UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Câmpus de Presidente Prudente

Teodoro Balbino Calvo

Predição de Partidas de Dota2 via *Machine Learning*

Revisado pelo(a)
Orientador(a)

Assinatura do(a)
Orientador(a)

PRESIDENTE PRUDENTE
2017

Teodoro Balbino Calvo

Predição de Partidas de Dota2 via *Machine Learning*

Relatório Final para Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Estatística da FCT/Unesp para aproveitamento na disciplina Trabalho de Conclusão de Curso.

Orientador(a): Prof(a). Dr(a). Manoel Ivanildo Silvestre Bezerra.

PRESIDENTE PRUDENTE

2017

Agradecimentos

Ser meticuloso, calculista e demasiadamente amoroso é este meu Deus, merecedor de todo agradecimento. Sempre refrigerando minha alma e auxiliando-me nas (difíceis) tomadas de decisão que tenho enfrentado, obrigado!

Natália Ataide, minha companheira que me dá forças e motivação desde o início de nossa caminhada. Sem você este trabalho estaria longe de ser o que é. Suas contribuições foram importantíssimas até aqui. Kira merece agradecimentos especiais por adentrar recentemente à família, contagiando-nos com sua alegria e afetos diários.

Há aqueles que têm me apoiado desde a pequena idade, não se esquivando dos problemas que surgiram ao decorrer dos tempos: meus pais, avós e minha querida irmã. Todo sustento até aqui foram fundamentais para esta jornada ser concretizada. É meu desejo retribuir isso com inúmeras conquistas e alegrias.

Dentre tantos amigos e companheiros de jornada, duas pessoas desempenharam papéis imprescindíveis: Adriana Silva e Marcelo Hartmann. Ambos são merecedores daquilo que procuram no mundo. Estão sempre dispostos a contribuir sem interesse algum em receber algo em troca, têm meu carinho.

A liberdade de escolha que tive desde a definição do tema, técnicas aplicadas e elaboração do texto como um todo, só foram possíveis graças ao meu orientador, Manoel Bezerra. Obrigado pela confiança e orientação de meu trabalho.

Há mais uma infinidade de pessoas que sou grato pela convivência, ajuda e força nos tempos difíceis. À todos estes, fica meu singelo agradecimento, admiração e torcida pelo sucesso pessoal de cada um.

O homem só conhecerá a verdadeira liberdade, individual e coletiva, quando se libertar da autoridade e de sua fé nela. A evolução humana nada mais é que uma penosa caminhada nessa direção.
O desenvolvimento não é em si nem invenção, nem a técnica. Correr a 150 quilômetros por hora não é prova de civilização. É pelo indivíduo, autêntico modelo social, que se mede nosso grau de civilização; é por suas faculdades individuais, pelas possibilidades de ele ser livremente o que é, de desenvolver-se e progredir sem intervenção de autoridade coercitiva e onipotente.

O indivíduo, a sociedade e o Estado, *Emma Goldman*

Resumo

Técnicas de aprendizado de máquina destacam-se pela versatilidade empregada em muitos âmbitos de negócios que envolvem problemas de predição e caracterização de dados. O presente trabalho aborda dois modelos de classificação utilizando Redes Bayesianas como enfoque, técnica cada dia mais requisitada e estudada no cenário de aprendizado de máquina e "BigData". O primeiro modelo, *Naïve*-Bayes, considera independência entre todas covariáveis do estudo, o que raramente pode ser comprovado. *Tree augmented Naïve*-Bayes (TAN), segundo modelo, possibilita definições de probabilidade condicionais entre as covariáveis. Ambos modelos são empregados para predição de partidas de Dota2, em que foi realizado se coletas de dados periódicas ao utilizar a API disponível pelo fabricante do jogo. As performances dos dois modelos foram aceitáveis para predições futuras, entretanto, é notório o desempenho superior do modelo TAN em comparação do modelo *Naïve*-Bayes.

Abstract

Machine learning techniques have been in increasingly focus of attention due to its versatility throughout many fields of business and big-data scenarios which involves prediction and classification (characterization) tasks. In this work we present two models for classification based on Bayesian networks. The first model, the näive Bayes, henceforth NB, assumes independency between the covariates involved in the study, which is a strong assumption and rarely seen in practical applications. The second model Tree Augmented Naïve Bayes (TAN), allows a more flexible approach by creating a link between covariates through conditional probabilities. We apply both models in a real study case scenario involving Dota2 matches. The performance of the first model is as expected but the second model shows surprising results.

Sumário

1 Introdução **5**

1.1 Estrutura do trabalho 6

2 Métodos Probabilísticos para Classificação **8**

2.1 Tópicos da Teoria de Probabilidade 8

2.2 Paradigma Bayesiano 10

2.3 Classificador Bayesiano 12

2.4 *Naïve*-Bayes 15

2.5 *Tree Augmented Naïve* Bayes (TAN) 19

2.6 Sugestões pré modelagem 23

3 Métodos para validação de ajuste de modelos **24**

3.1 Validação cruzada k-partições 24

3.2 Análise ROC 26

3.3 Considerações sobre métricas de desempenho 29

4 Aplicação ao Dota2 **30**

4.1 O jogo 30

4.2 Coleta de dados 31

4.3 Modelagem 32

4.3.1 Modelagem *Naïve*-Bayes 32

4.3.2 Modelagem *Modelagem Tree Augmented Naïve* Bayes (TAN) 36

4.3.3 Comparação entre modelos 39

5 Considerações finais **43**

Capítulo 1

Introdução

A indústria de jogos eletrônicos tem evoluído rapidamente, uma atividade que até pouco tempo era vista como diversão e passatempo de crianças e jovens, torna-se algo muito mais sério. O número de jogadores profissionais neste meio tem aumentado de modo nunca visto. Tais fatos são comprovados pela crescente divulgação de torneios mundiais e grande quantidade de times profissionais participantes. Um termo que se ouve com maior frequência é "*eSports*"(Esporte Eletrônico), já o próprio nome faz referência aos jogos eletrônicos, porém apresentando-os como verdadeiro esporte que possui: seus atletas profissionais, disputas acirradas em nível mundial, equipes favoritas e uma crescente audiência. Nota-se também o aumento significativo dos valores das premiações dos campeonatos promovidos neste meio, encontrando-se na casa de milhões de dólares.

Dota2 é um jogo eletrônico com impacto mundial. A empresa desenvolvedora, *Valve*, disponibiliza publicamente os dados de partidas históricas, incentivando estudos relacionados à Dota2 em diversos âmbitos de pesquisa. O presente trabalho realiza uma abordagem de aprendizado de máquina para predição de suas partidas. Ao utilizar dados históricos, é possível criar modelos de predição baseados em probabilidade, prevendo com precisão estatística qual equipe será vencedora de determinada partida.

