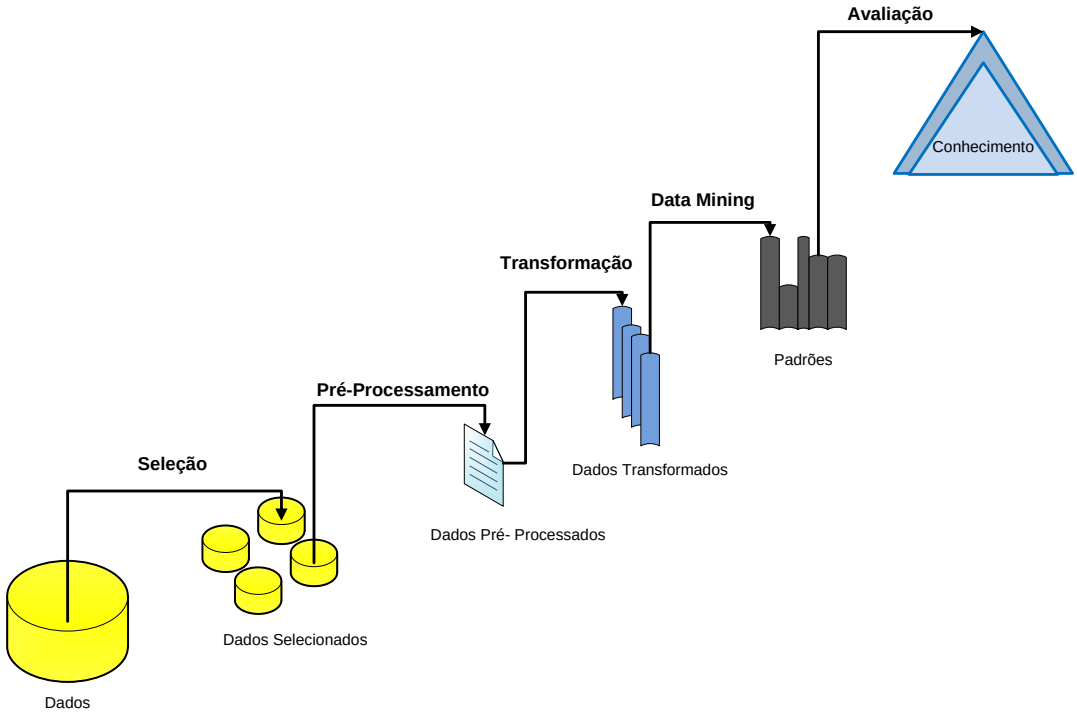

Figura 7 – Fases do Modelo KDD [Fayyad, U.M. et al.,1996]

São diversos os métodos de *Data Mining* e podem ser classificados como não supervisionados e supervisionados. No primeiro, não é preciso trabalho prévio, ou seja, os atributos são agrupados mediante a correlação existente entre eles. No segundo caso, para inferir conhecimento é preciso uma pré-classificação, em que os atributos a serem processados percorrem as regras pré-definidas para serem agrupados num padrão que permita obter novo conhecimento.

Os métodos não supervisionados mais usuais são os de agrupamento e de associação e supervisionados os métodos de classificação.

Para cada método existem diversas técnicas e dentro destas, diversos algoritmos à escolha. Para um mesmo resultado estão disponíveis diversas técnicas, a sua escolha tem por base o tipo de dados de entrada e a apresentação do resultado.

Selecionada a técnica passa-se à tomada de decisão de qual o algoritmo a utilizar tendo em consideração questões de precisão de previsões, rapidez de processamento de dados e retorno de resultados e o tipo de dados existente na amostra de dados. Os dados podem ser caracterizados como paramétricos e não paramétricos. Num a distribuição dos mesmos é conhecida, no outro não.

Para a escolha adequada do método, técnica e respetivo algoritmo a utilizar tem de ser conhecida a área de negócio, os tipos de dados existentes e que tipo de problema é pretendido dar resposta.

2.2.1 Métodos de Agrupamento (*Clustering*)

No método de agrupamento os dados são analisados e agrupados por similaridade entre atributos. É do tipo não supervisionado, isto é, não necessita de uma classificação prévia. Os grupos retornados têm de ser os mais semelhantes em si e os mais opostos entre si [Bramer, 2007].

Quando uma amostra de dados tem muitos atributos de entrada este método pode tornar-se muito lento a processá-los. Caso não exista uma medida de distância óbvia entre os elementos da amostra de dados, tem de ser definida uma, o que nem sempre é uma tarefa fácil. O resultado retornado muitas vezes pode ser interpretado de diferentes maneiras.

Na Figura 8 são identificados facilmente quatro grupos, caracterizados pela distância. Cada elemento pertence a um determinado grupo se tiver perto da distância definida para esse grupo e longe da definida para os outros.

Este método é muito usado para agrupar clientes, possibilitando uma traçagem de perfis para os diversos grupos. Assim, a definição da estratégia é direcionada ao perfil.

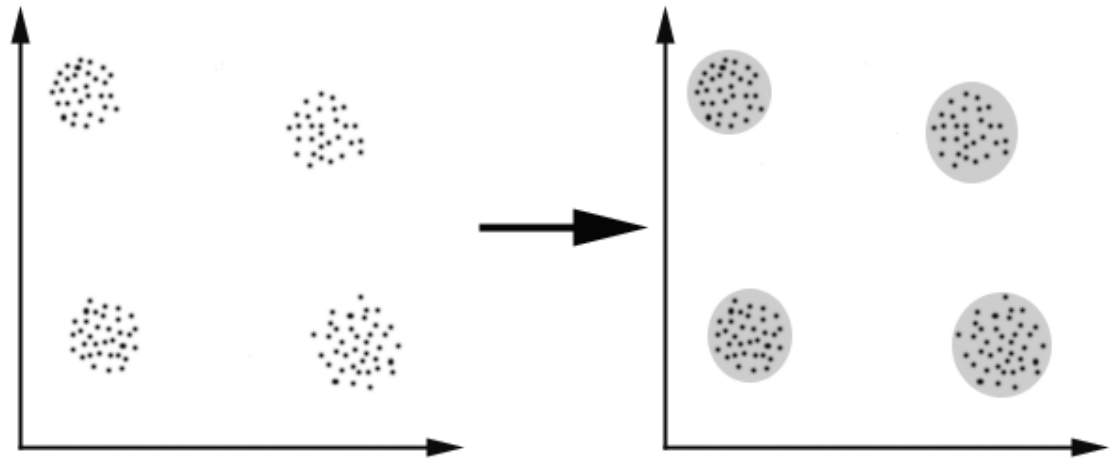

Figura 8 – Exemplo de Agrupamento

2.2.2 Métodos de Regras de Associação

É um método não supervisionado que procura elos de ligação entre os dados de um conjunto de atributos. Estes relacionamentos são chamados de regras de associação. Uma regra de associação é A implica B, para A e B conjuntos disjuntos de itens de dados. [Burkle, 2006].

Para validar uma regra, ou melhor, para verificar o seu grau de importância é preciso calcular o Suporte e a Confiança. O primeiro valida a importância estatística da regra, o segundo a sua força. O suporte equivale ao número de vezes que A e B ocorrem na amostra de dados. A confiança é o número de vezes que B ocorre em A.

Com a amostra de dados representada na Tabela 3, podem ser definidas as seguintes regras:

- {fralda,leite} $\rightarrow$ {cerveja} (S=0.4, C=0.67);
- {cerveja,leite} $\rightarrow$ {fralda} (S=0.4, C=1.0);
- {cerveja,fralda} $\rightarrow$ {leite} (S=0.4, C=0.67);
- {cerveja}$\rightarrow${fralda,leite} (S=0.4,C=0.67);
- {fralda} $\rightarrow$ {cerveja,leite} (S=0.4, C=0.5);
- {leite} $\rightarrow$ {cerveja,fralda} (S=0.4, C=0.5) [Alvares, 2013].

Em termos de suporte têm todas o mesmo valor mas, como a confiança da regra é maior que as outras {cerveja,leite} $\rightarrow$ {fralda} (S=0.4, C=1.0), é esta a regra mais forte nesta amostra de dados.

As regras de associações são muito usadas como retratado neste exemplo, de análise de cesto ou carrinho de compras. Este tipo de análise possibilita a organização de prateleiras de um supermercado de uma forma estratégica e a realização de promoções e campanhas de incentivo ao consumo.

Tabela 3 – Amostra de Dados

ID	**PRODUTOS**
1	leite, pão
2	cerveja, fralda, ovos, pão
3	cerveja, fralda, leite, refrigerante
4	cerveja, fralda, leite, pão
5	fralda, leite, pão, refrigerante

2.2.3 Métodos de Classificação

A classificação está presente em todo o nosso quotidiano. Diariamente são observadas características dos objetos que nos rodeiam e de uma forma, quase automática os organizamos em grupos.

Os métodos de classificação permitem através da análise de atributos de um conjunto de entidades, estabelecer uma relação que permita agrupá-las em classes.

Para a escolha do método de classificação é preciso ter em conta o tipo de informação a tratar. Nos métodos paramétricos a informação disponível tem uma distribuição que possibilita a indução dos dados que se seguem, fazendo com que o conhecimento seja condicionado a alguns parâmetros, como a regressão linear [Rodrigues, 2005].

Nos métodos não paramétricos a informação não tem uma distribuição normal, não se conseguindo perceber qualquer sequência de valores, permitindo que a inferência de conhecimento seja mais abrangente [Rodrigues, 2005].

A classificação é do tipo supervisionado, ou seja, para inferir conhecimento precisa de uma classificação prévia, antes de começar a processar os dados da amostra.

Este método é muitas vezes utilizado para problemas, como autorizar um pedido de empréstimo, classificando um conjunto de perfis em empréstimo autorizado e não autorizado. De todas as técnicas de classificação de *Data Mining* conhecidas, as mais utilizadas são as árvores de decisão, redes neuronais e classificação Bayesiana.

2.2.3.1 Árvores de Decisão

Árvores de decisão é uma das técnicas de prospeção de dados, em que através dos atributos de entrada selecionados, são definidas classes [Bramer, 2007].

A sua representação é hierárquica, sob a estrutura de uma árvore invertida, por outras palavras, construída da raiz para as folhas.

É um modelo considerado supervisionado pois, os dados fazem-se acompanhar de um valor alvo, que é definido através de análise estatística, subdividindo a amostra em conjuntos.

Ao longo da sua construção tenta descobrir uma ligação entre os valores de entrada e o valor alvo. Quando é identificada uma relação forte entre ambos, os atributos de entrada identificados são todos agrupados dando origem a um ramo. Cada ramo dá origem a uma árvore mais pequena, até atingir as folhas - os nós de decisão. É construída do geral para o particular, em que a variável alvo, é classificada através dos valores de entrada.

As árvores de decisão têm como principais vantagens [Rokach, 2008]:

- Representação fácil de interpretar e avaliar;
- Não requer uma preparação de dados muito minuciosa;
- Permite trabalhar com variáveis de diferentes tipos na mesma amostra de dados;
- Os resultados podem ser validados através da aplicação de conhecimentos estatísticos.

Este tipo de modelo pode originar instabilidade nos resultados pois, pequenas variações de dados podem gerar árvores completamente diferentes. O excesso de dados de entrada pode provocar um problema de *overfitting*, dando origem a estruturas demasiado complexas em que os dados não são bem agrupados, tornando difícil o processo de interpretação das relações identificadas. Este tipo de ocorrência existia no algoritmo ID3, mas com a evolução para o C4.5 em que é aplicada a técnica de *pruning* que consiste em retirar apenas as amostras necessárias para a caracterização da classificação final - variável alvo, foi corrigido.

No exemplo representado na Figura 9 são identificadas os diversos componentes de uma árvore de Decisão. O objetivo nesta estrutura gerada é de classificar os dados em [Sair de Casa] ou [Ficar em Casa]. Os registos que têm sim na variável [Está Sol] e na variável [Está Quente] pertencem à classe [Sair de Casa]. Os que têm sim na variável [Está Sol] e não na variável [Está Quente] são da classe [Ficar em Casa]. Os registos que têm não na variável [Está Sol] e na variável [Está Vento] são do grupo [Sair de Casa]. Os registos que têm não na variável [Está Sol] e sim na variável [Está Vento] estão aglomerados na classe [Ficar em Casa].

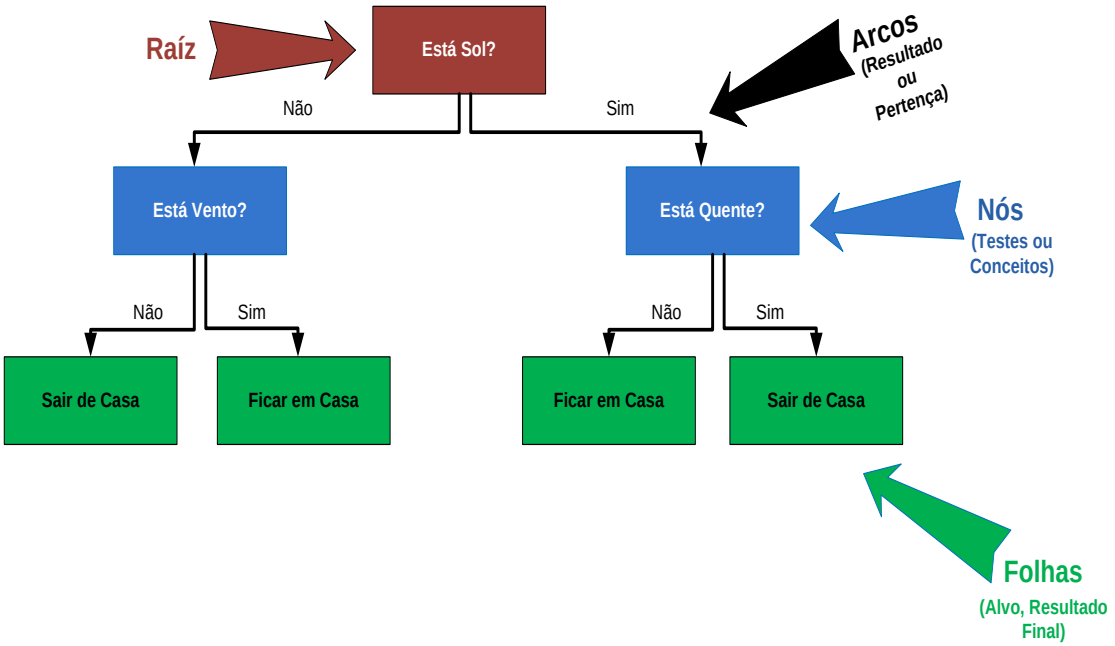

Figura 9 – Exemplo de uma Árvore de Decisão

2.2.3.2 Redes Neuronais

Redes neuronais são um tipo de modelo que tem uma estrutura e funcionamento semelhante ao cérebro humano [Moreira,1997]. É composta por um conjunto de nós (neurónios) organizados em camadas e interligados entre si. Estas ligações funcionam como caminhos com diferentes pesos atribuídos a cada um, levando o sinal de entrada até ao nó de saída, ou seja, a aprendizagem é feita através da modificação entre as camadas, os percursos tomados (Figura 10) [Moreira,1997]. A sua estrutura é afetada através da informação externa e interna que circula na rede durante a fase de aprendizagem.

A rede recebe um conjunto de dados de treino para inferir o primeiro conhecimento, a partir daí está apta para fazer predições a partir de novos dados, alterando a sua estrutura conforme a sua base de conhecimento aumenta.

Existem diferentes topologias:

- Rede Totalmente Conectada - cada neurónio está ligado a todos os outros;
- Rede de Camada Única - constituído no mínimo por duas camadas e a informação nesta topologia circula apenas numa direção, dos nós de entrada para os de saída;
- Rede de Múltiplas-Camadas - constituída por diferentes camadas dispostas paralelamente, o fluxo de informação é bidirecional, isto é, os dados de camadas posteriores de processamento podem difundir-se para camadas anteriores.

Os pesos das interligações dos nós da estrutura vão sofrendo alterações ao longo do processo de aquisição de conhecimento, a fim de satisfazer o melhor possível, a relação nós de entrada versus nós de saída.

Subsistem três métodos de aprendizagem [Cruz, 2007]:

- Aprendizagem por Reforço - para um dado conjunto de dados de entrada não são definidas as saídas corretas, aplicando o conceito de prémio ou castigo se as saídas retornarem informação útil ou não, respetivamente;

- Aprendizagem Supervisionada - o sistema tem as respostas e a aprendizagem é feita através do conhecimento construído entre os dados de entrada e os de saída. Se os resultados obtidos forem idênticos aos definidos, os pesos das ligações estão corretos, caso contrário, terão de ser determinados novos pesos;

- Aprendizagem Não Supervisionada - é usada quando não se possui uma classificação inicial, a aprendizagem ocorre por adaptação da estrutura às associações e padrões identificados na amostra de dados de entrada.

A utilização de redes neuronais tem inúmeras vantagens, como [Singh e Chauhan, 2009]:

- Alta precisão - permite a criação de mapas não lineares complexos;

- Tolerância ao ruído - flexibilidade quanto ao processo de limpeza dos dados de entrada;

- Processamento paralelo - quando um nó da rede falha, a mesma continua em funcionamento, não existindo quebra no processo;

- Auto Aprendizagem - não necessita de conhecimentos de peritos para a tomada de decisão, as decisões são tomadas com base no histórico que possui.

Este tipo de estrutura apresenta as seguintes desvantagens:

- Apresentação por vezes, de resultados dúbios, isto é, a resposta que obtemos contraria as regras do negócio;

- Todo o processo de treino de uma rede é demorado e necessita de um grande volume de dados para que o processo de aprendizagem seja o mais fidedigno possível.

