



DETERMINAÇÃO DE PADRÕES DE DESISTÊNCIA EM GINÁSIOS

Autor: Paulo Pinheiro, Nº 901659
Orientador: Professor Doutor Luis Cavique

Licenciatura em Informática
UC 21095: Projeto Final – Relatório final
6 de Julho de 2015

Agradecimentos

Ficam os agradecimentos ao Professor Doutor Luis Cavique por ter aceitado a orientação deste trabalho, pelo incentivo e colaboração dada ao longo do seu desenvolvimento e que foram determinantes para o resultado final.

Fica também uma palavra de apreço para todos os professores que ao longo do curso foram dando as suas indicações e sugestões, fundamentais neste tipo de aprendizagem.

Finalmente, fica o agradecimento especial à família, quer pela paciência e compreensão nas horas mais difíceis, quer pelas ausências a que por vezes tive de sujeitá-la para conseguir levar este projeto até ao fim.

1 Índice

Agradecimentos	1
1 Índice	2
2 Introdução e enquadramento	3
3 Objetivos e resultados esperados	5
4 Determinação do modelo de Data Mining	6
4.1 Compreender o problema	6
4.2 Perceber os dados	7
4.3 Preparar os dados	11
4.4 Modelação	21
Sobre os atributos a utilizar	21
Sobre os algoritmos a utilizar e procedimentos a efetuar	23
Aplicação dos algoritmos disponibilizados pelo Analysis Services	24
Aplicação dos algoritmos disponibilizados no R	30
Conclusões finais sobre algoritmos e atributos escolhidos	35
5 Desenvolvimento	39
5.1 Análise da aplicação	39
A primeira camada: procedimentos implementados na base de dados	40
A segunda camada: os serviços web (“webservices”)	46
A camada de apresentação: os formulários	69
5.2 Manual da aplicação	79
Informação Estatística	79
Simular	80
Relatório de Desistentes	81
Atualizar Modelo	82
6 Conclusões	83
Sobre o projeto	83
Subdomínios a explorar	84
Sobre a implementação do projeto	85
Lista de termos	86
Referências	88

2 Introdução e enquadramento

Atualmente, reter os clientes tem vindo a ser considerado tão importante como adquirir novos clientes. Como em qualquer outra organização com fins lucrativos, os ginásios perceberam que é mais barato manter longas relações com os clientes atuais do que investir só na aquisição de novos clientes ([Dhurup, 2012](#)).

No sentido de melhorar a retenção, os operadores da indústria do fitness têm vindo a medi-la através da utilização da taxa de retenção (“Retention Rate”) - definida como a proporção dos utentes que permanecem por um determinado período de tempo – e do valor do tempo de vida (“Lifetime value”) – a média de tempo que os utentes permanecem e quanto pagam pela sua permanência ([IHRSA Member Retention Report, Vol 1 Issue 1, 2012](#)). No entanto, estes indicadores apenas “medem” e não permitem definir um padrão sobre quem são os utentes que abandonam o ginásio.

Abordagens mais recentes promovidas pela International Health, Racquet & Sportsclub Association (IHRSA) vão mais além traçando o perfil do potencial desistente através da classificação dos utentes em promotores, passivos e detratores. Esta classificação é efetuada através da aplicação de um questionário onde o utente é inquirido sobre se recomendaria o ginásio a um amigo ou colega, pontuando essa possibilidade de 0 a 10. Sobre os dados obtidos do questionário, são posteriormente procuradas algumas características comuns procurando traçar o perfil tendo em consideração alguns dados demográficos dos utentes (sexo, idade) e a antiguidade da sua inscrição ([IHRSA Member Retention Report, Vol 1 Issue 4, 2013](#)).

Uma vez que a indústria do Fitness tem apostado na instalação massificada de sistemas avançados de “Customer Relationship Management” (CRM) (Correia & Sacavém, 2006) existem atualmente bases de dados com dados históricos de grande valia. Para além dos dados históricos, as funcionalidades de sistema de fidelização, de faturação, de controlo de acessos e de “checkin” (registo de entrada) às atividades permitem conhecer com pormenor as preferências e comportamento dos utentes durante o seu “tempo de vida” no ginásio.

É neste contexto que se propõe a elaboração deste projeto. Apesar de, como referimos anteriormente, a retenção ser uma questão comum a todas as

atividades comerciais e particularmente às organizações que prestam serviços desportivos, este trabalho limita o âmbito da sua aplicabilidade aos Ginásios e Academias de Fitness, e às atividades de Fitness. Tal deve-se ao fato da relação comercial noutras atividades e organizações desportivas apresentarem relações de fidelização não contínuas ou estarem previamente segmentadas, o que faria aumentar a complexidade e a possibilidade de divergência quanto aos resultados que se espera obter.

3 Objetivos e resultados esperados

Este projeto tem por principal objetivo, através de técnicas de Classificação, determinar o algoritmo mais adequado para encontrar padrões que permitam prever quais os utentes que irão abandonar ou cancelar a sua inscrição num próximo período, atendendo ao padrão de perfil dos utentes que desistiram nos últimos meses, ao seu padrão de consumo e utilização e a eventuais dados demográficos que se venham a considerar relevantes para o estudo em questão.

Depois de determinado o algoritmo para o problema em questão, pretende-se implementar um conjunto de ferramentas em linguagem C# que, utilizando o algoritmo escolhido, permita aplicar ao caso prático de um software de gestão de ginásios, permitindo que este determine o perfil de risco e produza relatórios de utentes que se enquadrem nesse perfil de risco.

O desenvolvimento deve resolver as seguintes necessidades:

- Atualização periódica do padrão de desistência;
- Através do perfil de um utente, no que diz respeito aos atributos considerados, classifica-lo, em termos da sua possível desistência ou cancelamento dos serviços de fitness;

Como suporte ao desenvolvimento, poderão ser invocadas funcionalidades dos packages de software utilizados no presente estudo (Analysis Services ou R).

4 Determinação do modelo de Data Mining

O desenvolvimento deste projeto seguiu as orientações da metodologia CRISP-DM, pelo que são apresentadas de seguida as abordagens efetuadas em cada uma das fases indicadas por esta metodologia.

4.1 Compreender o problema

Tal como referido anteriormente, o que se pretende é determinar um padrão de comportamento que permita aferir, com antecedência, quais os utentes que terão maior probabilidade de virem a desistir da frequência do Ginásio ou Academia. Tal aferição permitirá ao gestor tomar medidas eventualmente personalizadas nos utentes que apresentem tais padrões e que permitam prolongar a duração da frequência dos utentes, resultando no aumento da faturação e consequentemente nos resultados da sua instalação.

Para que seja possível determinar periodicamente estes padrões recorrendo a métodos de Data Mining é necessário recorrer a bases de dados que detenham uma relação adequada de atributos e histórico de informação.

No intuito de reduzir os constrangimentos com base no volume de dados e de alargar as possibilidades de escolha de atributos para o problema em questão, selecionou-se uma base de dados que, para além de apresentar um volume de dados adequado, apresentasse também registos de controlo de acessos e de frequência das atividades. A base de dados em questão suporta o sistema de informação de 13 clubes em Braga, Cascais, Lagos, Lisboa, Matosinhos, Oeiras, Porto, Viana do Castelo e Vila Nova de Gaia.

Esta base de dados, implementada em Microsoft SQL*Server®, contém toda a informação do negócio do cliente, não só a informação referente ao core-business – atividades de fitness -, como também a informação relativa a outras atividades e serviços desportivos que são prestados nas várias instalações espalhadas pelo país.

O carregamento de dados por parte dos operadores é feito através de uma aplicação comercializada no mercado (e@sport®). Algumas das características desta aplicação incluem o registo dos dados demográficos dos utentes, a faturação efetuada sobre os contratos de adesão e outros produtos e serviços adquiridos pelo utente, bem como o registo dos acessos e das atividades diárias

praticadas pelos utentes dentro dos vários ginásios. Alguns dos clubes têm particularidades que não permitem a recolha de alguns dados, particularidades essas assinaladas em [Perceber os dados](#).

Conhecendo-se estas características e após algum tratamento, mesmo não havendo registo de alguns dados que poderiam ser importantes para afinação dos modelos (ver [Perceber os dados](#) e [Preparar os dados](#)), reconhece-se ser uma boa base de trabalho para aferir conclusões após aplicação dos modelos.

Não obstante, é importante referir que a aplicação evoluiu ao longo do tempo, apresentando diferentes graus de exigência em termos de validação de dados pelo que se torna importante não descuidar a tarefa de análise da qualidade dos dados.

Para o desenvolvimento do projeto foram utilizadas as ferramentas de Data Mining (DM): Microsoft SQL*Server[®] 2014 (MS-SQL), Analysis Services Ver. 12.0.2000.8 (MS-SQL AS) e R version 3.1.3 (2015-03-09) da The R Foundation for Statistical Computing. A opção pela utilização das duas ferramentas permitirá comparar e complementar resultados.

O MS-SQL é a ferramenta destinada a implementação de interrogações diretas à base de dados.

O MS-SQL AS permitirá definir os modelos de DM e obter resultados de forma mais rápida e simples. Numa segunda fase, o R permitirá obter resultados de forma mais concisa e pormenorizada.

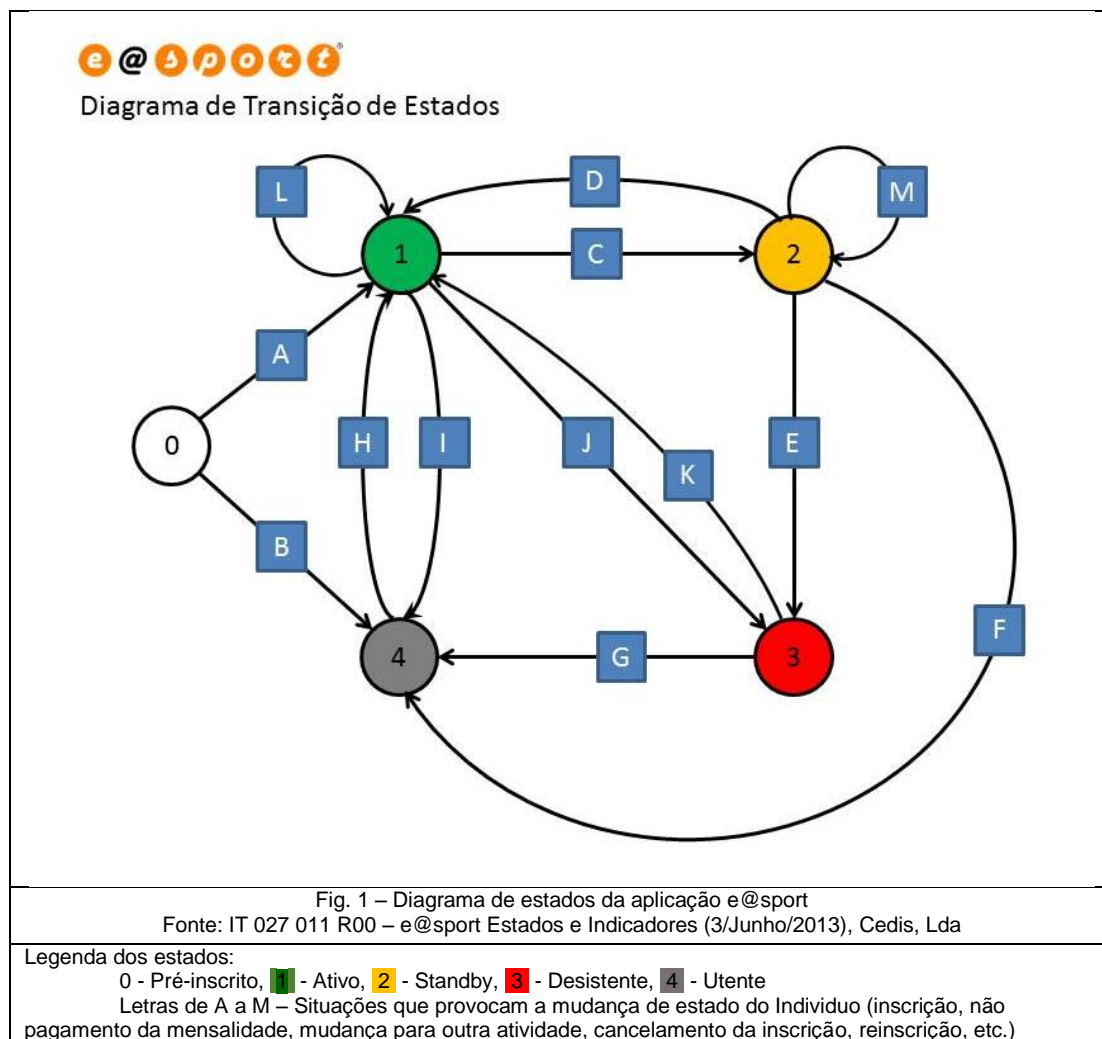
Tanto o MS-SQL como o R serão também utilizados para proceder à deteção e correção de dados na fase de preparação e limpeza de dados e também na implementação das rotinas a desenvolver em C#.

4.2 Perceber os dados

Como referido anteriormente, interessa captar os padrões de desistência de utentes de atividades de Fitness. Estas atividades estão classificadas, a nível da base de dados, como atividades referentes a receita principal. Interessa por isso considerar apenas os utentes inscritos, ou, sendo desistentes, que tiveram pelo menos uma inscrição numa atividade de receita principal.

Em função das atividades em que estão inscritos, os utentes podem assumir em cada instante apenas um dos seguintes estados: Ativo, Standby, Desistente, Utente ou Pré-Inscrito.

O diagrama de estados da figura seguinte ilustra as transições de estados possíveis.



Interessa-nos para este projeto as transições para o estado de Desistente (3) ou Utente (4) provenientes dos estados Ativo (1) ou Standby (2).

A observação das tabelas da base de dados do Cliente, em conjunto com estudos e literatura especializada sobre o assunto, permitiu identificar os atributos mencionados na [Tab. 1](#).

Nº	Atributo a considerar	Justificação
1	Situação atual	Estado do Utente em termos da sua permanência. Este é o atributo que se pretende prever.
2	Idade do Utente	Idade do Utente, em anos
3	Género do Utente	Sexo do utente

Nº	Atributo a considerar	Justificação
4	Antiguidade da inscrição	Duração da inscrição, em meses – pode ser utilizado como o atributo base para a “recenticidade” – o R - do método RFM
5	Frequência média de visitas ao clube	Frequência média semanal de visitas ao clube – pode ser utilizado como o atributo base para a “frequência” – o F - do método RFM
6	Volume de negócios	Volume de negócios total – pode ser utilizado como o atributo base para o “monetário” – o M – do método RFM
7	Número de dias sem frequentar o clube	Último intervalo de dias em que o utente não frequentou o clube
8	Frequência de Aulas de Grupo	Número total de aulas de grupo frequentadas
9	Número de contatos entre o Clube e o Utente	Contagem do número de registos de contatos entre a data da inscrição e a data do cálculo ou a data da desistência, excluindo-se registos de informações internas
10	Reclamações efetuadas pelo Utente	Contagem do número de registos de reclamações/queixas, por gravidade, entre a data da adesão e a data do cálculo ou a data da desistência
11	Familiars ou amigos que também frequentam o clube	Contagem do número de familiares associados e de referências dadas entre a data da adesão e a data do cálculo ou a data da desistência
12	Distância a percorrer para chegar ao clube	Conteúdo do campo adicional destinado ao efeito e que contém a distância – em tempo ou quilómetros – a percorrer a partir do local de onde o utente sai para se deslocar ao Ginásio

Tab.1 – Atributos promissores identificados na observação da base de dados

Apesar de poderem ser registados pela aplicação utilizada, constatou-se que a base de dados em análise não possui informação registada sobre os atributos 9, 10, 11 e 12, pelo que se optou por não ter os mesmos em conta nos modelos a construir.

A partir da tabela anterior, identificam-se as tabelas e atributos de origem, e respetivas formas de cálculo para obter os atributos promissores para os modelos de DM a aplicar.

Nº	Atributo	Tipo	Como pode ser obtido	Tabela/Atributo
1	Situação atual	Bit (calculado)	Estado do Utente em termos da sua permanência. Considerar 1 para o estado Desistente ou 0 (zero) caso contrário	spouteestados.ust_inicio
2	Idade do Utente	Int (calculado)	Calculada pela diferença em anos entre a data de nascimento e a data da desistência ou a data do último dia do mês anterior à data atual	spoidividuos.ind_dtnasc
3	Género do Utente	Varchar(1)	Sexo do utente, indicado como M para masculino e F para feminino	spoidividuos.ind_sexo
4	Antiguidade da inscrição	Int (calculado)	Diferença, em meses, entre a menor data em que o Utente assumiu o estado Ativo e a data da desistência ou a data do último dia do mês anterior à data atual	spoidividuos.ind_dtinsert
5	Frequência média semanal de visitas ao clube	Int (calculado)	Contagem do número de visitas ao clube entre a data da inscrição e a data da desistência ou a data do último dia do mês anterior à data atual a dividir pelo número de semanas da inscrição	spomovime.mov_data
6	Volume de negócios	Numeric(12,2) (calculado)	Somatório de todos os valores faturados entre a data de início da inscrição e a data da desistência ou a data do último dia do mês anterior à data atual	spodocumentos_master.cma_iliquido
7	Número de dias sem frequentar o clube	Int (calculado)	Diferença entre a data mais recente de um movimento no controlo de acessos ou no checkin e a data do cálculo ou a data da desistência	spomovime.mov_data
8	Frequência de Aulas de Grupo	Int (calculado)	Número total de aulas de grupo frequentadas entre a data da adesão e a data do cálculo ou a data da desistência	spomovime.mov_data

Nº	Atributo	Tipo	Como pode ser obtido	Tabela/Atributo
9	Identificação única do registo	Uniqueidentifier (Primary key)		spindividuos.ind_id
10	Identificação única do registo	Int (Identity)		spindividuos.ind_saft

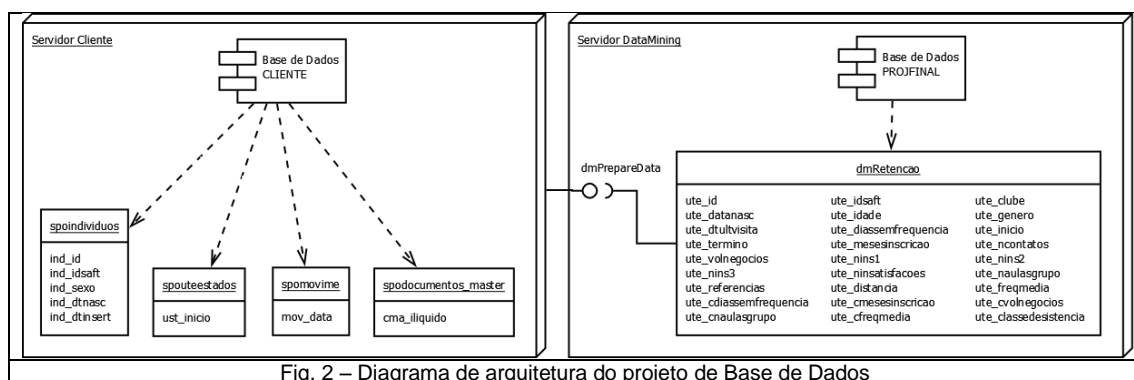
Tab. 2 – Atributos promissores
A coluna Tipo indica o tipo do atributo na base de dados do Cliente ou o tipo do resultado do cálculo
Os atributos 9 e 10 são mencionados como identificadores únicos na base de dados do cliente e na base de dados de suporte aos modelos de DM, respetivamente

No sentido de ter uma noção exata do volume de dados a tratar, contou-se o número de registos das tabelas que contêm os atributos relevantes.

Tabela	Tipo de conteúdo	Nº de registos
spindividuos	Contém dados genéricos e demográficos sobre todos os utentes do Ginásio	165.370
spouteestados	Contém as transições de estado dos utentes enquanto clientes do Ginásio	1.713.800
spmovime	Contém os registos de acesso e de presenças em aulas de grupo	20.083.391
spodocumentos_master	Contém o cabeçalho dos documentos de faturação emitidos	1.518.583

Tab. 3 – Dimensão, em número de registos, das tabelas que contêm ou a partir das quais se obtêm os atributos promissores, à data da obtenção da base de dados (31/Março/2015)

Tendo em atenção o número considerável de registos a tratar nas várias tabelas, ao fato de se saber que existem na base de dados registos de utentes que não devem ser considerados, à possibilidade de ocorrência de erros relativos à qualidade dos dados, bem como a necessidade de não afetar a base de dados do cliente optou-se por criar uma estrutura de suporte ao modelo de DM separada, de acordo com o diagrama da [Fig. 2](#):



São assim levadas a cabo diversas operações de seleção, integração e limpeza, operações de transformação e operações de redução que se refletem numa tabela única, e que pode ser eventualmente alojada noutra base de dados e/ou noutra servidor. Esta abordagem, para além de simplificar a aplicação dos modelos de DM, reduz o volume de dados a tratar resultando por isso em menores tempos de execução dos modelos.

Uma vez que os modelos de DM irão ser aplicados apenas a um subconjunto dos utentes registados na base de dados – apenas os que praticam ou alguma vez praticaram Fitness e que são cerca de 30% dos utentes – a referência a inconsistências (noise) óbvias ou menos óbvias, a valores duplicados ou em falta, e a valores com características consideravelmente diferentes (outliers) foi deixada para a fase de [Preparar os dados](#).

É de realçar que se pretende que os padrões de desistência detetados evoluam ao longo do tempo com as alterações de comportamento dos utentes, no que diz respeito aos atributos considerados. Por isso, após a identificação das operações a efetuar sobre os dados e após a deteção de todas as falhas na qualidade dos dados, será de implementar uma rotina ou procedimento que execute todas as operações requeridas e que possa ser agendada para execução periódica, atualizando os dados de suporte ao modelo escolhido.

Atendendo a que os indicadores de retenção são atualmente analisados ao mês, considera-se como base de trabalho para este projeto, os dados apurados entre 01 de Agosto de 2012 – dia em que os clubes iniciaram a utilização do atual software - e o último dia do mês anterior ao corrente. No entanto, todos os valores correspondentes aos modelos de Data Mining apresentados na primeira parte deste relatório foram obtidos até Março de 2015, inclusive.

Uma vez que a transformação dos dados (ver [Preparar os dados](#)) prevê a acumulação dos mesmos ao longo do tempo, prevê-se que o aumento do volume de dados de mês para mês permita afinar a precisão de apuramento do padrão.

4.3 Preparar os dados

De acordo com a perceção obtida na fase de [Perceber os dados](#), criou-se uma tabela designada por **dmRetencao** para suporte dos dados e aplicação dos modelos que se apresenta de seguida. A [Tab. 4](#) refere o nome e tipo do atributo e o nome do procedimento da base de dados (“storage procedure”) que concretiza a transformação especificada na [Tab. 2](#) da fase de [Perceber os dados](#).

Nº	Atributo	Tipo	Procedimento	Observações
1	Ute_id	Uniqueidentifier	dmlImportaUtentes	
2	Ute_idsaft	Int	dmlImportaUtentes	
3	Ute_clube	Varchar(2)	dmlImportautentes	
4	Ute_datanasc	Datetime	dmlImportaUtentes	
5	Ute_idade	Int	dmlIdadeUtentes	

Nº	Atributo	Tipo	Procedimento	Observações
6	Ute_genero	Bit	dmImportaUtentes	
7	Ute_dtultvisita	Datetime	dmUltFrequencia	
8	Ute_diassemfrequencia	Int	dmUltFrequencia	
9	Ute_cdiassemfrequencia	Int	dmClassificaDiasSemFrequencia	
10	Ute_inicio	Datetime	dmImportaUtentes	
11	Ute_termino	Datetime	dmPeriodoAtivo	
12	Ute_mesesinscricao	Int	dmPeriodoAtivo	
13	Ute_cmeseinscricao	Int	dmClassificaMesesDaInscricao	
14	Ute_ncontatos	Int	dmContatosUtentes	*** não utilizado ***
15	Ute_volnegocios	Numeric(12,2)	dmVolNegociosUtentes	
16	Ute_cvolnegocios	Int	dmClassificaVolNegocios	
17	Ute_nins1	Int	dmInsatisfacao	*** não utilizado ***
18	Ute_nins2	Int	dmInsatisfacao	*** não utilizado ***
19	Ute_nins3	Int	dmInsatisfacao	*** não utilizado ***
20	Ute_ninsatisfacoes	Calculado		*** não utilizado ***
21	Ute_aulasgrupo	Int	dmAulasGrupo	
22	Ute_cnaulasgrupo	Int	dmClassificaNAulasGrupo	
23	Ute_referencias	Bit	dmReferencias	*** não utilizado ***
24	Ute_distancia	Int	dmDistancia	*** não utilizado ***
25	Ute_freqmedia	Numeric(12,2)	dmFreqMedia	
26	Ute_cfreqmedia	Int	dmClassificaFreqMedia	
27	Ute_classedesistencia	Bit	dmPeriodoAtivo	

Tab. 4 – Lista dos atributos da tabela dmRetencao

O processo de preparação dos dados na tabela **dmRetencao** é descrito por fases para que se possam identificar as tarefas de análise e tratamento dos dados.

Na fase inicial processa-se a integração dos dados provenientes das várias tabelas da base de dados do Cliente, através da execução de uma sequência de procedimentos. A [Tab. 5](#) identifica a ordem e também a função de cada um dos procedimentos executados nesta fase.

Ordem de execução	Procedimento	Função
1	dmImportaUtentes	Importa os atributos de identificação únicos e dados demográficos (data de nascimento e género) de todos os Utentes da base de dados do Cliente. De forma a permitir a adição de novos utentes para períodos sucessivos, os utentes são inseridos por ciclos periódicos mensais.
2	dmInicializaUtentes	Coloca a NULL todos os campos calculados e cujo valor pode alterar de um período para outro
3	dmPeriodoAtivo	Determina a situação atual dos Utentes, preenchendo o campo dmRetencao.ute_classedesistencia=1 se o utente desistiu, ou dmRetencao.ute_classedesistencia=0 se ainda se encontra a praticar. Os Utentes que não se enquadram na prática do Fitness mantêm o atributo com o valor NULL nesta fase
4	dmIdadeUtentes	Calcula a idade dos Utentes à data da desistência ou à data final do período em processamento (normalmente o último dia do mês anterior). O procedimento utiliza a data do dia como data de nascimento caso a data de nascimento não se encontre preenchida
5	dmUltFrequencia	Determina a última data de frequência de cada Utente igual ou anterior à data da desistência, ou à data final do período em processamento. Os utentes que não apresentam qualquer registo de controlo de acessos ou de checkin mantêm o atributo dmRetencao.ute_dtultvisita=NULL e consequentemente também o atributo dmRetencao.ute_diassemfrequencia=NULL
6	dmVolNegociosUtentes	Calcula o volume de negócios entre a data de adesão e a data da desistência ou à data final do período de processamento. Os utentes para os quais não foi emitido nenhum documento neste período manterão o atributo dmRetencao.ute_volnegocios=NULL
7	dmAulasGrupo	Conta, através do registo de checkin, o número de aulas de grupo frequentadas por cada utente entre a data de adesão e a data da desistência ou à data final do período de processamento.

Ordem de execução	Procedimento	Função
		Os utentes que nunca frequentaram uma aula de grupo ou estão inscritos num ginásio que não disponibiliza o checkin aos seus utentes apresentam o atributo <code>dmRetencao.ute_naulasgrupo=NULL</code>
8	<code>dmFreqMedia</code>	Conta, através de um qualquer registo de presença (controlo de acessos ou checkin) o número de presenças entre a data de adesão e a data da desistência ou à data final do período de processamento, e obtém a média de frequências semanais dividindo a contagem pelo nº de semanas calculado pela diferença das datas utilizadas para contar os registos de presença. Se não for detetado nenhum registo de presença, o atributo <code>dmRetencao.ute_freqmedia=NULL</code>

Tab. 5 - Ordem e função dos procedimentos executados na fase 1

Após a execução dos referidos registos, a tabela ***dmRetencao*** apresenta um total de 49.875 registos. A diferença entre o número de utentes na base de dados do cliente (165.370 – ver [Tab. 3](#)) e este número reside no fato de terem sido tratados apenas utentes que tenham sido inseridos na base de dados a partir de 01 de Agosto de 2012. Foi considerada esta data uma vez que foi quando se iniciou a utilização da aplicação atual e se pode garantir a uniformização do registo dos dados em todos os clubes.

Tendo os dados preenchidos nesta fase, procedeu-se à localização de valores duplicados ou em falta, inconsistências (noise) óbvias ou menos óbvias, e a valores com características consideravelmente diferentes (outliers), como explicado nos seguintes parágrafos.

Para detetar estas situações, utilizaram-se algumas ferramentas disponibilizadas pelo R e alguns queries executados diretamente na base de dados. A [Fig. 3](#) apresenta um sumário estatístico dos atributos relevantes do “data frame” carregado com todos os registos da tabela ***dmRetencao*** após a execução dos procedimentos indicados na [Tab. 5](#).

Uma vez que ainda não foi efetuada nenhuma operação de limpeza, ignora-se nesta fase os dados estatísticos (Mínimos e máximos, média e quartis) e observa-se apenas os valores em falta (NAs) de cada atributo, procurando obter-se a justificação para a ausência dos mesmos e decidir sobre qual a melhor opção a tomar para resolver a ausência dos valores, tendo em atenção os possíveis métodos de resolução.

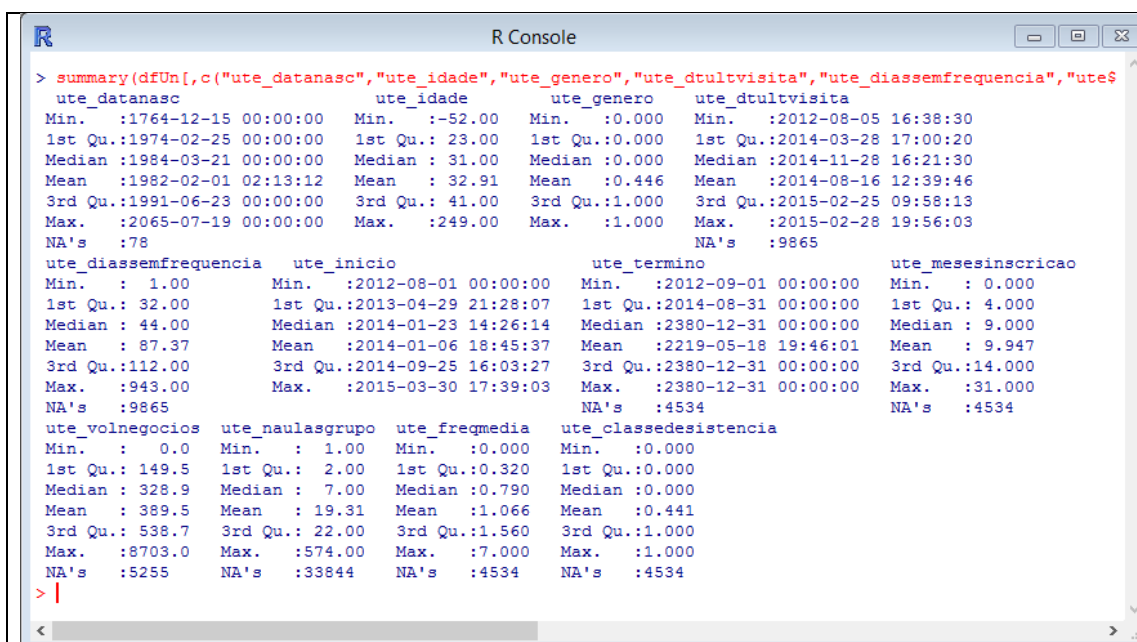


Fig. 3 – Sumário das características e quantis dos atributos promissores após a integração dos dados a partir da base de dados CLIENTE

A [Tab. 6](#) apresenta os atributos que contêm valores por preencher, o nº de registos afetados, e a decisão sobre o que fazer com esses registos.