A utilização de métodos de Aprendizado de Máquina (AM), do inglês *Machine Learning*, tem permitido resolver problemas cada vez mais complexos. Embora sua fundamentação teórica não seja tão recente, o avanço da tecnologia permitiu tanto sua aplicação quanto escalabilidade (Faceki et al., 2015). Aborda-se duas técnias de AM no presente trabalho: *Naïve*-Bayes (NB) e *Tree Augmented Naïve Bayes Network* (TAN) (Friedman et al., 1997). Ambas contempladas pela teoria de Redes Bayesianas para classificação, que

a partir de variáveis auxiliares atualiza a probabilidade de determinada classe (categoria, nível, evento) da variável resposta por meio do Teorema de Bayes.

Entende-se que técnicas estatísticas, aprendizado de máquina e conhecimento do jogo Dota2, sejam insumos necessários para realização da tarefa de criar modelos de previsão de partidas de Dota2. No primeiro momento, um modelo que utilize apenas as informações de quais personagens estão presentes nas partidas. Já em um segundo plano, estende-se os conceitos para atribuir os desenvolvimentos respectivos de cada personagem durante a partida, isto é, "ouro por minuto" e/ou "experiência por minuto". Acredita-se que aumentando o nível de sofisticação os resultados sejam mais satisfatórios e a margem de erro em predições diminua significativamente.

A ferramenta gerada a partir do presente trabalho poderá fornecer informações relevantes em tempo real para equipes no cenário competitivo, bem como para o crescente nicho de apostadores de desfechos de partidas de Dota2.

1.1 Estrutura do trabalho

Dando continuidade no trabalho e para melhor situar o leitor, esboça-se a estrutura do trabalho desenvolvido e uma breve síntese dos assuntos em cada capítulo:

Capítulo 2

Aborda-se toda a teoria utilizada para a aplicação dos modelos. Expõem-se desde os conceitos básicos de teoria de probabilidade, paradigma Bayesiano e os modelos de classificação propostos. A inferência dos parâmetros dos modelos utilizados também são abordados nesta sessão.

Capítulo 3

Métodos de validação de ajustes de modelos são indispensáveis no processo de modelagem. Tais métodos fornecem informações a respeito da performance de determinado modelo para explicar a variabilidade dos dados. Dois métodos são abordados: Validação cruzada k-partições e análise ROC.

Capítulo 4

Após os conceitos teóricos apresentados, parte-se para abordagem prática do problema, isto é, ajustar os modelos propostos sob a base de dados coletados de Dota2. Neste capítulo, utiliza-se também as técnicas de validação de ajuste para verificar a adequabilidade dos modelos.

Capítulo 5

Durante o desenvolvimento de todo o trabalho, expõem-se os resultados obtidos. Esta sessão sintetiza toda informação gerada durante o estudo, havendo ainda o reconhecimento da necessidade da criação de próximos passos, aprimorando os conhecimentos adquiridos.

Capítulo 2

Métodos Probabilísticos para Classificação

Neste capítulo apresentaremos a formulação teórica de dois modelos de classificação de redes Bayesianas, suas respectivas propriedades e singularidades. Entretanto é necessário abordar primeiramente tópicos da teoria de probabilidade: axiomas de probabilidade, probabilidade condicional, probabilidade conjunta e o Teorema de Bayes.

2.1 Tópicos da Teoria de Probabilidade

Tendo em mente a realização de um experimento qualquer, denotamos o conjunto de seus possíveis resultados como *espaço amostral* (Ω). Para um possível resultado em particular, da-se o nome de *evento* (E). Logo, a probabilidade P do evento E ocorrer é denotada por $P(E)$, onde a mesma deve satisfazer os seguintes axiomas (Sheldon, 2010):

- $0 \leqslant P(E) \leqslant 1$;
- $P(\Omega) = 1$;
- Para todos eventos E_1, E_2... mutuamente exclusivos temos: $P(\bigcup_{i=1}^{\infty} E_i) = \sum_{i=1}^{\infty} P(E_i)$

Como vimos no primeiro axioma, toda e qualquer probabilidade está definida entre os valores 0 e 1 (inclusive ambos), representando a nossa crença em relação a ocorrência de determinado evento. Quanto mais próximo de 0, menor nossa convicção da ocorrência do evento. Já quando este valor se aproxima de 1, somos mais convictos em relação a ocorrência do evento.

Probabilidade Condicional

Probabilidade condicional é a probabilidade baseada em alguma informação previamente conhecida. Isto é, predizer a respeito do evento A, quando já sabemos a ocorrência do evento B. Denotaremos por $P(A|B)$, leia-se probabilidade do evento A dado o evento B. Dizemos ainda que, caso $P(A|B) = P(A)$ então A e B são eventos independentes.

Probabilidade Conjunta

A probabilidade conjunta é a probabilidade de dois ou mais eventos ocorrerem simultaneamente. Num caso particular de apenas dois eventos, seguimos com a notação $P(A \cap B)$. Pode-se obter esta probabilidade a partir de:

$$P(A \cap B) = P(A)P(B|A)$$

Note que, se e somente se A e B independentes, temos:

$$P(A \cap B) = P(A)P(B|A) = P(A)P(B)$$

Teorema de Bayes

A partir dos resultados anteriores, temos condições suficientes para calcular $P(A|B)$, isto é, a probabilidade condicional de A em relação a B. Os passos seguintes demonstram como chegamos até a formulação do Teorema de Bayes:

$$P(A \cap B) = P(B)P(A|B)$$

Realizando operações algébricas necessárias:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

Como visto anteriormente, $P(A \cap B) = P(A)P(B|A)$. Assim:

$$P(A|B) = \frac{P(A)P(B|A)}{P(B)} \tag{2.1}$$

2.2 Paradigma Bayesiano

Afim de apresentar a importância do Teorema de Bayes na aplicação do presente trabalho, o examinaremos de maneira mais detalhada nesta seção.

