

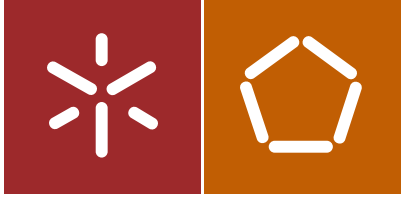
Universidade do Minho
Escola de Engenharia

Nuno Alberto Ribeiro da Silva

Poker Learner: Modelação de Jogadores através de Data Mining

Poker Learner: Modelação de Jogadores através de Data Mining
Nuno Alberto Ribeiro da Silva
UMinho | 2015

outubro de 2015



Universidade do Minho
Escola de Engenharia

Nuno Alberto Ribeiro da Silva

Poker Learner: Modelação de Jogadores através de Data Mining

Dissertação de Mestrado
Ciclo de Estudos Integrados Conducentes ao Grau de
Mestre em Engenharia e Gestão de Sistemas de Informação

Trabalho efectuado sob a orientação do
Professor Doutor Luis Paulo Reis

outubro de 2015

Declaração

Nome: Nuno Alberto Ribeiro da Silva

Endereço eletrónico: a62851@alunos.uminho.pt

Telefone: 967161737

Bilhete de Identidade: 12924208

Título da dissertação: *Poker Learner:* Modelação de jogadores através de *Data Mining*

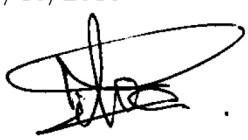
Orientador: Professor Doutor Luís Paulo Reis

Ano de conclusão: 2015

Designação do Mestrado: Mestrado Integrado em Engenharia e Gestão de Sistemas de Informação

É AUTORIZADA A REPRODUÇÃO INTEGRAL DESTA TESE/TRABALHO APENAS PARA EFEITOS DE INVESTIGAÇÃO, MEDIANTE DECLARAÇÃO ESCRITA DO INTERESSADO, QUE A TAL SE COMPROMETE.

Universidade do Minho, 30/10/2015

Assinatura: 

“Aim for the sky and you'll reach the ceiling. Aim for the ceiling and you'll stay on the floor.”

(Bill Shankly)

Agradecimentos

O desenvolvimento de uma dissertação de mestrado é o culminar de cinco anos de aprendizagem e de uma etapa na vida. Ao longo destes cinco anos a nossa vida é tocada por um variado tipo de pessoas, pessoas essas que deixam um pedaço de si e levam um pedaço de nós.

Neste percurso académico que agora termina, tive a oportunidade de conhecer excelentes profissionais aos quais agradeço pelo conhecimento transmitido e, por muitas vezes, ouvirem e compreenderem os problemas dos alunos. São estes profissionais que fazem da Universidade do Minho e do Departamento de Sistemas de Informação referências a nível nacional e que permitem que os seus alunos cresçam e adquiram o conhecimento necessário para enfrentar a nova etapa que se surge à sua frente.

Tenho que agradecer ao meu orientador Luis Paulo Reis, por toda a ajuda e acompanhamento que me deu ao longo desta longa caminhada que foi a realização desta minha última etapa no ensino superior.

Agradeço também a todas as pessoas que me souberam acolher no meu primeiro ano, que conseguiram transmitir as suas ideias e me ajudaram quando necessitei.

Não posso deixar de agradecer a uma face diferente da Universidade do Minho, que são os seus grupos culturais, nomeadamente à iPUM, que me acolheu no seu seio e me acompanhou nos bons e maus momentos.

Gostaria de agradecer às pessoas que me acompanharam ao longo de quase todo o percurso académico e que muito me ajudaram a ultrapassa-lo com sucesso, que me apoiaram nos piores momentos, que festejaram comigo nos bons, que riram e sofreram com o meu humor. A essas pessoas, a esse grupo de amigos, o meu muito obrigado!

Além dessas pessoas quero agradecer a uma pessoa muito especial para mim, que deu uma nova cor à minha vida e que muita felicidade e alegria me vem trazendo ao longo destes anos. Muito obrigado pela pessoa que és Sofia.

Por fim quero agradecer à minha família, em especial aos meus pais e à minha irmã que me apoiaram nesta longa caminhada e me ajudaram a tomar uma das decisões mais complicadas da minha da vida.

Resumo

O poker tem vindo a obter cada vez mais atenção por parte da população em geral devido, essencialmente, à existência de grandes torneios que são transmitidos para todo o mundo via canais televisivos. O interesse pelo poker tem vindo a alargar-se, nos últimos anos, também à comunidade científica, mais propriamente aos investigadores que se encontram ligados à área da inteligência artificial. Isto acontece devido aos enormes desafios que o poker proporciona à investigação nesta área. Ao contrário de muitos outros jogos, o poker é um jogo estocástico e de informação imperfeita, o que cria uma elevada quantidade de possibilidades para cada estado do jogo.

Muitos têm sido os esforços para se criar um agente virtual capaz de se bater com os melhores jogadores humanos de poker. No entanto, até ao momento, tal tem-se revelado infrutífero uma vez que, apesar dos resultados demonstrarem uma evolução positiva ao longo do tempo sobretudo em variantes simplificadas do Poker, a criação de um agente que supere os melhores jogadores humanos na vertente *Texas Hold'em, No Limit, Multi-Player Poker* ainda não foi conseguida.

Nesta dissertação é seguida uma linha de pensamento diferente da grande maioria dos trabalhos de investigação anteriores sobre este jogo. Em vez de ser tentada a criação de um agente virtual imbatível, será tentada a criação de um agente capaz de reproduzir a forma de jogar de um jogador humano. Para tal, é necessária a construção de um modelo de dados de alto nível, capaz de abarcar o máximo de informação relevante para cada estado de jogo. Esse modelo é depois carregado com dados provenientes de uma base de dados que contém milhares de jogadas de um jogador humano real. Ao modelo de dados são aplicadas técnicas de *data mining* de forma a extrair um modelo de jogador. São extraídos vários modelos de jogador para cada fase do jogo, sendo eles comparados entre si (dentro de cada fase de jogo) através da utilização de vários dados estatísticos.

Como resultado desta dissertação foi possível verificar a hipótese da criação de um modelo de jogador capaz de copiar as ações de um jogador profissional de poker. Para tal, foram criados quatro modelos de dados, um para cada fase do jogo, sendo depois populados com os dados de um jogador profissional, através de um processo de extração e carregamento de dados. Os resultados atingidos foram superiores nas duas primeiras fases do poker em comparação com as duas últimas onde os resultados foram de menor relevância. Esta discrepância nos resultados entre as primeiras e as últimas

fases do poker pode-se dever à acentuada diminuição de dados disponíveis para utilização nas últimas fases.

Palavras-Chave: *poker, inteligência artificial, agentes inteligentes, data mining.*

Abstract

Poker has been gradually gaining the attention of people in general, in great part due to the worldwide broadcast of big tournaments by TV channels. In recent years, the interest in poker has also been extended to the scientific community especially to artificial intelligence researchers, since poker provides great challenges to the research in the area. Unlike many other games, poker is a stochastic game of imperfect information, which creates a high amount of possibilities to every state of the game.

Many have been the efforts to create a virtual agent capable of standing against the best human poker players. Yet, so far, they have proved to be unfruitful. Despite the positive evolution of the results over time, mainly in simplified poker variants, the creation of an unbeatable agent in the variant Texas Hold'em, No limit, Multi-Player Poker has yet to be accomplished.

In this thesis a different line of thought is followed: instead of trying to create an unbeatable agent, as previous research works have tried; the aim is to create one capable of reproducing the way a human player plays. With that end, a high level data model able to comprehend the maximum of information relevant to every state of the game has to be built. That model is, then, loaded with data from a database containing millions of plays made by a human player. Data mining techniques are applied to the data model, in such a way as to extract a model of player. Several models are extracted for each stage of the game, and then compared among themselves (in every game stage) by applying several statistic data.

As a result of this thesis, it was possible to verify the hypothesis of creating a player model able to copy the moves of a professional poker player. In order to accomplish that, four data models were created (one for each stage of the game), which were then populated with data from a professional player, using a data extraction and loading process. The results achieved were superior in the first two stages of poker compared to the last two, in which the results were of less relevance. This discrepancy in the results can be explained by a pronounced decreasing of available data for the last two stages.

Keywords: poker, artificial intelligence, intelligent agent, data mining.

Índice

AGRADECIMENTOS IV
RESUMO V
ABSTRACT VII
1 INTRODUÇÃO 1
1.1 ENQUADRAMENTO 1
1.2 O POKER 2
1.2.1 Ranking de Mãos 3
1.2.2 Fases do Poker 6
1.3 PROBLEMA DE INVESTIGAÇÃO 7
1.4 ABORDAGEM METODOLÓGICA 8
1.4.1 Metodologia de Investigação 8
1.4.2 Metodologia de Desenvolvimento 10
1.4.3 Questões Éticas 11
1.5 OBJETIVOS 11
1.6 ESTRUTURA DO DOCUMENTO 11
2 ESTADO DA ARTE 13
2.1 PROCESSO DE IDENTIFICAÇÃO DA LITERATURA E ESTRUTURAÇÃO DA REVISÃO 13
2.2 POKER COMPUTORIZADO 14
2.2.1 Locais de Investigação 14
2.2.2 Abordagens para a Construção de Agentes de Poker 15
2.2.3 Agentes de Poker 16
2.2.4 Abstrações 17
2.2.5 PokerLang 18
2.2.6 Framework HoldemML 18
2.3 *DATA MINING* 19
2.3.1 Metodologias 20
2.3.2 Métodos de Data Mining 24
2.3.3 Ferramentas Data Mining 29
2.4 *DATA MINING* EM DIVERSAS ÁREAS 29
2.4.1 Problemas e Desafios 30

2.4.2 Data Mining para User Modeling 31
2.4.3 Data Mining no Poker 32
2.5 Conclusões 33
3 DESENVOLVIMENTO 34
3.1 Compreensão do Negócio 34
3.2 Compreensão de Dados 36
3.3 Casos de estudo 37
3.3.1 Experiência preliminar – Pre-flop 38
3.3.2 Caso de estudo 1 – Fold ou Call 38
3.3.3 Caso de estudo 2 – Call, Check, Raise, Fold ou All-In 38
3.3.4 Caso de estudo 3 e 4 – Diferentes Modelos de Dados 38
3.4 Preparação dos dados 39
3.4.1 Experiência Preliminar 39
3.4.2 Casos de estudo 1, 2, 3 e 4 41
3.5 Modelo de Dados 44
3.5.1 Experiência Preliminar 44
3.5.2 Caso de Estudo 1 e 2 45
3.5.3 Caso de estudo 3 - Seleção automática de atributos 50
3.5.4 Caso de estudo 4 - Seleção manual de atributos 52
3.6 Modelação 53
3.6.1 Modelação – Experiência preliminar 53
3.6.2 Modelação – Caso de estudo 1, 2, 3 e 4 53
3.7 Conclusões 54
4 EXPERIÊNCIAS E RESULTADOS 55
4.1 Experiência preliminar 55
4.2 Caso de estudo 1 - *Fold* ou *Call* 58
4.2.1 Pre-Flop 59
4.2.2 Flop 60
4.2.3 Turn 62
4.2.4 River 63
4.3 Caso de Estudo 2 – Modelo de Dados Completo 64
4.3.1 Pre-Flop 65

4.3.2 Flop 67
4.3.3 Turn 69
4.3.4 River 71
4.4 Caso de estudo 3 – Atributos Automaticamente Selecionados 73
4.4.1 Pre-Flop 73
4.4.2 Flop 75
4.4.3 Turn 77
4.4.4 River 79
4.5 Caso de estudo 4 – Atributos Manualmente Selecionados 80
4.5.1 Pre-Flop 81
4.5.2 Flop 82
4.5.3 Turn 84
4.5.4 River 86
4.6 Conclusões 88
4.6.1 Experiência preliminar 88
4.6.2 Caso de estudo 1 88
4.6.3 Caso de estudo 2 90
4.6.4 Caso de estudo 3 e 4 91
5 CONCLUSÕES E TRABALHO FUTURO 93
REFERÊNCIAS BIBLIOGRÁFICAS 95

Índice de Tabelas

TABELA 1 - DISTRIBUIÇÃO DA MÃO VENCEDORA 5

TABELA 2 - COMPARAÇÃO DE CARACTERÍSTICAS ENTRE SEMMA E CRISP-DM 21

TABELA 3 - COMPARAÇÃO DE FERRAMENTAS DE *DATA MINING* 29

TABELA 4 - TÉCNICAS DE *DATA MINING* UTILIZADAS PARA MODELAÇÃO DE UTILIZADOR 32

TABELA 5 - COMPARAÇÃO DE OUTROS TRABALHOS ENVOLVENDO POKER E *DATA MINING* 33

TABELA 6 – SELEÇÃO AUTOMÁTICA DE ATRIBUTOS PARA A FASE DE *PRE-FLOP* 50

TABELA 7 – SELEÇÃO AUTOMÁTICA DE ATRIBUTOS PARA A FASE DE *FLOP* 51

TABELA 8 – SELEÇÃO AUTOMÁTICA DE ATRIBUTOS PARA A FASE DE *TURN* 51

TABELA 9 – SELEÇÃO AUTOMÁTICA DE ATRIBUTOS PARA A FASE DE *RIVER* 52

TABELA 10 - RESULTADOS OBTIDOS ATRAVÉS DA EXECUÇÃO DOS ALGORITMOS 56

TABELA 11 - MATRIZ DE CONFUSÃO DOS VÁRIOS ALGORITMOS TESTADOS 57

TABELA 12 – RESULTADOS OBTIDOS NO CASO DE ESTUDO 1 *PRE-FLOP* 59

TABELA 13 – MATRIZ DE CONFUSÃO NO CASO DE ESTUDO 1 *PRE-FLOP* 59

TABELA 14 – RESULTADOS OBTIDOS NO CASO DE ESTUDO 1 *FLOP* 60

TABELA 15 – MATRIZ DE CONFUSÃO NO CASO DE ESTUDO 1 *FLOP* 61

TABELA 16 – RESULTADOS OBTIDOS NO CASO DE ESTUDO 1 *TURN* 62

TABELA 17 – MATRIZ DE CONFUSÃO NO CASO DE ESTUDO 1 *TURN* 62

TABELA 18 – RESULTADOS OBTIDOS NO CASO DE ESTUDO 1 *RIVER* 63

TABELA 19 – MATRIZ DE CONFUSÃO NO CASO DE ESTUDO 1 *RIVER* 64

TABELA 20 – RESULTADOS OBTIDOS NO CASO DE ESTUDO 2 *PRE-FLOP* 65

TABELA 21 – MATRIZ DE CONFUSÃO NO CASO DE ESTUDO 2 *PRE-FLOP* 66

TABELA 22 – RESULTADOS OBTIDOS NO CASO DE ESTUDO 2 *FLOP* 67

TABELA 23 – MATRIZ DE CONFUSÃO NO CASO DE ESTUDO 2 *FLOP* 68

TABELA 24 – RESULTADOS OBTIDOS NO CASO DE ESTUDO 2 *TURN* 69

TABELA 25 – MATRIZ DE CONFUSÃO NO CASO DE ESTUDO 2 *TURN* 70

TABELA 26 – RESULTADOS OBTIDOS NO CASO DE ESTUDO 2 *RIVER* 71

TABELA 27 – MATRIZ DE CONFUSÃO NO CASO DE ESTUDO 2 *RIVER* 72

TABELA 28 – RESULTADOS OBTIDOS NO CASO DE ESTUDO 3 - ATRIBUTOS AUTOMATICAMENTE SELECIONADOS *PRE-FLOP* 73

TABELA 29 – MATRIZ DE CONFUSÃO NO CASO DE ESTUDO 3 - ATRIBUTOS AUTOMATICAMENTE SELECIONADOS *PRE-FLOP*. 74

TABELA 30 – RESULTADOS OBTIDOS NO CASO DE ESTUDO 3 - ATRIBUTOS AUTOMATICAMENTE SELECIONADOS *FLOP* 75

TABELA 31 – MATRIZ DE CONFUSÃO NO CASO DE ESTUDO 3 - ATRIBUTOS AUTOMATICAMENTE SELECIONADOS *FLOP*.....76
TABELA 32 – RESULTADOS OBTIDOS NO CASO DE ESTUDO 3 - ATRIBUTOS AUTOMATICAMENTE SELECIONADOS *TURN*.....77
TABELA 33 – MATRIZ DE CONFUSÃO NO CASO DE ESTUDO 3 - ATRIBUTOS AUTOMATICAMENTE SELECIONADOS *TURN*.....78
TABELA 34 – RESULTADOS OBTIDOS NO CASO DE ESTUDO 3 - ATRIBUTOS AUTOMATICAMENTE SELECIONADOS *RIVER*.....79
TABELA 35 – MATRIZ DE CONFUSÃO NO CASO DE ESTUDO 3 - ATRIBUTOS AUTOMATICAMENTE SELECIONADOS *RIVER*.....80
TABELA 36 – RESULTADOS OBTIDOS NO CASO DE ESTUDO 4 - ATRIBUTOS AUTOMATICAMENTE SELECIONADOS *PRE-FLOP* 81
TABELA 37 – MATRIZ DE CONFUSÃO NO CASO DE ESTUDO 4 - ATRIBUTOS AUTOMATICAMENTE SELECIONADOS *PRE-FLOP*.81
TABELA 38 – RESULTADOS OBTIDOS NO CASO DE ESTUDO 4 - ATRIBUTOS AUTOMATICAMENTE SELECIONADOS *FLOP*.....83
TABELA 39 – MATRIZ DE CONFUSÃO NO CASO DE ESTUDO 4 - ATRIBUTOS AUTOMATICAMENTE SELECIONADOS *FLOP*.....83
TABELA 40 – RESULTADOS OBTIDOS NO CASO DE ESTUDO 4 - ATRIBUTOS AUTOMATICAMENTE SELECIONADOS *TURN*.....84
TABELA 41 – MATRIZ DE CONFUSÃO NO CASO DE ESTUDO 4 - ATRIBUTOS AUTOMATICAMENTE SELECIONADOS *TURN*.....85
TABELA 42 – RESULTADOS OBTIDOS NO CASO DE ESTUDO 4 - ATRIBUTOS AUTOMATICAMENTE SELECIONADOS *RIVER*.....86
TABELA 43 – MATRIZ DE CONFUSÃO NO CASO DE ESTUDO 4 - ATRIBUTOS AUTOMATICAMENTE SELECIONADOS *RIVER*.....87

Índice de Figuras

Figura 1 - Classificação de jogos 2

Figura 2 – Mãos existentes no poker, ordenadas pelo maior valor 5

Figura 3 - Distribuição das mãos vencedoras 5

Figura 4 - Mesa de poker em *pre-flop* 6

Figura 5 - Ciclo completo de um jogo de Poker Hold'em 7

Figura 6 - *Design Research Model Process* 8

Figura 7 - Exemplo da linguagem PokerLang 18

Figura 8 - Processo KDD 20

Figura 9 - Ciclo da metodologia SEMMA 22

Figura 10 - Ciclo da metodologia CRISP-DM 23

Figura 11 Tarefas a realizar em fase do CRISP-D 24

Figura 12 - Árvore de decisão para a compra de um carro 25

Figura 13 – a) Exemplo de uma neuronal simples; b) Função de transferência e função de ativação num neurónio 26

Figura 14 - a) Plano espacial com 3 tipos de registos e duas distâncias K; b) Plano espacial com registos de 3 tipos distintos) 27

Figura 15 - a) Exemplo de SVM; b) Projeção de dados num plano multidimensional 28

Figura 16 - Ciclo completo de um jogo de *Poker Hold'em* 35

Figura 17 - Amostra de um log de um jogo de poker 37

Figura 18 - Processo ETL 40

Figura 19 - Amostra de dados estruturados após processo ETL 40

Figura 20 - Distribuição do campo Jogada 41

Figura 21 – Primeira parte do processo de ETL 42

Figura 22 – Segunda parte do processo de ETL 42

Figura 23 – Número de jogos e atributos por fase de jogo 43

Figura 24 – Dispersão do valor do atributo "Jogada" na fase de jogo *Pre-flop* e *Flop* 43

Figura 25 – Dispersão do valor do atributo "Jogada" na fase de jogo *Turn* e *River* 44

Figura 26 - Matriz de confusão do algoritmo Nayve Bayes 58

Figura 27 - Nayve Bayes com *cross validation* 58

Figura 28 – Caso de estudo 1 fase de *pre-flop* a) Instâncias corretamente classificadas; b) *Kappa Statistic ;* 60

FIGURA 29 – CASO DE ESTUDO 1 FASE DE *PRE-FLOP ROC AREA* 60
FIGURA 30 – CASO DE ESTUDO 1 FASE DE *FLOP* A) INSTÂNCIAS CORRETAMENTE CLASSIFICADAS; B) *KAPPA STATISTIC ;*.. 61
FIGURA 31 – CASO DE ESTUDO 1 FASE DE *FLOP ROC AREA* 61
FIGURA 32 – CASO DE ESTUDO 1 FASE DE *TURN* A) INSTÂNCIAS CORRETAMENTE CLASSIFICADAS; B) *KAPPA STATISTIC ;*.. 63
FIGURA 33 – CASO DE ESTUDO 1 FASE DE *TURN ROC AREA* 63
FIGURA 34 – CASO DE ESTUDO 1 FASE DE *TURN* A) INSTÂNCIAS CORRETAMENTE CLASSIFICADAS; B) *KAPPA STATISTIC ;*.. 64
FIGURA 35 – CASO DE ESTUDO 1 FASE DE *TURN ROC AREA* 64
FIGURA 36 – *CASO DE ESTUDO 2 FASE DE* PRE-FLOP A) INSTÂNCIAS CORRETAMENTE CLASSIFICADAS; B) *KAPPA STATISTIC ;* 66
FIGURA 37 – *CASO DE ESTUDO 2 FASE DE* PRE-FLOP *ROC AREA* 67
FIGURA 38 – *CASO DE ESTUDO 2 FASE DE* FLOP A) INSTÂNCIAS CORRETAMENTE CLASSIFICADAS; B) *KAPPA STATISTIC ;*.. 68
FIGURA 39 – *CASO DE ESTUDO 2 FASE DE* PRÉ *ROC AREA* 68
FIGURA 40 – *CASO DE ESTUDO 2 FASE* DE *TURN* A) INSTÂNCIAS CORRETAMENTE CLASSIFICADAS; B) *KAPPA STATISTIC ;*.. 70
FIGURA 41 – *CASO DE ESTUDO 2 FASE* DE *TURN ROC AREA* 71
FIGURA 42 – *CASO DE ESTUDO 2 FASE* DE *RIVER* A) INSTÂNCIAS CORRETAMENTE CLASSIFICADAS; B) *KAPPA STATISTIC ;* .. 72
FIGURA 43 – *CASO DE ESTUDO 2 FASE* DE *RIVER ROC AREA* 72
FIGURA 44 – *CASO DE ESTUDO 3 FASE* DE *PRE-FLOP* A) INSTÂNCIAS CORRETAMENTE CLASSIFICADAS; B) *KAPPA STATISTIC ;* 74
FIGURA 45 – *CASO DE ESTUDO 3 FASE* DE *PRE-FLOP ROC AREA* 74
FIGURA 46 – *CASO DE ESTUDO 3 FASE* DE *FLOP* A) INSTÂNCIAS CORRETAMENTE CLASSIFICADAS; B) *KAPPA STATISTIC ;*.. 76
FIGURA 47 – *CASO DE ESTUDO 3 FASE* DE *FLOP ROC AREA* 76
FIGURA 48 – *CASO DE ESTUDO 3 FASE* DE *TURN* A) INSTÂNCIAS CORRETAMENTE CLASSIFICADAS; B) *KAPPA STATISTIC ;*.. 78
FIGURA 49 – *CASO DE ESTUDO 3 FASE* DE *TURN ROC AREA* 78
FIGURA 50 – *CASO DE ESTUDO 3 FASE* DE *RIVER* A) INSTÂNCIAS CORRETAMENTE CLASSIFICADAS; B) *KAPPA STATISTIC ;* .. 80
FIGURA 51 – *CASO DE ESTUDO 3 FASE* DE *RIVER ROC AREA* 80
FIGURA 52 – *CASO DE ESTUDO 4 FASE* DE *PRE-FLOP* A) INSTÂNCIAS CORRETAMENTE CLASSIFICADAS; B) *KAPPA STATISTIC ;* 82
FIGURA 53 – *CASO DE ESTUDO 4 FASE* DE *PRE-FLOP ROC AREA* 82
FIGURA 54 – *CASO DE ESTUDO 4 FASE* DE *FLOP* A) INSTÂNCIAS CORRETAMENTE CLASSIFICADAS; B) *KAPPA STATISTIC ;*.. 84
FIGURA 55 – *CASO DE ESTUDO 4 FASE* DE *FLOP ROC AREA* 84
FIGURA 56 – *CASO DE ESTUDO 4 FASE* DE *TURN* A) INSTÂNCIAS CORRETAMENTE CLASSIFICADAS; B) *KAPPA STATISTIC ;*.. 86
FIGURA 57 – *CASO DE ESTUDO 4 FASE* DE *TURN ROC AREA* 86

FIGURA 58 – *CASO DE ESTUDO 4 FASE* DE *RIVER* A) INSTÂNCIAS CORRETAMENTE CLASSIFICADAS; B) *KAPPA STATISTIC ;* .87
FIGURA 59 – *CASO DE ESTUDO 4 FASE* DE *RIVER ROC AREA* 88
FIGURA 60 – EVOLUÇÃO DAS MÉTRICAS AO LONGO DAS 4 FASES DO POKER NO CASO DE ESTUDO 1 89
FIGURA 61 – EVOLUÇÃO DAS MÉTRICAS AO LONGO DAS 4 FASES DO POKER NO CASO DE ESTUDO 2 90
FIGURA 62 – COMPARAÇÃO ENTRE OS CASOS DE ESTUDO NA FASE DE *PRE-FLOP* E NA FASE DE *FLOP*.....91
FIGURA 63 – COMPARAÇÃO ENTRE OS CASOS DE ESTUDO NA FASE DE *TURN* E NA FASE DE *RIVER*.....92

Lista de acrónimos

LIACC – Laboratório de Inteligência Artificial e Ciência de Computadores.

I.A. – Inteligência Artificial

ETL – *Extract, Transform, Load*

SEMMA – *Sample, Explore, Modify, Model and Assess*

CRISP-DM – *Cross Industry Standard Process for Data mining*

DSRIS – D*esign Science Research in Information Systems*

TCP – *Transmission Control Protocol*

KDD – *knowledge Discovery in Databases*

SVM – *Suport Vector Machines*

XML – *Extensible Markup Language*

ROC – *Receiver Operator Characteristic*

CSV – Comma-Separated Values

ARFF – *Attribute-Relation File Format*

RAM – *Random Access Memory*

1 Introdução

1.1 Enquadramento

Ao longo do tempo, a investigação na área da inteligência artificial (I.A.), sempre foi tendo uma certa vocação para a resolução de problemas que os jogos (principalmente jogos de tabuleiro) apresentam. Exemplo disso é o *Deep Blue* (Campbell, Hoane Jr., & Hsu, 2002) que em 1997 se tornou o primeiro computador a vencer, o na altura campeão mundial de xadrez, *Gary Kasparov*.

Esta apetência da I.A. pelos jogos é bastante natural, pois estes além de conterem problemas complexos de resolver para um agente virtual, possibilitam o confronto direto entre um jogador humano com um agente virtual, facilitando assim a avaliação da capacidade do agente virtual. Além disso os jogos são uma área bastante atraente para o público em geral, o que acaba por levar a que mais pessoas se interessem pela área, ajudando assim ao seu desenvolvimento (Schaeffer & Van den Herik, 2002).

Nos tempos mais recentes, o poker tem ganho um maior destaque, isto porque apresenta problemas completamente diferentes da maioria dos jogos de tabuleiro. Muitos dos jogos de tabuleiro são considerados jogos com informação perfeita (Schaeffer & Van den Herik, 2002), isto é, os jogadores têm acesso a todo o estado do jogo em qualquer momento deste. Ao contrário destes, o poker é considerado um jogo de informação imperfeita (Rubin, 2013), pois um jogador não tem, por exemplo, acesso às cartas do adversário. Além desse fator, o poker é também considerado um jogo estocástico (Johanson, 2013), pois contém o elemento de sorte, ou seja, quando as cartas são distribuídas pelos jogadores passa a existir o elemento de aleatoriedade, podendo sair qualquer uma das cartas que se encontram no baralho. Estes dois fatores tornam o poker um desafio bastante complexo de resolver para os agentes virtuais (Lanctot, Lisý, & Bowling, 2014).

Como se observa pela Figura 1 pode-se classificar os diversos jogos existentes pelas suas características, sendo que os jogos mais complexos para a I.A. situam-se no quadrante superior direito, enquanto no quadrante inferior esquerdo observa-se os jogos onde existe maiores possibilidades da I.A. se superiorizar.

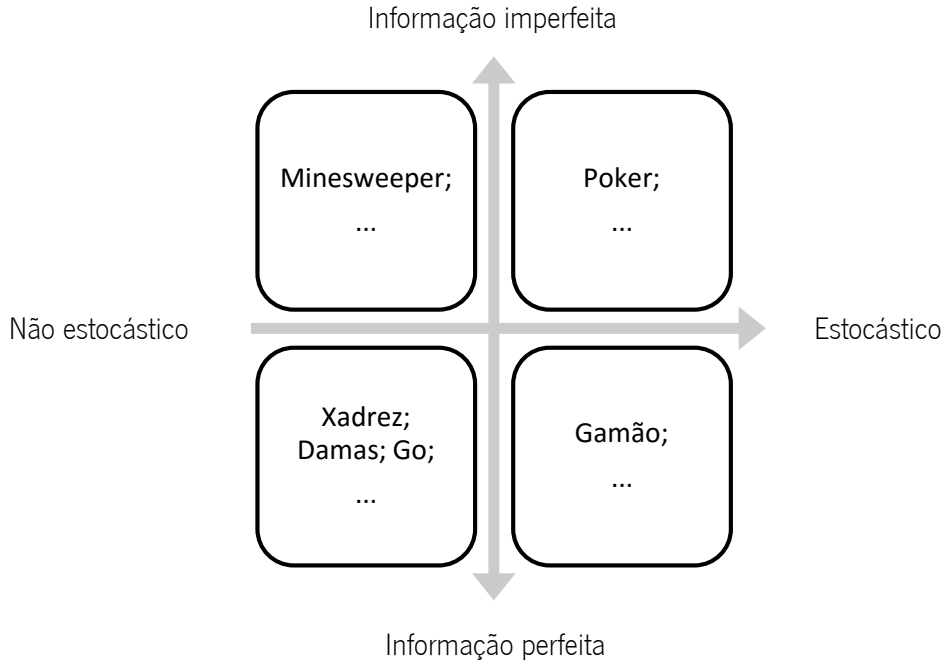


Figura 1 - Classificação de jogos, adaptado de (L. F. Teófilo & Reis, 2011a)

Este elevado interesse pelo poker, surge também pelo recente interesse da população no jogo. Grandes torneios, com prémios elevados como o *European Poker Tour* [1] são televisionados em todo o mundo, tornando-o num jogo bastante conhecido do público em geral. Prova disso é o crescente número de jogadores de poker *online*. Em 2010, Portugal era o terceiro país do mundo com maior percentagem de jogadores de poker *online* por utilizadores de internet (Fiedler & Wilcke, 2011). O mercado do poker *online* tem a capacidade de movimentar milhões de Euros por ano (Fiedler & Wilcke, 2011), tornando-se um mercado bastante atrativo para os jogadores de poker, que podem jogar e alcançar os seus prémios sem saírem da sua residência.