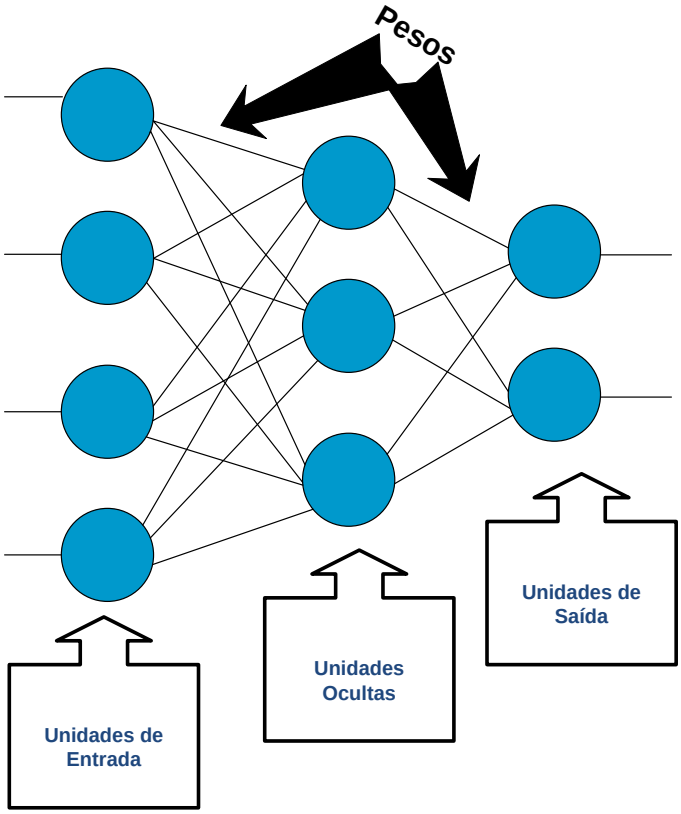

Figura 10 – Estrutura das Redes Neuronais

2.2.3.3 Classificação Bayesiana

Modelo baseado na teoria de probabilidade de Thomas Bayes, que consiste no cálculo da probabilidade de que um novo atributo de entrada faça parte de uma classe definida à priori [Fred, 2006].

Um dos seus algoritmos mais conhecidos é o *Naive Bayes*, devido à sua simplicidade e ao facto de o peso de um atributo sobre uma determinada classe ser totalmente independente dos outros atributos.

A Classificação Bayesiana usa o método de aprendizagem supervisionada pois, necessita de um conjunto de dados de treino já estruturados em classes. A estrutura recebe uma nova amostra de dados e com base na classificação referida devolve a classe mais provável a que os atributos recebidos pertencem, através da execução de cálculos probabilísticos [Singh e Chauhan, 2009].

De uma maneira simplista este algoritmo processa-se em três passos:

- Cálculo de probabilidade das classes definidas previamente;
- Cálculo da probabilidade da amostra desconhecida;
- Cálculo da probabilidade da amostra desconhecida a cada uma das classes definidas, a que tiver uma maior probabilidade é o resultado esperado.

Este modelo necessita de um conjunto de treino com muitos registos e os atributos que caracterizam cada instância têm de ser condicionalmente independentes.

No exemplo representado na Figura 11, é usada uma rede de Bayes dinâmica para a classificação de utilizadores e em que os nós são ajustados ao longo do tempo. Os nós representados nesta rede apresentam as seguintes descrições:

- Ajuda - número de vezes que o utilizador acionou a ajuda da aplicação;
- Cliques - Corresponde ao número de cliques que o utilizador necessitou para executar uma determinada ação;
- Interação - número de vezes que o utilizador interagiu de forma errónea na aplicação.

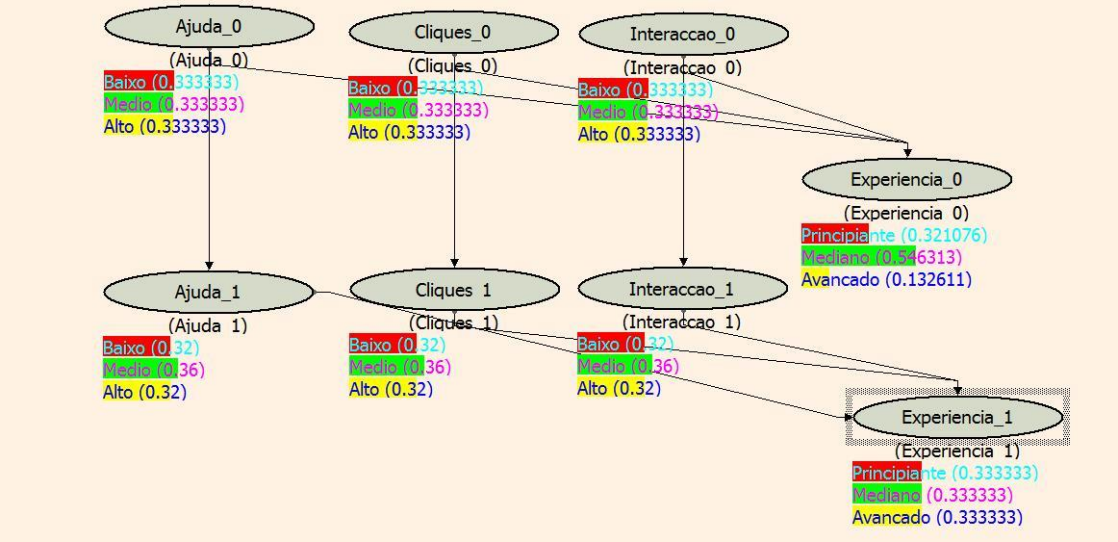

Figura 11 – Exemplo de uma Rede de Bayes Dinâmica

A definição do tipo de utilizador é obtida consoante os valores dos nós apresentados anteriormente. Para esta rede, o nó “Ajuda” influencia de forma negativa o tipo de utilizador,

sendo que quanto mais vezes o utilizador pede ajuda menos experiência tem. O nó cliques também influencia negativamente o tipo de utilizador, sendo que quanto mais cliques o utilizador der para realizar uma ação menos experiência tem. No sentido contrário, o nó interação influencia positivamente o tipo de utilizador, sendo que quanto mais sugestões forem aceites e seguidas pelo utilizador mais experiência este irá ter. Os utilizadores são classificados em três tipos: principiante, mediano e avançado.

2.2.3.4 Algoritmos de Árvores de Decisão

Neste subcapítulo é feito um pequeno estudo dos algoritmos de geração de árvores de decisão mais conhecidos. São abordados o ID3, C4.5, C5.0, CHAID, CART, QUEST.

2.2.3.4.1 ID3

Foi desenvolvido por J. Ross Quinlan na Universidade de Sydney, em 1986 [Quinlan,1986a]. Este algoritmo cria árvores de decisão usando o conceito de Entropia de Claude Shannon, 1948 [Crawford, 2011].

A entropia permite quantificar o grau de incerteza dos valores da amostra de dados. Quanto menor a entropia menor é o grau de incerteza. É calculada para todos os atributos, exceto os que foram usados na classificação inicial.

Na construção da árvore é escolhido o valor com menor entropia pois, é o que tem maior ganho de informação e é criado um nó desse atributo e assim sucessivamente até não restarem mais atributos.

Os resultados retornados por este algoritmo são de fácil perceção, graças às regras de previsão que gera. A estrutura de classificação em árvore é construída rapidamente e não atinge uma dimensão com muitos níveis, precisando de testar apenas atributos suficientes para todos os dados serem classificados. O número de testes a realizar é reduzido porque na construção dos nós folha, ou seja, das classes, vai testando alguns dados que vão sendo cortados da realização dos testes [Rizvi, 2010].

A utilização do ID3 para classificação contínua, exige elevados recursos computacionais pois, testa um atributo de cada vez para a tomada de decisão.

2.2.3.4.2 C4.5

É baseado no ID3, é uma evolução do mesmo elaborado por Quinlan em 1993 [Kohavi e Quinlan, 1999]. Foi desenvolvido para corrigir problemas do ID3 como [Hamilton, H., 2012]:

- Determinar até que nível deve crescer uma árvore de decisão;
- Regras pós-poda;

- Utilização de atributos do tipo contínuos;
- Selecionar a medida de seleção apropriada para um atributo;
- Suporte de amostras de dados com atributos a vazio;
- Utilização de atributos com diferentes custos;
- Melhoria da performance de execução.

Assim sendo, o C4.5 é uma melhoria do ID3.

2.2.3.4.3 C5.0

É uma evolução do C4.5 que surgiu em 1997, com uma vertente mais comercial [Quinlan, 2010b].

Neste algoritmo as árvores de decisão retornadas são mais pequenas que nas versões anteriores. Permite atributos do tipo data, tempo e valores nominais e possibilita a aplicação de custos à má classificação de atributos.

Tem presente a propriedade de *Boosting,* conceito baseado no estudo feito por Freud e Schapire, em 1997, que possibilita a combinação de vários classificadores para melhorar a precisão da previsão [See5, 2012].

O algoritmo possui um mecanismo de pré-seleção dos atributos mais relevantes.

Em suma, as árvores criadas são cinco vezes mais rápidas que as C4.5 e 40% mais pequenas. As regras são geradas mil vezes mais rápido e usando 30% menos de memória. Para além disso, o sistema é *multi-thread* tirando melhor partido da tecnologia *multi-core* dos dias de hoje, usufruindo ao máximo de um servidor composto por mais do que um CPU [Quinlan, 2010b].

2.2.3.4.4 CHAID

É um dos algoritmos mais antigos de árvores de decisão, definido inicialmente por Kass em 1980. Baseado em conceitos estatísticos, que variam consoante o tipo de variável alvo [StatSoft, 2013].

Se a variável for do tipo contínua é-lhe aplicado o teste de Fisher, que calcula a probabilidade de associação dos valores em análise [Lowry, 2013].

Caso seja uma variável nominal é usado o teste Qui-Quadrado de Pearson, utilizado para validar a dependência ou independência entre variáveis, em que estas são colocadas em tabelas de contingência [Viali, 2012].

No caso das variáveis ordinais é feito o teste de rácio de verosimilhança, que possibilita a análise do vínculo entre um atributo e uma previsão.

Para este algoritmo assegurar que os dados retornados são significativos necessita de uma grande quantidade de dados. Não tem problemas de *overfitting,* ou seja, as árvores retornadas não são excessivamente complexas, independentemente da quantidade de dados processados [Rodrigues, 2005].

2.2.3.4.5 CART

O CART foi desenvolvido por Breiman, Freidman, Olshen, Stone em 1984 [Kohavi e Quinlan, 1999]. Gera árvores de decisão através do particionamento repetido dos dados. Inicia o particionamento dividindo os dados em duas partes, e essas partições podem ser separadas em partições, continuando sucessivamente até obter uma classificação. Como tal, o processo de particionamento é binário e recursivo.

Para a seleção da melhor partição de dados resultante, são usadas três medidas [Wu, X., Kumar, V., Quinlan, J.R., et al, 2007]:

- Entropia, também usada no algoritmo ID3;
- Critério de Gini - utilizado para calcular a desigualdade da distribuição;
- Critério de Twoing - separação da variável alvo em duas classes, tornando mais fácil o processo de descoberta da melhor partição.

A estrutura de árvores que é retornada é bastante complexa, composta por diversos níveis, dificultando a análise e avaliação dos resultados.

Este algoritmo permite o uso de variáveis independentes de diversos tipos e os dados não necessitam de sofrer transformações para serem processados.

O processamento deste sistema é demorado devido à complexidade dos cálculos que realiza para determinar a melhor partição [Steinberg, D., 1999].

2.2.3.4.6 QUEST

Algoritmo criado por Wei-Yin Loh em 1997 [Rodrigues, 2005]. O método utilizado para a classificação é, tal como no CART, construção de árvores binárias recursivas. As variáveis podem ser de qualquer tipo. No entanto, a variável alvo tem de ser nominal. O seu processo é dividido em duas partes, a seleção da variável alvo e seleção dos atributos de particionamento, isto é, os que são responsáveis pela classificação [Shih, 2012].

Para garantir a independência entre as partes referidas, aplica os seguintes testes estatísticos:

- Qui-Quadrado de Pearson, tal como no algoritmo de CHAID;
- F de ANOVA - verifica se a diferença entre as médias das partições é estatisticamente diferente [Balducci, 2012];
- F de Levene - para avaliar a igualdade entre variâncias de uma variável calculada para dois ou mais grupos.

A principal desvantagem deste algoritmo prende-se com a variável alvo pois, só pode ser de um tipo e como vantagem tem a rapidez na construção do modelo de classificação.

3 Estudo da Solução

Neste capítulo são definidas as necessidades sentidas através da identificação dos requisitos funcionais e não funcionais. Através deste estudo foi alcançada a solução proposta.

3.1 Âmbito

O âmbito desta dissertação de mestrado é o desenvolvimento de uma ferramenta denominada por IVR Analyzer que permita classificar, com base nalguns parâmetros de desempenho o estado atual duma solução IVR e que identifica a que classe pertence os atributos e valores que a caracterizam, apresentando recomendações perante o resultado obtido. Este sistema melhora a tomada de decisão na área de negócio das Telecomunicações.

3.2 Ambiente de Operação

O ambiente de operação do IVR Analyzer é a Internet. Para esta aplicação não é necessária a utilização de um sistema operativo específico pois, deverá estar acessível através de qualquer *browser*.

A implementação e estrutura estão pensadas para garantir uma forte usabilidade para o utilizador. Pretendeu-se que o ambiente fosse intuitivo desde a primeira utilização.

Não foram definidas restrições, mas ao longo da idealização do projeto, foram tidas em conta as necessidades sentidas pelos utilizadores finais.

3.3 Requisitos Funcionais

O sistema tem como objetivo principal permitir a análise de desempenho de plataformas do tipo IVR. Para tal tem de dispor das seguintes funcionalidades:

- Consulta de histórico do IVR;
- Visualização do estado atual da plataforma;
- Análise do percurso da plataforma;
- Retorno da consulta no formato de gráficos;
- Exportação da informação para Excel e editor de texto;
- Registos dos servidores e suas alterações poderem ser feitos a qualquer momento;
- Registo de utilizadores e sistema de *login* para acesso à aplicação.

O IVR Analyzer deve disponibilizar a função de registo de IVR e através das suas MIB a recolha de dados do seu desempenho. A informação recolhida é apresentada em gráficos no espaço temporal definido pelo utilizador, calculando os máximos, mínimos e médias atingidas. Com toda a informação armazenada classifica os servidores quanto ao desempenho, apresentando-a sob a forma de uma árvore de decisão.

3.4 Requisitos Não Funcionais

O sistema deve ter em conta os seguintes requisitos não funcionais:

- **Requisitos de Segurança** - as áreas restritas dos utilizadores devem ser protegidas através de *passwords* definidas pelos próprios. Não são, no entanto, exigidas especiais especificações de segurança. O utilizador autorizado deve efetuar *login* no sistema para poder realizar operação de manutenção do seu perfil;
- **Requisito de Interface** - deve ter uma interface de fácil utilização;
- **Requisito de Acessibilidade** - o sistema deve estar disponível 24 horas por dia durante 7 dias por semana, de forma a garantir aos seus utilizadores, o serviço sempre que necessitem;
- **Requisito de Integridade** - deve assegurar o devido funcionamento dos níveis de acesso à informação sem possibilidade de acesso ou manipulação, a utilizadores não autorizados.

3.5 Tipo de Utilizador

A aplicação deve ter dois tipos de utilizadores: o utilizador registado e o Administrador. Cada um deles possui um perfil com os seus dados que pode ser editado em qualquer momento.

Para efetuar o registo, o utilizador tem de preencher os seguintes campos:

- Nome de acesso;
- *Password*;
- Endereço de *email*.

Dentro do grupo dos Administradores existe o Super Administrador, ou seja, o administrador do sistema, que tem permissões totais, inclusive sobre os outros administradores.

O Administrador do Sistema tem permissões para eliminação de utilizadores e administradores, caso seja notificado ou em caso de desrespeito das regras de funcionamento da aplicação.

Tendo em conta os tipos de utilizadores e o ambiente em que a aplicação é implementada, o sistema deve cumprir os requisitos de ergonomia e interface de forma a qualquer pessoa usar a aplicação de maneira simples e intuitiva, tirando o maior proveito da solução.

3.6 Arquitetura

Tendo em conta as necessidades sentidas na empresa e os recursos que esta dispunha, foi desenhada uma solução.

Durante todo esse processo foram especificados os requisitos necessários e tendo como um dos principais a rapidez de retorno de informação. Para o alcance desse requisito, optou-se por uma arquitetura modular. Este tipo de estrutura visa minimizar problemas de performance e tempo de resposta. Para além disso, a arquitetura organizada sob a forma de componentes facilita o seu desenvolvimento, gestão e manutenção.

A solução foi agrupada por funcionalidades dando origem aos seguintes módulos:

- IVR Data Collector;
- Service IVR Analyzer;
- IVR Analyzer Portal.

A estrutura modular foi definida para resolver problemas de performance na recolha e tratamento dos dados. Toda a solução centra-se nos IVR, sendo esta a sua principal componente e fonte de dados.

3.6.1 IVR

É uma plataforma multisserviço, flexível e baseada numa arquitetura aberta, que disponibiliza aos Operadores de Telecomunicações, uma variedade de possibilidades para o fornecimento de serviços multimédia sobre redes *Public Switch Telephone Network* (PSTN), convergentes (protocolo mais utilizado SS7) ou totalmente IP (recorrendo ao protocolo SIP) (Figura 12). Possibilita a criação de serviços com interação DTMF ou por reconhecimento de fala, *Unified Messaging* para redes fixas e móveis, portais de voz, serviços de diretório e *voice browsing*, funcionalidades de *Media Resource Function* (MRF) para serviços de base numa arquitetura *IP-based Multimedia Services* (IMS).

As principais funcionalidades desta solução são:

- Atendimento e geração automática de chamadas;
- Deteção/geração de marcações DTMF;
- Reprodução de mensagens pré-gravadas, concatenadas ou não;
- Digitalização, compressão e armazenamento de sinais de voz e vídeo;
- Síntese por concatenação de datas, horas, quantias monetárias e números naturais (em múltiplas Línguas). São anúncios indexados, funcionam como uma matriz em que cada posição tem um anúncio diferente, concatenando as posições pode dar origem ao toque de datas, horas, entre outros;
- Deteção, atendimento e geração de chamadas de vídeo;
- Reprodução de vídeo;
- Gravação sincronizada de áudio e vídeo;
- Reprodução de ficheiros de vídeo;
- Reconhecimento Automático de Fala (ASR);
- Síntese de Texto para Fala (TTS);
- Conferência;
- FAX;
- *Frequency-Shift Keying* (FSK).