Utilizaremos uma abordagem menos convencional, apresentando uma interpretação baseada em hipóteses e fatos observados. A ideia central neste tipo de abordagem consiste em frisar que as probabilidades de nossas hipóteses iniciais se alteram dado o surgimento de novas evidências (dados). Desta forma, reescreveremos o Teorema de Bayes da seguinte forma:

$$P(H|D) = \frac{P(H)P(D|H)}{P(D)} \tag{2.2}$$

onde H é a hipótese, e D são as evidências. Assim, temos a descrição de cada fator da equação (2.2):

- $P(H)$ é a probabilidade da hipótese antes mesmo do surgimento de qualquer dado, isto é, probabilidade a priori, ou apenas *priori*.
- $P(H|D)$ é a probabilidade que se deseja calcular, sendo esta, uma atualização da probabilidade de nossa hipótese após observar os dados, *posteriori*.
- $P(D|H)$ é a probabilidade dos dados terem ocorrido da forma observada, sob a hipótese H, *verossimilhança*.
- $P(D)$ é a probabilidade dos dados terem ocorrido sob quaisquer hipóteses, *constante normalizadora*.

Introduzimos a seguir um exemplo amplamente conhecido para melhor ilustrar a aplicabilidade dos conceitos expostos: o problema de Monty Hall, originalmente proposto no show de televisão *"Let's Make a Deal"* (Downey, 2012).

Exemplo 2.2.1. José está em um programa de televisão e lhe é apresentado 3 portas fechadas: A, B, C. Por detrás de uma dessas portas há um grande prêmio, já nas outras duas, não há nada. É solicitado que José escolha uma das portas, suponha que seja escolhida a porta A. Em seguida, o apresentador abre a porta B, onde não há nada. Novamente é solicitada outra tomada de decisão á José: permanecer com sua escolha a

priori, porta A, ou trocar para porta C. A pergunta que devemos fazer é se José trocar de porta, suas chances de ganhar o prêmio aumentam, diminuem, ou se mantêm?

É comum pensar que para esta nova escolha as probabilidades de acertar a porta com o grande prêmio é igual a $1/2$, uma vez que há apenas duas portas. Entretanto, utilizaremos o Teorema de Bayes para contrariar isto. Assim, elencamos três hipóteses essenciais para dar início à resolução do problema:

- H_1: O grande prêmio está atrás da porta A;
- H_2: O grande prêmio está atrás da porta B;
- H_3: O grande prêmio está atrás da porta C.

É razoável atribuir probabilidades iguais para cada uma das hipóteses a priori, isto é:

$$P(H_1) = P(H_2) = P(H_3) = 1/3$$

Avançamos para o próximo passo, definir o evento D: "revelou-se a porta B" e calcular $P(D|H_i)$ para todo $i = 1, 2, 3$, isto é, calcular a probabilidade do apresentador abrir a porta B dado cada uma das hipóteses iniciais.

Para $i = 1$:

Supondo que H_1 seja a hipótese verdadeira, o apresentador tem liberdade total para escolher quaisquer das duas outras portas para abrir, pois ambas estão vazias. Assim, a probabilidade da porta B ter sido revelada é $1/2$. Ou seja, $P(D|H_1) = 1/2$.

Para $i = 2$:

Supondo que H_2 seja a hipótese verdadeira, não haveria outra escolha para o apresentador senão revelar a porta C, uma vez que o grande prêmio está atrás da porta B. Assim, a probabilidade da porta B ter sido revelada, quando o prêmio se encontra na mesma, é igual a 0. Ou seja, $P(D|H_2) = 0$.

Para $i = 3$:

Supondo que H_3 seja a hipótese verdadeira, a única opção que o apresentador tem é abrir a porta B, pois a porta A já havia sido escolhida por José e a porta C contém

o grande prêmio. Assim, a probabilidade da porta B ter sido revelada supondo que o grande prêmio está na porta C é 1. Ou seja, $P(D|H_3) = 1$.

O último passo necessário para obtermos todos os fatores do Teorema de Bayes, é calcularmos $P(D)$. Tal cálculo pode ser obtido através da lei da probabilidade total:

$$P(D) = \sum_{i=1}^{3} P(H_i)P(D|H_i)$$

Ao realizar os cálculos necessários, obtemos $P(D) = 1/2$. Basta então calcular a probabilidade a posteriori para cada uma das hipóteses.

- $P(H_1|D) = \dfrac{(1/3)(1/2)}{(1/2)} = 1/3$
- $P(H_2|D) = \dfrac{(1/3)(0)}{(1/2)} = 0$
- $P(H_3|D) = \dfrac{(1/3)(1)}{(1/2)} = 2/3$

Percebe-se então que a probabilidade a posteriori da hipótese em que a porta C contém o grande prêmio é duas vezes maior comparada a hipótese em que A contém o grande prêmio. Logo, é melhor que José troque de porta. Ou seja, a hipótese H_3 possui a maior probabilidade a posteriori, consequentemente é a que possui maiores chances de ocorrência.

Pensar de forma Bayesiana nos auxiliou da escolha da porta com maiores chances de conter o grande prêmio. Podemos utilizar a mesma ideia no nosso cotidiano, basta criar uma estrutura de como inserir as informações observadas no Teorema de Bayes, afim de atualizar a probabilidade do evento de interesse. Com isso em mente, apresentamos na seção seguinte a utilização do Teorema de Bayes como classificador.

2.3 Classificador Bayesiano

Mesmo não definindo anteriormente o conceito de classificação, o utilizamos de maneira empírica. O exemplo 2.2.1 trata exatamente em classificar qual das portas é a correta. Classificação é a criação de um modelo (matemático ou computacional) a partir de dados históricos, onde o mesmo possibilitará rotular exemplos futuros a partir de atributos de entrada conhecidos previamente (Faceki et al., 2015). Ressaltamos ainda que este rótulo é um valor discreto, isto é, uma classe ou nível.

A seguir definimos alguns termos importantes para construção dos classificadores, como: instância, atributo alvo, atributo de entrada, estimativa MAP.

Instância

Na área da Estatística a nomenclatura usual é "unidade amostral" e "amostra" quando nos referirmos aos dados coletados. No presente trabalho utilizaremos o conceito de "instância". Uma instância corresponde a uma unidade amostral, seja ela: indivíduo, objeto, etc. Logo, um conjunto de instâncias (ou simplesmente "instâncias") é o mesmo que "amostra".

Atributos

Atributos nada mais são do que as características observadas em uma ou um conjunto de instâncias. Tais características podem ser: peso, altura, cor da pele, renda, etc. Definimos a seguir dois tipos distintos de atributo: atributo alvo e atributos de entrada.