Outro enorme contributo para o crescente interesse da I.A. no poker são as competições a nível mundial de agentes virtuais, como por exemplo a *Annual Computer Poker Competition* [2]. Nestas competições os agentes virtuais são postos à prova competindo uns contra os outros em várias vertentes de poker. Este espirito de competição acaba por fomentar o interesse dos investigadores na área.

1.2 O Poker

O poker é um jogo de cartas muito conhecido nos dias de hoje sendo jogado com um baralho de 52 cartas. Existem várias vertentes do jogo que se podem dividir em três grandes grupos:

[1] http://www.europeanpokertour.com/
[2] http://www.computerpokercompetition.org/

- *Draw poker:* esta variante tem perdido alguma popularidade comercialmente mas ainda é uma variante muito utilizada em jogos que decorrem dentro de casa (entre amigos). No *Draw poker* são distribuídas a cada jogador 5 cartas viradas para baixo (apenas visíveis para o jogador) sendo que a cada ronda de apostas, cada jogador pode escolher uma ou mais cartas para trocar por outras vindas do baralho (Brown, 2011);
- *Stud poker:* uma das primeiras variantes do poker, sendo a sua principal característica o facto de na primeira ronda serem entregues duas cartas a cada jogador, sendo que uma delas se encontra virada para baixo e outra virada para cima. Nas rondas seguintes de apostas, todas as cartas entregues são viradas para cima. Desta forma todos os jogadores na mesa observam todas as cartas dos adversários com exceção da primeira que apenas é visível para o seu dono (Brown, 2011);
- *Community card poker:* esta é provavelmente a variante mais jogada do poker, principalmente na sua vertente *Texas Hold'em* e *Omaha hold'em*. Aqui existem dois tipos de cartas, as cartas do jogador (apenas visíveis e jogáveis para ele) e as cartas comunitárias que se encontram na mesa visíveis e utilizáveis para todos os jogadores. No fim ganha o jogador que no conjunto das suas cartas com as cartas comunitárias consegue ter a mão com o *ranking* mais elevado (Brown, 2011).

Este trabalho científico abrange uma variante do *Community card poker – Texas Hold'em no limit*. Aqui são distribuídas duas cartas a cada jogador e cinco cartas comunitárias. No ponto seguinte iremos verificar o ranking de mãos desta vertente do poker.

1.2.1 Ranking de Mãos

No *Texas Hold'em* existem duas possibilidades de ganhar as fichas que estão em jogo, ou os adversários desistem e o jogador restante em jogo ganha, ou então no fim verifica-se as cartas de todos os jogadores em jogo para se confirmar qual é a mão mais forte (com maior *ranking*). Existem dez mãos possíveis de obter e que estão explicadas de seguida na ordem de menor para maior *ranking*. Na Figura 2 pode-se visualizar as possíveis mãos do jogo, também elas ordenadas de forma decrescente tendo em consideração o seu valor:

- *Straight-flush:* esta é a mão "mais forte do jogo", sendo que é constituída por um conjunto de cartas do mesmo naipe e em sequência. Na Figura 2 pode-se ver um *Straight-flush* de copas, sendo a carta mais alta o 10. Existe ainda um caso especial a que se chama *Royal-flush*, que é a sequência com maior valor e da mesma cor que se pode obter. Um exemplo desta mão é visível na Figura 2, onde se vê um *Royal-flush* de copas.
- *4-of-a-kind:* como o próprio nome indica é a junção de quatro cartas iguais, obviamente de naipes diferentes como se pode ver na Figura 2, onde está presente um *4-of-a-kind* de ases.
- *Full-house:* é composto pela conjugação de duas mãos inferiores: *1-pair* e *3-of-a-kind* como se pode ver na Figura 2, onde se encontra *1-pair* de reis e um *3-of-a-kind* de 8.
- *Flush:* é uma mão constituída por cartas da mesma cor (mesmo naipe). Pode-se ver um exemplo de um *flush* de paus, com a carta mais alta a ser o rei na Figura 2.
- *Straight:* tal como o *straight-flush* trata-se de uma sequência de cartas, mas desta forma podem ser de naipes diferentes, tal como se demonstra na Figura 2 onde o *Straight* tem como carta mais alta o 9 e carta mais baixa o 5.
- *3-of-a-kind:* é constituído por um conjunto de três cartas iguais. Em português é conhecido como um trio. Segue um exemplo na Figura 2, onde se vê um *3-of-a-kind* de *Jack* (Valetes em português).
- *2-pairs:* mais uma vez estamos perante uma mão constituída por duas mãos inferiores. Para uma explicação simples: ver Figura 2 onde estão representados *1-pair* de *Queen* (Damas em português) e *1-pair* de 7.
- *1-pair:* uma das mãos de menor valor no poker sendo constituída por duas cartas iguais de naipes diferentes, como se pode ver na Figura 2, onde se apresenta o par mais alto do jogo, *1-pair* de *Aces* (Ases em português).
- *High-hand*: quando nenhum dos jogadores consegue obter uma das mãos acima citadas é dado como vencedor o jogador que possuir a carta de maior valor. Na Figura 2, pode-se ver um exemplo onde o *King* (Rei em português) é a carta mais alta.



Figura 2 – Mãos existentes no poker, ordenadas pelo maior valor, adaptado de (Bankrollup, 2014)

Apesar do que no *Texas hold'em* ser bastante complicado calcular as probabilidades de uma mão sair vencedora devido à existência das cartas comunitárias, foi efetuada uma simulação para um milhão de mãos (Barr & Dowie, 2011). Os resultados estão presentes na Tabela 1 e na Figura 3.

Tabela 1 - Distribuição da mão vencedora (retirado de (Barr & Dowie, 2011))

	High card	1-pair	2-pairs	3-of-kind	Straight	Flush	Full house	4-of-kind	Straight flush
1 player	0.17412	0.43823	0.23496	0.04830	0.046194	0.030255	0.025961	0.001681	0.000311
2 players	0.05904	0.37712	0.30853	0.07279	0.080101	0.051789	0.047191	0.002927	0.000512
3 players	0.01926	0.30670	0.33242	0.09382	0.108246	0.069779	0.06467	0.004349	0.000748
4 players	0.00588	0.24755	0.33124	0.11183	0.131222	0.084131	0.081349	0.005809	0.000981
5 players	0.00168	0.20114	0.31919	0.12660	0.150822	0.096243	0.095854	0.007193	0.001287
6 players	0.00045	0.16476	0.30221	0.13968	0.167007	0.107222	0.108618	0.008588	0.001472
7 players	0.00011	0.13749	0.28112	0.15091	0.17996	0.118014	0.120452	0.010163	0.0017893

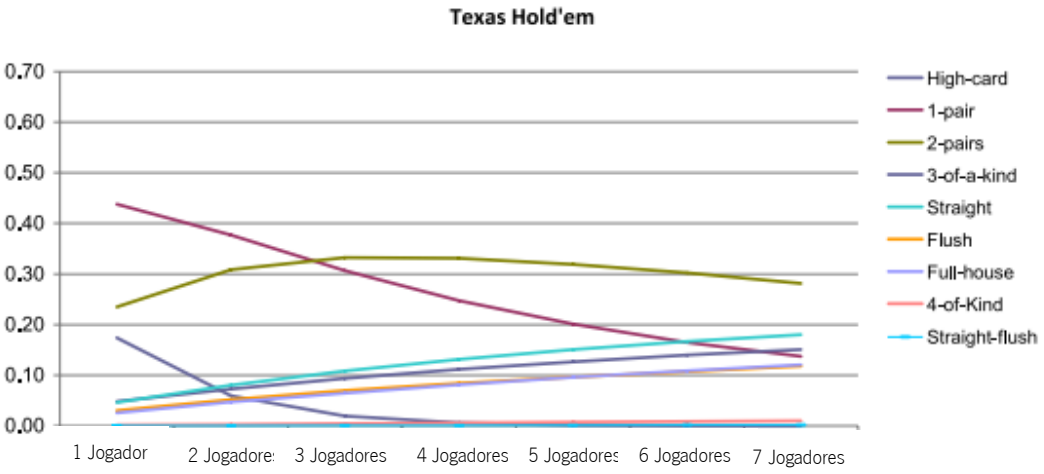


Figura 3 - Distribuição das mãos vencedoras retirado de (Barr & Dowie, 2011)

1.2.2 Fases do Poker

Um jogo de poker começa com uma aposta de um jogador (*small blind*) seguida de outra aposta no dobro do valor de outro jogador (*big blind*). Após estas duas apostas, o dealer (pessoa que distribuí as cartas pelos jogadores) entrega duas cartas a cada jogador, como se observa pela Figura 4 onde encontramos uma mesa no seu estado inicial. Daqui em diante pode-se dividir o poker em duas grandes fases: fase de apostas e fase de colocação de cartas comunitárias.

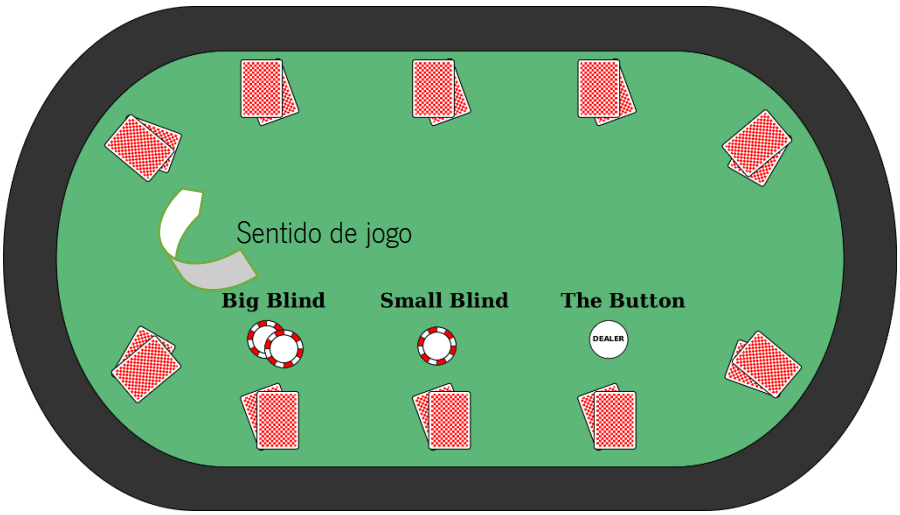


Figura 4 - Mesa de poker em *pre-flop*, adaptado de (Wikipedia, 2014a)

Na Figura 5 verifica-se de uma forma gráfica as fases que compõem um jogo de poker. De salientar que nas fases de apostas, os jogadores que desejarem permanecer em jogo têm que igualar a maior aposta. Caso não possuam valor suficiente para tal, têm que apostar tudo (*all in*) sendo que é criado um novo *pot* (valor apostado por todos os jogadores até ao momento) para os restantes jogadores que ainda tenham possibilidades de apostar.

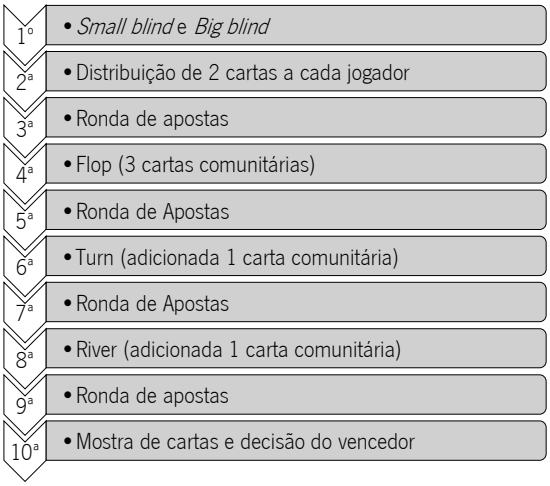


Figura 5 - Ciclo completo de um jogo de Poker Hold'em

Na última fase do jogo são verificadas as cartas dos jogadores que ainda se encontram a disputar o encontro, sendo declarado o jogador que possuir a mão com maior ranking. Ficando este com o valor que se encontra no *pot*.

1.3 Problema de Investigação

Quando falamos de investigação na área de I.A. nos jogos, nomeadamente no poker, normalmente a linha seguida pelos investigadores tem como principal objetivo a criação de um agente virtual capaz de vencer os melhores jogadores humanos. Exemplos disso são os trabalhos de (Darse Billings, Davidson, Schaeffer, & Szafron, 2002; Teofilo, Reis, & Cardoso, 2014). Infelizmente esse caminho é bastante complicado, isto porque um agente de poker tem que lidar com vários tipos de conhecimento (Darse Billings et al., 2002; Korf, 2010) e com um sem número de hipóteses em cada estado do jogo (M. Johanson, 2013).

Desta forma, propomos uma nova linha de estudo com a seguinte questão de investigação:

- Será possível modelar o modo de jogar de um jogador de Poker profissional e transpô-lo para um agente virtual?

Esta questão de investigação será respondida através da criação de um modelo de alto nível do modo de jogar de um jogador de Poker. Seguidamente, através de uma base de dados contendo milhares de jogadas provenientes de vários jogadores profissionais, será realizada a instanciação desse modelo de jogador através de técnicas de data mining que replique a forma de atuar do jogador que lhe

dá origem. Finalmente, será realizado um vasto conjunto de experiências de modo a comparar o modo de jogar do agente virtual e do correspondente jogador real em distintas situações de jogo.

1.4 Abordagem Metodológica

Para esta dissertação, tratando-se de um problema de investigação, mas também com uma componente prática bastante assinalável, será utilizada uma metodologia de investigação e uma metodologia para o desenvolvimento de projetos de *data mining*.

1.4.1 Metodologia de Investigação

Como metodologia de investigação, a escolha recaiu na metodologia *design science research in information systems* (DSRIS) apresentada em (Peffers, Tuunanen, Rothenberger, & Chatterjee, 2008). Na Figura 6 encontra-se uma adaptação dessa metodologia à dissertação que será desenvolvida, onde vemos as seis fases e os resultados que serão alcançados em cada uma delas.

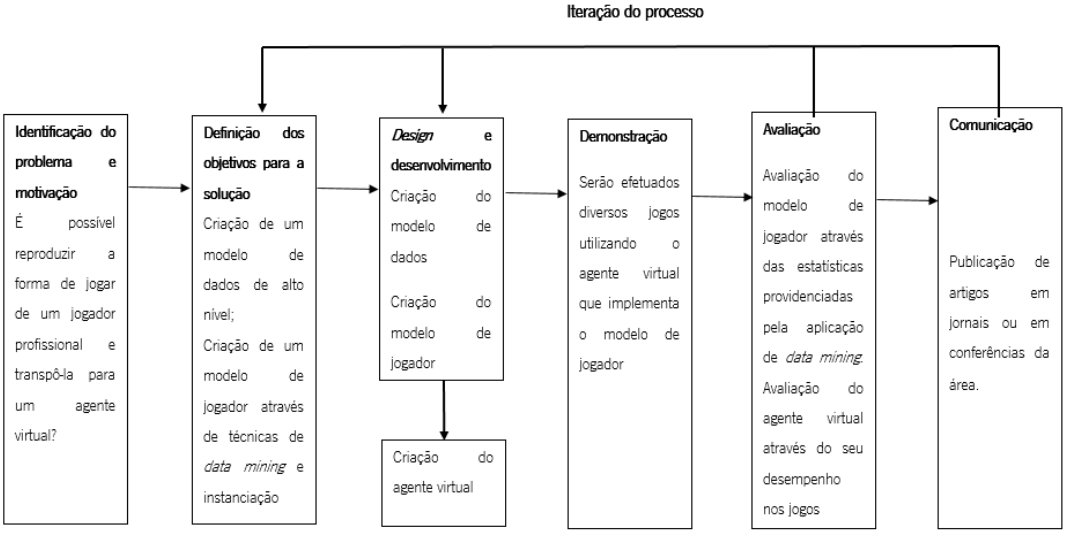


Figura 6 - *Design Research Model Process*, adaptado de (Peffers et al., 2008)

Esta metodologia está dividida em seis atividades, que explicitamos de seguida, assim como uma pequena síntese do que será efetuado em cada uma delas:

- 1ª Identificação do problema e motivação: como problemática, foi identificada a possibilidade de criar um agente virtual capaz de emular a forma de jogar de um

determinado jogador humano através da utilização de técnicas de *data mining*. Sabendo que o poker em si é um problema que tem suscitado muito interesse na comunidade científica, principalmente na investigação na área da I.A. e tratando-se de um jogo muito popular entre a população que tem movimentado milhões de euros anualmente, torna-se bastante estimulante o desenvolvimento de um agente virtual com tais capacidades.

- 2ª Definição de objetivos para a solução: os objetivos passam pela criação de um modelo de dados capaz de comportar a informação necessária à criação de um modelo de jogador e a sua instanciação num agente virtual.
- 3ª *Design* e desenvolvimento: nesta etapa, o objetivo passa pela criação de elementos ou artefactos que contenham modelos ou métodos. Desta forma, nesta fase será executado o trabalho de criação do modelo de dados, do modelo do jogador e do agente virtual
- 4ª Demonstração: como forma de demonstração, o agente virtual será posto à prova em vários jogos contra outros agentes virtuais.
- 5ª Avaliação: a avaliação do trabalho será efetuada em duas situações distintas. Numa primeira etapa será necessário avaliar o modelo do jogador, que será efetuado recorrendo às várias estatísticas oferecidas pela ferramenta de *data mining*. Numa segunda etapa será necessário avaliar o agente virtual, que será feito recorrendo a simulações de jogo contra outros agentes virtuais e com jogadores reais para verificar a sua fiabilidade.
- 6ª Comunicação: a par de todo o trabalho científico desenvolvido, esta dissertação possui também o objetivo de dar origem a algumas publicações relevantes para a comunidade científica. Desta forma espera-se que sejam publicados, ou pelo menos submetidos alguns artigos científicos para algumas conferências relevantes na área das tecnologias de informação.

Completando estas seis etapas com sucesso, espera-se obter um trabalho de investigação científica sério e que cumpra todos os requisitos para tal.

1.4.2 Metodologia de Desenvolvimento

No que toca à metodologia de desenvolvimento foi escolhido o CRISP-DM, pois trata-se de uma metodologia que se adapta inteiramente aos requisitos da parte prática deste trabalho de investigação. Além disso trata-se de uma metodologia simples e com uma curva de aprendizagem bastante rápida.

A metodologia em questão é explicada em 2.3.1.2 CRISP-DM que se situa na página 22, desta forma será apenas explicitado em síntese o trabalho que será efetuado em cada uma das suas fases.

- Compreensão do negócio: apesar de já posuir algum conhecimento sobre o jogo, foi necessário aprofundar o mesmo através da leitura de alguns livros de referência.
- Compreensão dos dados: devido à grande quantidade de dados existente (cerca de 7,5 Gigabytes) será necessário uma boa observação dos mesmos, e provavelmente uma seleção dos dados que contenham mais informação, pois é provável que existam dados em que falte informação essencial para a construção de um modelo de dados;
- Preparação dos dados: como os dados encontram-se em ficheiros de texto, será necessário uma transformação e seleção dos mesmos. Deverá ser utilizada uma ferramenta de ETL (*Extract, Transform, Load*) que permita a leitura e transformação dos mesmos, assim como o carregamento para uma base de dados;
- Modelação: nesta etapa serão testados vários algoritmos de *data mining* de forma a encontrar o que melhor se adapta à nossa questão. Espera-se encontrar algumas dificuldades nesta fase, pois existem vários algoritmos que necessitam de uma grande capacidade de processamento.
- Avaliação: a avaliação dos modelos testados na etapa anterior será efetuada através da comparação de vários dados estatísticos fornecidos pela aplicação de *data mining* utilizada;

A fase de Implementação do CRISP-DM não foi considerada num projeto deste tipo dado não ser possível realizá-la com as restrições temporais do mesmo. Deverá ser realizada em trabalhos futuros.

De salientar que existirá uma tendência para a metodologia CRISP-DM se sobrepor a algumas fases do DSRIS pois abarca todo o trabalho prático a ser realizado.

1.4.3 Questões Éticas

Este trabalho levanta algumas questões éticas, principalmente no que toca à proteção de dados pessoais, e ao anonimato das pessoas, isto porque os dados fornecidos e que foram utilizados na implementação da solução contêm dados pessoais, como por exemplo o nome de utilizador usado durante os jogos.

Apesar de a base de dados ter sido fornecida sem qualquer pedido de confidencialidade relativamente a esses dados, foram efetuados todos os cuidados possíveis para que nenhuma informação relevante fosse divulgada.

Salienta-se também, que toda a informação recolhida, perfis criados, ou outro qualquer tipo de informação criada, nunca será utilizada para fins comerciais, sendo que esta investigação será apenas utilizada para fins académicos.

1.5 Objetivos

Esta dissertação tem como principal objetivo a modelação de jogadores reais de poker que posteriormente serão utilizados num simulador, de modo a que seja possível verificar a fidelidade do modelo.

Para tal foram definidos os seguintes três objetivos específicos:

- Criação de um modelo de alto-nível (isto é, baseado em informação de alto-nível e ações de alto-nível) do que é um jogador de Poker;
- Definir e implementar métodos de análise e transformação de dados para que seja possível aplicar técnicas de *data mining* em bases de dados com milhares de jogadas de jogadores reais de Poker;
- Aplicação de algoritmos de *data mining* para prever que jogada seria efetuada pelo jogador real tendo em conta as circunstâncias de jogo;

Por fim, é esperado que o agente tome decisões semelhantes às decisões que o jogador, que lhe serve de modelo, tomaria perante determinada situação.

1.6 Estrutura do documento

Para além da presente introdução, o restante documento contém mais cinco capítulos:

- Revisão do estado da arte: aqui são abordados os dois temas principais e necessários para o desenvolvimento desta dissertação. O poker computorizado, onde se pode ver alguns dos maiores centros de investigação na área, as estratégias utilizadas para a construção de agentes virtuais e alguns dos agentes criados ao longo do tempo. Foram também analisadas abstrações utilizadas de forma a diminuir a complexidade do jogo. Em relação ao *data mining* abordamos as metodologias e métodos existentes.
- Desenvolvimento: nesta secção está presente uma explicação extensiva de todo o trabalho realizado. Está estruturada de forma semelhante ao CRISP-DM, metodologia seguida para o desenvolvimento do trabalho prático desta dissertação.
- Experiências e Resultados: nesta parte da dissertação é possível verificar os resultados obtidos durante as experiências efetuadas para a realização desta dissertação.
- Conclusões e trabalho futuro: neste ponto pode-se verificar as conclusões que foram possíveis retirar do trabalho desenvolvido.

No fim deste documento é possível encontrar as referências bibliográficas que o suportam.

2 Estado da Arte

Um estado de arte bem efetuado é um elemento essencial para a conclusão de um processo de investigação com sucesso. Neste capítulo será efetuado o desenvolvimento do estado da arte relativo à investigação sobre a inteligência artificiar no poker.

Na primeira secção será demonstrada a forma como foi efetuada a pesquisa e como foram selecionados os artigos que fazem parte deste estado da arte.

Na secção 2.2 serão abordadas várias temáticas sobre o poker computorizado, desde as abordagens utilizadas, até aos agentes que foram desenvolvidos ao longo do tempo.

Na secção 2.3 abordamos a temática do *data mining*, onde se verifica em que consiste, as metodologias e métodos utilizados.

Por fim no último ponto, pode-se observar algumas conclusões retiradas desta revisão do estado da arte.

2.1 Processo de Identificação da Literatura e Estruturação da Revisão

Para se efetuar um bom estado de arte é necessário primeiro executar uma boa seleção de bibliografia para ser lida. Desta forma foi criado um processo de identificação de literatura de forma a escolher os artigos mais adequados para serem lidos na íntegra.

Desta forma e dado o atual estado tecnológico da era em que vivemos, toda a pesquisa bibliográfica foi efetuada *online*. Dado que a maior parte dos investigadores possuem os seus trabalhos publicados em serviços de indexação *online* como por exemplo:

- Scopus:
- ISI Web of Knowledge;
- Google Scholar;
- IEEE Xplore;
- Springer Link;
- RepositóriUM;
- Repositório-aberto da FEUP - Faculdade de Engenharia.

Foi nestes serviços de indexação que foram utilizadas palavras-chave como: "poker survey", "*poker investigation review*", "*poker review*", "*case based reasoning poker*", "*modeling poker player*", "*opponent modeling in poker* ", "*poker agents*", "*data mining*", "*data mining techniques*","*user modeling*", "*data mining survey*".

Para filtrar os resultados obtidos nestas pesquisas, foram selecionados os resultados a partir de 2010 (inclusive) e, caso existam, o número de citações. De realçar que o número de citações tem que ser observado com algumas reticências, pois além de nem todos os serviços de indexação disponibilizarem o número de citações, esse mesmo número poderá não indicar a totalidade de citações feitas ao artigo, devido a essas mesmas citações poderem ser efetuadas de diversas formas e nem todas elas serem registadas. Após esta primeira seleção, foi lido o *abstract* dos artigos de forma a aferir a sua importância para o tema.

Além destes resultados, foram incluídos ainda alguns trabalhos de referência na área mesmo tendo já alguma uma idade, pois são trabalhos que ajudam a compreender o âmbito do tema e artigos de qualidade irrefutável. Foram também incluídos trabalhos indicados pelo orientador.

2.2 Poker Computorizado

2.2.1 Locais de Investigação

A investigação em torno do poker tem em muito beneficiado do esforço de alguns centros de investigação espalhados pelo mundo. A nível nacional, a investigação nesta área ainda não se encontra muito difundida, centrando-se essencialmente no LIACC – Laboratório de Inteligência Artificial e Ciência de Computadores[3].

Ao longo da sua existência foram desenvolvidos vários trabalhos de boa qualidade, de entre os quais se pode destacar: (L. F. Teófilo, Reis, Cardoso, et al., 2013) onde se vê vários dos trabalhos desenvolvidos.

A nível internacional, pode-se afirmar que a *University of Alberta* com o seu *Computer Poker Research Group* [4] é provavelmente uma das mais importantes universidades no que toca à investigação de inteligência artificial no poker. No seu já longo reportório, o grupo conta com já vários agentes desenvolvidos, como o *Loki, Poki, PsOpti/Sparbot, Vexbot, Hyperborean, Polaris, Hyperborean No-Limit e*

[3] http://www.liacc.up.pt/
[4] http://poker.cs.ualberta.ca/

Hyperborean Ring (University of Alberta Computer Poker Research Group, 2015). Grande parte do sucesso destes agentes deve-se em grande parte à utilização da estratégia do Equilíbrio de *Nash* (Nash, 1950, 1951).

Outra universidade internacional que se encontra na vanguarda da investigação de agentes de poker é a *Carnegie Mellon University*, que conta também com vários documentos científicos publicados. Diversos outros centros de investigação têm utilizado também este domínio de investigação, tais como *University of South Wales, University College London, Georgetown University, John Hopkins University, Knowledge Engineering Group - Technische Universität Darmstadt* e *The University of Auckland.*

Todos estes centros de investigação, assim como o proliferar de competições e o aumento do interesse do poker a nível internacional, tem feito com que a investigação neste domínio também tenha sofrido um elevado acréscimo.

2.2.2 Abordagens para a Construção de Agentes de Poker

Ao longo dos anos foram desenvolvidas várias abordagens que permitiram a construção de agentes de poker. Nesta secção vamos dar uma visão geral das abordagens mais significativas utilizadas e das suas vantagens e desvantagens.

- *Rule-Based:* esta abordagem consiste num conjunto de regras baseadas em conhecimento de pessoas com profunda compreensão da área. São simples regras do tipo "se – então" criadas para todos os cenários possíveis. Cada regra pode ter uma ou mais condições de forma a ser ativada (Rubin & Watson, 2011).
- *Formula-Based:* bastante idêntica à abordagem *Rule Based* mas mais generalista, não sendo necessária a criação de regras para todos os cenários, pois uma fórmula que conjugue toda a informação possível sobre o estado de jogo, irá resultar numa probabilidade tripla e uma decisão de aposta, sendo depois selecionada uma opção tendo em conta as probabilidades calculadas (Rubin & Watson, 2011).
- *Simulation-Based:* a estratégia baseada em simulação cria uma árvore de pesquisa com todas as possibilidades, de forma a encontrar a melhor jogada possível. Pode também ser utilizado o algoritmo de *Monte-Carlo* (Rubin & Watson, 2011).
- *Nash Equilibrium:* são estratégias estáticas e robustas com o objetivo de limitar ao máximo as suas debilidades contra um adversário perfeito. Apesar de ser uma estratégia bastante forte num jogo de 2 jogadores, necessita também de um bom

modelo do adversário para poder tirar o máximo partido de jogadores mais fracos (Rubin & Watson, 2011).

- *Frequentist Best Response:* é uma abordagem que constrói um modelo do adversário (*offline*) através de jogos históricos tentando explorar as debilidades da estratégia do adversário. É uma abordagem estática, não se adapta a mudanças na estratégia do oponente (Rubin & Watson, 2011).

2.2.3 Agentes de Poker

Ao longo que a investigação nesta temática foi evoluindo, foram-se construindo e testando vários agentes virtuais, sempre com a intenção de melhorar os resultados obtidos anteriormente.

Um dos primeiros agentes virtuais a aparecer foi o *Loki* (Papp, 1998), que teve várias melhorias ao longo do tempo e que implementou com sucesso a modelação do oponente (Darse Billings, Papp, Schaeffer, & Szafron, 1998). Isto permitiu ao *Loki* adaptar-se dinamicamente à estratégia utilizada pelo seu adversário, melhorando assim os seus resultados.

Mais tarde surgiu o *Poki* (Davidson, 2002), que foi uma evolução do *Loki*, reescrevendo o seu sistema para uma abordagem baseada em fórmulas. Utilizando também a modelação do oponente, conseguiu atingir um lucro consistente nos jogos do servidor do IRC. (Rubin, 2013)

Encontra-se outros exemplos de agentes virtuais como o *PsOpti*/ *Sparbot* (D Billings et al., 2003) (Schauenberg, 2006), *Hyperborean* (Zinkevich, Bowling, Johanson, & Piccione, 2007). *Polaris* (Johanson, 2007), *Hyperborean No-Limit* (Schnizlein, 2009), *Hyperborean* Ring (Risk, 2009) e mais recentemente o *Cepheus* (Bowling, Burch, Johanson, & Tammelin, 2015) que é o primeiro agente virtual a tornar-se invencível contra jogadores reais na vertente *heads'up* (apenas 2 jogadores).

Na vertente *Texas Hold'em No Limit* encontramos o primeiro agente capaz de igualar jogadores reais em ambientes de multijogadores. Criado através de uma estratégia baseada em regras *(rules-based)* e utilizando modelação do oponente, foi capaz de obter lucros em jogos *online* com mesas até seis jogadores (Teofilo et al., 2014). Foram também tentadas outras abordagens como a utilização de algoritmos genéticos, demonstrado a sua possível utilização (Nicolai & Hilderman, 2009).

2.2.4 Abstrações

Um dos fatores que tornam o poker tão interessante para a I.A. é o facto de ser um jogo com um tamanho bastante elevado, sendo desta forma complicado de reproduzir em agentes virtuais de forma eficiente (L. F. Teófilo, Rossetti, Reis, Cardoso, & Nogueira, 2013). Uma das formas de contornar esse problema é recorrer a abstrações de alto nível que retiram tamanho ao jogo.

Existem dois tipos de abstrações: as que implementam o jogo sem alterar a sua complexidade (*lossless abstractions*) e as que desrespeitam isso (*lossy abstractions*) (Rubin & Watson, 2011).