Nº	Atributo	NºRegistos com NA	Operação a efetuar
1	Ute_datanasc	78	Tratando-se de ausência de dados, opta-se por remover os registos que não têm este atributo preenchido
2	Ute_dtultvisita	9865	Uma vez que a data da última visita é obtida a partir da ocorrência de pelo menos um registo no controlo de acessos, a ausência deste valor pode significar nunca ter havido uma visita, pelo que é necessário cruzar esta observação com outros atributos. O atributo ute_diassemfrequencia é calculado a partir da data da última visita, pelo que o número de valores em falta é igual
3	Ute_diassemfrequencia	9865	
4	Ute_volnegocios	5255	A ausência de um valor relativo ao volume de negócios significa que nunca houve qualquer pagamento e portanto o utente nunca usufruiu das atividades de fitness, pelo que os registos nesta situação não devem ser considerados. Opta-se por remover os registos que não tiverem este atributo preenchido
5	Ute_naulasgrupo	33844	Sabe-se à partida que muitos dos utentes que frequentam Ginásios não frequentam aulas de grupo, pelo que, após a remoção dos registos por ausência de preenchimento de outros atributos, este atributo deverá ser colocado com o valor 0 (zero)
6	Ute_freqmedia	4534	A fórmula da frequência média é obtida pela divisão do número de presenças diárias (contando apenas 1 presença por dia, quer seja no controlo de acessos, quer seja no checkin) pelo número de semanas que decorrem entre a data de início e de término da inscrição. O número de registos sem valor obtido decorre portanto quer da ausência de registos de presença, quer da ausência de indicação do término da inscrição, pelo que a opção do tratamento a dar a estes registos depende também do cruzamento das observações dos outros atributos.
7	Ute_termino	4534	A data de término é obtida através da data de início e de término de um dos estados relevantes (Ativo, Standby ou Desistente) para a atividade Fitness, pelo que a ausência deste dado significa que o utente não usufrui ou usufruiu dessa atividade não devendo ser considerado. O mesmo resultado reflete-se no atributo ute_classedesistencia . Opta-se por remover os registos que não tiverem este atributo preenchido. O atributo ute_mesesinscricao é calculado a partir da data de término, pelo que o número de valores em falta é igual.
8	Ute_mesesinscricao	4534	
9	Ute_classedesistencia	4534	

Tab. 6 – Atributos com valores em falta e opções de resolução

Relativamente aos valores em falta e à observação cruzada dos atributos, há ainda que referir:

- a existência de registos com data de termino inferior à data de inicio e com ausência de informação sobre volume de negócios;
- a existência de registos sem os atributos **ute_dtultvisita** e **ute_volnegocios** preenchidos;

Sendo a informação coerente e de acordo com as indicações de funcionamento da aplicação utilizada, e cujo significado é o de ter havido um registo de utente mas não um pagamento, conclui-se que também nestas situações os registos devem ser removidos.

Tendo em atenção todas as observações e constatações efetuadas sobre os valores em falta, desenvolveu-se o procedimento **dmLimpaUtentes**, cujas operações são descritas na [Tab. 7](#) e executadas pela ordem indicada.

Procedimento	Ordem de execução	Operações efetuadas
dmLimpaUtentes	1	Remover os registos nas seguintes condições: (ute_inicio > ute_termino AND ute_volnegocios is null) OR (ute_dtultvisita is null AND ute_volnegocios is null) OR ute_datanasc is null
	2	À data da obtenção da base de dados, o Clube 14, apesar de já ter diversas inscrições e pagamentos efetuados, não tinha ainda aberto, pelo que nestes utentes tinha-se verificado a presença de volume de negócios mas não de frequências. Por outro lado, uma vez que a inscrição obriga a produção de um Cartão de Utente com o qual se regista o acesso ou o checkin, e em várias situações o cartão não é produzido no ato da inscrição, colocou-se o atributo ute_dtultvisita=ute_inicio em todas as inscrições em que ute_dtultvisita=null AND ute_volnegocios<>null
	3	Colocar o atributo ute_naulasgrupo=0 nos registos na condição: ute_naulasgrupo is null No único clube onde não existe Checkin não são ministradas aulas de grupo, pelo que se irá colocar o atributo ute_naulasgrupo=0
	4	Colocar o atributo ute_volnegocios=0 nos registos na condição: ute_volnegocios is null muito embora este comando seja efetuado apenas por precaução, dado que após as limpezas anteriores, não se espera haver registos nestas condições.
Tab. 7 – Funções do procedimento dmLimpaUtentes		

Atendendo ao exposto, não se verificou a necessidade de se aplicar algum método mais elaborado de correção de dados, como o de imputação do valor médio, de determinação por interpolação ou regressão linear, “vizinho mais próximo” ou outros, tendo-se optado pela remoção dos registos ou pelas correções “ad-hoc” mencionadas.

Após a execução do procedimento mencionado, o sumário do “data frame” obtido em R apresenta a configuração indicada na [Fig. 4](#):

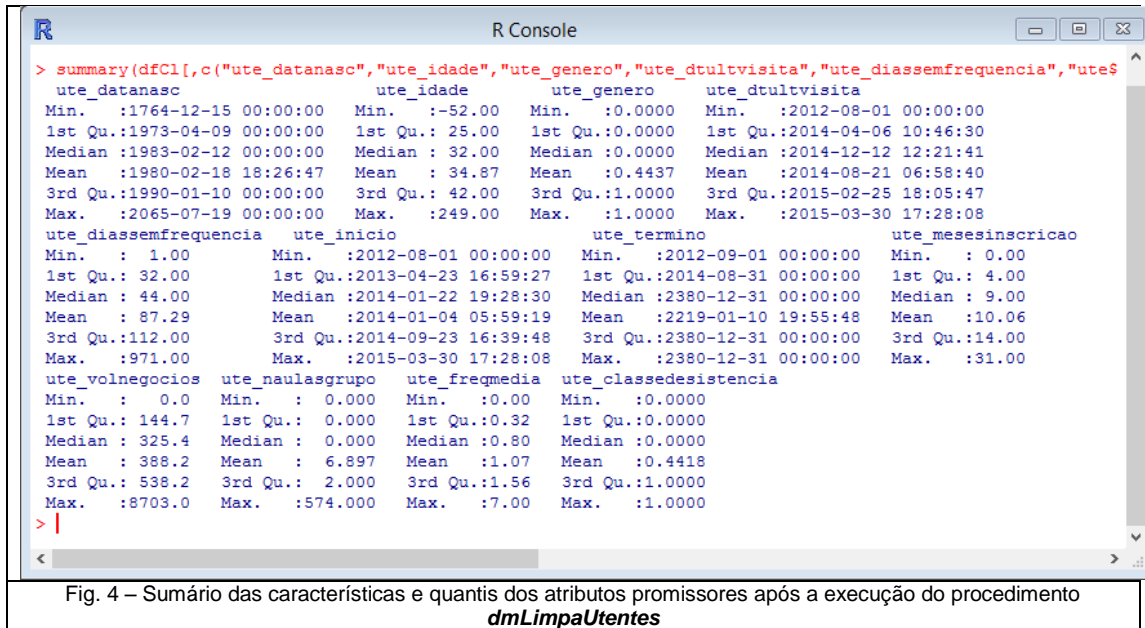


Fig. 4 – Sumário das características e quantis dos atributos promissores após a execução do procedimento *dmLimpUtentes*

Depois de tratadas as incorreções provocadas pelos valores em falta, está-se agora em condições de observar os valores com características consideravelmente diferentes (outliers).

A observação dos gráficos efetuados com a instrução **boxplot** em R para alguns dos atributos (Fig. 5), resultou nas seguintes conclusões:

- O atributo **ute_idade** contém valores errados, dado se constatar a existência de idades inferiores aos 16 – idade normalmente considerada mínima para a prática do fitness - e acima dos 93 anos - idade a partir da qual se considera difícil a prática do Fitness. Existem 332 registos nestas condições, pelo que se irá atribuir **ute_idade=NULL**¹ aos registos nestas condições;
- O atributo **ute_diassemfrecuencia** apresenta 11107 registos acima do 3ºquartil (mais de 112 dias sem frequência), sendo que 10.069 são inscrições com menos de 365 dias sem frequência, e 986 são inscrições com menos de 730 dias sem frequência (sensivelmente 2 anos). Apesar de parecerem valores fora do normal, existem muitas situações em que os utentes assinam contratos de fidelização com preços muito favoráveis – normalmente de 12 ou 24 meses – e depois acabam por frequentar algumas vezes ou mesmo nenhuma;

¹ Optou-se por colocar a idade a NULL porque não sendo o atributo ute_idade utilizado em alguns modelos de DM definidos, e não se considerando haver erros de outras espécies nos restantes atributos destes registos, não se vê necessidade em remover estes registos.

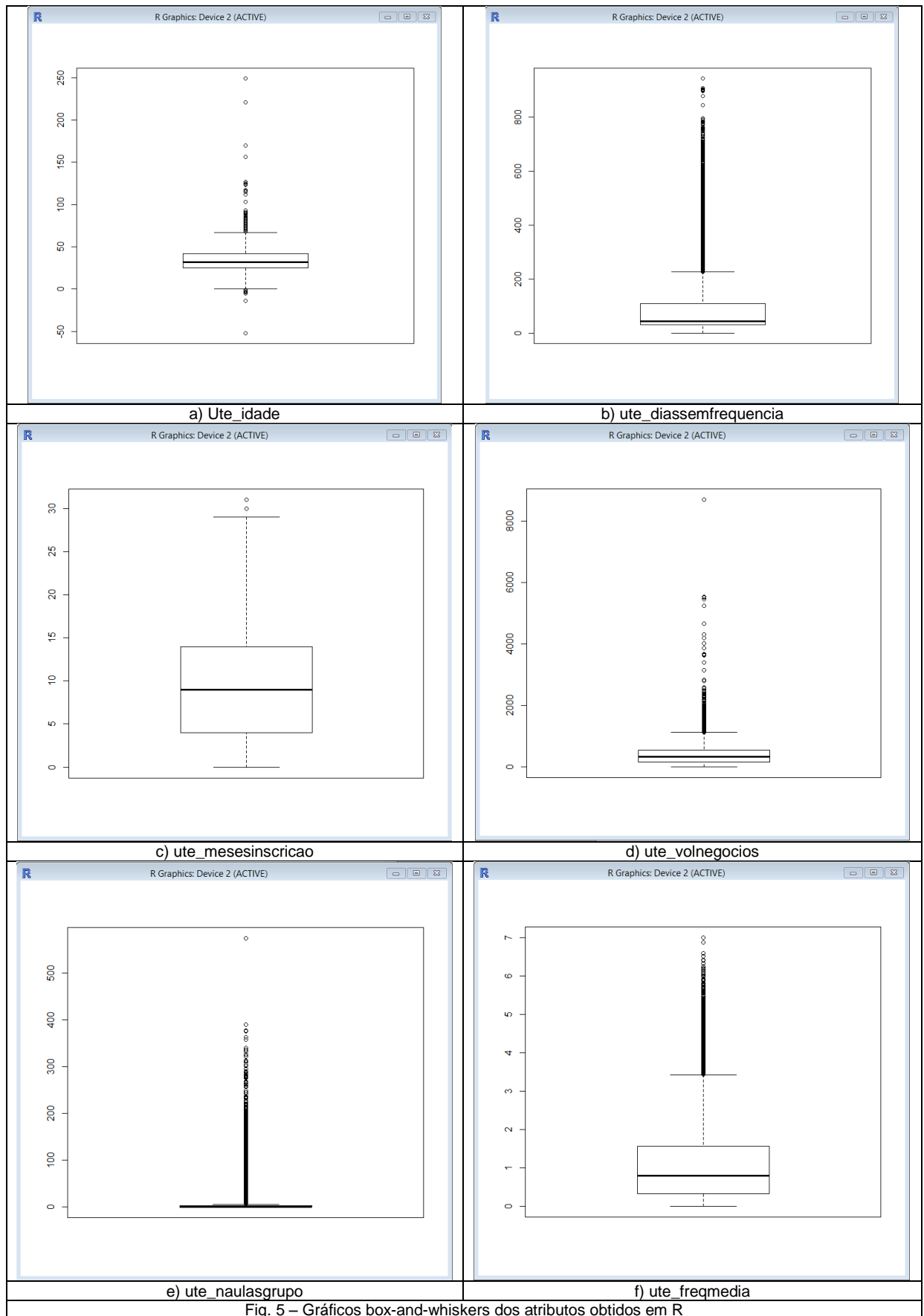


Fig. 5 – Gráficos box-and-whiskers dos atributos obtidos em R

- O atributo ***ute_mesesinscricao*** apresenta-se correto não se encontrando nenhuma duração de inscrição anômala, já que sendo considerado apenas o período entre 01 de Agosto de 2012 e 31 de Março de 2015, não pode haver inscrições com mais de 32 meses de duração, o que se verifica;

- Apesar do gráfico do atributo ***ute_volnegocios*** apresentar uma forma mais compacta revelando alguns outliers acima do 3ºquartil (538,20€), a observação dos registos levou a concluir que não há qualquer anormalidade. Os valores que aparentam ser excessivamente altos referem-se a inscrições com mais meses de inscrição. Um query aplicado na base de dados, com as cláusulas:

ute_volnegocios>538,2 and ute_volnegocios/ute_mesesinscricao>100

revelou que apenas 103 utentes são listados. Note-se que facilmente um utente que usufrui de um serviço de treino personalizado pode apresentar uma mensalidade superior a 100,00€;

- Para o atributo ***ute_naulasgrupo***, e dado que o gráfico apresenta um aspeto completamente atípico, procedeu-se de forma idêntica ao do volume de negócios, executando-se um query sobre a base de dados com as seguintes cláusulas:

ute_naulasgrupo>2 and ute_naulasgrupo/ute_mesesinscricao>28

Sendo que 2 é o valor do 3ºquartil e 28 seria o número de aulas de grupo frequentadas por mês atendendo a uma frequência de 7 aulas por semana e considerando meses de 4 semanas (note-se que estes clubes estão abertos todos os dias da semana). Obteve-se 27 registos e a observação do extrato de checkin de alguns utentes desta lista confirmou que realmente os checkins ocorreram, efetuados pelos próprios utentes nos equipamentos self-service (o que confirma a presença dos mesmos no local) e que, em algumas situações, são efetuados mais do que um checkin por dia, ou seja, alguns utentes fazem mais do que uma aula de grupo por dia. Conclui-se portanto que não há qualquer correção a efetuar neste atributo.

- Finalmente, para confirmar os valores do atributo ***ute_freqmedia*** que apresenta ainda um número razoável de outliers, determinou-se o número de utentes que têm médias de frequência superiores a 3 vezes por semana, apresentado na [Tab. 8](#):

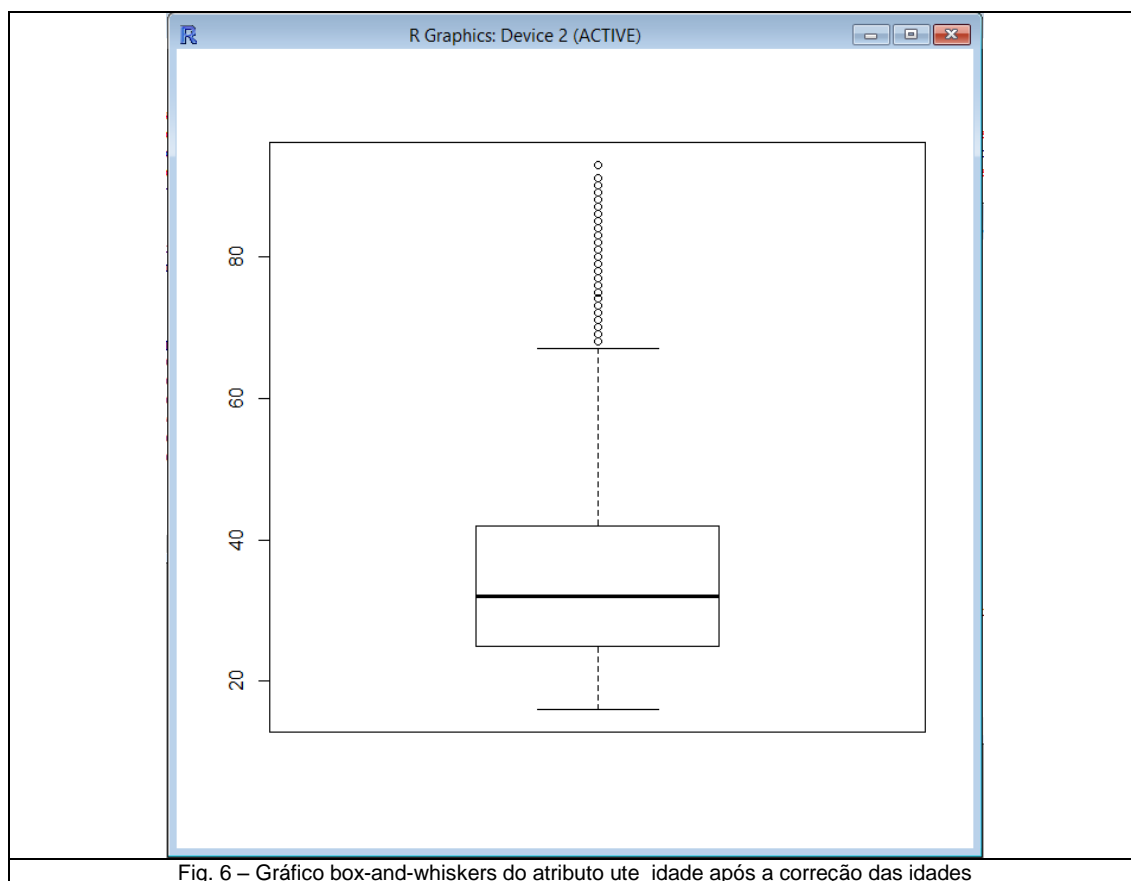
Média de frequência	Nº de Utentes
>3	1754
>4	560
>5	125
>6	29
>=7	1

Tab. 8 – Média de frequência semanal do Ginásio superior a 3/semana

Note-se que este atributo só contabiliza, no máximo, uma frequência por dia, pelo que também aqui não se deteta nenhuma anormalidade e não haverá, consequentemente nenhuma correção a fazer.

Analizadas as questões relacionadas com os valores dos atributos, implementou-se e executou-se o procedimento **dmCorrigelidades** para corrigir o único atributo que apresenta valores indevidos (**ute_idade**).

Após as correções, o gráfico executado sobre o atributo **ute_idade** apresenta agora a configuração apresentada na [Fig. 6](#).



Os histogramas de cada atributo, após correções, são apresentados na [Fig. 7](#).

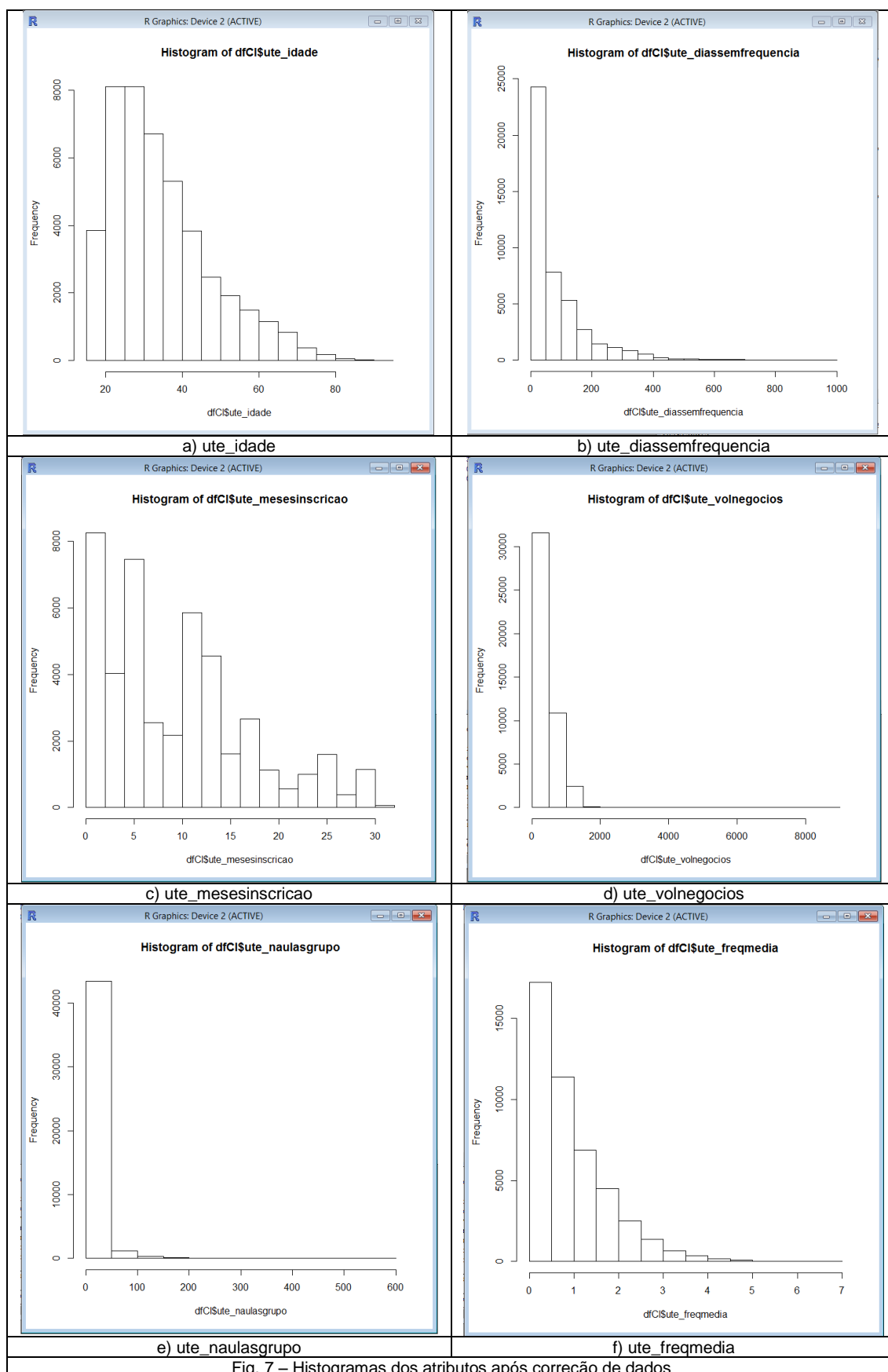


Fig. 7 – Histogramas dos atributos após correção de dados

Atendendo à dispersão observada em alguns atributos e à concentração de valores noutros, o que pode complicar não só a performance, como também a observação do resultado dos modelos, optou-se por criar classificações, criando novos atributos que resultam da classificação dos anteriores. Este processo, como se verá à frente, transforma o tipo dos atributos classificados (atributos contínuos antes da classificação, e discretos após a classificação).

Para cada um dos atributos em questão utilizou-se uma técnica semelhante á de Hughes (2015), ordenando cada um dos atributos e agrupando os registos em grupos de 1 a 5.

O processo de classificação foi codificado nos procedimentos da base de dados indicados na [Tab. 9](#) e executados.

Ordem de execução	Procedimento	Atributo preenchido na tabela dmRetencao
1	dmClassificaDiasSemFrequencia	ute_cdiassemfrequencia
2	dmClassificaMesesDaInscricao	ute_cmeseinscricao
3	dmClassificaVolNegocios	ute_cvolnegocios
4	dmClassificaNAulasGrupo	ute_cnaulasgrupo
5	dmClassificaFreqMedia	ute_cfreqmedia

Tab. 9 – Procedimentos de classificação de atributos

Tendo sido detetadas e corrigidas as anomalias nos dados, está-se em condições de avançar para o teste dos modelos. Atendendo ao número de registos final na tabela **dmRetencao** (44.768), determina-se que serão utilizados 70% dos registos para treino dos modelos, e os restantes 30% para teste. Em algumas situações de teste, embora escolhidos aleatoriamente, os registos de treino e teste serão mantidos para que se possam estabelecer comparações. Esta situação será referenciada sempre que se verificar.

4.4 Modelação

Sobre os atributos a utilizar

A título de resumo e introdução para a modelação, como resultado dos passos anteriores resultaram os atributos promissores apresentados na [Tab. 10](#).

Atributo	Tipo	Observações
Ute_idsaft		Atributo-chave de identificação única do registo
Ute_idade	Contínuo	
Ute_genero	Discreto	
Ute_diassemfrequencia	Contínuo	
Ute_cdiassemfrequencia	Discreto	Classificação de 1 a 5 do atributo ute_diassemfrequencia
Ute_mesesinscricao	Contínuo	
Ute_cmeseinscricao	Discreto	Classificação de 1 a 5 do atributo ute_mesesinscricao
Ute_volnegocios	Contínuo	
Ute_cvolnegocios	Discreto	Classificação de 1 a 5 do atributo ute_volnegocios

Atributo	Tipo	Observações
Ute_naulasgrupo	Contínuo	
Ute_cnaulasgrupo	Discreto	Classificação de 1 a 5 do atributo ute_naulasgrupo
Ute_freqmedia	Contínuo	
Ute_cfreqmedia	Discreto	Classificação de 1 a 5 do atributo ute_freqmedia
Ute_classedesistencia	Discreto	Atributo que se pretende prever

Tab. 10 – Tipo dos atributos promissores

Analisou-se a correlação entre os atributos, quer dos contínuos, quer dos “discretizados”, que se apresenta nas [Fig. 8](#) e [9](#) – análise numérica das correlações, e [Fig. 10](#) e [11](#) – análise gráfica das correlações.

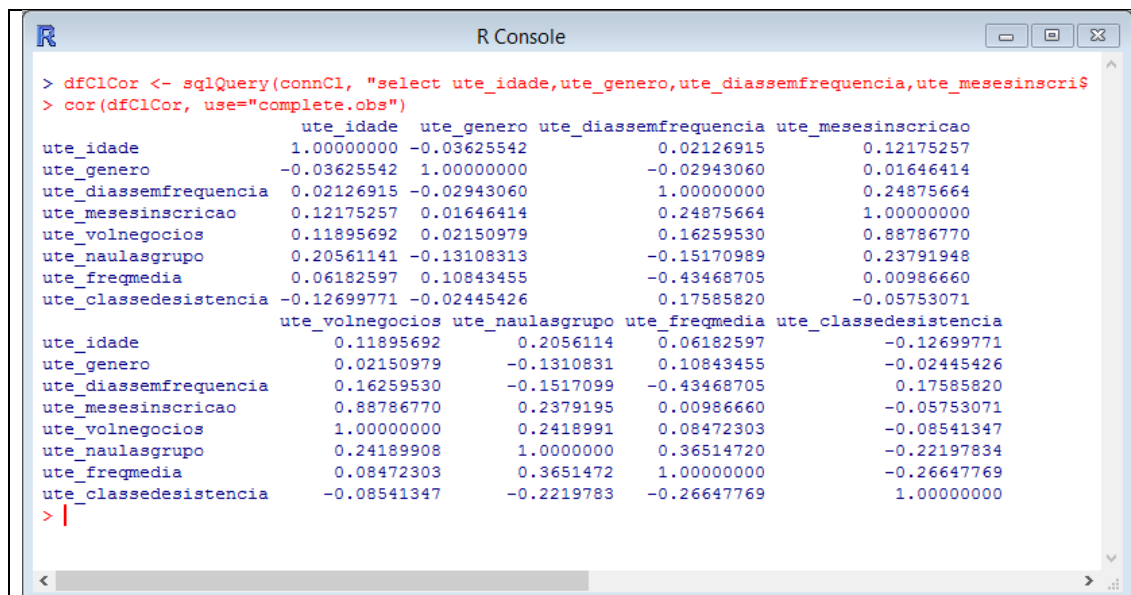


Fig. 8 - Correlação dos atributos contínuos obtida em R

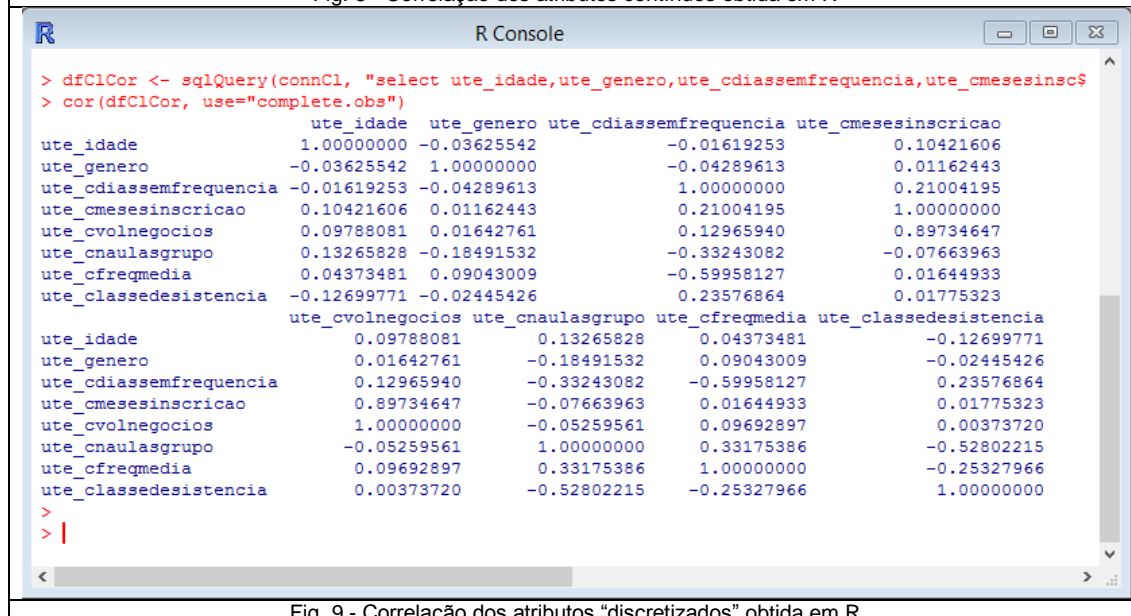
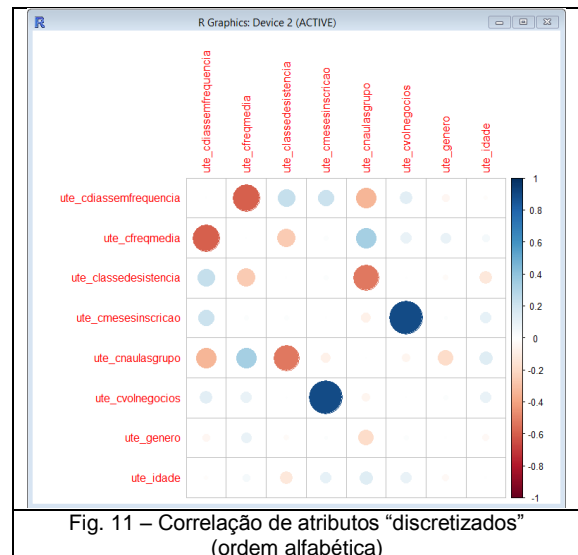
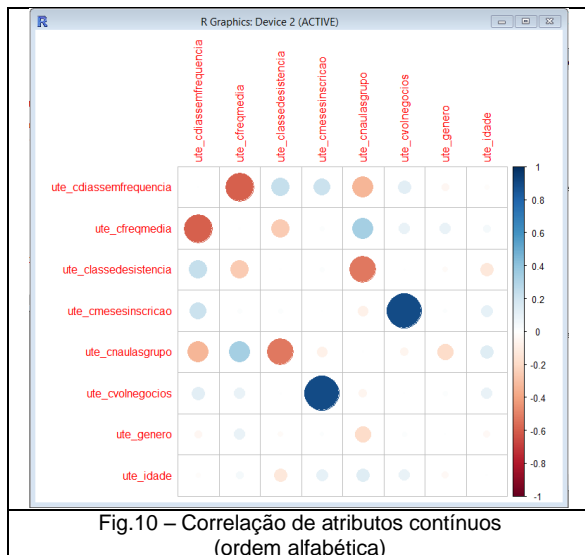


Fig. 9 - Correlação dos atributos “discretizados” obtida em R



Sobre a análise da correlação, algumas conclusões se podem tirar de imediato:

- Algumas correlações sofrem alterações significativas quando os atributos são “discretizados”;
- Há uma correlação elevada entre o número de meses da inscrição e o volume de negócios, o que é normal dado que havendo um pagamento regular periódico, quando maior for a duração da inscrição, maior será o volume de negócios;
- Há uma correlação elevada entre o número de aulas de grupo e a frequência média, o que é um fato conhecido no “negócio do Fitness”;
- Há uma correlação “negativa” relativamente forte entre o número de aulas de grupo e a desistência;
- Há uma correlação “negativa” relativamente forte entre os dias sem frequência e a frequência média;
- Há uma correlação forte entre o atributo referente ao número de dias sem frequência e a desistência, o que é naturalmente uma evidência.

Estas observações tornam-se importantes uma vez que se poderão reduzir o número de atributos a aplicar nos modelos, nomeadamente utilizando apenas um dos atributos, quando dois apresentam correlações muito fortes.

Sobre os algoritmos a utilizar e procedimentos a efetuar

Optou-se pela utilização de algoritmos de Classificação, uma vez que de acordo com Tan (2005), as técnicas de Classificação são adequadas a previsão ou

descrição de conjunto de dados com categorias binárias ou nominais, o que se enquadra com o objetivo proposto em que se pretende determinar se o utente virá a ser desistente ou não (categoria binária).

Optou-se por dividir a fase de aplicação dos modelos em 3 partes:

- Numa primeira fase são aplicados os algoritmos de classificação disponibilizados pelo MS SQL AS (Microsoft Decision Trees, Logistic Regression e Neural Network)² para observar, através das informações disponibilizadas, os resultados sobre as várias combinações de atributos no sentido de perceber qual o “melhor” algoritmo a aplicar;
- Numa segunda fase, aplica-se alguns algoritmos disponibilizados no R para obter outras métricas e estabelecer comparações com os resultados obtidos no MS SQL AS.

Em qualquer das situações, além da utilização de informação apresentada em cada sistema, pretende-se comparar os resultados através da elaboração da matriz de confusão - elaborada com a aplicação do modelo ao conjunto de teste -, e determinados os indicadores apresentados na [Tab. 11](#).

