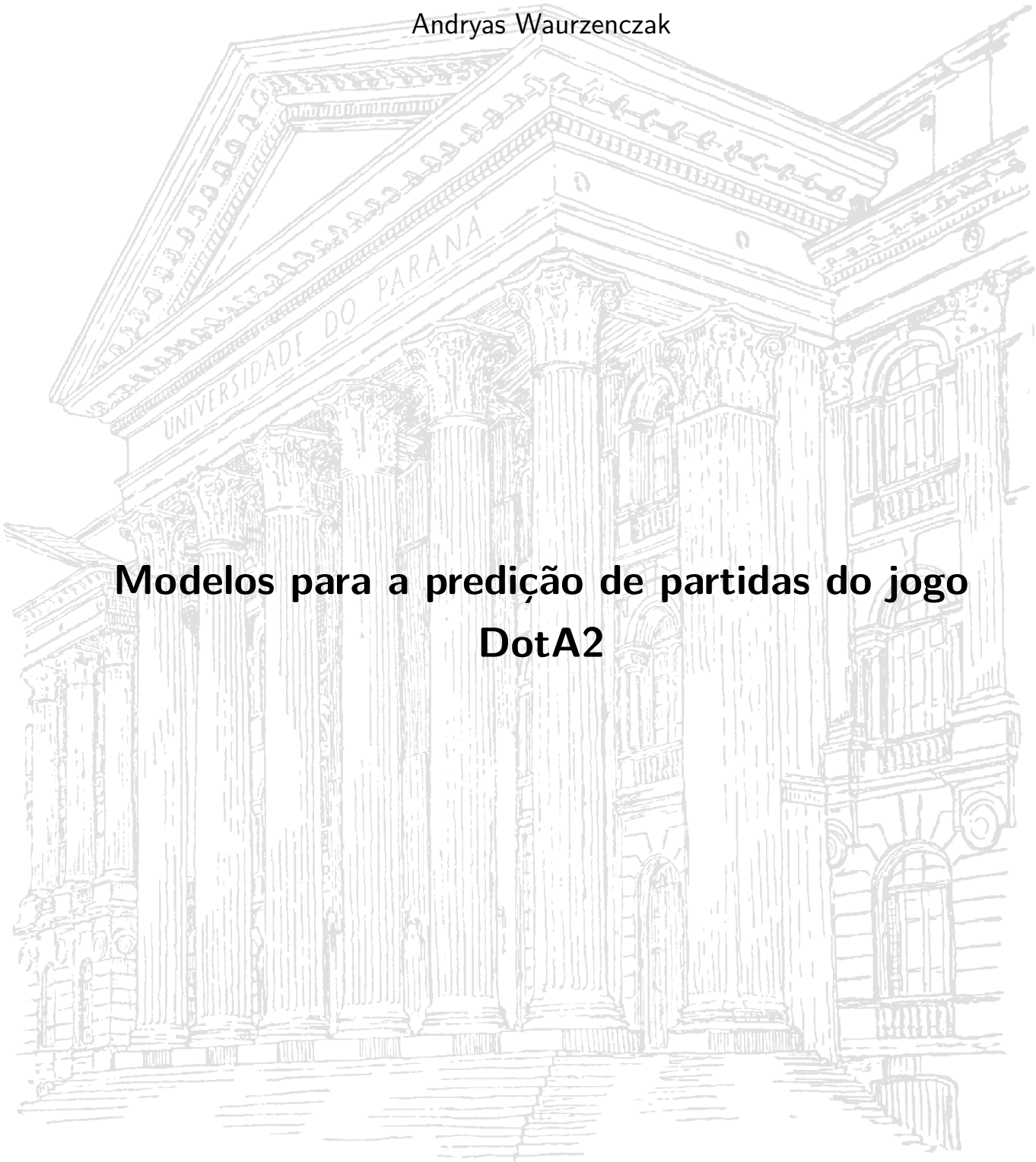
Universidade Federal do Paraná

Andryas Waurzenczak



Modelos para a predição de partidas do jogo DotA2

Curitiba

2019-06-17

Andryas Waurzenczak

Modelos para a predição de partidas do jogo DotA2

Trabalho de Conclusão de Curso apresentado à disciplina Laboratório B do Curso de Graduação em Estatística da Universidade Federal do Paraná, como exigência parcial para obtenção do grau de Bacharel em Estatística.

Universidade Federal do Paraná

Setor de Ciências Exatas

Departamento de Estatística

Orientador: Prof. Dr. Walmes Marques Zeviani

Curitiba

2019-06-17

"As nossas obrigações perante à sociedade são proporcionais aos privilégios que nos foram dados."

— Autor desconhecido

Resumo

Palavras-chave: DotA2, Predição, Engenharia de Características

O presente trabalho buscou adicionar características passadas de partidas de DotA2 com o intuito de melhorar a capacidade preditiva de modelos propostos na literatura e também propor um modelo em que não fosse necessário informações pré-partida. Os resultados obtidos foram diferentes daqueles alcançados por outros autores sugerindo que o jogo ficou mais complexo. Além disso outro resultado obtido sugere que o acréscimo de informações passadas de cada time tem um efeito minímo na performace preditiva do modelo.

Lista de ilustrações

Lista de tabelas

Sumário

1 Introdução

Dos astrágalos da antiga Grécia aos jogos de console e computador do século XXI, ao longo da trajetória humana os jogos tem desempenhado um papel essencial para o desenvolvimento da ciência e da tecnologia. No século XV, por exemplo, o jogo foi o principal motivador, segundo (BERNSTEIN, 1997), para Girolamo Cardano, junto com Blaise Pascal, enunciarem as primeiras linhas do que mais tarde viria ser conhecido como as leis da probabilidade. Essa ideia postulada a mais de cinco séculos atrás foi fundamental para o desenvolvimento e o firmamento posterior de novas áreas do conhecimento, como a Estatística, que até então não existiam.

Já na atualidade, os jogos, principalmente os de computador e console, tem servido como verdadeiros laborátorios para o desenvolvimento de áreas de Inteligência Artificial e Aprendizado de Máquina, cujas raizes estão na Estatística e na Matemática. Isto porque, os jogos, proporcionam um ambiente de teste controlado, que aliado à grande de quantidade de dados produzidos por eles e disponibilizados via API[1] permitem que pesquisadores possam desenvolver e comparar seus agentes com humanos (SILVA; CHAIMOWICZ, 2017). Um exemplo disso é o projeto openAI, que tem como um dos fundadores o empresário Elon Musk, que no campeonato mundial de DotA2 de 2017, The International 2017, testou seu agente contra jogadores profissionais em um estilo de jogo 1vs1 e venceu quase todas as partidas! No ano seguinte foi posto a prova contra times experientes, ou seja jogos 5vs5, e o resultado não foi tão bom quanto o primeiro, monstrando assim que trabalho em equipe é fundamental para um bom desempenho no jogo e que existe muito espaço para crescimento.

Além disso, os jogos de computador passaram recentemente a serem considerados como um esporte, segundo (CALVO, 2017), "um termo que se ouve com maior frequência é *eSports* (Esportes Eletrônicos), o próprio nome faz referência aos jogos eletrônicos, porém apresentando-os como verdadeiros esportes que possui: seus atletas profissionais, disputas acirradas em nível mundial, equipes favoritas e uma crescente audiência". Assim os jogos podem ser considerados tanto um objeto de entretenimento como de mercado de trabalho e de pesquisa científica, sendo este último o foco desse trabalho.

O tema desse trabalho é o jogo DotA2 que tem suas raizesem nos jogos de Real-time Strategy (RTS), ele acontece em uma arena fechada (1) que é divida em dois times de cinco jogadores, os Iluminados (The Radiant) e os Temidos (The Dire). O jogo se inicia escolhendo um dos 117 personagens possíveis, cada persona, comumente chamado de

[1] Application Programming Interface

herói, tem traços únicos, desde as habilidades especiais, itens e até os atributos como nível de força, agilidade, inteligência, resistência a magia e a golpes, velocidade de movimentação, campo de visão entre outros. Após a formação do time, os heróis se posicionam em um dos três caminhos do mapa para defender o avanço do time inimigo assim como obter ouro e experiência e consequentemente o avanço do próprio time à base inimiga com intuito de destruir a estrutura central, que da à vitoria ao time, e que faz juz ao nome do jogo Defense of the Ancient.