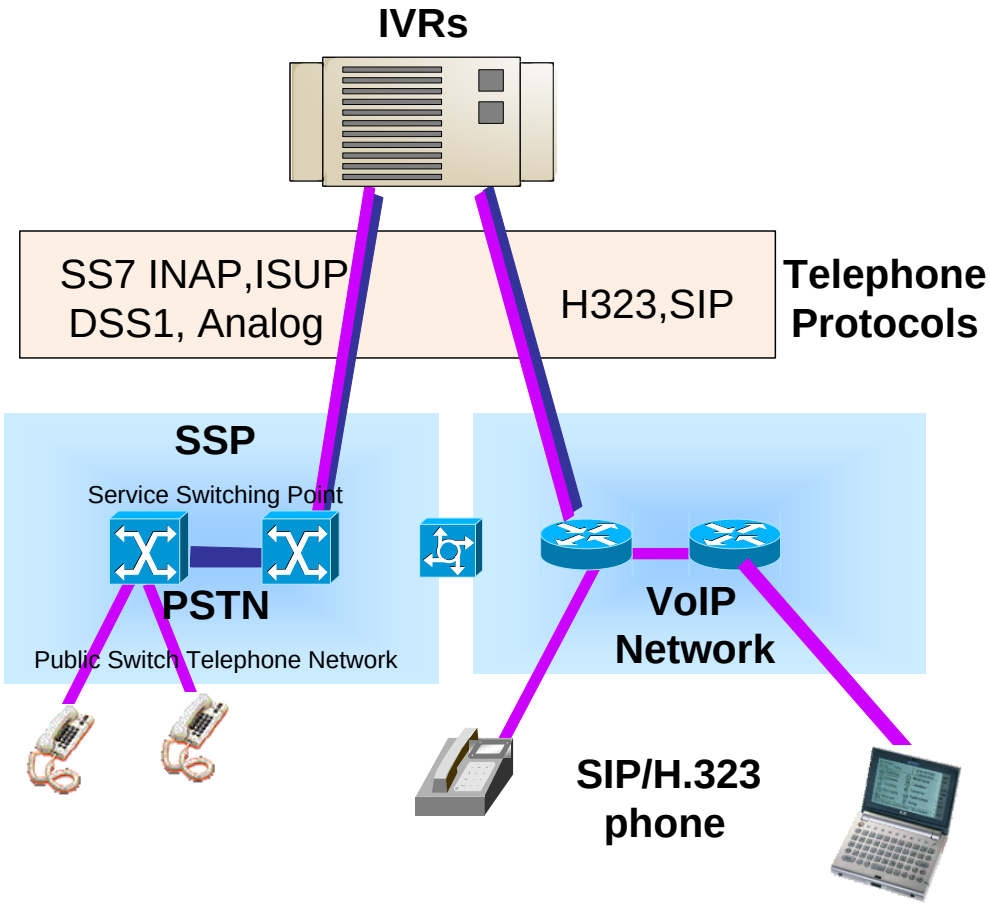

Figura 12 – Visão Geral de uma Solução IVR

3.6.2 IVR Data Collector

Este módulo é responsável pela recolha e tratamento dos dados da MIB dos servidores IVR, disponibilizando-os em ficheiros (Figura 13). Para isso, necessita de conectividade para a plataforma em análise e o uso de um protocolo para comunicação com o agente do sistema, por forma a recolher os dados selecionados. O IVR Data Collector é independente e autónomo, funcionando sem necessitar de qualquer ação por parte dos utilizadores.

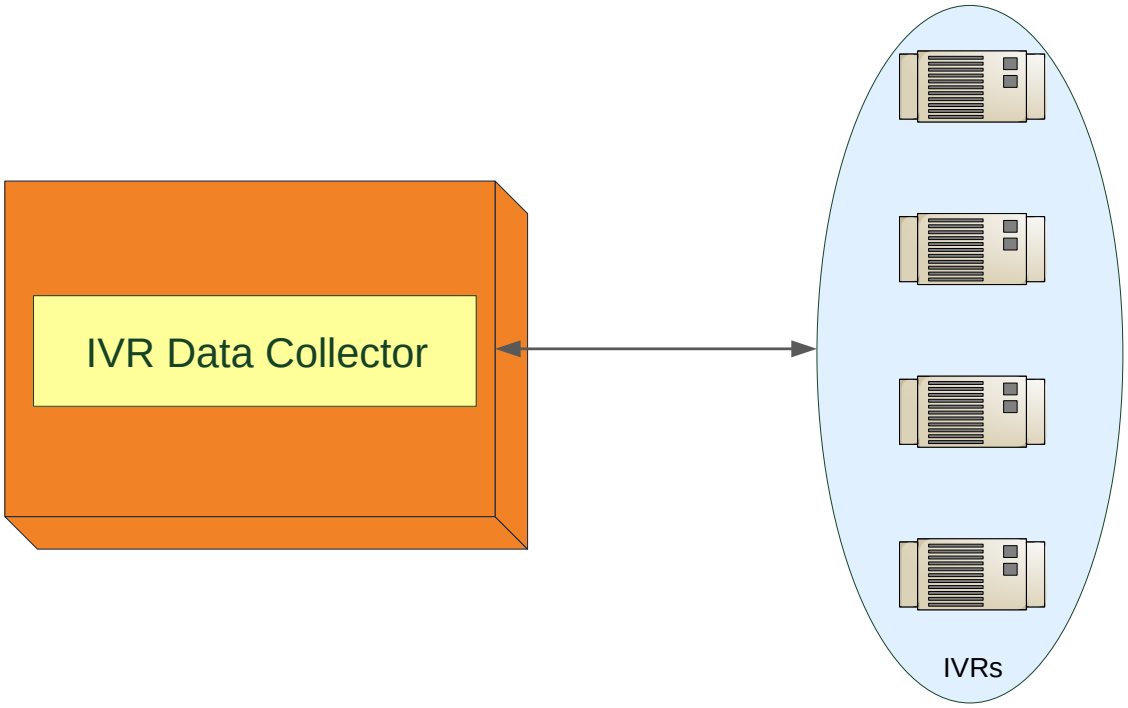

Figura 13 – Módulo IVR Data Collector

3.6.3 Service IVR Analyzer

Esta componente importa os ficheiros gerados com os dados da MIB do IVR, transforma-os e armazena-os em base de dados relacional (Figura 14). De seguida, aplica um algoritmo de *data mining* sobre amostra de dados armazenada, obtendo a classificação da amostragem.

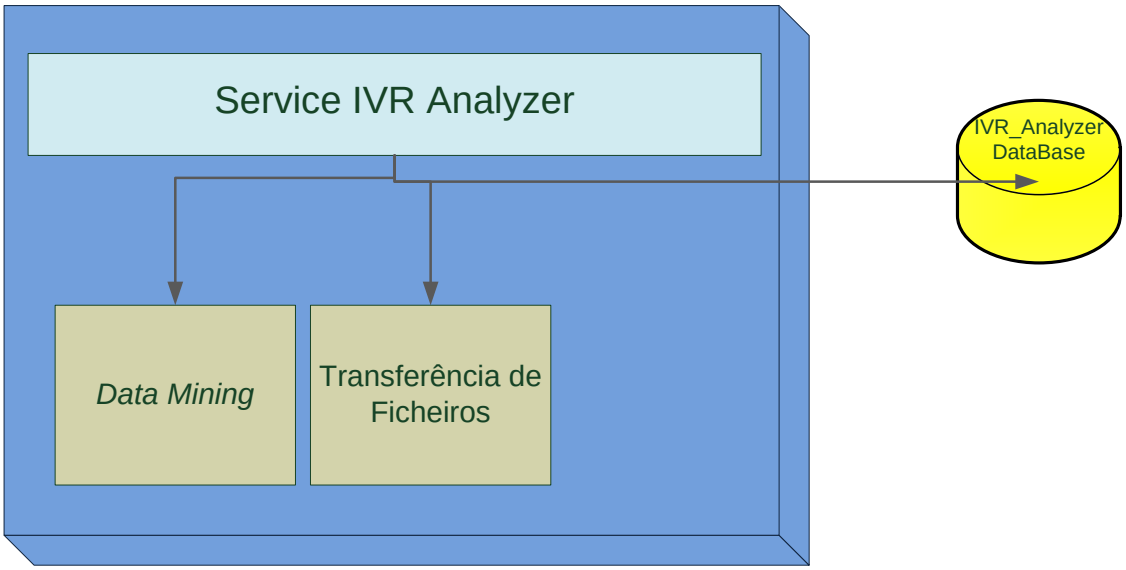

Figura 14 – Módulo Service IVR Analyzer

3.6.4 IVR Analyzer Portal

Aplicação web para disponibilização da informação ao utilizador (Figura 15). A informação é apresentada no formato de gráficos para tornar mais fácil a identificação de padrões, e sequências. A classificação das plataformas é visível juntamente com a análise das recomendações apresentadas.

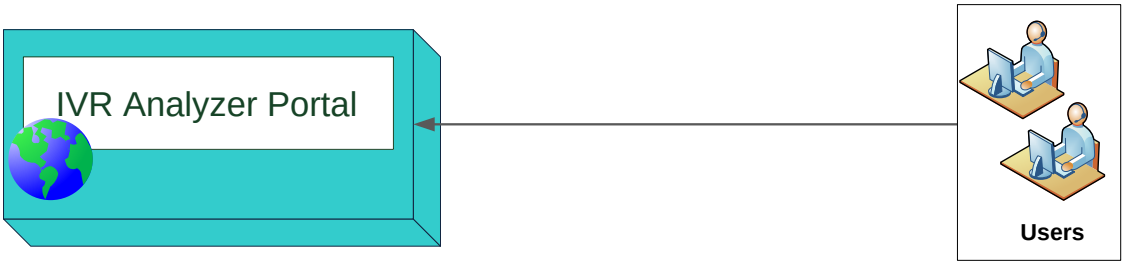

Figura 15 – Módulo IVR Analyzer Portal

4 Implementação

Este capítulo apresenta a solução proposta para a análise de desempenho de IVRs, no qual é detalhada a arquitetura utilizada, bem como os objetivos alcançados, tecnologias e protocolos utilizados.

4.1 Conceção

Ao longo da definição da solução esteve sempre presente a ideia de manter a solução modular. A razão para esta opção prende-se com a performance da aplicação e com a dependência de componentes externos, como o WinSCP[5] e o See5[6]. Com as componentes trabalhadas separadamente agiliza a etapa de desenvolvimento.

Um dos defeitos encontrados nas ferramentas estudadas no estado de arte relacionava-se com a demora no processamento de informação, por isso, todo esse processo foi separado permitindo ao utilizador final (IVR Analyzer Portal) dispor da informação de forma imediata.

Para além disso, este modo de implementação torna mais eficiente a deteção e correção de erros, bem como a extensão da solução.

Para a criação da solução foram tidas em conta as fases do modelo KDD, agrupando algumas fases por componente. Assim, foram definidos três módulos (Figura 16):

- IVR Data Collector (recolha e pré-processamento de dados);

[5] WinSCP - Ferramenta gratuita para exportação e importação de ficheiros entre um servidor local e um remoto

[6] See5 - Ferramenta de *Data Mining* para a criação de árvores de decisão recorrendo ao algoritmo C5.0

- Service IVR Analyzer (transformação dos dados e aplicação do algoritmo de *Data Mining*);
- IVR Analyzer Portal (interpretação e avaliação dos resultados).

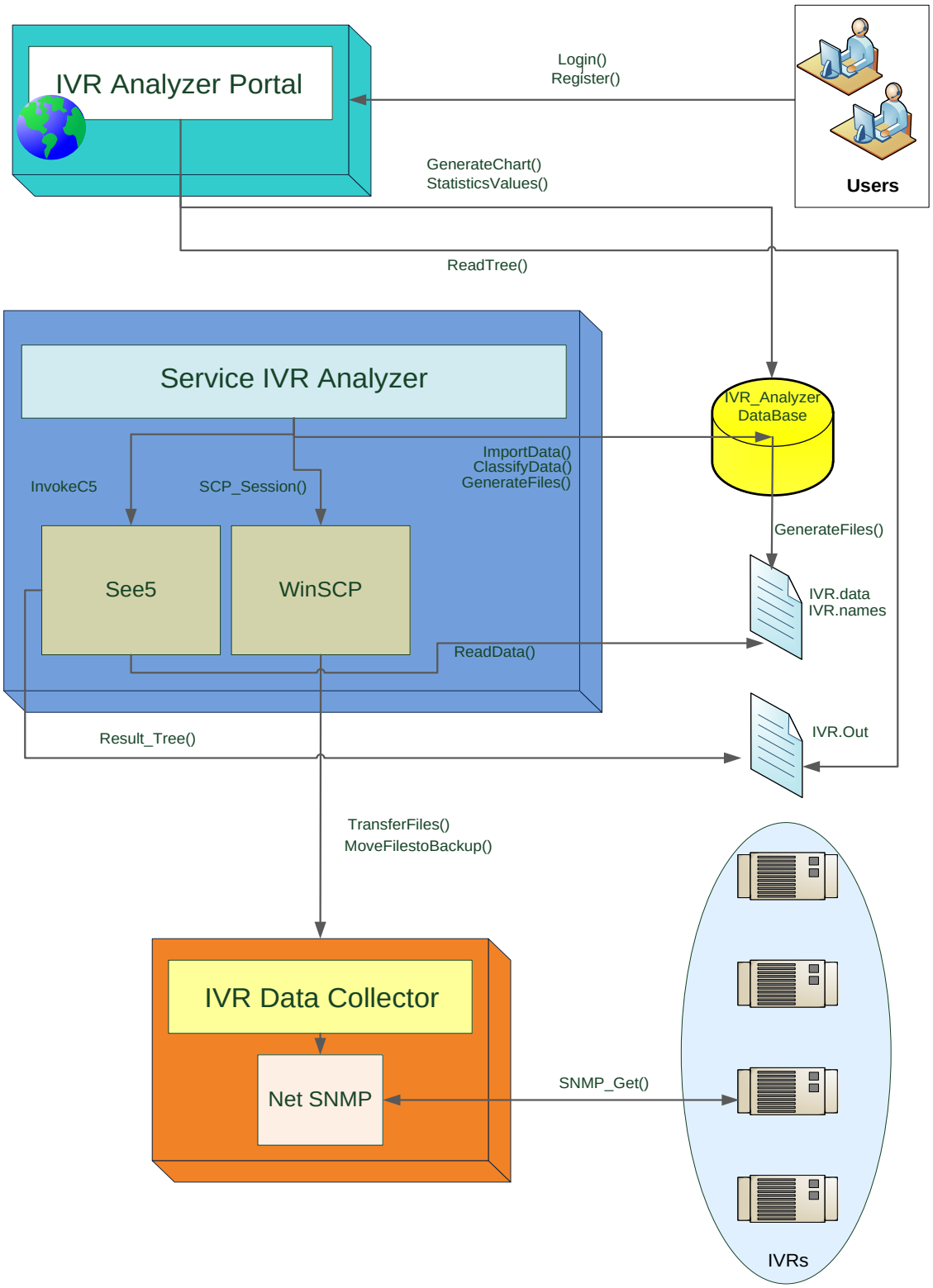

Figura 16 – Arquitetura da Solução IVR Analyzer

No primeiro módulo (IVR Data Collector) é feita a seleção e pré-processamento dos dados, que são depois guardados em ficheiros. No segundo (Service IVR Analyzer) é feita a transformação dos dados e aplicação do algoritmo de *data mining* C5.0. Por último, no IVR Analyzer Portal são disponibilizados os dados recolhidos para ser feita a avaliação do modelo de conhecimento gerado.

A informação recolhida é armazenada numa base de dados relacional, garantindo o histórico da solução.

4.1.1 IVR Data Collector

Este módulo é responsável pela recolha de dados da MIB dos IVR. Para a consulta da informação, foram identificados os OID[7] (*Object Identifier*) dos atributos pretendidos e a leitura da informação é feita através do utilitário snmpget disponibilizado pelo pacote net-snmp-utils, que está integrado neste módulo (Figura 17). A execução do comando com sucesso entre o cliente e servidor só é possível se os dois tiverem a mesma comunidade configurada no agente SNMP. O conceito de comunidade consiste em ambos pertencerem ao mesmo grupo para poderem trocar informação entre si. Pode definir-se mais do que uma comunidade, uma para cada objetivo.

```
snmpget -v 2c -c [Comunidade] [Nome_Servidor/IP] [OID]
```

O resultado retornado da MIB traz um cabeçalho que indica a versão do protocolo SNMP utilizado, o OID invocado e o tipo de resultado, por isso, esta componente também tem a funcionalidade de transformar os dados, retirar o cabeçalho e armazenar em ficheiro o valor. É gerado um ficheiro por dia e por IVR, com o nome definido NomeIVR_Data. A invocação da MIB para recolha de valores está parametrizada no cron.d do servidor para se realizar de cinco em cinco minutos. O cron.d é um programa independente que é executado automaticamente sem ação de um utilizador, executando comandos agendados. Os comandos estão definidos num ou mais ficheiros que se encontram no diretório [/etc/cron.d/].

Os atributos que estão a ser chamados na MIB são:

- *NoCalls* - número de chamadas a decorrer;
- CPU - percentagem de utilização da unidade central de processamento;
- *Memory* - memória que está em uso em KBytes;
- CPS (*Calls per Second*) - número de chamadas por segundo;
- *Airtime* - total de tempo em que os recursos de voz estão ocupados;

[7] OID - identificação de um nó numa estrutura hierárquica

- CCErrors (*Call Control Errors*) - são erros que acontecem do decorrer de uma tentativa falhada de alterar o estado de uma chamada sob o ponto de vista de sinalização;
- *VoiceErrors* - são erros que ocorrem do decorrer de uma tentativa falhada de reproduzir ou gravar um anúncio;
- *ClaimEpError* (*Claim Endpoint Error*) - este erro surge quando o IVR não tem disponíveis canais e ou recursos para efetuar uma chamada de saída;
- AppReject (*Application Reject*) - chamadas rejeitadas pelo serviço;
- NoServicesReg - número de serviços registados no IVR.