Indicador	Fórmula
Taxa de Precisão (Accuracy)	$(TP + TN) / n$
Taxa de Erro	$(FP + FN) / n$
Taxa de Verdadeiros Positivos (Recall)	$TP / (TP + FN)$
Taxa de Falsos positivos	$FP / (FP + TN)$
Taxa de Precisão (Precision)	$TP / (TP + FP)$
Kappa	$K_e = [(TN + FN) * (TN + FP) + (FP + TP) * (FN + TP)] / n^2$ $K_o = (TN + TP) / n$ $K = [K_o - K_e] / [1 - K_e]$
Legenda: TP – True positive (verdadeiro positivo) – valor corretamente assinalado como positivo (desistente) TN – True negative (verdadeiro negativo) – valor corretamente assinalado como negativo (não desistente) FP – False positive (falso positivo) – valor incorretamente assinalado como positivo (falso desistente) FN – False negative (falso negativo) – valor incorretamente assinalado como negativo (falso não desistente)	

Tab. 11 – Fórmulas para a determinação da Taxa de exatidão e da Taxa de erro

- Finalmente, na última fase, são apresentadas as conclusões finais sobre os algoritmos e atributos escolhidos, e outros fatores ainda não abordados.

Aplicação dos algoritmos disponibilizados pelo Analysis Services

Atendendo às observações efetuadas sobre os atributos, e tendo em atenção que o atributo a prever é sempre o mesmo (**ute_classedesistencia**) foram criados os modelos baseados nos atributos apresentados na [Tab. 12](#).

² O MS SQL AS disponibiliza ainda o algoritmo de Classificação Naive Bayes, mas não permite a sua utilização devido aos atributos de entrada a utilizar.

	Modelo	Atributos utilizados	Algoritmos aplicados
Métodos aplicados sobre os atributos não classificados			
1 2 3	Dm Retencao FM	Ute_freqmedia, ute_volnegocios	Decision Tree Logistic Regression Neural Network
4 5 6	Dm Retencao RFM	Ute_diassemfrequencia, ute_freqmedia, ute_volnegocios	Decision Tree Logistic Regression Neural Network
7 8 9	Dm Retencao All	ute_diassemfrequencia, ute_freqmedia, ute_mesesinscricao, ute_naulasgrupo, ute_volnegocios, ute_genero, ute_idade	Decision Tree Logistic Regression Neural Network
10	Dm Retencao All-Genero	ute_diassemfrequencia, ute_freqmedia, ute_mesesinscricao, ute_naulasgrupo, ute_volnegocios, ute_idade	Decision Tree
Métodos aplicados sobre atributos classificados			
11 12 13	Dm Retencao FMClass	Ute_cfrequencia, ute_cvolnegocios	Decision Tree Logistic Regression Neural Network
14 15 16	Dm Retencao RFMClass	Ute_cdiassemfrequencia, ute_cfrequencia, ute_cvolnegocios	Decision Tree Logistic Regression Neural Network
17 18 19	Dm Retencao AllClass	ute_cdiassemfrequencia, ute_cfrequencia, ute_mesesinscricao, ute_cnaulasgrupo, ute_cvolnegocios, ute_genero, ute_idade	Decision Tree Logistic Regression Neural Network
20	Dm retencao AllClass-DiasSem	ute_cfrequencia, ute_mesesinscricao, ute_cnaulasgrupo, ute_cvolnegocios, ute_genero, ute_idade	Decision Tree
21	Dm Retencao AllClass-Genero	ute_cdiassemfrequencia, ute_cfrequencia, ute_mesesinscricao, ute_cnaulasgrupo, ute_cvolnegocios, ute_idade	Decision Tree
22 23 24	Dm Retencao AllClass-GeneroIdade	ute_cdiassemfrequencia, ute_cfrequencia, ute_mesesinscricao, ute_cnaulasgrupo, ute_cvolnegocios	Decision Tree Logistic Regression Neural Network
25	Dm Retencao AllClass-GeneroIdadeVolNegocios	ute_cdiassemfrequencia, ute_cfrequencia, ute_mesesinscricao, ute_cnaulasgrupo	Decision Tree
Tab. 12 – Modelos criados, atributos e algoritmos utilizados com melhores resultados			

A [Tab. 13](#) apresenta as matrizes de confusão obtidas em cada método e os indicadores obtidos a partir das mesmas.

Na [Tab. 13](#) observa-se que:

- Para o caso em questão, o algoritmo Microsoft Decision Tree, qualquer que seja o conjunto de atributos escolhido (modelos 1, 4, 7, 11, 14, 17 e 22), obtém sempre melhores taxas de precisão (accuracy), e consequentemente menores taxas de erro, que os algoritmos Microsoft Logistic Regression (modelos 2, 5, 8, 12, 15, 18 e 23) e Microsoft Neural Network (modelos 3, 6, 9, 13, 16, 19 e 24);
- Os modelos FM e RFM baseados em Microsoft Decision Tree que utilizam apenas os atributos não classificados (modelo 1 com atributos **ute_freqmedia**, **ute_volnegocios**, e modelo 4 com atributos **ute_diassemfrequencia**, **ute_freqmedia**, **ute_volnegocios**) apresentam melhores taxas de precisão (accuracy) que os respetivos

modelos que utilizam atributos classificados (modelos 11 e 14 respetivamente);

		Confusion Matrix			Accuracy Rate	True Positive Rate	Precision
		Predicted	TRUE	FALSE	Error Rate	False Positive Rate	Kappa
Dm Retencao FM	1 Decision Tree	TRUE	3086	1373	68,34%	51,74%	69,21%
		FALSE	2879	6092	31,66%	18,39%	34,21%
	2 Logistic Regression	TRUE	5143	5289	54,50%	86,22%	49,30%
		FALSE	822	2176	45,50%	70,85%	14,29%
	3 Neural Network	TRUE	5003	5059	55,17%	83,87%	49,72%
		FALSE	962	2406	44,83%	67,77%	15,06%
Dm Retencao RFM	4 Decision Tree	TRUE	4550	1699	77,51%	77,25%	72,81%
		FALSE	1340	5922	22,49%	22,29%	54,58%
	5 Logistic Regression	TRUE	5146	5306	55,22%	87,37%	49,23%
		FALSE	744	2315	44,78%	69,62%	16,31%
	6 Neural Network	TRUE	5106	5005	57,15%	86,69%	50,50%
		FALSE	784	2616	42,85%	65,67%	19,43%
Dm Retencao All	7 Decision Tree	TRUE	4744	1976	76,73%	80,50%	70,60%
		FALSE	1149	5561	23,27%	26,22%	53,47%
	8 Logistic Regression	TRUE	5199	5115	56,75%	88,22%	50,41%
		FALSE	694	2422	43,25%	67,87%	18,82%
	9 Neural Network	TRUE	5079	4922	57,29%	86,19%	50,78%
		FALSE	814	2615	42,71%	65,30%	19,41%
Dm Retencao All-Genero	10 Decision Tree	TRUE	4395	1642	76,54%	74,44%	72,80%
		FALSE	1509	5884	23,46%	21,82%	52,50%
Dm Retencao FMClass	11 Decision Tree	TRUE	3586	2082	66,39%	59,59%	63,27%
		FALSE	2432	5330	33,61%	28,09%	31,67%
	12 Logistic Regression	TRUE	4864	4907	54,87%	80,82%	49,78%
		FALSE	1154	2505	45,13%	66,20%	13,81%
	13 Neural Network	TRUE	4864	4907	54,87%	80,82%	49,78%
		FALSE	1154	2505	45,13%	66,20%	13,81%
Dm Retencao RFMClass	14 Decision Tree	TRUE	4001	1612	73,26%	66,91%	71,28%
		FALSE	1979	5838	26,74%	21,64%	45,54%
	15 Logistic Regression	TRUE	5160	3855	65,19%	86,29%	57,24%
		FALSE	820	3595	34,81%	51,74%	32,90%
	16 Neural Network	TRUE	5167	4375	61,37%	86,40%	54,15%
		FALSE	813	3075	38,63%	58,72%	26,14%
Dm Retencao AllClass	17 Decision Tree	TRUE	4626	832	84,38%	78,51%	84,76%
		FALSE	1266	6706	15,62%	11,04%	68,02%
	18 Logistic Regression	TRUE	5259	3389	70,05%	89,26%	60,81%
		FALSE	633	4149	29,95%	44,96%	42,15%
	19 Neural Network	TRUE	5306	3450	69,95%	90,05%	60,60%
		FALSE	586	4088	30,05%	45,77%	42,05%
Dm Retencao AllClass-DiasSem	20 Decision Tree	TRUE	4257	828	81,19%	71,49%	83,72%
		FALSE	1698	6647	18,81%	11,08%	61,32%
Dm Retencao AllClass-Genero	21 Decision Tree	TRUE	4631	792	84,39%	78,02%	85,40%
		FALSE	1305	6702	15,61%	10,57%	68,06%
Dm Retencao AllClass- GeneroIdade	22 Decision Tree	TRUE	4698	859	84,38%	79,13%	84,54%
		FALSE	1239	6634	15,62%	11,46%	68,12%
	23 Logistic Regression	TRUE	5293	3492	69,20%	89,15%	60,25%
		FALSE	644	4001	30,80%	46,60%	40,53%
	24 Neural Network	TRUE	5328	3467	69,65%	89,74%	60,58%
		FALSE	609	4026	30,35%	46,27%	41,40%
Dm Retencao AllClass-GeneroIdadeVolNegocios	25 Decision Tree	TRUE	4695	938	83,78%	79,11%	83,35%
		FALSE	1240	6557	16,22%	12,52%	66,95%

Tab. 13 – Modelos criados, matrizes de confusão e indicadores calculados em Análisis Servicos

Tab. 13 – Modelos criados, matrizes de confusão e indicadores calculados em Analysis Services

- c) O modelo baseado em Microsoft Decision Tree que utiliza todos os atributos classificados (modelo 17) apresenta um incremento substancial na taxa de precisão (accuracy=84,38%) relativamente ao modelo que utiliza os atributos não classificados (modelo 7 com accuracy=76,73%);
- d) A não inclusão dos atributos **ute_genero** (modelo 21) e **ute_genero** e **ute_idade** (modelo 22) praticamente não altera a taxa de precisão (accuracy) relativamente ao modelo que inclui estes dois atributos (igual no modelo sem os 2 atributos, e ligeiramente melhor - +0,01 - no modelo que exclui apenas o atributo **ute_genero**). No entanto, a taxa de verdadeiros positivos tem um aumento de 0,62% no caso do modelo que não inclui os atributos **ute_genero** e **ute_idade** (modelo 22);

- e) O indicador Kappa reflete muito ligeiramente a não inclusão do atributo **ute_genero** no modelo, melhorando o indicador de 68,02% para 68,06% no caso do modelo que não utiliza o atributo **ute_genero** (modelo 21) e para 68,12% no modelo que não utiliza o atributo **ute_genero** e **ute_idade** (modelo 22);

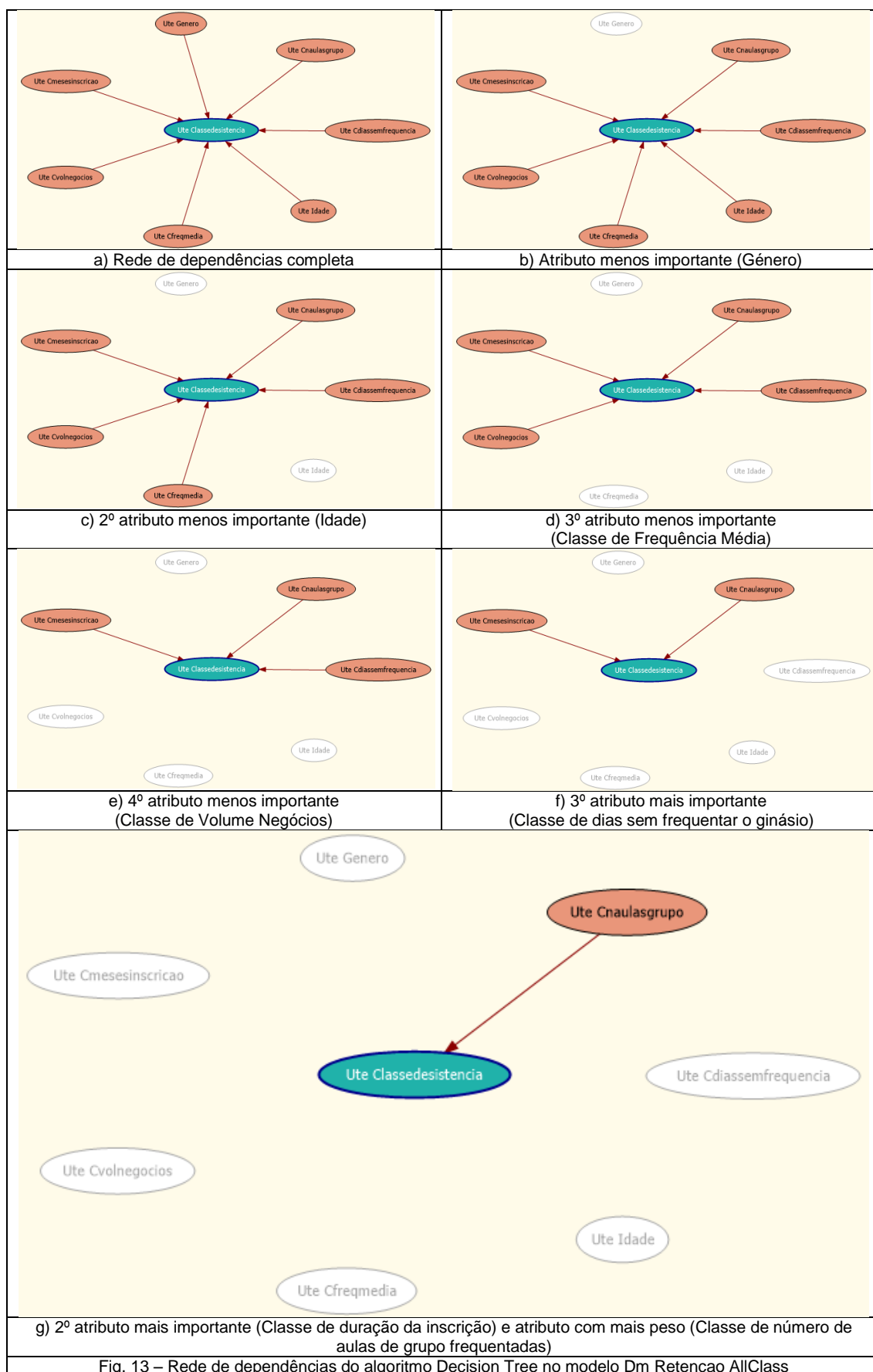
O modelo Dm Retencao AllClass utiliza os atributos demográficos referentes ao género e idade do utente. No entanto, a observação da árvore de decisão criada pelo algoritmo ([Fig. 12](#)) apresenta a utilização do atributo **ute_genero** pontualmente em alguns nós que incluem um número reduzido de registos, e embora apresente uma taxa de precisão idêntica (84,38%), tal deve-se ao fato de o modelo apresentar um maior número de negativos verdadeiros, que compensa a menor identificação de verdadeiros positivos. Recorda-se que o objetivo é identificar potenciais desistentes, pelo que, em caso de taxas de precisão idênticas se deve sempre privilegiar modelos que apresentem um maior número de verdadeiros positivos.

Por outro lado, a rede de dependências ([Fig. 13](#)) aponta os atributos demográficos **ute_genero** e **ute_idade** como os que menos contribuem para a performance do algoritmo.

Na secção [Conclusões finais sobre algoritmos e atributos escolhidos](#) são especificados outros testes efetuados para despiste da ocorrência de overfitting no modelo Dm Retencao AllClass.



Fig. 12 – Árvore de decisão do modelo 17 (DM Retencao AllClass)



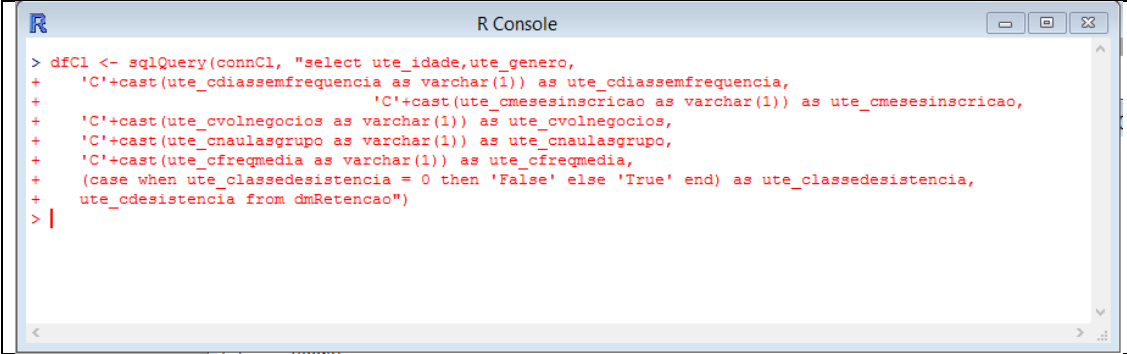
Aplicação dos algoritmos disponibilizados no R

Para implementar os modelos, tal como já tinha sido efetuado na fase de [Preparação dos Dados](#), utilizou-se a conexão do R à base de dados PROJFINAL que contém a tabela com os atributos contínuos e os atributos discretizados (classificados), conseguindo-se assim obter modelos criados sobre os mesmos dados que os modelos implementados em MS SQL AS.

Os dados da tabela **dmRetencao** são carregados num dataframe (dfCl), que por sua vez é desdobrado em dois: um para treino e outro para teste dos modelos.

Uma vez que os atributos **ute_classedesistencia** (o que se pretende prever) é do tipo bit, e os atributos classificados/discretizados do tipo inteiro são considerados contínuos pelos packages utilizados, houve a necessidade de converter os atributos para um tipo que os packages considerassem para classificação. Para resolver esta questão, o query de carregamento do “dataframe” foi alterado, acrescentando o carácter “C” à esquerda de cada classificação, e convertendo o atributo de desistência em False ou True, conforme apresente o valor 0 ou 1, respetivamente.

O query é apresentado na [Fig. 14](#).



```
> dfCl <- sqlQuery(connCl, "select ute_idade,ute_genero,
+ 'C'+cast(ute_cdiasssefrecuencia as varchar(1)) as ute_cdiasssefrecuencia,
+ 'C'+cast(ute_cmeseinscricao as varchar(1)) as ute_cmeseinscricao,
+ 'C'+cast(ute_cvolnegocios as varchar(1)) as ute_cvolnegocios,
+ 'C'+cast(ute_cnaulasgrupo as varchar(1)) as ute_cnaulasgrupo,
+ 'C'+cast(ute_cfregmedia as varchar(1)) as ute_cfregmedia,
+ (case when ute_classedesistencia = 0 then 'False' else 'True' end) as ute_classedesistencia,
+ ute_cdesistencia from dmRetencao")
> |
```

Fig. 14 – Query de carregamento do “dataframe” com conversão dos atributos para string

Apos o carregamento, são criados os “dataframe” de **treino** – que contém 70% dos dados -, e o de **teste** – que contém os restantes 30% -, estabelecendo-se assim condições de implementação dos modelos idênticas às implementadas no MS SQL AS.

A [Fig. 15](#) ilustra a preparação dos “dataframes” de treino e de teste em ambiente R para a criação dos modelos, conforme descrita nos parágrafos anteriores:

```

> #### carregamento dos conjuntos de Treino e Teste
> #define a semente
> set.seed(1234)
> #cria vetor com 1 ou 2
> tipo <- sample(2, nrow(dfCl), replace = TRUE, prob = c(0.7, 0.3))
> #carrega data frame de treino
> treino <- dfCl[tipo == 1, ]
> nrow(treino)
[1] 31355
> #carrega data frame de teste
> teste <- dfCl[tipo == 2, ]
> nrow(teste)
[1] 13413
> |

```

Fig. 15 – Carregamento dos “dataframes” de treino e teste

Apos a construção dos modelos em MS SQL AS e as observações que se fizeram optou-se por implementar em R apenas os modelos que se verificaram ser mais eficientes, quer em termos de algoritmo utilizado (Decision Tree), quer no que diz respeito aos atributos a utilizar. Por estes motivos, utilizaram-se três packages conhecidas do Sistema R: tree, party e rpart.

Executando os algoritmos para cada conjunto de atributos, obtém-se as matrizes de confusão e respetivos indicadores apresentados na [Tab. 14](#).

		Confusion Matrix			Accuracy Rate	True Positive Rate	Precision
		Predicted	Actual		Error Rate	False Positive Rate	Kappa
ute_cdiassfreqencia+ ute_cfrequencia+ ute_cmeseinscricao+ ute_cnaulasgrupo+ ute_cvolnegocios+ ute_genero+ ute_idade	26 tree	TRUE	4167	1786	82,24%	87,49%	70,00%
		FALSE	596	6864	17,76%	20,65%	63,29%
	27 ctrees	TRUE	4654	1299	84,39%	85,41%	78,18%
		FALSE	795	6665	15,61%	16,31%	68,10%
	28 rpart	TRUE	4142	1811	81,83%	86,87%	69,58%
		FALSE	626	6834	18,17%	20,95%	62,44%
ute_cdiassfreqencia+ ute_cfrequencia+ ute_cmeseinscricao+ ute_cnaulasgrupo+ ute_cvolnegocios+ ute_idade	29 tree	TRUE	4167	1786	82,24%	87,49%	70,00%
		FALSE	596	6864	17,76%	20,65%	63,29%
	30 ctrees	TRUE	4651	1302	84,40%	85,46%	78,13%
		FALSE	791	6669	15,60%	16,33%	68,12%
	31 rpart	TRUE	4142	1811	81,83%	86,87%	69,58%
		FALSE	626	6834	18,17%	20,95%	62,44%
ute_cdiassfreqencia+ ute_cfrequencia+ ute_cmeseinscricao+ ute_cnaulasgrupo+ ute_cvolnegocios	32 tree	TRUE	4167	1786	82,24%	87,49%	70,00%
		FALSE	596	6864	17,76%	20,65%	63,29%
	33 ctrees	TRUE	4570	1383	84,21%	86,15%	76,77%
		FALSE	735	6725	15,79%	17,06%	67,66%
	34 rpart	TRUE	4142	1811	81,83%	86,87%	69,58%
		FALSE	626	6834	18,17%	20,95%	62,44%

Tab. 14 – Modelos criados, matrizes de confusão e indicadores calculados em Sistema R

No que diz respeito à análise da [Tab. 14](#), realçam os seguintes fatos:

- Não há qualquer alteração de valores na matriz de confusão da aplicação do algoritmo Decision Tree do Package **tree** para os três conjuntos de atributos (modelos 26, 29 e 32), o que é facilmente compreensível pela

observação da árvore de decisão gerada pelo algoritmo (Fig. 16), que não faz uso dos referidos atributos;

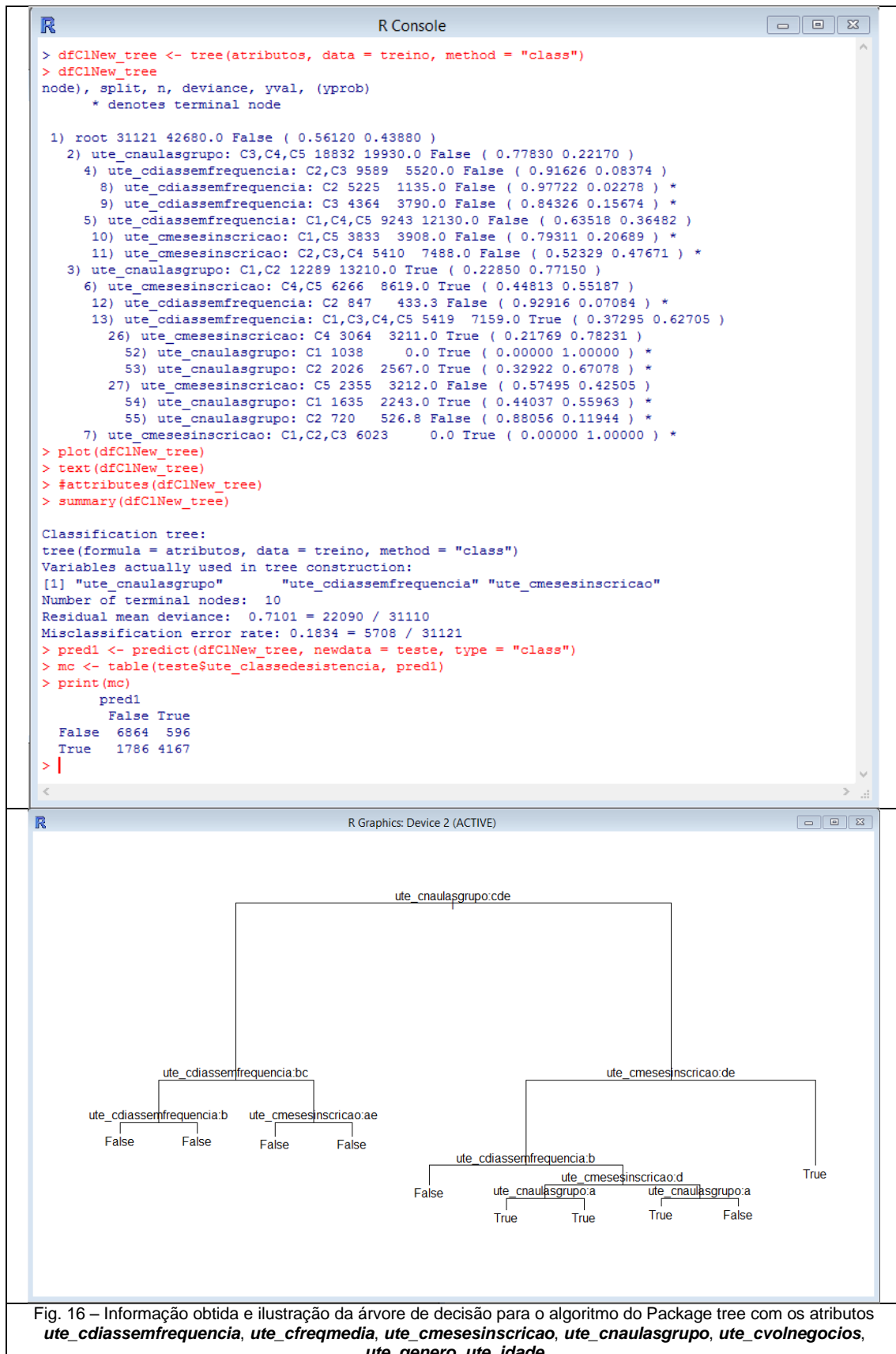
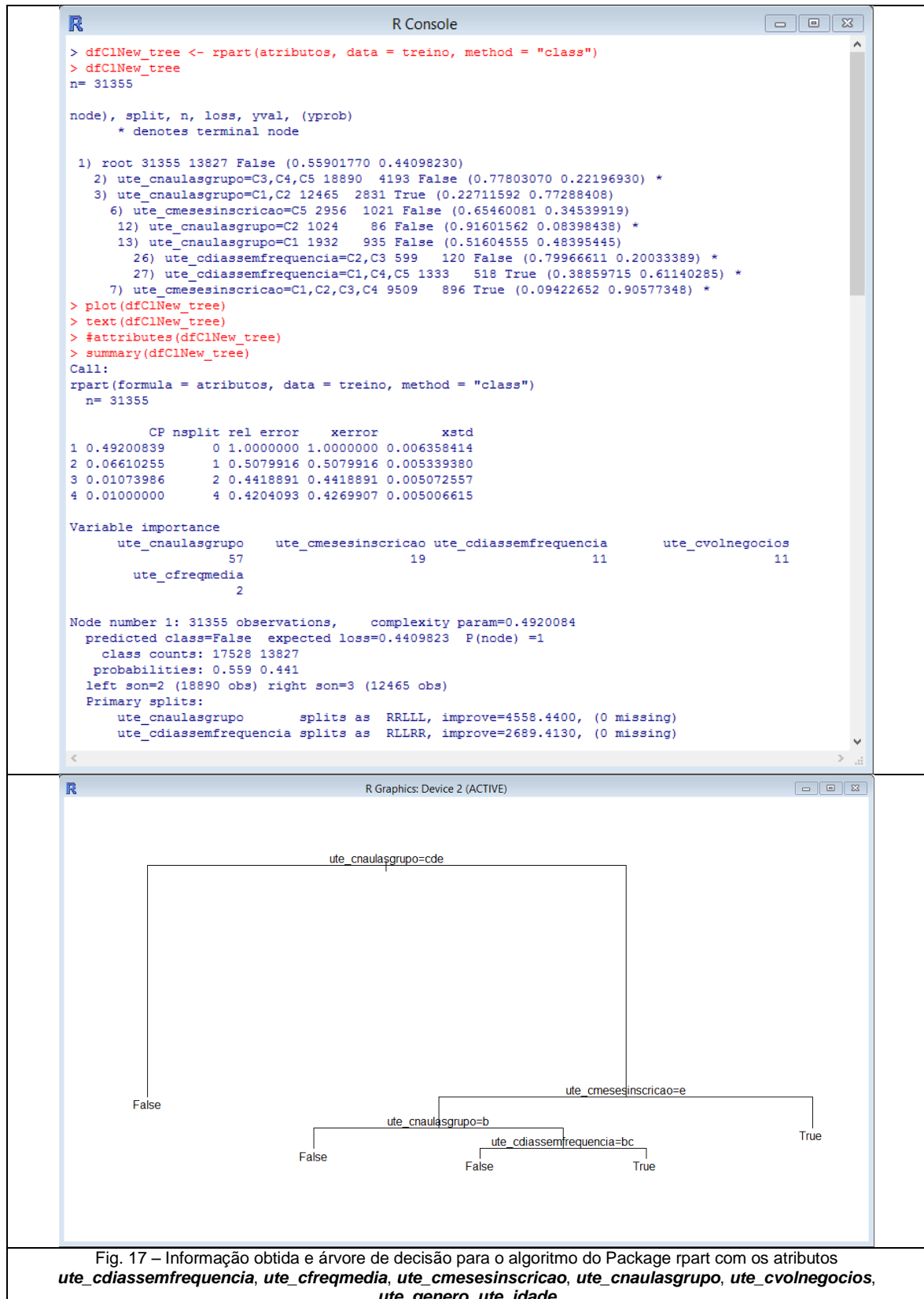


Fig. 16 – Informação obtida e ilustração da árvore de decisão para o algoritmo do Package tree com os atributos *ute_cdiassemfrequencia*, *ute_cfrequencia*, *ute_cmeseinscricao*, *ute_cnaulasgrupo*, *ute_cvolnegocios*, *ute_genero*, *ute_idade*

- b) O mesmo acontece para o algoritmo Decision Tree do Package **rpart** nos modelos 28, 31 e 34 (Fig. 17), que também não faz uso dos atributos **ute_genero** e **ute_idade**;



- c) Ao contrário do que acontece nos modelos em MS SQL AS (modelos 17, 21 e 22), os modelos **ctree** correspondentes (modelos 27, 30 e 33) melhoram ligeiramente a taxa de precisão (accuracy) quando se retira o primeiro campo demográfico (**ute_genero**) e ambos os campos demográficos do modelo (**ute_genero** e **ute_idade**).
- d) Tal como acontece nos modelos em MS SQL AS (modelos 17, 21 e 22), o indicador Taxa de Verdadeiros Positivos melhora quando o modelo não inclui os 2 atributos (modelos 27, 30 e 33);
- e) Ao contrário dos outros modelos em R, o modelo ctree faz uso de todos os atributos indicados na construção da árvore ([Fig. 18](#)).



Fig. 18 – Partes da árvore de decisão para o algoritmo do Package party (**ctree**) onde se verifica a utilização de todos os atributos

- f) O indicador Kappa apresenta o seu melhor valor no modelo que não inclui o campo **ute_genero** (modelo 30) e pior no modelo que não inclui os dois atributos demográficos (modelo 33).

Conclusões finais sobre algoritmos e atributos escolhidos

Não perdendo de vista que o objetivo principal é determinar um padrão de desistência (positivos verdadeiros) tendo em atenção que se pretende tomar medidas de gestão que diminuam as desistências, e que é preferível apresentar um modelo que tendencialmente privilegie os positivos (utentes desistentes), ainda que sejam falsos positivos, em detrimento de uma maior precisão a nível da determinação dos negativos verdadeiros (não desistentes), os resultados obtidos em MS SQL AS permitem concluir que o algoritmo Decision Tree é aquele que apresenta melhores resultados nos indicadores observados, uma vez que apresenta melhores taxas de precisão (accuracy) e menores taxas de erro, em qualquer conjunto de atributos analisado.