O jogo pode ser dividido em diversas fases, no entanto é comumente dividido em três, começo do jogo (early game), meio do jogo (mid game) e fim de jogo (late game), isto porque a combinação dos heróis em times de 5 determinam estrátegias mais eficientes para cada fase do jogo, ou seja, algumas combinações tem vantagens no inicio, e portanto devem ter prioridade em acabar o jogo o quanto antes, enquanto que outras tem vantagens apenas em jogos mais longos, e portanto devem ter prioridade em manter um jogo mais equilibrado em seu inicio para que se possa extender até um determinado tempo em que seus heróis mais fortes tenham acumulado grandes quantidades de ouro e experiência para entrar fortemente no jogo.

A dependência da quantidade de ouro e experiência acumulado varia de herói para herói, sendo que uns necessitam mais de um que o outro, e as vezes precisam dos dois, por isso a formação de um time deve ser equilibrada para que todos possam maximizar suas principais características. Os heróis, como dito anteriormente, são únicos porém suas principais funções podem ser resumidas, de modo geral, em três principais, sendo elas Suport, Offlane e Carry, onde o Suport tem como papel comprar itens auxiliares do jogo e principalmente ajudar o seu Carry a acumular ouro e experiência, já o Offlane tem como objetivo acumular experiência e fazer itens de sustância para seu time e o Carry tem como objetivo acumular ouro e experiência para conseguir fazer itens e aniquilar seu inimigos rápidamente.

O jogo é complexo, apresentando um leque de possibilidades desde a escolha dos heróis e a formação de times até as estrátegias durante o jogo com posições e itens. No entanto, mesmo com toda essa complexidade alguns trabalhos relacionados ao tema foram feitos explorando principalmente a presença ou não de um herói em uma partida sem levar em conta os demais aspectos e obtiveram resultados interessantes. Um dos primeiros trabalhos foi feito por (CONLEY; PERRY, 2013) que atráves de um modelo de regressão lógistica obteve uma acurácia de teste de 69.8% na predição do desfecho final do jogo, outro trabalho foi feito por (CALVO, 2017) porém utilizando o modelo Naive-Bayes, ele obteve uma acurácia de 72.21% na base de teste. A diferença entre os trabalhos está na época em que cada um foi feito e no método empregado, no entanto o ponto de partida é o mesmo, utilizar a informação dos heróis presentes na partida e o time o qual pertencem, *radiant* ou *dire*, para prever seu desfecho, porém para isso

é necessário saber a priori quais personagens foram escolhidos e quais times eles se encontram, de modo que a previsão do resultado da partida está restrita as informações que dela são necessárias.

Assim esse trabalho tem como objetivo construir e adicionar novas características a partir do histórico de partidas de cada jogador que compoem cada time de modo a a avaliar se existe um ganho preditivo nos modelos já feitos e também propor um modelo que independa das informações no instante inicial do jogo, de modo a prever o desfecho da partida entre dois times sem ter a necessidade de informações iniciais do jogo.

O presente trabalho foi estruturado em três partes sendo a primeira, Materias e Métodos, onde é apresentado o conjunto de dados, o processo de aquisição junto com algumas características e restrições, é também apresentado três técnicas usualmente utilizadas em modelagem preditiva que são: o modelo de regressão logística, a curva ROC e a validação cruzada. Na segunda parte, Resultados, é apresentado os modelos propostos e discutido seus resultados. Por último, em Considerações finais, é feito um apanhado geral dos resultados obtidos e feito algumas recomendações para estudos futuros.

Figura 1 – Mapa do jogo - EHOME vs ALLIANCE

Fonte o autor

2 Materias e Métodos

2.1 Recursos Computacionais

O software R (R Core Team, 2019) foi utilizado tanto para a coleta quanto para as análises dos dados, e os principais pacotes auxiliares utilizados no trabalho são: *RDota2Plus*[1], desenvolvido pelo autor para coleta dos dados, *tidyverse* e *data.table* para preprocessamento e manipulação dos dados, *RMySQL* para conexão com banco de dados MySQL, *mongolite* para conexão com o banco de dados MongoDB e por último *caret* para ajuste e validação dos modelos.

Além disso dois bancos de dados foram utilizados, são eles: MongoDB[2] e MySQL[3], sendo o primeiro para o armazenamento dos dados coletados da API, na sua forma desestruturada, e o segundo para a tabulação das informações no formato linhas e colunas para modelagem estatística.

2.2 Conjunto de Dados

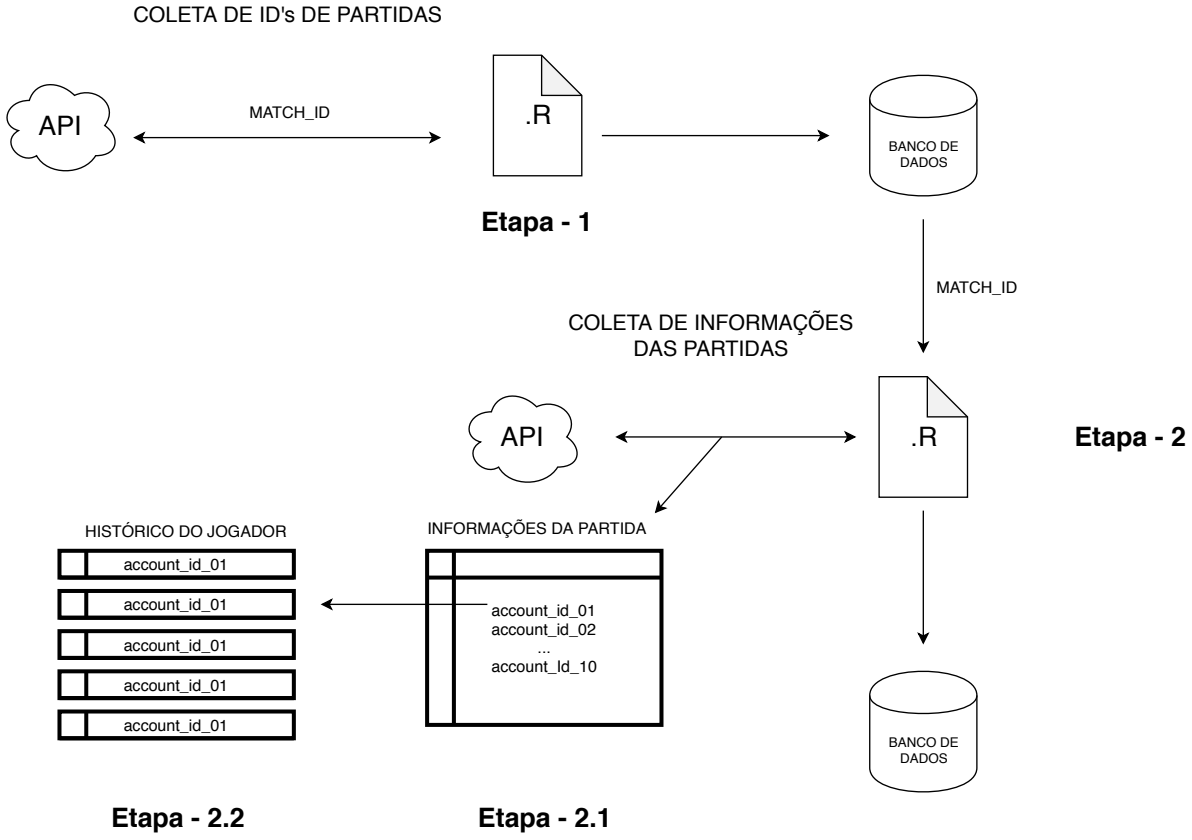
O conjunto de dados foi coletado atráves de uma *Application Programming Interface* (API) fornecida pela plataforma de jogos Steam[4]. Os dados são disponibilizados no formato JSON, que é a forma mais comumente usada para transferir informações via internet, e em seguida armazenados em um banco de dados. No apêndice A é disponibilizado a representação da coleta de uma única partida no formato JSON. O período de coleta dos dados foi de 19 de Fevereiro de 2019 à 28 de Abril de 2019.