Para além destes dados, nos ficheiros são guardados o nome, data e hora de recolha de informação. Cada linha tem a seguinte estrutura:

[Nome_IVR];[Data_Hora];[NoCalls];[CPU];[Memory];[CPS];[Airtime];[CCErrors];[VoiceErrors];[AppReject];[ClaimEpError];[NoServicesReg].

Esta componente é responsável pelas duas primeiras fases do Modelo KDD: seleção e pré-processamento de dados, ou seja, limpeza e tratamento dos dados.

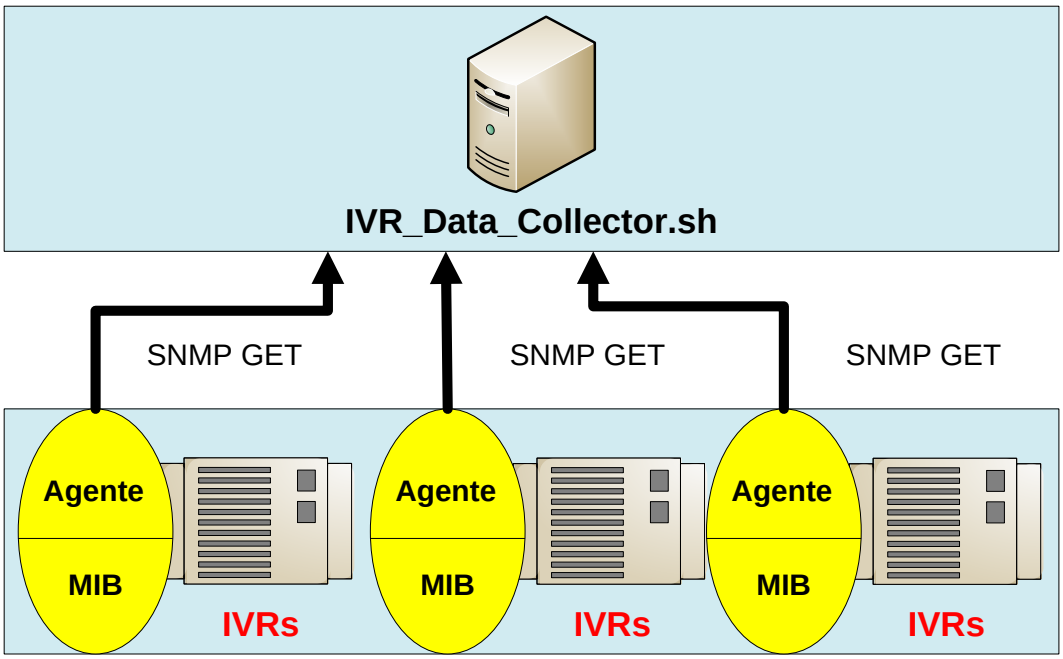

Figura 17 – Arquitetura do módulo IVR_Data_Collector

4.1.2 Service IVR Analyzer

O Service IVR Analyzer recolhe os ficheiros processados pelo IVR Data Collector e importa-os para a base de dados. Para a recolha dos ficheiros estabelece uma sessão SCP, através do IP, nome de utilizador, palavra-chave e SSH *Host Key FingerPrint*. Este último é gerado quando a ligação cliente/servidor é estabelecida pela primeira vez, ambos trocam as suas chaves públicas e através de algoritmos de criptografia que o SSH possui, gera uma chave que apenas pode ser entendida por este par. Assim, são garantidas a autenticidade e integridade dos pacotes trocados.

Os dados antes de serem importados são validados mais uma vez para evitar registos nulos. Neste módulo são executadas as duas fases seguintes do modelo de conhecimento: a transformação e *Data Mining*, ou seja, a descoberta de padrões que permitam inferir novo conhecimento. O algoritmo de mineração de dados utilizado na solução é o modelo de árvores de decisão C5.0, que é do tipo supervisionado. A escolha deste algoritmo foi feita mediante o objetivo principal para o negócio, que era apresentar um modelo de classificação. A estrutura em árvore é das mais simples de interpretar, mesmo por pessoas com pouco conhecimento na área. Atendendo a que os dados não estavam pré-classificados, isto é, não existia nenhum atributo que pudesse ser considerado classe, foi necessário recorrer à criação de *clusters*, de modo a conseguir uma classificação artificial. Assim sendo, foi feito um estudo dos dados para se identificar o número ideal de *clusters* a criar, bem como, a caracterização de cada um para servir de base para a classificação inicial desde modelo.

O estudo começou utilizando o algoritmo k-Means para a criação dos *clusters* e usando um conjunto de treino para a criação dos modelos de árvores de decisão para a identificação das regras e depois avaliou-se cada um dos modelos, usando um outro conjunto de teste, para garantir que a classificação realizada é fidedigna.

Esta análise teve início com a criação de dois *clusters*. Os parâmetros que o algoritmo utilizou para a classificação foram Airtime, CallErrors, ClaimEpError, CPS, CPU, Memory, NoCalls, Voice Errors, AppReject e NoServicesReg, sendo que os dois últimos não foram relevantes para o agrupamento (Figura 18).

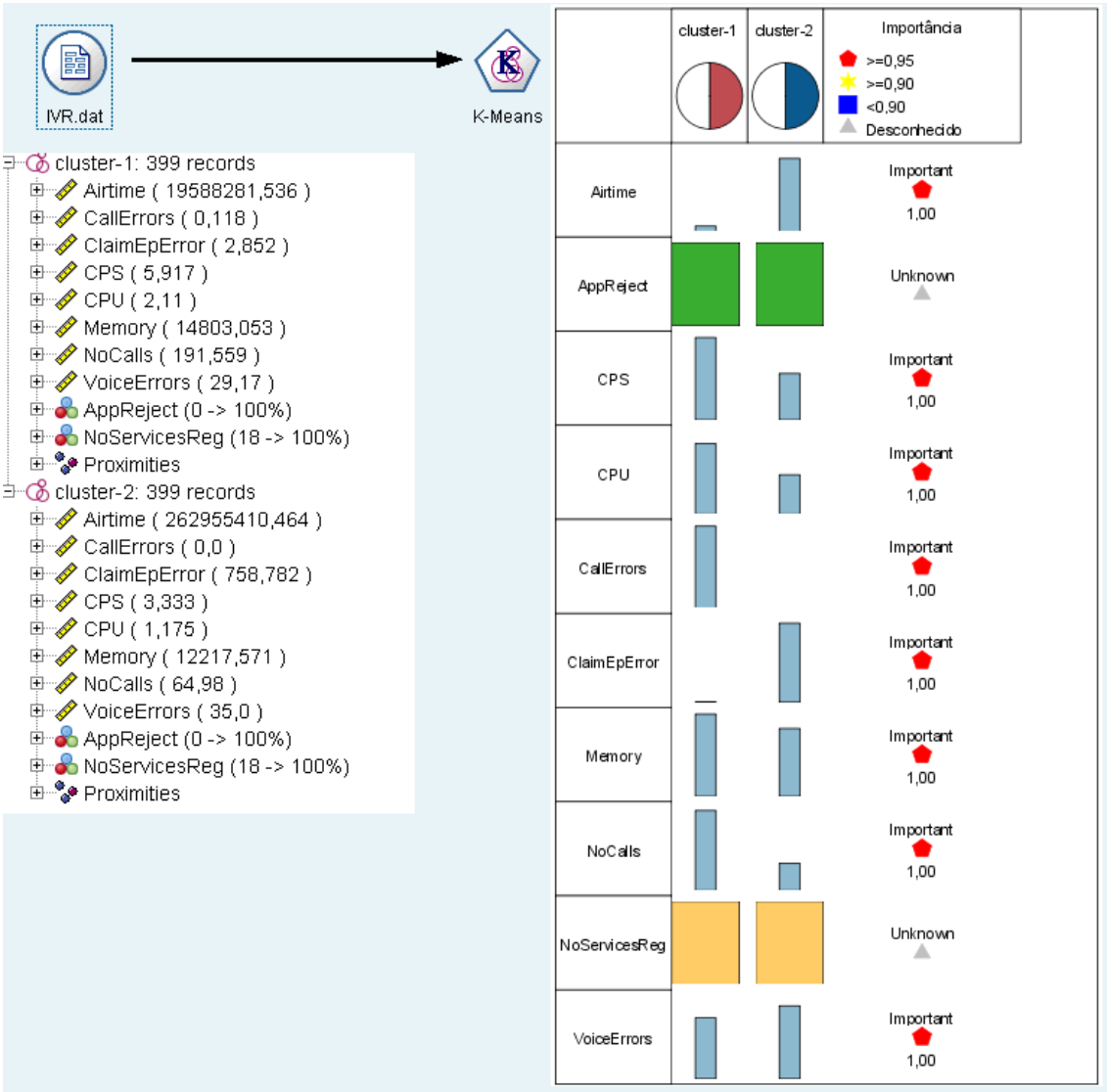

Figura 18 - Criação de Dois Clusters com o K-Means

Posteriormente criou-se um modelo de classificação, recorrendo ao algoritmo C5.0, que fosse capaz de explicar a divisão efetuada em dois *clusters*. Tendo por base o conjunto de dados, selecionou-se uma amostra para treino.

Do modelo obtido, foi possível constatar que a classificação em Cluster 1 ou Cluster 2 é feita exclusivamente com base no atributo Airtime, em função de um determinado limiar (Figura 19). O modelo criado deu origem às seguintes regras:

- Airtime <= 21.252.296 - Cluster 1;
- Airtime > 21.252.296 - Cluster 2.

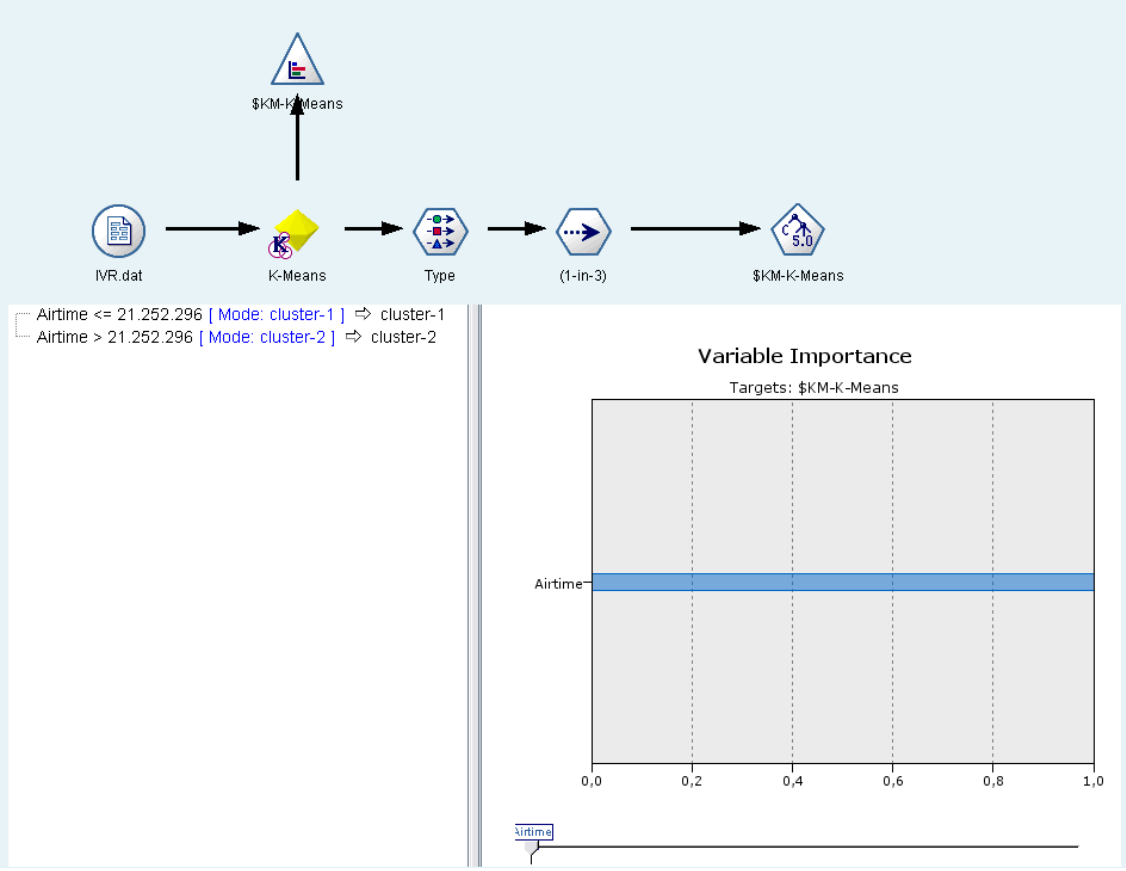

Figura 19 - Criação do Modelo para Dois Clusters

Recorrendo ao conjunto de dados de teste foi feita a avaliação do modelo, tendo-se verificado uma taxa de erro de 0,75% (Figura 20).

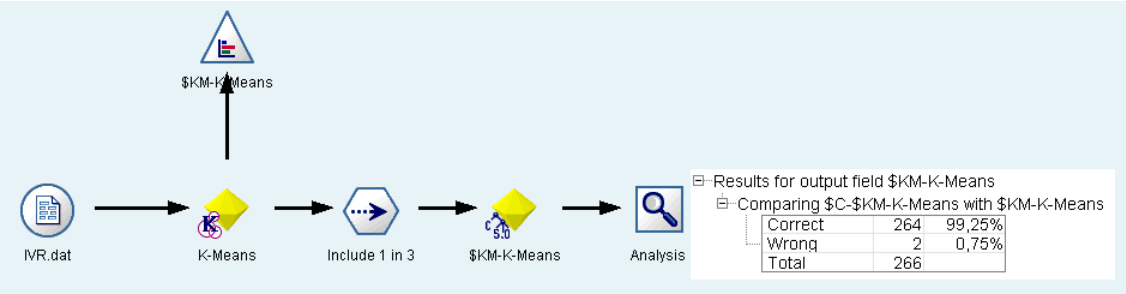

Figura 20 - Avaliação do Modelo Criado para dois *Clusters*

Na segunda fase procedeu-se à criação de três *clusters*. O K-Means para os criar utilizou as variáveis Airtime, CallErrors, ClaimEpError, CPS, CPU, Memory, NoCalls, Voice Errors,

AppReject e NoServicesReg. As duas últimas variáveis não são consideradas importantes para o agrupamento (Figura 21).

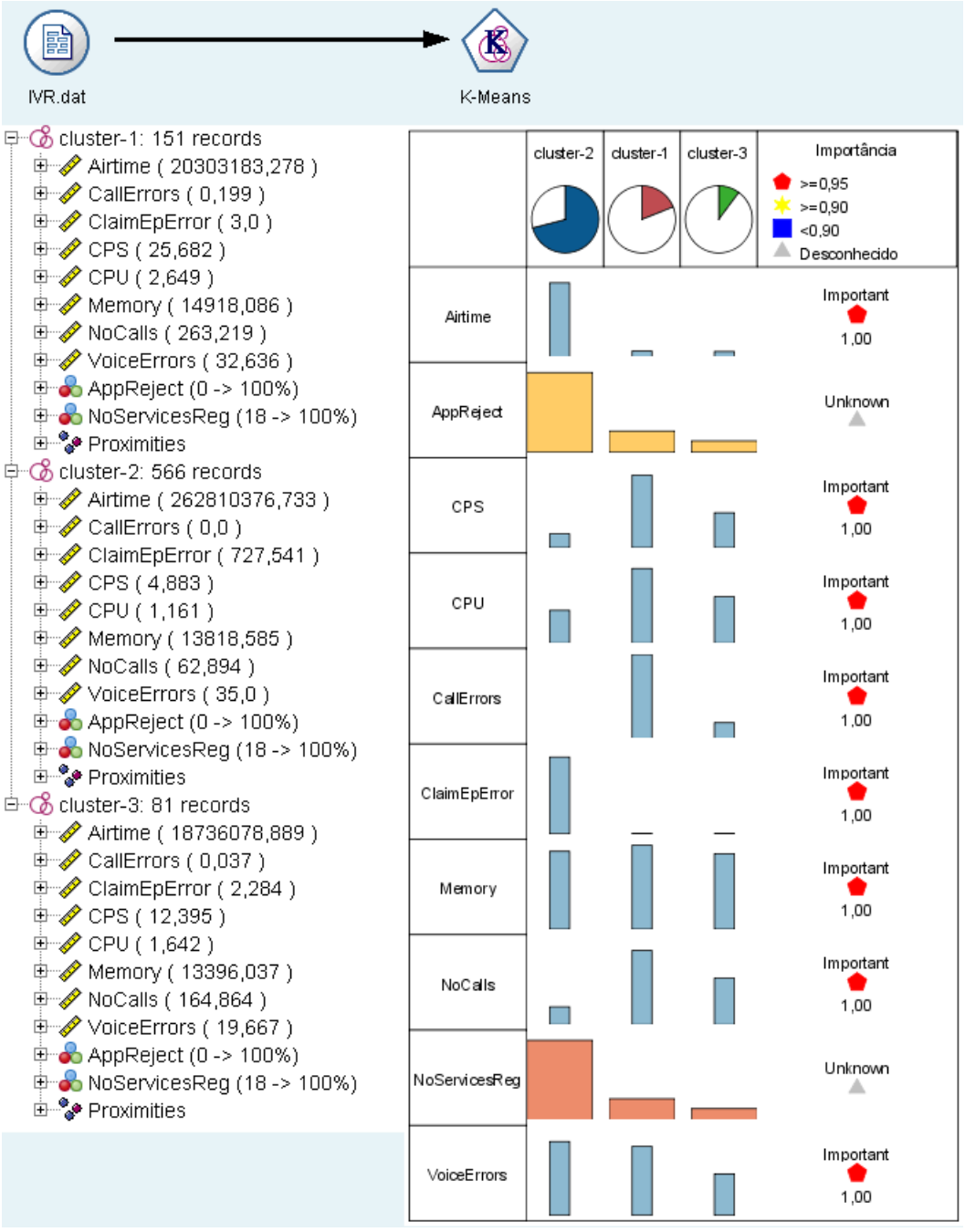

Figura 21 - Criação de Três Clusters com o K-Means

Para gerar o modelo de classificação que explique os *clusters* obtidos foram usadas as variáveis Airtime, VoiceErrors e NoCalls originando as seguintes regras (Figura 22):

- Airtime <= 21.252.296 e VoiceErrors <= 32 e NoCalls <= 238 - Cluster 3;
- Airtime <= 21.252.296 e VoiceErrors <= 32 e NoCalls > 238 e VoiceErrors <= 20 - Cluster 3;
- Airtime <= 21.252.296 e VoiceErrors <= 32 e NoCalls > 238 e VoiceErrors > 20 - Cluster 1;
- Airtime <= 21.252.296 e VoiceErrors > 32 - Cluster 1;
- Airtime > 21.252.296 - Cluster 2.