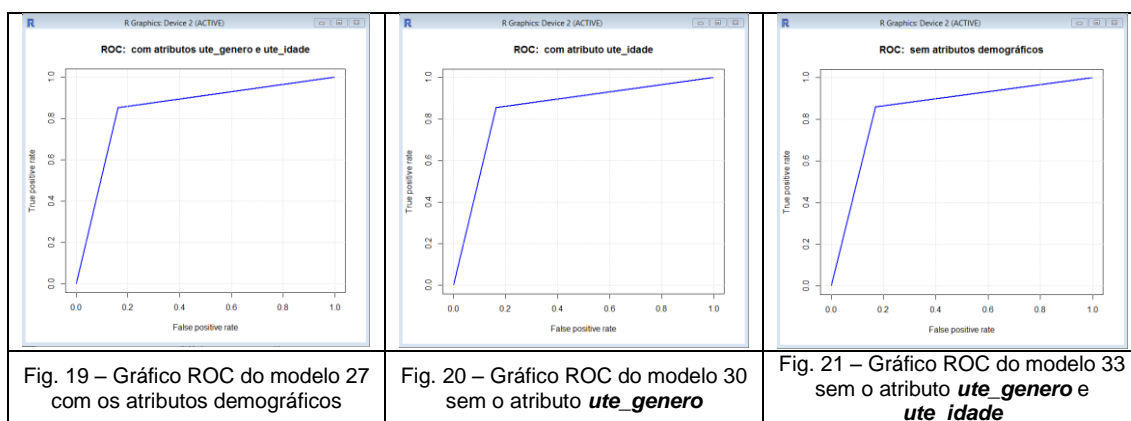
Já no que concerne aos atributos a utilizar, a observação da [Tab. 13](#) permite numa primeira abordagem identificar que os melhores modelos são os que utilizam os atributos classificados (modelos 11 a 34).

Por outro lado, sobre os modelos que utilizam os atributos classificados, os que apresentam melhores resultados são os que combinam os atributos **ute_cdiassemfrequencia**, **ute_cfreqmedia**, **ute_cmesesinscricao**, **ute_cnaulasgrupo**, **ute_cvolnegocios** e os atributos demográficos **ute_genero** e **ute_idade** (modelos 17 a 34).

Sobre estes últimos interessa avaliar algumas questões com maior pormenor.

Os algoritmos que utilizam os atributos **ute_genero** e **ute_idade** (modelo 17 em MS SQL AS Decision Tree com taxa de verdadeiros positivos de 78,51% e modelo 27 do Package party (**ctree**) em R com taxa de verdadeiros positivos de 85,41%) apresentam uma taxa de verdadeiros positivos (alguns autores designam por Sensibilidade) inferior aos modelos sem os referidos atributos - 79,13% e 86,15% respetivamente.

Comparando as taxas de Verdadeiros Positivos com a taxa de Falsos Positivos dos 3 modelos em R (ver [Tab. 14](#) para os valores exatos), obtemos os gráficos ROC apresentados nas [Fig. 19](#), [20](#) e [21](#).



Para os 3 gráficos apresentados, o modelo sem os dados demográficos **ute_genero** e **ute_idade** é o que melhor se apresenta.

A ligeira alteração das taxas de precisão (accuracy) dos modelos 17 (84,38%), 21 (84,39%) e 22 (84,38%) em MS SQL AS, e dos modelos 27 (84,39%), 30 (84,40%) e 33 (84,21%) revelam a presença de overfitting e sugerem a não utilização dos atributos **ute_genero** e **ute_idade**, tendo ainda em atenção que este último atributo apresenta valores null (332) que podem interferir – embora tratando-se de poucos registos – na construção da árvore de decisão.

Nos testes de Cross Validation realizados sobre o modelo 22 (testes efetuados sobre dados de teste) observam-se variações muito ténues entre os vários grupos de teste, não se verificam grandes flutuações no desvio padrão obtido, e constata-se um aumento da sensibilidade com o aumento do nº de registos testados, o que ilustra o bom comportamento do modelo.

A [Tab. 15](#) apresenta os valores obtidos em AS para o modelo 22, que melhoram à medida que as subdivisões são aplicadas sobre um maior número de registos (evitando-se o “underfitting”).

Número Registos de teste	200	1000	5000	10000	20000	30000
Verdadeiros						
Média	14,60	82,31	408,80	831,90	1671,00	2527,10
Desvio Padrão	1,56	2,83	5,64	5,86	13,45	20,73
Falsos						
Média	5,40	17,70	91,20	168,10	329,00	472,90
Desvio Padrão	1,56	2,75	5,64	5,70	13,23	20,83

Tab. 15 – Teste de cross-validation com 10 subgrupos no modelo 22

Constata-se ainda haver uma ligeira diferença entre a taxa de precisão (accuracy) obtida no modelo 22 em MS SQL AS (84,38%) e a obtida no modelo

33 em R (84,21%). Apesar dos dois modelos utilizarem o mesmo conjunto de atributos, a forma de criação da árvore difere entre os dois modelos, uma vez que o Microsoft Decision Tree utiliza, por defeito, um “split” múltiplo, e o ctree em R utiliza um “split” binário.

A [Tab. 16](#) apresenta os métodos de “split” e de ganho de informação de cada um dos Algoritmos relativos a árvores de decisão utilizados no MS SQL AS e no Sistema R.

Método Característica	Microsoft Decision Tree	<i>Package tree</i>	<i>Package rpart</i>	<i>Package party ctree</i>
Split Method	Complete (Multiway)	<i>Binary</i>	<i>Binary</i>	<i>Binary</i>
Information Gain	Bayesian Dirichlet Equivalent with uniform prior		<i>Gini index</i>	<i>Condition Inference (p-value)</i>

Tab. 16 – Métodos de split e de information gain dos algoritmos Decision Tree utilizados

De acordo com Landis e Koch (1977) e a escala de interpretação do indicador Kappa, o valor obtido nos modelos 22 (68,12%) em MS SQL AS e 33 (67,66%) no sistema R significam uma concordância substancial com o modelo perfeito, pelo que consideramos estar perante um modelo satisfatório para concluir o projeto.

Conclui-se portanto que a melhor opção para a construção do modelo em questão - tendo em atenção que não foi possível, por falta de dados, considerar alguns atributos que se poderiam revelar decisivos -, deve utilizar os atributos classificados ([Tab. 17](#)) excluindo do modelo os atributos demográficos referentes ao género e à idade.

Atributo	Conteúdo
ute_cdiassemfrecuencia	Classificação de 1 (menos dias sem frequentar) a 5 (mais dias sem frequentar)
ute_cmeseinscricao	Classificação de 1 (menos meses de inscrição) a 5 (mais meses de inscrição)
ute_cvolvegocio	Classificação de 1 (menor volume de negócios) a 5 (maior volume de negócios)
ute_cnaulasgrupo	Classificação de 1 (menor número de aulas de grupo frequentadas) a 5 (maior número de aulas de grupo frequentadas)
ute_cfrequencia	Classificação de 1 (menor frequência média) a 5 (maior frequência média)

Tab. 17 – Atributos a utilizar no modelo de previsão do padrão de desistências

Após estas observações, conclui-se poder avançar para o desenvolvimento proposto, tendo em atenção a utilização do modelo Decision Tree com base nos atributos apresentados na [Tab. 17](#) e sobre as ferramentas Microsoft (MS SQL AS) – modelo 22 designado por **Dm Retencao AllClass-Generoldade** - de

forma a que, após concluído e devidamente testado, se possa proceder à integração com a aplicação da empresa (e@sport®).

Não condicionando, mas completando a informação obtida do modelo e da ferramenta MS SQL AS, o gráfico Lift apresentado na [Fig. 22](#) para o modelo de referência selecionado permite cruzar o número total de utentes com o número de desistentes identificados no conjunto de teste e determinar que observando, por exemplo, 50% dos utentes, obteremos 86,21% dos utentes desistentes em que a probabilidade de virem a ser desistentes (predict probability) é igual ou superior a 34,66%.

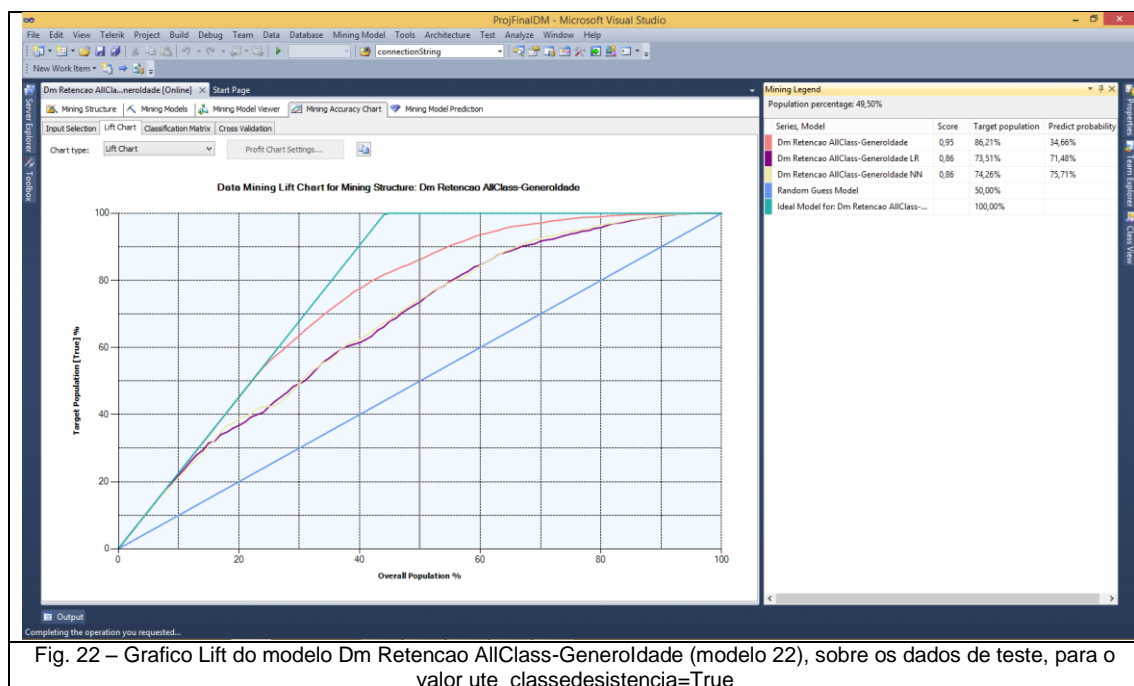


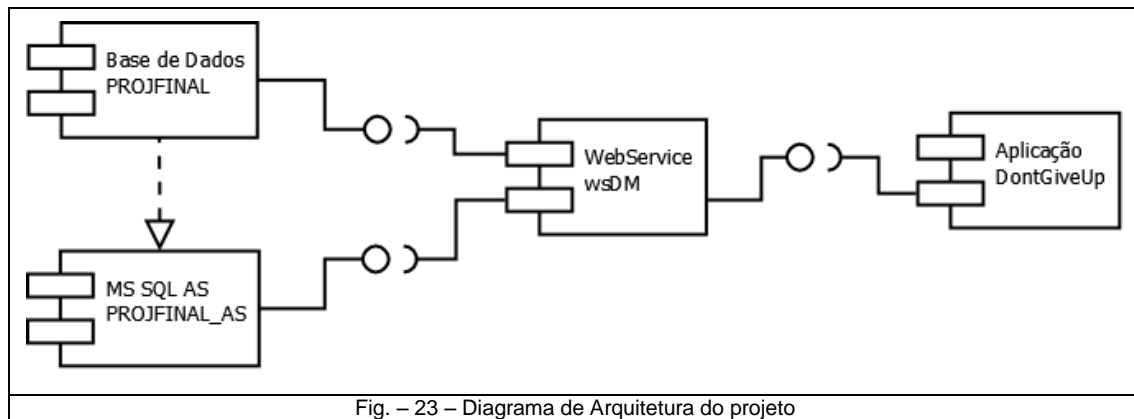
Fig. 22 – Gráfico Lift do modelo Dm Retencao AllClass-Generoldade (modelo 22), sobre os dados de teste, para o valor ute_classedesistencia=True

A apresentação deste valor (Predict Probability da [Fig. 22](#)) é apresentada nas estatísticas do modelo e importante para a execução do relatório de potenciais desistentes, funcionalidades apresentadas na aplicação desenvolvida.

5 Desenvolvimento

5.1 Análise da aplicação

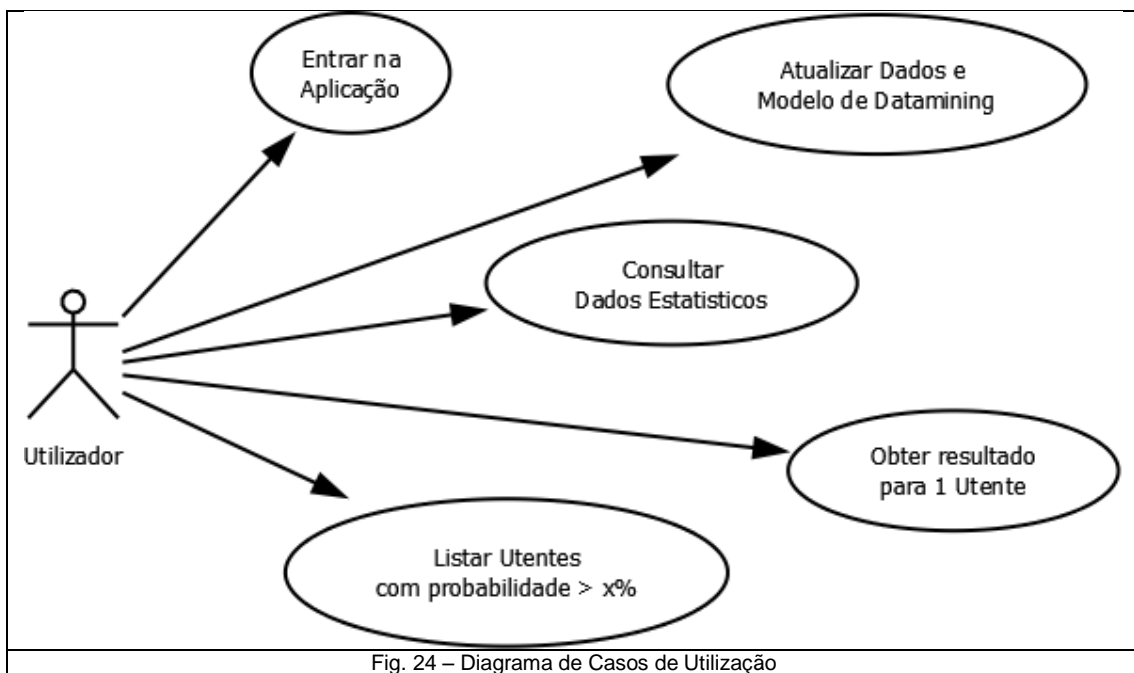
O desenvolvimento inclui a implementação do software em camadas, como ilustrado na [Fig. 23](#).



- a) Uma primeira camada composta por diversos procedimentos (storage procedures) desenvolvidos em Transact-SQL³:
 - i) Alguns dos procedimentos desenvolvidos dizem respeito à importação, limpeza de dados e discretização dos atributos e foram já referidos anteriormente;
 - ii) Implementação de um procedimento para retorno dos utentes de um clube e das suas características, em função dos atributos necessários para aplicação ao modelo;
 - iii) Implementação de funções escalares para simplificação e generalização de cálculos;
- b) Uma segunda camada, composta por um webservice com os seguintes métodos:
 - i) Atualizar os dados – transferência e tratamento dos dados da base de dados do cliente para a base de dados de suporte – e atualização do modelo;
 - ii) Retornar informação estatística sobre a performance do modelo;
 - iii) Retornar a probabilidade de um Utente ser desistente a partir dos valores indicados para cada atributo;

³ Transact-SQL é o dialeto SQL implementado no MS-SQL

- iv) Retornar a lista de utentes ativos que se encontram no padrão de desistência;
- c) Uma terceira camada, de formulários, correspondendo ao diagrama de casos de utilização da [Fig. 24](#), que consomem os métodos do webservice desenvolvido na segunda camada; a lista de formulários inclui:
 - i) Formulário de arranque da aplicação;
 - ii) Formulário para invocação do método de atualização dos dados e de atualização do modelo;
 - iii) Formulário para consulta da informação estatística do modelo;
 - iv) Formulário para indicação dos valores dos atributos e obtenção do resultado;
 - v) Formulário para listagem dos utentes ativos que se encontram no padrão de desistência – limitar por % de probabilidade;



A primeira camada: procedimentos implementados na base de dados

Como especificado anteriormente foram desenvolvidos vários procedimentos para importação ([Tab. 5](#)), limpeza dos dados ([Tab. 7](#)) e procedimentos para classificação dos registos ([Tab. 9](#)), de acordo com as características dos atributos.

Todos os procedimentos foram implementados na base e dados PROJFINAL,

de forma a evitar alterações na base de dados do cliente.

Uma vez que a função de cada procedimento já foi explicada anteriormente, nesta secção ilustra-se e explica-se apenas alguns dos procedimentos mais relevantes e funções não mencionadas anteriormente.

O primeiro procedimento analisado importa os dados dos Utentes da base de dados CLIENTE para a base de dados PROJFINAL.

```
1  -- =====
2  -- Author:      Paulo Pinheiro
3  -- Create date: 08/Abril/2015
4  -- Description: Procedimento de importação de registos de utentes
5  --             A importação deve ocorrer todos os dias 1 de cada mês e deve importar
6  --             todos os registos do mês anterior. São preenchidos os campos:
7  --             ute_id, ute_idsaft, ute_clube, ute_datanasc, ute_genero, ute_inicio
8  -- =====
9  ALTER PROCEDURE [dbo].[dmImportaUtentes]
10     @DATAINICIAL      datetime,
11     @DATAFINAL        datetime,
12     @NREGISTOS         int OUTPUT
13 AS
14 BEGIN
15     DECLARE          @anomesinicial      VARCHAR(6),
16                     @anomesfinal        VARCHAR(6),
17                     @anomestratamento    VARCHAR(6),
18                     @proxmes             DATETIME,
19                     @inicio              DATETIME,
20                     @data_ultimodiames    DATETIME;
21
22     SET NOCOUNT ON;
23
24     -- obtem ultima data de inscricao de um utente ja importado
25     SELECT @inicio = MAX(UTE_INICIO)
26     FROM dmRetencao
27
28     -- utiliza a datainicial passado por parametro ou a ultima data de inscricao, se for maior
29     IF @inicio IS NOT NULL AND @inicio > @DATAINICIAL
30         SET @anomesinicial = CONVERT(CHAR(6), DATEPART(mm, DATEADD(mm, +1, @inicio)), 112);
31     ELSE
32         SET @anomesinicial = CONVERT(CHAR(6), @DATAINICIAL, 112);
33
34     SET @proxmes = @DATAINICIAL
35     SET @anomestratamento = CONVERT(CHAR(6), @proxmes, 112);
36     SET @anomesfinal = CONVERT(CHAR(6), DATEADD(mm, -1, @DATAFINAL), 112);
37     SET @data_ultimodiames = DATEADD(dd, -1, CAST(CONVERT(CHAR(6), @DATAFINAL, 112) + '01' AS datetime))
38
39     SET @NREGISTOS = 0;
40
41     -- faz em ciclo desde o mês da data inicial
42     -- ate ao mes atual - 1
43     WHILE @anomestratamento <= @anomesfinal
44     BEGIN
45         BEGIN TRY
46             INSERT INTO dmRetencao (UTE_ID, UTE_IDSAFT, UTE_CLUBE, UTE_INICIO,
47                                     UTE_DATANASC, UTE_GENERO)
48             SELECT IND_ID, IND_IDSAFT, IND_CENIDINSC, IND_DTINSERT,
49                    IND_DTNASC,
50                    CASE IND_SEXO WHEN 'M' THEN 1 ELSE 0 END
51             FROM [CLIENTE].[dbo].[spoindivíduos]
52             WHERE CONVERT(CHAR(6), IND_DTINSERT, 112) = CONVERT(CHAR(6), @proxmes, 112)
53             SET @NREGISTOS = @NREGISTOS + @@ROWCOUNT;
54         END TRY
55         BEGIN CATCH
56             RETURN 3;
57         END CATCH
58
59         SET @proxmes = DATEADD(mm, +1, @proxmes);
60         SET @anomestratamento = CONVERT(CHAR(6), @proxmes, 112);
61     END
62
63     RETURN 0;
64 END
```

Fig. 25 – Storage Procedure *dmImportaUtentes*

Relativamente ao procedimento **dmImportaUtentes** apresentado na [Fig. 25](#) destaca-se os seguintes aspetos:

- O parâmetro **@NREGISTOS** (linha 12) declarado como parâmetro de output, permite obter o número de registos tratados no procedimento;
- De forma a delimitar o tratamento a períodos mensais, o procedimento calcula a data do ultimo dia do mês anterior à data indicada no parâmetro **@DATAFINAL** (linhas 34 e 35);
- De forma a evitar uma sobrecarga motivada na seleção de dados de um elevado número de registos, a importação é efetuada através de um ciclo WHILE (linhas 41 a 59) que importa as inscrições de cada mês, entre o mês da data inicial (**@DATAINICIAL**) e o mês final (**@DATAFINAL**);
- Realça-se a referência à base de dados CLIENTE para localização da tabela **spoindividuos** (linha 49).

O procedimento **dmVolNegociosUtentes** apresentado na [Fig. 26](#) calcula o volume de negócios de cada utente desde a data da sua inscrição até à data de atualização do modelo ou, caso tenha desistido, até à data da desistência.

Neste procedimento são de realçar os seguintes aspetos:

- A utilização de duas tabelas temporárias em memória (**@anulados** e **@volNegocios**). Na primeira é colocada a referência a documentos de Estorno de Recibos, documentos que não são relevantes para o volume de negócios, através do query indicado nas linhas 22 a 35. Na segunda, é colocado o somatório do volume de negócios de cada utente – query indicado nas linhas 42 a 54. Finalmente, a tabela **dmRetencao** é atualizada com o valor do volume de negócios de cada utente anteriormente colocado na tabela **@volNegocios** (linhas 60 a 63);
- Uma característica comum a todos os procedimentos é o controlo dos erros através das clausulas **TRY/CATCH** que possibilitam retornar um erro facilmente identificável por utilizadores ou aplicações que consomem estes “storages procedures”;

```
1  -- =====
2  -- Author:      Paulo Pinheiro
3  -- Create date: 13/Abril/2015
4  -- Description: Procedimento para calcular o volume de negocios de cada utente
5  -- =====
6  ALTER PROCEDURE [dbo].[dmVolNegociosUtentes]
7      @DATAINICIAL      datetime,
8      @DATAFINAL        datetime,
9      @NREGISTOS         int OUTPUT
10 AS
11 BEGIN
12     DECLARE @volNegocios TABLE (UTE_ID          UNIQUEIDENTIFIER PRIMARY KEY,
13                                  UTE_VOLNEGOCIOS NUMERIC(12,2));
14     DECLARE @anulados   TABLE (DOCUMENTO VARCHAR(40) COLLATE SQL_Latin1_General_CP1_CI_AS primary key);
15     DECLARE @data_ultimodiamas DATETIME;
16
17     SET NOCOUNT ON;
18
19     SET @data_ultimodiamas = DATEADD(dd, -1, CAST(CONVERT(CHAR(6), @DATAFINAL, 112) + '01' AS datetime))
20     -- insere os anulados que nao contam para uma tabela temporaria
21     BEGIN TRY
22         INSERT INTO @anulados (documento)
23         SELECT distinct cde_centro+cde_codtd+CAST(cde_numero AS VARCHAR(8))+CAST(cde_empid AS VARCHAR(8))
24         FROM [CLIENTE].[dbo].spodocumentos_detalhe,
25              [CLIENTE].[dbo].spodocumentos_master, dmRetencao
26         WHERE cde_codtd LIKE 'ES%'
27               AND cde_estcodtd LIKE 'RE%'
28               AND cde_empid = cma_empid
29               AND cde_centro = cma_centro
30               AND cde_codtd = cma_codtd
31               AND cde_numero = cma_numero
32               AND cma_status = 1
33               AND cma_indid = ute_id
34               AND cma_dataemi >= @DATAINICIAL
35               AND cma_dataemi <= ute_termino;
36     END TRY
37     BEGIN CATCH
38         RETURN 10;
39     END CATCH
40     -- calcula o volume de negocios
41     BEGIN TRY
42         INSERT INTO @volNegocios (UTE_ID, UTE_VOLNEGOCIOS)
43         SELECT CMA_INDID, SUM(cma_total)
44         FROM [CLIENTE].[dbo].spodocumentos_master, dmRetencao
45         WHERE CMA_INDID = UTE_ID
46               AND cma_dataemi >= @DATAINICIAL
47               AND cma_dataemi <= ute_termino
48               AND cma_dataemi <= @data_ultimodiamas
49               AND (cma_empid IS NULL OR cma_empid=1)
50               AND LEFT(cma_codtd,2) IN ('VD','ES','FA','NC','ND')
51               AND cma_status = 1
52               AND cma_centro+cma_codtd+CAST(cma_numero AS VARCHAR(8))+CAST(cma_empid AS VARCHAR(8)) NOT IN
53               (SELECT documento FROM @anulados)
54         GROUP BY CMA_INDID;
55     END TRY
56     BEGIN CATCH
57         RETURN 11;
58     END CATCH
59     BEGIN TRY
60         UPDATE dmRetencao
61         SET UTE_VOLNEGOCIOS = C.ute_volnegocios
62         FROM @volNegocios C
63         WHERE C.ute_id = dmRetencao.UTE_ID
64         SET @NREGISTOS = @@ROWCOUNT;
65     END TRY
66     BEGIN CATCH
67         RETURN 12;
68     END CATCH
69     RETURN 0;
70 END
```

Fig. 26 – Storage Procedure *dmVolNegociosUtentes*

O procedimento ***dmPrepareData*** sequencia todos os procedimentos de importação e limpeza dos dados, e é utilizado pelo método ***wsDataAndModelUpdate***. O seu código é apresentado na [Fig. 27](#).

```
1  -- =====
2  -- Author:      Paulo Pinheiro
3  -- Create date: 18/Jun/2015
4  -- Description: Procedimento geral de carregamento e limpeza dos dados
5  -- =====
6  ALTER PROCEDURE [dbo].[dmPrepareData]
7      @DATAINICIAL    datetime,
8      @DATAFINAL      datetime
9  AS
10 BEGIN
11     DECLARE @resultados TABLE (ID          INT IDENTITY(1,1) PRIMARY KEY,
12                                  ITEM        VARCHAR(30),
13                                  NREGISTOS    INT,
14                                  ERROR_N     INT)
15
16     DECLARE @nregistos    INT,
17             @ativos       INT,
18             @desistentes  INT,
19             @error        BIT;
20
21     IF @DATAINICIAL IS NULL OR @DATAFINAL IS NULL
22     BEGIN
23         RETURN -1;
24     END
25
26     SET NOCOUNT ON;
27     -- erro n=2
28     EXEC @error = dmApagaUtentes @nregistos OUTPUT;
29     INSERT INTO @resultados (ITEM, NREGISTOS, ERROR_N)
30     VALUES ('DeletedUsers', @nregistos, @error);
31     -- erro n=3
32     EXEC @error = dmImportaUtentes @DATAINICIAL, @DATAFINAL, @nregistos OUTPUT;
33     INSERT INTO @resultados (ITEM, NREGISTOS, ERROR_N)
34     VALUES ('ImportedUsers', @nregistos, @error);
35     -- erro n=4
36     EXEC @error = dmInicializaUtentes @nregistos OUTPUT;
37     INSERT INTO @resultados (ITEM, NREGISTOS, ERROR_N)
38     VALUES ('InitializeUsers', @nregistos, @error);
39     -- erro n=5 ou 6
40     EXEC @error = dmPeriodoAtivo @DATAFINAL, @ativos OUTPUT, @desistentes OUTPUT;
41     INSERT INTO @resultados (ITEM, NREGISTOS, ERROR_N)
42     VALUES ('ActiveUsers', @ativos, @error);
43     INSERT INTO @resultados (ITEM, NREGISTOS, ERROR_N)
44     VALUES ('GiveUpUsers', @desistentes, @error);
45     -- erro n=7
46     EXEC @error = dmIdadeUtentes @DATAFINAL, @nregistos OUTPUT;
47     INSERT INTO @resultados (ITEM, NREGISTOS, ERROR_N)
48     VALUES ('UsersAge', @nregistos, @error);
49     -- erro n=8 ou 9
50     EXEC @error = dmUltFrequencia @DATAINICIAL, @DATAFINAL, @nregistos OUTPUT;
51     INSERT INTO @resultados (ITEM, NREGISTOS, ERROR_N)
52     VALUES ('UsersDaysWithoutFrequency', @nregistos, @error);
53     -- erro n=10, 11 ou 12
54     EXEC @error = dmVolNegociosUtentes @DATAINICIAL, @DATAFINAL, @nregistos OUTPUT;
55     INSERT INTO @resultados (ITEM, NREGISTOS, ERROR_N)
56     VALUES ('UsersBusinessAmount', @nregistos, @error);
57     -- erro n=13, 14 ou 15
58     EXEC @error = dmAulasGrupo @DATAINICIAL, @DATAFINAL, @nregistos OUTPUT;
59     INSERT INTO @resultados (ITEM, NREGISTOS, ERROR_N)
60     VALUES ('UsersGroupClassFrequency', @nregistos, @error);
61     -- erro n=16 ou 17
62     EXEC @error = dmFreqMedia @DATAINICIAL, @DATAFINAL, @nregistos OUTPUT;
63     INSERT INTO @resultados (ITEM, NREGISTOS, ERROR_N)
64     VALUES ('UsersAvgFrequency', @nregistos, @error);
65     -- erro n=18, 19 ou 20
66     EXEC @error = dmLimpaUtentes @DATAFINAL, @nregistos OUTPUT;
67     INSERT INTO @resultados (ITEM, NREGISTOS, ERROR_N)
68     VALUES ('DataCleaning', @nregistos, @error);
69     -- erro n=21
70     EXEC @error = dmCorrigeIdades @nregistos OUTPUT;
71     INSERT INTO @resultados (ITEM, NREGISTOS, ERROR_N)
72     VALUES ('CleanUsersAge', @nregistos, @error);
73
74     SELECT *
75     from @resultados
76     ORDER BY id;
77     RETURN 0;
78 END
```

Fig. 27 – Storage Procedure *dmPrepareData*

Realça-se o fato deste procedimento executar todos os restantes sequencialmente, independentemente da ocorrência de erro. O objetivo é

apresentar, no final, uma lista de todos os procedimentos que correram, com ou sem erro, e no caso de terem executado sem erro, indicar o número de registos que afetaram. Para obter este resultado, é declarada a tabela **@resultados** onde é inserido o resultado da execução de cada procedimento. No final, é efetuado um query SELECT que retorna todos os registos (linhas 73 a 75).

Por fim, apresenta-se na [Fig. 28](#) o procedimento de classificação do atributo referente ao número de aulas de grupo, **dmClassificaNAulasGrupo**.

```
1  -- =====
2  -- Author:      Paulo Pinheiro
3  -- Create date: 26/Abril/2015
4  -- Description: Procedimento de classificacao do numero de aulas de grupo frequentadas
5  -- =====
6  ALTER PROCEDURE [dbo].[dmClassificaNAulasGrupo]
7  AS
8  BEGIN
9      DECLARE @classifica TABLE (ID INT IDENTITY(1,1) PRIMARY KEY,
10                                  IDSAFT INT)
11
12      DECLARE @maximo    int,
13              @parte     int,
14              @linferior int,
15              @lsuperior int,
16              @i          int;
17
18      SET NOCOUNT ON;
19
20      BEGIN TRY
21          INSERT INTO @classifica (IDSAFT)
22          SELECT ute_idsaft
23          FROM dmRetencao
24          ORDER BY ute_naulasgrupo
25      END TRY
26      BEGIN CATCH
27          RETURN 109;
28      END CATCH
29
30      SELECT @maximo=MAX(id)
31      FROM @classifica
32
33      -- retorno da parte inteira truncada
34      -- por isso o ultimo intervalo pode ter menos registos
35      SET @parte = @maximo / 5;
36      SET @i = 1;
37      SET @linferior = 1;
38      SET @lsuperior = @parte;
39
40      WHILE @i <= 5
41      BEGIN
42          IF @i = 5
43          BEGIN
44              BEGIN TRY
45                  -- faz para todos os restantes
46                  UPDATE dmRetencao
47                  SET UTE_CNAULASGRUPO = @i
48                  FROM @classifica C
49                  WHERE dmRetencao.UTE_IDSAFT = C.idsaft
50                      AND C.id >= @linferior
51              END TRY
52              BEGIN CATCH
53                  RETURN 110;
54              END CATCH
55          END
56          ELSE
57          BEGIN
58              BEGIN TRY
59                  -- faz para @parte registos
60                  UPDATE dmRetencao
61                  SET UTE_CNAULASGRUPO = @i
62                  FROM @classifica C
63                  WHERE dmRetencao.UTE_IDSAFT = C.idsaft
64                      AND C.id >= @linferior
65              END TRY
```

66	AND C.id <= @lsuperior
67	END TRY
68	BEGIN CATCH
69	RETURN 111;
70	END CATCH
71	END
72	SET @linferior = @linferior + @parte;
73	SET @lsuperior = @lsuperior + @parte;
74	SET @i = @i + 1
75	END
76	RETURN 0;
77	END

Fig. 28 – Storage Procedure dmClassificaNAulasGrupo

Recorda-se que foi também implementado o procedimento **dmClassificaAtributos** que executa todos os procedimentos de classificação, e que também é utilizado pelo método **wsDataAndModelUpdate** implementado no webservice apresentados a seguir.

A segunda camada: os serviços web (“webservices”)

“Um serviço web geralmente consiste numa coleção de operações que podem ser utilizadas por um cliente através da internet. [...] Um serviço web pode ser gerido por um servidor web juntamente com outras páginas; ou pode ser um serviço totalmente separado.