O objetivo do estudo é adicionar variáveis do histórico dos jogadores presentes na partida, contudo para realizar tal feito foi necessário o desenvolvimento de algoritmos capazes de coletar essas informações. Para coletar somente uma partida é necessário fazer duas requisições, sendo a primeira para coletar o *match_id*, que é o identificar único de cada partida, no momento em que está está ocorrendo e somente então após um certo período de tempo, (2-3 horas), após o jogo terminar, fazer a requisição das informações finais do jogo. Após a coleta das informações finais de cada partida é preciso coletar o histórico de cada jogador presente nela, para isso é necessário fazer uma requisição para cada jogador das últimas partidas jogadas, armazenar seus respectivos *match_id* para então coletar as suas estatísticas passadas. O fluxograma a

[1] https://github.com/andryas/rdota2plus
[2] https://www.mongodb.com/
[3] https://www.mysql.com/
[4] https://steamcommunity.com/

seguir resume o processo.

Figura 2 – Fluxograma do processo de coleta de dados



Na **etapa - 1** é coletado os identificadores das partidas e armazenados no banco de dados. No outro dia, **etapa - 2**, é feito as requisições dos *match_id* coletados no dia anterior, onde a **etapa - 2.1** representa a coleta da partida e **etapa - 2.2** a coleta do histórico de partidas passadas de um jogador. Uma das principais dificuldades na coleta dos dados consistia na quantidade de requisições feitas à API, de modo que para coletar uma única partida com todo o histórico dos jogadores presentes eram necessários no minímo 112 requisições sendo que nenhum poderia apresentar problemas.

Além disso, na coleta dos dados foi considerado somente partidas com os 13% melhores jogadores, isto é, somente partidas em que estavam presente os melhores jogadores de DotA2, além disso era necessário que todos os jogadores estivessem presentes na partida (min_players: 10), e ainda, que seus ID's fossem públicos para poder ter acesso a seu histórico de partidas passadas. Aplicou-se também um filtro na seleção do tipo de jogo e na duração da partida, isto é, somente jogos rankeados (game_mode: 22 e lobby_type: 7) e partidas com uma duração superior a 15 minutos foram coletadas. A escolha do modo rankeado é justificada pelo fato de que esses tipos de partida são o que elevam a pontuação dos jogadores no rank geral do DotA2.

Abaixo é listado somente as variáveis utilizadas da API no presente trabalho.

- **players** Array com as informações de cada jogador presente na partida.
  - **account_id** ID da conta com 32-bits.
  - **hero_id** ID único do héroi.
  - **kills** A quantidade de mortes atribuídas ao jogador.
  - **deaths** A quantidade de vezes que o jogador morreu durante a partida.
  - **assists** A quantidade de assistencias atribuídas a este jogador.
  - **gold** A quantidade de ouro que o jogador terminou ao final da partida.
  - **last_hits** A quantidade de criaturas finalizadas ao longo da partida.
  - **denies** A quantidade de criaturas negadas ao longo da partida.
  - **gold_per_min** Ouro por minuto do jogador.
  - **xp_per_min** Experiência por minuto do jogador.
  - **gold_spent** Ouro gasto durante a partida
  - **hero_damage** Total de dano causado a heroes inimigos.
  - **tower_damage** Total de dano causado a torres inimigas.
  - **hero_healing** Total de cura dada ao time.
  - **scaled_hero_damage** Total de dano recebido de heroes inimigos.
  - **scaled_tower_damage** Total de dano recebido de torres inimigoss.
  - **scaled_hero_healing** Total de cura recebida de heroes inimigos.
- **radiant_win** Booleano indicando se o time dos Iluminados (VERDADE) venceu ou se o time dos Temidos (FALSO) venceu.
- **duration** Duração da partida em segundos
- **start_time** Timestamp Unix de quando o jogo começou.
- **match_id** ID único da partida.
- **lobby_type**

Tabela 1 – Tipos de lobbies disponíveis

Inválido	-1
Partida pública	0
Prática	1
Torneio	2
Tutorial	3
Partida com bots	4
Partida de times	5
Solo	6
Ranked	7
1vs1 Mid	8

- **humant_players** Quantidade de jogadores humanos presente na partida.
- **game_mode**

Tabela 2 – Tipos de jogos disponíveis

None	0
All pick	1
Captain's mode	2
Random draft	3
Single draft	4
All random	5
Intro	6
Diretide	7
Reverse Captain's Mode	8
The Greevilling	9
Tutorial	10
Only Mid	11
Least Played	12
New Player Pool	13
Compendium Matchmaking	14
Co-op vs Bots	15
Captains Draft	16
Ability Draft	18
All Random Deathmatch	20
1vs1 Only Mid	21
Ranked	22

Na tabulação de dados foi adicionado 5 variáveis, sendo 3 identificadoras e 2 duas atributos resultantes da partida históricas de cada jogador, são elas

- **team** radiant ou dire (times)
- **match_id2** ID único das partidas anteriores ao **match_id**
- **hero_id2** herói selecionado nas partidas anteriores ao **match_id**
- **duration2** duração das partidas anteriores ao **match_id**
- **win_player** resultado, vitória ou derrota atribuida ao jogador, das partidas anteriores ao **match_id**

Por último, foi aplicado dois filtros no conjunto de dados, um considerando somente partidas em que todos os jogadores estavam presentes até o seu termino e outro selecionando somente as partidas em que todos os jogadores tinham um histórico de partidas passadas igual a 10. Para sintetizar os filtros aplicados tem-se a tabela 3.

Tabela 3 – Filtros aplicados durante a coleta e no conjunto final de dados

Descrição do filtro	Argumento na API	Valor correspondente ao filtro na API
Aplicado na coleta		
Nível dos jogadores	skill	3
Quantidade mínima de jogadores	min_players	10
Modo do jogo	game_mode	22
Tipo do jogo	lobby_type	7
Duração miníma da partida	duration	15m
Aplicado no conjunto de dados tabulado		
Conexão do jogador	leaver_status	0
Quantidade mínima de partidas passadas por jogador	-	10

Nota:
O último filtro não corresponde a um argumento da API.

O conjunto de dados final tem 13 milhões de linhas e 31 colunas, totalizando 130 mil partidas únicas. As 6 primeiras linhas do conjunto de dados finais estão na tabela 4.

Por questõe de conveniência os atributos foram renomeados, **kills** = k, **deaths** = d, **assists** = a, **last_hits** = lh, **denies** = de, **gold_per_min** = gpm, **xp_per_min** = xpm, **hero_damage** = hd, **tower_damage** = td, **hero_healing** = hh, **gold** = g, **gold_spent** = gs, **scaled_hero_damage** = shd, **scaled_tower_damage** = std, **scaled_hero_healing** = shh.

Tabela 4 – As 6 primeiras linhas do conjunto de dados

match_id	account_id	team	match_id2	radiant_win	wp	duration	duration2	hero_id	hero_id2
4394396047	110369369	radiant	4293560910	0	0	2299	1412	61	29
4394396047	110369369	radiant	4293605312	0	1	2299	2160	61	100
4394396047	110369369	radiant	4379592998	0	1	2299	2014	61	114
4394396047	110369369	radiant	4380215336	0	0	2299	1939	61	81
4394396047	110369369	radiant	4380307614	0	0	2299	1608	61	94
4394396047	299329752	radiant	4384779519	0	1	2299	3335	26	7

k	d	a	lh	de	gpm	xpm	hd	td	hh	g	gs	shd	std	shh
0	4	0	127	5	333	385	4495	135	0	1966	5965	3912	44	0
18	2	14	309	39	798	785	38258	3482	0	2349	27180	24369	2028	0
11	6	10	180	34	531	731	26887	4503	0	539	17110	17541	2976	0
8	9	4	155	24	434	533	17192	445	0	2390	9790	10166	201	0
3	5	11	147	32	443	389	16208	6792	0	2229	9450	11330	3377	0
2	13	17	138	5	289	431	23176	0	0	2280	11115	9631	0	0