Atributo alvo

O atributo alvo representa dentre todos os atributos das instâncias, aquele que o pesquisador procura explicar por meio dos outros atributos. No âmbito estatístico é conhecido como "variável resposta". O conjunto dos possíveis rótulos (classes) para uma instância é representado pelo vetor $\boldsymbol{y} = (y_1, .., y_c)$, ou seja, espaço amostral do atributo alvo. Assim, para cada y_i existe uma probabilidade associada à sua ocorrência, isto é, $P(y_i)$. Note que uma instância só pode ser classificada com um único valor de $y_1, ..., y_c$, logo, os mesmos são mutuamente excludentes, consequentemente temos: $\sum_{i=1}^{c} P(y_i) = 1$.

Atributos de entrada

Os atributos de entrada são as características que ajudarão o pesquisador a explicar a ocorrência do atributo alvo, isto é, as características da instância que justificam a ocorrência de y_i. O vetor de atributos de entrada é denotado por $\boldsymbol{x} = (x_1, ..., x_p)$. Dizemos que $\boldsymbol{x}$ é o conjunto de valores observados da ocorrência do vetor aleatório $\boldsymbol{X}$, isto é, cada valor x_j observado está associado à ocorrência de uma variável aleatória X_j. Perceba que diferentemente dos modelos estatísticos usuais, as ditas "covariáveis" são consideradas variáveis aleatórias, bem como a variável resposta, isto é, atributo alvo.

Estimativa MAP

Classificadores Bayesianos seguem o mesmo princípio do exemplo 2.2.1, em que se deseja encontrar a hipótese H_i cuja probabilidade, dado que o evento observado tenha ocorrido, seja máxima. Este conceito é denominado estimativa MAP (do inglês, *Maximum A Posteriori*) (Faceki et al., 2015). Ou seja, a instância a ser classificada receberá como rótulo, a classe do atributo alvo que maximiza a ocorrência de suas características observadas (atributos de entrada).

A partir disso temos condições suficientes para formalizar um classificador Bayesiano de forma genérica. Pretende-se calcular a probabilidade de y_i ocorrer, dado que o vetor $\boldsymbol{x}$ foi observado, isto é, $P(y_i|\boldsymbol{x})$. Recorrendo a equação 2.2 e realizando as substituições necessárias em cada parcela, tem-se:

$$P(y_i|\boldsymbol{x}) = \frac{P(y_i)P(\boldsymbol{x}|y_i)}{P(\boldsymbol{x})} \tag{2.3}$$

Como $P(\boldsymbol{x})$ é constante para todo i, podemos reescrever 2.3 por meio de:

$$P(y_i|\boldsymbol{x}) \propto P(y_i)P(\boldsymbol{x}|y_i) \tag{2.4}$$

Sendo assim, para encontrar qual classe possui a maior probabilidade a posteriori, maximiza-se $P(y_i|\boldsymbol{x})$. Formalizamos então o classificador Bayesiano para caso geral:

$$\hat{y} = arg\,max_y\ P(y_i)P(\boldsymbol{x}|y_i) \tag{2.5}$$

Dizemos então, que $\hat{y}$ é a classe com maior probabilidade a posteriori.

A partir dos resultados obtidos, nos deparamos com dois desafios ao utilizar classificadores Bayesianos:

1. Definir as distribuições de probabilidade que serão adotadas para cada índice do vetor aleatório $\boldsymbol{x}$.

2. Definir como serão tratadas as possíveis dependências entre os atributos, isto é, como encarar $P(\boldsymbol{x}|y_i)$.

Sobre o primeiro desafio, para atributos binários define-se distribuição Bernoulli. No caso de outros atributos categóricos com mais de uma categoria, é comum a criação das variáveis *dummy*, isto é, são geradas $(m-1)$ novas variáveis a partir das m categorias da variável original. Assim adota-se novamente a distribuição Bernoulli.

Para os atributos contínuos, temos uma infinidade de distribuições de probabilidade que poderiam ser empregadas, entretanto, é comum utilizar a distribuição Gaussiana, uma vez que muitos fenômenos na natureza seguem este comportamento, havendo ainda facilidades na estimativa de seus parâmetros, como também é fácil obter sua densidade multivariada.

A inferência dos parâmetros das distribuições propostas nos leva ao segundo desafio. Havendo dependência entre os atributos, a estimativa dos parâmetros pode se tornar algo muito complexo. As sessões seguintes apresentam dois casos particulares do classificador Bayesiano, abordando formas diferentes de como encarar $P(\boldsymbol{x}|y_i)$, contornando o problema da dependência entre atributos. A respeito ainda da inferência dos parâmetros, dado que esta etapa não é o enfoque do presente trabalho, o método de estimação utilizado é o de máxima verossimilhança.

2.4 *Naïve*-Bayes

A maneira mais simples de representar a dependência entre os atributos na equação 2.5 é utilizando o método *Naïve*-Bayes. Como o próprio nome diz, este classificador é ingêuo (tradução do inglês, *Naïve*), pois supõem que todos os atributos de entrada são independentes entre si, dado uma classe específica do atributo alvo (Zhang, 2004). Tal artifício facilita a inferência dos parâmetros, bem como a implementação computacional. Reescreveremos a equação 2.5 sob a suposição de independência condicional dos atributos.

$$\hat{y} = arg\,max_y\, P(y_i) \prod_{j=1}^{p} P(x_j|y_i) \tag{2.6}$$

Por finalidades computacionais, aplicamos o logaritmo neperiano nesta equação, pois conforme p cresce, isto é, o número de atributos de entrada aumenta, o valor resultante deste produto tende a zero. A justificativa de tal artifício se dá pelo fato de que maximizar o logaritmo neperiano da probabilidade condicional é o mesmo que maximizar a própria probabilidade.

$$\hat{y} = arg\,max_y \ \ln P(y_i) + \sum_{j=1}^{p} \ln P(x_j|y_i) \tag{2.7}$$

A aplicação deste modelo é demasiadamente simples, uma vez que necessitamos realizar apenas a soma do logaritmo neperiano de densidades independentes.

Apesar disso, há sempre dúvidas em relação a suposição inicialmente feita, isto é, se os atributos de entrada de fato são independentes ao condiciona-los ao atributo alvo. Zhang (2004), conclui que o classificador *Naïve*-Bayes mesmo com a sua suposição violada continua com altos índices de acerto na classificação. A justificativa para isso, é que as dependências locais e global de cada atributo excluem-se mutuamente, não interferindo na classificação, tais resultados são válidos para distribuição Gaussiana.

Ao realizar inferência dos parâmetros do modelo proposto, nos limitaremos aos casos da distribuição Bernoulli e Gaussiana.