A característica chave da maioria dos serviços web é de que podem processar mensagens SOAP⁴ em formato XML⁵.” (in Distributed Systems Concepts and Design, Pag. 383)

A citação anterior descreve resumidamente o objetivo da implementação do serviço web apresentado nos parágrafos seguintes, que descrevem as suas características gerais, bem como o processo de implementação específico de cada método.

Aspetos gerais do serviço web “wsDM”

Os serviços web implementados recebem parâmetros de entrada e retornam informação em formato XML. Por questões de simplificação de desenvolvimento e teste, embora formatado em XML, o parâmetro de entrada é na realidade uma string⁶.

De forma geral, todos os serviços web desenvolvidos exigem que os parâmetros

⁴ SOAP – Simple Object Access Protocol

⁵ XML – eXtensible Markup Language

⁶ O “encapsulamento” da mensagem XML em string permite a realização de testes pela execução direta do serviço web em modo de desenvolvimento através da ferramenta de desenvolvimento

de entrada apresentam a seguinte forma:

Tag XML	Descrição
<WSDM></WSDM>	Tag global que indica tratar-se de um documento do serviço web wsDM Encapsula todas as restantes <i>tags</i> do documento XML
<HELP></HELP>	Tag do tipo booleana para indicação de que se pretende a apresentação dos atributos e respetivo tipo dos parâmetros de saída Indicar 1 para apresentar os atributos e respetivos tipos, 0 em caso contrário <i>Parent Tag:</i> <WSDM>
<INPARAM></INPARAM>	Tag que engloba todos os parâmetros de entrada <i>Parent Tag:</i> <WSDM>
<parametro..n></parametro..n>	Tag respeitante ao parâmetro <i>Parent Tag:</i> <INPARAM>
Exemplo de um documento XML com parâmetros de entrada	
<pre> <WSDM> <HELP>1</HELP> <INPARAM> <START_DATE>2012-08-01</START_DATE> <END_DATE>2015-04-15</END_DATE> </INPARAM> </WSDM> </pre>	
Tab. 18 – Documento XML para definição de parâmetros de entrada	

Relativamente aos parâmetros de entrada há que referenciar em especial a *tag* **HELP**, que permite indicar ao serviço web que se pretende obter nos parâmetros de saída, a especificação de todos os parâmetros de saída e os respetivos tipos de dados. Todos os parâmetros de entrada devem ser encapsulados no parâmetro **INPARAM**.

Os parâmetros de saída apresentam a seguinte forma:

Tag XML	Descrição
<WSDM></WSDM>	Tag global que indica tratar-se de um documento do serviço web wsDM Encapsula todas as restantes <i>tags</i> do documento XML
<OUTPARAM></OUTPARAM>	Tag que engloba todos os parâmetros de entrada <i>Parent Tag:</i> <WSDM>
<parametro..n></parametro..n>	Tag respeitante ao atributo de retorno <i>Parent Tag:</i> <OUTPARAM>
<ERROR></ERROR>	Tag de status de retorno <i>Parent Tag:</i> <OUTPARAM>
Exemplo de um documento XML com parâmetros de saída	
<pre> <WSDM> <OUTPARAM> <CLASSIFICACAO> <REGISTRATION_PERIOD> <CLASS>2</CLASS> <VALUE>2</ VALUE > </ REGISTRATION_PERIOD > ...itens excluídos por questão de simplificação ... <REGISTRATION_PERIOD> <CLASS>2</CLASS> <VALUE>2</ VALUE > </ REGISTRATION_PERIOD > </CLASSIFICACAO> </OUTPARAM> <ERROR> <ERROR_STATUS> OK </ERROR_STATUS> <ERROR_NUMBER> </ERROR_NUMBER> <ERROR_MESSAGE> </ERROR_MESSAGE> </ERROR> </WSDM> </pre>	
Tab. 19 – Documento XML para definição de parâmetros de saída	

No que concerne aos parâmetros de saída, estes apresentam as seguintes características:

- Todos os parâmetros de saída são encapsulados na *tag* **OUTPARAM**;
- Todos os documentos XML que referenciam parâmetros de saída incluem uma *tag* **ERROR** onde é especificado o resultado da operação a que diz respeito o método. A *tag* **ERROR_STATUS** pode assumir o valor **OK** quando o método se executa sem erro, ou **NOK** caso contrário. No caso de ter sido executado com erro, a *tag* **ERROR_NUMBER** indica o número do erro, e a *tag* **ERROR_MESSAGE** a respetiva mensagem de erro.

Para o desenvolvimento do serviço web em questão utilizou-se o ambiente de desenvolvimento Visual Studio 2010, a linguagem C# com Framework 4.0.

Foi também utilizada a Entity Framework para simplificar o processo de trabalho com as estruturas de dados nas bases de dados. São também referidas pontualmente, na explicação de cada método, a utilização de outras formas de obtenção de dados.

Os próximos parágrafos apresentam os vários métodos desenvolvidos e os pormenores específicos de cada um.

Método wsDataAndModelUpdate

Este procedimento importa os dados dos Utentes da base de dados **CLIENTE** para a base de dados **PROJFINAL**, procede ao processo de limpeza dos utentes e classificação dos atributos explicado anteriormente, atualiza o modelo e insere os dados estatísticos referentes à performance do modelo na tabela de indicadores (**dmlIndicadores**), de forma a que se possa acompanhar a evolução do modelo ao longo do tempo.

A lógica de desenvolvimento do método é apresentada na [Fig. 29](#) que apresenta o seu Diagrama de Atividades.

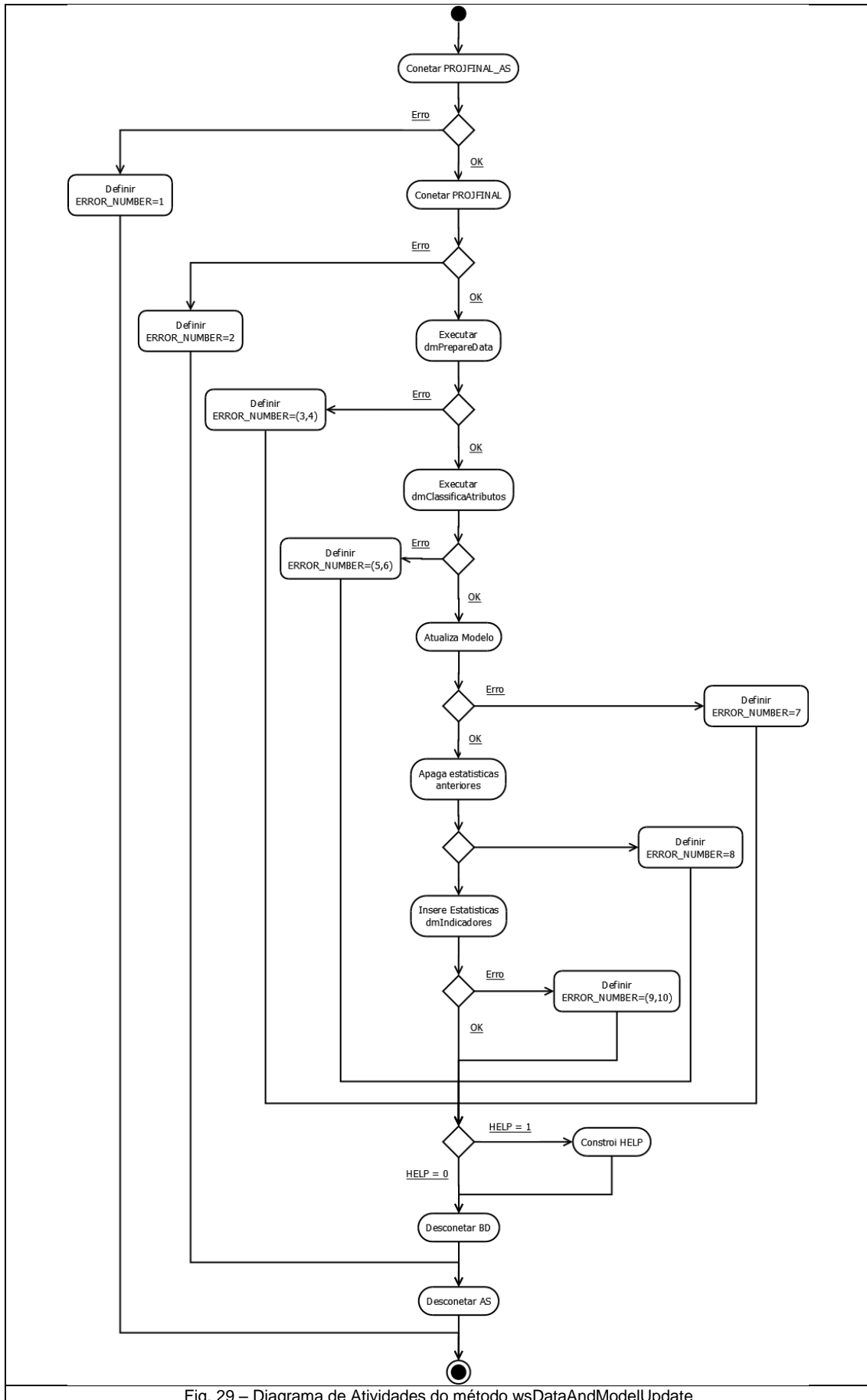


Fig. 29 – Diagrama de Atividades do método wsDataAndModelUpdate

A [Tab. 20](#) apresenta o formato dos parâmetros de entrada e de saída, bem como o exemplos dos mesmos.

Parâmetros de entrada (String XML)		
<HELP> </HELP>	Opção de retorno de informação de ajuda	Boolean 1 para True 0 para False
<START_DATE> </START_DATE>	Data a partir da qual se pretende atualizar o modelo Por defeito, 01/Agosto/2012	Date YYYYMMDD
<END_DATE> </END_DATE>	Data à qual se pretende atualizar o modelo – a atualização é efetuada até ao último dia do mês anterior à data indicada	Date YYYYMMDD
Formato do documento XML a utilizar como parâmetros de entrada		
<WSDM> <HELP>1</HELP> <INPARAM> <START_DATE>2012-08-01</START_DATE> <END_DATE>2015-04-15</END_DATE> </INPARAM> </WSDM>		
Parâmetros de saída (XML)		
<HELP> </HELP>	Tag que encapsula a informação sobre os atributos de retorno <i>Parent tag: <WSDM></i>	
<ATTRIBUTE> </ATTRIBUTE>	Tag que encapsula cada os itens explicativos de cada atributo <i>Parent tag: <HELP></i>	
<ATTRIBUTE_NAME> </ATTRIBUTE_NAME>	Nome da tag (atributo) retornada <i>Parent tag: <ATTRIBUTE></i>	String
<ATTRIBUTE_TYPE> </ATTRIBUTE_TYPE>	Tipo do atributo retornado <i>Parent tag: <ATTRIBUTE></i>	String
<OUTPARAM> </OUTPARAM>	Tag que encapsula todos os itens de dados <i>Parent tag: <WSDM></i>	
<ITEM> </ITEM>	Tag que encapsula cada registo de dados <i>Parent tag: <DATA></i>	
<TYPE> </TYPE>	Tipo de processo executado – corresponde à execução de um storage procedure na base de dados <i>Parent tag: <ITEM></i>	String
<NRECORDS> </NRECORDS>	Nº de registos afetados pela execução do processo Este parâmetro não é preenchido na fase de execução dos procedimentos de classificação <i>Parent tag: <ITEM></i>	Integer
<ERRORN> </ERRORN>	Apresenta 0 se o storage procedure executou sem erros Apresenta um número diferente de 0 correspondente à fase do storage procedure onde ocorreu o erro <i>Parent tag: <ITEM></i>	Integer
<ERROR> </ERROR>	Encapsula a informação sobre um erro fatal ocorrido na execução do método, ou a sua conclusão com sucesso <i>Parent tag: <WSDM></i>	
<ERROR_STATUS> </ERROR_STATUS>	Preenchido com OK ou NOK , caso não ocorra, ou ocorra um erro, respetivamente <i>Parent tag: <ERROR></i>	String OK / NOK
<ERROR_NUMBER> </ERROR_NUMBER>	Numero do erro (ver diagrama) <i>Parent tag: <ERROR></i>	Integer
<ERROR_MESSAGE> </ERROR_MESSAGE>	Descrição do erro tal como é retornado pela exceção <i>Parent tag: <ERROR></i>	String
Formato do documento XML com parâmetros de saída		
<WSDM> <HELP> <ATTRIBUTE> <ATTRIBUTE_NAME>ITEM</ATTRIBUTE_NAME> <ATTRIBUTE_TYPE>string</ATTRIBUTE_TYPE> </ATTRIBUTE> ...itens excluídos por questão de simplificação ... <ATTRIBUTE> <ATTRIBUTE_NAME>ERRORN</ATTRIBUTE_NAME> <ATTRIBUTE_TYPE>integer</ATTRIBUTE_TYPE> </ATTRIBUTE> </HELP> <OUTPARAM> <ITEM> <TYPE>DeletedUsers</TYPE>		

<pre> <NRECORDS>23033</NRECORDS> <ERRORN>0</ERRORN> </ITEM> ...itens excluídos por questão de simplificação ... <ITEM> <TYPE>dmClassificaFreqMedia</TYPE> <ERRORN>0</ERRORN> </ITEM> </DATA> <ERROR> <ERROR_STATUS>OK</ERROR_STATUS> <ERROR_NUMBER></ERROR_NUMBER> <ERROR_MESSAGE></ERROR_MESSAGE> </ERROR> </WSDM> </pre>
Tab. 20 – Especificação do método wsDataAndModelUpdate

Neste método, como em todos restantes, o código é escrito em regiões que agrupam as funcionalidades do mesmo. No caso deste método, foram criadas as subdivisões de código explicadas em pormenor nos próximos parágrafos. Dado que os métodos têm todos a mesma organização, nos restantes métodos serão apontadas apenas situações específicas que se considerem relevantes mencionar.

i) Declaração de variáveis

<pre> 1 # region Declaracao de variaveis 2 3 bool needHelp = false, erro = false; 4 String anomes; 5 DateTime dataInicial = Convert.ToDateTime("08/01/2012"), dataFinal, dataTmp; 6 7 XmlDocument xmliParam = new XmlDocument(); 8 XmlDocument xmliParam = new XmlDocument(); 9 10 XmlDocument xmlwsDM = xmliParam.CreateNode(XmlNodeType.Element, "WSDM", null); 11 XmlNode xmlError = null; 12 XmlNode xmlData = null; 13 XmlNode xmlHelp = null; 14 15 XmlNode xmlAttr = null; 16 XmlNode xmlItem = null; 17 XmlNode xmlType = null; 18 XmlNode xmlRecords = null; 19 XmlNode xmlErrorN = null; 20 21 PROJFINALEntities entModel = null; 22 List<dmPrepareData_Result> preparaData = null; 23 List<dmClassificaAtributos_Result> classificaAtributos = null; 24 25 string connectionString = null; 26 AdomdConnection SSASConnection = new AdomdConnection(); 27 AdomdCommand cmd = new AdomdCommand(); 28 AdomdDataReader reader; 29 30 XmlaClient clienteXmla = new XmlaClient(); 31 32 # endregion FIM Declaracao de variaveis </pre>
Fig. 30 – Região de declaração de variáveis do método wsDataAndModelUpdate

No que diz respeito à declaração de variáveis, realça-se a declaração dos documentos XML que assinalam os parâmetros de entrada (**xmliParam**, linha 7) e saída (**xmliParam**, linha 8), e as declarações dos nós e atributos XML que constituem os documentos XML recebido e a retornar (linhas 10 a 19);

As linhas 21 a 30 declaram os objetos necessários para:

- Estabelecer a comunicação com a base de dados PROJFINAL, que é efetuada através do modelo **Entities** previamente criado (**PROJFINALEntities**), sendo declarado o objeto **entModel**. O modelo criado é ilustrado na [Fig. 31](#);

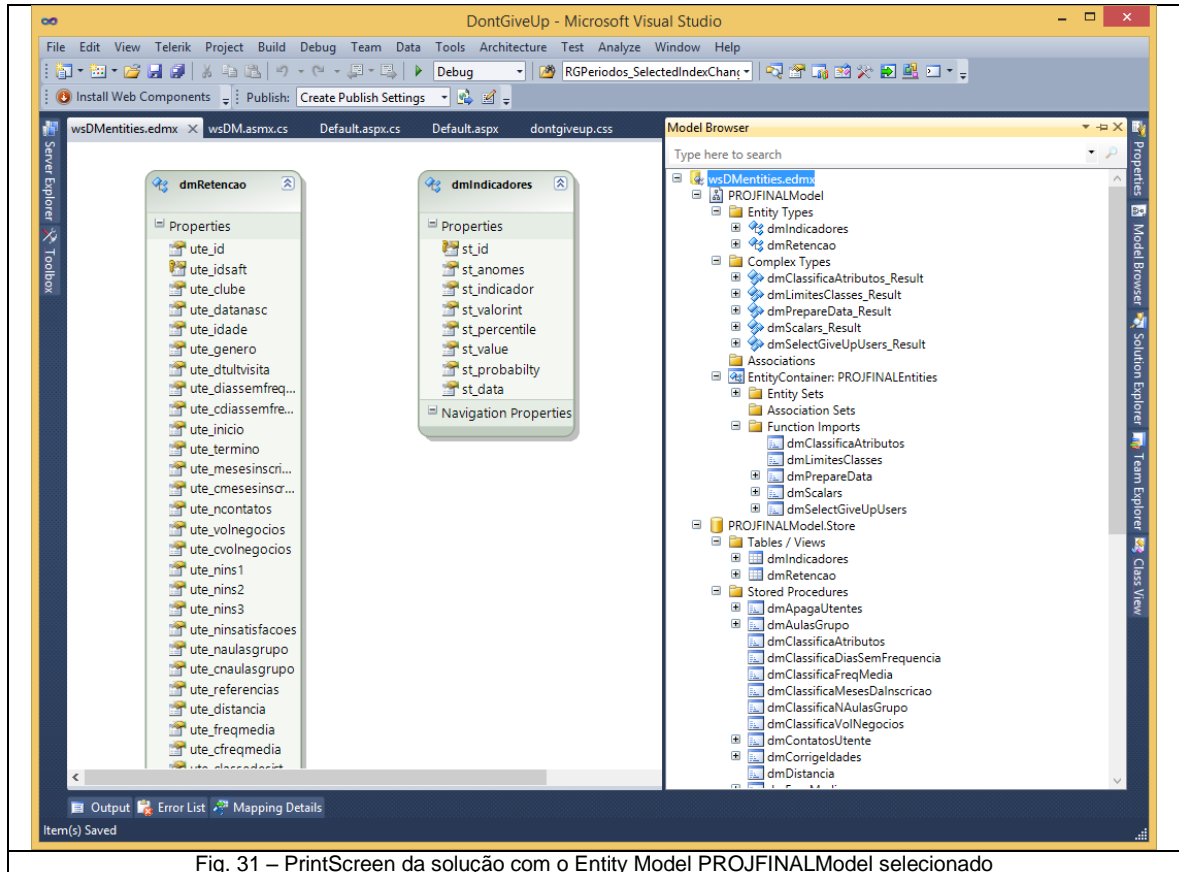


Fig. 31 – PrintScreen da solução com o Entity Model PROJFINALModel selecionado

São também declaradas as listas de objetos que acolhem os dados retornados pela execução dos 2 procedimentos (linha 22 e 23) de preparação e classificação dos dados;

- Estabelecer a ligação com o MS SQL AS. Quer a ligação, quer os restantes objetos necessários para obter comandos e guardar os respetivos resultados, são objetos que fazem parte de uma framework própria (**ADOMD.NET**) para comunicação com o MS SQL AS desenvolvida pela Microsoft (linhas 25 a 28);
- O Cliente XMLA⁷ destina-se a executar o comando de atualização do modelo de Data Mining;

⁷ XMLA – XML for Analysis é um protocolo baseado em SOAP (de formato XML) criado para aceder a fornecedores de dados multidimensionais standards pela web

ii) Conexão ao MS SQL AS

A Fig. 32 ilustra o processo de estabelecer comunicação com o MS SQL AS - PROFINAL_AS. É definida a string de conexão (linha 4) e afetado o parâmetro correspondente no objeto de ligação (linha 5).

Posteriormente tenta-se estabelecer a comunicação (linha 9). Caso ocorra um erro, a exceção é apanhada (linha 11), sendo preparado o XML de retorno com a especificação do erro (linhas 13 a 16).

```
1  # region Conexao a base de dados AS
2
3  // SSAS Model
4  connectionString = "Data Source=PP-PORTO\\PP;Initial Catalog=PROJFINAL_AS;User ID=sa; Password=xxxx;";
5  SSASConnection.ConnectionString = connectionString;
6
7  try
8  {
9      SSASConnection.Open();
10 }
11 catch (AdomdConnectionException ex)
12 {
13     xmlError = returnErrorInXml(xmliParam, "NOK", "1", ex.Message);
14     xmlwsDM.AppendChild(xmlError);
15     xmliParam.AppendChild(xmlwsDM);
16     return xmliParam.OuterXml.ToString();
17 }
18
19 # endregion FIM Conexao a base de dados AS
20
```

Fig. 32 – Região de código relativa à conexão ao MS SQL AS

iii) Conexão à base de dados

Como ilustrado na [Fig. 33](#), embora com objetos de tipos diferentes, o código de ligação à base de dados PROJFINAL é muito semelhante. Uma vez que são efetuados pesquisas complexas sobre a base de dados, aumentou-se o tempo de vida desta conexão (*Timeout*).

```
1  # region Conexao Base de Dados PROJFINAL
2
3  entModel = new PROJFINALEntities();
4  try
5  {
6      entModel.CommandTimeout = int.MaxValue;
7      entModel.Connection.Open();
8  }
9  catch (Exception ex)
10 {
11     xmlError = returnErrorInXml(xmliParam, "NOK", "2", ex.Message);
12     xmlwsDM.AppendChild(xmlError);
13     xmliParam.AppendChild(xmlwsDM);
14     return xmliParam.OuterXml.ToString();
15 }
16
17 # endregion FIM Conexao Base de Dados PROJFINAL
```

Fig. 33 – Região de código relativa à conexão ao MS SQL AS

iv) Invocação dos procedimentos de conversão, limpeza e classificação

A [Fig. 34](#) apresenta uma região de código correspondente à execução dos procedimentos, e formatação do documento XML de saída com os respetivos dados retornados pelos procedimentos.

```
1  # region Invocacao dos procedimentos de conversao, limpeza e classificacao
2
3  # region (dmPrepareData) Preparacao e limpeza de dados
4
5  try
6  {
7      preparaData = entModel.dmPrepareData(dataInicial, dataFinal).ToList();
8
9      foreach(dmPrepareData_Result _lista in preparaData)
10     {
11         xmlItem = xmloParam.CreateElement("ITEM");
12
13         xmlType = xmloParam.CreateElement("TYPE");
14         xmlType.InnerText = _lista.item;
15
16         xmlRecords = xmloParam.CreateElement("NRECORDS");
17         xmlRecords.InnerText = _lista.nregistos.ToString();
18
19         xmlErrorN = xmloParam.CreateElement("ERRORN");
20         xmlErrorN.InnerText = _lista.error_n.ToString();
21
22         if (_lista.error_n != 0)
23             erro = true;
24
25         xmlItem.AppendChild(xmlType);
26         xmlItem.AppendChild(xmlRecords);
27         xmlItem.AppendChild(xmlErrorN);
28         xmlData.AppendChild(xmlItem);
29     }
30 }
31 catch (EntityException ex)
32 {
33     SSASConnection.Close();
34     entModel.Connection.Close();
35     xmlError = returnErrorInXml(xmloParam, "NOK", "3", ex.Message);
36     xmlwsDM.AppendChild(xmlError);
37     xmloParam.AppendChild(xmlwsDM);
38     return xmloParam.OuterXml.ToString();
39 }
40
41 if (erro)
42 {
43     SSASConnection.Close();
44     entModel.Connection.Close();
45     xmlError = returnErrorInXml(xmloParam, "NOK", "4", "ERROR: dmPrepareData");
46     xmlwsDM.AppendChild(xmlError);
47     xmloParam.AppendChild(xmlwsDM);
48     return xmloParam.OuterXml.ToString();
49 }
50
51 # endregion (dmPrepareData)
52
53 (dmClassificaAtributos) Classificacao dos atributos
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98 # endregion FIM Invocacao dos procedimentos de conversao, limpeza e classificacao
```

Fig. 34 – Região de invocação dos **storage procedures** *dmPrepareData* e *dmClassificaAtributos*

A execução do procedimento (linha 7) coloca o resultado na variável **preparaData**. De seguida (linhas 9 a 29) a variável **preparaData** é iterada, cada atributo é colocado num elemento XML (linhas 13 a 20), agrupado num nó XML (linhas 25 a 27) e adicionado ao documento XML (linha 28).

Caso ocorra um erro na execução do procedimento, a exceção é captada e as conexões abertas são fechadas (linha 33 e 34). A informação de erro é colocada num elemento XML, adicionado ao documento de retorno, e a execução do função retorna o documento XML (linhas 35 a 38).

Uma vez que a informação resultante da execução do procedimento pode trazer a indicação da ocorrência de um erro (linhas 22 e 23), neste caso um erro “tratado” pelo próprio **storage procedure**, a situação é tratada da mesma forma que a indicada no paragrafo anterior (linhas 41 a 49).

Uma vez que a execução do procedimento de classificação dos atributos é tratado da mesma forma, optou-se por manter essa região de código omitida.

v) Atualização do modelo

Para efetuar a atualização do modelo basta estabelecer a conexão via XMLA (linha 6), afetar uma string com o script gerado pelo MS SQL AS referente ao modelo em questão, e enviar a respetiva mensagem (linha 24).

```
1  # region Atualização do Modelo
2
3  // utiliza cliente XMLA para processar o modelo
4  try
5  {
6      clienteXmla.Connect(connectionString);
7      string xmla = "<Batch xmlns=\"http://schemas.microsoft.com/analysisservices/2003/engine\">" +
8          "<Parallel>" +
9          "<Process xmlns:xsd=\"http://www.w3.org/2001/XMLSchema\" " +
10         "xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\" " +
11         "xmlns:ddl2=\"http://schemas.microsoft.com/analysisservices/2003/engine/2\" " +
12         "xmlns:ddl2_2=\"http://schemas.microsoft.com/analysisservices/2003/engine/2/2\" " +
13         "xmlns:ddl100_100=\"http://schemas.microsoft.com/analysisservices/2008/engine/100/100\" " +
14         "xmlns:ddl200=\"http://schemas.microsoft.com/analysisservices/2010/engine/200\" " +
15         "xmlns:ddl200_200=\"http://schemas.microsoft.com/analysisservices/2010/engine/200/200\" " +
16         "xmlns:ddl300=\"http://schemas.microsoft.com/analysisservices/2011/engine/300\" " +
17         "xmlns:ddl300_300=\"http://schemas.microsoft.com/analysisservices/2011/engine/300/300\" " +
18         "xmlns:ddl400=\"http://schemas.microsoft.com/analysisservices/2012/engine/400\" " +
19         "xmlns:ddl400_400=\"http://schemas.microsoft.com/analysisservices/2012/engine/400/400\">" +
20         "<Object>" +
21         "<DatabaseID>PROJFINAL_AS</DatabaseID>" +
22         "<MiningStructureID>Dm Retencao AllClass-GeneroIdade</MiningStructureID>" +
23         "</Object>" +
24         "<Type>ProcessFull</Type>" +
25         "<WriteBackTableCreation>UseExisting</WriteBackTableCreation>" +
26         "</Process>" +
27         "</Parallel>" +
28         "</Batch>";
29
30         clienteXmla.Send(xmla, null);
31         clienteXmla.Disconnect();
32     }
33     catch (XmlaException ex)
34     {
35         SSASConnection.Close();
36         entModel.Connection.Close();
37         xmlError = returnErrorInXml(xmlParam, "NOK", "7", ex.Message);
38         xmlwsDM.AppendChild(xmlError);
39         xmlParam.AppendChild(xmlwsDM);
40         return xmlParam.OuterXml.ToString();
41     }
42 # endregion FIM Atualização do Modelo
```

Fig. 35 – Região de código para atualização do modelo de Data Mining

O tratamento de erro efetuado já foi explicado anteriormente.

vi) Seleção e inserção dos indicadores do modelo relativos ao período

A [Fig. 36](#) ilustra o código desenvolvido para obter os dados estatísticos após a atualização do modelo.

```
1  # region Insere indicadores relativos ao periodo
2
3  apaga indicadores registados anteriormente para o mesmo mes
19
20  cmd.Connection = SSASConnection;
21
22  # region Matriz de Classificacao do modelo
23
24  // execute o CALL ao procedimento para obter a matrix de classificacao
25  // modelo [DM Retencao AllClass-GeneroIdade]
26  // dados de teste = 2
27  // atributo a prever = 'ute classedesistencia'
28  cmd.CommandText = "CALL SystemGetClassificationMatrix( [DM Retencao AllClass-GeneroIdade], 2, 'ute
classedesistencia')";
29
30  try
31  {
32
33      reader = cmd.ExecuteReader();
34      while (reader.Read())
35      {
36
37          dmIndicadores dmIndicadores_insert = new dmIndicadores();
38          dmIndicadores_insert.st_anomes = anomes;
39          dmIndicadores_insert.st_data = DateTime.Now;
40          if (reader[2].ToString() == "False" && reader[3].ToString() == "False")
41              dmIndicadores_insert.st_indicador = "CM True Negative";
42          else if (reader[2].ToString() == "True" && reader[3].ToString() == "True")
43              dmIndicadores_insert.st_indicador = "CM True Positive";
44          else if (reader[2].ToString() == "False" && reader[3].ToString() == "True")
45              dmIndicadores_insert.st_indicador = "CM False Positive";
46          else if (reader[2].ToString() == "True" && reader[3].ToString() == "False")
47              dmIndicadores_insert.st_indicador = "CM False Negative";
48          dmIndicadores_insert.st_valorint = Convert.ToInt32(reader[4].ToString());
49          entModel.dmIndicadores.AddObject(dmIndicadores_insert);
50
51      }
52
53      reader.Close();
54  }
55  catch (AdomdException ex)
56  {
57      SSASConnection.Close();
58      entModel.Connection.Close();
59      xmlError = returnErrorInXml(xmloParam, "NOK", "9", ex.Message);
60      xmlwsDM.AppendChild(xmlError);
61      xmloParam.AppendChild(xmlwsDM);
62      return xmloParam.OuterXml.ToString();
63  }
64
65  # endregion FIM Matriz de Classificacao do modelo
66
67  Grafico lift
104
105  // grava dados na bd
106  try
107  {
108      entModel.SaveChanges();
109  }
110  catch (EntityException ex)
111  {
112      SSASConnection.Close();
113      entModel.Connection.Close();
114      xmlError = returnErrorInXml(xmloParam, "NOK", "11", ex.Message);
115      xmlwsDM.AppendChild(xmlError);
116      xmloParam.AppendChild(xmlwsDM);
117      return xmloParam.OuterXml.ToString();
118  }
119  # endregion FIM Insere indicadores relativos ao periodo
```

Fig. 36 – Seleção dos dados estatísticos de MS SQL AS e inserção na tabela **dmIndicadores**

Após a atualização do modelo efetuada pelo código da [Fig. 35](#), o MS SQL AS gera um conjunto de informação estatística sobre a performance do modelo. Neste código, são executados dois procedimentos: um para obter a matriz de confusão - **SystemGetClassificationMatrix** (linha 28 a 33) - e outro para obter os registos para produção do gráfico Lift – **SystemGetLiftTable** (código omitido nas linhas 67 a 104). Os resultados obtidos pelos dois procedimentos são registados na tabela **dmIndicadores**. Na [Fig. 36](#) as linhas 37 a 48 preenchem os atributos, e a linha 49 adiciona o objeto à tabela.

Nas linhas 106 a 118 é concretizada a inserção dos registos na base de dados PROJFINAL.

Método *wsSelectModelStat*

Este método retorna a informação estatística – matriz de confusão e indicadores da **lift table** – da atualização do modelo ao longo dos períodos mensais em que o mesmo é calculado, seleccionando da tabela **dmIndicadores**, como ilustra o diagrama de atividades da [Fig. 37](#).