2.3 Especificação do Modelo

Os modelos propostos tem como base um modelo paramétrico de regressão logística que constitui um método de classificação supervisionada e trata-se de um caso particular dos modelos lineares generalizados (MCCULLAGH; JA, 81), definido pela distribuição binomial com função de ligação canônica (logit), usado para a modelar casos onde a variável resposta é binária ou categórica de dois níveis. O modelo é escrito da seguinte forma

$$Y_i|\pi_i \sim Binom(n_i, \pi_i) \tag{2.1}$$

para i de 1 até n e a variável resposta dos modelos propostos está definida como

$$Y_i = \begin{cases} 1, & \text{se o time dos iluminados (radiant) venceu} \\ 0, & \text{caso contrário} \end{cases} \tag{2.2}$$

ou seja, a variável resposta está em função de um time específico, podendo ser interpretada como o time mandante venceu ou perdeu. Reescrevendo a equação 2.1 tem-se

$$Y_i|\pi_i \sim Bernoulli(\pi_i) \tag{2.3}$$

que é um caso particula da distribuição Binomial quando n_i é igual a 1, e π_i é dado por

$$\pi_i = \frac{exp^{(\beta_0+\beta_1.x_{i1}+\beta_2\cdot x_{i2}+\cdots+\beta_p\cdot x_{ip})}}{1+exp^{(\beta_0+\beta_1\cdot x_{i1}+\beta_2\cdot tx_{i2}+\cdots+\beta_p\cdot x_{ip})}} \tag{2.4}$$

aplicando a transformação, que linearaliza a função, tem-se que

$$log(\frac{\pi_i}{1-\pi_i}) = \beta_0 + \beta_1 \cdot x_{i1} + \beta_2 \cdot x_{i2} + +\beta_p \cdot x_{ip} \tag{2.5}$$

que é chamado de função de ligação canônica (logito) associada ao modelo binomial, no caso bernoulli, que mapea os valores de π_i, que pertencem ao intervalo (0,1), para que tenha um correspondente no intervalo $(-\infty, \infty)$, de modo que a média seja uma função linear das covariáveis. Os parâmetros β_j são obtidos atráves do método da máxima verossimilhança dado por

$$L(\beta) = \prod_{i=1}^{n} \pi_i^{y_i}(1-\pi_i)^{1-y_i} \tag{2.6}$$

por restrições computacionais a equação acima é rescrita como

$$l(\beta) = \sum_{i=1}^{n} y_i.log(\pi_i) + (1 - y_i).log(1 - \pi_i) \tag{2.7}$$

e os parâmetros estimados são os que maximizam a função de verossimilhança.

2.3.1 Modelo 1

O primeiro modelo foi proposto pela primeira vez por (CONLEY; PERRY, 2013) utilizando somente a informação dos heróis presentes na partida, na época totalizando 106 heróis, agora, no presente trabalho, 117 heróis. O modelo foi construído com base nos heróis escolhidos por cada time, de modo que se tenha um vetor binário $x \in \mathbb{R}^{234}$, onde

$$x_{i,j} = \begin{cases} 1, & \text{se pertece ao time dos temidos (dire)} \\ 0, & \text{caso contrário.} \end{cases} \tag{2.8}$$

e

$$x_{i,j+117} = \begin{cases} 1, & \text{se pertece ao time dos iluminados (radiant)} \\ 0, & \text{caso contrário} \end{cases} \tag{2.9}$$

O modelo linear em função de π_i fica dado por

$$log(\frac{\pi_i}{1 - \pi_i}) = \beta_0 + \underbrace{\sum_{j=1}^{117} \beta_j.x_{ij}}_{dire} + \underbrace{\sum_{j=118}^{234} \beta_j.x_{ij}}_{radiant} \tag{2.10}$$

2.3.2 Modelo 2

O segundo modelo proposto utiliza as informações do histórico de cada jogador de modo a compor uma estatística média para cada atributo de cada time. O modelo fica expresso da seguinte forma:

$$\bar{x}_{ita} = \frac{\sum_{i=1}^{n} x_{ita}}{n}, \;\; i = 1, ...n; \;\; t = 1, 2 \;\; e \;\; a = 1, ..., A \tag{2.11}$$

onde i é a i-esima partida, t denota o time da i-esima partida e a é o a-eismo atributo.

O preditor linear do quarto modelo em função de π_i fica dado por

$$log(\frac{\pi_i}{1-\pi_i}) = \beta_0 + \underbrace{\sum_{j=1}^{117} \beta_j \cdot x_{ij}}_{dire} + \underbrace{\sum_{j=118}^{234} \beta_j \cdot x_{ij}}_{radiant} + \underbrace{\sum_{a=1}^{A} \beta_a \cdot \bar{x}_{i1a}}_{dire} + \underbrace{\sum_{a=A+1}^{2.A} \beta_a \cdot \bar{x}_{i2a}}_{radiant} \tag{2.12}$$

que representa um modelo preditivo com base na média geral de cada atributo de cada time das n partidas anteriores de cada jogador presente na partida.

2.3.3 Modelo 3

O terceiro modelo proposto é construído da mesma forma que o modelo 2 porém subtrai-se o efeito médio de cada herói para cada atributo em tempos distintos de modo a compor uma observação da "habilidade" do jogador.

Primeiramente as partidas são categorizadas em três tempos, com base na duração de cada uma, que são **inicio_do_jogo**, **meio_do_jogo** e **final_do_jogo**.

$$d_i = \begin{cases} inicio\_do\_jogo, & \text{se a duração da partida for menor ou igual à 25 minutos} \\ meio\_do\_jogo, & \text{se a duração da partida for maior} \\ & \text{que 25 minutos e menor ou igual à 35 minutos} \\ fim\_do\_jogo, & \text{caso contrário} \end{cases} \tag{2.13}$$

Então calcula-se uma estatística média para cada um dos heróis

$$\tilde{x}_{ha} = \frac{\sum_{i=1}^{n} x_{ihat}}{n}, \;\; h = 1, ..., 117; \;\; a = 1, ..., A \;\; e \;\; t = 1, 2, 3. \tag{2.14}$$

é feito então uma nova variável com as informações de cada jogador e a estatística média de cada herói por atributo em cada tempo

$$x^*_{ia} = x_{iha} - \tilde{x}_{iha}, i = 1, ..., n; \;\; a = 1, ..., A \;\; e \;\; h = 1, ..., 117 \tag{2.15}$$

O preditor linear do quarto modelo em função de π_i fica dado por

$$log(\frac{\pi_i}{1-\pi_i}) = \beta_0 + \underbrace{\sum_{j=1}^{117} \beta_j.x_{ij}}_{dire} + \underbrace{\sum_{j=118}^{234} \beta_j.x_{ij}}_{radiant} + \underbrace{\sum_{a=1}^{A} \beta_a.\bar{x}^*_{i1a}}_{dire} + \underbrace{\sum_{a=A+1}^{2*A} \beta_a.\bar{x}^*_{i2a}}_{radiant} \tag{2.16}$$

2.3.4 Modelo 4

O quarto modelo proposto se baseia somente nas informações do histórico de cada time de forma que não é necessário ter informações da partida no seu instante inicial. Para tal, a variável resposta deve-se ser tranformada em função do time em que venceu e não mais em função do time *radiant*. A nova variável resposta fica expressa da seguinte forma

$$Y^* = \begin{cases} \text{vitória}, & \text{se radiant\_win é igual a 1 e o team é igual a radiant} \\ \text{vitória}, & \text{se radiant\_win é igual a 0 e o team é igual a dire} \\ \text{derrota}, & \text{caso contrário} \end{cases} \tag{2.17}$$

O preditor linear do quarto modelo em função de π_i fica dado por

$$log(\frac{\pi_i}{1-\pi_i}) = \beta_0 + \sum_{a=1}^{A} \beta_a \cdot \bar{x}_a \tag{2.18}$$

2.4 Validação Cruzada

A validação cruzada é uma técnica amplamente utilizada para avaliar a capacidade preditiva de um modelo por meio de um processo iterativo de treinamento e teste, utilizando alguma métrica de desempenho. Neste trabalho a métrica de desempenho utilizada é a acurácia, devido a natureza da variável resposta (vitória/derrota), que nada mais é que a quantidade de observações classificadas corretamente sobre o total de observações a serem classificadas. Ou seja, se existem 100 partidas classificadas como vitória ou derrota e dessas 100 somente 70 foram classificadas corretamente, pode-se dizer, neste caso, que 0.7 ou 70% das partidas foram classificadas corretamente.

O processo inicia-se particionando os dados em treino e teste, por exemplo, 90% dos dados para treino e 10% dos dados para teste. É importante salientar que a base inicialmente separada para teste, nesse primeiro momento, não fará parte da validação cruzada própriamente dita, será utilizada como uma medida para mensurar a qualidade do ajuste do método após a validação cruzada.

Após esse primeiro passo, é particionado a base de treino em k partes iguais e utiliza-se $k-1$ partes para treino e a parte de fora como validação. Esse processo é feito até que todas as k partes da base de treino tenham sido utilizadas como validação. Para cada uma das k validações é calculado a acurácia e no final dessa etapa é calculado a acurácia média que será utilizada para comparar com a acurácia da base teste, inicialmente deixada de fora. A acurácia média da validação cruzada é dada por:

$$CV_{(n)} = \frac{1}{n}\sum_{i=1}^{n} Err_i \tag{2.19}$$

onde $Err_i = I(y_i \neq \hat{y}_i)$, que representa a proporção de classificações incorretas.