Imaginemos apenas a fase de *pre-flop* (fase onde ainda não existem cartas comunitárias), onde são distribuídas duas cartas a cada jogador. Isto gera 1 624 350 combinações possíveis para apenas dois jogadores. De forma a reduzir-se este número de possibilidades, e tendo consciência que na fase de *pre-flop* a única informação relativa ao naipe das cartas que é necessário reter é o facto de as cartas pertencerem ou não ao mesmo naipe, pode-se transformar a informação das cartas. Ou seja, obter A♥K♣ é totalmente indiferente de obter A♦K♠, sendo assim é possível abstrair estas duas mãos para "AKd", onde o "d" significa que são de naipes diferentes. Enquanto que obter A♦K♦ pode ser abstraído para "AKi", onde "i" significa que pertencem ao mesmo naipe. Esta simples abstração na fase de *pre-flop* permite reduzir as possibilidades de 1 624 350 para 28 561 e não é perdida nenhuma informação relevante para esta fase do jogo, por isso pode-se considerar uma *lossless abstraction*. Mas se pensarmos na fases seguintes do jogo, onde são distribuídas as cartas comunitárias, a informação do naipe das cartas torna-se muito relevante e neste momento torna-se impossível reaver essa informação (Rubin & Watson, 2011).

Outra abstração muito utilizada em agentes virtuais é o *bucketing*, que agrupa as mãos dos jogadores em várias classes tendo em conta a sua possibilidade de vitória (*roll-out hand strength*) e/ou a possibilidade de se tornarem mãos de valor elevado com o decorrer do jogo (*expected hand strenght* (L. F. Teófilo, Reis, & Cardoso, 2013)). Tendo em conta que se perde toda a informação relativa às cartas, esta abstração é considerada uma *lossy abstraction* (Rubin & Watson, 2011).

As abstrações são ferramentas poderosas, mas é necessário alguma precaução na sua utilização, pois pode-se perder informação relevante e necessária para fases posteriores do jogo.

2.2.5 PokerLang

O PokerLang é uma linguagem que foi criada com o objetivo de formalizar, utilizando *Backus Naur Form* (McCracken & Reilly, 2003), os principais conceitos e regras do poker para a criação de agentes virtuais (L. F. Teófilo, Reis, Cardoso, et al., 2013). Desta forma, qualquer utilizador sem grande experiência de programação, mas habituado à temática do poker, poderia criar novos agentes.

Baseada no *Coach Unilang*, define todos os conceitos utilizados de forma hierárquica, começando por definir o conceito de "estratégia" sendo esta um conjunto de "táticas". Na Figura 7 vê-se um pequeno excerto da definição da linguagem PokerLang onde estão definidos conceitos como "estratégia", "tática", "comportamento", entre outros.

```
<STRATEGY>::= {<TACTIC>}
<TACTIC>::= <ACTIVATION_CONDITION> <TACTIC_BEHAVIOUR>
<ACTIVATION_CONDITION>::= {<EVALUATOR>}
<TACTIC_BEHAVIOUR>::= <PREDEFINED_BEHAVIOUR>|<BEHAVIOUR>
<PREDEFINED_BEHAVIOUR>::= loose_agressive|loose_passive|
                          tight_agressive|tight_passive
<BEHAVIOUR>::= {<RULE>}
<RULE>::= {<EVALUATOR> | <PREDICTOR>} <ACTION> <VALUE>
<ACTION>::={<PREDEFINED_ACTION><PERC>|
             <DEFINED_ACTION><PERC>}
```

Figura 7 - Exemplo da linguagem PokerLang (retirado de (L. Teófilo, Reis, Cardoso, & Mendes, 2014))

Devido ao facto de ser um jogo estocástico e de informação imperfeita, o poker torna-se um jogo demasiado grande para calcular todas as possibilidades em cada uma das suas fases (M. Johanson, 2013). De forma a contornar essa característica, o PokerLang implementa várias abstrações como por exemplo o *Hand Region* que faz uso de cinco grupos de cartas para as agrupar tendo em conta a sua probabilidade de vitória. Existem outros conceitos utilizados no PokerLang, como a mão do oponente (calcula de uma forma probabilística a mão do adversário tendo em conta as cartas conhecidas), tipo de jogador (agressivo, passivo) e posição na mesa.

2.2.6 Framework HoldemML

A *Framework HoldemML* foi construída com o intuito de facilitar a criação de agentes virtuais de poker através de registos antigos em mesas *online* (Teofilo & Reis, 2011).

A *framework* é capaz de gerar um agente apenas com o *input* de fontes de dados. Para tal a *framework* inclui vários módulos como o *Converter* e o módulo *Validator*. O módulo *Converter*, transforma

as fontes de dados num ficheiro chamado *Game Log* que contém uma estrutura fixa e escrito em XML (*Extensible Markup Language*). O modo *Validator* trata de validar se o ficheiro XML se encontra bem preenchido tendo em conta uma estrutura previamente definida.

Utilizando o ficheiro *Game Log*, o módulo *Player List Extractor* cria um novo ficheiro com a lista de jogadores encontrados assim como a sua *performance* durante os jogos que participou. O módulo *Stats Extractor* cria um segundo ficheiro com as estatísticas de jogo.

Estes dois novos ficheiros são utilizados pelo *Strategy Generator* que cria as estratégias a serem utilizadas pelos agentes criados.

Esta *framework* mostra-se capaz de criar agentes virtuais de uma forma fácil e rápida. Porém os agentes criados demonstraram-se pouco eficazes contra bons jogadores humanos, muito por causa da utilização de uma estratégia fixa e da ausência da modulação do oponente.

2.3 *Data Mining*

Ao longo dos últimos anos, a quantidade de dados movimentada e guardada em bases de dados tornou-se algo praticamente imensurável. Para as organizações os dados armazenados não são benéficos a não ser que seja possível extrair conhecimento dos mesmos. O *data mining* recorre a ténicas e à estatística para descobrir padrões e relações nos dados (Drachen, Thurau, Togelius, & Bauckhage, 2013), fornecendo assim conhecimento. Este conhecimento adquirido através de padrões e relações nos dados pode depois ser utilizado em diversos campos da sociedade, como o marketing, investimentos, deteção de fraude entre outros (Fayyad, Piatetsky-Shapiro, & Smyth, 1996), (Liao, Chu, & Hsiao, 2012).

A este processo de descobrir conhecimento numa base de dados dá-se o nome de KDD (*knowledge discovery in databases*) – Descorberta de Conhecimento em Base de Dados. Na Figura 8 pode-se visualizar todas as etapas de um processo KDD, onde o *data mining* aparece como uma das etapas finais do mesmo.

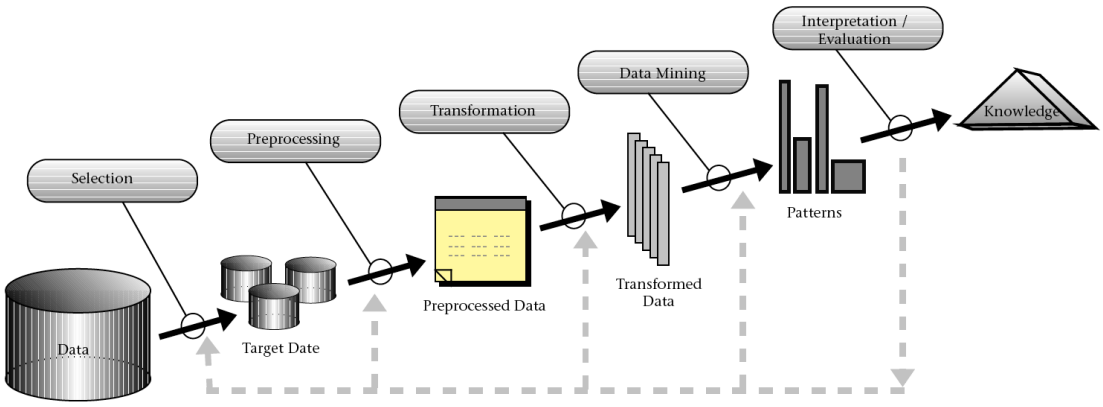


Figura 8 - Processo KDD (retirado de (Fayyad et al., 1996))

Através do conhecimento adquirido é possível criar modelos capazes de prever certos acontecimentos, categorizar dados e criar perfis de pessoas que possuem algo em comum. Apesar de ser possível responder a algumas questões, o *data mining* detém algumas limitações. Este baseia-se em dados históricos podendo acontecer vários problemas provenientes dos mesmos, como por exemplo serem bastante antigos, ou serem em quantidade insuficiente para representar toda a população (North, 2012). Além dessas situações, é preciso relembrar que o *data mining* procura padrões e relações nos dados, por isso caso não exista nenhuma relação ou nenhum padrão, não será possível obter bons resultados.

Para a obtenção de bons resultados é também necessário ter uma abordagem correta ao problema. Devem ser criados objetivos sérios e capazes de serem atingidos, recolhendo os dados corretos e modificando-os caso seja necessário, entre outros. Para auxiliar nesses passos ou tarefas, foram criadas algumas metodologias que serão abordadas no próximo ponto.

2.3.1 Metodologias

As metodologias servem principalmente como guias para a aplicação de uma determinada tecnologia. No que concerne ao *data mining* existem duas metodologias muito usadas, o CRISP-DM (*Cross Industry Standard Process for Data mining*) e a SEMMA (*Sample, Explore, Modify, Model and Assess*).

Apesar destas duas metodologias serem parecidas, cada uma possui as suas especificidades e as suas características. Na Tabela 2 pode-se ver as várias fases das duas metodologias de forma a obter uma comparação mais precisa.

Tabela 2 - Comparação de características entre SEMMA e CRISP-DM (adaptado de (Azevedo & Santos, 2008))

SEMMA	CRISP-DM
————	Compreensão do negócio
Amostra	Compreensão de dados
Exploração	
Modificação	Preparação de dados
Modelo	Modelação
Avaliação	Avaliar
————	Implementação

Nas próximas duas subseções será dada uma visão mais aprofundada das duas metodologias em questão.

2.3.1.1 SEMMA

A metodologia SEMMA foi desenvolvida pela empresa *SAS Institute*, e consiste num ciclo de cinco passos (*Sample, Explore, Modify, Model e Assess*) como se pode ver na Figura 9 (Rohanizadeh & Moghadam, 2009).

A primeira fase da metodologia – *Sample* (amostra) – consiste na análise de uma pequena amostra dos dados (Rohanizadeh & Moghadam, 2009) de forma a adquirir um conhecimento dos dados. De seguida passamos para a fase da Explore (exploração) dos dados de forma a fazer uma análise exploratória dos dados (Obenshain, 2004) para que seja possível verificar a existência de erros nos dados (Rohanizadeh & Moghadam, 2009), como por exemplo *nulls* ou *outliers*. Tendo em conta a informação recolhida na fase anterior, na maior parte das situações surge a necessidade de fazer transformações aos dados adquiridos, e dessa necessidade surge a terceira fase da metodologia – *Modify* (modificação) – onde o utilizador pode modificar, transformar e selecionar (Rohanizadeh & Moghadam, 2009) os dados que pretende utilizar no decorrer do projeto. Na seguinte fase – *Model* (modelo) – são executadas as técnicas de *data mining* para a construção do modelo. Por fim, na fase de *Assess* (avaliação) é efetuada a avaliação ao modelo através das estatísticas recolhidas na aplicação utilizada.

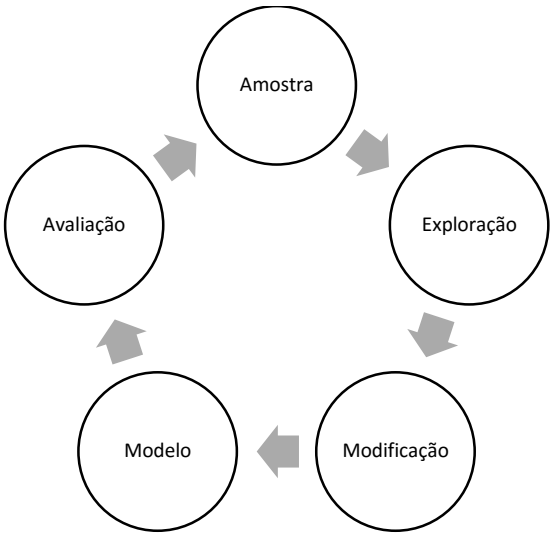


Figura 9 - Ciclo da metodologia SEMMA

Ao contrário do CRISP-DM, a metodologia SEMMA não comporta algumas das partes fundamentais de um sistema de informação (Rohanizadeh & Moghadam, 2009), como por exemplo a compreensão do negócio que é essencial quando se fala de um projeto de *data mining* (Shearer, 2000). Outra limitação da metodologia prende-se com o facto desta se encontrar desenhada para trabalhar com o *software* da *SAS Institute*, não sendo assim utilizável com outras ferramentas (Rohanizadeh & Moghadam, 2009).

2.3.1.2 CRISP-DM

O CRISP-DM é uma metodologia que foi criada em 1999, numa altura em que o mercado de *data mining* ainda era bastante novo e reduzido. Um conjunto de empresas tomaram a decisão de trabalhar juntas de modo a construírem um método *standard* para a abordagem de problemas de *data mining* (North, 2012). A ideia principal para o desenvolvimento desta metodologia foi a criação de algo que fosse facilmente adaptável a todos os mercados e indústrias (Chapman et al., 2000) e independente das tecnologias escolhidas (Santos, 2001).

O CRISP-DM tem por base um modelo que mostra o ciclo de vida de um projeto de *data mining* (Santos, 2001) assim como as suas principais fases. Essas fases encontram-se depois divididas em várias tarefas e em artefactos que devem ser desenvolvidos.

Na Figura 10 pode-se ver o ciclo de vida sugerido pelo CRISP-DM de um projeto de *data mining*. Observa-se as suas seis fases principais, assim como a sua ordem de execução e todas as possibilidades de retroceder para uma fase anterior. Pode-se ainda verificar que se trata de um ciclo sem fim que tem

como principal objetivo o melhoramento contínuo do negócio através de incessantes melhorias no projeto.

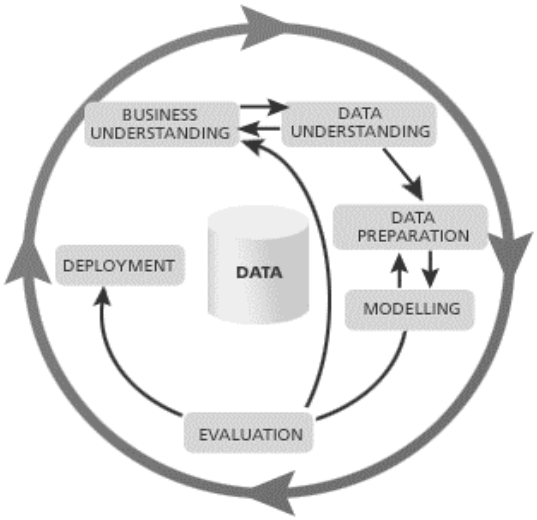


Figura 10 - Ciclo da metodologia CRISP-DM (retirado de (Patel, 2012))

Cada fase do CRISP-DM tem uma função muito específica dentro da metodologia e todas elas são necessárias para que se possa efetuar um projeto de *data mining* com sucesso. De seguida fica uma explicação sucinta do que se pretende em cada uma das fases. Uma explicação mais aprofundada pode ser encontrada em (Chapman et al., 2000).

1. **Compreensão do negócio:** nesta primeira fase do CRISP-DM o foco principal é no negócio e na sua compreensão, isto é, antes de se partir para um projeto de *data-mining* é necessário perceber todo o âmbito do negócio sobre o qual iremos trabalhar.
2. **Compreensão dos dado**s: após ter toda a informação necessária sobre o negócio, chega a altura de se olhar para os dados de forma a se perceber que dados se possui e qual a sua qualidade.
3. **Preparação dos dados:** esta fase é muito importante para que se consiga obter um resultado final excelente. É aqui que se seleciona os dados de forma a construir um *data set* com informação de qualidade para que numa fase posterior seja possível aplicar técnicas de *data mining*.
4. **Modelação:** é aqui que serão executadas as mais variadas técnicas de *data mining*, que podem ser agrupadas em árvores de decisão, regras de associação, redes neuronais, algoritmos genéticos, aproximação de vizinhanças (Santos, 2001).

5. **Avaliação:** após terem sido executados os modelos, analisa-se os seus resultados verificando se são suficientemente bons para serem implementados. Nesta fase olhamos a curva ROC (*receiver operator characteristic*) ou a matriz confusão.
6. **Implementação:** depois de executadas as fases anteriores, atinge-se por fim a fase de implementação do projeto.

Na Figura 11 temos acesso às tarefas que devem realizar-se em cada uma das seis fases do CRISP-DM. Como é possível verificar, as fases são: compreensão do negócio (*Business Understanding*), compreensão dos dados (*Data Understanding*), preparação dos dados (*Data Preparation*), modelação (*Modelling*), avaliação (*Evaluation*) e implementação (*Deployment*) contêm várias tarefas a realizar (a negrito) e artefactos a realizar (em itálico) (Chapman et al., 2000).

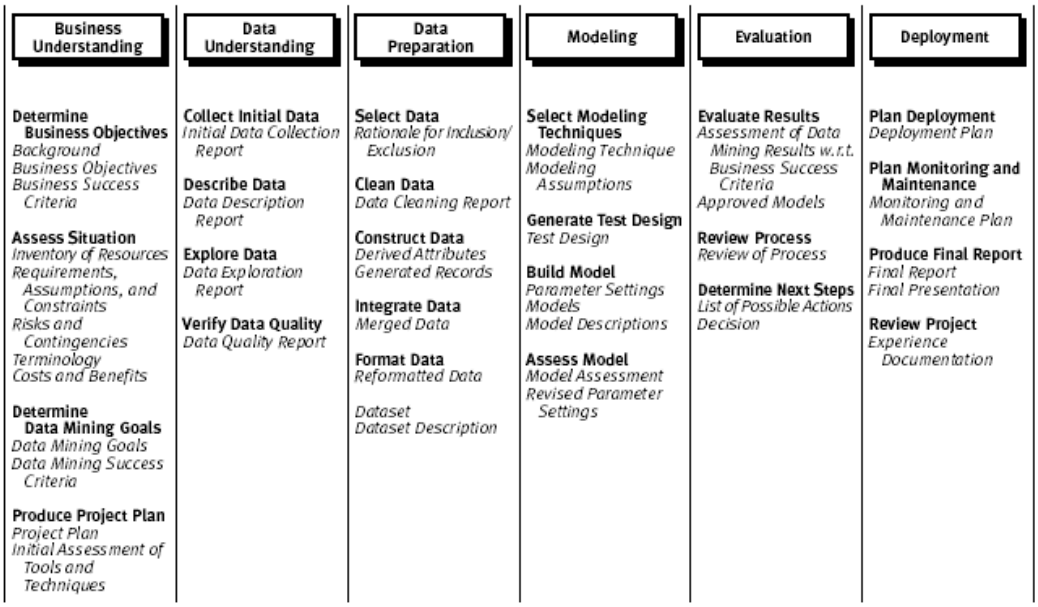


Figura 11 Tarefas a realizar em fase do CRISP-DM (retirado de (Dataprix, 2015))

Apesar do CRISP-DM não sofrer atualizações há bastante tempo, o que tem feito com que fique desatualizado tendo em conta a evolução e introdução de novos conceitos e tecnologias como o *Big Data*, esta continua a ser uma das mais utilizadas a nível mundial (Piatetsky-shapiro, 2014).

2.3.2 Métodos de Data Mining

Ao longo da evolução do *data mining* foram desenvolvidos vários métodos e a classificação dos mesmos é muitas vezes uma matéria muito sensível, com a exceção da classificação em descritivos/preditivos e supervisionada/não supervisionada (Drachen, Thurau, et al., 2013).

Nas próximas subsecções serão demonstrados alguns dos métodos mais usuais em projetos de *data mining* sendo que serão analisadas algumas das suas principais qualidades bem como das suas lacunas.

2.3.2.1 Árvores de decisão

As árvores de decisão são, tal como o nome indica, estruturas em forma de árvore, constituídas por nodos e ramos. Nos nodos encontram-se representados os atributos, enquanto nos ramos encontram-se os valores que esses atributos podem possuir (Santos, 2001). Na Figura 12 observa-se um exemplo de uma possível árvore de decisão para a compra de um carro.

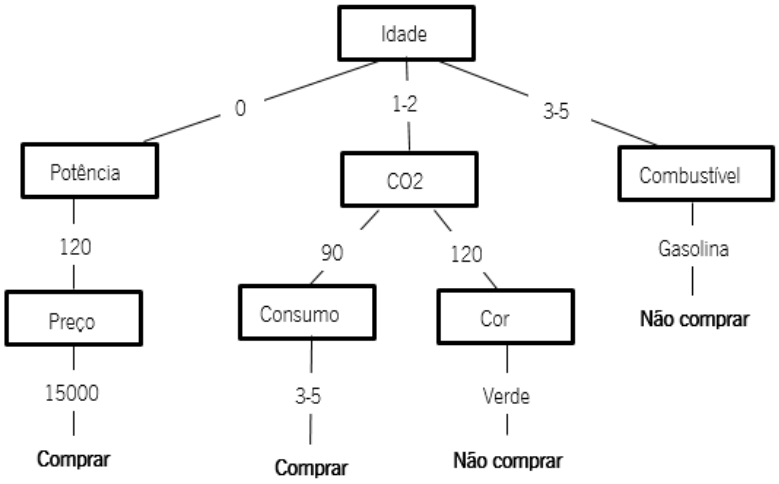


Figura 12 - Árvore de decisão para a compra de um carro

As árvores de decisão possuem a vantagem de serem facilmente interpretadas, sendo que mesmo após a construção da árvore, a mesma pode ser melhorada através de um *tree-pruning* (podar a árvore) melhorando assim a sua capacidade de modelar o problema (Faria, 2013). O facto de a árvore apresentar apenas os atributos necessários para determinar as classes, poderá mais tarde ajudar a reduzir o tamanho do problema quando se experimentar métodos que necessitem de uma maior capacidade de computação, como as redes neuronais (Santos, 2001). Apesar destas vantagens, as árvores de decisão são suscetíveis ao fenómeno de *over-fitting* (Faria, 2013), principalmente quando existem poucos dados de teste.

2.3.2.2 Regras de associação

As regras de associação permitem relacionar os atributos existentes num *data set* formando uma representação do tipo: Se *X* então *Y*, sendo *X* um subconjunto de atributos contidos no *data set* e *Y*

representa um único atributo do mesmo *data set* e que não esteja incluído em *X* (Santos, 2001). São normalmente utilizadas em atributos não numéricos (Witten, Frank, & Hall, 2001).

A qualidade das regras de associação é medida através de três fatores: força, confiança e significância estatística. Por exemplo a regra: "fato" e "sapatos" = "camisa" (223: 20%, 0.73) indica que na base de dados é possível encontrar 223 vezes a ocorrência entre a compra de "fato", "sapatos" e "camisa", sendo que esses 223 registos representam 20% da base de dados. Por fim, o último valor representa que 73% das pessoas que compra "fato" e "sapatos" compra também "camisa".

2.3.2.3 Redes Neuronais

As redes neuronais consistem na criação de uma rede de nodos interligados e dispostos em vários níveis de forma a simular o sistema nervoso humano (Santos, 2001). Os nodos apenas se encontram ligados aos nodos do nível seguinte (Faria, 2013). Uma rede neuronal é composta por pelo menos três níveis, sendo que o primeiro nível é o de entrada e o último é o de saída. No meio destes dois níveis é possível ter um ou mais níveis escondidos, sendo que quanto maior o número de níveis, maior a complexidade da rede neuronal e maior o tempo de computação. Na Figura 13 a) observa-se um exemplo simplificado de uma rede neuronal com três níveis, sendo um deles oculto. Na Figura 13 b) observamos o que se passa num neurónio artificial.

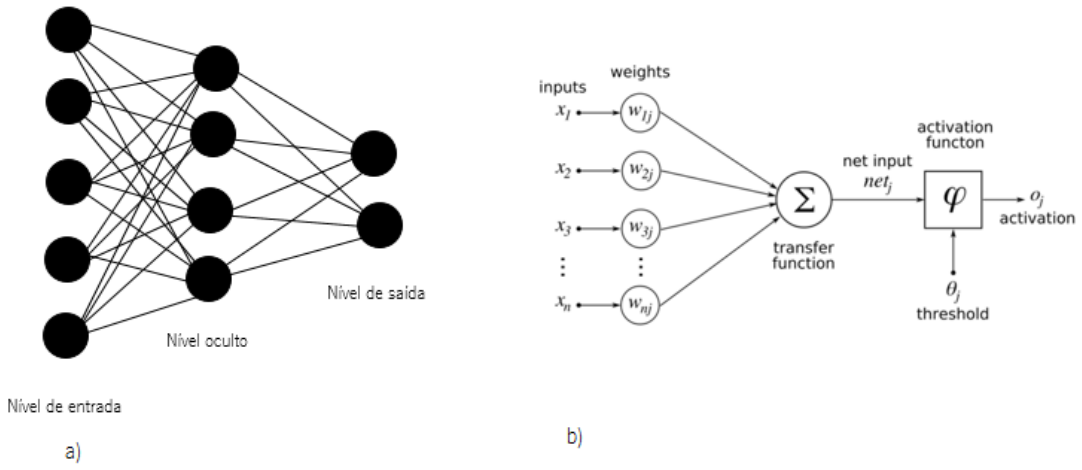


Figura 13 – a) Exemplo de uma neuronal simples; b) Função de transferência e função de ativação num neurónio (retirado de (Wikibooks, 2015))

Os nodos funcionam como os neurónios que enviam sinais aos outros através das suas ligações, sendo que existe também o conceito de peso associado a cada ligação. Este peso é utilizado para estabelecer a importância da ligação entre os nodos. (Faria, 2013).

No início são atribuídos pesos semelhantes a todas as ligações, sendo que durante o treino do modelo os pesos vão sendo alterados consoante as diferenças entre a saída da rede e os dados de treino (Santos, 2001).

As redes neuronais, apesar de serem um dos métodos mais capazes são também um dos que mais capacidade de computação necessita, sendo que a utilização deste método em *data sets* com um elevado número de registos pode tornar-se impraticável.

2.3.2.4 Aproximação de vizinhanças

A aproximação de vizinhanças (*nearest neighbour*) é fundamentada partindo do princípio que registos semelhantes se encontram próximos uns dos outros quando são colocados num plano espacial (Santos, 2001). Na Figura 14 a) pode-se ver que o registo será classificado de forma diferente tendo em conta a distância K utilizada. Na Figura 14 b) observa-se um plano que contém três tipos de registos diferentes. A "estrela" no centro do círculo corresponde a um registo de tipo desconhecido, sendo que neste caso o algoritmo atribuiria o tipo "círculo" a esse registo pois encontra-se perto de outros registos desse tipo.

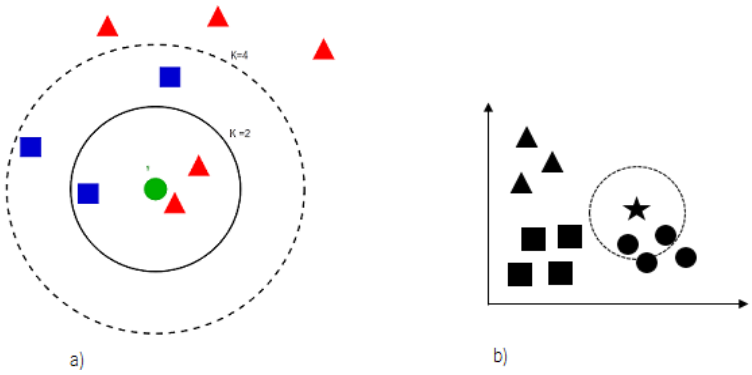


Figura 14 - a) Plano espacial com 3 tipos de registos e duas distâncias K (retirado de (Wikipedia, 2014b); b) Plano espacial com registos de 3 tipos distintos)

As classes podem ser representadas pelo seu centro de gravidade (localização média de todos os seus registos) ou por um registo da sua classe próximo do seu centro (Santos, 2001).

De forma a calcular a posição de um dado registo no plano espacial, são utilizadas medidas de semelhança como a distância Euclidiana ou *Coisine Similarity* (Faria, 2013).

Infelizmente este método também carece de escalabilidade, pois cada registo é comparado com todos os outros, fazendo com que se torne um método lento quando se trata de um número elevado de registos (Santos, 2001).

2.3.2.5 Algoritmos genéticos

Os algoritmos genéticos, tal como as redes neuronais, tentam de certa forma, emular características de seres biológicos. Enquanto as redes neuronais simulam o processo ocorrido nos nossos cérebros através dos neurónios e das sinapses, os algoritmos genéticos tentam simular o processo de evolução genética que ocorre na grande maioria das espécies do nosso planeta.

No que toca ao *data mining*, os algoritmos genéticos são normalmente utilizados como classificadores, ou então como algoritmos de otimização de parâmetros em outros classificadores como por exemplo os algoritmos de aproximação de vizinhança (Verma & Verma, 2012).

2.3.2.6 Support Vector Machines

As Máquinas de Suporte Vetorial (SVM – S*upport Vector Machines)* são uma técnica baseada na teoria de aprendizagem estatística e tem como objetivo a maximização da separação entre as duas classes num híper-plano. Para uma melhor compreensão ver exemplo na Figura 15 a).

Caso não seja possível encontrar uma divisão linear (num plano de duas dimensões), os dados são (normalmente) projetados para um plano multidimensional como se pode observar na Figura 15 b).

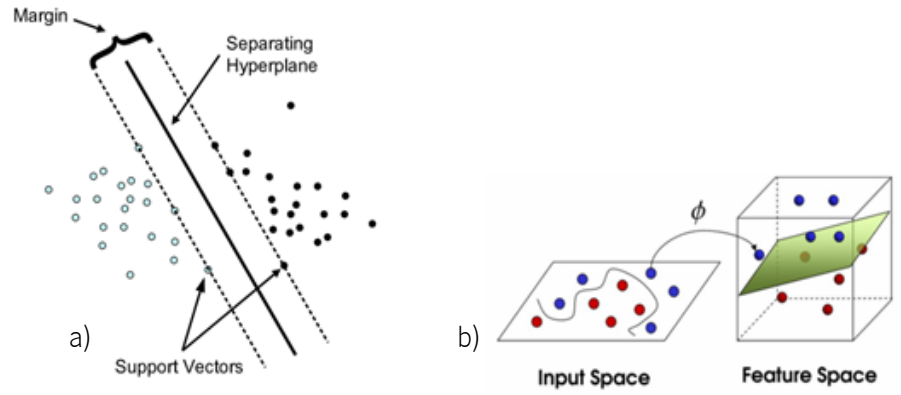


Figura 15 - a) Exemplo de SVM (retirado de (Meyer, 2014)); b) Projeção de dados num plano multidimensional (retirado de (L. F. Teófilo & Reis, 2011a))

Apesar de ser um ótimo classificador, este algoritmo é pouco escalável devido à sua função quadrática de otimização. Uma boa escolha dos parâmetros é também crucial para a obtenção de bons resultados (Meyer, 2014).

2.3.3 Ferramentas Data Mining

No mercado atual existem bastantes ferramentas que nos possibilitam efetuar projetos de *data mining*. Nesta secção vamos verificar quais as ferramentas que cumprem os requisitos de forma serem utilizadas neste projeto.

Para esta análise foi selecionada uma amostra das muitas ferramentas existentes no mercado, sendo que nesta amostra apenas se encontram as ferramentas com as quais já existiu contacto e que podem cumprir com os requisitos definidos, sendo eles: ferramentas que sejam de utilização gratuita, que permitam a integração com aplicações java, que abarquem a maioria dos algoritmos de *data mining* e sejam de fácil utilização. Pode ser analisada uma comparação exaustiva em (Cruz & Cortez, 2009).