Distribuição Bernoulli

Seja $\boldsymbol{X} = (X_1, ..., X_p)$ variáveis aleatórias que seguem distribuição Bernoulli com parâmetros $\boldsymbol{\theta} = (\theta_1, ..., \theta_p)$, a função de probabilidade de um particular X_j pode ser expressa por:

$$p(x_j|\theta_j) = \theta_j^{x_j}(1-\theta_j)^{(1-x_j)} \tag{2.8}$$

Assim, ao observar n ocorrências de um particular X_j, podemos estimar o parâmetro θ_j por meio de $\hat{\theta}_j$ fazendo uso do método de máxima verossimilhança:

$$\hat{\theta}_j = \sum_{k=1}^{n} \frac{x_{jk}}{n}$$

No entanto, precisamos estimar θ_j dado y_i , isto é, estimar o parâmetro θ_j condicionado a ocorrência de y_i , obtendo $p(x_j|y_i)$. Denotaremos por x_{ijk} a k-ésima observação da j-ésima variável aleatória na i-ésima classe de $\boldsymbol{y}$. Assim, temos:

$$\hat{\theta}_{ij} = \sum_{k=1}^{n_i} \frac{x_{ijk}}{n_i}$$

Onde n_i é o tamanho da amostra observada em que ocorre y_i. Podemos expressar o parâmetro $\boldsymbol{\Theta}$ por meio de uma matriz:

$$\boldsymbol{\Theta} = \begin{bmatrix} \theta_{11} & \cdots & \theta_{1j} & \cdots & \theta_{1p} \\ \vdots & & \vdots & & \vdots \\ \theta_{i1} & \cdots & \theta_{ij} & \cdots & \theta_{ip} \\ \vdots & & \vdots & & \vdots \\ \theta_{c1} & \cdots & \theta_{cj} & \cdots & \theta_{cp} \end{bmatrix}$$

Logo, a quantidade de parâmetros a serem estimados depende tanto do número de atributos de entrada, quanto o número de classes do atributo alvo, isto é, c classes *versus* p atributos de entrada. Consequentemente, a matriz do estimador $\hat{\boldsymbol{\Theta}}$ é dada por:

$$\hat{\boldsymbol{\Theta}} = \begin{bmatrix} \hat{\theta}_{11} & \cdots & \hat{\theta}_{1j} & \cdots & \hat{\theta}_{1p} \\ \vdots & & \vdots & & \vdots \\ \hat{\theta}_{i1} & \cdots & \hat{\theta}_{ij} & \cdots & \hat{\theta}_{ip} \\ \vdots & & \vdots & & \vdots \\ \hat{\theta}_{c1} & \cdots & \hat{\theta}_{cj} & \cdots & \hat{\theta}_{cp} \end{bmatrix}$$

Como mencionado, o classificador *Naïve*-Bayes parte do princípio que todos os atributos de entrada são independentes dado uma classe específica de $\boldsymbol{y}$, assim, podemos escrever a distribuição conjunta condicional do vetor aleatório $\boldsymbol{X}$ da seguinte forma:

$$P(\boldsymbol{x}|y_i) = \prod_{j=1}^{p} p(x_j|\theta_{ij}, y_i)$$

Aplicando o logaritmo neperiano temos:

$$\ln P(\boldsymbol{x}|y_i) = \sum_{j=1}^{p} \ln p(x_j|\theta_{ij}, y_i)$$

Logo, o classificador *Naïve*-Bayes pode ser reescrito ao utilizarmos todos atributos de entrada com distribuição Bernoulli da seguinte forma:

$$\hat{y} = arg\,max_y \ \ln P(y_i) + \sum_{j=1}^{p} \ln p(x_j|\theta_{ij}, y_i) \tag{2.9}$$

Distribuição Gaussiana

Seja $\boldsymbol{X} = (X_1, ..., X_p)$ variáveis aleatórias que seguem distribuição Gaussiana (ou Normal) com parâmetros $\boldsymbol{\mu} = (\mu_1, ..., \mu_p)$ e $\boldsymbol{\sigma} = (\sigma_1, ..., \sigma_p)$, a função densidade de probabilidade de um particular X_j pode ser expressa por:

$$f(x_j|\mu_j, \sigma_j) = \frac{1}{\sqrt{2\pi\sigma_j^2}} e^{\left(-\frac{(x_j-\mu_j)^2}{2\sigma_j^2}\right)} \tag{2.10}$$

De forma semelhante à distribuição Bernoulli, pode-se realizar inferência dos parâmetros da distribuição Gaussiana por meio de seus estimadores dado uma amostra de tamanho n:

$$\hat{\mu}_j = \sum_{k=1}^{n} \frac{x_{jk}}{n}; \quad \hat{\sigma}_j^2 = \sum_{k=1}^{n} \frac{(x_{jk} - \hat{\mu_j})^2}{n-1}$$

Novamente é necessário estimar μ_j e σ_j dado uma particular classe da variável resposta, obtendo assim $f(x_j, \mu_{ij}, \sigma_{ij}|y_i)$. Tem-se então:

$$\hat{\mu}_{ij} = \sum_{k=1}^{n_i} \frac{x_{ijk}}{n_i}; \quad \hat{\sigma}_{ij}^2 = \sum_{k=1}^{n_i} \frac{(x_{ijk} - \hat{\mu}_{ij})^2}{n_i - 1}$$

Não há necessidade em se inferir as covariâncias entre as X_j e X_l uma vez que o modelo proposto utilizará o produtório das funções densidade de probabilidade de cada variável aleatória, isto é, trata-se todas relação de X_j e X_l com independência condicionada. Portanto $\sigma_{ijl}^2 = 0$, leia-se covariância da variável X_j e X_l dado y_i é igual a zero.

Em forma matricial tem-se a matriz $\boldsymbol{\mu}$ que corresponde a todas as médias de todas as variáveis dado cada valor de y_i:

$$\boldsymbol{\mu} = \begin{bmatrix} \mu_{11} & \cdots & \mu_{1j} & \cdots & \mu_{1p} \\ \vdots & & \vdots & & \vdots \\ \mu_{i1} & \cdots & \mu_{ij} & \cdots & \mu_{ip} \\ \vdots & & \vdots & & \vdots \\ \mu_{c1} & \cdots & \mu_{cj} & \cdots & \mu_{cp} \end{bmatrix} \; ; \quad \boldsymbol{\Sigma} = \begin{bmatrix} \sigma_{11}^2 & \cdots & \sigma_{1j}^2 & \cdots & \sigma_{1p}^2 \\ \vdots & & \vdots & & \vdots \\ \sigma_{i1}^2 & \cdots & \sigma_{ij}^2 & \cdots & \sigma_{ip}^2 \\ \vdots & & \vdots & & \vdots \\ \sigma_{c1}^2 & \cdots & \sigma_{cj}^2 & \cdots & \sigma_{cp}^2 \end{bmatrix}$$