Se a acurácia média de treino produzir valores muito superiores à base de teste pode-se dizer que possivelmente há presença de sobreajuste, que é um termo para descrever quando um modelo se ajusta muito bem aos dados. O ideal é que a acurácia média produzida na base de treino tenha um desempenho próximo da acurácia na base de teste.

2.5 Curva ROC

Quando se tenta prever o desfecho de uma classificação binária, o problema consiste em determinar o melhor ponto de corte na probabilidade que o modelo atribui ao sucesso de uma particular resposta. Em outras palavras, se um modelo-X atribui probabilidade de 0.4 ao sucesso de uma determinada resposta deve-se classifica-la como sucesso ou como fracaso? Para isso, pode-se utilizar a Curva ROC, do inglês Receiver Operating Characteristic Curve, que é um método amplamente usado para avaliar a capacidade de modelos de classificação, que produzem estimativas de probabilidade.

Abaixo é descrito os quatro tipos de ocorrência que acontecem quando se tenta classificar um determinado fenomeno em apenas duas classes.

- Verdadeiro Positivo (VP): Número de observações positivas classificadas corretamente.
  No contexto do trabalho: Número de partidas classificadas como vitória corretamente.

- Verdadeiro Negativo (VN): Número de observações negativas classificadas corretamente.
  No contexto do trabalho: Número de partidas classificadas como derrota corretamente.

- Falso Positivo (FP): Número de observações negativas classificadas como positivas.
  No contexto do trabalho: Número de partidas classificadas como derrota quando era vitória.

- Falso Negativo (FN): Número de observações positivas classificadas como negativas.
  No contexto do trabalho: Número de partidas classificadas como vitória quando era derrota.

A partir das ocorrências apresentadas acima pode-se então calcular as duas métricias utilizadas para a construção da Curva ROC, são elas:

- Sensibilidade: Proporção de verdadeiros positivos.
  No contexto do trabalho: Proporção de partidas classificadas como vitória quando de fato era vitória.

$$Sensibilidade = \frac{VP}{VP + FN} \tag{2.20}$$

- Especificidade: Proporção de verdadeiros negativos.
  No contexto do trabalho: Proporção de partidas classificadas como derrota quando de fato era derrota.

$$Especificidade = \frac{VN}{VN + FP} \tag{2.21}$$

Assim, variando o ponto de corte e calculando as métricas apresentadas acima iterativamente pode-se contruir um gráfico com a Sensibilidade no $eixo - y$ e, 1 - Especificidade no $eixo - x$. Desse modo o ponto que apresentar a maior Sensibilidade e Especificidade conjuntamente será considerado o melhor ponto de corte que discrimina sucessos de fracassos, ou vitórias de derrotas.

3 Resultados e Discussão

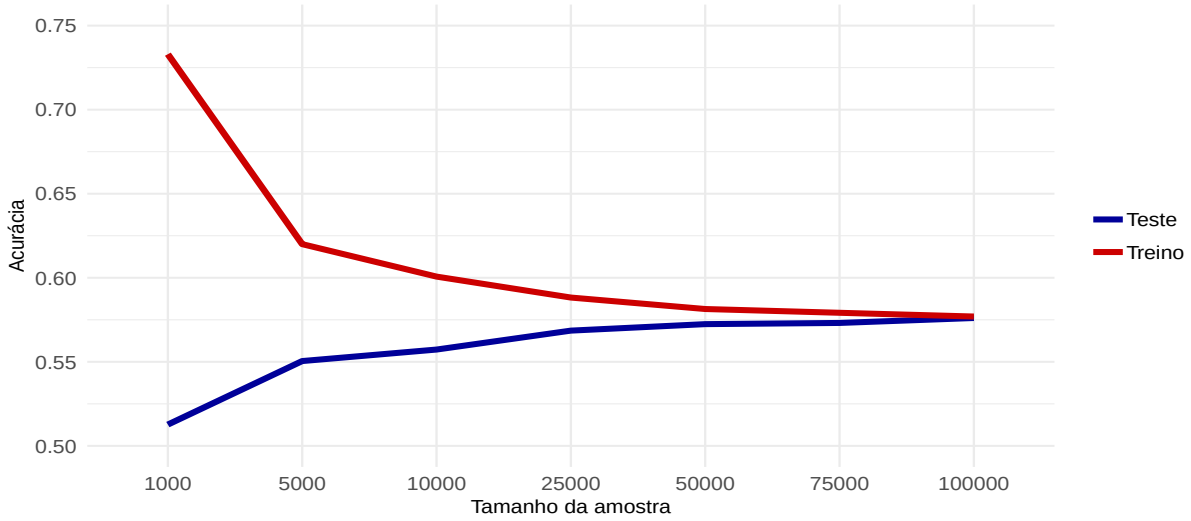
Todos modelos apresentados a seguir foram ajustados utilizando o pacote *caret* com 5 validações cruzadas e o ponte corte, para classificação entre vitória/derrota, definido pela curva ROC de modo que se maximizasse a sensibilidade e especificidade conjuntamente. As bases foram separadas em treino e teste de forma que para o treino tivesse k-fold nos 90% dos dados e para teste 10% dos dados. Para verificar os resultados utilizou-se curvas de aprendizado onde no eixo-x consta o tamanho da amostra e o no eixo-y a acurácia do modelo. O desempenho da base de treino está representado em vermelho e o da base de teste, a qual não utilizou-se para ajuste do modelo, em azul.

3.1 Modelo 1

Para o primeiro modelo foi necessário balancear as classes pois a variável resposta apresentava uma proporção de 0.4554 de derrotas e 0.5446 de vitórias, o conjunto final de dados ficou então com 119718 linhas por 236 colunas, sendo uma coluna a variável resposta e a outra uma coluna identificadora.

O primeiro modelo ajustado teve uma acurácia de 57.6% na base de teste para o tamanhos de amostra de 100000, no gráfico 3 tem-se o resultado do modelo.

Figura 3 – Curva de aprendizado para validação cruzada para o modelo 1



Fonte o autor

Nota-se que os resultados foram bem diferentes daqueles obtidos por outros

autores citados anteriormente. Diversos motivos podem ter causado essa diferença observada, abaixo é listado alguns

1. Introdução de um assistente de desempenho (DotaPlus) (12/03/2018)
2. Mudanças radicais na arena do jogo, disposição da floresta, disposição dos campos de criaturas da floresta, posição do rocha etc...
3. Atualizações mais frequentes nas habilidades e atributos dos heróis, o que resulta em parâmetros mudando ao longo do tempo.
4. Novos heróis
5. Competividade do jogo aumentou

O primeiro ponto, introdução de um assistente, pode mudar completamente o comportamento dos jogadores. As três principais funções do asssiste é auxiliar na escolha dos heróis no inicio de cada jogo, fornecer estatísticas de desempenho ao longo do jogo e sugestões de itens, e análises individuais de desempenho com cada herói. Com isso o jogador estará muito mais atento as suas fragilidades em tempo real, podendo então tentar compensar de alguma maneira essas debilidades, além de dispor de informações extras sobre seu desempenho, e como o público alvo desse trabalho foi os melhores jogadores de DotA2, é provável que todos eles tenham esse assistente, deixando assim o jogo mais competitivo.

O segundo, terceiro e quarto pontos modificam principalmente a estratégia "padrão" do jogo, ou seja, o que antes, devido as poucas atualizações, era algo mais estatístico, agora tornou-se mais dinâmico fazendo com que algo seja uma vantagem nesta semana, passe a causar menos impacto na semana seguinte.

O quinto tem uma implicação dos pontos citados anteriormente e também o constante crescimento do jogo devido à valorização do esporte no cenário competitivo, e que aliado ao surgimento de diversas plataformas de apostas podem ter causado um maior incentivo a jogar o jogo, não somente como entreterimento mas de forma profissional.