Na Tabela 3 pode-se observar uma comparação entre as ferramentas de *data mining* que foram selecionadas. Pode-se verificar que o *Weka* cumpre efetivamente todos os requisitos, sendo que o *Rapid Miner* apenas falha no facto de na sua versão gratuita conter algumas limitações no que toca à utilização de fontes de dados e algoritmos de *data mining*. Por sua vez, o *R* sendo uma ferramenta mais orientada para a estatística, mas muito poderosa, cumpre todos os requisitos, exceptuando o facto de ser uma ferramenta que requer conhecimentos aprofundados da mesma para se efetuar *data mining,* tornando-se assim complicada de utilizar. Apesar de ser um programa bastante completo, a sua utilização é mais complexa do que a utilização do *Weka*.

Tabela 3 - Comparação de ferramentas de *data mining*

Ferramentas/ Características	Gratuita	Contempla a maioria dos algoritmos	Fácil utilização	Sistema operativo Windows	Conexão à base de dados	Arquitetura
Weka	Sim	Sim	Sim	Sim	ASCII	*Stand alone*
Rapid Miner	Sim mas limitada	Sim	Sim	Sim	JDBC	*Stand alone*
R	Sim	Sim	Não	Sim	ASCII	*Stand alone*
SPSS	Não	Não	Sim	Sim	ODBC	*Stand alone*

2.4 *Data Mining* em Diversas Áreas

O *data mining* tem sido utilizado em várias áreas da nossa sociedade, com vários propósitos, como por exemplo na saúde onde se tenta detetar se anomalias nos pulmões são malignas ou benignas(Kusiak, Kernstine, Kern, Mclaughlin, & Tseng, 2000). Na educação à distância, onde os

professores podem fazer melhorias na forma como lecionam as aulas ou nos materiais que apresentam (Romero, Ventura, & De Bra, 2004). Na banca, onde algoritmos ajudam na decisão de aceitar um crédito de um cliente (Koh, Tan, & Goh, 2006), e até na segurança de redes onde a análise de código TCP (*Transmission Control Protocol*) pode detetar uma intrusão na rede (Cai & Li, 2004), podem ser encontrados mais exemplos em (Padhy, Mishra, & Panigrahi, 2012).

Além de todas estas aplicações, o *data mining* é muitas vezes utilizado na área dos videojogos, nomeadamente na telemetria de jogo (*game telemetry*).Telemetria de jogo é a capacidade de recolher dados à distância que serão depois utilizados para auxiliar no desenvolvimento de novos jogos (Drachen, El-Nasr, & Canossa, 2013). A quantidade de informação que pode ser recolhida através deste método é na ordem dos gigabytes, sendo complicado retirar valor da mesma sem o uso das técnicas adequadas (Drachen, Thurau, et al., 2013).

Existem vários exemplos da utilização de *data mining* no contexto de jogos de vídeo como por exemplo (Drachen, Thurau, et al., 2013):

- Previsão de quando deixará de ser jogado (Bauckhage et al., 2012; Mahlmann, Drachen, Togelius, Canossa, & Yannakakis, 2010);
- Previsão do que eles fazem enquanto jogam (Weber & Mateas, 2009);
- Quantidade de tempo despendida a jogar (Williams, Consalvo, Caplan, & Yee, 2009).

Tal como acontece nas outras áreas, o *data mining* por si só não responde a todas as perguntas, nem torna indispensáveis outras técnicas, outros conhecimentos ou a possibilidade de despedir recursos humanos. (Drachen, Thurau, et al., 2013) apresenta seis mitos sobre o *data mining* nos jogos, entre os quais encontramos a possibilidade de não recorrer a *users testing* nem *a designers*; que o *data mining* é capaz de identificar todos os problemas e que é totalmente autómato e não precisa de qualquer intervenção humana. Estes mitos, entre muitos outros, são várias vezes ditos pelos clientes, daí que seja essencial uma clarificação dos objetivos assim como uma boa gestão das expectativas desde o início do projeto.

2.4.1 Problemas e Desafios

Além dos mitos, existem outras questões a ter em conta, como a transparência dos modelos. Existem modelos que por si só são bastante fáceis de compreender como as árvores de decisão, pelo

contrário as redes neuronais tornam-se bastante opacas para quem não tem conhecimentos aprofundados na área.

A qualidade dos dados é também muito importante. Muitas vezes os dados recebidos podem ter vários problemas, como dados duplicados, nulos ou até dados inválidos. Todos estes problemas vão resultar em modelos de qualidade inferior, que não correspondem à verdade, podendo por em risco todo o projeto.

A questão da *performance* é também ela muito importante, pois a maioria dos métodos de *data mning* não se encontram desenhados para *data sets* de grande dimensão, e quando falamos de telemetria de jogos, os dados podem chegar com alguma facilidade aos *terabytes*. Nestes casos, a utilização de amostras de dados que representem fielmente a população torna-se indispensável.

Um outro problema, transversal a muitas áreas, é a segurança. Neste caso falamos principalmente da confidencialidade dos dados, pois estes contêm informação muito relevante e devem ser mantidos longe do acesso de outras pessoas.

Como se pode ver, o *data mining* em jogos sofre muitas vezes dos mesmos problemas e das mesmas questões que o *data mining* usualmente executado nas diversas áreas de negócio.

2.4.2 Data Mining para Modelação de Utilizador

A modelação de utilizadores (*user modeling*) teve o seu princípio com a necessidade de obter uma melhor interação no binómio humano-computador (Fischer, 2001). Quando se fala na interação de uma pessoa com uma máquina é necessário que a mesma responda corretamente na altura exata, para que a interação possa ocorrer da melhor forma. Com o passar dos anos, a utilização da modelação de utilizadores foi-se diversificando, passando a ser utilizada pelos grandes supermercados com os cartões de clientes, pelas lojas *online* através das suas sugestões e *web sites* que visitamos (Fischer, 2001).

A proliferação das redes sociais, assim como o aumento da utilização das mesmas pelas pessoas, levou a que estas disponibilizassem (muitas vezes sem consciência disso) uma elevada quantidade de informação pessoal. Com isso, têm sido feitos esforços para utilizar essa informação disponível para a criação de perfis dos utilizadores (Abel, Celik, Hauff, Hollink, & Houben, 2011). Esses perfis são depois utilizados para uma personalização das plataformas (*web sites,* aplicações *web*) melhorando assim o serviço prestado e a experiência do utilizador.

A análise das redes sociais tem vindo a ser utilizada para testar várias teorias ou para provar conceitos. Exemplo disso foi o *Twitter,* onde foram analisados milhões de *tweets* efetuados pelos seus utilizadores, assim como as notícias que foram sendo divulgadas pelos principais canais noticiosos. O objetivo foi chegar a uma *framework* capaz de efetuar recomendações de artigos cujo utilizador tivesse um interesse elevado (Abel, Gao, Houben, & Tao, 2011).

São vários os métodos de *data mining* que têm sido utilizados para modelação de utilizadores, e cada um capta diferentes elementos e comportamentos do utilizador (Frias-Martinez, Chen, & Liu, 2006). Na Tabela 4 pode-se visualizar várias técnicas utilizadas para a modelação de utilizadores tendo em conta a tarefa a executar e o tipo de dados que se possui.

Tabela 4 - Técnicas de *data mining* utilizadas para modelação de utilizador, adaptada de (Frias-Martinez et al., 2006)

	Labeled Data		*Unlabelled Data*	
Tarefa	Modelo legível	Modelo não legível	Modelo legível	Modelo não legível
Recomendação	Árvores de decisão Regras de classificação	Redes neuronais K-NN	Regras de associação *self-organizing maps*	*K-means clustering* *Fuzzy Clustering*
Classificação	Árvores de decisão Regras de classificação	SVM Redes neuronais K-NN	*self-organizing maps*	*K-means clustering* *Fuzzy Clustering*

Todas estas utilizações têm em comum a utilização de informação pessoal dada pelos utilizadores quando usam os seus serviços, e esse facto levanta sérias questões de privacidade. Toda esta informação armazenada deve ser utilizada para beneficiar o cliente, mas pode ser utilizada de forma a obter informações pessoais.

2.4.3 Data Mining no Poker

No que concerne ao poker, foram encontrados também alguns artigos que fazem uso do *data mining* para a identificação da estratégia dos adversários (L. F. Teófilo & Reis, 2011b) ou para a construção de um agente virtual (L. F. Teófilo & Reis, 2011a),(Beattie & Nicolai, 2007), como se pode ver na Tabela 5.

Tabela 5 - Comparação de outros trabalhos envolvendo poker e *data mining*

	(L. F. Teófilo & Reis, 2011a)	(Beattie & Nicolai, 2007)	(L. F. Teófilo & Reis, 2011b)
Objetivo	Criação de agente	Criação de agente	Modelação de oponente
Algoritmo usado	Árvore de pesquisa	Algoritmos genéticos	*Clusters*

Nestes últimos casos vemos que são diversas as técnicas de *data mining* utilizadas, assim como o objetivo das mesmas. Darse Billings et al.(1998) demonstra a importância da modelação do adversário para atingir melhores resultados, pois a estratégia e a sua flexibilidade assim como o conhecimento da forma de jogar do adversário são muito importantes no que toca à construção de agentes virtuais. Neste caso verifica-se a utilização de *clusters* de forma a agrupar os jogadores em vários tipos conforme as suas ações (L. F. Teófilo & Reis, 2011b).

Para a construção de agentes observa-se a utilização de outros algoritmos como as árvores de pesquisa (L. F. Teófilo & Reis, 2011a) ou então a utilização de algoritmos genéticos de forma a ganhar o máximo de fichas possíveis ao adversário (Beattie & Nicolai, 2007). O objetivo nestas duas abordagens passa pela criação de um agente virtual vencedor, enquanto o objetivo deste trabalho passa pela criação de um agente capaz de igualar a forma de jogar de um determinado jogador, porém as técnicas a serem utilizadas podem ser semelhantes.

2.5 Conclusões

Ao longo desta revisão de literatura foi possível verificar a existência de uma elevada quantidade de abordagens para a construção de agentes virtuais, assim como um elevado número desses mesmos agentes, cada um com os seus aspetos positivos e negativos. A investigação nesta área continua a desenvolver vários métodos, abordagens e agentes que obtêm cada vez melhores resultados, sendo que se prevê um futuro próspero nesta área.

No *data mining* observamos que é uma área bem estabelecida e consolidada, com metodologias de desenvolvimento e métodos bastante maduros. Apesar disso nota-se a existência de alguma falta de atualização nas metodologias existentes, pois não têm acompanhado o desenvolvimento em outras áreas, como por exemplo o *big data.*

3 Desenvolvimento

Na secção de desenvolvimento será demonstrado todo o trabalho prático efetuado durante a dissertação.

A metodologia seguida para efetuar o trabalho prático desta dissertação foi o CRISP-DM, onde foram seguidas as cinco primeiras fases da metodologia, deixando de fora a implementação devido a restrições de tempo.

Foram executados três casos de estudo, sendo que o primeiro (experiência preliminar) foi efetuado numa fase muito primitiva do desenvolvimento da dissertação, servindo para conhecimento das ferramentas, dos dados e para a obtenção de alguns resultados.

Numa fase posterior foram executados dois novos casos de estudo, caso de estudo 1 – *Fold ou Call* e caso de estudo 2 - *Call, Check, Raise, Fold* ou *All-In*, onde foram utilizados 2 modelos de dados diferentes e testados vários algoritmos de *data mining*.

Todo o desenvolvimento prático será efetuado num sistema constituído por um *Intel Core Duo P7450* com 4GB de memória *RAM (Random Access Memory)* e com o *Windows 10* de 64 bits como sistema operativo.

A estrutura desta secção é idêntica às fases do CRISP-DM com a exceção da apresentação dos modelos de dados e dos casos de estudo. A avaliação dos modelos encontra-se no capítulo seguinte sob o nome de "Experiências e Resultados".

3.1 Compreensão do Negócio

Neste trabalho o foco não está um negócio mas sim num jogo, nomeadamente poker na sua versão "*Hold'em No Limit*", por isso neste caso está-se perante um âmbito muito bem limitado e definido através das regras do jogo.

De forma a se compreender melhor o jogo em si, a retirar os objetivos e a perceber que dados seriam necessários para atingir esses mesmo objetivos, será dada uma explicação em síntese do que é um jogo de poker na vertente "*Hold'em No Limit*".

Como se observa pela Figura 16, é possível afirmar que esta vertente do poker se divide em 10 etapas, sendo que 4 delas são de apostas, onde os jogadores tendo em conta as suas cartas e as cartas

que se encontram na mesa (cartas comunitárias) apostam o seu dinheiro ou as suas fichas. Esta vertente ("*No limit*") distingue-se das outras por não haver um limite máximo de unidades monetárias ou fichas a serem apostados, sendo que um jogador é livre de apostar todas as suas fichas ou unidades monetárias logo na primeira ronda, ou em qualquer outra ronda de apostas.

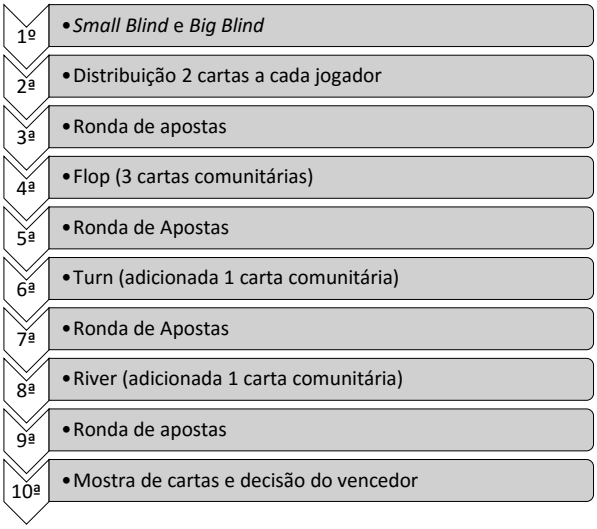


Figura 16 - Ciclo completo de um jogo de *Poker Hold'em*

De realçar que o jogador não tem qualquer interferência na primeira etapa do jogo (*Small blind* e *Big blind*), pois o valor de cada uma destas apostas já se encontra definido, e a cada jogo, a *Big blind* e a *Small blind* desloca-se pela mesa no sentido dos ponteiros do relógio.

O poker é um jogo bastante complexo, isto devido às inúmeras variáveis a ter em conta para se possuir uma estratégia vencedora. Além da grande variedade de mãos de jogo que se pode possuir, é também necessário ter em conta outros fatores, como a parte psicológica dos nossos adversários, a posição na mesa, a quantidade de fichas ou unidades monetárias que se possui e as cartas comunitárias que vão sendo postas na mesa (Skalinsky & Malmuth, 1999).

Tratando-se este trabalho de uma prova de conceito, e devido às limitações de tempo, reduziu-se a complexidade de um jogo completo, selecionando assim apenas as três primeiras etapas, isto é, a fase de *pre-flop*.

Nestas etapas, além da *Big blind* e da *Small blind*, são distribuídas duas cartas a cada jogador presente, sendo que cada jogador só pode visualizar as suas próprias cartas, ficando assim sem o conhecimento das cartas que estão na posse de outros jogadores.

Após a distribuição das cartas pelos jogadores, estes são convidados a efetuar as suas apostas, sendo que possuem quatro possibilidades:

- *Fold* – Desistem do jogo não apostando qualquer valor;
- *Call* – Igualam a maior aposta que se encontra na mesa;
- *Raise* – Apostam um valor superior à maior aposta que se encontra na mesa;
- *Check* – Utilizada quando um jogador efetua a *Big blind* e a maior aposta na mesa não superou esse valor
- *All in* – Quando um jogador aposta todas as suas fichas ou unidades monetárias.

Neste momento o jogador só tem à sua disposição a informação relativa às cartas que possui, desconhecendo por completo que cartas têm os outros jogadores, nem que cartas serão colocadas na mesa como cartas comunitárias. É neste momento que cada jogador começa a colocar em prática a sua estratégia. Apesar de existirem vários livros (Skalinsky & Malmuth, 1999) com linhas orientadoras, onde se inclui, por exemplo, as cartas se encontram divididas por classes que determinam a sua possibilidade de vencer. Cada jogador é livre de aplicar a sua própria estratégia.

3.2 Compreensão de Dados

Como já foi referido, os dados foram obtidos através de jogadores profissionais que disponibilizaram amostras contendo milhares de jogos efetuados, tanto com dinheiro virtual, como com dinheiro real.

Esses dados encontram-se em ficheiros de texto e na sua maioria sem qualquer estrutura visível e onde não se pode aplicar técnicas de *data mining*.

Como se pode observar pela Figura 17, o nosso jogador de nome "DonksAreUs" encontra-se no quarto lugar de uma mesa de seis jogadores (1), foi o jogador que efetuou a *Big blind* (2), as suas cartas foram "Tc – Jc" – "Dez e Valete de paus" – (3), sendo que efetuou "*Call*" (4) na sua jogada. Nos pontos 5, 6, e 7 observa-se a informação relativa às fases posteriores do jogo, *flop, turn,* e *river* respetivamente. Por fim pode-se visualizar informação relativa às cartas dos jogadores e ao vencedor do jogo.

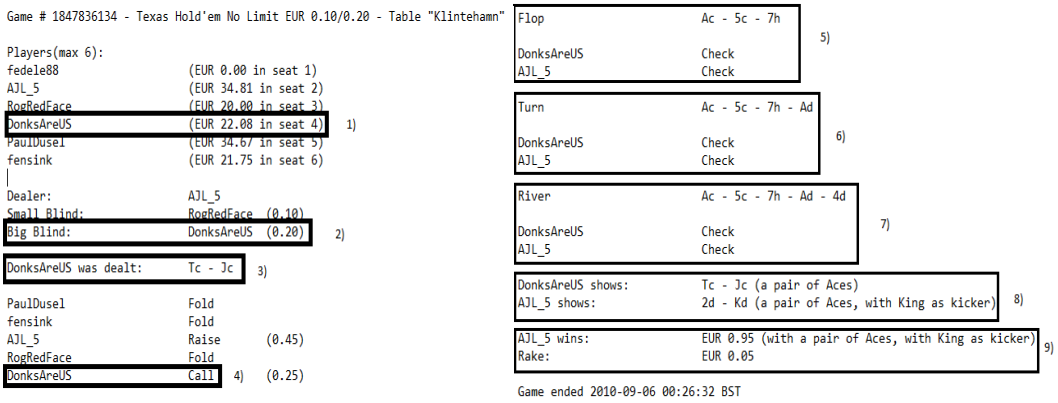


Figura 17 - Amostra de um log de um jogo de poker

Como se verifica nesta amostra, os dados foram obrigatoriamente tratados linha a linha, verificando se a linha contém informação relevante, tornando o processo de codificação e execução bastante complexo e moroso.

Nesta fase, o ideal seria ter um conhecimento mais aprofundado dos dados, isto é, ter uma perceção da quantidade de mãos de jogo, do número de mãos iguais, número de "*Call*", "*Check*", "*Raise*", "*Fold*" e "*All in*" existentes, mas infelizmente da forma como os dados se encontravam impossibilitou a obtenção dessa informação. Assim sendo, só após o processo ETL é que será possível a obtenção da mesma.

3.3 Casos de estudo

De forma a verificar-se a possibilidade de reprodução por parte de um agente virtual das ações tomadas por um jogador real, dividiu-se o problema em duas partes, sendo o primeiro uma experiência preliminar efetuada numa fase inicial da dissertação e posteriormente quatro casos de estudo. A cada caso de estudo corresponde um problema para o qual foram utilizadas técnicas de *data mining* de forma a obter a resposta.

Para cada caso de estudo foram utilizados os mesmos dados do mesmo jogador mas com um modelo de dados adequado para responder ao problema. Os algoritmos de *data mining* experimentados também poderiam sofrer alterações consoante o tipo de problema.

Assim espera-se obter uma perceção dos algoritmos e dos modelos de dados que melhor conseguem representar as decisões tomadas pelos jogadores reais.

3.3.1 Experiência preliminar – Pre-flop

Durante a elaboração inicial da dissertação foi possível desenvolver algum trabalho prático, demonstrando assim a capacidade de atingir os objetivos propostos. Este trabalho prático foi desenvolvido com uma parte dos dados disponíveis, assim como com uma versão limitada do modelo de dados devido a tratar-se de uma abordagem exploratória. Este trabalho deu origem a um artigo científico que foi publicado na conferência CISTI'2015[5] (10ª Conferência Ibérica em Sistemas de Tecnologias de Informação) (Silva & Reis, 2015).

O trabalho foi realizado recorrendo à utilização da metodologia CRISP-DM, sendo que não foi executada a fase de implementação. De seguida será demonstrado o trabalho realizado em cada uma das fases da metodologia.

3.3.2 Caso de estudo 1 – Fold ou Call

Neste caso de estudo espera-se obter resposta a uma pergunta bastante simples:

- O jogador continua ou não em jogo?

Desta forma diminui-se a complexidade do jogo. As decisões que o jogador pode tomar foram reduzidas a duas – *Fold* ou *Call* – assim, apesar de existir apenas duas ações, é possível saber se o jogador se deve manter em jogo ou desistir. Todas as outras possibilidades (*Raise, Check* e *All-In*) foram transformadas em *Call*.

3.3.3 Caso de estudo 2 – Call, Check, Raise, Fold ou All-In

No segundo caso de estudo, o modelo de jogador tentou prever qual das cinco jogadas possíveis seria efetuada pelo jogador. É um caso de estudo mais complexo e mais próximo da realidade de um jogo de poker que o primeiro, pois em vez de duas possibilidades passam a existir cinco o que incrementa a possibilidade de erro.

3.3.4 Caso de estudo 3 e 4 – Diferentes Modelos de Dados

Nos casos de estudo 3 e 4 pretende-se demonstrar a possibilidade de se desenvolver um modelo de dados com um menor número de atributos de forma a que seja possível a utilização de algoritmos de

[5] http://www.aisti.eu/cisti2015

data mining mais complexos. Espera-se verificar também a possibilidade de obtenção de resultados semelhantes com um modelo de dados mais simples.

O atributo em estudo continuará a ser a jogada efetuada pelo jogador, sendo que se mantém as suas cinco possibilidades *(Call, Check, Raise, Fold* e *All-In).*

3.4 Preparação dos dados

A preparação dos dados é uma fase de máxima importância no desenvolvimento de um projeto de *data mining*. É nesta fase que se recolhem e transformam os dados para que seja possível obter resultados através da implementação de várias técnicas de *data mining*.

Esta secção está dividida em dois pontos. O primeiro sobre a experiência preliminar e o segundo ponto sobre os casos de estudo.

3.4.1 Experiência Preliminar

Após a compreensão dos dados e a seleção da informação que queremos retirar, chega a altura de estruturar os dados. Para isso foi escolhida uma ferramenta de ETL bastante completa, o *Talend Data Integration*. O *Talend Data Integration* é uma ferramenta gratuita que nos permite criar um processo onde os dados podem ser lidos de uma enorme variedade de fontes (onde se inclui os ficheiros de texto) e que no decorrer desse processo podem sofrer inúmeras transformações para que no fim sejam escritos numa outra forma, normalmente com uma estrutura homogeneizada. Uma das grandes vantagens do *Talend Data Integration* é não limitar apenas à ferranmenta em si, pois possibilita escrever código java na maior parte dos componentes que estão à disposição.

Na Figura 18 pode-se observar o processo ETL efetuado sobre os dados utilizados. Toda a parte de seleção dos dados foi efetuada no componente tJavaRow_1, onde código java é executado a cada linha de cada ficheiro de texto. No fim os dados são escritos para um ficheiro ARFF – tipo de ficheiro suportado pela ferramenta de *data mining Weka*.

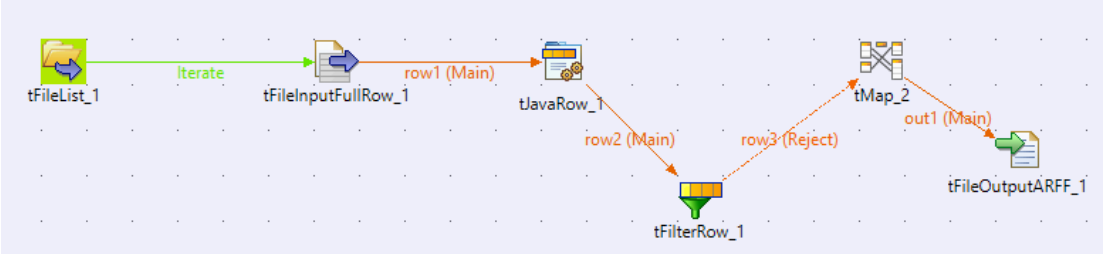


Figura 18 - Processo ETL

Na Figura 19 pode-se ver uma amostra do resultado final após o processo ETL. Pode-se verificar que os dados se encontram perfeitamente estruturados, assim como os campos selecionados. Como dito anteriormente, poderia ser selecionada outra informação, mas com a informação selecionada poderá ser criado um bom modelo de previsão para o campo "Jogada". O campo "Nome" é obviamente igual para todas as entradas, o que o torna desnecessário nesta fase, mas mesmo assim foi decidido incluir de modo a saber que estas entradas pertencem a um determinado jogador.

Row No.	Nome	Mao	BigBlind	SmallBlind	Jogada
1	DonksAreUS	Tc-Jc	true	false	Call
2	DonksAreUS	Ks-Qh	false	true	Raise
3	DonksAreUS	9d-3c	false	false	Fold
4	DonksAreUS	7s-Qh	false	false	Fold
5	DonksAreUS	Ks-9d	true	false	Fold
6	DonksAreUS	4c-Ts	false	true	Fold
7	DonksAreUS	Kd-4h	false	false	Fold
8	DonksAreUS	4d-Qs	false	false	Fold
9	DonksAreUS	Qs-8h	false	true	Fold
10	DonksAreUS	Tc-8d	true	false	Check

Figura 19 - Amostra de dados estruturados após processo ETL

No ficheiro ARFF obteve-se um total de 85441 entradas, e como se pode ver pela Figura 20 existe uma elevada predominância de "*Fold*'s" o que não é estranho, pois a quantidade de mãos de jogo consideradas fortes (com maiores possibilidades de vitória) são menores do que as outras (Skalinsky & Malmuth, 1999).

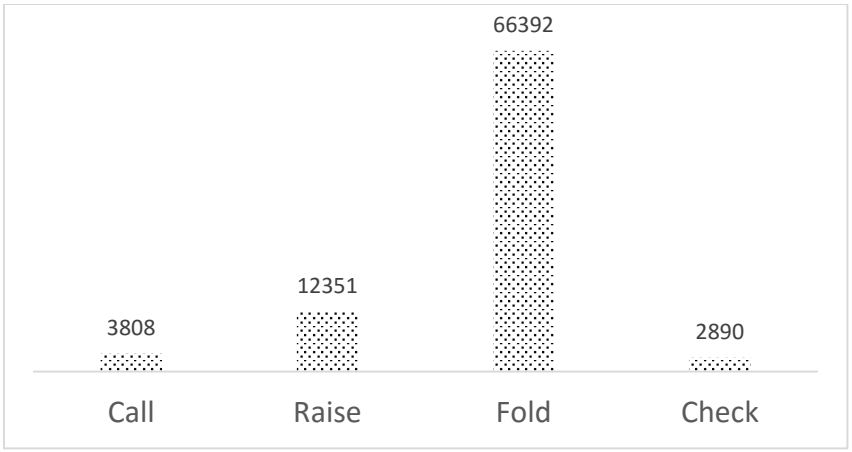


Figura 20 - Distribuição do campo Jogada

Após os dados estarem estruturados, foi possível passar para a fase de modelação, onde foram aplicadas técnicas de *data mining* como regras de associação, árvores de decisão e redes neuronais.

3.4.2 Casos de estudo 1, 2, 3 e 4

Os dados utilizados para a realização desta dissertação de mestrado foram cedidos por um jogador profissional de poker e retratam vários jogos reais e estando divididos em vários ficheiros de texto, sendo que cada ficheiro contém informação relativa a vários jogos.

Para o desenvolvimento desta dissertação foram utilizados 2172 ficheiros, que após o tratamento de dados representaram 225796 registos, sendo que cada linha representa a informação recolhida de um jogo.

De forma a tratar e a estruturar os dados, foi selecionada a ferramenta *Talend Data Integration*. O *Talend Data Integration* é uma ferramenta que permite fazer a extração e tratamento de dados de várias fontes, permitindo depois a colocação dos dados tratados em diversos formatos (base de dados, ficheiros CSV – *Comma-separated values*, entre outros).

Devido à grande quantidade de dados que foi tratada, o processo de ETL foi dividido em duas partes. Durante a primeira parte foi extraída e tratada toda a informação contida nos ficheiros de texto, sendo depois transformada em dados estruturados. A informação foi armazenada num ficheiro CSV, tal como pode ver-se na Figura 21.

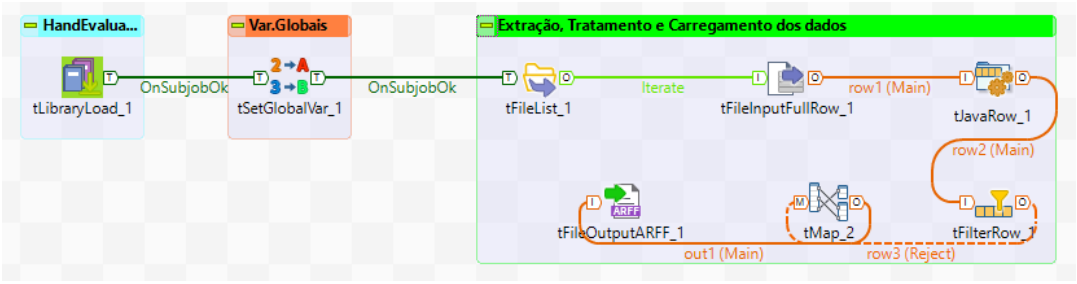


Figura 21 – Primeira parte do processo de ETL

Nesta primeira fase, o grande esforço no tratamento de dados foi concentrado no componente *"tJavaRow_1"*. Neste componente podem ser encontradas cerca de 1300 linhas de código java para tratamento de todas as linhas referentes a um jogo. Outro componente muito importante nesta fase foi o *"tLibraryLoad_1"* que permitiu a utilização da biblioteca externa *HandEvaluator* desenvolvida por *Aaron Davidson, Darse Billings* e *Denis Papp* e possibilitou o cálculo dos *ranks* das mãos dos jogadores, assim como atribuir um nome a essas mesmas mãos.

Na segunda etapa do processo ETL, o ficheiro CSV criado anteriormente foi carregado e dividido em quatro ficheiros ARFF – *Attribute-Relation File Format,* para depois serem utilizados na ferramenta de *data mining – Weka*. Cada ficheiro é referente a uma fase do poker e contém apenas os atributos e dados que o jogador tem acesso para decidir que ação efetuar em cada fase (Figura 22).

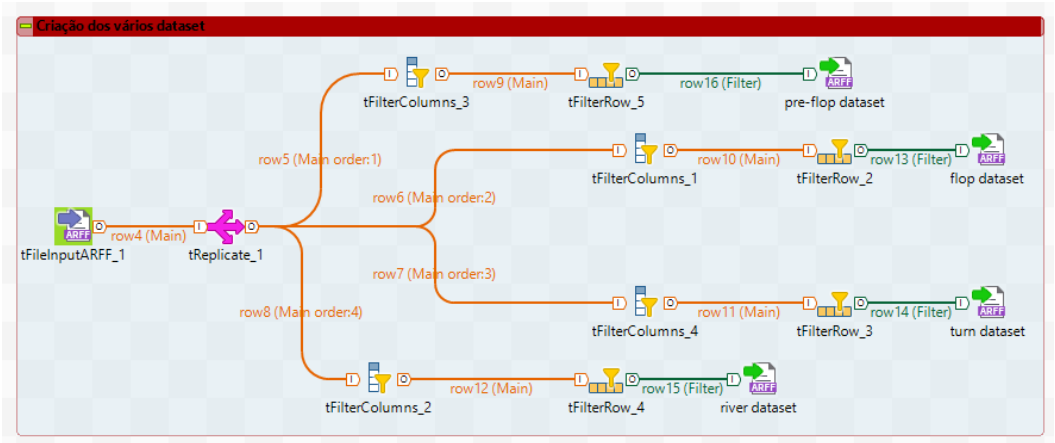


Figura 22 – Segunda parte do processo de ETL

Após o processo de ETL estar concluído e analisados os dados, foi possível verificar que existe um elevado número de jogos que terminam na fase de *pre-flop* (apenas cerca de 15% dos jogos passam esta fase), sendo que é nesta fase que o jogador tem menos informação disponível (apenas 16 atributos), tal como seria de esperar. Também de forma natural, o número de jogos disponíveis vai diminuindo ao

longo das fases do jogo, assim como a informação disponível vai aumentando, tal como é possível verificar na Figura 23.