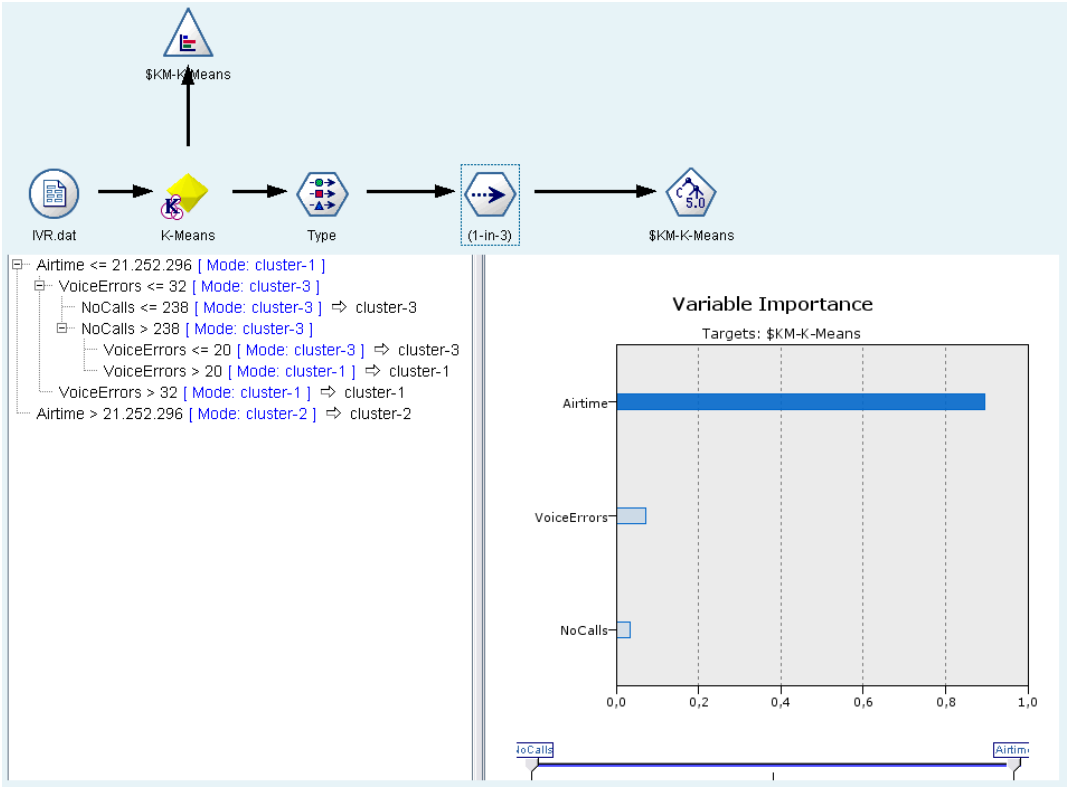

Figura 22 - Criação do Modelo para três Clusters

A avaliação do modelo gerado para os três *clusters* teve uma taxa de erro de 0,38% (Figura 23).

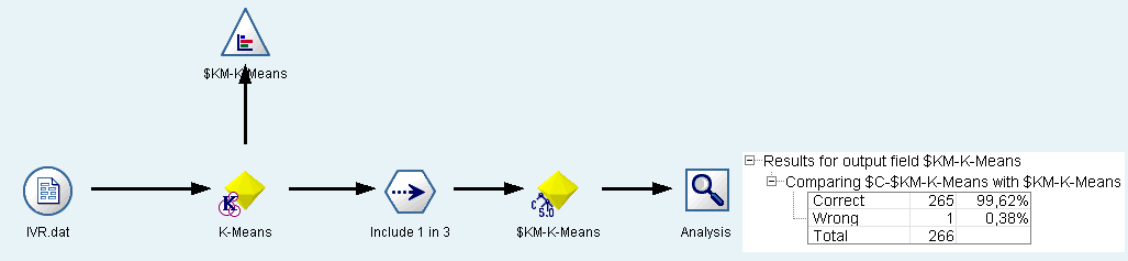

Figura 23 - Avaliação do Modelo criado para três Clusters

Para a criação de quatro *clusters* a partir da amostra de dados o algoritmo k-Means recorreu as seguintes variáveis Airtime, CallErrors, ClaimEpError, CPS, CPU, Memory, NoCalls, Voice Errors, AppReject e NoServicesReg. Porém, as duas últimas variáveis são irrelevantes para a criação de cada um dos grupos (Figura 24).

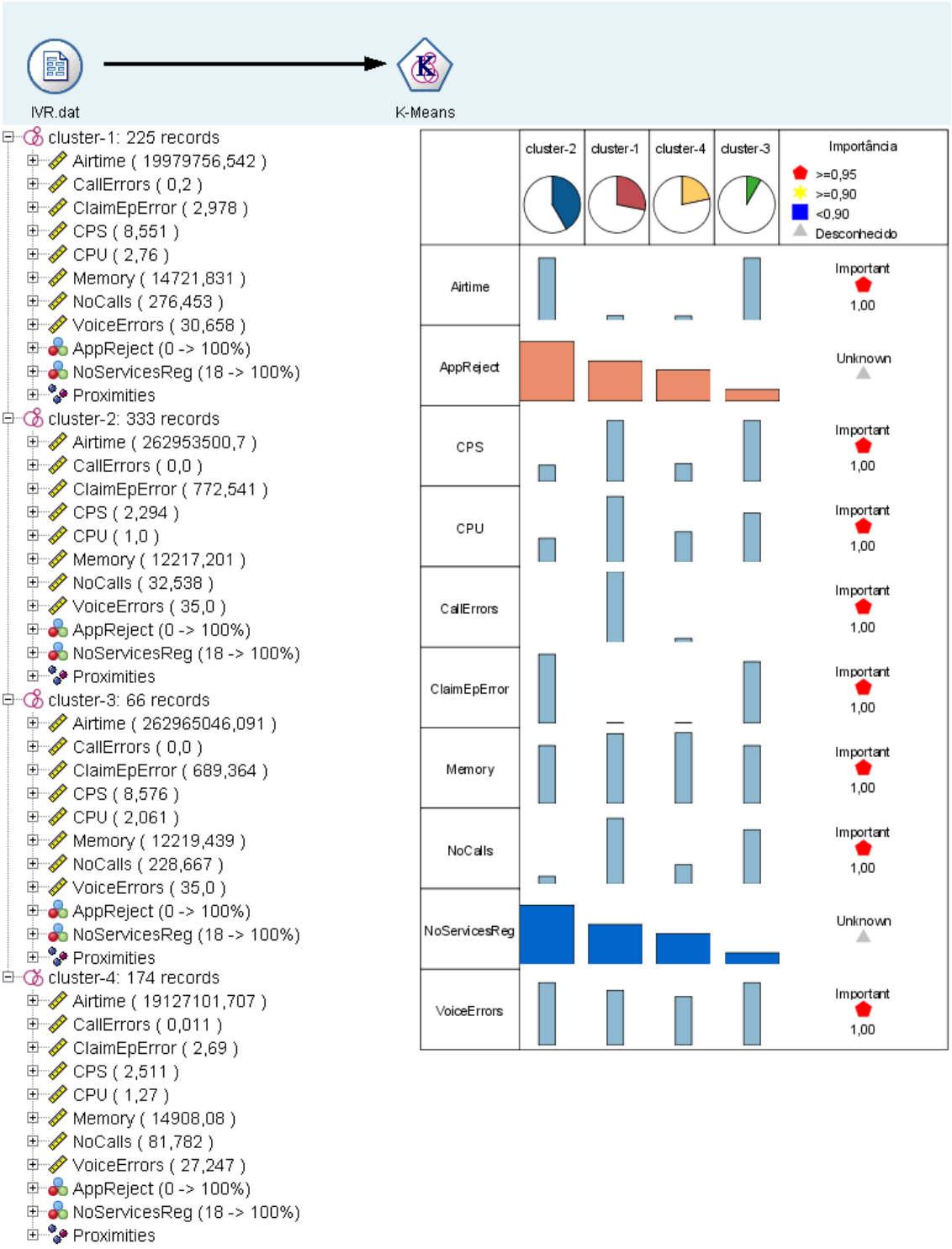

Figura 24 - Criação de Quatro Clusters com o K-Means

O modelo criado a partir dos *clusters* gerados no passo anterior originou as seguintes regras (Figura 25):

- Airtime <= 21.252.296 e CPU <= 1 - Cluster 4;
- Airtime <= 21.252.296 e CPU > 1 e VoiceErrors <= 19 e NoCalls <= 263 - Cluster 4;
- Airtime <= 21.252.296 e CPU > 1 e VoiceErrors <= 19 e NoCalls > 263 - Cluster 1;
- Airtime <= 21.252.296 e CPU > 1 e VoiceErrors > 19 - Cluster 1;
- Airtime > 21.252.296 e CPU <= 1 - Cluster 2;
- Airtime > 21.252.296 e CPU > 1 - Cluster 3.

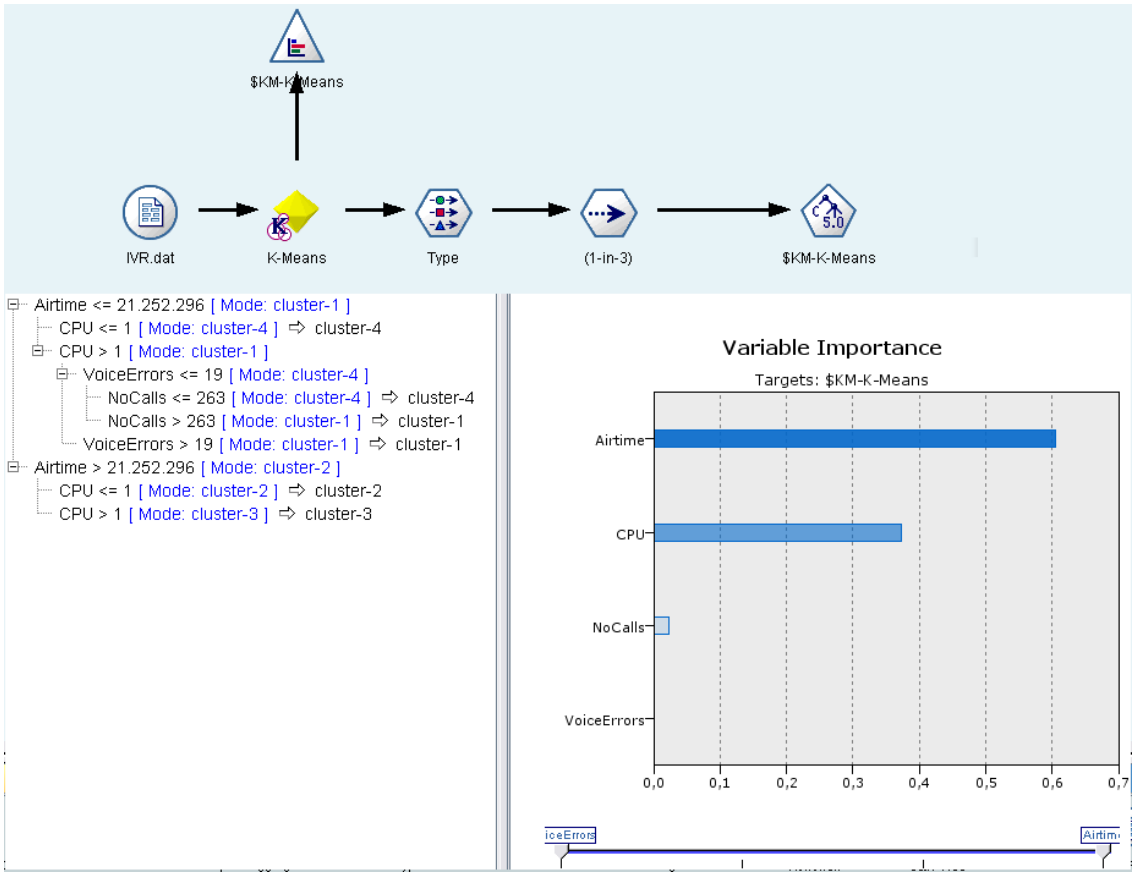

Figura 25 - Criação do Modelo para Quatro *Clusters*

Na avaliação do modelo gerado foi obtida uma taxa de erro de 1,13% (Figura 26).

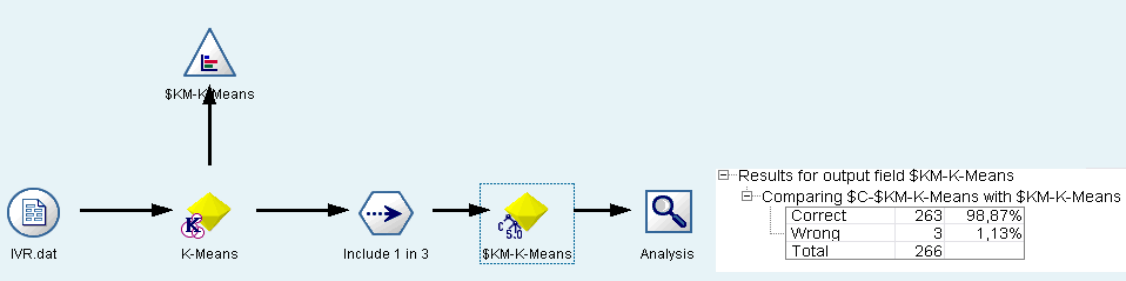

Figura 26 - Avaliação do Modelo Criado para quatro Clusters

Com o estudo realizado verificou-se que o ideal é dividir os dados em três *clusters* pois, dos modelos gerados foi o que obteve menor taxa de erro, embora a diferença seja mínima.

Nos diversos modelos concebidos constatou-se que todos eles usavam a variável Airtime.

Para validar que o sistema teria um comportamento preciso, optou-se por gerar novamente o modelo de classificação sobre os três *clusters* mas, recorrendo ao See5, ferramenta integrada nesta aplicação.

A estrutura em árvore gerada pelo See5 foi similar à gerada pelo Clementine, bem como a taxa de erro do modelo criado, enriquecendo o estudo com mais informação pois, permite ver o número de instâncias classificadas em cada *cluster* (Figura 27).

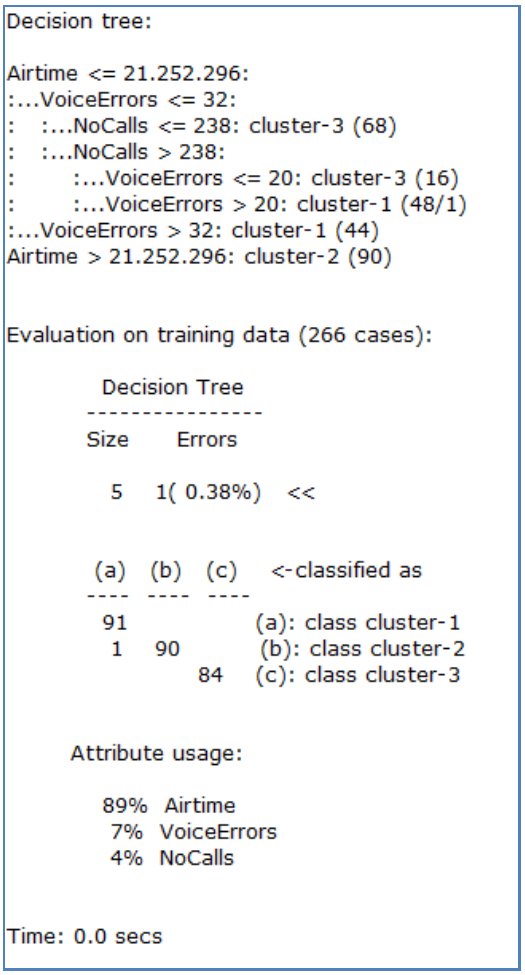

Figura 27 – Modelo de Árvore de Decisão do See5

As regras geradas por este modelo são o ponto de partida para a criação dos modelos de cada um dos IVR registados no IVR Analyzer. Os modelos de criação de árvores de decisão são do tipo supervisionado, ou seja, que precisa de uma classificação inicial.

Analisando as regras geradas e o peso de cada uma das variáveis na classificação dos *clusters*, optou-se por renomear cada um dos *clusters* quanto à sua carga de tráfego de chamadas. Como o Airtime consiste no total de tempo em que os recursos de voz estão ocupados e visto que as regras geradas começaram a ser divididas em Airtime <= 21.252.296 e Airtime > 21.252.296, renomeou-se o Cluster 2 de BestCharge porque as suas instâncias terão maior valor de Airtime. Um número superior de VoiceErrors pode indicar elevado processamento de chamadas pois, são erros que ocorrem do decorrer de uma tentativa falhada de produzir ou gravar a anúncio de áudio, que pode indicar falta de recursos que muitas vezes implica elevada taxa de ocupação. Partindo deste pressuposto e olhando para o número de canais ocupados (NoCalls) o Cluster 1 passa a ser designado como GoodCharge e o Cluster 3 como LowCharge.

Com toda a informação resultante da análise dos dados e das regras geradas no modelo selecionado optou-se por agrupar os dados usando as seguintes regras:

- Airtime <= 21.252.296 e VoiceErrors <= 32 e NoCalls <= 238 - LowCharge;
- Airtime <= 21.252.296 e VoiceErrors <= 32 e NoCalls > 238 e VoiceErrors <= 20 - LowCharge;
- Airtime <= 21.252.296 e VoiceErrors <= 32 e NoCalls > 238 e VoiceErrors > 20 - GoodCharge;
- Airtime <= 21.252.296 e VoiceErrors > 32 - LowCharge;
- Airtime > 21.252.296 - BestCharge.