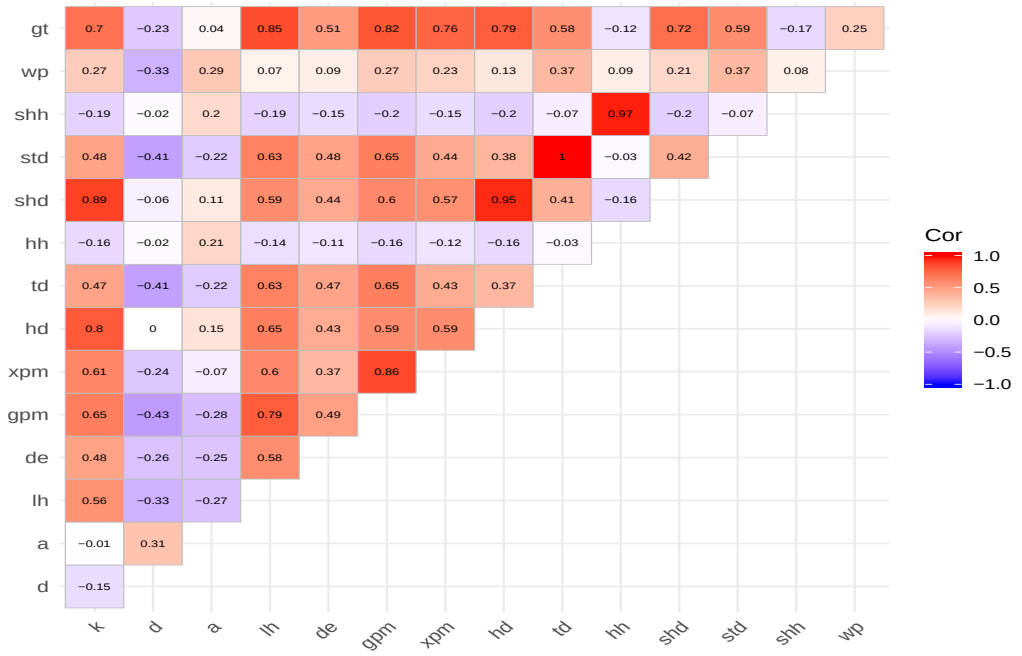
3.2 Modelo 2

Para a construção do segundo modelo fez-se necessário aplicação de mais um filtro, pois, os seguintes atributos atributos *hero_damage* (hd), *tower_damage* (td), *hero_healing* (hh), *gold* (g), *gold_spent* (gs), *scaled_hero_damage* (shd), *scaled_tower_damage* (std) e *scaled_hero_healing* (shh) apresentaram valores ausentes em alguma das partidas.

O filtro aplicado removeu os jogos em que pelo menos um jogador na partida apresentou mais de 5 valores ausentes para alguns dos atributos citados anteriormente, também foi criado uma nova variável, *gold_total* (gt), a partir da soma das variáveis *gold* (g) e *gold_spent* (gs), sendo que *gold* (g) é o total de ouro em que o jogador tinha quando a partida terminou e *gold_spent* (gs) é o total de ouro gasto durante a partida. Após esses procedimentos os dados foram sumarizados a nível de time e balanceados, o conjunto final de dados, ficou com 70724 observações.

Antes do ajuste do modelo foi feito a matriz de correlação para a análise das correlaçõs entre os atributos.

Figura 4 – Matriz de correlação das variáveis

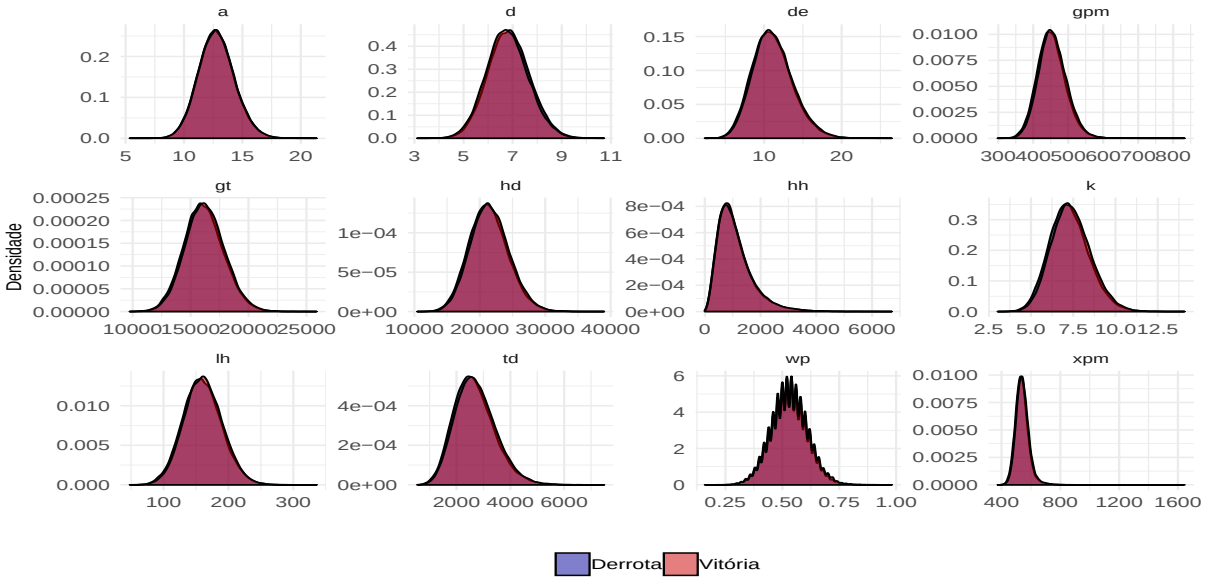


Fonte o autor

A partir da matriz de correlação constata-se correlações altas entre as variáveis *hero_healing* (hh) com *scaled_hero_healing* (shh), *hero_damage* (hd) e *scaled_hero_damage* (shd), *tower_damage* (td) e *scaled_tower_damage* (std), e portanto opta-se em remover as variáveis *scaled_hero_healing* (shh), *scaled_hero_damage* (shd) e *scaled_tower_damage* (std).

Em seguida é feito um gráfico 5 constratando a distribuição empírica dos vencedores com a dos perdedores para cada atributo.

Figura 5 – Distribuição empírica dos vencedores/perdedores para cada atributo

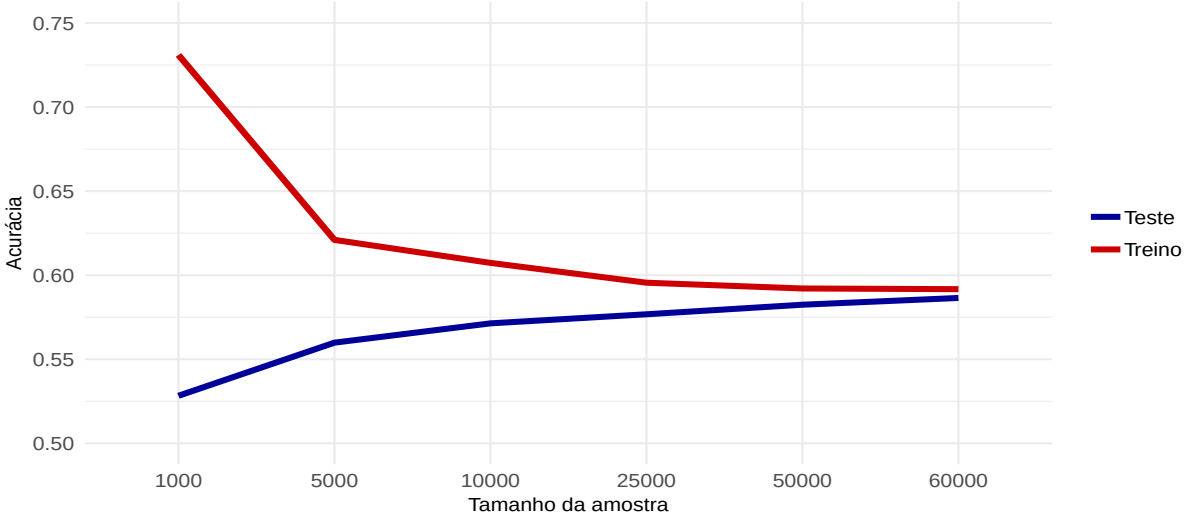


Fonte o autor

Primeiramente observa-se que as distribuições empíricas dos vencedores com a dos perdedores estão sobrepostas para todos os atributos, ou seja, individualmente nenhum atributo tem um poder discriminativo relevante.

Por último é feito o ajuste do modelo, que está no gráfico 6, e observa-se que não houve um ganho de performance substancial. A acurácia na base de teste para uma base de treino de 50000 foi de 58.24%.