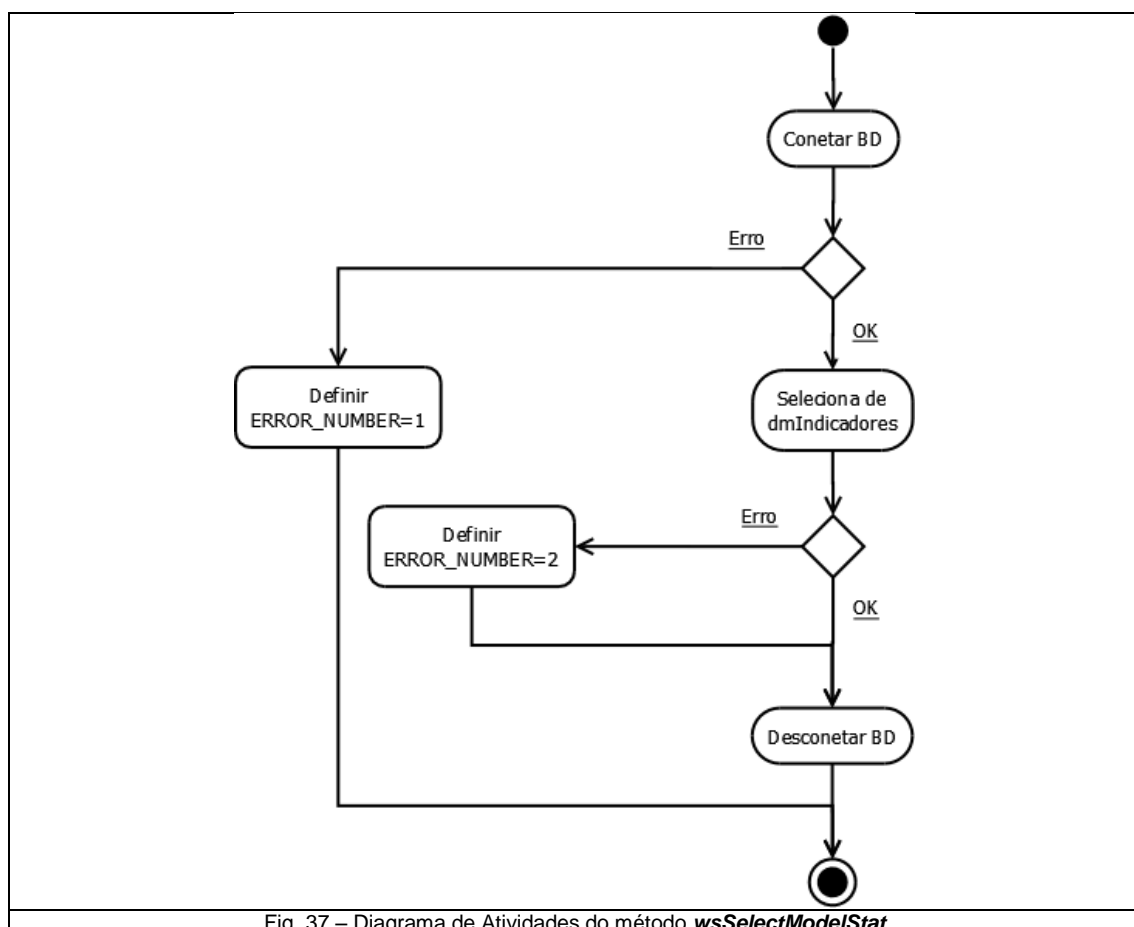


Fig. 37 – Diagrama de Atividades do método **wsSelectModelStat**

A [Tab. 21](#) apresenta o formato dos parâmetros de entrada e de saída do método.

Parâmetros de entrada (String XML)		
<HELP> </HELP>	Opção de retorno de informação de ajuda	Boolean 1 para True 0 para False
<START_DATE> </START_DATE>	Data a partir da qual se pretende atualizar o modelo	Date YYYYMMDD
<END_DATE> </END_DATE>	Data à qual se pretende atualizar o modelo – a atualização é efetuada até ao último dia do mês anterior	Date YYYYMMDD
Exemplo de documento XML a utilizar com parâmetros de entrada		
<pre> <WSDM> <HELP>1</HELP> <INPARAM> <START_DATE>2012-08-01</START_DATE> <END_DATE>2015-04-15</END_DATE> </INPARAM> </WSDM> </pre>		
Parâmetros de saída (XML)		
<HELP> </HELP>	Tag que encapsula a informação sobre os atributos de retorno <i>Parent tag: <WSDM></i>	
<OUTPARAM> </OUTPARAM>	Encapsula toda a informação estatística retornada <i>Parent tag: <WSDM></i>	
<MODEL> </MODEL>	Encapsula toda a informação referente ao modelo de um período <i>Parent tag: <OUTPARAM></i>	
<PERIOD> </PERIOD>	Período a que respeita o modelo e as respetivas estatísticas <i>Parent tag: <MODEL></i>	
<CONFUSION_MATRIX> </CONFUSION_MATRIX>	Encapsula os quatro indicadores da matriz de confusão <i>Parent tag: <MODEL></i>	
<CM_ENTRY> </CM_ENTRY>	Encapsula cada uma das combinações dos valores previstos e reais <i>Parent tag: <CONFUSION_MATRIX></i>	
<TYPE> </TYPE>	Assume um dos quatro seguintes valores: <i>Parent tag: <CM_ENTRY></i>	String CM False Negative CM True Negative CM False Positive CM True Positive
<VALUE> </VALUE>	Nº de registos obtidos referentes ao <TYPE> correspondente. O resultado obtido refere-se a dados de teste. <i>Parent tag: <CM_ENTRY></i>	
<LIFT_TABLE> </LIFT_TABLE>	Encapsula toda a informação sobre a Lift table <i>Parent tag: <MODEL></i>	
<LT_ENTRY> </LT_ENTRY>	Encapsula uma linha de informação da Lift Table <i>Parent tag: <LIFT_TABLE></i>	
<PERCENTILE> </PERCENTILE>	Apresenta o valor do percentil da Lift Table <i>Parent tag: <LT_ENTRY></i>	Double
<VALUE> </VALUE>	Apresenta o valor da Lift Table <i>Parent tag: <LT_ENTRY></i>	Double
<PROBABILITY> </PROBABILITY>	Apresenta a probabilidade da Lift Table <i>Parent tag: <LT_ENTRY></i>	Double
<ERROR> </ERROR>	Encapsula a informação sobre um erro fatal ocorrido na execução do método, ou a sua conclusão com sucesso <i>Parent tag: <WSDM></i>	
<ERROR_STATUS> </ERROR_STATUS>	Preenchido com OK ou NOK , caso não ocorra, ou ocorra um erro, respetivamente <i>Parent tag: <ERROR></i>	String OK / NOK
<ERROR_NUMBER> </ERROR_NUMBER>	Numero do erro (ver diagrama) <i>Parent tag: <ERROR></i>	Integer
<ERROR_MESSAGE> </ERROR_MESSAGE>	Descrição do erro tal como é retornado pela exceção <i>Parent tag: <ERROR></i>	String
Exemplo de documento XML com parâmetros de saída		
<pre> <WSDM> <HELP> <ATTRIBUTE> <ATTRIBUTE_NAME>ITEM</ATTRIBUTE_NAME> <ATTRIBUTE_TYPE>string</ATTRIBUTE_TYPE> </ATTRIBUTE> ...itens excluídos por questão de simplificação ... </pre>		

```

<ATTRIBUTE>
  <ATTRIBUTE_NAME>ERRORN</ATTRIBUTE_NAME>
  <ATTRIBUTE_TYPE>integer</ATTRIBUTE_TYPE>
</ATTRIBUTE>
</HELP>
<OUTPARAM>
  <MODEL>
    <PERIOD>201503</ PERIOD >
    <CONFUSION_MATRIX>
      <CM_ENTRY>
        <TYPE>CM False Negative</TYPE>
        <VALUE>1150</VALUE>
      </CM_ENTRY>
      ...itens excluídos por questão de simplificação ...
      <CM_ENTRY>
        <TYPE>CM True Positive</TYPE>
        <VALUE>4787</VALUE>
      </CM_ENTRY>
    </CONFUSION_MATRIX>
    <LIFT_TABLE>
      <LT_ENTRY>
        <PERCENTILE>
          0.997766195085629
        </PERCENTILE>
        <VALUE>
          2,2570321711302
        </VALUE>
        <PROBABILITY>
          0,999848771266541
        </PROBABILITY>
      </LT_ENTRY>
      ...itens excluídos por questão de simplificação ...
      <LT_ENTRY>
        <PERCENTILE>
          100
        </PERCENTILE>
        <VALUE>
          100
        </VALUE>
        <PROBABILITY>
          8,57191839533688E-05
        </PROBABILITY>
      </LT_ENTRY>
    </LIFT_TABLE>
  </MODEL>
</OUTPARAM>
<ERROR>
  <ERROR_STATUS>OK</ERROR_STATUS>
  <ERROR_NUMBER></ERROR_NUMBER>
  <ERROR_MESSAGE></ERROR_MESSAGE>
</ERROR>
</WSDM>

```

Tab. 21 – Formato dos documentos XML referentes a parâmetros de entrada e a atributos retornados pelo método **wsSelectModelStat**

Este procedimento não apresenta nenhum aspeto relevante em termos de código desenvolvido, pelo que se omite por completo a sua apresentação.

Método *wsSelectUserProbability*

Este método recebe valores correspondentes aos atributos utilizados no modelo, obtém as respetivas classificações e obtém, a partir de informações do modelo, via linguagem DMX, a probabilidade de um utente se tornar desistente.

A [Fig. 38](#) apresenta o diagrama de atividades do método.

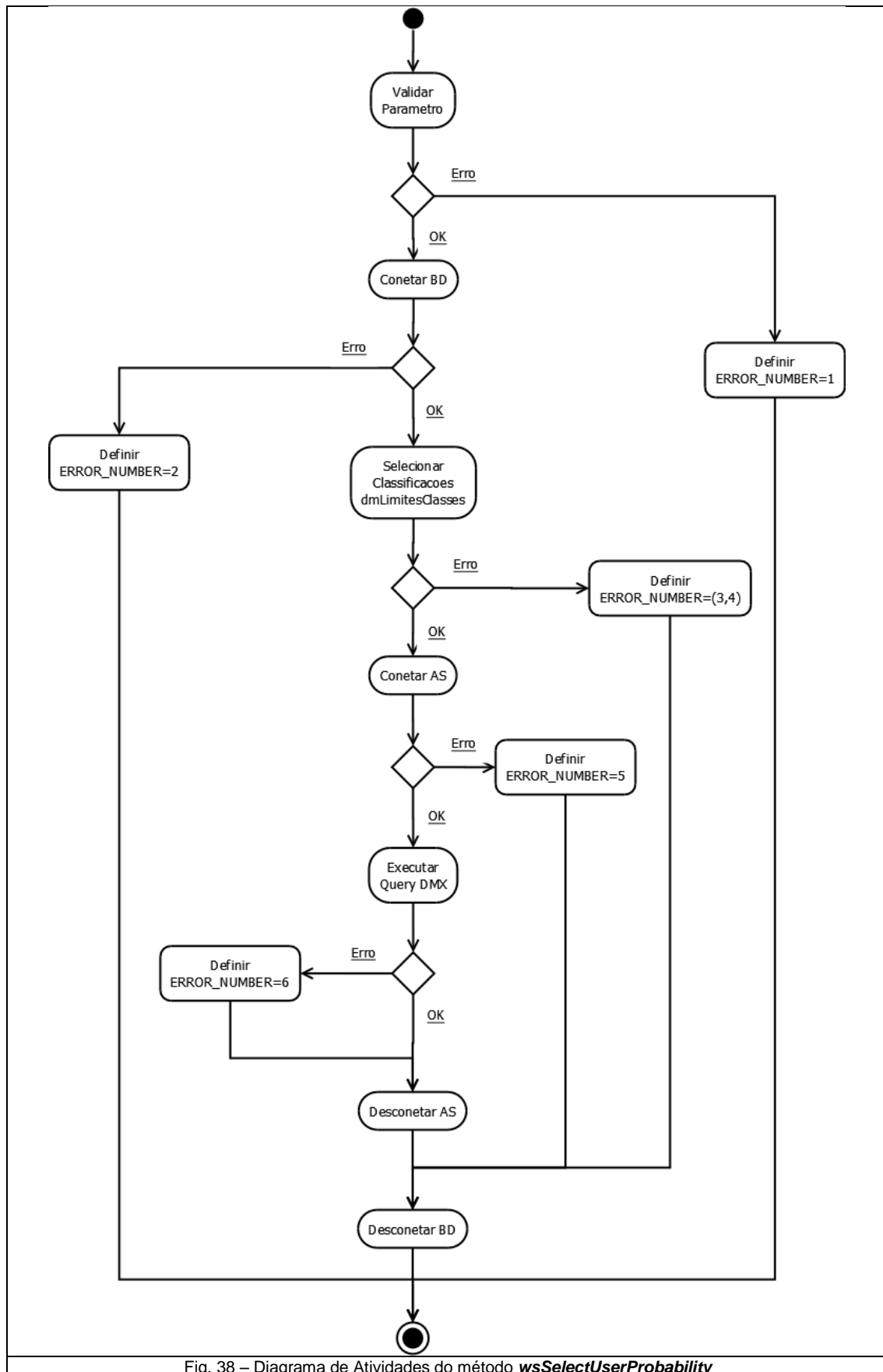


Fig. 38 – Diagrama de Atividades do método *wsSelectUserProbability*

A [Tab. 22](#) apresenta o formato dos parâmetros de entrada e de saída do método.

Parâmetros de entrada (String XML)		
<HELP> </HELP>	Opção de retorno de informação de ajuda	Boolean 1 para True 0 para False
<REGISTRATION_DATE> </REGISTRATION_DATE>	Data de inscrição do utente	Date YYYYMMDD
<TOTAL_FREQUENCY_DAYS_SINCE_REGISTRATION> </TOTAL_FREQUENCY_DAYS_SINCE_REGISTRATION>	Nº de dias de frequência do clube desde o início da inscrição	Integer
<GROUP_CLASSES_TOTAL_FREQUENCY> </GROUP_CLASSES_TOTAL_FREQUENCY>	Nº de Aulas de Grupo frequentadas desde o início da inscrição	Integer
<TOTAL_BILLING_SINCE_REGISTRATION> </TOTAL_BILLING_SINCE_REGISTRATION>	Total pago pelo Utente durante o período da inscrição	Double
<DAYS_WITHOUT_ATTENDING> </DAYS_WITHOUT_ATTENDING>	Nº de dias desde a última visita do utente ao Ginásio	Integer
Exemplo de documento XML a utilizar com parâmetros de entrada		
<pre> <WSDM> <HELP>1</HELP> <INPARAM> <REGISTRATION_DATE>2015-01-01</REGISTRATION_DATE> <TOTAL_FREQUENCY_DAYS_SINCE_REGISTRATION> 20 </TOTAL_FREQUENCY_DAYS_SINCE_REGISTRATION> <GROUP_CLASSES_TOTAL_FREQUENCY>3</GROUP_CLASSES_TOTAL_FREQUENCY> <TOTAL_BILLING_SINCE_REGISTRATION>179.94</TOTAL_BILLING_SINCE_REGISTRATION> <DAYS_WITHOUT_ATTENDING>6</DAYS_WITHOUT_ATTENDING> </INPARAM> </WSDM> </pre>		
Parâmetros de saída (XML)		
<HELP> </HELP>	Tag que encapsula a informação sobre os atributos de retorno <i>Parent tag:</i> <WSDM>	
<OUTPARAM> </OUTPARAM>	Encapsula toda a informação estatística retornada <i>Parent tag:</i> <WSDM>	
<GIVEUPUSER> </GIVEUPUSER>	Indica a classificação encontrada para os parâmetros especificados <i>Parent tag:</i> <OUTPARAM>	Boolean False / True
<PROBABILITY> </PROBABILITY>	Indica a probabilidade relativa ao atributo anterior para os parâmetros especificados <i>Parent tag:</i> <OUTPARAM>	Double
<TREE_NODE_DATA> </TREE_NODE_DATA >	Encapsula os valores da classe/nó para os atributos indicados <i>Parent tag:</i> <OUTPARAM>	
<GIVEUP_CLASS> </GIVEUP_CLASS>	Classe do atributo <i>Parent tag:</i> <TREE_NODE_DATA>	Boolean False / True
<CLASS_NRECORDS> </CLASS_NRECORDS>	Nº de registos com False / True (atributo anterior) <i>Parent tag:</i> <TREE_NODE_DATA>	Integer
<CLASS_PROBABILITY> </CLASS_PROBABILITY>	Probabilidade associada ao classe <i>Parent tag:</i> <TREE_NODE_DATA>	Double
<ERROR> </ERROR>	Encapsula a informação sobre um erro fatal ocorrido na execução do método, ou a sua conclusão com sucesso <i>Parent tag:</i> <WSDM>	
<ERROR_STATUS> </ERROR_STATUS>	Preenchido com OK ou NOK, caso não ocorra, ou ocorra um erro, respetivamente <i>Parent tag:</i> <ERROR>	String OK / NOK
<ERROR_NUMBER> </ERROR_NUMBER>	Numero do erro (ver diagrama) <i>Parent tag:</i> <ERROR>	Integer
<ERROR_MESSAGE> </ERROR_MESSAGE>	Descrição do erro tal como é retornado pela exceção <i>Parent tag:</i> <ERROR>	String
Exemplo de documento XML com parâmetros de saída		
<pre> <WSDM> <HELP> </pre>		

<pre> <ATTRIBUTE> <ATTRIBUTE_NAME>GIVEUPUSER</ATTRIBUTE_NAME> <ATTRIBUTE_TYPE>string</ATTRIBUTE_TYPE> </ATTRIBUTE> ...itens excluídos por questão de simplificação ... <ATTRIBUTE> <ATTRIBUTE_NAME>CLASS_PROBABILITY</ATTRIBUTE_NAME> <ATTRIBUTE_TYPE>double</ATTRIBUTE_TYPE> </ATTRIBUTE> </HELP> <OUTPARAM> <CLASSIFICACAO> <REGISTRATION_PERIOD> <CLASS>18</CLASS> <VALUE>5</VALUE> </REGISTRATION_PERIOD> <DAYS_WITHOUT_ATTENDING> <CLASS>1</CLASS> <VALUE>1</VALUE> </DAYS_WITHOUT_ATTENDING> <BUSINESS_AMOUNT> <CLASS>1</CLASS> <VALUE>11</VALUE> </BUSINESS_AMOUNT> <GROUP_CLASS_FREQUENCY> <CLASS>4</CLASS> <VALUE>1</VALUE> </GROUP_CLASS_FREQUENCY> <MEAN_ATTENDING> <CLASS>3</CLASS> <VALUE>0,01</VALUE> </MEAN_ATTENDING> </CLASSIFICACAO> <RESULTADO> <GIVEUPUSER> True </GIVEUPUSER> <PROBABILITY> 0.8456645556 </PROBABILITY> <TREE_NODE_DATA> <GIVEUP_CLASS> False </GIVEUP_CLASS> <CLASS_NRECORDS> 112 </CLASS_NRECORDS> <CLASS_PROBABILITY> 0.1544555666 </CLASS_PROBABILITY> <GIVEUP_CLASS> True </GIVEUP_CLASS> <CLASS_NRECORDS> 345 </CLASS_NRECORDS> <CLASS_PROBABILITY> 0.8456645556 </CLASS_PROBABILITY> </TREE_NODE_DATA> </RESULTADO> </OUTPARAM> <ERROR> <ERROR_STATUS>OK</ERROR_STATUS> <ERROR_NUMBER></ERROR_NUMBER> <ERROR_MESSAGE></ERROR_MESSAGE> </ERROR> </WSDM> </pre>	
<p>Tab. 22 – Formato dos documentos XML referentes a parâmetros de entrada e a atributos retornados pelo método <i>wsSelectUserProbability</i></p>	

Este método apresenta uma característica interessante em termos de código, dado que executa um query DMX⁸ – um “natural join” -, que permite obter o resultado pretendido. O query pode ser executado diretamente no MS SQL AS, e apresenta a forma do exemplo seguinte:

```
1 SELECT [Ute Classedesistencia] AS Desistente,  
2       PredictHistogram([Ute Classedesistencia]) AS Statistics,  
3       PredictProbability([Ute Classedesistencia])  
4 FROM [Dm Retencao AllClass-GeneroIdade]  
5 NATURAL PREDICTION JOIN  
6 (SELECT '1' AS [Ute Cnaulasgrupo],  
7        '5' AS [Ute Cmesesinscricao],  
8        '2' AS [Ute Cdiassemfrecuencia],  
9        '4' AS [Ute Cfrequencia])  
10 AS [t]
```

Fig. 39 – Query DMX para obter a classe, a probabilidade e o histograma para o valor dos atributos indicados

Note-se que os valores 1, 5, 2 e 4 são os valores para as classes das quais se pretende obter a probabilidade de desistência.

A [Fig. 40](#) apresenta o código que implementa o referido query:

```
1 # region Executa o query e obtém % de probabilidade  
2 // Query DMX  
3 cmd.CommandText = "SELECT [Ute Classedesistencia] AS Desistente, " +  
4 "PredictProbability([Ute Classedesistencia]) AS Probabilidade, " +  
5 "PredictHistogram([Ute Classedesistencia]) AS Estatistica " +  
6 "FROM [Dm Retencao AllClass-GeneroIdade] " +  
7 "NATURAL PREDICTION JOIN " +  
8 "(SELECT '" + cMesesInscricao.ToString() + "' AS [Ute Cmesesinscricao] ";  
9  
10 xmlClass = xmloParam.CreateElement("CLASS");  
11 xmlClass.InnerText = mesesInscricao.ToString();  
12 xmlValue = xmloParam.CreateElement("VALUE");  
13 xmlValue.InnerText = cMesesInscricao.ToString();  
14 xmlClassEntry = xmloParam.CreateElement("REGISTRATION_PERIOD");  
15 xmlClassEntry.AppendChild(xmlClass);  
16 xmlClassEntry.AppendChild(xmlValue);  
17 xmlClassificacao.AppendChild(xmlClassEntry);  
18  
19 Adicao de Parametros Opcionais  
20  
21 cmd.CommandText = cmd.CommandText + ") AS [t]";  
22  
23 cmd.Connection = SSASConnection;  
24  
25 try  
26 {  
27     reader = cmd.ExecuteReader();  
28     while (reader.Read())  
29     {  
30         xmlClass = xmloParam.CreateElement("GIVEUPUSER");  
31         xmlClass.InnerText = reader[0].ToString();  
32  
33         xmlValue = xmloParam.CreateElement("PROBABILITY");  
34         xmlValue.InnerText = reader[1].ToString();  
35  
36         xmlResultado.AppendChild(xmlClass);  
37         xmlResultado.AppendChild(xmlValue);  
38  
39         xmlClassEntry = xmloParam.CreateElement("TREE_NODE_DATA");  
40  
41         subreader = reader.GetDataReader(2);  
42         while (subreader.Read())  
43         {  
44             giveUpUser = Convert.ToBoolean(subreader[0]);  
45             xmlClass = xmloParam.CreateElement("GIVEUP_CLASS");
```

⁸ DMX – Data Mining Extensions é uma linguagem do MS SQL AS que pode ser utilizada para criar modelos, treiná-los e para obter a informação de previsão relativa às classes que se pretendem prever

116	xmlClass.InnerText = giveUpUser.ToString();
117	
118	xmlValue = xmloParam.CreateElement("CLASS_NRECORDS");
119	xmlValue.InnerText = subreader[1].ToString();
120	
121	xmlProb = xmloParam.CreateElement("CLASS_PROBABILITY");
122	xmlProb.InnerText = subreader[2].ToString();
123	
124	xmlClassEntry.AppendChild(xmlClass);
125	xmlClassEntry.AppendChild(xmlValue);
126	xmlClassEntry.AppendChild(xmlProb);
127	
128	}
129	xmlResultado.AppendChild(xmlClassEntry);
130	}
131	reader.Close();
132	}
133	catch (AdomdException ex)
134	{
135	SSASConnection.Close();
136	entModel.Connection.Close();
137	xmlError = returnErrorInXml(xmloParam, "NOK", "6", ex.Message);
138	xmlwsDM.AppendChild(xmlError);
139	xmloParam.AppendChild(xmlwsDM);
140	return xmloParam.OuterXml.ToString();
141	}
142	
143	# endregion FIM Executa o query e obtem % de probabilidade

Fig. 40 – Código para seleção dos dados da folha da Árvore de Decisão correspondente aos atributos indicados

Salienta-se os seguintes aspetos:

- Em linhas anteriores de código não apresentado, os valores são classificados através da execução do procedimento **dmLimitesClasses** (ver Fig. 38);
- As linhas 3 a 8 apresentam a construção inicial do comando, ao qual são depois concatenadas as classificações obtidas anteriormente (código também omitido – linhas 19 a 87). Nota-se que se o parâmetro não for indicado, o atributo não é concatenado;
- Por fim o comando é executado (linha 96), e o resultado colocado no objeto reader do qual depois são extraídos os valores a colocar no documento XML (linhas 97 a 130);
- O ciclo WHILE interior (linhas 111 a 128) extraem o histograma correspondente aos dados da folha / nó onde o modelo obteve a probabilidade e a classificação.

Método wsSelectGiveUpUsers

O último método analisado é o que produz um relatório com a lista dos Utentes Ativos de um Clube que se encontram acima de uma determinada probabilidade de se virem a tornar desistentes.

O Clube e a probabilidade são indicados como parâmetros ao serviço web.

A lógica e parâmetros de entrada e saída são especificado pela [Fig. 41](#) e pela [Tab. 23](#).

Este método não apresenta novidades relativamente aos desenvolvimentos já apresentados, uma vez que se limita a executar um procedimento que retorna todos os Utentes Ativos de um determinado clube e os seus atributos relevantes para a aplicação do modelo, juntamente com as respetivas classificações.

Por cada registo retornado pelo procedimento, é aplicado o query DMX especificado no método anterior, para determinar a classe de desistência e a respetiva probabilidade. Se a classe e a probabilidade corresponderem ao parâmetro de entrada, o utente é incluído no documento XML a retornar.

Parâmetros de entrada (String XML)		
<HELP> </HELP>	Opção de retorno de informação de ajuda	Boolean 1 para True 0 para False
<CLUB> </CLUB>	Clube do qual se pretende listar os utentes potenciais desistentes	String
<PROBABILITY> </PROBABILITY>	Probabilidade a considerar	Double
Exemplo de documento XML a utilizar com parâmetros de entrada		
<WSDM> <HELP>1</HELP> <INPARAM> <CLUB>10</CLUB> <PROBABILITY>40</PROBABILITY> </INPARAM> </WSDM>		
Parâmetros de saída (XML)		
<HELP> </HELP>	Tag que encapsula a informação sobre os atributos de retorno <i>Parent tag:</i> <WSDM>	
<OUTPARAM> </OUTPARAM>	Encapsula toda a informação estatística retornada <i>Parent tag:</i> <WSDM>	
<GIVEUPUSER> </GIVEUPUSER>	Indica a classificação encontrada para os parâmetros especificados <i>Parent tag:</i> <OUTPARAM>	Boolean False / True
<PROBABILITY> </PROBABILITY>	Indica a probabilidade relativa ao atributo anterior para os parâmetros especificados <i>Parent tag:</i> <OUTPARAM>	Double
<TREE_NODE_DATA> </TREE_NODE_DATA>	Encapsula os valores da classe/nó para os atributos indicados <i>Parent tag:</i> <OUTPARAM>	
<GIVEUP_CLASS> </GIVEUP_CLASS>	Classe do atributo <i>Parent tag:</i> <TREE_NODE_DATA>	Boolean False / True
<CLASS_NRECORDS> </CLASS_NRECORDS>	Nº de registos com False / True (atributo anterior) <i>Parent tag:</i> <TREE_NODE_DATA>	Integer
<CLASS_PROBABILITY> </CLASS_PROBABILITY>	Probabilidade associada a classe <i>Parent tag:</i> <TREE_NODE_DATA>	Double
<ERROR> </ERROR>	Encapsula a informação sobre um erro fatal ocorrido na execução do método, ou a sua conclusão com sucesso <i>Parent tag:</i> <WSDM>	
<ERROR_STATUS> </ERROR_STATUS>	Preenchido com OK ou NOK, caso não ocorra, ou ocorra um erro, respetivamente <i>Parent tag:</i> <ERROR>	String OK / NOK
<ERROR_NUMBER> </ERROR_NUMBER>	Número do erro (ver diagrama) <i>Parent tag:</i> <ERROR>	Integer

<ERROR_MESSAGE> </ERROR_MESSAGE>	Descrição do erro tal como é retornado pela exceção <i>Parent tag: <ERROR></i>	String
Exemplo de documento XML com parâmetros de saída		
<pre> <WSDM> <HELP> <ATTRIBUTE> <ATTRIBUTE_NAME>USER_NAME</ATTRIBUTE_NAME> <ATTRIBUTE_TYPE>string</ATTRIBUTE_TYPE> </ATTRIBUTE> ...itens excluídos por questão de simplificação ... <ATTRIBUTE> <ATTRIBUTE_NAME>GIVEUP_PROBABILITY</ATTRIBUTE_NAME> <ATTRIBUTE_TYPE>double</ATTRIBUTE_TYPE> </ATTRIBUTE> </HELP> <OUTPARAM> <GIVEUP_USER> <USER_NAME> Paulo Sergio Nunes Pinheiro </USER_NAME> <USER_IDENTIFICATION> 112 </USER_IDENTIFICATION> <REGISTRATION_PERIOD> 45 </REGISTRATION_PERIOD> <REGISTRATION_PERIOD_CLASS> 3 </REGISTRATION_PERIOD_CLASS> <DAYS_WITHOUT_ATTENDING> 7 </DAYS_WITHOUT_ATTENDING> <DAYS_WITHOUT_ATTENDING_CLASS> 1 </DAYS_WITHOUT_ATTENDING_CLASS> <MEAN_FREQUENCY> 4.5 </MEAN_FREQUENCY> <MEAN_FREQUENCY_CLASS> 5 </MEAN_FREQUENCY_CLASS> <GROUP_CLASS_FREQUENCY> 45 </GROUP_CLASS_FREQUENCY> <GROUP_CLASS_FREQUENCY_CLASS> 45 </GROUP_CLASS_FREQUENCY_CLASS> <TOTAL_BUSINESS> 235.00 </TOTAL_BUSINESS> <TOTAL_BUSINESS_CLASS> 2 </TOTAL_BUSINESS_CLASS> <GIVEUP_PROBABILITY> 0,45566566566 </GIVEUP_PROBABILITY> </GIVEUP_USER> ...itens excluídos por questão de simplificação ... <GIVEUP_USER> ...itens excluídos por questão de simplificação ... </GIVEUP_USER> </OUTPARAM> <ERROR> <ERROR_STATUS>OK</ERROR_STATUS> <ERROR_NUMBER></ERROR_NUMBER> <ERROR_MESSAGE></ERROR_MESSAGE> </ERROR> </WSDM> </pre>		
Tab. 23 – Formato dos documentos XML referentes a parâmetros de entrada e a atributos retornados pelo método wsSelectGiveUpUsers		

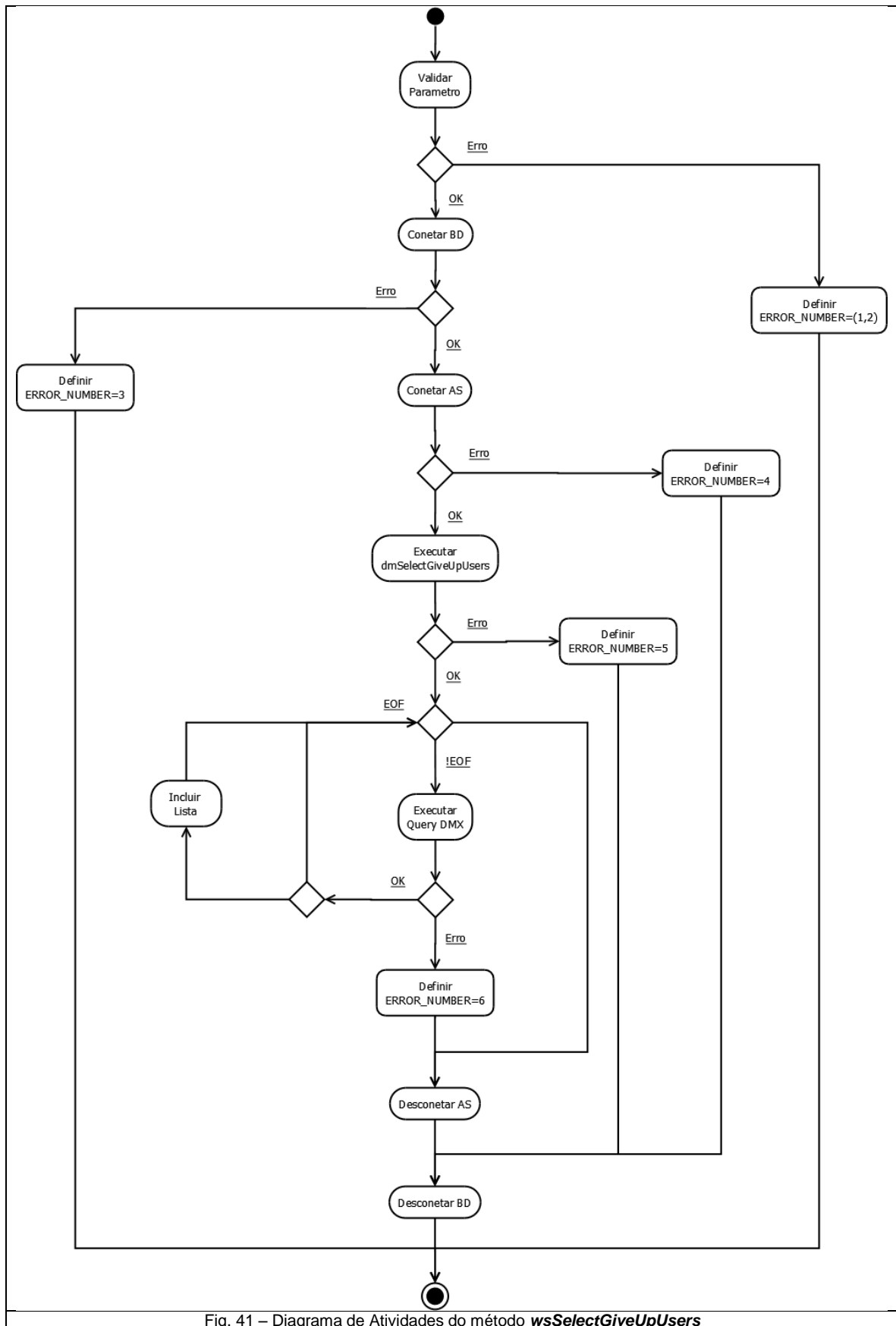


Fig. 41 – Diagrama de Atividades do método ***wsSelectGiveUpUsers***

Execução e Teste dos Serviços Web

O teste dos métodos incluídos foi efetuado executando os mesmos diretamente no ambiente de desenvolvimento Visual Studio, como ilustrado na [Fig. 43](#).