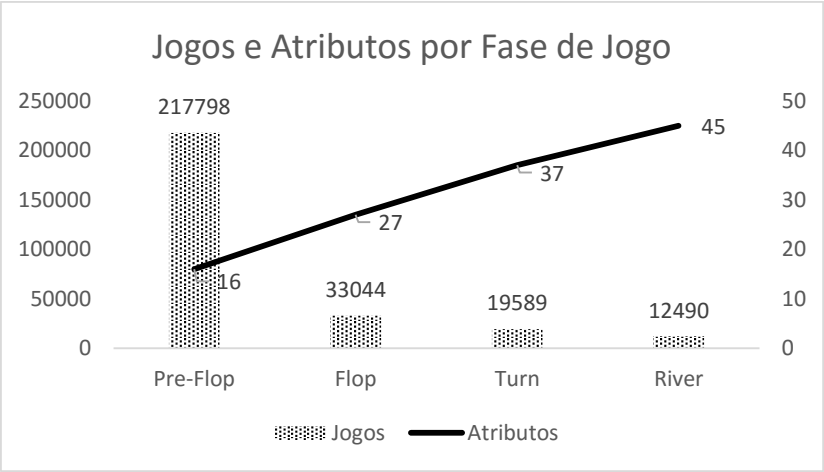


Figura 23 – Número de jogos e atributos por fase de jogo

Devido à diminuição do número de jogos disponíveis nas fases mais adiantadas e apesar da existência de mais atributos, pode-se prever que, provavelmente, os resultados obtidos através das técnicas de *data mining* estejam sujeitos a um maior número de erros.

Olhando para a dispersão do valor do atributo "Jogada", observar-se que, com a exceção da primeira fase de jogo, o mesmo é consistente no que refere à ordem dos valores tendo em conta a sua quantidade de ocorrências. Na primeira etapa do jogo, verifica-se uma elevada predominância do valor *"Fold"* com mais de 169000 ocorrências, seguido do valor *"Raise"* com mais de 28000 ocorrências. Neste caso o valor *"Fold"* detém um elevado número de ocorrências comparado com os outros quatro valores, o que requer uma especial atenção na análise dos resultados obtidos. Na segunda fase de jogo, verifica-se que o valor com maior número de ocorrências é o *"Check",* com cerca de 11000 registos, seguido de perto do *"Call"* com perto de 9500 registos e do *"Fold"* com mais de 8600 ocorrências. De salientar apenas o baixo número de ocorrências do valor *"Raise"* com menos de 300 (Figura 24).

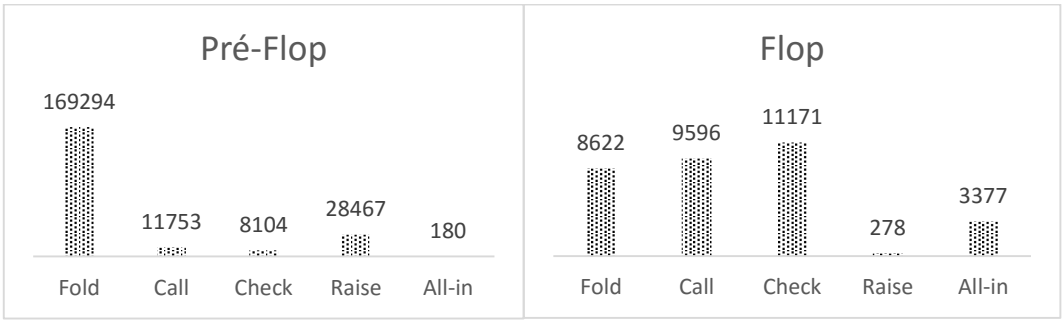


Figura 24 – Dispersão do valor do atributo "Jogada" na fase de jogo *Pre-flop* e *Flop*

Nas duas últimas fases verifica-se uma predominância do valor *"Check"* com cerca de 8100 ocorrências no *Turn* e quase 6000 no *River.* Os valores *"Call"* e *"Fold"* distam praticamente o mesmo valor nas duas fases como se vê na Figura 25.

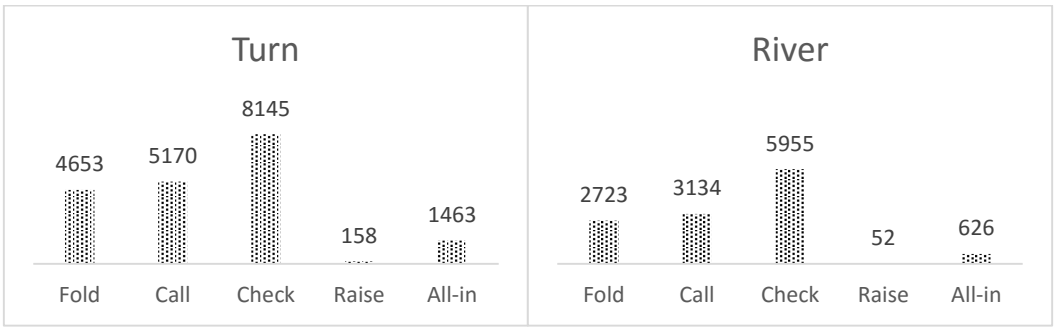


Figura 25 – Dispersão do valor do atributo "Jogada" na fase de jogo *Turn* e *River*

Após esta análise à distribuição dos dados, observa-se uma elevada escassez de dados sobre o valor *"Raise"* nas três últimas fases de jogo. Esta característica dos dados irá, muito provavelmente, provocar uma elevada dificuldade na previsão deste valor, o mesmo poderá acontecer ao valor *"All-in"*. Por sua vez o valor *"Fold"* é extremamente elevado na primeira parte do jogo, o que poderá provocar uma predominância do mesmo na previsão do modelo e uma curva ROC elevada.

3.5 Modelo de Dados

Uma das partes essenciais para obtenção de bons resultados é a criação de um modelo de dados que seja capaz de abarcar toda a informação relevante de um jogo de poker.

A informação contida no modelo de dados deverá estar num estado de alto nível, de forma a que seja possível aplicar técnicas de *data mining* e obter um modelo de jogador.

Devido à especificidade do poker é possível observar que a informação vai variando consoante a fase em que nos encontramos (número de jogadores, valor do pote, valor de aposta), desta forma foi utilizado um modelo para cada fase de apostas.

3.5.1 Experiência Preliminar

Para a experiência preliminar foi utilizado um modelo de dados bastante simples de forma a agilizar o processo de obtenção de resultados.

Desta forma o modelo de dados é constituído pelos seguintes campos:

- **Nome do jogador:** *string* que apenas contém informação relativa ao nome do jogador ao qual pertence o jogo.
- **Mão:** *string* que contém toda a informação referente à mão detida pelo jogador;
- ***Big Blind:*** *string* que contém o valor *true* ou *false* caso o jogador em questão tenha efetuado a *Big Blind;*
- ***Small Blind:*** *string* que contém o valor *true* ou *false* caso o jogador em questão tenha efetuado a *Small Blind;*

3.5.2 Caso de Estudo 1 e 2

Nos seguintes pontos serão demonstrados os modelos utilizados nos casos de estudo 1 e 2, sendo que para cada fase do jogo foi utilizado um modelo o modelo de jogo correspondente.

3.5.2.1 Modelo de dados Pre-Flop

Para a fase de *pre-flop* será utilizado o seguinte modelo de dados:

- **Mão abstração:** *string* com o conjunto de duas cartas detidas pelo jogador. O naipe das cartas foi ocultado de forma a obter menos possibilidades (KQs; KQo), onde a letra "s" representa cartas do mesmo naipe e a letra "o" representa cartas de naipe distinto.
- ***Big Blind:*** *string* que contém o valor *true* ou *false* caso o jogador em questão tenha efetuado a *Big Blind;*
- ***Small Blind:*** *string* que contém o valor *true* ou *false* caso o jogador em questão tenha efetuado a *Small Blind;*
- ***Stack:*** *string* que contém o valor monetário ou de fichas que o jogador possui no momento;
- ***Stack/Big Blind:*** *string* que contém o resultado da divisão da *stack* do jogador pelo valor da *Big Blind;*
- **Classe Mão:** *string* com o valor de 1 a 9 que divide a mão de jogador em classes segundo (Darse Billings et al., 2002);
- **Número de jogadores:** *string* que contém um dos seguintes valores – "Elevado";"Medio";"Reduzido" – tendo em conta o número de jogadores em jogo;

- **Posição:** *string* que contém a posição na mesa em que o jogador se encontra. A posição do jogador será dividida em três possibilidades – "Início"; "Meio"; "Fim" – tendo em consideração a número de jogadores entre o jogador e o *dealer*;
- **Mão:** *string* que contém toda a informação referente à mão detida pelo jogador;
- ***Dealer:*** *string* contendo *true* ou *false* caso o jogador em questão seja o *dealer;*
- ***Pot:*** string que demonstra o valor do *pot*;
- ***Harrington Group:*** *string* com o valor da mão detida pelo jogador segundo os grupos de *Harrington* (L. Teófilo et al., 2014);
- **M:** *string* com o valor da divisão da *stack* do jogador pela soma do valor da *big blind, small blind* e valor do *pot;*
- **Nome da mão:** *string* com o nome da mão, por exemplo: "*Two High"; "a Pair of Jacks"; "a Pair of Aces";*
- **Rank da mão:** *string* que contém o valor do rank da mão que o jogador possui [6]*;*
- ***Fold*** **ou** ***Call:*** *string* com o valor "*Call"* caso o jogador tenha continuado em jogo ou com o valor *"Fold"* caso o jogador tenha desistido. Este atributo foi apenas utilizado no caso de estudo 1;
- **Jogada:** *string* com a decisão efetuada pelo jogador. Este atributo apenas foi apenas utilizado no caso de estudo 2, 3 e 4.

3.5.2.2 Modelo de dados Flop

Na fase do *flop,* modelo de dados utilizado foi idêntico ao anterior, acrescentando os seguintes atributos:

- **Valor da aposta:** *string* com o valor da aposta efetuado pelo jogador para continuar em jogo;
- **Aposta/*Big Blind*:** *string* com o resultado da divisão do valor da aposta efetuado pelo jogador sobre o valor da *big blind;*
- **Aposta/*Pot:*** *string* com o resultado da divisão do valor da aposta efetuado pelo jogador sobre o valor do *pot;*
- **Mão do jogador *Flop:*** *string* com a mão inicial do jogador, mais as três cartas comunitárias da fase de *flop;*

[6] Segundo a libraria HandEvaluator de Aaron Davidson, Darse Billings e Denis Papp

- ***Best 5 Cards Flop:*** *string* com a melhor mão possível incluindo as cartas comunitárias da fase de *flop.* As cartas encontram-se ordenadas tendo em conta o seu valor em conjunto, isto é, sequências, pares, trios;
- **Rank da mão *Flop:*** *string* contendo o valor do rank da mão que o jogador possui em conjunto com as cartas comunitárias da fase de *flop* [7];
- **Nome da mão *Flop:*** *string* com o nome da melhor mão possível incluindo a mão inicial e as cartas comunitárias da fase de *flop,* por exemplo: "*Two High"; "a Pair of Jacks"; "a Pair of Aces";*
- **Valor do *Pot Flop:*** *string* com o valor do *pot* na fase de *flop;*
- **Número de jogadores *Flop***: *string* que contém um dos seguintes valores – "Elevado";"Medio";"Reduzido" – tendo em conta o número de jogadores em jogo na fase de *flop*;
- **Posição:** *string* que contém a posição na mesa em que o jogador se encontra. A posição do jogador será dividida em três possibilidades – "Início"; "Meio"; "Fim" – tendo em consideração a número de jogadores entre o jogador e o *dealer* na fase de *flop;*
- ***Fold*** **ou** ***Call Flop:*** *string* com o valor "*Call"* caso o jogador tenha continuado em jogo ou com o valor *"Fold"* caso o jogador tenha desistido nesta fase de jogo. Este atributo apenas foi utilizado no caso de estudo 1;
- **Jogada *Flop:*** *string* com a decisão efetuada pelo jogador nesta fase. Este atributo apenas foi utilizado no caso de estudo 2, 3 e 4.

De salientar também a remoção do campo ***Fold*** **ou** ***Call*** referente à fase de *pre-flop* devido a esse atributo apenas conter o valor *"Call"*, pois nesta fase, apenas são analisados os jogos no qual o jogador transita da fase anterior.

3.5.2.3 Modelo de dados Turn

Tal como na transição entre as duas fases anteriores, na fase de *Turn* foram adicionados os seguintes atributos:

- **Aposta *flop:*** *string* que contém o valor da aposta efetuada na fase de *flop;*

[7] Segundo a libraria HandEvaluator de Aaron Davidson, Darse Billings e Denis Papp

- Valor total do *pot flop:* *string* que contém o valor do *pot* na fase de *flop* adicionando o valor da aposta efetuado pelo jogador;
- Cartas *Turn:* *string* com todas as cartas em jogo, isto é, as cartas pertencentes à mão inicial do jogador, mais todas as cartas comunitárias em jogo;
- *Best 5 Cards Turn:* *string* com a melhor mão possível (5 cartas de seis possíveis) incluindo as cartas comunitárias da fase *turn.* As cartas encontram-se ordenadas tendo em conta o seu valor em conjunto, isto é, sequências, pares, trios;
- Rank da mão *Turn:* *string* que contém o valor do rank da mão que o jogador possui em conjunto com as cartas comunitárias da fase de *flop* e *turn* [8];
- Nome da mão *Turn:* *string* com o nome da melhor mão possível incluindo a mão inicial e as cartas comunitárias da fase de *flop* e *turn,* por exemplo: "*Two High"; "a Pair of Jacks"; "Three of a Kind, Queens";*
- Número de jogadores: *string* que contém um dos seguintes valores – "Elevado";"Medio";"Reduzido" – tendo em conta o número de jogadores em jogo na fase de *turn;*
- Posição: *string* que contém a posição na mesa em que o jogador se encontra. A posição do jogador será dividida em três possibilidades – "Início"; "Meio"; "Fim" – tendo em consideração a número de jogadores entre o jogador e o *dealer* na fase de *turn;*
- *Fold* ou *Call Turn:* *string* com o valor "*Call"* caso o jogador tenha continuado em jogo ou com o valor *"Fold"* caso o jogador tenha desistido nesta fase de jogo. Este atributo apenas foi apenas utilizado no caso de estudo 1;
- Jogada *Turn:* *string* com a decisão efetuada pelo jogador nesta fase. Este atributo apenas foi utilizado no caso de estudo 2, 3 e 4.

Para esta fase de jogo, foi também retirado do modelo de dados, o atributo *Fold* ou *Call* devido a apenas conter o valor *"Call"*, tal como na fase anterior.

3.5.2.4 Modelo de dados River

Por fim, para esta última fase, as alterações ao modelo de dados foram idênticas às alterações efetuadas na fase de *Turn,* apenas alterando a informação contida, pois esta refere-se à fase de *River*.

[8] Segundo a libraria HandEvaluator de Aaron Davidson, Darse Billings e Denis Papp

- Aposta *Turn:* *string* que contém o valor da aposta efetuada na fase de *turn;*
- Jogada *Turn:* *string* que contém a jogada efetuada pelo jogador na fase de *turn;*
- Valor do Pot *Turn:* *string* que contém o valor total que o *pot* atingiu na fase de *turn;*
- Nome da Mão *River:* *string* com o nome da melhor mão possível incluindo a mão inicial e as cartas comunitárias da fase de *flop* e *turn,* por exemplo: "*Two High"; "a Pair of Jacks"; "Three of a Kind, Queens";*
- *Best 5 Cards River:* *string* com a melhor mão possível (5 cartas de seis possíveis) incluindo as cartas comunitárias da fase *river.* As cartas encontram-se ordenadas tendo em conta o seu valor em conjunto, isto é, sequências, pares, trios;
- Rank da mão *River:* *string* que contém o valor do *rank* da mão que o jogador possui em conjunto com as cartas comunitárias da fase de *flop, turn* e *river* [9];
- Cartas *River:* *string* com todas as cartas em jogo, isto é, as cartas pertencentes à mão inicial do jogador, e também todas as cartas comunitárias em jogo;
- Número de jogadores *River:* *string* que contém um dos seguintes valores – "Elevado";"Medio";"Reduzido" – tendo em conta o número de jogadores em jogo na fase de *river;*
- Posição *River:* *string* que contém a posição na mesa em que o jogador se encontra. A posição do jogador será dividida em três possibilidades – "Início"; "Meio"; "Fim" – tendo em consideração a número de jogadores entre o jogador e o *dealer* na fase de *river;*
- *Fold* ou *Call River:* *string* com o valor "*Call"* caso o jogador tenha continuado em jogo ou com o valor *"Fold"* caso o jogador tenha desistido nesta fase de jogo. Este atributo apenas foi utilizado no caso de estudo 1;
- Jogada *River:* *string* com a decisão efetuada pelo jogador nesta fase. Este atributo apenas foi utilizado no caso de estudo 2, 3 e 4.

Com estes quatro modelos de dados, espera-se abarcar a maioria da informação necessária para implementar a estratégia utilizada pelo jogador original.

[9] Segundo a libraria HandEvaluator de Aaron Davidson, Darse Billings e Denis Papp

3.5.3 Caso de estudo 3 - Seleção automática de atributos

Utilizando a opção de seleção automática de atributos presentes na ferramenta *Weka,* é possível reduzir o número de atributos do modelo de dados, utilizando apenas os mais importantes para a previsão do atributo que pretendemos.

Tendo em conta a quantidade de "*atribute evaluator"* presentes na ferramenta assim como métodos de pesquisa, foi necessário proceder a uma seleção. Assim sendo, foram escolhidos para análise o "*CfsSubsetEval"* com os seguintes métodos de pesquisa: *"BestFirst"* e *"GeneticSearch"*. Foi também escolhido o *"ChiSquaredAttributeEval"* e o *GainRatioAttributeEval"* ambos com o método de pesquisa *"Ranker"*.

Depois de utilizados estes algoritmos foi possível chegar aos seguintes resultados (quanto maior o resultado menos importante é o atributo):

- *Pre-flop*

Tendo em conta os resultados obtidos e verificando que os dois primeiros algoritmos apenas apresentam os três atributos mais importantes, escolhemos os seis com melhores resultados: *Big Blind*, Classe Mão, *Harrington Group*, Mão, Mão abstração, *Pot* (Tabela 6).

Tabela 6 – Seleção automática de atributos para a fase de *Pre-flop*

Atributos / Algoritmos	Big Blind	Classe Mão	Harrington Group	Mão	Mão abstração	Pot	Stack / Big Blind	Posição
CfsSubsetEval BestFirst	1	2	3	-	-	-	-	-
CfsSubsetEval GeneticSearch	1	2	3	-	-	-	-	-
ChiSquaredAttributeEval Ranker	6	3	4	1	2	5	7	9
GainRatioAttributeEval Ranker	5	2	1	4	3	6	10	7
Total	**13**	**9**	**11**	**5**	**5**	**11**	**17**	**16**

- *Flop*

Devido à impossibilidade de obtenção de resultados com os dois primeiros algoritmos utilizados para a fase de *pre-flop*, foi necessário proceder à escolha de novos algoritmos de forma a substituírem os anteriores.

Verificando os resultados, é possível observar que os seis atributos com melhor *rank* são: Mão jogador Flop, *Best 5 Cards Flop, Rank* Mão *Flop, Stack, Pot* Total *Flop* e *Stack / Big Blind Mao* (Tabela 7).

Tabela 7 – Seleção automática de atributos para a fase de *Flop*

Atributos / Algoritmos	Mão jogador Flop	Best 5 Cards Flop	Rank Mão Flop	Stack	Pot Total Flop	Stack / Big Blind Mao	Mão	Nome da Mão	Jogada	Aposta Pot
FilteredAttributeEval Ranker	1	2	3	5	4	7	6	8	10	13
ChiSquaredAttributeEval Ranker	1	2	3	4	6	7	5	8	10	14
GainRatioAttributeEval Ranker	1	2	5	7	3	16	8	9	4	6
Total	**3**	**6**	**11**	**16**	**13**	**30**	**19**	**25**	**24**	**33**

- *Turn*

Desta forma os atributos com melhor *rank* foram: Cartas *Turn*, Cartas *Flop*, *Best 5 Cards Flop*, *Best 5 Cards Turn*, *Rank* Mão *Flop*, *Rank* Mão *Turn* (Tabela 8).

Tabela 8 – Seleção automática de atributos para a fase de *Turn*

Atributos / Algoritmos	Cartas Flop	Cartas Turn	Best 5 Cards Flop	Best 5 Cards Turn	Rank Mão Flop	Rank Mão Turn	Stack	Stack /Big Blind	Mao	Nome Mão Turn	Jogada Flop
FilteredAttributeEval Ranker	1	2	3	4	5	6	7	8	9	10	15
ChiSquaredAttributeEval Ranker	1	2	3	4	5	6	7	8	9	10	15
GainRatioAttributeEval Ranker	1	2	3	4	5	6	7	8	13	10	9
Total	**3**	**6**	**9**	**12**	**15**	**18**	**21**	**24**	**31**	**30**	**39**

- *River*

Para a última parte do jogo, os atributos com melhor *rank* são: Cartas *Flop*, Cartas *Turn*, Cartas *River*, Best 5 Cards *Flop*, Best 5 Cards *Turn*, Best 5 Cards *River* (Tabela 9).

Tabela 9 – Seleção automática de atributos para a fase de *River*

Atributos / Algoritmos	Cartas Flop	Cartas Turn	Cartas River	Best 5 Cards Flop	Best 5 Cards Turn	Best 5 Cards River	Rank Mão Flop	Rank Mão Turn	Rank Mão River	Stack
FilteredAttributeEval Ranker	1	2	3	4	5	6	7	8	9	10
ChiSquaredAttributeEval Ranker	1	2	3	4	5	6	7	8	9	10
GainRatioAttributeEval Ranker	1	2	3	4	5	6	7	8	9	10
Total	**3**	**6**	**9**	**12**	**15**	**18**	**21**	**24**	**27**	**30**

3.5.4 Caso de estudo 4 - Seleção manual de atributos

Para o caso de estudo 4 foi efetuada uma seleção manual dos atributos tendo em conta a sua importância para um jogo de poker e a qualidade dos seus dados.

Assim sendo, os atributos escolhidos para cada fase de um jogo de poker foram:

- ***Pre-flop:*** Mão Abstração; *Big Blind; Small Blind;* Número Jogadores; Posição; Mão; Dealer; *Pot; Harrington Group;* M; Nome da Mão; *Rank* da Mão; Jogada;
- ***Flop:*** *Big Blind; Small Blind;* Número Jogadores; Posição; Mão; Dealer; *Pot;* M; Nome da Mão; *Rank* da Mão; Jogada; *Rank* da Mão *Flop;* Nome da Mão *Flop; Pot* Total *Flop;* Posição do Jogador *Flop;* Número de Jogadores *Flop;* Jogada *Flop;*
- ***Turn:*** *Big Blind; Small Blind;* Número Jogadores; Posição; Mão; Dealer; *Pot;* M; Jogada; Nome da Mão *Flop; Pot* Total *Flop;* Posição do Jogador *Flop;* Número de Jogadores *Flop;* Jogada *Flop;* Nome da Mão *Turn;* Posição Jogador *Turn;* Número de Jogadores *Turn;* Jogada *Turn;*
- ***River:*** *Big Blind; Small Blind;* Número Jogadores; Posição; Mão; Dealer; *Pot;* M; Jogada; Nome da Mão *Flop; Pot* Total *Flop;* Posição do Jogador *Flop;* Número de Jogadores *Flop;* Jogada *Flop;* Nome da Mão *Turn;* Posição Jogador *Turn;* Número de

Jogadores *Turn;* Jogada *Turn;* Nome da Mão *River;* Posição Jogador *River;* Número de Jogadores *River;* Jogada *River;*

3.6 Modelação

Devido à existência de uma experiência preliminar, foram efetuadas duas modelações com características distintas. Assim sendo, o primeiro ponto desta secção dedica-se à modelação efetuada durante a experiência preliminar e o segundo ponto à modelação efetuada para os dois casos de uso.

3.6.1 Modelação – Experiência preliminar

Tendo em conta que se pretende prever qual a jogada que será efetuada pelo jogador na fase de *pre-flop*, está assim evidente que um problema de classificação. Para a resolução desse tipo de problemas, existem vários tipos de algoritmos, desde os mais simples, como regras de classificação, aos mais complexos como as redes neuronais.

Para este trabalho, tomou-se a decisão de se experimentar pelo menos um algoritmo de cada tipo, sendo que não se foram efetuadas alterações nas suas especificações definidas por defeito. Além disso a ferramenta utilizada foi o *Weka*, pelo facto de ser gratuita, de fácil utilização e com todos os algoritmos necessários para o estudo.

Todos os modelos foram testados com uma divisão de 66% do conjunto de dados para aprendizagem e 34% para teste, sendo que foi também efetuado *cross validation* para despistar qualquer problema com o *data set* que pudesse influenciar os resultados.

Além de *cross validation* foi também utilizado o algoritmo *ZeroR* para ter um ponto inicial de comparação, sendo que este é um algoritmo com bastantes deficiências pois apenas prevê o mais comum para todos os registos.

Para avaliar os vários algoritmos foram utilizadas as várias estatísticas proporcionadas pelo *Weka*, como a curva ROC, matriz de confusão, instâncias classificadas correta e incorretamente.

3.6.2 Modelação – Caso de estudo 1, 2, 3 e 4

Tal como na experiência preliminar, a ferramenta utilizada foi o *Weka* pela facilidade de utilização e pelo facto de ser uma ferramenta de *data mining* bastante completa.

Foram testados vários algoritmos, sendo que devido a limitações de *hardware* não foi possível a utilização de alguns deles como por exemplo as redes neuronais. Foi utilizado *cross validation* em todos os algoritmos testados e avaliados manualmente através da percentagem de instâncias corretamente classificadas, pela curva ROC, *Kappa Statistic* e pelo *Root mean squared error*. Por fim os algoritmos serão avaliados automaticamente pela opção *"Experimenter"* pressente no *Weka*.

3.7 Conclusões

No decorrer do terceiro capítulo foi possível observar o trabalho realizado nas primeiras fases do CRISP-DM. Foi desenvolvida uma melhor perceção do jogo, de forma a que fosse possível elaborar um modelo de dados capaz de absorver a informação necessária, para se realizar posteriormente as experiências para a obtenção de resultados. Foi ainda explicitada uma experiência preliminar e quatro casos de estudo a desenvolver.

Nesta etapa foram encontrados vários problemas, nomeadamente no que toca a capacidade de processamento do *hardware* utilizado. Devido à elevada quantidade de dados a processar, a preparação de dados revelou-se extremamente complexa e morosa em termos de tempo.

4 Experiências e Resultados

A avaliação dos modelos de jogadores será efetuada em duas etapas distintas. A primeira parte consiste na análise dos dados recolhidos após a execução dos algoritmos como a matriz de confusão, a curva ROC, a percentagem e número de instâncias previstas corretamente. Esta análise será realizada individualmente para cada fase do jogo *(pre-flop, flop, turn* e *river)* e para cada caso de estudo.

Numa segunda etapa, e usando a função *"Experimenter"* da ferramenta *Weka*, serão confrontados os modelos de dados e a capacidade de previsão dos algoritmos utilizando uma significância de 5%.

Apesar de terem sido selecionados e testados os mesmos algoritmos de *data mining* para todos os casos de teste (com a exceção da experiência preliminar) e para todas as fases de jogo, não foi possível a obtenção de resultados de todos eles em todas experiências efetuadas. Desta forma existem alguns algoritmos que não constam nas tabelas de resultados.

Com estas duas análises distintas espera-se conseguir compreender qual será o melhor algoritmo para prever qual a jogada a efetuar pelo nosso jogador.

4.1 Experiência preliminar

De forma a facilitar a comparação entre os vários algoritmos utilizados, vamos proceder à sua avaliação utilizando uma tabela comparativa, com as estatísticas retiradas da ferramenta *Weka*.

Tal como se pode verificar na Tabela 10, existem resultados muito semelhantes. Com exceção do Zero R (que é obviamente o mais fraco, tal como era esperado), todos os algoritmos se aproximam de uma área ROC na ordem dos 0.9, destacando-se o Nayve Bayes pelos seus 0.934. Também o Nayve Bayes se destaca na percentagem de instâncias classificadas incorretamente, atingindo apenas as 13.149%, assim como na precisão média, que atinge um valor de 0.838.

Tabela 10 - resultados obtidos através da execução dos algoritmos

	ROC Area	%classificados incorretamente	Precisão média
Zero R	0.5	22.176	0.606
IBK	0.915	14.303	0.837
AODE	0.932	13.181	0.836
Decision Table	0.881	13.862	0.813
J48	0.882	13.587	0.837
Naive Bayes	**0.934**	**13.149**	**0.838**
Random Forest	0.904	14.578	0.837

É possível observar na Tabela 11, as matrizes de confusão dos algoritmos utilizados mostram onde efetivamente os algoritmos falham nas suas previsões. Por exemplo, verifica-se que o algoritmo *Decision Table* acaba por falhar na classe que têm menos registos *(Call* e *Check)*. Pelo contrário o *IBK* e o *Random Forest* obtêm melhores resultados nessas mesmas classes. Por sua vez, na classe *Raise*, destaca-se o algoritmo *J48* tendo-se verificado um grande equilíbrio nos restantes algoritmos. No que toca à classe com o maior número de registos *(Fold)*, o algoritmo com melhor performance, é o *Naive Bayes*, havendo ainda uma diferença significativa para alguns algoritmos como o *Random Forest* que obtém o pior resultado.

Tendo em conta as matrizes de confusão e os resultados anteriormente mencionados, é possível afirmar que o Nayve Bayes obtém o resultado mais equilibrado.

Analisando a Figura 26 é possível verificar mais detalhadamente os resultados do *Nayve Bayes*. Na matriz de confusão verifica-se que as jogadas *Call* estão a ser na sua maioria classificadas como *Raise* ou como *Fold*. As jogadas *Check* também estão a ser confundidas com *Fold*, aliás 91% dos *Check*'s são classificados como *Fold*. Estes dois erros podem estar ligados à especificidade das jogadas, pois são jogadas que estão diretamente ligadas às apostas que se efetuam no jogo, e sobre as quais não existe informação neste conjunto de dados. Além disso a enorme quantidade de *Fold*'s pode estar a influenciar o modelo.