Consequentemente fica trivial escrever as matrizes de seus respectivos estimadores:

$$\hat{\boldsymbol{\mu}} = \begin{bmatrix} \hat{\mu}_{11} & \cdots & \hat{\mu}_{1j} & \cdots & \hat{\mu}_{1p} \\ \vdots & & \vdots & & \vdots \\ \hat{\mu}_{i1} & \cdots & \hat{\mu}_{ij} & \cdots & \hat{\mu}_{ip} \\ \vdots & & \vdots & & \vdots \\ \hat{\mu}_{c1} & \cdots & \hat{\mu}_{cj} & \cdots & \hat{\mu}_{cp} \end{bmatrix} \; ; \quad \hat{\boldsymbol{\Sigma}} = \begin{bmatrix} \hat{\sigma}_{11}^2 & \cdots & \hat{\sigma}_{1j}^2 & \cdots & \hat{\sigma}_{1p}^2 \\ \vdots & & \vdots & & \vdots \\ \hat{\sigma}_{i1}^2 & \cdots & \hat{\sigma}_{ij}^2 & \cdots & \hat{\sigma}_{ip}^2 \\ \vdots & & \vdots & & \vdots \\ \hat{\sigma}_{c1}^2 & \cdots & \hat{\sigma}_{cj}^2 & \cdots & \hat{\sigma}_{cp}^2 \end{bmatrix}$$

O classificador *Naïve*-Bayes para distribuição Gaussiana pode ser escrito a partir da equação (2.7):

$$\hat{y} = arg\,max_y \; \ln P(y_i) + \sum_{j=1}^{p} \ln f(x_j|\mu_{ij}, \sigma_{ij} y_i) \tag{2.11}$$

Em grande parte dos problemas práticos é raro possuir apenas atributos de uma única natureza, isto é, atributos exclusivamente binários (distribuição Bernoulli) ou exclusivamente contínuos (distribuição Gaussiana). É ideal então propor um classificador que possua a flexibilidade de aplicação de ambos tipos de atributo. Por meio dos resultados obtidos em (2.9) e (2.11), é possível combinar tais equações a fim de criar um classificador que utilize ambos tipos de atributos de entrada:

$$\hat{y} = arg\,max_y \ \ln P(y_i) + \sum_{j=1}^{p} \ln p(v_j|\theta_{ij}, y_i) + \sum_{l=1}^{q} \ln f(w_l|\mu_{il}, \sigma_{il}, y_i) \tag{2.12}$$

onde $v_j \sim \text{Bernoulli}(\theta_{ij})$ e $w_l \sim \text{N}(\mu_{il}, \sigma_{il})$.

2.5 *Tree Augmented Naïve* Bayes (TAN)

Como já dito anteriormente, a pressuposição de que todos os atributos de entrada são independentes dado uma classe específica de y é algo que dificilmente pode ser comprovada. Nesta seção será apresentada uma alternativa ao método *Naïve*-Bayes, tratando dependências entre os atributos de entrada.

Um pouco mais flexível que o método *Naïve*-Bayes, *Tree Augmented Naïve* Bayes (TAN) possibilita condicionar um atributo de entrada a um (e somente um) outro atributo de entrada. Com isso, e possível tratar $P(\boldsymbol{x}|y_i)$ em (2.5) de uma maneira diferente do que apenas aplicar o produtórios das probabilidades supondo a independência condicional. Para tanto, existe uma infinidade de possibilidades de combinações de dependência entre os atributos de entrada, possibilitando ao pesquisador condicionar os eventos de interesse de maneira que melhor satisfaça o propósito de seu modelo (Friedman et al., 1997).

A flexibilidade de determinar as probabilidades condicionais permite reescrever a equação (2.5) de várias maneiras. No presente trabalho nos limitamos em apresentar apenas duas formas. A primeira, mais simples, considera que apenas um dos atributos de entrada é independente de todos os outros, da seguinte forma:

$$\hat{y} = arg\,max_y \ P(y_i)P(x_1|y_i)\prod_{j=2}^{p} P(x_j|x_{j-1}, y_i) \tag{2.13}$$

assim, o pesquisador pode definir x_1 como sendo o atributo de entrada que de fato é independente de todos os demais dado a i-ésima classe de $\boldsymbol{y}$. Para os outros tantos

atributos de entrada, a relação de dependência é tratada de maneira sequencial, isto é, x_2 é dependente de x_1 ; x_3 é dependente de x_2 ; x_p é dependente de x_{p-1}. Note que é necessário tomar o cuidado de definir qual será o atributo que será independente dos demais. Bem como a ordem das variáveis é importante ao criar as dependências.

A segunda forma de apresentar o modelo fornece um caso mais geral de como tratar as dependências entre os atributos de entrada:

$$\hat{y} = arg\,max_y\ P(y_i) \prod_{j=1}^{p} P(x_j|x_j^*, y_i) \tag{2.14}$$

onde, para ao menos um x_j, $P(x_j|x_j^*, y_i) = P(x_j|y_i)$, sendo x_j^* atributo de entrada correspondente à dependência do atributo x_j. No caso em que x_2 é dependente de x_1 tem-se $P(x_2|x_2^*, y_i) = P(x_2|x_1, y_i)$ pois $x_2^* = x_1$.

Em relação à atribuição de distribuições de probabilidade para cada x_j, tem-se os mesmos resultados obtidos na seção anterior, em que há necessidade apenas de alterações na forma de estimar os respectivos parâmetros. No presente trabalho só será abordado a estimação de parâmetros quando a variável x_j^* é discreta. Ou seja, para cada classe de x_j^* e y_i há um comportamento distinto que é esperado para x_j. Um caso em que $\boldsymbol{y} = [0, 1]$ e $x_j^* = [0, 1]$ é necessário estimar os parâmetros de x_j nas seguintes situações:

$$P(x_j|x_j^* = 0, y = 0)\ ;\ P(x_j|x_j^* = 1, y = 0)\ ;\ P(x_j|x_j^* = 0, y = 1)\ ;\ P(x_j|x_j^* = 1, y = 1).$$

A seguir é apresentado como inferir sobre os parâmetros da distribuição Bernoulli e Gaussiana.