Na execução direta dos serviços web salienta-se a apresentação e explicação dos respetivos parâmetros de entrada, que resulta do código apresentado na [Fig. 42](#).

```
[WebMethod(Description = "<table><tr><td colspan='4'>Select probability to quit given some attributes from the user</td><tr> +
    "<tr><td colspan='4'>Parameters list:</td><tr> +
    "<tr><td>HELP</td><td>Flag to return help node with output structure, field type and translated field names</td><td>1 for True,0 for False</td><td>Optional</td></tr> +
    "<tr><td>REGISTRATION_DATE</td><td>Number of months of registration</td><td>Integer</td><td>Mandatory</td></tr> +
    "<tr><td>TOTAL_FREQUENCY_DAYS_SINCE_REGISTRATION</td><td>Number of average weekly frequency</td><td>Integer</td><td>Optional</td></tr> +
    "<tr><td>GROUP_CLASSES_TOTAL_FREQUENCY</td><td>Number of attended group classes for the registration period</td><td>Integer</td><td>Optional</td></tr> +
    "<tr><td>TOTAL_BILLING_DAYS_SINCE_REGISTRATION</td><td>Total billing during the registration period</td><td>Float</td><td>Optional</td></tr> +
    "<tr><td>DAYS_WITHOUT_ATTENDING</td><td>Number of days that do not attend the club</td><td>Integer</td><td>Optional</td></tr> +
    "</table>"]
public string wsSelectUserProbability(string striParam)...

[WebMethod(Description = "<table><tr><td colspan='4'>Select the users with probability for give up</td><tr> +
    "<tr><td colspan='4'>Parameters list:</td><tr> +
    "<tr><td>HELP</td><td>Flag to return help node with output structure, field type and translated field names</td><td>1 for True,0 for False</td><td>Optional</td></tr> +
    "<tr><td>CLUB</td><td>Club to analyse</td><td>String</td><td>Mandatory</td></tr> +
    "<tr><td>PROBABILITY</td><td>Probability of Giving Up</td><td>Float</td><td>Optional</td></tr> +
    "</table>"]
public string wsSelectGiveUpUsers(string striParam)...
```

Fig. 42 – Código necessário para a apresentação da explicação dos parâmetros de entrada na execução do serviço web em teste

The screenshot shows a web browser window with the address bar displaying 'http://localhost:43846/wsDM.aspx'. The page title is 'WsDM'. The main content area lists supported operations and their parameters.

The following operations are supported. For a formal definition, please review the [Service Description](#).

- [wsDataAndModelUpdate](#)**
 Update the data needed to Data Mining model
 Parameters list:

HELP	Flag to return help node with output structure, field type and translated field names	1 for True,0 for False	Optional
START_DATE	Initial date to consider - 20120801 by default	YYYYMMDD	Optional
END_DATE	Final date to determine mining model (wsDM will consider last day of previous month) - today's date by default	YYYYMMDD	Optional
- [wsSelectGiveUpUsers](#)**
 Select the users with probability for give up
 Parameters list:

HELP	Flag to return help node with output structure, field type and translated field names	1 for True,0 for False	Optional
CLUB	Club/Unit to analyse	string	Mandatory
PROBABILITY	Probability of Giving Up	Float	Optional
- [wsSelectModelStat](#)**
 Select Statistics from the Data Mining model
 Parameters list:

HELP	Flag to return help node with output structure, field type and translated field names	1 for True,0 for False	Optional
START_DATE	Initial date to consider - 20120801 by default	YYYYMMDD	Optional
END_DATE	Final date (wsDM will consider last day of previous month) - today's date by default	YYYYMMDD	Optional
- [wsSelectUserProbability](#)**
 Select probability to quit given some attributes from the user
 Parameters list:

HELP	Flag to return help node with output structure, field type and translated field names	1 for True,0 for False	Optional
REGISTRATION_DATE	Number of months of registration	Integer	Mandatory
TOTAL_FREQUENCY_DAYS_SINCE_REGISTRATION	Number of average weekly frequency	Integer	Optional
GROUP_CLASSES_TOTAL_FREQUENCY	Number of attended group classes for the registration period	Integer	Optional
TOTAL_BILLING_SINCE_REGISTRATION	Total billing during the registration period	Float	Optional
DAYS_WITHOUT_ATTENDING	Number of days that do not attend the club	Integer	Optional

A seleção de um método invoca o formulário de teste de um método, como ilustrado na [Fig. 44](#).

The screenshot shows a web browser window with the URL `http://localhost:43846/wsDM.aspx?o`. The page title is "WsDM Web Service". The main heading is "wsSelectUserProbability". Below it, there is a link to a complete list of operations. The "Parameters list:" section contains a table with the following data:

Parameter	Description	Type	Optional
HELP	Flag to return help node with output structure, field type and translated field names	Integer	Mandatory
REGISTRATION_DATE	Number of months of registration	Integer	Optional
TOTAL_FREQUENCY_DAYS_SINCE_REGISTRATION	Number of average weekly frequency	Integer	Optional
GROUP_CLASSES_TOTAL_FREQUENCY	Number of attended group classes for the registration period	Integer	Optional
TOTAL_BILLING_SINCE_REGISTRATION	Total billing during the registration period	Float	Optional
DAYS_WITHOUT_ATTENDING	Number of days that do not attend the club	Integer	Optional

The "Test" section includes a description: "To test the operation using the HTTP POST protocol, click the 'Invoke' button." Below this is a form with a "Parameter" label and a "Value" input field. The input field contains the XML string: `<WSDM><HELP>1</HELP><INPARAM><REGISTF X`. An "Invoke" button is next to the input field.

The "SOAP 1.1" section shows a sample SOAP 1.1 request and response. The request is a POST to `/wsDM.aspx HTTP/1.1` with the following headers:

```
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://tempuri.org/wsSelectUserProbability"
```

The request body is an XML document with the following structure:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <wsSelectUserProbability xmlns="http://tempuri.org/">
      <striParam><string></string></striParam>
    </wsSelectUserProbability>
  </soap:Body>
</soap:Envelope>
```

Fig. 44 – Formulário de entrada dos parâmetros para a execução do método **wsSelectUserProbability**

A indicação de uma string em formato XML no campo **striParam** provoca a execução do método e o retorno do procedimento. Um exemplo é apresentado na Fig. 45.

The screenshot shows the same web browser window as Fig. 44, but now displaying the result of the execution. The result is an XML document with the following structure:

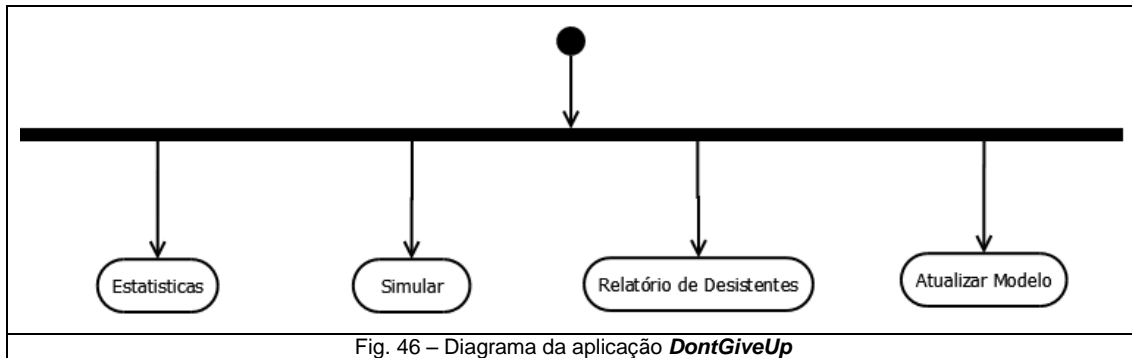
```
<?xml version="1.0" encoding="UTF-8"?>
<string
  xmlns="http://tempuri.org/"><WSDM><OUTPARAM><CLASSIFICACAO><REGISTRATION_PERIOD><CLASS>2</CLASS><VALUE>1</VA
```

Fig. 45 – Exemplo do resultado da execução do método **wsSelectUserProbability**

A camada de apresentação: os formulários

Para a implementação da camada de apresentação, optou-se por uma implementação simples e essencialmente gráfica, que permita ao utilizador analisar rapidamente a performance do modelo, e obter os resultados pretendidos.

Para atingir este objetivo, a aplicação apresenta 4 opções de trabalho, como ilustrado na Fig. 46.



Na realidade, cada uma das opções corresponde à chamada de um método do serviço web **wsDM** explicado nos pontos anteriores, pelo que a aplicação é extremamente simples, e limita-se a obter os resultados retornados pelos métodos em XML e preencher ou ativar os objetos escolhidos para a apresentação.

Dada a natureza dos dados a apresentar, foram escolhidos objetos do tipo grelha (Grid) para apresentar listas de dados e tabelas (Table) para apresentar dados simples e limitados. As taxas e a evolução de dados estatísticos são apresentadas através de objetos gráficos adequados. São também utilizados campos para “input” de dados por parte do utilizador, sempre que tal é requerido.

Para acelerar o desenvolvimento e criar um interface mais apelativo, utilizaram-se componentes de um produtor conhecido no mercado (Telerik), nomeadamente o conjunto de componentes que fazem parte do pacote UI for ASP.NET AJAX.

O código da aplicação está definido em 3 ficheiros:

- **dontgiveup.css**, que especifica a forma e propriedades dos elementos apresentados no formulário;
- **Default.aspx**, que contém o código fonte que define o interface - elementos HTML e scripts de código em JavaScript – e que se executa do lado do cliente;
- **Default.aspx.cs**, que contém o código fonte em linguagem C# que define a funcionalidade dos eventos e que se executa do lado do servidor;

Dada a pequena dimensão da aplicação a abordagem seguida foi a de criar todos os objetos na mesma “página” em 4 áreas principais, correspondentes a cada opção do programa. Sempre que uma opção é escolhida, a área

correspondente à opção escolhida é tornada visível, e as áreas das restantes opções são escondidas. Esta situação é conseguida através de funções JavaScript implementadas diretamente no ficheiro **Default.aspx**, como apresentado na [Fig. 47](#).

```
<head runat="server">
  <title>DontGiveUp</title>
  <link rel="stylesheet" type="text/css" href="Styles/dontgiveup.css">
  <script type="text/javascript">
    function employeeMouseOver(sender) {
      sender.className += " rlvHover";
    }
    function employeeMouseOut(sender) {
      sender.className = sender.className.replace(" rlvHover", "");
    }
    function toggleDivs() {
      var hdn = document.getElementById('hdnDiv');
      if (hdn != null && hdn.value != null && hdn.value != "") {
        var i = (hdn != null && hdn.value != null && hdn.value != "" ? hdn.value : 1);

        document.getElementById("DIVEstadisticas").style.display = (i == 1 ? "inline-block" : "none");
        document.getElementById("tdEstadistica").className = (i == 1 ? "rlvSelected" : "rlvItem");

        document.getElementById("DIVSimular").style.display = (i == 2 ? "inline-block" : "none");
        document.getElementById("tdSimular").className = (i == 2 ? "rlvSelected" : "rlvItem");

        document.getElementById("DIVRelatorioDesistentes").style.display = (i == 3 ? "inline-block" : "none");
        document.getElementById("tdRelatorioDesistentes").className = (i == 3 ? "rlvSelected" : "rlvItem");

        document.getElementById("DIVAtualizarModelo").style.display = (i == 4 ? "inline-block" : "none");
        document.getElementById("tdAtualizarModelo").className = (i == 4 ? "rlvSelected" : "rlvItem");
      }
    }
    function employeeMouseOutSelected(sender) {
      sender.className = "rlvSelected";
    }
  </script>
</head>
```

Fig. 47 – Funções JavaScript para esconder/apresentar as zonas correspondentes a cada opção
(Ficheiro Default.aspx)

As várias áreas são elementos **<div>** com identificação própria, que incluem todos os restantes objetos necessários à correta funcionalidade da respetiva opção.

São ainda áreas de relevo as **<div>** da classe “**box**” - que inclui todos os objetos da aplicação - e a da classe “**rlv**” - que inclui o menu da aplicação implementado através de uma tabela (elemento **<table>** com 1 linha – elemento **<tr>** - e 4 células – elementos **<td>**).

A simulação de menu, alternando a cor quando o cursor passa por cima da célula e quando a opção está selecionada é conseguida também através das funções apresentadas na [Fig. 47](#). As classes de estilo estão definidas no ficheiro **dontgiveup.css** conforme apresentadas na [Fig. 48](#).


```
.rlvItem, .rlvSelected
{
    width: 25%;
    height: 87px;
}

.rlvSelected a
{
    display: block;
    width: 100%;
    height: 77px;
    background-color: Silver;
    color: #fff;
    text-decoration: none;
    padding-top: 10px;
    position: relative;
    font-size: 22px;
    text-align: left;
    text-indent: 10px;
}

.rlvItem a
{
    display: block;
    width: 100%;
    height: 77px;
    background-color: #111;
    color: Silver;
    text-decoration: none;
    padding-top: 10px;
    position: relative;
    font-size: 22px;
    text-align: left;
    text-indent: 10px;
}

.rlvHover a
{
    background-color: #88B1FC;
    color: White;
}
```

Fig. 48 – Classes definidas para implementação do menu da aplicação (Ficheiro dontgiveup.css)

As áreas que contêm os objetos correspondentes a cada uma das opções estão dentro de um elemento `<div>` que as inclui e tem a classe de estilo “**content**”. Por sua vez, a `<div>` do estilo “**box**” permite definir as cores de fundo.

```
<body>
<form id="DontGiveUpFrm" runat="server">
<asp:HiddenField ID="hdnDiv" ClientIDMode="Static" runat="server" />
<telerik:RadScriptManager ID="RadScriptManager1" runat="server" AsyncPostBackTimeout="84000">
</telerik:RadScriptManager>
<telerik:RadFormDecorator ID="RadFormDecorator1" runat="server" ResolvedRenderMode="Classic"
Skin="Black" />
<div class="box">
<div style="display: inline-block; height: 100px;">
<h1>
Dont Give Up
</h1>
</div>
<div style="display: inline-block;">
<h3 style="margin-left: 20px;">
Implementação de um modelo de Data Mining para determinação de padrões de desistência<br />
</h3>
</div>
</div class="rlv">...</div>
<div class="content">
<telerik:RadAjaxPanel ID="RADGeral" runat="server" Width="100%">
<div class="DIVEstatisticas">...</div>
<div class="DIVSimular">...</div>
<div class="DIVRelatorioDesistentes" class="filtros" ...>...</div>
<div class="DIVAtualizarModelo" class="atualizar" ...>...</div>
</telerik:RadAjaxPanel>
</div>
</div class="footer">...</div>
</div>
</form>
</body>
```

Fig. 49 – Áreas de apresentação de dados da aplicação **DongGiveUp** (Ficheiro Default.aspx)

Analisados os aspetos básicos da estrutura da aplicação, passamos agora à análise do código associado. Como já foi referido, a aplicação obtém os dados essencialmente dos serviços web implementados. Uma vez que o processo é idêntico, limitaremos a explicação da implementação ao formulário inicial que apresenta os dados estatísticos sobre os modelos.

O formulário de arranque da aplicação tem a apresentação da [Fig. 50](#). Para melhor compreensão do exposto, numerou-se as áreas específicas que apresentam diferentes formas de carregamento.

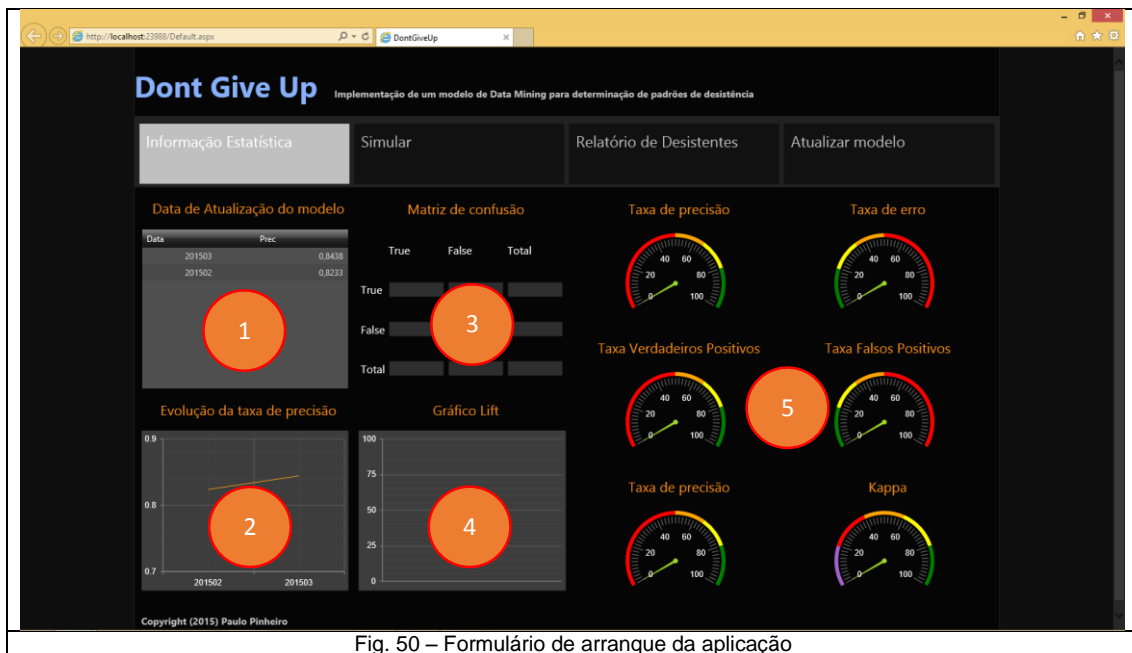


Fig. 50 – Formulário de arranque da aplicação

Ao entrar na aplicação, é despoletado um evento designado por **Page_Load** que corresponde ao carregamento do formulário. Neste evento é chamada a função **loadStats** que contém o código para carregamento da grelha da zona 1 da [Fig. 50](#).

A função **loadStats**, apresentada na [Fig. 51](#), invoca o método **wsSelectModelStat** que retorna a informação estatística do modelo referente aos períodos já calculados. Uma vez que o serviço web necessita, como parâmetros de entrada, a data de início e de término, a função indica a data corrente como a data final, e subtrai 6 meses à data corrente para atribuir a data inicial (linhas 12 a 16).

```
1 private void loadStats()
2 {
3     XmlDocument xmlDocParamIn = new XmlDocument();
4     XmlDocument paramOutXml = new XmlDocument();
5     XElement xelDontGiveUp = new XElement("WSDM", "");
6     XElement xelINPARAM = new XElement("INPARAM");
7     XElement xelHELP = new XElement("HELP", 0);
8     xelDontGiveUp.Add(xelHELP);
9
10    DateTime final = DateTime.Today;
11    DateTime inicial = final.AddMonths(-6);
12    XElement xelSTARTDATE = new XElement("START_DATE", inicial.ToShortDateString());
13    xelINPARAM.Add(xelSTARTDATE);
14    XElement xelENDDATE = new XElement("END_DATE", final.ToShortDateString());
15    xelINPARAM.Add(xelENDDATE);
16    xelDontGiveUp.Add(xelINPARAM);
17
18    using (var wsSimular = new wsDM.WsDM())
19    {
20        var wsSelectModelStat_ResultXml = wsSimular.wsSelectModelStat(xelDontGiveUp.ToString());
21        paramOutXml.LoadXml(wsSelectModelStat_ResultXml.ToString());
22        if (paramOutXml.HasChildNodes && paramOutXml.GetElementsByTagName("ERROR_STATUS").Count == 1)
23        {
24            var errStatus = paramOutXml.GetElementsByTagName("ERROR_STATUS")[0];
25            if (errStatus.InnerText.Equals("OK", StringComparison.OrdinalIgnoreCase) &&
26                paramOutXml.GetElementsByTagName("OUTPARAM").Count == 1)
27            {
28                modelsList = new List<modelo>();
29                XmlNamespaceManager nsm = new XmlNamespaceManager(paramOutXml.NameTable);
30                XmlNodeList modelsXml = paramOutXml.SelectNodes("//OUTPARAM/MODEL", nsm);
31                foreach (XmlNode _modelsNode in modelsXml)
32                {
33                    modelo modeloPeriodo = new modelo();
34                    foreach (XmlNode _psNode in _modelsNode.ChildNodes)
35                    {
36                        if (_psNode.Name == "PERIOD")
37                        {
38                            modeloPeriodo.Anomes = _psNode.InnerText.ToString();
39                        }
40                        else if (_psNode.Name == "CONFUSION_MATRIX")
41                        {
42                            modeloPeriodo.Cm = new confusionMatrix();
43                            foreach (XmlNode _cmNode in _psNode)
44                            {
45                                if (_cmNode.ChildNodes[0].InnerText.ToString() == "CM True Positive")
46                                    modeloPeriodo.Cm.TP =
47                                        Convert.ToInt32(_cmNode.ChildNodes[1].InnerText.ToString());
48                                else if (_cmNode.ChildNodes[0].InnerText.ToString() == "CM False Positive")
49                                    modeloPeriodo.Cm.FP =
50                                        Convert.ToInt32(_cmNode.ChildNodes[1].InnerText.ToString());
51                                else if (_cmNode.ChildNodes[0].InnerText.ToString() == "CM True Negative")
52                                    modeloPeriodo.Cm.TN =
53                                        Convert.ToInt32(_cmNode.ChildNodes[1].InnerText.ToString());
54                                else if (_cmNode.ChildNodes[0].InnerText.ToString() == "CM False Negative")
55                                    modeloPeriodo.Cm.FN =
56                                        Convert.ToInt32(_cmNode.ChildNodes[1].InnerText.ToString());
57                            }
58                        }
59                        else if (_psNode.Name == "LIFT_TABLE")
60                        {
61                            modeloPeriodo.Lt = new List<liftTable>();
62                            foreach (XmlNode _cmNode in _psNode)
63                            {
64                                liftTable lt = new liftTable();
65                                lt.PERCENTILE = Convert.ToDouble(_cmNode.ChildNodes[0].InnerText.ToString(),
66                                    System.Globalization.CultureInfo.InvariantCulture);
67                                lt.VALUE = Convert.ToDouble(_cmNode.ChildNodes[1].InnerText.ToString(),
68                                    System.Globalization.CultureInfo.InvariantCulture);
69                                lt.PROBABILITY = Convert.ToDouble(_cmNode.ChildNodes[2].InnerText.ToString(),
70                                    System.Globalization.CultureInfo.InvariantCulture);
71                                modeloPeriodo.Lt.Add(lt);
72                            }
73                        }
74                    }
75                    modelsList.Add(modeloPeriodo);
76                }
77                RGPeriodos.DataSource = modelsList.OrderByDescending(am => am.Anomes).ToList();
78                RGPeriodos.DataBind();
79                drawAccuracyGraph();
80            }
81        }
82    }
83 }
```

Fig. 51 – Código da função loadStats (Ficheiro Default.aspx.cs)

Após a instanciação do serviço web (linha 18), é transmitida a mensagem com os parâmetros de entrada ao método **wsSelectModelStat** (linha 20) e retornada a string com o resultado da execução do procedimento, posteriormente convertido em documento XML (linha 21).

Após a validação do retorno do procedimento quanto à inexistência de erros e à existência de informação de retorno (linhas 22 a 26), são localizados os nós do tipo **/OUTPARAM/MODEL** no XML (linha 30). Cada um dos nós deste tipo contém toda a informação do modelo, nomeadamente os dados que compõem a Matriz de Confusão e os registos necessários à apresentação do gráfico Lift (zona 4 da [Fig. 50](#)).

Dado que o método pode retornar vários nós deste tipo, foi implementado um ciclo **foreach** que itera por cada um dos nós, e em cada um dos nós organiza a informação recolhida num objeto instanciado a partir da classe **modelo** (linha 33) apresentada na [Fig. 52](#).

O objeto da classe **modelo** acolhe as informações correspondentes ao nó **<MODEL>** do documento XML retornado (ver [Tab. 21](#)). Note-se que no nó **<MODEL>** do documento XML pode haver um atributo **<PERIOD>** que especifica o período a que se refere o modelo (linhas 36 a 39), um nó correspondente à matriz de confusão (**<CONFUSION_MATRIX>**), que contém quatro nós **<CM_ENTRY>** com conjuntos de atributos **<TYPE>** e **<VALUE>**, e finalmente um nó **<LIFT_TABLE>** com nós **<LT_ENTRY>** que contém os atributos **<PERCENTILE>**, **<VALUE>** e **<PROBABILITY>** necessários à construção do gráfico Lift.

Os nós da matriz da confusão e da tabela Lift são criados através da iteração dos elementos XML desse tipo, respetivamente linhas 40 a 58, e linhas 59 a 72.

Por sua vez, cada elemento do tipo modelo (**modeloPeriodo**) é adicionado a uma lista declarada em Viewstate (linha 75), como apresentado na [Fig. 52](#).

Por fim, a lista de modelos (**modelsList**), ordenada descendentemente pela indicação do período (atributo **Anomes**) é associada ao objeto de grelha (**RGPeriodos**) colocado na zona 1 da [Fig. 50](#) (linhas 77 e 78), e desenhado o gráfico da zona 2 da [Fig. 50](#) (linha 79).

```
1 private List<modelo> modelsList
2 {
3     get { return ViewState["modelsList"] == null ? null : (List<modelo>)ViewState["modelsList"]; }
4     set { ViewState["modelsList"] = value; }
5 }
6
7 [Serializable]
8 private class modelo
9 {
10     private string anomes;
11     public string Anomes
12     {
13         get { return anomes; }
14         set { anomes = value; }
15     }
16
17     public double Accuracy
18     {
19         get
20         {
21             return Math.Round((cm.TP + cm.TN) / Convert.ToDouble(cm.TP + cm.TN + cm.FP + cm.FN),4);
22         }
23     }
24
25     public double ErrorRate
26     {
27         get
28         {
29             return Math.Round((cm.FP + cm.FN) / Convert.ToDouble(cm.TP + cm.TN + cm.FP + cm.FN),4);
30         }
31     }
32
33     public double TruePositiveRate
34     {
35         get
36         {
37             return Math.Round(cm.TP / Convert.ToDouble(cm.TP + cm.FN),4);
38         }
39     }
40
41     public double FalsePositiveRate
42     {
43         get
44         {
45             return Math.Round(cm.FP / Convert.ToDouble(cm.FP + cm.FN),4);
46         }
47     }
48
49     public double PrecisionRate
50     {
51         get
52         {
53             return Math.Round(cm.TP / Convert.ToDouble(cm.TP + cm.FN),4);
54         }
55     }
56
57     public double Kappa
58     {
59         get
60         {
61             double ke = ((cm.TN + cm.FN) * (cm.TN + cm.FP) + (cm.FP + cm.TP) * (cm.FN + cm.TP)) /
62                 Convert.ToDouble(Math.Pow((cm.TP + cm.FP + cm.FN + cm.TN),2));
63             double k0 = (cm.TN + cm.TP) / Convert.ToDouble(cm.FN + cm.FP + cm.TN + cm.TP);
64             return Math.Round((k0-ke) / Convert.ToDouble(1-ke),4);
65         }
66     }
67
68     private confusionMatrix cm;
69     public confusionMatrix Cm
70     {
71         get { return cm; }
72         set { cm = value; }
73     }
74
75     private List<liftTable> lt;
76     public List<liftTable> Lt
77     {
78         get { return lt; }
79         set { lt = value; }
80     }
81 }
```

Fig. 52 – Classe *modelo* e declaração da lista *modelsList*

Refere-se ainda na classe **modelo**, apresentada na [Fig. 52](#), a presença de alguns atributos relevantes, que apenas apresentam o método **get**, dado que representam apenas cálculos efetuados sobre outros atributos. Esses atributos representam alguns dados estatísticos, tais como a taxa de Precisão (**Accuracy** – linhas 17 a 23), a taxa de Erro (**ErrorRate** – linhas 25 a 31), a taxa de Verdadeiros Positivos (**TruePositiveRate** – linhas 33 a 39), a taxa de Falsos Positivos (**FalsePositiveRate** – linhas 41 a 47), a taxa de Precisão (**PrecisionRate** – linhas 49 a 55) e o **Kappa** (linhas 57 a 66).

O desenho do gráfico de Evolução da Taxa de Precisão é efetuado através da função **drawAccuracyGraph** apresentado na [Fig. 53](#).

```
1 private void drawAccuracyGraph()
2 {
3     RCAccuracyEv.PlotArea.XAxis.Items.Clear();
4     RCAccuracyEv.PlotArea.Series.Clear();
5     RCAccuracyEv.PlotArea.YAxis.MaxValue =
6         Convert.ToDecimal(modelsList.Max(am => am.Accuracy).ToString("#.#")) + Convert.ToDecimal(0.1);
7     RCAccuracyEv.PlotArea.YAxis.MinValue =
8         Convert.ToDecimal(modelsList.Min(am => am.Accuracy).ToString("#.#")) - Convert.ToDecimal(0.1);
9
10    LineSeries sAccuracy = new LineSeries();
11
12    sAccuracy.LabelsAppearance.Visible = false;
13    sAccuracy.MarkersAppearance.Visible = false;
14
15    foreach (modelo model in modelsList.OrderBy(am => am.Anomes).ToList())
16    {
17        sAccuracy.Items.Add(Convert.ToDecimal(model.Accuracy));
18        RCAccuracyEv.PlotArea.XAxis.Items.Add(model.Anomes);
19    }
20    RCAccuracyEv.PlotArea.Series.Add(sAccuracy);
21 }
```

Fig. 53 – Função **drawAccuracyGraph**

Tendo a informação necessária associada a grelha da zona 1 da [Fig. 50](#) (RGPeriodos), o evento de selecção permite-nos obter o modelo correspondente ao período selecionado, e preencher os restantes objetos de forma simples, como indicado na [Fig. 54](#).

As linhas 7 a 12 da [Fig. 54](#) preenchem os objetos do tipo “Gauge” que se encontram na zona 5 da [Fig. 50](#).

As linhas 14 a 22 da [Fig. 54](#) preenchem os campos da Matriz de Confusão – zona 3 da [Fig. 50](#).

As linhas 24 e 25 preenchem os gráficos das zonas 2 e 4 da [Fig. 50](#).