Tabela 11 - matriz de confusão dos vários algoritmos testados

Algoritmos	Classificados como ——>	*Call*	*Raise*	*Fold*	*Check*
Zero R	*Call*	0	0	1299	0
	Raise	0	0	4189	0
	Fold	0	0	22608	0
	Check	0	0	954	0
IBK	*Call*	**411**	570	282	36
	Raise	274	282	1020	3
	Fold	213	609	21513	273
	Check	72	7	796	79
AODE	*Call*	219	638	390	52
	Raise	90	2909	1185	5
	Fold	43	414	22066	85
	Check	35	5	887	27
Decision Table	*Call*	94	885	320	0
	Raise	179	2876	1134	0
	Fold	23	532	22053	0
	Check	5	67	882	0
J48	*Call*	263	701	310	25
	Raise	139	**2966**	1081	3
	Fold	89	527	21822	170
	Check	40	28	834	52
Naive Bayes	*Call*	187	671	349	92
	Raise	68	284	1212	15
	Fold	39	375	**22106**	88
	Check	34	8	869	43
Random Forest	*Call*	328	598	313	60
	Raise	209	296	1066	8
	Fold	142	569	21458	439
	Check	50	12	769	**123**

```
=== Confusion Matrix ===

     a     b     c     d   <-- classified as
   187   671   349    92 |     a = Call
    68  2894  1212    15 |     b = Raise
    39   375 22106    88 |     c = Fold
    34     8   869    43 |     d = Check
```

Figura 26 - Matriz de confusão do algoritmo Nayve Bayes

De forma a evitar qualquer tipo de erro com a avaliação através da divisão do conjunto de dados, o Nayve Bayes foi também executado em *cross validation*.

Na Figura 27 pode-se verificar que tanto a curva ROC como a percentagem de instâncias classificadas incorretamente sofreram algumas melhorias, porém apesar disso é possível verificar que os erros na matriz de confusão continuam a acontecer. As melhorias observadas são normais, já que são usadas mais instâncias para teste.

Classifier output

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

Correctly Classified Instances       74303               86.9641 %
Incorrectly Classified Instances     11138               13.0359 %
Kappa statistic                          0.5968
Mean absolute error                      0.1093
Root mean squared error                  0.2209
Relative absolute error                 58.7162 %
Root relative squared error             72.4112 %
Total Number of Instances            85441

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.163     0.006      0.545     0.163     0.25       0.927    Call
                 0.695     0.042      0.738     0.695     0.716      0.941    Raise
                 0.979     0.376      0.901     0.979     0.938      0.938    Fold
                 0.037     0.005      0.206     0.037     0.063      0.929    Check
Weighted Avg.    0.87      0.298      0.838     0.87      0.846      0.938

=== Confusion Matrix ===

     a     b     c     d   <-- classified as
   619  1970  1036   183 |     a = Call
   236  8580  3494    41 |     b = Raise
   140  1064 64996   192 |     c = Fold
   140    15  2627   108 |     d = Check
```

Figura 27 - Nayve Bayes com *cross validation*

4.2 Caso de estudo 1 - *Fold* ou *Call*

Neste caso de estudo foram analisadas as quatro fases de poker tentando prever em cada fase que jogada é que seria efetuada pelo jogador, sendo que apenas existirão duas possibilidades; A possibilidade de continuar em jogo – *Call*, ou a possibilidade de desistir do jogo – *Fold.*

4.2.1 Pre-Flop

Como se pode observar pela Tabela 12 o algoritmo *Decision Table* obteve os melhores resultados nas três estatísticas estudadas destacando-se dos demais com alguma significância. Os algoritmos que mais se aproximaram dos melhores resultados foram o *Naive Bayes* e o *RBF Network*. Por sua vez, o *Hyper Pipes* foi claramente o algoritmo que piores resultados obteve.

Tabela 12 – resultados obtidos no caso de estudo 1 *pre-flop*

	Corretamente Classificadas	*Kappa Statistic*	*ROC Area*
Naive Bayes	89.20%	0.6861	0.917
Decision Table	**91.28%**	**0.7468**	**0.959**
Hyper Pipes	23.27%	-0.0023	0.52
Ada Boost M1	88.63%	0.6338	0.929
Attribute Selected Classifier	88.47%	0.6202	0.869
RBF Network	90.11%	0.71	0.924

Analisando as matrizes de confusão dos algoritmos presentes na Tabela 13 e analisando os resultados mais mais detalhadamente, verifica-se que apesar do *Decision Table* ter dominado as estatísticas anteriores, o *Ada Boost M1* obtém melhores resultados que os restantes no que diz respeito aos verdadeiros positivos para o tipo de jogada *Fold.* Estes resultados foram obtidos devido a uma clara tendência do algoritmo classificar as instâncias como *Fold,* sendo o que mais instâncias classificou desta forma. Observando os resultados dos *Call,* verifica-se que os melhores resultados são obtidos pelo *Decision Table.* Por sua vez o *Naive Bayes* e o *RBF Network* obtiveram resultados semelhantes.

Tabela 13 – matriz de confusão no caso de estudo 1 *pre-flop*

Algoritmos	Classificados como –>	***Call***	***Fold***
Naive Bayes	*Call*	36342	12162
	Fold	11371	157923
Decision Table	*Call*	**38672**	9832
	Fold	9167	160127
Ada Boost M1	*Call*	29134	19370
	Fold	5404	**163890**
RBF Network	*Call*	36739	11765
	Fold	9786	159508

Na análise estatística presente na Figura 28 e Figura 29 observa-se que estatisticamente o algoritmo *RBF Network* é inferior ao *Decision Table* para uma significância de 5%.

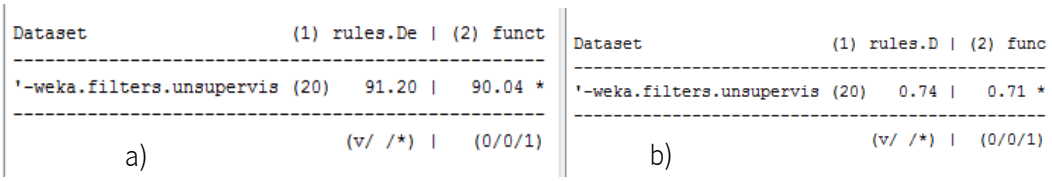


Figura 28 – Caso de estudo 1 fase de *pre-flop* a) Instâncias corretamente classificadas; b) *Kappa Statistic ;*

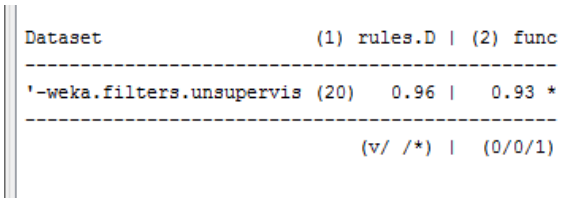


Figura 29 – Caso de estudo 1 fase de *pre-flop ROC Area*

4.2.2 Flop

No caso do *Flop* verifica-se um decréscimo nos resultados de cerca de 10% em relação aos resultados obtidos na fase anterior pelos melhores algoritmos. Nesta fase do jogo, o *Decision Table* não obtém os melhores resultados nas três estatísticas estudadas, mas sim em duas (instâncias corretamente classificadas e *ROC Area)*, sendo o *Naive Bayes* o algoritmo que lidera no *Kappa Statistic,* representando assim uma correlação entre o atributo jogada e os restantes atributos do modelo de dados. Verifica-se também que os resultados obtidos pelos três melhores algoritmos foram muito aproximados.

Tabela 14 – resultados obtidos no caso de estudo 1 *flop*

	Corretamente Classificadas	*Kappa Statistic*	*ROC Area*
Naive Bayes	80.79%	**0.5144**	0.864
Decision Table	**81.75%**	0.4864	**0.865**
Hyper Pipes	72.95%	0.0232	0.586
IBK	74.66%	0.2601	0.688
Ada Boost M1	73.91%	0	0.817
RBF Network	80.92%	0.5055	0.848

Através das matrizes de confusão dos três melhores algoritmos presentes na Tabela 15, verifica-se que os algoritmos com melhores resultados em cada tipo de jogada possuem uma tendência para classificar um maior número de instâncias desse mesmo tipo de jogada. *Decision Table* possui uma tendência para classificar as instâncias como *Call* e o *Naive Bayes* como *Fold*. Por sua vez o *RBF Network* demonstra ser o algoritmo mais equilibrado.

Tabela 15 – matriz de confusão no caso de estudo 1 *flop*

Algoritmos	Classificados como ——>	*Call*	*Fold*
Naive Bayes	*Call*	20904	3518
	Fold	2830	**5792**
Decision Table	*Call*	**22455**	1967
	Fold	4064	4558
RBF Network	*Call*	21260	3162
	Fold	3144	5478

Na Figura 31 não existe nenhum algoritmo que se superiorize em relação aos restantes. No que concerne à *ROC Area*, apenas é possível ter a certeza que o *RBF Network* é inferior aos restantes. O mesmo acontece na Figura 30 b) onde o *Decision Table* é inferior aos 2 outros algoritmos analisados.

Na Figura 30 a) verifica-se uma superioridade estatística do *Decision Table* para uma significância de 5%.

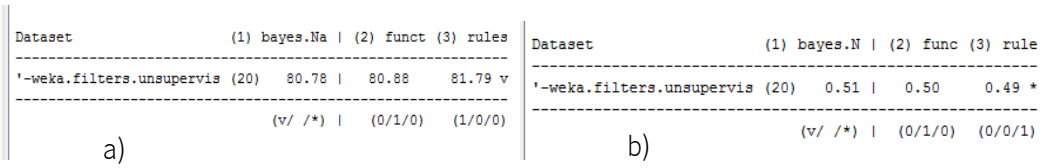


Figura 30 – Caso de estudo 1 fase de *flop* a) Instâncias corretamente classificadas; b) *Kappa Statistic ;*

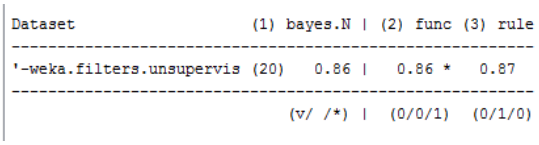


Figura 31 – Caso de estudo 1 fase de *flop ROC Area*

4.2.3 Turn

Pelos resultados apresentados na Tabela 16 nota-se um acentuado decréscimo no que toca ao *Kappa Statistic.* Apesar de existirem algoritmos com uma elevada taxa de instâncias corretamente classificadas, o seu *Kappa Statistic* é praticamente zero. Tal facto demonstra a inexistência de correlação entre os atributos do modelo de dados com o atributo a ser estudado. Apesar disso, pode-se verificar que o *Naive Bayes* apresenta uma *ROC Area* com alguma relevância e em conjunto com o *RBF Network* são os algoritmos mais equilibrados.

Tabela 16 – resultados obtidos no caso de estudo 1 *turn*

	Corretamente Classificadas	*Kappa Statistic*	*ROC Area*
Naive Bayes	59.23%	**0.1524**	**0.643**
Decision Table	76.19%	0.0002	0.584
Hyper Pipes	74.20%	0.0328	0.559
Ada Boost M1	**76.25%**	0	0.628
RBF Network	64.46%	0.1465	0.619

Na Tabela 17 é possível verificar que efetivamente o algoritmo *Ada Boost M1* apenas classifica todas as instâncias como *Call*, ao contrario do *Naive Bayes* e do *RBF Network* que conseguem classificar as instâncias para ambos os tipos de jogada.

Tabela 17 – matriz de confusão no caso de estudo 1 *turn*

Algoritmos	Classificados como ——>	***Call***	***Fold***
Naive Bayes	*Call*	8725	6211
	Fold	1776	**2877**
Ada Boost M1	*Call*	**14936**	0
	Fold	4653	0
RBF Network	*Call*	10446	4490
	Fold	2471	2182

Os resultados do estudo estatístico apresentado na Figura 40 e na Figura 41 demonstram que o *Ada Boost M1* obter melhores resultados no número de instâncias corretamente classificadas. O mesmo é inferior aos outros algoritmos nas restantes estatísticas.

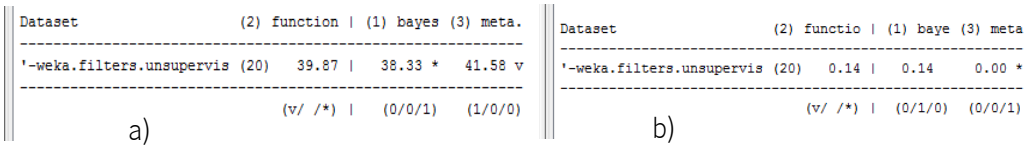


Figura 32 – Caso de estudo 1 fase de *turn* a) Instâncias corretamente classificadas; b) *Kappa Statistic ;*

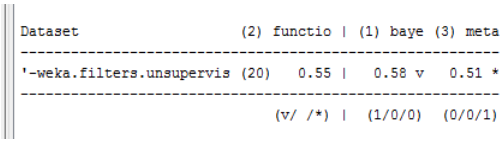


Figura 33 – Caso de estudo 1 fase de *turn ROC Area*

4.2.4 River

No *River* os resultados obtidos pelos algoritmos voltam a decrescer. É de salientar que o valor do *Kappa Statistic* passa a ser praticamente zero em todos os algoritmos. Apesar desse valor, o *Naive Bayes* e o *RBF Network* voltam a destacar-se dos demais, obtendo os melhores resultados nessa estatística. Por sua vez, o *Ada Boost M1*, volta a ser o algoritmo com melhor percentagem de instâncias corretamente classificadas, mas com o pior valor de *Kappa Statistic* (Tabela 22).

Tabela 18 – resultados obtidos no caso de estudo 1 *river*

	Corretamente Classificadas	*Kappa Statistic*	*ROC Area*
Naive Bayes	53.51%	**0.0952**	**0.604**
Decision Table	78.13%	0.0047	0.545
Hyper Pipes	75.39%	0.0063	0.549
Ada Boost M1	**78.20%**	0	0.59
RBF Network	62.82%	0.0862	0.575

Tal como na fase anterior, o *Ada Boost M1* classifica todas as instâncias como *Call.* Sendo o *Naive Bayes* e o *RBF Network* os algoritmos que conseguem classificar instâncias de ambos os tipos (Tabela 23).

Tabela 19 – matriz de confusão no caso de estudo 1 *river*

Algoritmos	Classificados como ——>	*Call*	*Fold*
Naive Bayes	*Call*	4949	4818
	Fold	989	**1734**
Ada Boost M1	*Call*	**9767**	0
	Fold	2723	0
RBF Network	*Call*	6715	3052
	Fold	1592	1131

Na análise estatística presente na Figura 34 a) e na Figura 35 pode verificar-se que o *Ada Boost M1* e que o *Naive Bayes,* respetivamente, são superiores para uma significância de 5% em relação aos outros algoritmos analisados.

Na Figura 34 b) não é possível retirar qualquer conclusão sobre a superioridade de algum algoritmo, apenas pode-se concluir que o *Ada Boost M1* é inferior aos restantes algoritmos.

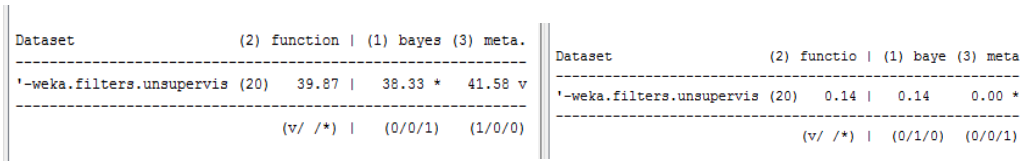


Figura 34 – Caso de estudo 1 fase de *turn* a) Instâncias corretamente classificadas; b) *Kappa Statistic ;*

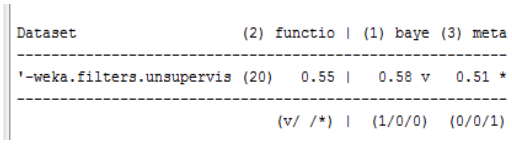


Figura 35 – Caso de estudo 1 fase de *turn ROC Area*

4.3 Caso de Estudo 2 – Modelo de Dados Completo

No caso de estudo 2 – Modelo de dados completo, é estudada a possibilidade de prever que tipo de jogada irá ser efetuada pelo jogador analisado, sendo que este pode efetuar uma de cinco tipos de jogada: *Call; Raise; Fold; Check* e *All-In.*

4.3.1 Pre-Flop

Todos os algoritmos, com exceção do *Hyper Pipes* e do *Ada Boost M1*, apresentam resultados acima dos 85% de instâncias corretamente classificadas, assim como uma *ROC Area* superior a 0.9. O *J48* obteve a melhor classificação de instâncias corretamente classificadas com 89.5% assim como o melhor *Kappa Statistic* com 0.704. Por sua vez na *ROC Area*, o algoritmo com melhor resultado foi o *Naive Bayes* com 0.953 (Tabela 20). Os restantes algoritmos testados apresentaram resultados bastante semelhantes entre eles, sendo que o *IBK* apresenta os segundos melhores resultados nos dois primeiros critérios.

Tabela 20 – resultados obtidos no caso de estudo 2 *pre-flop*

	Corretamente Classificadas	*Kappa Statistic*	*ROC Area*
Naive Bayes	86.08%	0.635	**0.953**
J48	**89.50%**	**0.704**	0.949
Decision Table	87.80%	0.6715	0.943
Hyper Pipes	34.60%	0.0736	0.697
IBK	88.07%	0.6827	0.938
Ada Boost M1	83.90%	0.4691	0.878
Attribute Selected Classifier	87.69%	0.6321	0.938
RBF Network	87.15%	0.6532	0.932

Em relação à matriz de confusão foram analisados os quatro melhores algoritmos tendo em conta a análise anterior. É possível verificar que o *Naive Bayes* consegue os melhores resultados quando a jogada efetuada é *Call* ou *All-In*, sendo que por sua vez o *J48* consegue melhores resultados quando a jogada é *Fold* ou *Check*. Quando a jogada efetuada é *Raise,* o algoritmo com melhores resultados é o *IBK.*

Tabela 21 – matriz de confusão no caso de estudo 2 *pre-flop*

Algoritmos	Classificados como ------>	*Call*	*Raise*	*Fold*	*Check*	*All-In*
Naive Bayes	*Call*	**6637**	3162	1398	351	205
	Raise	4386	19040	4770	29	242
	Fold	4761	4533	157688	2275	37
	Check	504	105	3525	3969	1
	All-In	20	7	2	0	**151**
J48	*Call*	4100	3393	3998	240	22
	Raise	1720	20674	6022	49	2
	Fold	990	2850	**163812**	1625	17
	Check	257	43	1534	**6270**	0
	All-In	63	26	15	0	76
Decision Table	*Call*	5504	3037	2906	302	4
	Raise	2325	20201	5865	73	3
	Fold	5416	2577	159666	1634	1
	Check	319	29	1901	5855	0
	All-In	38	8	131	0	3
IBK	*Call*	5832	3304	2293	308	16
	Raise	2904	**21103**	4387	70	3
	Fold	2910	4900	159617	1864	3
	Check	412	43	2401	5248	0
	All-In	40	111	8	0	21

Em termos estatísticos, foram analisados os algoritmos *Naive Bayes* e *J48,* sendo que o primeiro é, com uma significância de 5%, superior ao algoritmo *J48* em termos de *ROC Area* (Figura 37). O *J48* é superior, para a mesma significância no número de instâncias corretamente classificadas e em termos de *Kappa Statistic* (Figura 36).

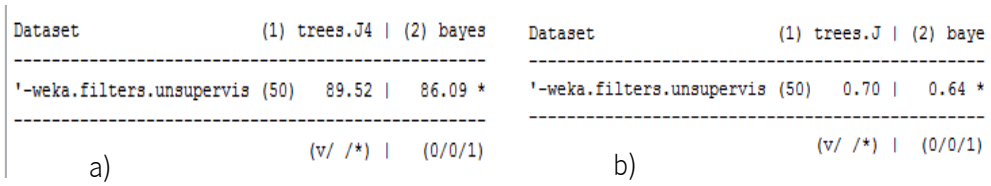


Figura 36 – *Caso de estudo 2 fase de* pre-flop a) Instâncias corretamente classificadas; b) *Kappa Statistic ;*

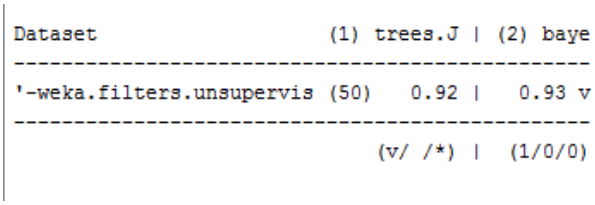


Figura 37 – *Caso de estudo 2 fase de* pre-flop *ROC Area*

4.3.2 Flop

Olhando para os resultados da Tabela 22, verifica-se que o algoritmo com melhor comportamento é o *Decision Table* em todos os aspetos, seguido pelo *RBF Network* a alguma distância.

Verifica-se também a existência de alguns algoritmos a classificar menos de 50% das instâncias corretamente, algo que não se verificava na fase anterior. Pensa-se que tal facto seja motivado pelo número bastante inferior de jogos que foram utilizados nesta fase. Apesar disso a *ROC Area* do algoritmo *Decision Table* possui um valor muito significativo.

Tabela 22 – resultados obtidos no caso de estudo 2 *flop*

	Corretamente Classificadas	*Kappa Statistic*	*ROC Area*
Naive Bayes	45.28%	0.2532	0.717
Decision Table	**64.76%**	**0.4948**	**0.859**
Hyper Pipes	35.92%	0.0565	0.563
IBK	50.03%	0.2969	0.697
Ada Boost M1	40.69%	0.154	0.597
RBF Network	52.40%	0.3407	0.733

Na matriz de confusão, Tabela 23, pode-se verificar que o algoritmo *Decision Table* comporta-se a um nível assinalável quando a jogada se trata de um *Check* ou *Call.* Isto porque obtém um resultado bastante pobre quando a mesma é um *All-in,* sendo que neste caso é o algoritmo *Naive Bayes* o que obtém o melhor resultado. Vendo os resultados obtidos, e quando se trata de um *Raise,* verifica-se que nenhum algoritmo conseguiu bons resultados, sendo o melhor o *Naive Bayes.* Esta carência de bons resultados deve-se muito provavelmente às poucas instâncias existentes em que a jogada é deste tipo. No que toca à jogada *Fold* é o *RBF Network* que se destaca dos demais.

Tabela 23 – matriz de confusão no caso de estudo 2 *flop*

Algoritmos	Classificados como ------>	*Call*	*Raise*	*Fold*	*Check*	*All-In*
Naive Bayes	*Call*	4508	84	1570	1302	2132
	Raise	133	4	57	30	54
	Fold	2218	29	3539	2218	618
	Check	2309	14	2165	5656	1030
	All-In	1396	9	308	407	1257
Decision Table	*Call*	6285	9	2246	947	109
	Raise	160	0	98	19	1
	Fold	3312	2	3826	1456	26
	Check	0	0	0	11171	0
	All-In	2552	0	587	118	120
RBF Network	*Call*	4463	48	2435	1245	1405
	Raise	148	0	85	12	33
	Fold	1798	16	4961	1444	403
	Check	1521	7	2275	7046	322
	All-In	1515	5	602	409	846

Tal como seria de esperar, a Figura 38 e a Figura 39 demonstram a superioridade estatística do algoritmo *Decision Table* em comparação com o *Nayve Bayes* em todos os atributos estatísticos utilizados.

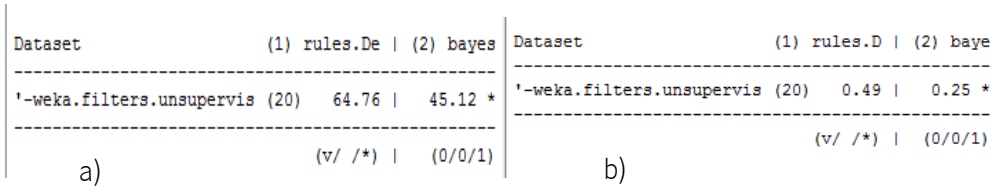


Figura 38 – *Caso de estudo 2 fase de* flop a) Instâncias corretamente classificadas; b) *Kappa Statistic ;*

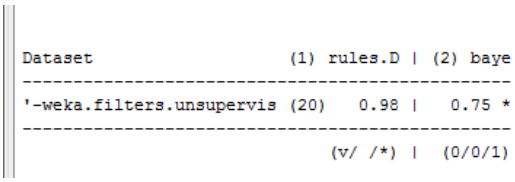


Figura 39 – *Caso de estudo 2 fase de* pré *ROC Area*

4.3.3 Turn

Mais uma vez existem dois algoritmos que se destacam dos demais. O *Naive Bayes* que obtém a melhor pontuação no que se refere ao *Kappa Statistic* e o *Decision Table* que obtém melhores resultados nas instâncias corretamente classificadas e na *ROC Area,* como se pode observar na Tabela 24.

De salientar o resultado do *Ada Boost M1* que apresenta 41.58% de instâncias corretamente classificadas. No entando apresenta um valor de *Kappa Statistic* igual a zero, demonstrando assim que apesar do bom resultado no primeiro ponto, não existe qualquer relação entre os atributos e a classe (jogada).

Tabela 24 – resultados obtidos no caso de estudo 2 *turn*

	Corretamente Classificadas	*Kappa Statistic*	*ROC Area*
Naive Bayes	38.53%	**0.1426**	0.628
Decision Table	**44.11%**	0.1038	**0.638**
Hyper Pipes	39.67%	0.0401	0.551
IBK	38.42%	0.0829	0.56
Ada Boost M1	41.58%	0	0.526
RBF Network	39.58%	0.1335	0.577

Observando as matrizes de confusão na Tabela 25, verifica-se que apesar do *Ada Boost M1* possuir o melhor resultado quando a jogada efetuada é *Call*, este consegue-o devido a um elevado número de falsos positivos, o mesmo acontece com a jogada do tipo *Check*.

Por outro lado, é notório que o *Naive Bayes* obtém os melhores resultados em quase todos os outros tipos de jogada *(Raise, Fold* e *All-In)*, com a exceção do *Check* onde é o *Decision Table* que obtém o melhor resultado, mas também à custa de um elevado número de falsos positivos.

Tabela 25 – matriz de confusão no caso de estudo 2 *turn*

Algoritmos	Classificados como ------>	*Call*	*Raise*	*Fold*	*Check*	*All-In*
Naive Bayes	*Call*	2186	89	968	1200	727
	Raise	82	**10**	18	28	20
	Fold	918	19	**1993**	1524	199
	Check	2095	52	2506	2987	505
	All-In	521	18	242	311	**371**
Decision Table	*Call*	1802	1	147	3174	46
	Raise	103	0	2	51	2
	Fold	357	1	193	4101	1
	Check	1241	1	263	**6625**	15
	All-In	646	2	28	767	20
RBF Network	*Call*	1977	38	976	1654	525
	Raise	85	2	22	39	10
	Fold	742	2	1854	1936	119
	Check	1638	20	2454	3646	387
	All-In	490	7	240	451	275
Ada Boost M1	*Call*	**9086**	0	0	510	0
	Raise	277	0	0	1	0
	Fold	7604	0	0	1018	0
	Check	6810	0	0	4361	0
	All-In	3313	0	0	64	0

Verificando a qualidade dos modelos na vertente estatística, pode-se atentar que nenhum dos dois modelos analisados (*Naive Bayes* e *Decision Table)* se destaca do demais, sendo que o *Naive Bayes* é melhor que o *Decision Table* no que concerne ao *Kappa Statistic* e por sua vez pode-se considerar o *Decision Table* melhor que o *Naive Bayes* quando se fala de instâncias corretamente classificadas. No caso da *ROC Area,* nenhum dos algoritmos pode ser considerado melhor do que o análogo para uma significância de 5% (Figura 40 e Figura 41).

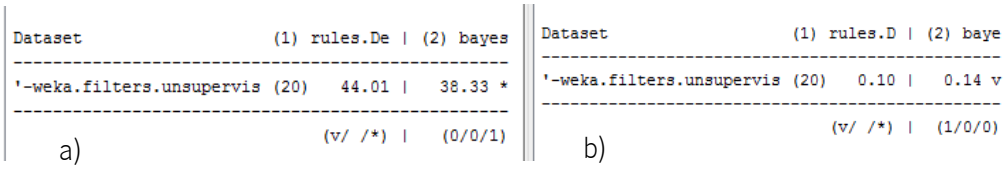


Figura 40 – *Caso de estudo 2 fase* de *turn* a) Instâncias corretamente classificadas; b) *Kappa Statistic ;*