O módulo Service IVR Analyzer é instalado como um serviço e com agendamento para ser executada uma vez por dia, realizando todas as tarefas que tem definidas sem ser necessário qualquer interferência por parte do utilizador.

4.1.3 IVR Analyzer Portal

O IVR Analyzer Portal foi criado para o utilizador final verificar o estado dos servidores, gerar relatórios que podem ser exportados permitindo a sua manipulação para apresentação de relatórios de desempenho. Esta componente foi criada tendo em conta os relatórios de informação construídos manualmente para análise do estado das soluções nos clientes. Para aceder ao portal é necessário que o utilizador esteja registado no sistema.

No portal é possível consultar o estado de um dado componente do IVR, num período de tempo definido. A informação é retornada em gráficos para tornar a sua análise mais expedita, indicando o máximo e mínimos atingidos e a média (Figura 28). Os dados e gráficos podem ser exportados para Excel para permitir ao utilizador manipular a informação e gerar outro tipo de análises com os dados disponíveis.

Todos os IVR registados no IVR Analyzer, bem como, todas a componentes dos servidores a serem monitorizadas são disponibilizadas no portal para serem selecionadas pelo utilizador. Os intervalos de tempo podem ser inseridos ou selecionados através do calendário (*Select date and Time*). A opção de exportar gera um ficheiro com os dados, coluna A- Data e Hora coluna B - Componente e o gráfico criado com esses dados.

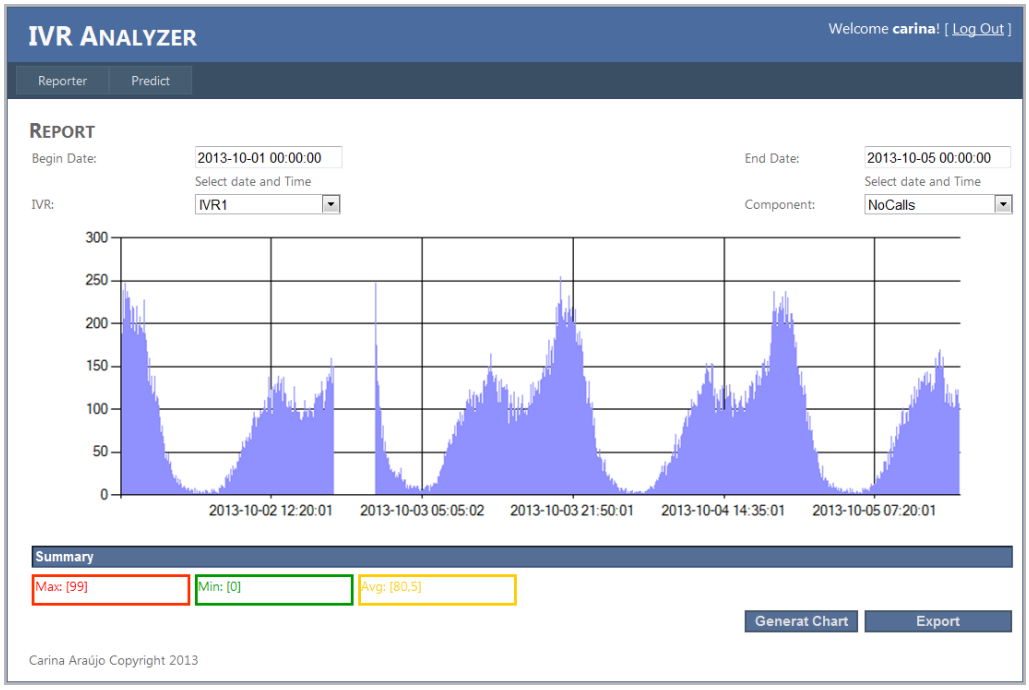

Figura 28 – IVR Analyzer Portal - Área Reporter

Para além da informação da situação atual, permite a classificação das plataformas IVR quanto ao seu estado, recomendando tomadas de decisão na área de negócio.

As recomendações são feitas tendo em conta a classificação do IVR e as componentes físicas da plataforma:

- Se for classificado como *Low Charge* o sistema sugere voltar a atenção para o utilizador final, ou seja, quem utiliza os serviços telefónicos para se criar um novo serviço que aumente o número de chamadas na plataforma;
- Caso seja classificado como *Good Charge* e o CPU esteja muitas vezes acima de 75% de taxa de ocupação, é sugerido a aquisição de um novo servidor para balanceamento de volume de chamadas;
- Se a classificação é *Good Charge* e a taxa de utilização de memória é superior a 75%, é aconselhada a verificação de *slots* livres de memória para adquirir mais para o servidor;
- Se a classificação é *Good Charge* o sistema sugere a criação de serviços para aumentar tráfego do IVR para transitar para a classe *Best Charge*;
- Sendo classificado como *Best Charge*, é sugerido o aumento de licenciamento de recursos telefónicos;
- Sendo pertencente à classe *Best Charge* e com uma taxa de ocupação de CPU superior a 75%, é alertado para a aquisição de um novo servidor e de aumento de licenciamento de recursos telefónicos para balanceamento de carga;
- Sendo caracterizado como *Best Charge* e com uma taxa de utilização de memória acima de 75%, o utilizador é aconselhado a validar o número de *slots* de memória disponíveis, para um possível aumento de memória e de licenciamento de recursos telefónicos.

O modelo de classificação é apresentado na estrutura de árvore e pode ser exportado para um editor de texto (Figura 29).

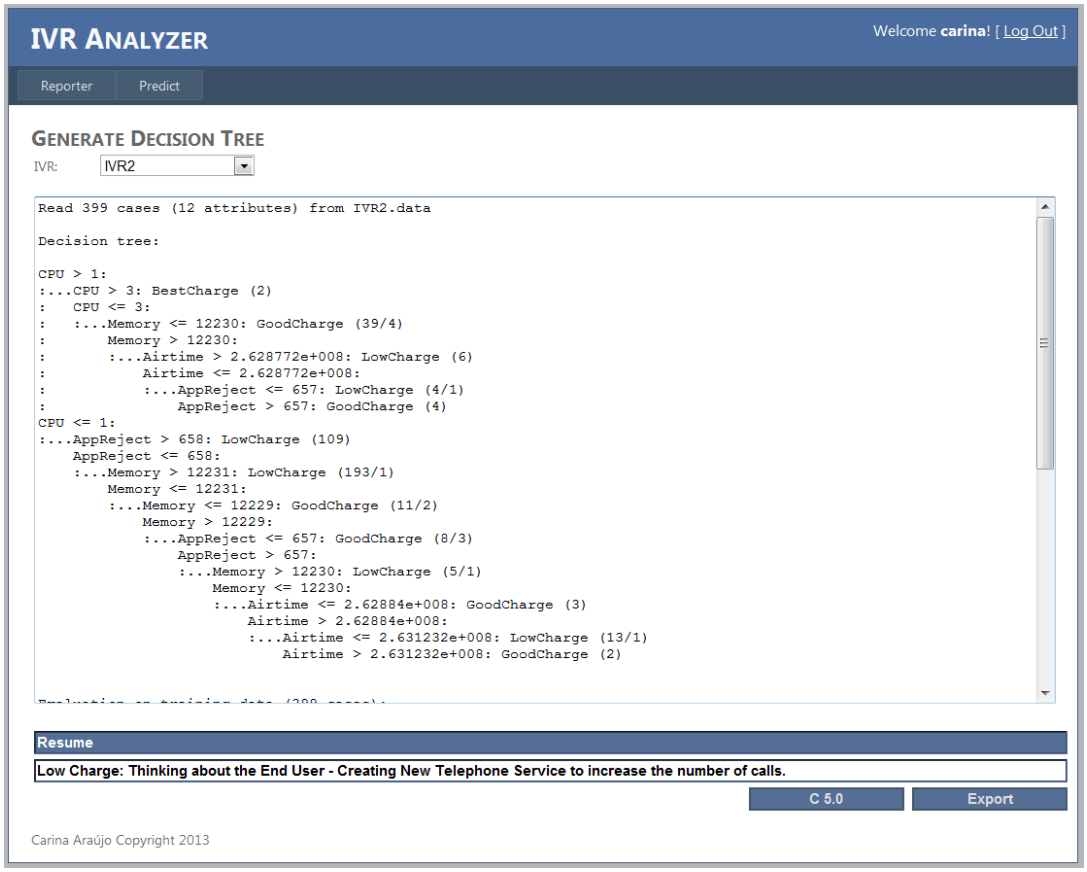

Figura 29 – IVR Analyzer Portal - Área Predict

4.2 Tecnologias e Componentes

Neste capítulo são descritas componentes, protocolos e linguagens de programação utilizadas para a construção do protótipo e a razão da sua escolha.

4.2.1 WinSCP.dll

O WinSCP.dll é uma biblioteca de ligação dinâmica gratuita, que permite a criação de sessões remotas através da utilização dos protocolos FTP, SFTP e SCP.

Possibilita operações como:

- *Download* de ficheiros;
- *Upload* de Ficheiros;
- Sincronização de diretórios - verificação de diferenças entre um diretório local e um remoto, ou vice-versa, em que os ficheiros novos da pasta de origem são transferidos para o outro.

Esta componente pode ser utilizada em aplicações desenvolvidas em C#, VB.Net, JScript, VBScript, VBA, Perl e Python [SNMP, 2013].

A decisão de integração do WinSCP com um dos módulos, o Service IVR Analyzer deveu-se à necessidade de estabelecer ligações para transferência de ficheiros remotos, de servidores Linux e Windows para um servidor Windows.

4.2.2 Protocolo para Transferência de Ficheiros

Para a transferência de ficheiros de um servidor remoto existem os protocolos FTP, SFTP e SCP.

O primeiro, *File Transfer Protocol*, é um protocolo de rede para arquiteturas cliente-servidor que atua sob a camada TCP. Não é de utilização segura pois, todos os dados dos pacotes circulam às claras na rede [Mitchell, 2013a]. Com uma aplicação de captura de pacotes na rede, como o WireShark, facilmente são obtidos os dados de autenticação utilizados.

O *Secure File Transfer Protocol* possibilita a transferência de ficheiros de uma forma segura, usando o SSH [Mitchell, 2013b]. É um protocolo de rede que utiliza criptografia para o envio e receção de pacotes. Ao contrário do anterior, encripta a informação transferida.

O *Secure Copy Protocol* tal como o anterior, recorre ao SSH, usando o mesmo sistema de autenticação, garantindo a autenticidade e confidencialidade dos ficheiros transferidos [Thomas, A., 2012].

Para a cópia dos ficheiros de dados gerados pelo módulo Data Collector teve de proceder-se à escolha do protocolo a utilizar. Descartou-se a primeira opção devido à falta de segurança. A decisão entre os restantes recaiu sobre o SCP porque permite a utilização de *wildcards* (*) nos comandos, criando a facilidade de cópia ou envio de vários ficheiros numa ligação.

4.2.3 Protocolo SNMP

SNMP é um protocolo para troca de informação de gestão de terminais de uma rede. Atua na camada de aplicação do Modelo OSI [SNMP, 2013]. A utilização deste protocolo possibilita a comunicação do IVR Data Collector com os agentes SNMP presentes nos servidores IVR, para recolha de informação da MIB.

É utilizado para monitorização de sistemas de rede, como demonstram as ferramentas referidas no estado de arte.

O SNMP é composto pelas seguintes componentes:

- *SNMP Agent* - é um utilitário instalado em servidores que quando ativo recolhe dados das suas componentes, atualiza e retorna dados da sua base de dados - MIB, enviando informação para *o SNMP Manager*. Podem ser agentes *standard* como Net-SNMP ou específico do fornecedor;
- *SNMP Manager* - é um sistema de monitorização, que tem como função comunicar com os agentes para saber o estado de cada um dos terminais identificados na rede;
- *Managed Devices* - dispositivos presentes na rede que estão a ser alvo de monitorização;
- *Management Information Base* - é uma base de dados que guarda informação de estatísticas e estado do sistema, que é mantida pelo agente e serve para retorno do ponto de situação do terminal para o SNMP Manager. Cada dado guardado tem um nome ou identificador denominado OID [Cisco, 2012].

A troca de informação é feita usando os seguintes comandos:

- GET - pedido de informação ao agente, retornando um ou mais valores;
- GET NEXT - semelhante ao anterior, diferindo em que devolve o valor do OID seguinte na MIB;
- GET BULK - para pedidos de grande volume de informação;
- SET - usado para alterar valores na MIB;

- TRAPS - é utilizado pelo agente para o envio de informação e ocorrência de eventos para o *SNMP Manager*;
- INFORM - semelhante ao anterior mas, tem confirmação de receção de informação;
- RESPONSE - comando utilizado pelo SNMP Manager para solicitação de dados [OpManager, 2013].

Este protocolo vai permitir a comunicação entre o módulo IVR Data Collector e os agentes SNMP dos servidores IVR, a fim de recolher os dados pretendidos das suas MIB.

4.2.4 Shell Script

Para que a recolha de dados do IVR se processe de uma forma rápida, definiu-se a utilização de um servidor dedicado para esta tarefa. O servidor tem como pré-requisito uma instalação básica de sistema operativo e sem ambiente gráfico, para minimizar a sua carga de processamento, otimizando a sua performance, para tornar a sua função o mais eficiente possível. Aproveitando os recursos que o sistema operativo disponibiliza para o desenvolvimento do módulo de recolha de informação, optou-se por *Shell Script*.

Shell script é uma linguagem de programação utilizada em vários sistemas operativos [Gite, 2002]. Está presente no dia-a-dia na execução de comandos de rotina no computador, sem ninguém se aperceber ou pensar nisso. A sua sintaxe depende do interpretador usado.

Neste caso de estudo, o interpretador é o Bash disponibilizado pelo sistema operativo Red Hat.

4.2.5 C

Para a cópia dos ficheiros gerados com a recolha dos dados das MIB dos IVR para um servidor, importação dos dados para base de dados e a criação de uma árvore de decisão com uma amostra da informação coletada foi definido a criação de uma componente do tipo *Windows Service*. Uma aplicação que esteja em execução, por trás do sistema operativo sem precisar da interação do utilizador.

A decisão para este tipo de implementação foi de encontro à possibilidade de ter dois modos de funcionamento, servidor ou cliente.

Este módulo quando instalado num cliente pode permitir a recolha de informação para o seu computador, possibilitando a sua consulta e manipulação sem ter de aceder ao portal web que a disponibiliza. Como a maioria dos utilizadores utiliza o sistema operativo Windows, torna-se mais fácil a instalação e gestão de um serviço deste tipo.

Para o desenvolvimento deste módulo foi escolhida a linguagem de programação orientada a objetos C#, derivada do C e C++ e suportada pela Framework.Net da Microsoft [Microsoft, 2012]. É de estrutura simples o que facilita o seu entendimento, permitindo agilizar a deteção e correção de erros.

A utilização desta linguagem traz muitas vantagens, entre as quais [Microsoft, 2012]:

- Possibilita o controlo de versões, cada *assembly* gerado, exe ou dll, tem informação da versão do código;
- Facilidade de integração e utilização de componentes do sistema operativo;
- A gestão de memória das aplicações é feita em tempo de execução através do *Garbage Collector*, permitindo a recuperação de área de memória inutilizada, evitando problemas de esgotamento de recursos de memória.

4.2.6 See 5

É uma ferramenta de *Data Mining* para a criação de árvores de decisão recorrendo ao algoritmo C5.0. Permite a construção de regras para a classificação que melhor definam o negócio, tendo capacidade para a análise de base de dados com milhões de registos.

Possui três versões do sistema, uma demo gratuita mas, com limitação de número de registos de entrada, uma para Windows comercial e uma para Linux de licenciamento GPL.

Para a execução desta aplicação são precisos pelo menos dois ficheiros de entrada. Um de extensão data com a amostra de dados a usar e um com a extensão names com a descrição dos atributos.

Os atributos podem ser definidos nos seguintes tipos [See5, 2013]:

- [ignore] - exclusão do atributo, ou seja, não é usado como atributo de entrada para a construção da árvore de decisão;
- [continuous] - valor numérico;
- [date] - datas nos formatos YYYY/MM/DD ou YYYY-MM-DD;
- [time] - tempo nos formatos HH:MM:SS;
- [timestamp] - data e tempo nos formatos YYYY/MM/DD HH:MM:SS ou YYYY-MM-DD HH:MM:SS;
- [discrete] - os atributos deste tipo são preenchidos pelos próprios dados;
- [label] - este atributo é um identificador, por isso, é ignorado pelos classificadores;
- Podem ser definidos atributos recorrendo a fórmulas (age:=year_now - year_birth).

Para além destes ficheiros que são de carácter obrigatório para a aplicação, podem ser criados ficheiros do tipo *test*, onde são definidos casos não detetados para testar o classificador, *cases*

para testar os casos à posteriori do classificador e *costs* definição dos custos de classificação incorreta.

Esta ferramenta tem disponíveis diversas técnicas para otimização do processo e minimização da taxa de erro [See5, 2013].

No *boosting* são construídos diversos classificadores, para a mesma amostra de dados gerando cada um, uma previsão. Entre estas previsões, a que surgir mais vezes é a que é encarada como resultado final.

O *winnowing attributes* é utilizado quando o número de atributos é muito grande, reduzindo o número de entradas através da seleção de um subconjunto.

A técnica de *pruning* é aplicada para a remoção dos ramos em que a previsão teve maior taxa de erro.

No *softening thresholds* é introduzido um valor inicial que é dividido em três partes, menor, maior e central. Se o valor de um dado atributo é menor ou maior, a classificação prossegue estabelecendo comparações do tipo maior ou igual (>=) ou menor (<). Caso o valor não se enquadre no quadro anterior de maior ou igual ou menor, é feita uma combinação probabilística dos resultados da árvore.

Para medir a precisão de resultados esta ferramenta tem a funcionalidade *cross-validation trials* que consiste em dividir os dados em blocos com o mesmo tamanho e número de classes e gerar um classificador com cada um. Posteriormente é executado cada classificador com dados de entrada que correspondem aos não utilizados na criação dele próprio. A precisão é estimada através da razão entre o número total de erros e o número total de casos.