```
1 protected void RGPeriodos_ItemCommand(object sender, Telerik.Web.UI.GridCommandEventArgs e)
2 {
3     if (e.Item is GridDataItem)
4     {
5         string anomes = (e.Item as GridDataItem).GetDataKeyValue("Anomes").ToString();
6         modeloSelecionado = modelsList.Where(itera => itera.Anomes == anomes).FirstOrDefault();
7         RGAccuracy.Pointer.Value = Convert.ToDecimal(modeloSelecionado.Accuracy * 100);
8         RGErorRate.Pointer.Value = Convert.ToDecimal(modeloSelecionado.ErrorRate * 100);
9         RGTruePositive.Pointer.Value = Convert.ToDecimal(modeloSelecionado.TruePositiveRate * 100);
10        RGFalsePositive.Pointer.Value = Convert.ToDecimal(modeloSelecionado.FalsePositiveRate * 100);
11        RGPrecision.Pointer.Value = Convert.ToDecimal(modeloSelecionado.PrecisionRate * 100);
12        RGKappa.Pointer.Value = Convert.ToDecimal(modeloSelecionado.Kappa * 100);
13
14        NTtp.Text = Convert.ToString(modeloSelecionado.Cm.TP);
15        NTfp.Text = Convert.ToString(modeloSelecionado.Cm.FP);
16        NTfn.Text = Convert.ToString(modeloSelecionado.Cm.FN);
17        NTtn.Text = Convert.ToString(modeloSelecionado.Cm.TN);
18        NTtotalTrue.Text = Convert.ToString(modeloSelecionado.Cm.TP + modeloSelecionado.Cm.FP);
19        NTtotalFalse.Text = Convert.ToString(modeloSelecionado.Cm.FN + modeloSelecionado.Cm.TN);
20        NTtotalCTrue.Text = Convert.ToString(modeloSelecionado.Cm.TP + modeloSelecionado.Cm.FN);
21        NTtotalCFalse.Text = Convert.ToString(modeloSelecionado.Cm.FP + modeloSelecionado.Cm.TN);
22        NTtotal.Text = Convert.ToString(modeloSelecionado.Cm.TP + modeloSelecionado.Cm.FP +
23            modeloSelecionado.Cm.FN + modeloSelecionado.Cm.TN);
24        drawLiftTableGraph(modeloSelecionado);
25        drawAccuracyGraph();
26    }
27 }
```

Fig. 54 – Função do evento **RGPeriodos_ItemCommand**

Relativamente ao desenvolvimento da camada de apresentação, tal como já tinha sido referido para o serviço web, foi utilizado o ambiente de desenvolvimento Visual Studio 2010, a linguagem C# com Framework 4.0.

A [Fig. 55](#) apresenta um panorama geral da solução.

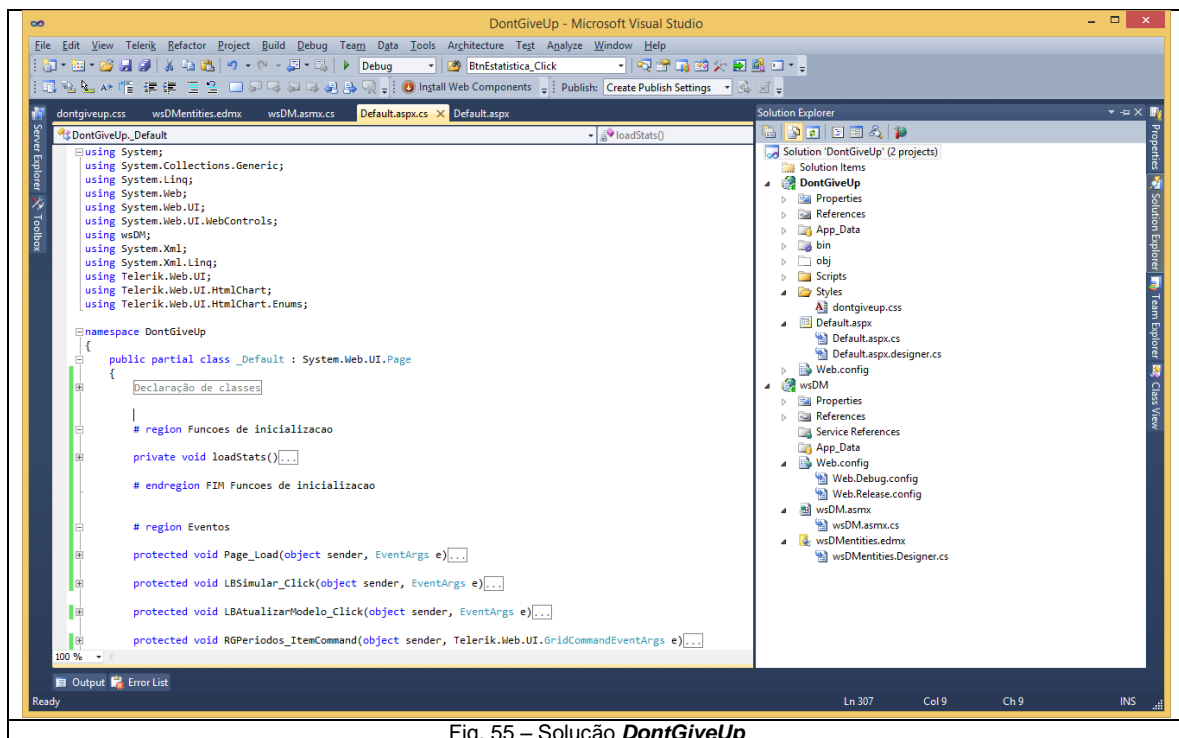
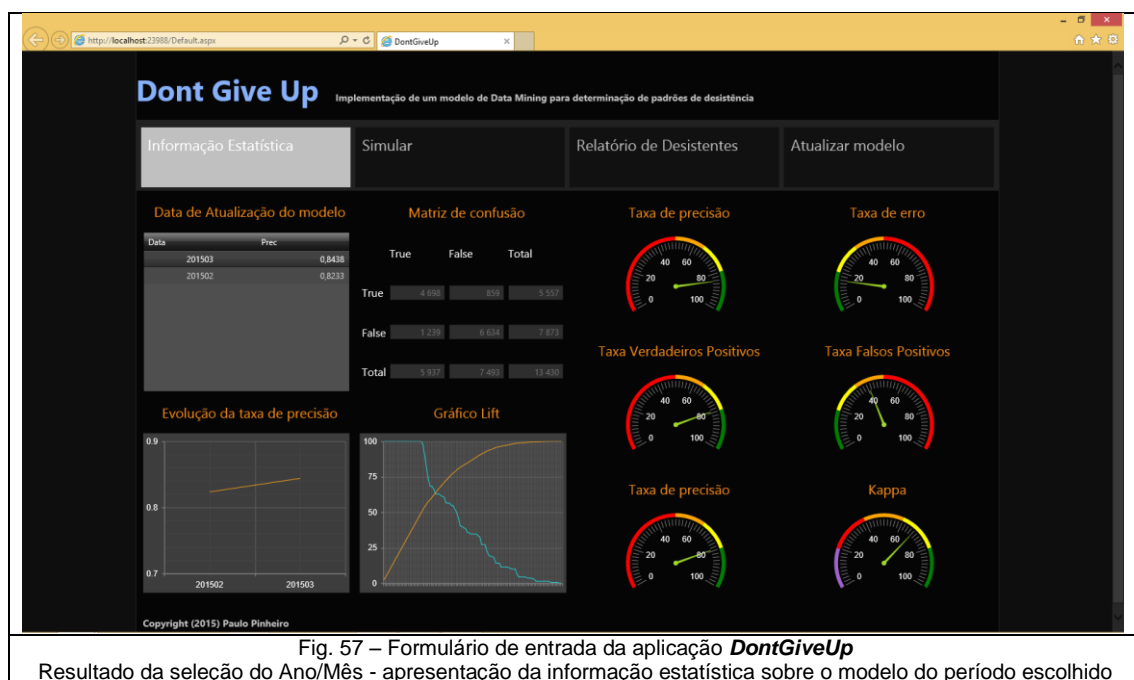
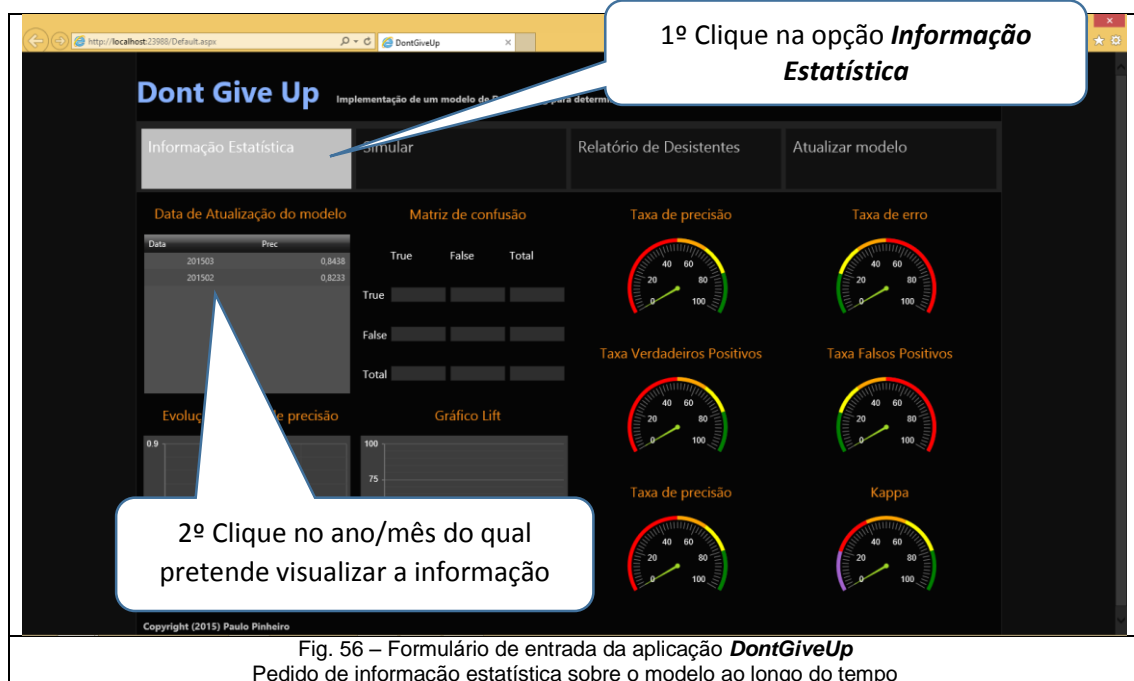


Fig. 55 – Solução **DontGiveUp**

5.2 Manual da aplicação

A aplicação é utilizável através de um navegador da Internet, e se instalada no computador local, acessível em <http://localhost>. Ao entrar na aplicação, é apresentado o formulário da [Fig. 56](#), que apresenta quatro opções na parte superior do formulário. Para utilizar a aplicação, siga as instruções nas figuras seguintes.

Informação Estatística



Simular

Esta opção permite-lhe simular os dados de um utente para obter a probabilidade de alguém com essas características se tornar um Desistente.

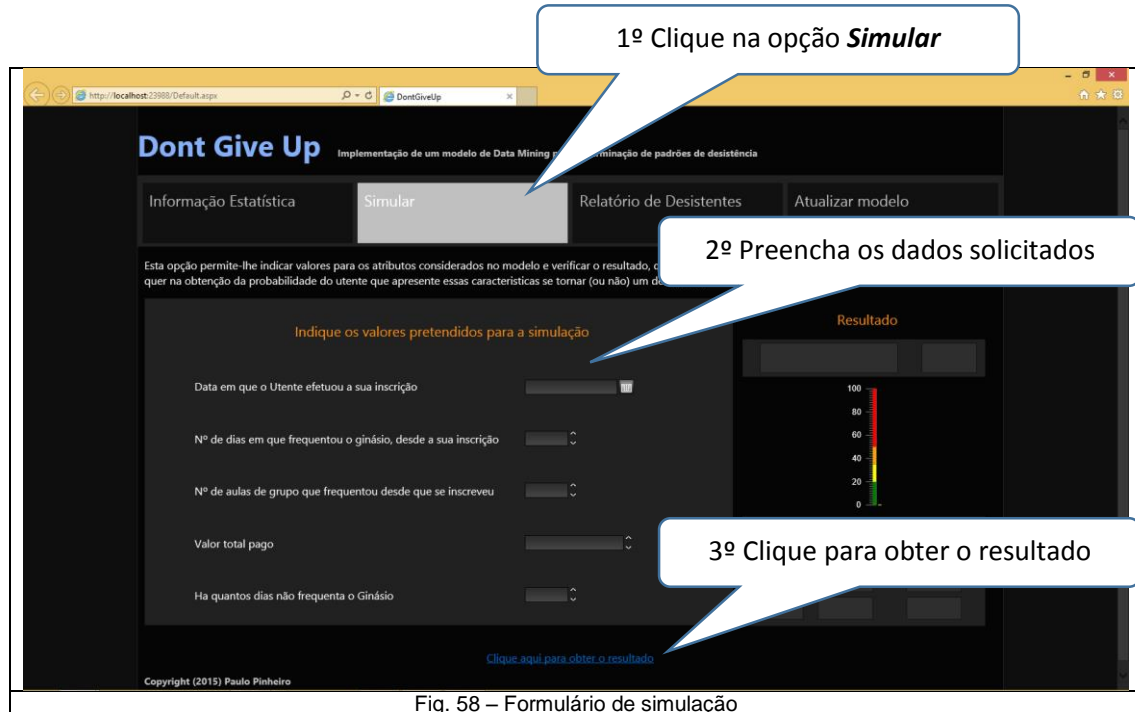


Fig. 58 – Formulário de simulação

Os resultados são apresentados do lado direito do formulário.

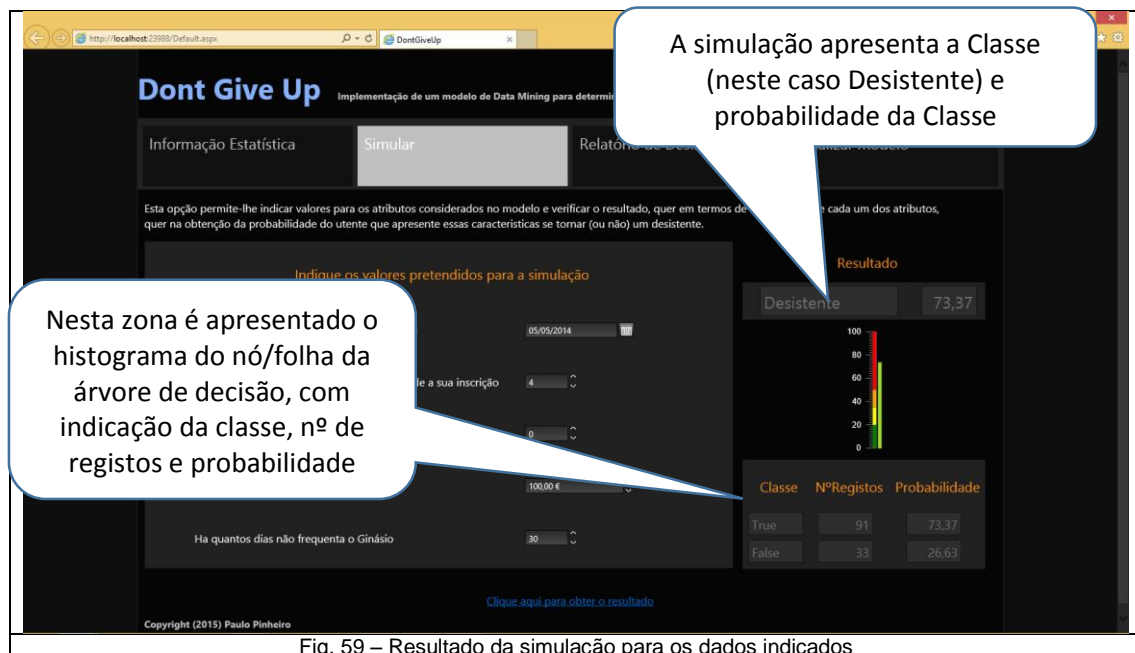


Fig. 59 – Resultado da simulação para os dados indicados

Relatório de Desistentes

Esta opção permite-lhe escolher um Clube e indicar uma probabilidade. A aplicação irá apresentar-lhe todos os utentes que, de acordo com as suas características atuais, apresentem uma probabilidade de desistência maior ou igual à que indicou.

1º Escolher a opção **Relatório de Desistentes**

2º Escolha o Clube e indique a probabilidade que pretende considerar

3º Clique para obter o resultado

Fig. 60 – Formulário de solicitação do Relatório de Desistentes

Nesta zona é apresentada a lista de utentes, o valor dos atributos e a classificação correspondente, bem como a probabilidade de se tornar desistente

Se a lista for demasiado grande, clique aqui para navegar nas páginas

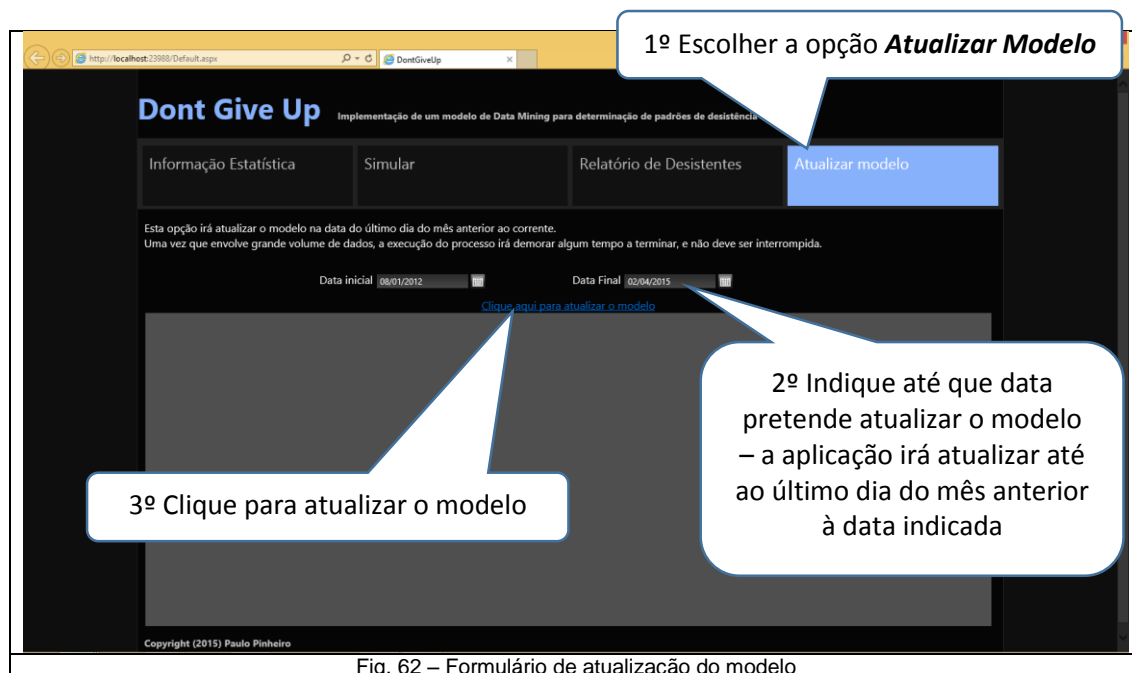
Nome do Utente	NºCartão	Dur. Inscrição	Classe	Dias sem Vistas	Classe	Freq. Média	Classe	Nº. Aulas Grupo	Classe	Vol. Negócios	Classe	Probabilidade
ANTONIO TAVARES FORTES	10048725	6	3	101	4	0.46	3	0	4	55.7	1	57.72
BLAS ALEXANDER SOUSA FUENTES	10045409	20	5	116	4	0.08	1	0	1	344.2	3	99.84
BRUNO MIGUEL FRANCISCO PEREIRA MARTINHO	10048804	6	2	99	4	0.58	3	0	2	85.6	1	99.80
CATARINA CASTANHEIRA NUNES ANTÓNIO	10048511	6	3	106	4	0.13	1	0	4	149.7	2	57.72
CÉLIA CATARINA OLIVA SARAIVA	10047417	12	3	196	5	0.81	3	24	5	279.5	3	51.03
CRISTINA MANUELA NUNES CANDEIAS DOS SANTOS	10048214	9	3									
DALIA CARINA VARELA	10047111	15	4									
DIOGO MIGUEL ALVES BÉLICO DE VELASCO	10047744	10	3									
DULCE BRITO ANDRADE	10047628	10	3									
EDUARDO JOSE LOPES PAO DURO FERNANDES	10048513											
EMÍLIA DO PATROCÍNIO TAVARES RAMOS	10047105	15	4	106	4	0.6	3	0	4	358.9	3	60.97
FILÍPE DE BRITO VIEIRA												

Fig. 61 – Apresentação do Relatório de Utentes com probabilidade de Desistência maior ou igual à indicada

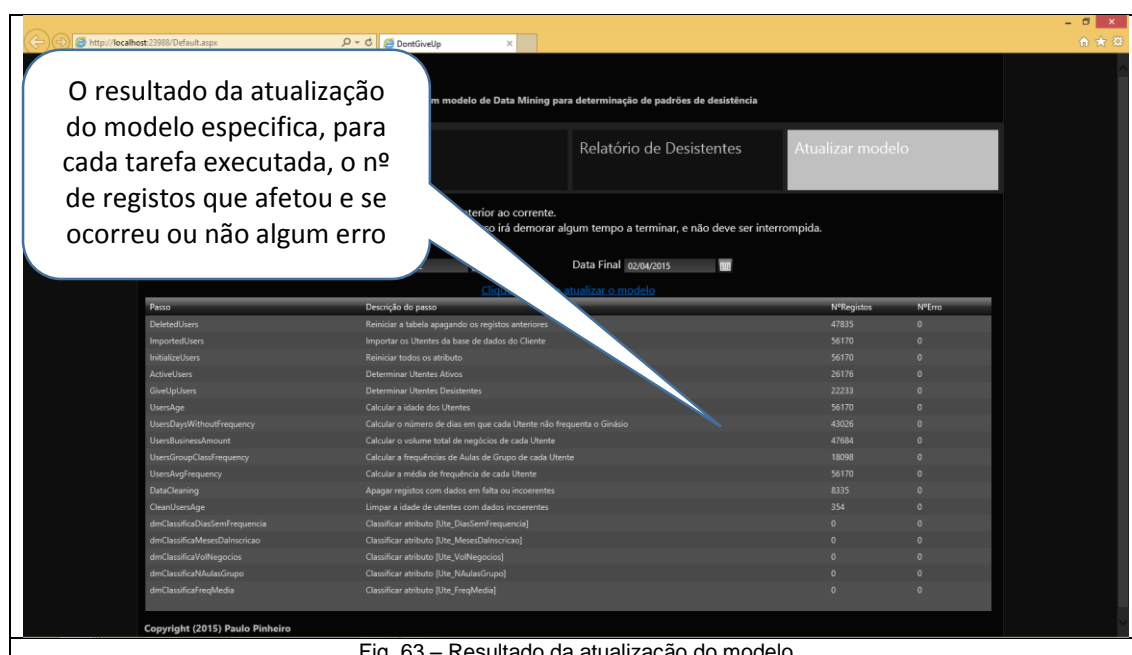
Note que, dado o volume de dados na base de dados, o relatório pode demorar vários minutos a ser apresentado.

Atualizar Modelo

Todos os dias há novos utentes admitidos e outros que desistem, há novas organizações do clube, há outros clubes que abrem, etc. Em resumo, há uma série de fatores que podem alterar o perfil dos Desistentes. Esta opção permite-lhe atualizar o modelo para que se adapte às novas situações, pelo que deve utilizá-la uma vez por mês.



Note que o processo de atualização é moroso e sobrecarrega o servidor, pelo que não deve executá-lo em horas de pico de utilização do sistema informático.



6 Conclusões

Sobre o projeto

O mercado do Fitness regista uma taxa de abandono na ordem dos 40 a 45%, taxa que é considerada muito elevada, tendo em consideração outros sectores de mercado.

Sendo conhecido que cerca de 1/3 das perdas de clientes ocorrem por razões não controláveis (McCarthy, 2004) – mudança de residência, mudança de local de trabalho, divórcio, etc. -, sobram 2/3 sobre os quais os clubes podem atuar na tentativa de reter, pelo menos durante mais algum tempo, esses clientes.

Tendo em conta estes dados, é fácil perceber que um clube com 4000 utentes, por exemplo, perderá cerca de 1800 utentes por ano, se considerarmos o valor máximo da taxa de abandono (45%) indicado acima. Se considerarmos um valor médio de 35,00€ por mensalidade, está-se perante uma perda de receita, no mínimo, da ordem dos 63.000,00€ por ano.

Assim, se o clube conseguir melhorar a taxa de retenção apenas em 1%, conseguirá manter mais 40 utentes ($4000 * 1\% = 40$), conseguindo assim um aumento da receita na ordem dos 16.800,00€ ($40 * 35,00€ * 12\text{meses}$).

É neste contexto que o projeto desenvolvido colabora, identificando os potenciais utentes desistentes sobre os quais o clube deve atuar. No entanto, haverá que considerar duas situações:

- a) Nem todos os potenciais desistentes apresentam a mesma probabilidade de virem a desistir;
- b) Nem sempre é possível intervir junto de todos os utentes que apresentam probabilidade de virem a desistir;

Por estes motivos, a aplicação desenvolvida apresenta as estatísticas do modelo calculadas para cada período que o utilizador deve ter em atenção ao utilizar as funcionalidades disponibilizadas.

Para levar a cabo o trabalho de prevenção das desistências, cujo valor se referiu nos parágrafos anteriores, o desenvolvimento efetuado neste trabalho permite a disponibilização de duas funcionalidades no Software da empresa, através da utilização dos serviços web:

- a) Apresentar ao utilizador - leia-se qualquer funcionário do clube com acesso ao utilizador - sempre que este acede à ficha de um Utente, a probabilidade que este apresenta no momento, de vir a tornar-se um Desistente.

Uma vez que os utilizadores poderão não estar sensibilizados para o valor da probabilidade que leva à desistência, a probabilidade pode ser apresentada numa escala de cores que facilite a compreensão da informação;

Para o fazer bastará consumir o serviço web [wsSelectUserProbability](#);

- b) Distribuir periodicamente pela equipa de vendas do clube, a lista dos clientes que apresentam uma probabilidade de desistência superior a x, para que de forma organizada e planeada se tomem as medidas necessárias para evitar que a desistência se venha a concretizar;

A relação entre o número de utentes que se podem abordar e a respetiva probabilidade de se virem a tornar desistentes pode ser obtida no gráfico Lift apresentado na opção Informação Estatística da aplicação desenvolvida.

Obter os utentes nesta situação pode ser conseguido através do consumo do serviço web [wsSelectGiveUpUsers](#) ou diretamente na aplicação desenvolvida, através da opção Relatório de Desistentes.

Por outro lado, o interface desenvolvido facilita a gestão do modelo ao longo do tempo, e possibilita a análise dos valores estatísticos do modelo ao longo do tempo e a sua respetiva evolução.

Subdomínios a explorar

Apesar do trabalho desenvolvido se apresentar já como uma mais-valia para o Software da empresa e principalmente para os seus utilizadores, é possível aprofundar mais o mesmo, quer pela afinação do modelo em termos de precisão, quer pela obtenção de informação adicional segmentada por Clube, por zona do país, por atividades e eventos em que o utente participa no clube, etc.. Tal afinação dotaria os gestores de clube com informação ainda mais rica que lhes permitiria tomar iniciativas para alterar a vontade de desistir por parte dos utentes.

Por outro lado, para que seja possível aprofundar e enriquecer o modelo criado, é necessário obter informação que não se encontra disponível. Recorda-se o abandono da análise de alguns atributos por falta de dados (registo de contatos com os clientes, de reclamações, de outros utentes relacionados – familiares e outras referências - e distância a percorrer para chegar ao clube – [Tab. 1](#)). Torna-se assim necessário, por um lado, sensibilizar os gestores de Clube da importância do registo adequado da informação, e por outro lado, verificar se o incremento de custo relacionado com o registo de mais informação se justifica, tendo em atenção o possível retorno.

Sugere-se também a criação de mecanismos adicionais de obtenção da informação, para além dos já existentes e disponibilizados ao utilizador e ao próprio cliente (utilização do controlo de acessos e do Checkin de Atividades), como portais e aplicações móveis, que permitam captar outras preferências dos clientes.

Sobre a implementação do projeto

O desenvolvimento deste projeto envolveu conhecimentos adquiridos em várias Unidades Curriculares do curso, nomeadamente e por ordem alfabética: Administração de Sistemas Informáticos, Análise de Sistemas, Desenvolvimento de Software, Fundamentos de Base de Dados, Gestão de Projetos Informáticos, Linguagens de Programação, Probabilidade e Estatística, Programação, Programação por Objetos, Sistemas de Gestão de Bases de Dados, Sistemas Distribuídos e Sistemas e Serviços Web.

Tendo em atenção o resultado obtido, o autor acredita ter aplicado adequadamente os conhecimentos e metodologias aprendidas ao longo do curso, e ter cumprido os objetivos a que se propôs inicialmente.

Lista de termos

Ativo: estado que um utente assume durante o período que está inscrito numa atividade de Fitness e tem a sua situação regularizada.

Atividade: modalidade que o utente pratica no Ginásio.

Checkin: ato de marcar a presença numa aula de grupo num equipamento self-service.

Controlo de Acessos: ato de entrar no Ginásio para realizar qualquer atividade, utilizando o seu cartão de utente no leitor destinado ao efeito.

Clube (ou Ginásio): entidade ou local que disponibiliza instalações e serviços desportivos.

Desistente: estado que um utente assume quando termina a sua inscrição numa atividade de Fitness e deixa de frequentar o Clube.

Gráfico ROC: gráfico que relaciona a Sensibilidade (taxa de verdadeiros positivos) com a Especificidade (taxa de verdadeiros negativos ou 1-taxa de falsos positivos) de um método. Neste gráfico, uma linha que esteja mais próxima do topo esquerdo do gráfico, tendo uma rápida subida até à linha horizontal superior e apresentando a maior área abaixo da linha sugere um modelo melhor.

Pré-inscrição: estado que um individuo assume ao ser registado na base de dados, mas não tendo pago ou iniciado nenhuma atividade no Clube.

Receita Principal: serviço ou artigo parametrizado como fazendo parte do core-business do Clube. É pela inscrição num serviço assinalado com esta característica que um Utente passa ao estado Ativo após se inscrever e iniciar o pagamento do serviço.

Standby: estado que um utente assume quando, estando inscrito num serviço parametrizado como Receita Principal, não tem a sua situação regularizada.

Utente: o termo é ambíguo e depende do contexto em que é utilizado. Pode referir-se à pessoa que utiliza e é cliente do Clube ou ao estado que a pessoa assume quando deixa de frequentar uma atividade de Receita Principal (Fitness no caso) mas mantém-se a usufruir, ou usufruiu de outros serviços prestados pelo Clube.

Gráfico Lift (Elevador): gráfico que relaciona o valor real do atributo previsível da população geral, com a previsão obtida para os dados de teste.

No caso do gráfico produzido pelo MS SQL AS, é apresentada uma linha correspondente ao resultado produzido por um modelo ideal, e outra linha representando o resultado de uma previsão aleatória. Pretende-se que a linha do modelo construído se apresente o mais próximo da linha do modelo ideal.

Este gráfico só se aplica a atributo discretos.

Referências

BERSON A., SMITH S. e THEARLING K. (1999) “Building Data Mining Applications for CRM”, McGraw Hill

CAVIQUE L., MENDES A.B., FUNK M., SANTOS J.M.A. (2013), “A Feature Selection Approach in the Study of Azorean Proverbs”, in Exploring Innovative and Successful Applications of Soft Computing, Advances in Computational Intelligence and Robotics (ACIR) Book Series

CAVIQUE L., Micro-Segmentação de Clientes com Base em Dados de Consumo: Modelo RM-Similis,

<http://www.univ-ab.pt/~lcavique/publicacoes/Similis%20RM.pdf> [27 de Março de 2015]

CHAPMAN P., CLINTON J., KERBER R., KHABAZA T., REINARTZ T., SHEARER C. e WIRTH R. (2000) CRISP-DM 1.0 Step-by-step data mining guide, SPSS

CORREIA A., SACAVÉM A., COLAÇO C. (2006) “A indústria do wellness” in Manual de Fitness & Marketing, Visão e Contextos

DHURUP M., SURUJLAL J., (2012) “Establishing and Maintaining Customer Relationships in Commercial Health and Fitness Centers in South Africa”, International Journal of Trade, Economics and Finance, Vol. 3, No. 1, February 2012

HAN J., KAMBER M., (2006) “Data Mining Concepts and Techniques”, Elsevier

HOTHORN T., HORNIK K., STROBL C., ZEILEIS A. (2015), <http://party.R-forge.R-project.org> [8 de Junho de 2015]

HUGHES A. M., (2015) “Why RFM works in Predicting Response”, <http://www.dbmarketing.com/articles/Art245.htm> [6 de Maio de 2015]

IBM SPSS Modeler CRISP-DM Guide (2011), IBM Corporation

IHRSA Member Retention Report, Vol 1 Issue 1 (2012) “Measuring Retention”, <http://www.ihrsa.org/member-only-updates/category/retention?SSLoginOk=true> [30 de Março de 2015]

IHRSA Member Retention Report, Vol 1 Issue 4, (2013) “Does the effect of individual Net Promoter Scores (NPS®) predict risk of cancellation?”, <http://www.ihrsa.org/member-only-updates/category/retention?SSLoginOk=true>

[30 de Março de 2015]

IMHOFF C., LOFTIS L., GEIGER J. G. (2001) “Building the Customer-Centric Enterprise”, Wiley

LANDIS R., KOCH G. (1977) “The Measurement of Observer Agreement for Categorical Data”, International Biometric Society

LAROSE D. (2006) “Data Mining Methods and Models”, Wiley-Interscience

LEDOLTER J. (2013) “Data Mining and Business Analytics with R”, Wiley

MICROSOFT, “Data Mining (SSAS)”, <https://msdn.microsoft.com/en-us/library/bb510516.aspx> [30 de Abril de 2015]

MURTEIRA B., RIBEIRO C. S., SILVA J. A., PIMENTA C. (2008) “Introdução à Estatística”, McGraw Hill

TAN P., STEINBACH M. e KUMAR V. (2006) “Introduction to Data Mining”, Addison-Wesley

RIPLEY B. (2015) “Package tree”, <http://cran.r-project.org/web/packages/tree/tree.pdf> [6 de Junho de 2015]

SILBERSCHATZ A., KORTH H. e SUDARSHAN S. (2006) “Database System Concepts”, McGraw Hill Higher Education

SING T., SANDER O., BEERENWINKEL N., LENGAUER T. (2015) “Package ROCR”, <http://rocr.bioinf.mpi-sb.mpg.de/> [22 de Maio de 2015]

<http://www.w3schools.com/css/> [02 de Julho de 2015]