```
Dataset                   (1) rules.D | (2) baye
-------------------------------------------------
'-weka.filters.unsupervis (20)   0.59 |   0.58
-------------------------------------------------
                                (v/ /*) |  (0/1/0)
```

Figura 41 – *Caso de estudo 2 fase* de *turn ROC Area*

4.3.4 River

Por fim, na última etapa de jogo, é possível analisar pela Tabela 26 que mais uma vez é o Naive Bayes e o Decision Table que obtêm os melhores resultados. Ao contrário do que sucedeu na fase anterior, nesta fase foi possível atingir valores superiores a 50% no que diz respeito a instâncias corretamente classificadas. Acontecendo também um pequeno incremento do valor de *Kappa Statistic*, apesar deste continuar ainda num patamar muito baixo.

Tabela 26 – resultados obtidos no caso de estudo 2 *river*

	Corretamente Classificadas	*Kappa Statistic*	*ROC Area*
Naive Bayes	42.85%	**0.1748**	0.656
Decision Table	**52.11%**	0.1591	**0.661**
Hyper Pipes	45.13%	0.0575	0.557
IBK	43.11%	0.0941	0.596
Ada Boost M1	48.63%	0.1466	0.573
RBF Network	44.58%	0.1611	0.598

Através das matrizes de confusão presentes na Tabela 27, depreende-se que o *Naive Bayes* consegue o maior número de verdadeiros positivos em quase todos os tipos de jogada, com a exceção do *Check* que é conseguido pelo *Decision Table*, verificando-se também um elevado número de falsos positivos.

Tabela 27 – matriz de confusão no caso de estudo 2 *river*

Algoritmos	Classificados como ------>	*Call*	*Raise*	*Fold*	*Check*	*All-In*
Naive Bayes	*Call*	1540	16	523	658	397
	Raise	29	0	10	8	5
	Fold	568	8	1001	1019	127
	Check	1331	11	1687	2675	251
	All-In	303	1	84	102	106
Decision Table	*Call*	1218	1	18	1876	21
	Raise	38	0	0	13	1
	Fold	208	0	16	2499	0
	Check	640	0	33	5273	9
	All-In	315	0	3	307	1
RBF Network	*Call*	1353	9	548	1001	223
	Raise	27	0	10	13	2
	Fold	431	5	882	1339	66
	Check	1048	5	1498	3261	143
	All-In	276	0	88	190	72

Na Figura 42 – a) verificamos que o *Decision Table* é (com uma significância de 5%) melhor do que o *Naive Bayes* quando se trata da quantidade de instâncias corretamente classificadas. Por sua vez na Figura 42 – b) e na Figura 43 não é possível afirmar que algum dos algoritmos seja superior (para uma significância de 5%).

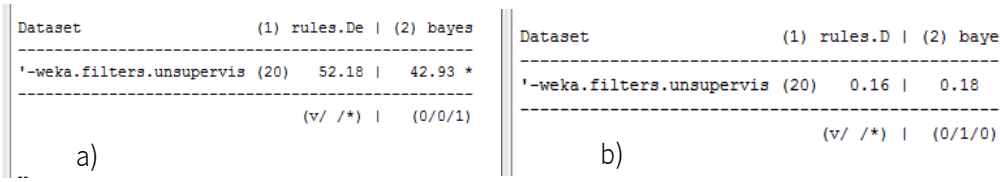


Figura 42 – *Caso de estudo 2 fase* de *river* a) Instâncias corretamente classificadas; b) *Kappa Statistic ;*

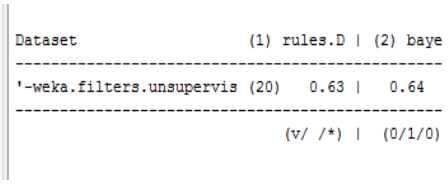


Figura 43 – *Caso de estudo 2 fase* de *river ROC Area*

4.4 Caso de estudo 3 – Atributos Automaticamente Selecionados

Neste caso de estudo, apesar do atributo em estudo continuar a ser a jogada efetuada pelo jogador, será testado um novo modelo de dados, sendo este escolhido automaticamente pelos algoritmos presentes na ferramenta *Weka*.

4.4.1 Pre-Flop

Pelos resultados apresentados na Tabela 28 é observável que, na sua maioria, foram obtidos resultados bastante satisfatórios, com taxas de instâncias corretamente classificadas superiores a 80% e *ROC Area* superiores a 0.9. Apenas o algoritmo *Hyper Pipes* obteve resultados bastante insatisfatórios em todas as estatísticas analisadas.

Apesar de alguma proximidade nos resultados, o *J48* destaca-se como o algoritmo com melhor *Kappa Statistic* e melhor percentagem de instâncias corretamente classificadas. Por sua vez o *IBK* apresenta a melhor *ROC Area*.

Tabela 28 – resultados obtidos no caso de estudo 3 - Atributos Automaticamente Selecionados *pre-flop*

	Corretamente Classificadas	*Kappa Statistic*	*ROC Area*
Naive Bayes	84.73%	0.5855	0.948
J48	**87.97%**	**0.6468**	0.938
Decision Table	87.01%	0.6317	0.943
Hyper Pipes	32.40%	0.0688	0.755
IBK	87.17%	0.6363	**0.95**
Ada Boost M1	83.90%	0.4691	0.878
Attribute Selected Classifier	87.69%	0.6321	0.938
RBF Network	85.41%	0.5955	0.937

Nas matrizes de confusão apresentadas na Tabela 29, verifica-se que apesar do domínio do *J48* na análise anterior, este apenas obtém melhores resultados que os restantes quando o tipo de jogada é *All-In*. Verificando os dados do *IBK*, observamos que o mesmo obtém melhores resultados que os demais em três tipos de jogada (*Call, Raise* e *Check)* mas com valores muito aproximados dos obtidos pelo *J48.*

Tabela 29 – matriz de confusão no caso de estudo 3 - Atributos Automaticamente Selecionados *pre-flop*

Algoritmos	Classificados como ------>	*Call*	*Raise*	*Fold*	*Check*	*All-In*
J48	*Call*	3531	5149	2999	52	22
	Raise	1063	20691	6704	6	3
	Fold	1161	3341	164712	62	18
	Check	736	9	4810	2549	0
	All-In	20	35	16	0	109
IBK	*Call*	3670	5286	2448	342	7
	Raise	1392	20735	6311	28	1
	Fold	1565	4088	162365	1275	1
	Check	621	9	4407	3067	0
	All-In	11	151	4	0	14
Attribute Selected Classifier	*Call*	3425	4675	3598	32	23
	Raise	1014	19797	7645	7	4
	Fold	850	3260	165135	37	12
	Check	708	4	4854	2538	0
	All-In	20	47	16	0	97

Na Figura 45 observa-se uma igualdade estatística entre os algoritmos analisados, ao contrário do que se verifica na Figura 44 a) e na Figura 44 b) onde se denota uma superioridade do *J48* em relação ao *IBK*.

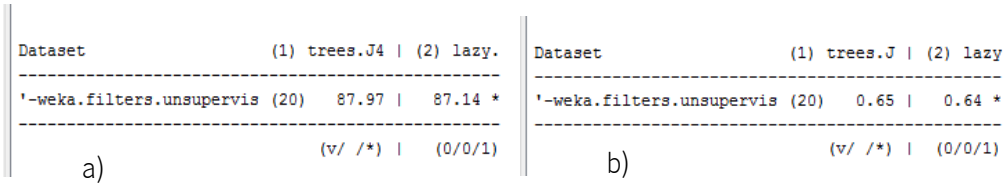


Figura 44 – *Caso de estudo 3 fase* de *pre-flop* a) Instâncias corretamente classificadas; b) *Kappa Statistic ;*

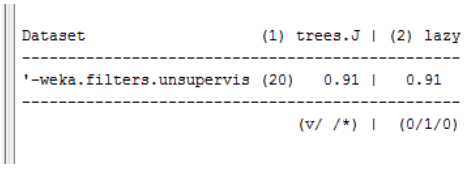


Figura 45 – *Caso de estudo 3 fase* de *pre-flop ROC Area*

4.4.2 Flop

Nesta fase do jogo verifica-se um elevado decréscimo na performance dos algoritmos relativamente à fase anterior. É de notar um decréscimo superior a 30% na percentagem de instâncias corretamente classificadas pelo algoritmo com melhores resultados, que neste caso foi obtido pelo *Decision Table*. É também observável um decréscimo acentuado no *Kappa Statistic* e na *ROC Area*, possuindo esta última um valor ainda com alguma significância onde em ambas o melhor resultado foi obtido pelo *Decision Table* (Tabela 30).

Tabela 30 – resultados obtidos no caso de estudo 3 - Atributos Automaticamente Selecionados *flop*

	Corretamente Classificadas	*Kappa Statistic*	*ROC Area*
Naive Bayes	49.95%	0.2984	0.726
Decision Table	**53.97%**	**0.3316**	**0.736**
Hyper Pipes	35.71%	0.0539	0.551
IBK	44.37%	0.2044	0.641
Ada Boost M1	40.69%	0.154	0.597
RBF Network	44.10%	0.2296	0.645

Observando os dados contidos na Tabela 31 verifica-se que apesar do *Decision Table* ter obtido os melhores resultados na análise anterior, os mesmos resultam de uma elevada tendência para classificar as instâncias como *Call* ou *Check.* Por sua vez, o Naive Bayes e o *RBF Network* são mais equilibrados apesar de perderem eficácia.

Observando-se a Figura 46 b e a Figura 47 verifica-se que o Naive Bayes se pode considerar superior ao *Decision Table.* Por sua vez, o Decision Table é superior em termos de instâncias corretamente classificadas (Figura 46 a).

Tabela 31 – matriz de confusão no caso de estudo 3 - Atributos Automaticamente Selecionados *flop*

Algoritmos	Classificados como ------>	*Call*	*Raise*	*Fold*	*Check*	*All-In*
Decision Table	*Call*	**6952**	0	740	1894	10
	Raise	185	0	72	21	0
	Fold	4687	0	778	3151	6
	Check	1011	0	65	**10095**	0
	All-In	2670	0	128	570	9
Naive Bayes	*Call*	4684	51	2193	1509	1159
	Raise	144	1	78	17	38
	Fold	2287	21	**3152**	2565	597
	Check	975	2	1781	8174	239
	All-In	1764	6	709	405	493
RBF Network	*Call*	3843	273	2190	1745	1545
	Raise	123	**25**	69	31	30
	Fold	2040	208	3142	2460	772
	Check	1285	21	2499	6872	494
	All-In	1385	55	746	502	**689**

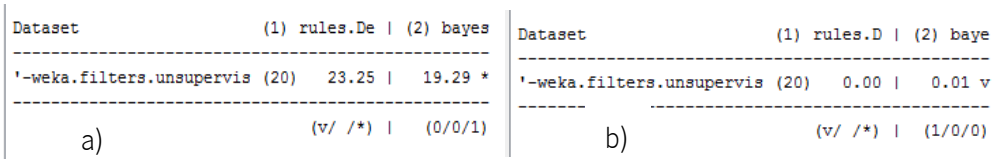


Figura 46 – *Caso de estudo 3 fase* de *flop* a) Instâncias corretamente classificadas; b) *Kappa Statistic ;*

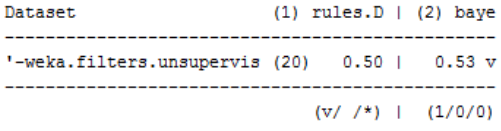


Figura 47 – *Caso de estudo 3 fase* de *flop ROC Area*

4.4.3 Turn

Na Tabela 32 pode-se verificar que apesar da percentagem de instâncias corretamente classificadas não ter sofrido um decréscimo acentuado, o *Kappa Statistic* e a *ROC Area* voltaram a decrescer de uma forma significativa, atingindo valores bastante diminutos ou até mesmo nulos.

O *Naive Bayes* foi o algoritmo que apresentou melhores resultados no *Kappa Statistic* e na *ROC Area*, mas no que toca às instâncias corretamente classificadas foi um dos que piores resultados obteve, ficando a cerca de 6% *do Ada Boost M1* que foi o que apresentou melhores resultados nesta estatística.

Tabela 32 – resultados obtidos no caso de estudo 3 - Atributos Automaticamente Selecionados *turn*

	Corretamente Classificadas	*Kappa Statistic*	*ROC Area*
Naive Bayes	35.87%	**0.0887**	**0.58**
J48	41.58%	0	0.5
Decision Table	41.58%	0	0.5
Hyper Pipes	40.26%	0.043	0.557
IBK	38.69%	0.0661	0.558
Ada Boost M1	**41.75%**	0.0038	0.502
RBF Network	22.93%	0.0455	0.531

Tal como era esperado devido aos valores da *ROC Area* e do *Kappa Statistic,* pode-se observar na Tabela 33 que foram vários os algoritmos que não classificaram nenhuma instância de algum dos tipos de jogadas analisados.

É possível ainda verificar que a o *Ada Boost M1* atingiu uma maior percentagem de instâncias corretamente classificadas derivado a este classificar quase na totalidade as instâncias como *Check*. Já o *Naive Bayes* conseguiu classificar instâncias de quase todos os tipos, assim como o *IBK*.

Tabela 33 – matriz de confusão no caso de estudo 3 - Atributos Automaticamente Selecionados *turn*

Algoritmos	Classificados como ——>	*Call*	*Raise*	*Fold*	*Check*	*All-In*
Naive Bayes	*Call*	**2021**	0	1098	1502	549
	Raise	78	0	25	26	29
	Fold	865	0	**1794**	1853	141
	Check	2007	0	2692	3041	405
	All-In	615	0	296	382	**170**
Ada Boost M1	*Call*	43	0	0	5127	0
	Raise	3	0	0	155	0
	Fold	1	0	0	4652	0
	Check	10	0	0	**8135**	0
	All-In	12	0	0	1451	0
IBK	*Call*	1460	24	682	2812	192
	Raise	58	0	15	76	9
	Fold	560	1	1047	2979	66
	Check	1377	12	1596	4993	167
	All-In	468	9	182	726	78

Como se observa na Figura 48 b e na Figura 49 o *Naive Bayes* supera o *Ada Boost M1* nas métricas *Kappa Statistic e ROC Area*. Já nas instâncias classificadas corretamente – Figura 48 a – *o Ada Boost M1* é superior ao *Naive Bayes* para uma significância de 5%.

```
Dataset                   (1) bayes.Na | (2) meta.
-----------------------------------------------------
'-weka.filters.unsupervis (20)   35.99 |   41.75 v
-----------------------------------------------------
                                 (v/ /*) |   (1/0/0)
```

a)

```
Dataset                   (1) bayes.N | (2) meta
-----------------------------------------------
'-weka.filters.unsupervis (20)   0.09 |   0.00 *
-----------------------------------------------
                                 (v/ /*) |  (0/0/1)
```

b)

Figura 48 – *Caso de estudo 3 fase* de *turn* a) Instâncias corretamente classificadas; b) *Kappa Statistic ;*

```
Dataset                   (1) bayes.N | (2) meta
-----------------------------------------------
'-weka.filters.unsupervis (20)   0.54 |   0.50 *
-----------------------------------------------
                                 (v/ /*) |  (0/0/1)
```

Figura 49 – *Caso de estudo 3 fase* de *turn ROC Area*

4.4.4 River

A Tabela 34 apresenta os resultados atingidos na última fase de jogo, e como é possível observar voltou a existir um decréscimo nos valores obtidos, com a exceção da percentagem de instâncias corretamente classificadas. Apesar da maioria dos algoritmos conterem valores idênticos aos da fase anterior, o *RBF Network* e o *Naive Baye*s atingiram os valores mais baixos já observados nesta estatística.

Tabela 34 – resultados obtidos no caso de estudo 3 - Atributos Automaticamente Selecionados *river*

	Corretamente Classificadas	*Kappa Statistic*	*ROC Area*
Naive Bayes	22.48%	0.0046	**0.503**
J48	**47.68%**	0	0.499
Decision Table	**47.68%**	0	0.499
Hyper Pipes	47.56%	**0.0048**	**0.503**
IBK	47.56%	**0.0048**	0.502
Ada Boost M1	**47.68%**	0	0.499
RBF Network	5.43%	0.0012	0.502

Tal como seria esperado, todos os algoritmos possuem uma tendência para classificar as instâncias com um certo tipo de jogada *(Check)*, sendo praticamente incapazes de classificarem as instâncias de outro tipo.

Como se pode notar na Figura 50 e na Figura 51 nenhum dos algoritmos submetidos à análise estatística se destaca dos demais, não podendo assim tirar nenhuma conclusão acerca da sua superioridade em relação aos outros.

Tabela 35 – matriz de confusão no caso de estudo 3 - Atributos Automaticamente Selecionados *river*

Algoritmos	Classificados como ------>	*Call*	*Raise*	*Fold*	*Check*	*All-In*
Hyper Pipes	*Call*	33	3	12	3078	8
	Raise	4	0	0	48	0
	Fold	12	0	20	2690	1
	Check	45	1	20	5883	6
	All-In	8	0	1	613	4
IBK	*Call*	33	3	12	3078	8
	Raise	4	0	0	48	0
	Fold	12	0	20	2690	1
	Check	45	1	20	5883	6
	All-In	8	0	1	613	4
J48	*Call*	0	0	0	3134	0
	Raise	0	0	0	52	0
	Fold	0	0	0	2723	0
	Check	0	0	0	5955	0
	All-In	0	0	0	625	0

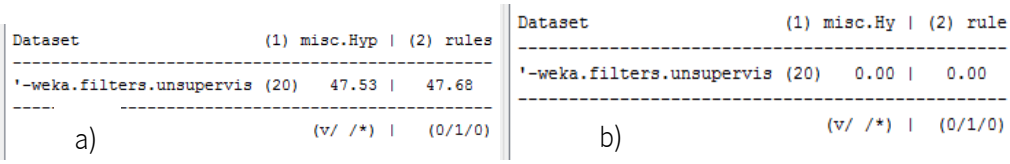


Figura 50 – *Caso de estudo 3 fase* de *river* a) Instâncias corretamente classificadas; b) *Kappa Statistic ;*

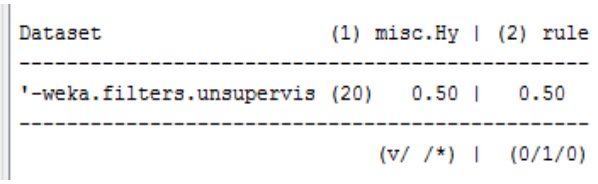


Figura 51 – *Caso de estudo 3 fase* de *river ROC Area*

4.5 Caso de estudo 4 – Atributos Manualmente Selecionados

Neste caso de estudo o modelo de dados utilizado será uma seleção manual dos atributos presentes no caso de estudo 2.

Ao invés de se utilizar algoritmos presentes na ferramenta *Weka* para a seleção dos atributos a utilizar, estes foram selecionados tendo em conta o conhecimento pessoal sobre o jogo poker e sobre *data mining*.

4.5.1 Pre-Flop

Como se pode verificar pela Tabela 36, na fase de *pre-flop* os algoritmos *J48* e *IBK* superaram os 90% em termos de instâncias corretamente classificadas. Em relação às duas outras estatísticas analisadas, pode-se verificar que os melhores resultados também provêm do *J48* e *IBK.*

Todos os algoritmos se comportaram de forma bastante positiva, tendo sido obtidos resultados acima dos 85% no que diz respeito a instâncias corretamente classificadas.

Tabela 36 – resultados obtidos no caso de estudo 4 - Atributos Automaticamente Selecionados *pre-flop*

	Corretamente Classificadas	*Kappa Statistic*	*ROC Area*
Naive Bayes	86.89%	0.6609	0.954
J48	**90.51%**	**0.7355**	0.956
Decision Table	88.10%	0.6814	0.946
IBK	90.04%	0.7284	**0.965**
RBF Network	88.18%	0.6793	0.944

Pela Tabela 37 pode-se verificar que são os algoritmos *J48* e *IBK* que obtêm os melhores resultados no que toca aos verdadeiros positivos.

Apesar dos bons resultados obtidos, verifica-se que existem bastantes jogadas do tipo *Call* que foram classificadas como jogadas de outro tipo.

Tabela 37 – matriz de confusão no caso de estudo 4 - Atributos Automaticamente Selecionados *pre-flop*

Algoritmos	Classificados como ——>	*Call*	*Raise*	*Fold*	*Check*	*All-In*
J48	*Call*	5373	2973	3094	292	21
	Raise	1862	21043	5483	77	2
	Fold	1329	2115	**164213**	1622	15
	Check	241	14	1430	**6419**	0
	All-In	71	20	14	0	**75**

Algoritmos	Classificados como ------>	*Call*	*Raise*	*Fold*	*Check*	*All-In*
Decision Table	*Call*	5685	2857	2906	304	1
	Raise	2380	20451	5566	70	0
	Fold	5672	2496	159595	1531	0
	Check	277	16	1655	6156	0
	All-In	25	0	153	0	2
IBK	*Call*	5970	2940	2566	250	27
	Raise	2494	21372	4521	78	2
	Fold	2111	3082	162745	1354	2
	Check	320	32	1802	5950	0
	All-In	73	38	10	0	59

Através dos dados presentes na Figura 52 a) e na Figura 52 b) pode-se afirmar que o *J48* é superior ao *IBK*. Por sua vez, na Figura 53 verifica-se uma equivalência entre ambos.

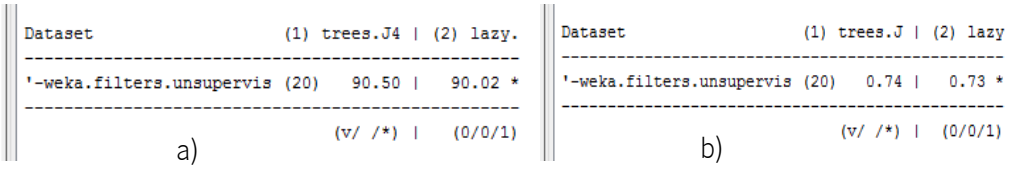


Figura 52 – *Caso de estudo 4 fase* de *pre-flop* a) Instâncias corretamente classificadas; b) *Kappa Statistic ;*

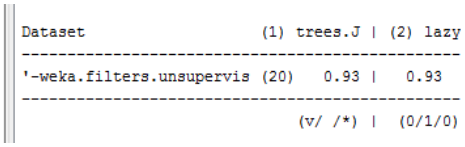


Figura 53 – *Caso de estudo 4 fase* de *pre-flop ROC Area*

4.5.2 Flop

A Tabela 38 apresenta os resultados obtidos pelos algoritmos na fase de *flop.* É possível verificar que os resultados foram inferiores aos obtidos na fase de *pre-flop,* sendo que o melhor resultado em termos de instâncias corretamente classificadas apenas atingiu os 67%, sendo esse resultado obtido pelo *Decision Table.* Este mesmo algoritmo atingiu os melhores valores no *Kappa Statistic* e na *ROC Area*.

Tabela 38 – resultados obtidos no caso de estudo 4 - Atributos Automaticamente Selecionados *flop*

	Corretamente Classificadas	Kappa Statistic	ROC Area
Naive Bayes	52.84%	0.3416	0.766
Decision Table	**67.77%**	**0.4948**	**0.859**
IBK	62.65%	0.4733	0.814
J48	63.81%	0.4807	0.836
RBF Network	60.62%	0.4477	0.824

Nas matrizes de confusão presentes na Tabela 39, apesar do *Decision table* dominar as estatísticas anteriores no que toca aos verdadeiros positivos, apenas consegue obter melhores resultados que os outros algoritmos no tipo de jogada *Check*. O *IBK* é o algoritmo que obtém melhores resultados em mais tipos de jogada (três tipos).

O tipo de jogada *Raise* é o que piores resultados obtém em todos os algoritmos testados.

Tabela 39 – matriz de confusão no caso de estudo 4 - Atributos Automaticamente Selecionados *flop*

Algoritmos	Classificados como ——>	*Call*	*Raise*	*Fold*	*Check*	*All-In*
Decision Table	*Call*	6285	9	2246	947	109
	Raise	160	0	98	19	1
	Fold	3312	2	3826	1456	26
	Check	0	0	0	**11171**	0
	All-In	2552	0	587	118	120
IBK	*Call*	4575	30	2860	1158	973
	Raise	131	**8**	66	53	20
	Fold	1736	4	**4842**	1752	288
	Check	217	3	291	10642	18
	All-In	1799	9	795	139	**635**
J48	*Call*	**6695**	4	2264	614	19
	Raise	161	0	117	0	0
	Fold	4196	2	3284	1133	7
	Check	43	0	26	11102	0
	All-In	2629	0	704	40	4

Como se pode ver pelo estudo estatístico presente na Figura 54 e Figura 55 o algoritmo *Decision Table* supera em todas as métricas o algoritmo *J48* para uma significância de 5%.

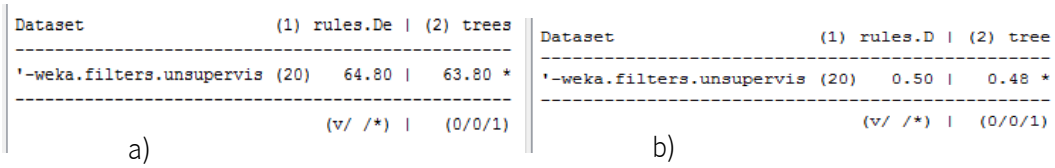


Figura 54 – *Caso de estudo 4 fase* de *flop* a) Instâncias corretamente classificadas; b) *Kappa Statistic ;*

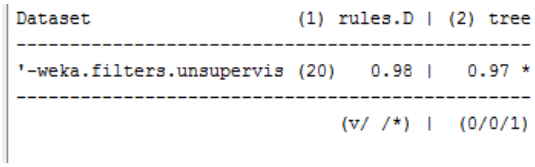


Figura 55 – *Caso de estudo 4 fase* de *flop ROC Area*

4.5.3 Turn

Nos resultados presentes na Tabela 40 pode-se verificar a existência de um decréscimo acentuado em relação aos resultados obtidos na fase anterior.

Em termos de instâncias corretamente classificadas, observa-se que nenhum algoritmo foi capaz de atingir os 50%, sendo que o *RBF Network* foi o que melhores resultados obteve nesta estatística, atingindo os 42%. No que toca ao *Kappa Statistic* e à *ROC Area,* os melhores resultados foram atingidos pelo *Naive Bayes*.

Tabela 40 – resultados obtidos no caso de estudo 4 - Atributos Automaticamente Selecionados *turn*

	Corretamente Classificadas	*Kappa Statistic*	*ROC Area*
Naive Bayes	42.31%	**0.1574**	**0.648**
J48	43.85%	0.1432	0.617
Decision Table	43.63%	0.0982	0.637
IBK	39.28%	0.0781	0.573
Lib SVM	42.00%	0.0133	0.506
RBF Network	**44.42%**	0.1329	0.643

Na Tabela 41 são apresentadas as matrizes de confusão dos algoritmos, sendo que é visível a capacidade do *Naive Bayes* em obter os melhores resultados na maioria dos tipos de jogada (4 tipos). Verifica-se também que o *Decision Table* a obteve o melhor resultado no tipo *Check*.

Tabela 41 – matriz de confusão no caso de estudo 4 - Atributos Automaticamente Selecionados *turn*

Algoritmos	Classificados como ——>	*Call*	*Raise*	*Fold*	*Check*	*All-In*
Naive Bayes	*Call*	**2129**	49	650	1826	516
	Raise	84	**9**	15	36	14
	Fold	744	8	**1545**	2265	91
	Check	1947	30	1578	4333	257
	All-In	569	19	126	477	**272**
RBF Network	*Call*	1987	6	256	2785	136
	Raise	86	1	3	64	4
	Fold	490	2	701	3450	10
	Check	1487	1	660	5944	53
	All-In	646	6	59	683	69
J48	*Call*	1982	23	671	2320	174
	Raise	92	2	7	49	8
	Fold	673	2	1006	2919	53
	Check	1365	14	1115	5533	118
	All-In	576	7	122	692	66
Decision Table	*Call*	1865	0	134	3138	33
	Raise	100	0	0	57	1
	Fold	430	0	171	4048	4
	Check	1386	1	239	**6497**	22
	All-In	649	2	23	775	14

Na análise estatística efetuada na Figura 56 e Figura 57 consta-se que, no que diz respeito às instâncias corretamente classificadas, a conclusão possível de extrair é que o *Naive Bayes* é inferior ao *J48* e ao *RBF Network.*

Por sua vez o *Naive Bayes* supera os restantes algoritmos no *Kappa Statistic* para uma significância de 5%.

Na *ROC Area*, os três algoritmos são equivalentes.

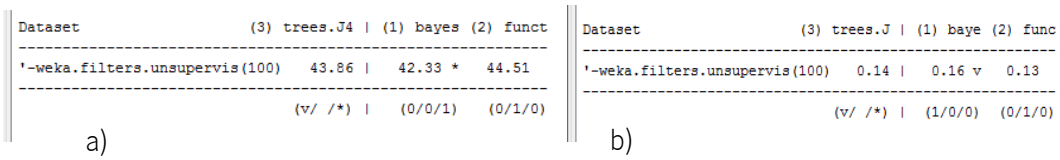


Figura 56 – *Caso de estudo 4 fase* de *turn* a) Instâncias corretamente classificadas; b) *Kappa Statistic ;*

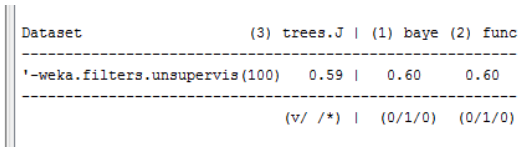


Figura 57 – *Caso de estudo 4 fase* de *turn ROC Area*

4.5.4 River

Na fase do *River,* observa-se que existiu uma melhoria dos resultados comparando com a fase anterior.

Na Tabela 42, pode-se observar que existem algoritmos capazes de superar os 50% quando se trata de instâncias corretamente classificadas. Também o *Kappa Statistic* sofreu uma melhoria significativa.

Tabela 42 – resultados obtidos no caso de estudo 4 - Atributos Automaticamente Selecionados *river*

	Corretamente Classificadas	*Kappa Statistic*	*ROC Area*
Naive Bayes	48.49%	**0.2038**	**0.676**
J48	50.58%	0.185	0.613
Decision Table	**52.11%**	0.1591	0.661
Lib SVM	47.81%	0.004	0.502
IBK	45.61%	0.1132	0.59
RBF Network	50.86%	0.1732	0.663

Nas matrizes de confusão presentes na Tabela 43 pode-se verificar que o *Naive Bayes* obtém os melhores resultados no que toca a verdadeiros positivos em quase todos os tipos de jogada com exceção do *Check*, que é obtido pelo *Decision Table.*

É notória uma tendência dos algoritmos classificarem as instâncias como *Call* ou *Check.*

Tabela 43 – matriz de confusão no caso de estudo 4 - Atributos Automaticamente Selecionados *river*

Algoritmos	Classificados como ------>	*Call*	*Raise*	*Fold*	*Check*	*All-In*
Naive Bayes	*Call*	**1559**	28	248	1020	279
	Raise	29	**2**	4	13	4
	Fold	542	4	**638**	1491	48
	Check	1250	9	819	3762	115
	All-In	326	6	49	150	**95**
Decision Table	*Call*	1218	1	18	1876	21
	Raise	38	0	0	13	1
	Fold	208	0	16	2499	0
	Check	640	0	33	**5273**	9
	All-In	315	0	3	307	1
RBF Network	*Call*	1407	0	63	1582	82
	Raise	28	0	1	23	0
	Fold	378	0	164	2168	13
	Check	944	0	222	4752	37
	All-In	342	0	15	240	29
J48	*Call*	1181	4	354	1506	89
	Raise	28	0	3	19	2
	Fold	334	1	501	1871	16
	Check	722	1	579	4612	41
	All-In	302	1	37	262	24

Na análise estatística verifica-se que o *Naive Bayes* se destaca no que concerne ao *Kappa Statistic* e à *ROC Area* (Figura 58 b e Figura 59). Verifica-se também que o Decision Table é superior nas instâncias corretamente classificadas (Figura 58 b).

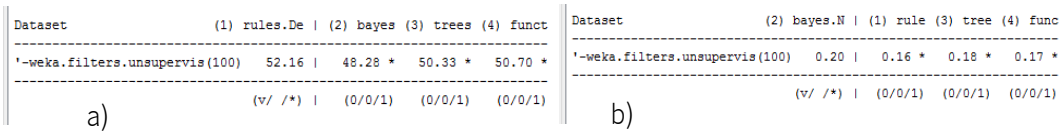


Figura 58 – *Caso de estudo 4 fase* de *river* a) Instâncias corretamente classificadas; b) *Kappa Statistic ;*