Nalgumas áreas de negócio a taxa de erro nos resultados pode acarretar graves problemas na tomada de decisão. Esta aplicação permite atribuir custos a cada um dos resultados a prever de acordo com a importância que têm para o negócio. No final, é gerada uma árvore de decisão com o custo de erro na classificação. A funcionalidade que possibilita esta ação denomina-se *differential misclassification costs.*

Para um melhor apoio à tomada de decisão podem ser atribuídos pesos para definição de grau de importância para o negócio a cada um dos resultados. No final o resultado gerado tem o custo relativo para cada classe retornada.

Por questões de performance o See5 permite a criação de classificadores retirando uma amostra de dados aleatória dos registos da base de dados, utilizando os restantes dados para a construção da árvore.

Esta ferramenta dá resposta aos objetivos do projeto e tem funcionalidades que servirão como trabalho futuro para enaltecer a solução, garantindo a sua continuidade.

5 Testes e Resultados

Neste capítulo são apresentados os testes realizados a cada um dos módulos. Os resultados obtidos pela solução são analisados e descritos. Para testar e avaliar a solução foram configurados dois IVR que estão ativos, ou seja, a processar chamadas a fim de obter dados reais.

5.1 Módulo Data Collector

Os testes realizados a este módulo são bastante objetivos. A sua função é recolher informação das MIB dos servidores IVR nele configurados, limpar os cabeçalhos gerados pelo protocolo SNMP e colocar a informação num ficheiro por servidor.

Cenário de Testes:

- Aceder ao servidor onde o módulo está instalado;

- Verificar se a informação está a ser recolhida de 5 em 5 minutos;

- Aceder aos ficheiros gerados e validar a informação;

- Confirmar que é criado um ficheiro por dia por servidor.

Os resultados obtidos comprovaram que este módulo funciona com rapidez e eficácia, não gerando atrasos na recolha de informação.

5.2 Service IVR Analyzer

Os testes a esta componente foram os mais demorados pois, esta é o agregador da solução, o núcleo.

Cenário de testes:

- Validar que é estabelecida uma sessão SCP com o servidor que tem o Data Collector instalado para transferência de ficheiros;

- Verificar que os ficheiros são descarregados e aceder ao outro servidor para verificar que os ficheiros copiados foram movidos para a pasta de *backup*;

- Analisar os dados importados para a base de dados, existência de registos válidos e não nulos;

- Traçagem dos valores registados para validar se a classificação das instâncias está de acordo com o modelo gerado para a divisão em três *clusters*;

- Averiguar que são gerados os ficheiros ivr.data (ficheiro com os registos e a classificação inicial) e ivr.names (ficheiro onde são enumeradas as variáveis e o seu tipo) corretamente (Figura 30);

- Validar se a ferramenta invoca o See5 sem retornar erro e validar que a árvore de decisão gerada tem uma percentagem de erro baixa (ficheiro ivr.out).

```
LowCharge, GoodCharge, BestCharge.	| attribute containing class to be predicted

HostName:	ignore.
DateTime:	ignore.
NoCalls: continuous.
CPU: continuous.
Memory: continuous.
CPS: continuous.
Airtime: continuous.
CCErrors: continuous.
VoiceErrors: continuous.
AppReject: continuous.
ClaimEpError: continuous.
NoServicesReg: continuous.
```

Figura 30 - Ficheiro ivr.names

Os testes realizados a este módulo fizeram com que o processo de transformação dos dados fosse retificado em algumas situações pois, as árvores de decisão estavam a retornar percentagens de erro elevadas.

Na fase final os resultados obtidos estavam de acordo com o esperado, comprovando o seu bom desempenho.

5.3 IVR Analyzer Portal

Esta componente apresenta os resultados das duas anteriores. Assim sendo, os testes realizados consistiram em validar que a informação era apresentada como o desejado, que a ferramenta cumpre os requisitos de usabilidade e ergonomia especificados e verificar para cada perfil de utilizador que acessos possuem, ou seja, quais as funções disponibilizadas.

5.4 IVR1

Na análise da árvore retornada para a amostra de dados do IVR1, verificou-se que a árvore tem 10 folhas e que 36 dos dados foram mal classificados, dando uma taxa de erro de 9,0% (Figura 31). O See5 retorna a árvore final pois, o processo de conjunto de treino e teste é automaticamente desempenhado pelo sistema, dando origem no final à árvore gerada e respetiva avaliação do modelo.

Dos dados mal classificados:

- 9 da classe *Low Charge*, foram classificados como *Best Charge*;
- 10 da classe *Good Charge* classificados como *Best Charge*;
- 17 da classe *Best Charge* classificados como *Low Charge*.

Dos atributos utilizados os que contribuíram para a classificação foram: o CPU, Airtime, Memory, VoiceErrors, ordenados do maior para o menor, de 100% de impacto para 5%.

A transformação da árvore de decisão em regras resulta no seguinte:

- Se CPU <=1 - *Low Charge*;
- Se CPU >2 - *Best Charge*;
- Se CPU <=2 e *Airtime* <=2.584694e+008 - *Low Charge*;
- Se CPU <=2 e *Airtime* > 2.584694e+008 e *Airtime* <=2.59102e+008 - *Best Charge*;
- Se CPU<=2 e *Airtime* >2.584694e+008 e *Memory* >12885 - *Low Charge*;
- Se CPU<=2 e *Airtime* > 2.584694e+008 e *Memory* <= 12885 e *VoiceErrors* > 0 - *Best Charge*;
- Se CPU<=2 e *Airtime* > 2.584694e+008 e *Memory* <= 12885 e *VoiceErrors* <=0 e *Airtime*<=2.595353e+008 - *Low Charge*;
- Se CPU<=2 e *Memory* <= 12885 e *VoiceErrors* <=0 e *Airtime* > 2.595907e+008 - *Best Charge*;
- Se CPU<=2 e *Memory* <= 12885 e *VoiceErrors* <=0 e *Airtime*>2.595353e+008 - *Low Charge*.

No final de cada ramo, ou seja, na folha aparece o valor do número de registos assim classificados. Quando surge na forma de fração, o primeiro valor reflete o número de registos assim classificados, o segundo os mal classificados daquela classe.

Na avaliação dos resultados concluiu-se que a maior parte das instâncias de entrada foram classificadas como Low Charge com uma taxa de sucesso superior a 90%.

```
Read 399 cases (12 attributes) from IVR1.data

Decision tree:

CPU <= 1: LowCharge (195)
CPU > 1:
:...CPU > 2: BestCharge (33/10)
    CPU <= 2:
    :...Airtime > 2.596589e+008: LowCharge (19)
        Airtime <= 2.596589e+008:
        :...Airtime <= 2.584694e+008: LowCharge (76/15)
            Airtime > 2.584694e+008:
            :...Airtime <= 2.59102e+008: BestCharge (39/7)
                Airtime > 2.59102e+008:
                :...Memory > 12885: LowCharge (16)
                    Memory <= 12885:
                    :...VoiceErrors > 0: BestCharge (7/2)
                        VoiceErrors <= 0:
                        :...Airtime <= 2.595353e+008: LowCharge (3)
                            Airtime > 2.595353e+008:
                            :...Airtime <= 2.595907e+008: BestCharge (4)
                                Airtime > 2.595907e+008: LowCharge (7/2)


Evaluation on training data (399 cases):

            Decision Tree
          ----------------
          Size      Errors

            10   36( 9.0%)   <<


           (a)   (b)   (c)    <-classified as
          ----  ----  ----
           299           9    (a): class LowCharge
                        10    (b): class GoodCharge
            17          64    (c): class BestCharge


        Attribute usage:

            100%  CPU
             43%  Airtime
              9%  Memory
              5%  VoiceErrors


Time: 0.0 secs
```

Figura 31 – Árvore de Decisão do IVR1

O tráfego neste IVR aumentou significativamente o que alterou a árvore retornada no espaço de um mês, ou seja, o número de chamadas processadas aumentou consideravelmente. Esta árvore tem 11 folhas e 27 dos registos foram mal classificados, dando uma taxa de erro de 6,8% (Figura 32).

Dos dados mal classificados:

- 7 da classe *Low Charge*, foram classificados como *Good Charge*;
- 2 da classe *Good Charge*, foram classificados como *Best Charge*;
- 7 da classe *Good Charge*, foram classificados como *Best Charge*;
- 11 da Classe *Best Charge*, foram classificados como *GoodCharge*.

Dos atributos usados, os que contribuíram para a classificação foram: *AppReject*, *Airtime*, *Memory*, *CCErrors*, *VoiceErrors*, ordenados do maior para o menor peso na classificação, de 100% a 8%.

Colocando a árvore sob a forma de regras:

- Se *AppReject* <=2 - *Good Charge*;
- Se *AppReject*>3 e *Airtime*>2.041721e+007 - *Best Charge*;
- Se *AppReject* >2 e *Airtime*<=2.041721e+007 - *Best Charge*;
- Se *AppReject*>2 e *Airtime*<=2.041721e+007 e *Memory* >14570 e *CCErrors*<=28 - *Good Charge*;
- Se *AppReject*>2 e *Airtime*<=2.041721e+007 e *Memory* >14570 e *CCErrors*>28 e *Memory*<=14577 - *Good Charge*;
- Se *AppReject*>2 e *Airtime*<=2.041721e+007 e *Memory* >14570 e *CCErrors*>28 e *Memory*>14577 - *Low Charge*;
- Se *AppReject*>2 e *Airtime*<=2.041721e+007 e *Memory* <=14570e - *Good Charge*;
- Se *AppReject*>2 e *Airtime*<=2.041721e+007 e *Memory* <=14570 e e *VoiceErrors*<=0 e *Memory*<=14564 -*Good Charge*;
- Se *AppReject*>2 e *Airtime*<=2.041721e+007 e *Memory* <=14570 e *VoiceErrors*<=0 e *Memory*>14564 -*Best Charge*;
- Se *AppReject*>2 e *Airtime*<=2.041721e+007 e *VoiceErrors*>0 e *Memory* <=14570- *Best* Charge;
- Se *AppReject*>2 e *Airtime*>1.922005e+007 e *Airtime*<=2.041721e+007 e *Memory* <=14570 e *VoiceErrors*>0 - *Good Charge*.

A previsão das classes deram todas resultados muito próximos, ou seja, o número de registos por classe são valores muito aproximados. A classe com maior probabilidade de ser obtida passou de *Low Charge* para *Good Charge* e tem boas possibilidades de passar a *Best Charge,* que é a segunda classe com mais registos classificados. O CPU deixou de ser o atributo usado na classificação e surgiram dois atributos novos: o *AppReject* e o *CCErrors*.

A maioria das instâncias foram classificadas como Good Charge e como a taxa de erro é menor que 10%, o IVR nesta fase é classificado como Good Charge. Como a diferença do número de elementos instanciados como Good Charge e Best Charge é pouco significativa este IVR pode estar a atingir um ponto de viragem na sua classificação (Good Charge para Best Charge).

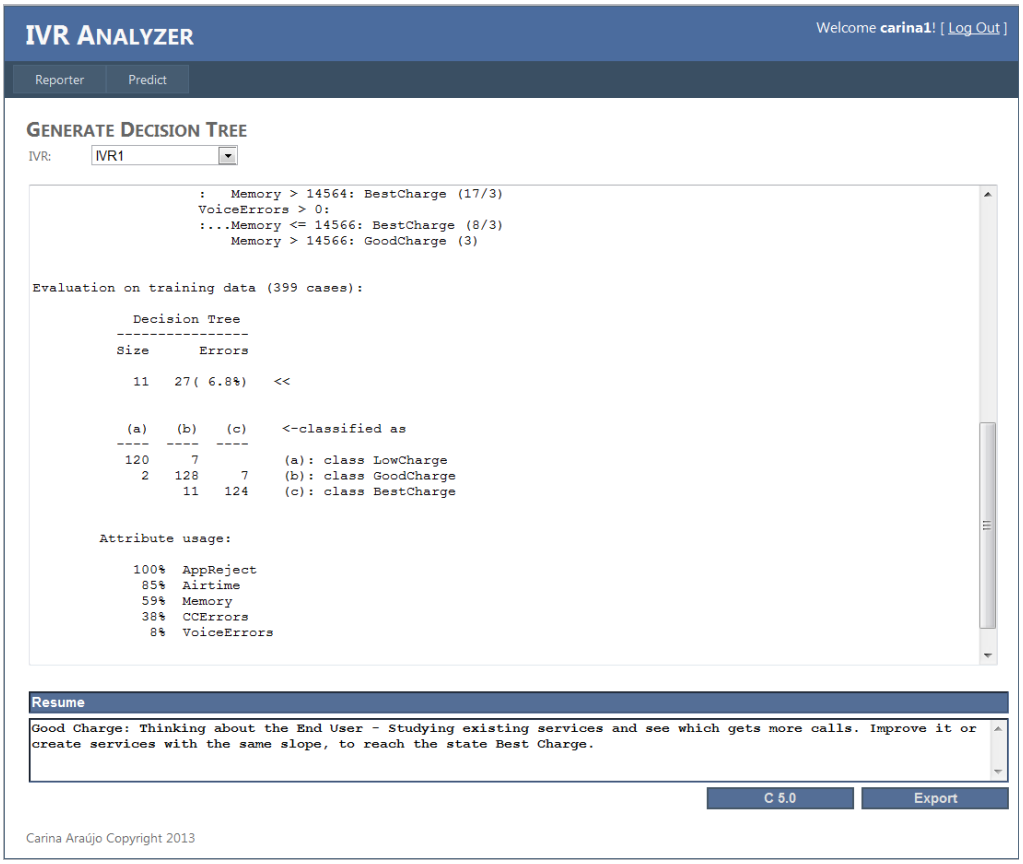

Figura 32 – Árvore de Decisão do IVR1

5.5 IVR2

Para a amostra de dados do IVR2 foi criada uma árvore com 13 folhas, em que 13 dos registos foram mal classificados, dando uma taxa de erro de 3,3 % (Figura 33).

Dos dados mal classificados:

- 8 da classe *Low Charge* classificados como *Good Charge*;
- 4 da classe *Good Charge* classificados como *Low Charge*;
- 1 da classe *Best Charge* classificado como *Good Charge*.

Dos atributos do IVR, os utilizados pelo classificador foram: o CPU, *AppReject*, *Memory* e *Airtime*, com as percentagens de contribuição de 100%, 88%, 72% e 8%, respetivamente.

Esta árvore lê-se da seguinte maneira:

- Se CPU > 3 - *Best Charge*;
- Se CPU <= 3 e *Memory*<=12230 - *Good Charge*;
- Se CPU <= 3 e *Memory*>12230 e *Airtime*>2.628772e+008- *Low Charge*;

- Se CPU <=3 e *Memory* >12230 e *Airtime*<=2.628772e+008 e *AppReject*<=657 *Low Charge*;
- Se CPU <=3 e *Memory* >12230 e *Airtime*<=2.628772e+008 e *AppReject*>657 - *Good Charge*;
- Se CPU<=1 e *AppReject* >638 - *Low Charge*;
- Se CPU<=1 e *AppReject*<=658 e *Memory* >12231 -*Low Charge*;
- Se CPU<=1 e *AppReject*<=658 e *Memory* <=12231 e *Memory*>12229 - *Good Charge*;
- Se CPU<=1 e *AppReject*<=658 e *Memory* <=12231 e *Memory*>12229 e *AppReject*<=657 - *Good Charge*;
- Se CPU<=1 e *AppReject*<=658 e *Memory* <=12231 e *Memory*>12229 e *AppReject*>657 e Memory>12230 - Low Charge;
- Se CPU<=1 e *AppReject*<=658 e *Memory* =12230 e *AppReject*>657 e *Airtime* <=2.62884e+008 - *Good Charge*;
- Se CPU<=1 e *AppReject*<=658 e *Memory*=12230 e *AppReject*>657 e *Airtime* >2.62884e+008 e *Airtime* <=2.631232e+008 - *Low Charge*;
- Se CPU<=1 e *AppReject*<=658 e *Memory*=12230 e *AppReject*>657 e *Airtime* >2.62884e+008 - *Good Charge*.

Na árvore gerada a maior parte das instâncias foram classificadas como Low Charge com mais de 90% de taxa de acerto. Assim sendo, este IVR foi classificado como Low Charge.

```
Read 399 cases (12 attributes) from IVR2.data

Decision tree:

CPU > 1:
:...CPU > 3: BestCharge (2)
:   CPU <= 3:
:   :...Memory <= 12230: GoodCharge (39/4)
:       Memory > 12230:
:       :...Airtime > 2.628772e+008: LowCharge (6)
:           Airtime <= 2.628772e+008:
:           :...AppReject <= 657: LowCharge (4/1)
:               AppReject > 657: GoodCharge (4)
CPU <= 1:
:...AppReject > 658: LowCharge (109)
    AppReject <= 658:
    :...Memory > 12231: LowCharge (193/1)
        Memory <= 12231:
        :...Memory <= 12229: GoodCharge (11/2)
            Memory > 12229:
            :...AppReject <= 657: GoodCharge (8/3)
                AppReject > 657:
                :...Memory > 12230: LowCharge (5/1)
                    Memory <= 12230:
                    :...Airtime <= 2.62884e+008: GoodCharge (3)
                        Airtime > 2.62884e+008:
                        :...Airtime <= 2.631232e+008: LowCharge (13/1)
                            Airtime > 2.631232e+008: GoodCharge (2)


Evaluation on training data (399 cases):

           Decision Tree
         ----------------
         Size      Errors

           13   13( 3.3%)   <<


          (a)   (b)   (c)    <-classified as
         ----  ----  ----
         326     8           (a): class LowCharge
           4    58           (b): class GoodCharge
                 1     2     (c): class BestCharge


        Attribute usage:

            100%  CPU
             88%  AppReject
             72%  Memory
              8%  Airtime


Time: 0.0 secs
```

Figura 33 – Árvore de Decisão do IVR2

5.6 Deteção de Problemas nos IVRs

Consultando a informação no formato de gráficos possibilita a identificação de padrões dos quais o utilizador não se havia apercebido. Como é possível comparar gráficos de diversas componentes do IVR, pode-se detetar anomalias e ocorrências de problemas. Analisando os gráficos das componentes do IVR1 verificou-se um declive abrupto nos gráficos e que se manteve a zero durante um período (Figura 34). Esta observação leva o utilizador a indagar as razões para que a plataforma tenha estado indisponível durante momentos. Este tipo de ocorrências podem ser do tipo perda de conectividade, avaria do servidor, problemas com a solução ou serviços instalados, ou *restart* ao servidor. No caso desta ocorrência, foi realizada uma intervenção na plataforma que levou ao *restart* do sistema.