Distribuição Bernoulli

Ao assumir que $P(x_j|x_j^*, y_i)$ segue distribuição Bernoulli tem-se:

$$p(x_j; \theta_{ihj}) = \theta_{ihj}^{x_j}(1 - \theta_{ihj})^{x_j}$$

onde θ_{ihj} é o parâmetro da distribuição de x_j na i-ésima classe de $\boldsymbol{y}$ e na h-ésima classe de x_j^*. E sua estimativa é dada por:

$$\hat{\theta}_{ihj} = \sum_{k=1}^{n_{ih}} \frac{x_{ihjk}}{n_{ih}}$$

onde x_{ihjk} é a k-ésima observação da j-ésima variável aleatória na h-ésima classe de x_j^* na i-ésima classe de $\boldsymbol{y}$ e n_{ih} é o tamanho da amostra na i-ésima classe de $\boldsymbol{y}$ na h-ésima classe de x_j^* .

Obtêm-se assim a matriz:

$$\hat{\boldsymbol{\Theta}} = \begin{bmatrix} \hat{\boldsymbol{\theta}}_{11} & \cdots & \hat{\boldsymbol{\theta}}_{1h} & \cdots & \hat{\boldsymbol{\theta}}_{1r} \\ \vdots & & \vdots & & \vdots \\ \hat{\boldsymbol{\theta}}_{i1} & \cdots & \hat{\boldsymbol{\theta}}_{ih} & \cdots & \hat{\boldsymbol{\theta}}_{ir} \\ \vdots & & \vdots & & \vdots \\ \hat{\boldsymbol{\theta}}_{c1} & \cdots & \hat{\boldsymbol{\theta}}_{ch} & \cdots & \hat{\boldsymbol{\theta}}_{cr} \end{bmatrix}$$

onde $\hat{\boldsymbol{\theta}}_{ih} = [\hat{\theta}_{ih1}, ..., \hat{\theta}_{ihp}]$. Sendo necessário então realizar a estimativa de $c \times r \times p$ parâmetros, em que c é a quantidade de classe de $\boldsymbol{y}$, r quantidade de classes das variáveis condicionais (no caso Bernoulli, $r = 2$) e p é a quantidade de atributos de entrada total . Utilizando a equação (2.14), têm-se o modelo TAN para distribuição Bernoulli da seguinte forma:

$$\hat{y} = arg\, max_y\, p(y_i) \prod_{j=1}^{p} p(x_j|\theta_{ihj}, x_j^*, y_i) \tag{2.15}$$

ao aplicar o logaritmo neperiano têm-se:

$$\hat{y} = arg\, max_y\ \ln p(y_i) + \sum_{j=1}^{p} \ln p(x_j|\theta_{ihj}, x_j^*, y_i) \tag{2.16}$$

Distribuição Gaussiana

Ao utilizar variáveis aleatória com distribuição Gaussina como atributos de entrada no modelo TAN, as probabilidades condicionais serão criadas a partir da ocorrência de variáveis categóricas. Ou seja, a média e a variância das variáveis aleatórias Gaussianas dependem de uma outra variável, esta por sua vez é categórica:

$$f(x_j|\mu_{ihj}, \sigma_{ihj}, v_j, y_i)$$

onde v_j é variável aleatória com distribuição de probabilidade discreta. A estimação dos parâmetros decorre da mesma forma do que foi apresentado no caso das variáveis Bernoulli:

$$\hat{\mu}_{ihj} = \sum_{k=1}^{n_{ih}} \frac{x_{ihjk}}{n_{ih}}; \quad \hat{\sigma}^2_{ihj} = \sum_{k=1}^{n_{ih}} \frac{(x_{ihjk} - \hat{\mu}_{ihj})^2}{n_{ih} - 1}$$

Novamente tem-se as matrizes correspondentes às estimativas:

$$\hat{\boldsymbol{\mu}} = \begin{bmatrix} \hat{\boldsymbol{\mu}}_{11} & \cdots & \hat{\boldsymbol{\mu}}_{1h} & \cdots & \hat{\boldsymbol{\mu}}_{1r} \\ \vdots & & \vdots & & \vdots \\ \hat{\boldsymbol{\mu}}_{i1} & \cdots & \hat{\boldsymbol{\mu}}_{ih} & \cdots & \hat{\boldsymbol{\mu}}_{ir} \\ \vdots & & \vdots & & \vdots \\ \hat{\boldsymbol{\mu}}_{c1} & \cdots & \hat{\boldsymbol{\mu}}_{ch} & \cdots & \hat{\boldsymbol{\mu}}_{cr} \end{bmatrix}; \quad \hat{\boldsymbol{\Sigma}} = \begin{bmatrix} \hat{\boldsymbol{\sigma}}_{11} & \cdots & \hat{\boldsymbol{\sigma}}_{1h} & \cdots & \hat{\boldsymbol{\sigma}}_{1r} \\ \vdots & & \vdots & & \vdots \\ \hat{\boldsymbol{\sigma}}_{i1} & \cdots & \hat{\boldsymbol{\sigma}}_{ih} & \cdots & \hat{\boldsymbol{\sigma}}_{ir} \\ \vdots & & \vdots & & \vdots \\ \hat{\boldsymbol{\sigma}}_{c1} & \cdots & \hat{\boldsymbol{\sigma}}_{ch} & \cdots & \hat{\boldsymbol{\sigma}}_{cr} \end{bmatrix}$$

onde $\hat{\boldsymbol{\mu}}_{ih} = [\hat{\mu}_{ih1}, ..., \hat{\mu}_{ihp}]$ e $\hat{\boldsymbol{\sigma}}_{ih} = [\hat{\sigma}^2_{ih1}, ..., \hat{\sigma}^2_{ihp}]$. Pode-se então apresentar um novo modelo TAN que leva em consideração as dependências das variáveis com distribuição Guassiana em relação á variáveis categóricas, havendo como suposição a independência da variáveis categóricas dado a classe y_i:

$$\hat{y} = arg\,max_y\, p(y_i) \prod_{j=1}^{p} P(v_l|y_i) \prod_{l=1}^{q} f(w_l|\mu_{ihj}, \sigma_{ihj}, v_j, y_i) \tag{2.17}$$

onde v_j tem distribuição de probabilidade discreta e $w_l \sim N(\mu_{ihj}, \sigma_{ihj})$. Ao aplicar o logaritmo neperiano tem-se:

$$\hat{y} = arg\,max_y\, \ln p(y_i) + \sum_{j=1}^{p} \ln P(v_j|y_i) + \sum_{l=1}^{q} \ln f(w_l|\mu_{ihj}, \sigma_{ihj}, v_l, y_i) \tag{2.18}$$