```
Dataset                  (2) bayes.N | (1) rule (3) tree (4) func
-------------------------------------------------------------------
'-weka.filters.unsupervis(100)   0.66 |   0.63 *   0.62 *   0.64 *
-------------------------------------------------------------------
                               (v/ /*) |  (0/0/1)  (0/0/1)  (0/0/1)
```

Figura 59 – *Caso de estudo 4 fase* de *river ROC Area*

4.6 Conclusões

Ao longo do trabalho prático que foi desenvolvido neste capítulo, foi apresentada uma experiência preliminar e quatro casos de estudos posteriores.

Sendo a experiência preliminar efetuada numa fase muito inicial da dissertação, a mesma teve como principal objetivo a experimentação da tecnologia e a observação do comportamento dos modelos de *data mining* neste contexto.

Os casos de estudo aqui apresentados, foram efetuados numa fase posterior da dissertação de forma a que fosse possível implementar o conhecimento adquirido na fase inicial da mesma.

Esta secção encontra-se dividida em quatro partes, sendo a primeira dedicada à experiência preliminar, a segunda parte sobre o caso de estudo 1, a terceira secção sobre o caso de estudo 2 e por fim, a quarta secção sobre o caso de estudo 3 e 4.

4.6.1 Experiência preliminar

Na experiência preliminar (página 55) foi possível verificar que mesmo apenas para uma estratégia de *pre-flop*, o modelo de dados torna-se muito importante na obtenção de resultados satisfatórios. Apesar dos resultados obtidos apresentarem uma curva ROC excecional, os mesmos apresentam alguns problemas na classificação de alguns tipos de instâncias, o que leva a crer que o modelo de dados precisa de ser melhorado. Apesar disso é pode-se afirmar que é possível obter um modelo de jogador através de técnicas de *data mining* com resultados bastante satisfatórios.

4.6.2 Caso de estudo 1

O caso de estudo 1 (página 58) tem como principal objetivo o teste de um modelo de dados mais complexo e a sua adequação a uma previsão binária: o jogador mantém-se em jogo ou desiste. O modelo de dados foi testado para a totalidade das fases do poker (*pre-flop, flop, turn e river)*, sendo que

foi possível verificar um decréscimo dos valores das métricas utilizadas ao longo da evolução do jogo com a exceção da última fase *(River)* onde o valor das instâncias corretamente classificadas sofreu um ligeiro aumento (Figura 60).

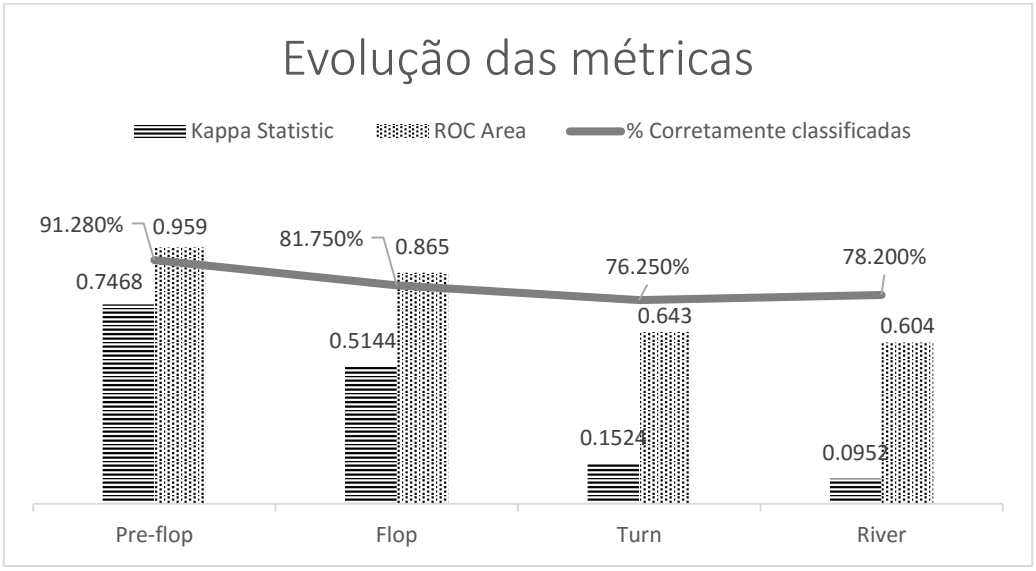


Figura 60 – Evolução das métricas ao longo das 4 fases do poker no caso de estudo 1

Pensa-se que o descrescimo dos valores das métricas deve-se principalmente a dois fatores: um decréscimo no número de jogos disponíveis nas fases adiantadas (Figura 23) e, ou, um modelo de dados desadequado para as fases posteriores.

O facto do valor da métrica instâncias corretamente classificadas se manter elevado ao longo das quatro fases de jogo, deve-se ao facto da existência de algoritmos que classificaram todas as instâncias como sendo de apenas um tipo de jogada (falsos positivos).

Dos algoritmos testados neste caso de estudo, destacaram-se três deles *(Decision Table, Naive Bayes e Ada Boost M1)* devido aos resultados obtidos nas três métricas analisadas. Verificando o estudo estatístico efetuado em cada uma das fases do poker verifica-se que o *Ada Boost M1* apenas é estatisticamente superior em duas de doze ocasiões possíveis, sendo estas o número de instâncias corretamente classificadas. No caso do *Decision Table,* o mesmo é superior aos demais em quatro de doze ocasiões (três na fase de *pre-flop).* O *Naive Bayes,* tal como o *Ada Boost M1,* apenas é superior em duas de doze métricas.

Apesar de não ser possível retirar uma conclusão única sobre qual o melhor algoritmo de *data mining* a utilizar neste caso de estudo, pode-se afirmar que o *Decision Table* consegue obter os melhores

resultados nas duas primeiras fases do poker, sendo que nas duas últimas o *Naive Bayes* destaca-se como sendo o mais equilibrado.

4.6.3 Caso de estudo 2

O caso de estudo 2 (página 64) assume-se como evolução na complexidade do caso de estudo 1. Esse aumento da complexidade advém do facto de que, em vez existirem apenas duas possibilidades no tipo de jogada a efetuar, passam a existir cinco *(Call, Fold, Raise, Check* e *All-In).* Além deste aumento de complexidade, o caso de estudo 2 é também o ponto de partido para os casos de estudo 3 e 4, sendo que nestes dois últimos casos de estudo existe uma alteração do modelo de dados.

Analisando a evolução das métricas ao longo das quatro fases do poker, pode-se verificar que as mesmas sofrem de um decréscimo ao longo do jogo, tal como visto no caso de estudo anterior. Neste caso nota-se um decréscimo mais acentuado na percentagem de instâncias corretamente classificadas devido ao aumento de hipóteses a serem previstas (Figura 61). Uma vez mais, o aumento desta métrica na última fase de jogo, deve-se aos resultados de um algoritmo com um elevado número de falsos positivos no tipo de jogada *Check* (Tabela 27).

A *ROC Area* e o *Kappa Statistic* apresentam valores razoáveis nas duas primeiras partes do jogo, sendo que nas duas últimas existe um decréscimo significativo, atingindo valores bastante baixos. Este decréscimo das duas métricas pode estar relacionado com um decréscimo no número de jogadas existentes para análise (Figura 23) e por um elevado número de amostras em que o tipo de jogada é *Check* (Figura 25).

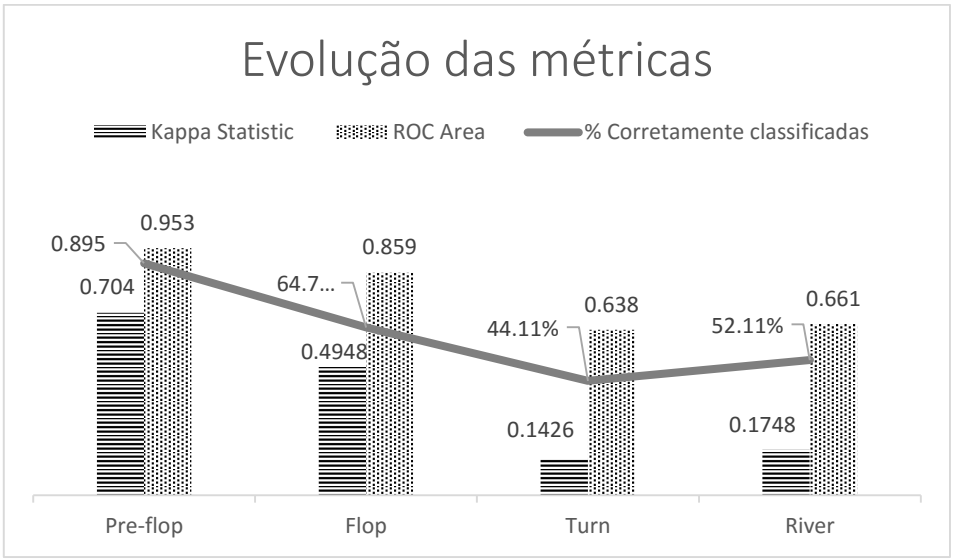


Figura 61 – Evolução das métricas ao longo das 4 fases do poker no caso de estudo 2

Na análise estatística dos algoritmos utilizados, destacam-se três algoritmos, sendo eles o *J48, o Naive Bayes* e o *Decision Table*. Este último algoritmo obteve resultados estatisticamente superiores aos restantes por cinco ocasiões das doze possíveis A contrastar com este, o *Naive Bayes* e o *J48* obteve apenas em duas ocasiões. O *J48* destaca-se principalmente na primeira fase do jogo, sendo as próximas dominadas pelo *Naive Bayes* e pelo *Decision Table*.

Uma vez mais não existe a possibilidade de afirmar que, dos algoritmos testados, existe um estatisticamente superior aos demais. Porém, tendo em conta os resultados obtidos, verifica-se que o *Decision Table,* poderá ser o mais equilibrado.

4.6.4 Caso de estudo 3 e 4

Os casos de estudo 3 e 4 foram efetuados de forma a demonstrar a possibilidade de existência de um modelo de dados com um menor número de atributos comparado ao inicial, mas em que fosse possível obter resultados semelhantes ou superiores.

Desta forma, no caso de estudo 3, foram utilizados atributos escolhidos por algoritmos de seleção presentes na ferramenta *Weka* (página 50) enquanto no caso de estudo 4 foram utilizados atributos escolhidos de forma manual (página 52).

Verificando os dados presentes na Figura 62 *(pre-flop* e *flop)*, pode-se observar que o modelo de dados utilizado no caso de estudo 4 possui uma ligeira vantagem sobre o modelo de dados utilizado no caso de estudo 2 em todas as métricas analisadas. Por sua vez, modelo de dados utilizado no caso de estudo 3 é inferior aos restantes.

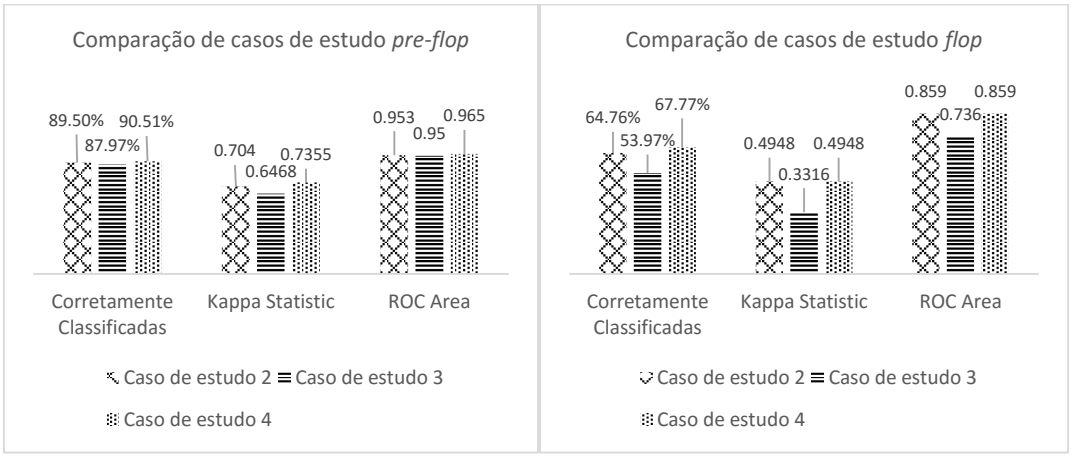


Figura 62 – Comparação entre os casos de estudo na fase de *pre-flop* e na fase de *flop*

Na Figura 63 observam-se resultados semelhantes aos anteriores, sendo que apenas existe uma igualdade entre o caso de estudo 2 e o caso de estudo 4 no que toca às instâncias corretamente classificadas na fase de *River.* Em todas as outras métricas, tanto na fase de *Turn* como na fase de *River* verifica-se uma ligeira vantagem do modelo de dados utilizado no caso de estudo 4. Por sua vez, o modelo de dados utilizado no caso de estudo 3 é inferior, e em alguns casos de forma mais significativa do que os outros modelos de dados utilizados.

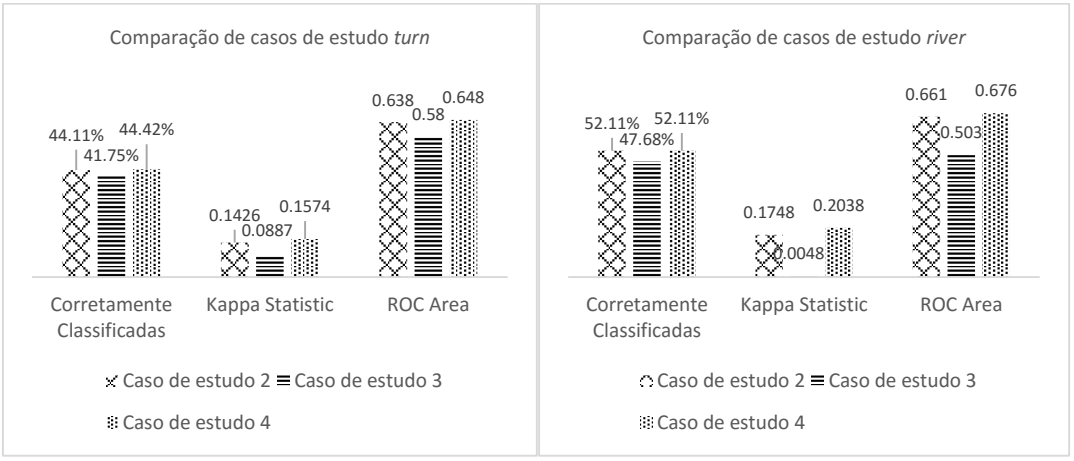


Figura 63 – Comparação entre os casos de estudo na fase de *turn* e na fase de *river*

Tendo em conta todos os resultados obtidos e discutidos anteriormente, pode-se afirmar que o modelo de dados utilizado no caso de estudo 4 é superior ao utilizado no caso de estudo 2.

5 Conclusões e Trabalho Futuro

O poker tem atraído cada vez mais as atenções do mundo científico, nomeadamente da comunidade que investiga a I.A. Devido ao facto de o poker ser um jogo estocástico e de informação imperfeita, características que o desmarcam dos outros jogos (como o xadrez), tem demonstrado ser um desafio de complexa resolução para a comunidade científica.

É neste âmbito que a presente dissertação foi desenvolvida, para explorar a possibilidade de replicar as acções de um jogador de poker profissional num agente virtual.

Neste trabalho foi efetuada uma revisão da literatura dos dois temas mais relevantes para o desenvolvimento da dissertação: abordagens na área da inteligência artificial para a criação de agentes capazes de jogar Poker e métodos e algoritmos de *data mining*.

No que toca à investigação da Inteligência Artificial na área do poker, verificou-se uma grande quantidade de possíveis abordagens para a construção de um agente virtual, sendo que nenhuma delas é capaz de bater os melhores jogadores humanos na vertente *Multi-Player Texas Hold'em no Limit Poker*. Na vertente *Heads up* (1 contra 1) foi criado recentemente o *Cepheus*, um agente que se diz capaz de vencer todos os seus oponentes. Para tal foi necessária a construção de um *cluster* com cerca de 4000 CPU's, sendo que o agente virtual foi capaz de aprender ao longo do tempo tendo em conta os resultados obtidos nos jogos efetuados. Na vertente *Texas Hold'em no Limit* existe um agente virtual testado em jogos *online* que foi capaz de obter lucros em vários jogos (Teofilo et al., 2014).

Apesar das várias abordagens existentes, verifica-se um reduzido número de trabalhos com utilização de *data mining* na construção de agentes capazes de jogar Poker. Além da pouca utilização deste conjunto de técnicas, foi possível verificar que ainda não foi tentada a criação de um agente virtual capaz de imitar ou procurar jogar de forma idêntica a um determinado jogador humano. Posto isto, conclui-se que esta dissertação pode apresentar algo de inovador para a comunidade científica.

Para o desenvolvimento da dissertação foram utilizadas duas abordagens metodológicas. Uma dedicada exclusivamente à área de investigação, o DSRIS e o CRISP-DM para o desenvolvimento da parte prática, sendo que não foi levada a cabo a fase de implementação desta metodologia. Apesar da utilização de duas abordagens distintas, estas acabam por se fundir, pois a execução da metodologia CRISP-DM ocorre durante as fases de *Design* e Desenvolvimento, Demonstração e Avaliação do DSRIS.

Do trabalho prático efetuado durante a dissertação, foi possível verificar a elevada importância do modelo de dados utilizado para atingir bons resultados. Apesar dos resultados atingidos não refletirem valores excelentes, dada a complexidade do problema em análise, foi possível atestar a possibilidade da construção de um modelo de jogador capaz de imitar a forma de jogar de um jogador profissional de poker. Para tal destacaram-se alguns algoritmos de *data mining* tais como o *Naive Bayes* e o *Decision Table.*

Foi também verificado que o modelo de dados não necessita de uma elevada quantidade de atributos para a obtenção de resultados satisfatórios. Poderá ser utilizado um modelo de dados com um menor número de atributos, sendo que se deve optar por atributos com poucos valores distintos, optando pela criação de classes quando existe uma elevada variedade de valores.

Ao longo de todo o trabalho efetuado, fica presente uma grande dificuldade na utilização de algumas ferramentas usadas para a execução do trabalho prático, nomeadamente no *Talend Data Integration* para a execução do ETL e no *Weka* na execução de vários algoritmos. Esta dificuldade deve-se à fraca capacidade de processamento da máquina utilizada, impedindo de certa forma, a realização de um conjunto de experiências mais alargadas. Apesar dessas dificuldades, foi possível concluir com sucesso a construção do modelo de dados e a sua população através do processo de ETL, assim como a execução de diversos algoritmos de *data mining*.

Tendo em consideração todo o trabalho desenvolvido ao longo desta dissertação bem como todas as dificuldades sentidas, espera-se que de futuro seja possível implementar o modelo de jogador num agente virtual de forma a serem efetuados testes para validar o modelo.

Numa fase posterior deve ser efetuada uma análise aprofundada de uma arquitetura de *hardware* e *software* capaz de suportar o processamento de um elevado número de registos. Após a construção e validação de uma nova arquitetura, deverá ser efetuada uma nova iteração da metodologia CRISP-DM. Isto para que seja revisto o modelo de dados na tentativa de construção de atributos que possam estar mais correlacionados com a decisão do jogador. Para esta tarefa é aconselhável que a mesma seja acompanhada por um jogador profissional de poker por forma a que seja facilitada a compreensão de qual a informação correlacionada com a tomada de decisão. Com uma nova arquitetura que suporte uma elevada necessidade de processamento, será possível a utilização de novos algoritmos de *data-mining*. Posto isto, aconselha-se a experimentação de SVM e de redes neuronais de forma a comprovar se são capazes de produzir melhores resultados.

Referências Bibliográficas

Abel, F., Celik, I., Hauff, C., Hollink, L., & Houben, G.-J. (2011). U-Sem: Semantic Enrichment, User Modeling and Mining of Usage Data on the Social Web, 10–13. Retrieved from http://arxiv.org/abs/1104.0126

Abel, F., Gao, Q., Houben, G. J., & Tao, K. (2011). Analyzing user modeling on Twitter for personalized news recommendations. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 6787 LNCS, pp. 1–12). http://doi.org/10.1007/978-3-642-22362-4_1

Azevedo, A., & Santos, M. F. (2008). Proceedings of Informatics 2008 and Data Mining 2008. In A. Weghorn, Hans & A. P. (Eds.), *Kdd, Semma and Crisp-Dm: A Parallel Overview* (pp. 182–185). Amsterdam.

Bankrollup. (2014). Poker Hand Rankings. Retrieved February 9, 2015, from http://www.bankrollup.com/poker-hand-rankings/

Barr, G. D. I., & Dowie, K. (2011). Swivelling Probabilities—The Winning Hand at Poker. *The American Statistician*, *65* (January), 170–176. http://doi.org/10.1198/tast.2011.09175

Bauckhage, C., Kersting, K., Sifa, R., Thurau, C., Drachen, A., & Canossa, A. (2012). How players lose interest in playing a game: An empirical study based on distributions of total playing times. *2012 IEEE Conference on Computational Intelligence and Games, CIG 2012*, 139–146. http://doi.org/10.1109/CIG.2012.6374148

Beattie, B., & Nicolai, G. (2007). Pattern classification in no-limit poker: A head-start evolutionary approach. *Advances in Artificial ….* Retrieved from http://link.springer.com/chapter/10.1007/978-3-540-72665-4_18

Billings, D., Burch, N., Davidson, A., Holte, R., Schaeffer, J., Schauenberg, T., & Szafron, D. (2003). *Approximating Game-Theoretic Optimal Strategies for Full-scale Poker*. Department of Computing Science, University of Alberta.

Billings, D., Davidson, A., Schaeffer, J., & Szafron, D. (2002). The challenge of poker. *Artificial Intelligence*, *134* (1-2), 201–240. http://doi.org/10.1016/S0004-3702(01)00130-8

Billings, D., Papp, D., Schaeffer, J., & Szafron, D. (1998). Opponent Modeling in Poker. In *Opponent Modeling in Poker*. American Association of Artificial Intelligence.

Bowling, M., Burch, N., Johanson, M., & Tammelin, O. (2015). *Heads-up Limit Hold ' em Poker is Solved*. University of Alberta.

Brown, A. (2011). *The Poker Face of Wall Street*. John Wiley & Sons.

Cai, W., & Li, L. (2004). Network Traffic Anomaly Detection Using TCP Header Information, (May 2004),

1–22. http://doi.org/10.1.1.111.1037

Campbell, M., Hoane Jr., a. J., & Hsu, F. (2002). Deep Blue. *Artificial Intelligence*, *134*, 57–83. http://doi.org/10.1016/S0004-3702(01)00129-1

Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., & Wirth, R. (2000). *CRISP-DM 1.0*.

Cruz, A., & Cortez, P. (2009). Data Mining via Redes euronais Artificiais e Máquinas de Vectores de Suporte. *Review Literature And Arts Of The Americas*, *VII*, 099–118.

Dataprix. (2015). O modelo de referência CRISP-DM. Retrieved February 4, 2015, from http://www.dataprix.net/pt-pt/o-modelo-refer-ncia-crisp-dm

Davidson, A. (2002). *Opponent modelling in poker: Learning and acting in a hostile and uncertain environment*. University of Alberta.

Drachen, A., El-Nasr, M. S., & Canossa, A. (2013). Game Analytics – The Basics. In *Game Analytics* (pp. 13–40). Springer. Retrieved from http://link.springer.com/10.1007/978-1-4471-4769-5

Drachen, A., Thurau, C., Togelius, J., & Bauckhage, C. (2013). Game data mining. In M. S. El-Nasr, A. Drachen, & A. Canossa (Eds.), *Game analytics* (pp. 205–253). Springer London. http://doi.org/10.1007/978-1-4471-4769-5_12

Faria, B. (2013). *Patient Classification for Intelligent Wheelchair Adaptation*. Universidade de Aveiro.

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From Data Mining to Knowledge Discovery in Databases. *AI Magazine*.

Fiedler, I., & Wilcke, A.-C. (2011). The Market for Online Poker. *SSRN Electronic Journal*, *16*, 7–19. http://doi.org/10.2139/ssrn.1747646

Fischer, G. (2001). User Modeling in Human-Computer Interaction. *User Modeling and User-Adapted Interaction*, 1–18.

Frias-Martinez, E., Chen, S. Y., & Liu, X. (2006). Survey of Data Mining Approaches to User Modeling for Adaptive Hypermedia. *IEEE Transactions on Systems, Man and Cybernetics-Part C: Applications and Reviews*, *36*(6), 734–749. http://doi.org/10.1109/TSMCC.2006.879391

Johanson, M. (2013). *Measuring the size of large no-limit poker games*. *arXiv preprint arXiv:1302.7008*. University of Alberta. Retrieved from http://arxiv.org/abs/1302.7008

Johanson, M. B. (2007). *Robust strategies and counter-strategies: Building a champion level computer poker player*. University of Alberta.

Koh, H. C., Tan, W. C., & Goh, C. P. (2006). A two-step method to construct credit scoring models with data mining techniques. *International Journal of Business and Information*, *1*(1), 96–118. Retrieved from http://www.knowledgetaiwan.org/ojs/files/Vol1No1/Paper_5.pdf

Korf, R. E. (2010). Artificial Intelligence Search Algorithms 22.1. In M. J. Atallah & M. Blanton (Eds.),

Algorithms and theory of computation handbook 2 (pp. 22–22). Chapman & Hall/CRC.

Kusiak, a, Kernstine, K. H., Kern, J. a, Mclaughlin, K. a, & Tseng, T. L. (2000). Data Mining : Medical and Engineering Case Studies The University of Iowa. *Industrial Engineering*, 1–7.

Lanctot, M., Lisý, V., & Bowling, M. (2014). Search in Imperfect Information Games using Online Monte Carlo Counterfactual Regret Minimization. *Proceedings of the AAAI Workshop on Computer Poker and Imperfect Information*, (Sandholm 2010).

Liao, S.-H., Chu, P.-H., & Hsiao, P.-Y. (2012). Data mining techniques and applications – A decade review from 2000 to 2011. *Expert Systems with Applications*, *39* (12), 11303–11311. http://doi.org/10.1016/j.eswa.2012.02.063

Mahlmann, T., Drachen, A., Togelius, J., Canossa, A., & Yannakakis, G. N. (2010). Predicting player behavior in Tomb Raider: Underworld. *Proceedings of the 2010 IEEE Conference on Computational Intelligence and Games, CIG2010*, 178–185. http://doi.org/10.1109/ITW.2010.5593355

McCracken, D. D., & Reilly, E. D. (2003). Backus-Naur form (BNF). In A. Ralston, E. D. Reilly, & D. Hemmendinger (Eds.), *Encyclopedia of Computer Science 4th* (pp. 129–131). John Wiley and Sons Ltd. Retrieved from http://dl.acm.org/ft_gateway.cfm?id=1074155&type=html

Meyer, D. (2014). Support vector machines: the interface to libsvm in package e1071. *... Systems and Their ....*

Nash, J. (1950). Equilibrium points in n-person games. In *Proceedings of the National Academy of Sciences* (Vol. 36, pp. 48–49).

Nash, J. (1951). *Non-Cooperative Games*. *Annals of Mathematics* (Second Ser). Annals of Mathematics.

Nicolai, G., & Hilderman, R. J. (2009). No-limit Texas Hold'em Poker agents created with evolutionary neural networks. In *CIG2009 - 2009 IEEE Symposium on Computational Intelligence and Games* (pp. 125–131). http://doi.org/10.1109/CIG.2009.5286485

North, M. (2012). Data Mining for the Masses, 264. Retrieved from http://1xltkxylmzx3z8gd647akcdvov.wpengine.netdna-cdn.com/wp-content/uploads/2013/10/DataMiningForTheMasses.pdf\nhttps://sites.google.com/site/dataminingforthemasses/

Obenshain, M. K. (2004). Application of data mining techniques to healthcare data. *Infection Control and Hospital Epidemiology : The Official Journal of the Society of Hospital Epidemiologists of America*, *25*(January), 690–695. http://doi.org/10.1086/502460

Padhy, N., Mishra, P., & Panigrahi, R. (2012). The Survey of Data Mining Applications And Feature Scope. *International Journal of Computer Science, Engineering and Information*, *2*(3), 43–58. http://doi.org/10.5121/ijcseit.2012.2303

Papp, D. R. (1998). *Dealing with Imperfect Information in Poker*. University of Alberta.

Patel, M. (2012). Analyzing the word Analytics in an ever-increasing Analytical world. Retrieved from http://www.function1.com/2012/03/analyzing-the-word-analytics-in-an-ever-increasing-analytical-world

Peffers, K., Tuunanen, T., Rothenberger, M. a., & Chatterjee, S. (2008). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, *24* (3), 45–77. http://doi.org/10.2753/MIS0742-1222240302

Piatetsky-shapiro, G. (2014). CRISP-DM, still the top methodology for analytics, data mining, or data science projects. Retrieved December 28, 2014, from http://www.kdnuggets.com/2014/10/crisp-dm-top-methodology-analytics-data-mining-data-science-projects.html

Risk, N. A. (2009). *Using counterfactual regret minimization to create a competitive multiplayer poker agent*. *Computing*. University of Alberta.

Rohanizadeh, S. S., & Moghadam, M. B. (2009). A Proposed Data Mining Methodology and its Application to Industrial Procedures. *Journal of Industrial Engineering*, *4*, 37–50. Retrieved from http://www.sid.ir/en/VEWSSID/J_pdf/1029920090405.pdf

Romero, C., Ventura, S., & De Bra, P. (2004). Knowledge discovery with genetic programming for providing feedback to courseware authors. *User Modelling and User-Adapted Interaction*, *14*(2004), 425–464. http://doi.org/10.1007/s11257-004-7961-2

Rubin, J. (2013). *On the Construction , Maintenance and Analysis of Case-Based Strategies in Computer Poker*. The University of Auckland.

Rubin, J., & Watson, I. (2011). Computer poker: A review. *Artificial Intelligence*, *175*(5-6), 958–987. http://doi.org/10.1016/j.artint.2010.12.005

Santos, M. Y. (2001). *Padrão Um Sistema de Descoberta de Conhecimento em Bases de Dados Geo-referenciadas*. Universidade do Minho.

Schaeffer, J., & Van den Herik, H. J. (2002). Games, computers, and artificial intelligence. *Artificial Intelligence*, *134*, 1–7. http://doi.org/10.1016/S0004-3702(01)00165-5

Schauenberg, T. (2006). *Opponent modelling and search in poker*. University of Alberta.

Schnizlein, D. P. (2009). *State translation in no-limit poker*. University of Alberta. Retrieved from http://repository.library.ualberta.ca/dspace/handle/10048/508

Shearer, C. (2000). The CRISP-DM Model: The New Blueprint for Data Mining. *Journal of Data Warehousing*, *5*(4), 9.

Silva, N., & Reis, L. P. (2015). Poker learner: Players modeling through data-mining. In *2015 10th Iberian Conference on Information Systems and Technologies (CISTI)* (pp. 1–6). IEEE. http://doi.org/10.1109/CISTI.2015.7170624

Skalinsky, B. D., & Malmuth, M. (1999). *Hold ' em Poker For Advanced Players* (3ª ed.). Two Plus Two

Pub.

Teofilo, L. F., & Reis, L. P. (2011). HoldemML: A framework to generate No Limit Hold'em Poker agents from human player strategies. *6th Iberian Conference on Information Systems and Technologies (CISTI 2011)*, 1–6.

Teófilo, L. F., & Reis, L. P. (2011a). Building a Poker Playing Agent based on Game Logs using Supervised Learning. *Lecture Notes in Computer Science Volume 6752*, 73–82. Retrieved from http://link.springer.com/chapter/10.1007%2F978-3-642-21538-4_8

Teófilo, L. F., & Reis, L. P. (2011b). Identifying Player's Strategies in No Limit Texas Hold ' em Poker through the Analysis of Individual Moves. In *EPIA'2011* (pp. 978–989).

Teofilo, L. F., Reis, L. P., & Cardoso, H. L. (2014). A Profitable Online No-Limit Poker Playing Agent. In *2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)* (pp. 286–293). Ieee. http://doi.org/10.1109/WI-IAT.2014.179

Teófilo, L. F., Reis, L. P., & Cardoso, H. L. (2013). Speeding-Up Poker Game Abstraction Computation : Average Rank Strength. In *Workshops at the Twenty-Seventh AAAI Conference on Artificial Intelligence* (pp. 59–64).

Teófilo, L. F., Reis, L. P., Cardoso, H. L., Félix, D., Sêca, R., Pedro, M., ... Passos, N. (2013). *Computer Poker Research at LI ACC*.

Teófilo, L. F., Rossetti, R., Reis, L. P., Cardoso, H. L., & Nogueira, P. A. (2013). *Simulation and performance assessment of Poker Agents*. *Multi-Agent-Based Simulation XIII* (Vol. 7838). Springer-Verlag Berlin Heidelberg. http://doi.org/10.1007/978-3-642-38859-0

Teófilo, L., Reis, L., Cardoso, H., & Mendes, P. (2014). Rule based strategies for large extensive-form games: A specification language for No-Limit Texas Hold'em agents. *Computer Science and Information Systems*, *11*, 1249–1269. http://doi.org/10.2298/CSIS130921029T

University of Alberta Computer Poker Research Group. (2015). Poker research. Retrieved January 5, 2015, from http://poker.cs.ualberta.ca/

Verma, G., & Verma, V. (2012). Role and Applications of Genetic Algorithm in Data Mining. *International Journal of Computer Applications*, *48*(17), 5–8. http://doi.org/10.5120/7438-0267

Weber, B. G., & Mateas, M. (2009). A data mining approach to strategy prediction. *CIG2009 - 2009 IEEE Symposium on Computational Intelligence and Games*, 140–147. http://doi.org/10.1109/CIG.2009.5286483

Wikibooks. (2015). Artificial Neural Networks/Activation Functions. Retrieved February 22, 2015, from http://en.wikibooks.org/wiki/Artificial_Neural_Networks/Activation_Functions#Activation_Functions

Wikipedia. (2014a). Blind (poker). Retrieved February 9, 2015, from http://en.wikipedia.org/wiki/Blind_(poker)

Wikipedia. (2014b). K-nearest neighbors algorithm. Retrieved from http://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

Williams, D., Consalvo, M., Caplan, S., & Yee, N. (2009). Looking for gender: Gender roles and behaviors among online gamers. *Journal of Communication*, *59*, 700–725. http://doi.org/10.1111/j.1460-2466.2009.01453.x

Witten, I. H., Frank, E., & Hall, M. A. (2001). *Data Mining Practical Machine Learning Tools and Techniques* (Third Edit, Vol. 40). Elsevier B.V. http://doi.org/10.1002/1521-3773(20010316)40:6<9823::AID-ANIE9823>3.3.CO;2-C

Zinkevich, M., Bowling, M., Johanson, M., & Piccione, C. (2007). *Regret Minimization in Games with Incomplete Information*.