Figura 6 – Curva de aprendizado para validação cruzada para o modelo 2

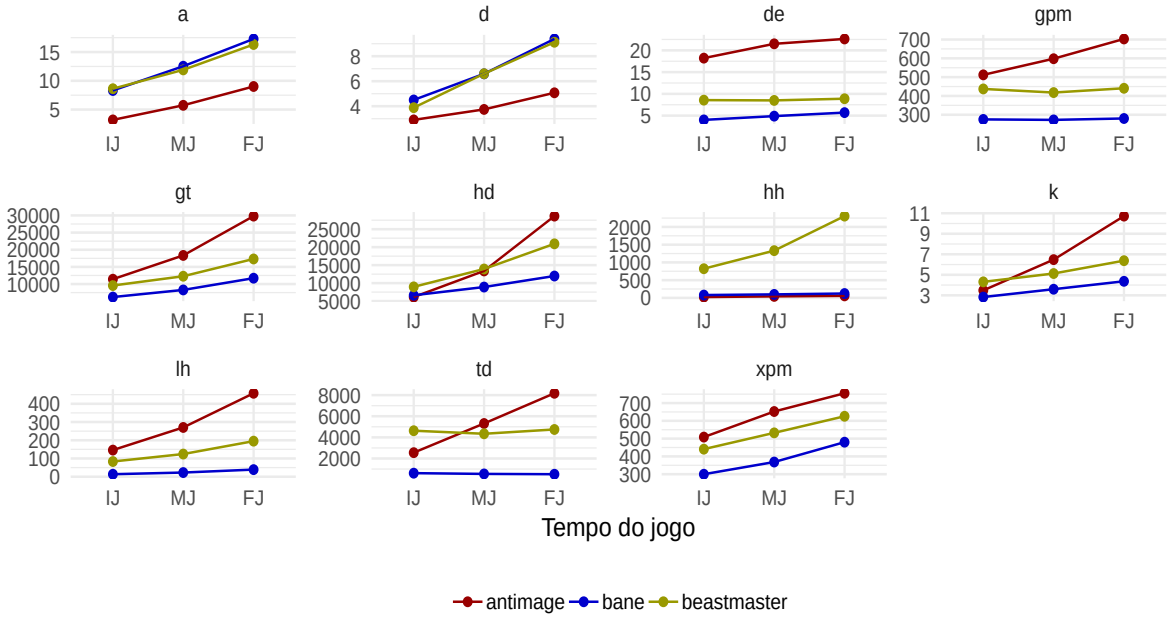


Fonte o autor

3.3 Modelo 3

A construção do terceiro modelo foi feito da mesma forma que o modelo 2 exceto por uma correção nas observações antes da análise. O objetivo dessa correção é a de minimizar a relação entre o atributo-herói-tempo buscando extrair somente o desempenho do jogador e consequentemente ter uma medida mais realista do desempenho médio do time. Para exemplificar o seguinte gráfico foi feito escolhendo três heróis que tem diferentes funções no jogo, sendo eles *antimage*, *bane* e *beastmaster*.

Figura 7 – Atributo médio por três diferentes heróis

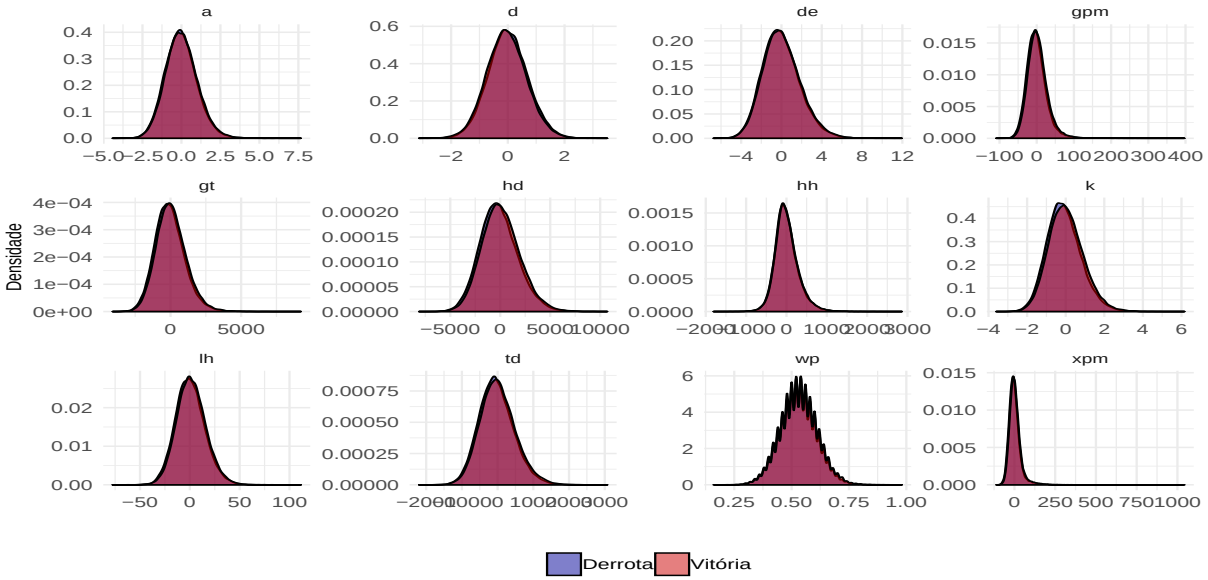


Fonte o autor

Nota-se pelo gráfico 7 que existe uma diferença significativa quanto a estatística média de cada atributo para cada herói. Desse modo tem-se que quando se calcula somente a média de um atributo que advém de diferentes jogadores com diferentes heróis e que obtiveram diferentes resultados a habilidade do jogador, e consequentemente do time, acaba se tornando menos realista.

Assim, as observações de cada jogador são corrigidas e aplicado os mesmos procedimentos que no modelo 2, obtem-se então o seguinte gráfico dos vitoriosos versus derrotados para cada um dos atributos.

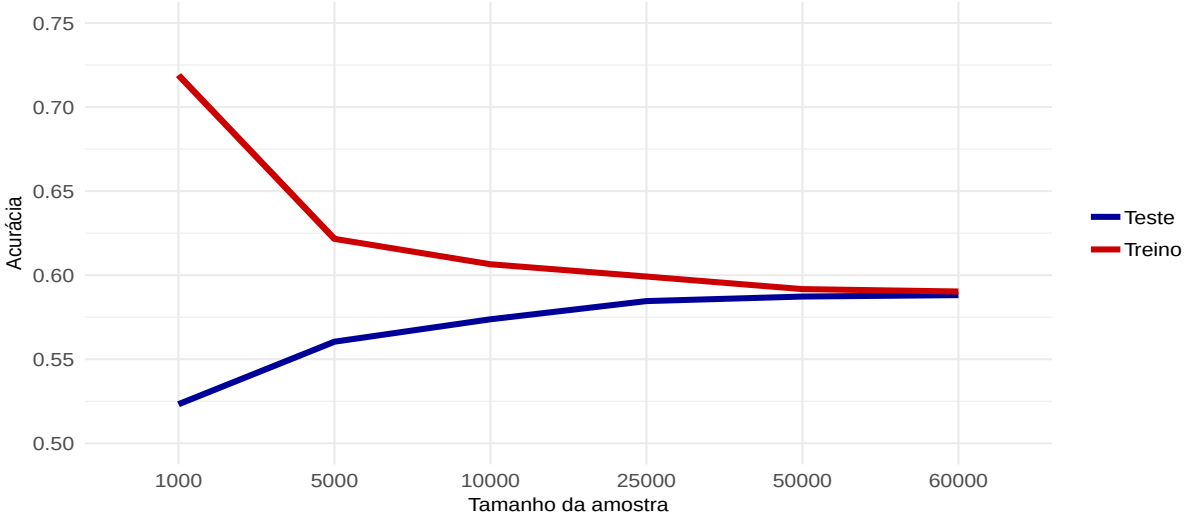
Figura 8 – Distribuição empírica dos vencedores/perdedores para cada atributo



Fonte o autor

Pelo gráfico 8 parece que as distribuições empíricas dos vencedores com a dos perdedores não difere dos resultados obtidos anteriormente. A curva de aprendizado do modelo é exibida abaixo, tendo uma acurácia de 58.72% na base de teste para uma amostra de 50000.