Este tipo de observação pode desencadear ações preventivas, tomando controlo da situação antes de surgir um caso de *helpdesk* solicitado pelo cliente.

Figura 34 – Gráficos Número de Chamadas e Airtime do IVR1 no mesmo Período

6 Conclusão

Neste capítulo é feito um balanço de todo o trabalho realizado, os objetivos alcançados e possíveis melhorias, bem como as dificuldades sentidas e como foram ultrapassadas. Por último, é descrito o trabalho futuro a ser realizado.

6.1 Trabalho Desenvolvido

Esta dissertação teve como principal objetivo o estudo e desenvolvimento de uma solução que permitisse a recolha de informação sobre o estado atual de uma plataforma IVR.

Através do estudo de ferramentas existentes no mercado validou-se o que tinham de melhor e pior para as necessidades sentidas que deram origem a este projeto e definiu-se a solução a desenvolver com base no que era pretendido e nos prós e contras encontrados. Tal como o referido no estado de arte, as ferramentas de monitorização estudadas quando tinham agregados muitos servidores, tornavam-se difíceis de gerir e o retorno de informação era lento. Para além disso, nenhuma das aplicações inferia conhecimento a partir dos dados que armazenavam, ou seja, a informação é subvalorizada.

Para tal foi criada uma solução que devido à sua estrutura modular, o utilizador final consegue consultar a informação sem demoras e através do uso de *Data Mining* consegue classificar o IVR quanto ao seu estado.

Toda a solução foi pensada em termos de usabilidade e ergonomia, para ser fácil de navegar e gerir.

A solução dá resposta às necessidades da empresa armazenando os dados das componentes das plataformas IVR em base de dados e possibilitando a criação de relatórios de desempenho de uma forma rápida e simples. O utilizador consegue gerar relatórios com diferentes

periodicidades por componente e por IVR e exportá-los para Excel. Pode também classificar com base nos dados existentes em histórico a plataforma através da criação de árvores de decisão, que podem ser guardadas em ficheiro de texto. Sempre que é gerada uma classificação o sistema reage à mesma, dando uma recomendação para a classe retornada, com o intuito de aumentar a área de negócio, ajudando no processo de tomada de decisão.

Com os testes realizados conseguiu-se analisar a evolução ao longo do tempo dos IVR e na alteração dos atributos utilizados no modelo de classificação. Também foram detetados problemas na plataforma através do IVR Analyzer, onde se identificou uma quebra de serviço devido a uma intervenção realizada na plataforma, que deu origem a um *restart* do sistema.

No final da realização deste trabalho, da análise dos resultados obtidos e das opiniões recolhidas, concluiu-se que este tipo de ferramenta é uma mais-valia para a empresa visto:

- Resolve o problema da não existência de histórico de performance das componentes do IVR;
- A informação armazenada deixa de estar limitada aos valores de pico dos diversos parâmetros de desempenho em análise;
- Permite a geração de relatórios de desempenho de uma forma expedita;
- Possibilita a verificação do estado do IVR e parâmetros que afetam o seu desempenho, dando origem a recomendações para atuação, visando a melhoria global da plataforma podendo até potenciar o negócio.

Todos os objetivos delineados para a criação desta solução foram alcançados e dão resposta às necessidades que lhes deram origem.

6.2 Principais Dificuldades

Uma das principais dificuldades sentidas no desenvolvimento da solução foi a obtenção de árvores de decisão fidedignas necessárias para a classificação do IVR. As primeiras geradas pela solução tinham taxas de erro próximas dos cem por cento. Inicialmente, a desconfiança da causa deste problema centrou-se na invocação do algoritmo C5.0, levando a pensar que faltariam parâmetros para a sua execução. Foram realizados inúmeros testes mas, a taxa de erro mantinha-se elevada. Se o problema não era da invocação do algoritmo tinha de ser dos dados. Nesta fase, os dados estavam a ser importados de uns ficheiros CSV que são gerados pelo IVR. O problema é que os ficheiros registavam apenas os picos, ou seja, registavam apenas os máximos atingidos num dado momento. Como nunca eram no mesmo instante os dados tornavam-se incomparáveis, tornando difícil a procura de padrões. Foi criado então, o módulo IVR Data Collector permitindo armazenar a informação das componentes dos IVRs no mesmo instante de tempo. Afinado este processo, as árvores geradas passaram a ter uma taxa de erro reduzida, tornando os seus resultados mais credíveis.

Outra dificuldade prendia-se com a lentidão do portal quando era solicitada a criação de relatórios e/ou árvores de decisão. O tempo que a informação demorava a ser tratada, classificada e processada pelo See5 tornava o acesso à informação demasiado lento. Realizaram-se vários testes e decidiu-se separar as etapas de transformação, carregamento de dados em base de dados e *data mining* do portal, desenvolvendo-se o módulo Service IVR Analyzer para o fazer de forma autónoma. Assim sendo, o portal tornou-se numa ferramenta de consulta, em que a informação é retornada instantaneamente.

6.3 Possíveis Melhorias

Para o crescimento da solução foram detetados os seguintes aspetos a melhorar:

- Aumentar o nível de configuração da solução para que esta funcione de uma maneira centralizada, possibilitando a configuração dos diversos módulos através de uma única ferramenta gráfica, ou seja, mecanismo de configuração centralizado;
- Aumentar o número de componentes a monitorizar;
- Continuar o estudo do modelo de negócio para aumentar o dicionário de recomendações, para o processo de tomada de decisão ser mais expedito;
- Alargar a solução a outro tipo de plataformas;
- Continuar o trabalho de definição da estrutura e apresentação dos dados no Excel;
- Criar relatórios agregadores, ou seja, que detenham informação de mais do que um IVR e mais que uma componente em simultâneo;
- Apresentar a árvore de decisão de outra forma, torná-la mais visual, apelativa e mais fácil de interpretar;
- Realizar testes com diversos grupos de utilizadores, que desempenham funções diferentes na empresa para verificar o ponto de vista de cada um e as necessidades sentidas com a utilização desta ferramenta, para identificar melhorias na estrutura e organização da informação e identificação de futuras funcionalidades.

6.4 Trabalho Futuro

A solução criada tem como foco os servidores IVR, sendo analisado todo o conjunto de componentes que ajudam à potencialização deste produto. Dada a versatilidade e modularidade da solução desenvolvida, esta tem potencial para ser estendida a outras áreas de negócio. Os IVR são um dos tipos de sistemas a serem estudados usando esta ferramenta, podem ser feitas análises de desempenho de servidores de base de dados, servidores web, entre outros. Algumas das componentes de *hardware* até são semelhantes e têm igual importância para ambos.

Outra vertente, que pode ser implementada, é a de orientação da solução ao serviço. Recolha e armazenamento de informação referentes aos serviços telefónicos, em que são identificados dados como tipo de serviço:

- *Inbound* - chamadas de entrada, *Outbound* - chamadas de saída, *InOut* com chamadas de entrada e saída;
- Número de chamadas processadas;
- Número de erros de acesso a recursos externos;
- Erros de acesso a base de dados;
- Erros na transferência de chamadas, entre outros.

Este tipo de informação permite traçar o perfil do cliente final, identificando os serviços mais utilizados, os horários de maior carga e problemas ocorridos. Toda esta informação, armazenada e devidamente tratada e aplicando-lhe algoritmos de prospeção de dados, pode possibilitar a identificação antecipada de serviços com tendência de crescimento ou diminuição de tráfego. Esta funcionalidade contribuiria, por sua vez, para aumentar o sucesso e satisfação das plataformas IVR no cliente.

O alargamento da solução para estender a recolha de informação para outro tipo de plataformas e serviços, implementação de outros algoritmos de *data mining* para inferir novo conhecimento noutras áreas importantes para o negócio, aumentando as recomendações para ajuda ao processo de tomada de decisão.

Referências

[Alvares, 2013] Alvares, L.O, Mineração de Dados - Técnicas de Associação, p11, 2013.

[Amo, 2012] Sandra de Amo, Mineração de Regras de Associaçao - O algoritmo Apriori, http://www.deamo.prof.ufu.br/arquivos/Aula2.pdf, [último acesso: Fev 2013]

[Andreolini, Colajanni e Pietri, 2012] Andreolini, M., Colajanni,M., Pietri, M., A Scalable Architecture for Real-Time Monitoring of Large Information Systems, University of Modena and Reggio Emilia, Italy, 2012.

[Balducci, 2012] Ivan Balducci, 2012, ANOVA - Análise de Variância, http://www.ebah.com.br/content/ABAAAAPxcAF/anova-analise-variancia, [último acesso: Abr 2013]

[Berry et al., 2013] Berry I., Roman T., Adams L., Pasnak J.P., Conner J., Scheck R., Braun A. The Cacti Manual, 2013.

[Berry e Linhoff, 1997] Berry, M.J.A. and Linhoff, G, Data Mining Techniques for Marketing, Sales and Customer Support, Wiley, p27-68, 1997.

[Black, 2008] Black, T.L., Comparação de Ferramentas de Gerenciamento de Redes, Curso de Especialização em Tecnologias, Gerência e Segurança de Redes de Computadores, Instituto de Informática, Universidade Federal do Rio Grande do Sul, p-59,2008

[Bramer, 2007] Bramer, M., Principles of Data Minig, Undergraduate Topics in Computer Science, Springer, p1-11, p23, p221-230, 2007.

[Burkle, 2006] Burkle, P.Y., Um Método de Pós-Processamento de Regras de Associação com Base nas Relações de Dependência entre os Atributos, Universidade Federal Fluminense, Niterói, p15-18, 2006.

[Cacti, 2004] Cacti, 2004, Cacti - The Complete RRDTool-based Graphing Solution, http://www.cacti.net/ [último acesso: Jul 2013]

[Cisco, 2012] Cisco Systems, Inc, Simple Network Management Protocol, http://docwiki.cisco.com/wiki/Simple_Network_Management_Protocol [último acesso: Mai 2013]

[Craven e Shavlik, 1997] Craven, M.W. and Shavlik, J.W., Using Neural Networks for Data Mining, School of Computer Science Carnegie Mellon University Pittsburgh and Computer Sciences Department University of Wisconsin - Madison.

[Crawford, 2011] Michael Crawford, ID3 Algorithm, 2011.

[Cruz, 2007] Cruz, A. J. R, Data Mining via Redes Neuronais Artificiais e Máquinas de Vectores de Suporte, Escola de Engenharia, Departamento de Sistemas de Informação, Universidade do Minho, 2007.

[Fayyad et al.,1996] Fayyad, U.M. et al., Advances in Knowledge Discovery and Data Mining. California, AAAI Press, 1996.

[Fred, 2006] Fred, A.L.N., Redes Bayesianas, Instituto Superior Técnico de Lisboa, p-1, 2006.

[Gite, 2002] Vivek G. Gite, Linux Shell Scripting Tutorial v1.05r3 - A Beginner's handbook, http://www.freeos.com/guides/lsst/, [último acesso: Mai 2013]

[Gonçalves, 2010] Gonçalves, F, Zabbix: o "state-of-the-art" na Monitorização de servidores Windows e Linux e dispositivos de rede, http://www.softelabs.com/Suporte/Tech_News/Zabbix%3A_o_State-of-the-

art_na_Monitoriza%C3%A7%C3%A3o_de_serviodres_Windows_e_Linsu_e_dispositivos_de_rede#attachForm, [último acesso: Jan 2013]

[Hamilton, 2012] Howard Hamilton, 2012,Knowledge Discovery in Databases, http://www2.cs.uregina.ca/~dbd/cs831/notes/ml/dtrees/c4.5/tutorial.html, [último acesso: Fev 2013]

[Livs e Sasha, 2012] Livs, Martins, Sasha, Wiper, Zabbix Manual, https://www.zabbix.com/documentation/2.2/manual/ [último acesso: Nov 2012]

[Kohavi e Quinlan, 1999] Kohavi, R e Quinlan, R., Decision Tree Discovery, p1-8 , 1999.

[Kovacs, 2013] Kovacs, K., Zabbix vs Nagios Comparison, http://kkovacs.eu/zabbix-vs-nagios, [Último Acesso: Mar 2013]

[Lowry, 2013] Richard Lowry, 2013, The Fisher Exact Probability Test, http://www.vassarstats.net/textbook/index.html, [último acesso: Mar 2013]

[Marik e Zitta, 2014] Marik, O. and Zitta, S., Comparative Analysis of Monitoring System for Data Networks, Faculty of Electrical Engineering and Informatics, University of Pardubice, Pardubice, Czech Republic, 2014.

[Microsoft, 2012] Microsoft, 2012, Introducting C#, http://msdn.microsoft.com/en-us/library/hh145616(VS.88).aspx, [último acesso: Mai 2013]

[Mitchell, 2013a] Mitchell, B., FTP - What Does FTP Stand For?, http://compnetworking.about.com/od/networkprotocols/g/bldef_ftp.htm, [último acesso: Abr 2013]

[Mitchell, 2013b] Mitchell, B., SFTP, http://compnetworking.about.com/od/ftpfiletransfer/g/sftp-definition.htm, [ultimo acesso: Abr 2013]

[Moreira, 1997] Moreira, M.A., Introdução às Redes Neuronais Artificias, 1997.

[Nagios, 2013] Nagios, 2009, Nagios Is The Industry Standard In IT Infrastructure Monitoring, http://www.nagios.org [último acesso: Jul 2013]

[Neville, 1999] Neville, P.G., Decision Trees for Predictive Modeling, SAS Institute Inc, 1999.

[Oetiker , 2013] Oetiker ,Tobias, 2013. RRDTool logging and graphing, http://oss.oetiker.ch/rrdtool/doc/rrdtool.en.html [último acesso: Fev 2013]

[OpManager, 2013] Zoho Corporation Pvt. Ltd., SNMP tutorial, http://www.manageengine.com/network-monitoring/what-is-snmp.html [último acesso: Mai 2013]

[Quinlan, 1986a] Quinlan, J.R, Induction of Decision Trees, Mach, p81-106, 1986.

[Quinlan, 2010b] J.R. Quinlan, 2010, C4.5 and Beyond, http://www.cs.uvm.edu/~xwu/kdd/Slides/C4.5byRossQuinlanforICDM06.pdf, [último acesso: Mar 2013]

[Rizvi, 2010] Abbas Rizvi, ID3 Algorithm, 2010.

[Rodrigues, 2005] Rodrigues, M.A.S., Monografia Árvores de Classificação, Universidade dos Açores, Departamento de Matemática, 2005.

[Rokach, 2008] Rokach, L., Decision Trees, Department of Industrial Engineering Tel-Aviv University, 2008.

[Ruivo, 2010] Ruivo, Ricardo. Nagios - Monitorização Open Source. PPLWARE, 2010.

[Santos, 2008] Santos, Mason, Zabbix - Uma poderosa Ferramenta de Monitoramento, http://www.softelabs.com/@api/deki/files/111/=ZABBIX_-_Uma_poderosa_ferramenta_de_monitoramento.pdf [último acesso: Jan 2013]

[See5, 2012]	See5, 2012, Is See5/C5.0 Better Than C4.5?, http://rulequest.com/see5-comparison.html, [último acesso: Mar 2013]
[See5, 2013]	See5, See5-An Informal Tutorial, http://www.rulequest.com/see5-win.html#CONSOLE, [último acesso: Mai 2013]
[Shih, 2012]	Yu-Shan Shih, QUEST Classification Tree, http://www.stat.wisc.edu/~loh/quest.html, [último acesso: Abr 2013]
[Singh e Chauhan, 2009]	Dr. Yashpal Singh and Alok Singh Chauhan, Neural Networks in Data Mining, Journal of Theoretical and Applied Information Technology, Bundelkhan Institute of Engineering & Tecnology, Jhansi, India and United Institute of Management, Allahabad, India, 2009.
[SNMP, 2013]	SNMP Research International, Inc, SNMP RFCs, http://www.snmp.com/protocol/snmp_rfcs.shtml [último acesso: Mai 2013]
[StatSoft, 2013]	StatSoft, 2013, Popular Decision Tree: CHAID Analysis, Automatic Interaction Detection, [ultimo Acesso: Mar 2013]
[Steinberg, 1999]	Dan Steinberg, Data Mining with Decision Trees: An Introduction to CART, Califórnia, 1999.
[Thomas, 2012]	Andy Thomas, About SCP, http://www2.imperial.ac.uk/~andy/sysnews/scp/, [último acesso: Abr 2013]
[Viali, 2012]	Prof. Dr. Lori Viali, Testes Não Paramétricos - Duas Amostras Independeentes, http://www.mat.ufrgs.br/~viali/estatistica/mat2282/material/laminaspi/Mat2282_2_Ind.pdf,[último acesso: Abr 2013]
[Vithlani, 2012]	Nikhil Vithlani, 2012, Apriori algorithm for Data Mining – made simple, http://nikhilvithlani.blogspot.pt/2012/03/apriori-algorithm-for-data-mining-made.html , [último acesso: Fev 2013]
[WinSCP, 2012]	WinSCP, WinSCP .NET Assembly and COM Library, http://winscp.net/eng/docs/library [último acesso: Abr 2013]
[Wu e Kumar e Quinlan, et al, 2007]	Xindong Wu, Vipin Kumar, J.Ross Quinlan, et all, Survey Paper - Top 10 Algorithms in Data Mining, http://www.cs.uvm.edu/~icdm/algorithms/10Algorithms-08.pdf, [último acesso: Abr 2013]
[Zabbix, 2001]	Zabbix, 2001, Zabbix - The Enterprise-class Monitoring Solution for Everyone, http://www.zabbix.com/ [último acesso: Jul 2013]