Figura 9 – Curva de aprendizado para validação cruzada para o modelo 3

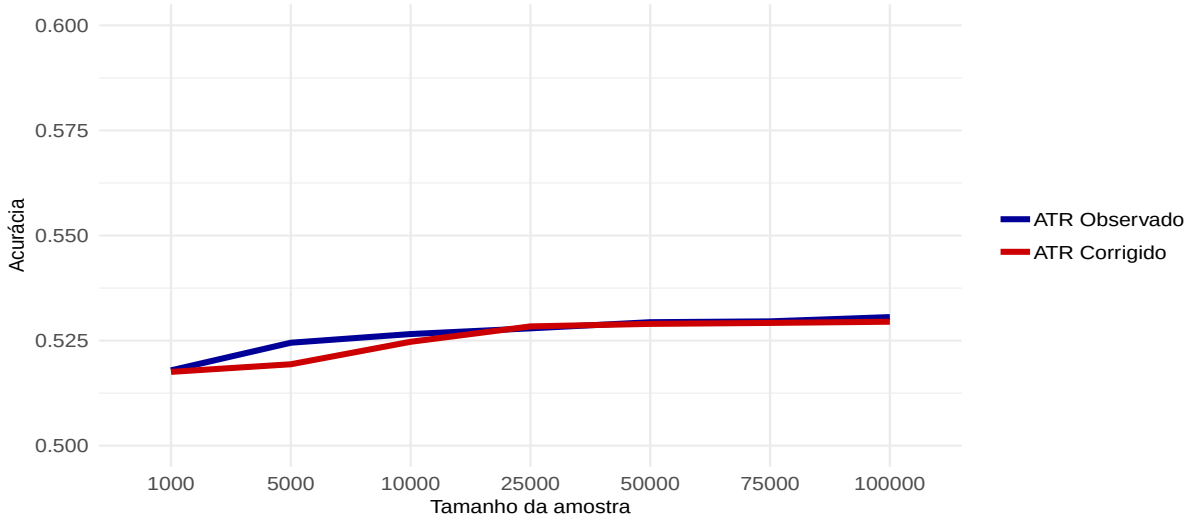


Fonte o autor

3.4 Modelo 4

O quarto modelo proposto é construído a partir do segundo modelo porém removendo as informações dos heróis e transformando a variável resposta em função de cada time, de modo a se ter linhas independentes ao invez de colunas. Para fins de comparação entre os atributos observados e os atributos corrigidos ambas as curvas de aprendizado para a base de teste são feitas no gráfico abaixo. O resultado da acurácia foi de 53.05% para os atributos brutos e 52.94% para os atributos corrigidos com uma base de treino de 100000.

Figura 10 – Curva de aprendizado para validação cruzada para o modelo 4



Fonte o autor

4 Considerações Finais

Embora o resultado do primeiro modelo não foi condizente com os de outros autores pode-se verificar ainda assim que a escolha dos heróis, e portanto, a composição do time algum impacto no desfecho final da partida. Além disso, os resultados obtidos pelos demais modelos demonstram que o acréscimo do desempenho médio de cada atributo por time tem um efeito minímo se não nulo na predição do resultado final. Um dos princpais motivos para os resultados obtidos, em relação aos atributos, é de que as informações disponíveis são resultados finais de partidas e devido o aumento da complexidade do jogo, talvez, seja necessário o uso de informações de desempenho ao longo da partida e não somente as estatísticas geradas no final do jogo.

Além disso vale lembrar que os dados foram coletados de partidas em que os times eram formados aleátoriamente, ou seja, pode-se concluir que o algoritmo utilizado para formação dos times é justo para o grupo dos melhores jogadores de DotA2.

Para trabalhos futuros recomenda-se o uso de estatísticas geradas durante a partida de forma que se possa extrair características de desempenho dos jogadores em diferentes momentos do jogo. Ademais, processar somente informações de jogos entre dois times profissionais, ou seja, somente partidas em que ambos os times jogam como um grupo profissional, pois como vimos no presente trabalho times formados aleatoriamente não são diferenciavéis quanto suas habilidades.

REFERÊNCIAS

BERNSTEIN, P. L. *Desafio aos deuses: a fascinante história do risco*. [S.l.]: Gulf Professional Publishing, 1997. Citado na página 13.

CALVO, T. B. Predição de partidas de dota2 via machine learning. 2017. Citado 2 vezes nas páginas 13 e 14.

CONLEY, K.; PERRY, D. How does he saw me? a recommendation engine for picking heroes in dota 2. *Np, nd Web*, v. 7, 2013. Citado 2 vezes nas páginas 14 e 24.

MCCULLAGH, P.; JA. Nelder. 1989. generalized linear models. *London, chapman hall*, 81. Citado na página 23.

R Core Team. *R: A Language and Environment for Statistical Computing*. Vienna, Austria, 2019. Disponível em: <https://www.R-project.org/>. Citado na página 17.

SILVA, V. d. N.; CHAIMOWICZ, L. Moba: a new arena for game ai. *arXiv preprint arXiv:1705.10443*, 2017. Citado na página 13.

Apêndices

APÊNDICE A – Dados no formato JSON

Listing A.1 – match


```
[
  {
    "players": ...,
    "radiant_win": false,
    "duration": 1869,
    "pre_game_duration": 90,
    "start_time": 1549408252,
    "match_id": 4394571998,
    "match_seq_num": 3799941136,
    "tower_status_radiant": 390,
    "tower_status_dire": 1974,
    "barracks_status_radiant": 51,
    "barracks_status_dire": 63,
    "cluster": 184,
    "first_blood_time": 97,
    "lobby_type": 7,
    "human_players": 10,
    "leagueid": 0,
    "positive_votes": 0,
    "negative_votes": 0,
    "game_mode": 22,
    "flags": 1,
    "engine": 1,
    "radiant_score": 19,
    "dire_score": 36,
    "picks_bans": [
      {
        "is_pick": false,
        "hero_id": 8,
        "team": 0,
        "order": 0
      },
      {
        "is_pick": false,
        "hero_id": 44,
        "team": 0,
        "order": 1
      }
    ]
  }
]
```

```
[{
  "players": [
  {
    "account_id": 292190898,
    "player_slot": 0,
    "hero_id": 54,
    "item_0": 50,
    "item_1": 151,
    "item_2": 11,
    "item_3": 252,
    "item_4": 112,
    "item_5": 36,
    "backpack_0": 0,
    "backpack_1": 0,
    "backpack_2": 0,
    "kills": 2,
    "deaths": 5,
    "assists": 6,
    "leaver_status": 0,
    "last_hits": 201,
    "denies": 28,
    "gold_per_min": 440,
    "xp_per_min": 462,
    "level": 18,
    "hero_damage": 16096,
    "tower_damage": 2198,
    "hero_healing": 1040,
    "gold": 524,
    "gold_spent": 13110,
    "scaled_hero_damage": 9318,
    "scaled_tower_damage": 1216,
    "scaled_hero_healing": 603,
    "ability_upgrades":
    {
      "ability": 5250,
      "time": 284,
      "level": 1
    },
    {
      "ability": 5251,
      "time": 395,
      "level": 2
    },
    {
      "ability": 5250,
      "time": 490,
      "level": 3
    },
    {
      "ability": 5249,
      "time": 566,
      "level": 4
    },
    {
      "ability": 5250,
      "time": 688,
      "level": 5
    },
    ...
  },
  ...
  ]
```


Listing A.2 – "players"


```
[{
  ...,
  {
    "ability": 5252,
    "time": 769,
    "level": 6
  },
  {
    "ability": 5249,
    "time": 846,
    "level": 7
  },
  {
    "ability": 5250,
    "time": 975,
    "level": 8
  },
  {
    "ability": 5251,
    "time": 1067,
    "level": 9
  },
  {
    "ability": 5906,
    "time": 1158,
    "level": 10
  },
  {
    "ability": 5249,
    "time": 1202,
    "level": 11
  },
  {
    "ability": 5253,
    "time": 1333,
    "level": 12
  },
  {
    "ability": 5249,
    "time": 1438,
    "level": 13
  },
  {
    "ability": 5251,
    "time": 1515,
    "level": 14
  },
  {
    "ability": 5939,
    "time": 1718,
    "level": 15
  },
  {
    "ability": 5251,
    "time": 1860,
    "level": 16
  },
  {
    "ability": 5252,
    "time": 2169,
    "level": 17
  }
  ]
}
```


Listing A.3 – "players.ability_upgrades"