ainda, por amor à simplicidade, considera-se apenas duas categorias destas variáveis, logo, são variáveis com distribuição Bernoulli, obtendo-se:

$$\hat{y} = arg\,max_y\, \ln p(y_i) + \sum_{j=1}^{p} \ln p(v_j|\theta_{ij}, y_i) + \sum_{l=1}^{q} \ln f(w_l|\mu_{ihj}, \sigma_{ihj}, v_l, y_i) \tag{2.19}$$

pode-se ainda utilizar o resultado (2.16) para aplicar as dependências entre as variáveis discretas:

$$\hat{y} = arg\,max_y\, \ln p(y_i) + \sum_{j=1}^{p} \ln p(v_j|\theta_{ihj}, v_j^*, y_i) + \sum_{l=1}^{q} \ln f(w_l|\mu_{ihj}, \sigma_{ihj}, v_l, y_i) \tag{2.20}$$

onde $v_j \sim$ Bernoulli(θ_{ihj}) e $w_l \sim$ N$(\mu_{ihj}, \sigma_{ihj})$. Sendo este o modelo mais completo abordado no trabalho, contemplando tanto variáveis discretas quanto contínuas e suas respectivas dependências.

2.6 Sugestões pré modelagem

Devido o alto nível de complexidade que os modelos apresentados podem suportar, é necessário tomar alguns cuidados no momento da modelagem estatística. Entende-se que embora os modelos propostos sejam robustos para suportar uma grande quantidade de dados e calcular os parâmetros condicionalmente, é importante que as dependências entre os atributos de entrada façam sentido no contexto do problema a ser resolvido.

Há a necessidade de verificar o quão complexo o modelo deve ser, dado a quantidade de parâmetros a ser estimada e o poder computacional alocado para tal tarefa. Muitas vezes métricas de associação entre os atributos de entrada e o atributo alvo auxiliam no descarte de alguns atributos de entrada, diminuindo por consequência a quantidade de dependências a serem criadas. É nítido que criar dependências entre atributos discretos é substancialmente mais fácil do que para atributos contínuos, deve-se então ser considerada a possibilidade de discretizar as variáveis contínuas, Dougherty et al. (1995) aborda alguns métodos de como realizar tal tarefa.

Verificar quais atributos de entrada possuem relação entre si auxiliará o pesquisador em definir as dependências necessárias. Mais do que isso, havendo evidência estatística de que os atributos são correlatos o pesquisador deve verificar se na prática isso deve ser ou não levado em consideração, isto é, se a correlação observada é devido o acaso ou realmente os atributos possuem relação causa-efeito.

Capítulo 3

Métodos para validação de ajuste de modelos

Após considerar a teoria dos modelos de classificação apresentados, é necessário abordar métodos que auxiliam na tomada de decisão se o ajuste realizado, isto é, estimação dos parâmetros, foi adequada ou não. Esses métodos fornecem coeficientes de desempenho do ajuste do modelo. Ressaltamos mais uma vez a relevância do pesquisador no processo de modelagem. É de grande importância que não se limite apenas a estas métricas, mas também faça uso do bom senso levando em consideração o nível de complexidade do modelo proposto, desempenho computacional, implementação de código e outros tantos fatores que podem inviabilizar a utilização do modelo que possui alta performance nos coeficientes de ajuste.

Dentre os métodos que verificam a qualidade do ajuste de modelos, o presente trabalho abordará apenas dois nas próximas seções, são eles: validação cruzada k-partições e curva ROC.

3.1 Validação cruzada k-partições

O método de validação cruzada k-partições é um processo iterativo que utiliza como métrica de desempenho a acurácia, desta forma é necessário dizer o que vem a ser acurácia de um modelo. A acurácia nada mais é que a razão entre a quantidade de instâncias (objetos, indivíduos, amostras) que o modelo classificou corretamente e a quantidade total de instâncias a serem classificadas. Ou seja, se há 100 instâncias a serem classificadas e o

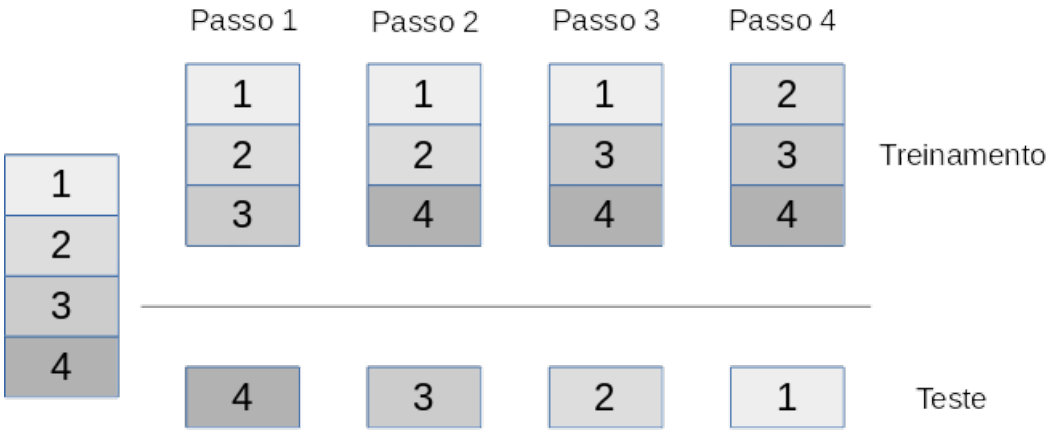
modelo classificou 80 delas de maneira correta, a acurácia do modelo é 0,80 ou 80%. Note que a acurácia não se trata do nível de certeza do modelo em classificar as instâncias, ou seja, pouco importa a probabilidade atribuída às instâncias em relação a classe que lhe foi destinada, apenas se a mesma foi classificada corretamente ou não.

O primeiro passo para realização da validação cruzada é particionar o conjunto de dados em k partições com aproximadamente mesmo tamanho e de forma aleatória. Novamente para um conjunto de 100 observações e $k = 4$ tem-se que cada partição é composta por 25 instâncias selecionadas aleatoriamente.

O passo seguinte é utilizar $(k-1)$ partições para realizar o ajuste do modelo. Dá-se o nome de base de treinamento para o conjunto de dados formado pelas $(k-1)$ partições. Calcula-se a acurácia do modelo na própria base de treinamento, onde é esperado que a mesma seja satisfatória; uma vez que o modelo está realizando predições em dados que foram utilizados para a estimativa de seus parâmetros.

A real acurácia do modelo ajustado será calculada na base de teste, que corresponde à partição desconsiderada no momento de estimativa de seus parâmetros. Guarda-se então os valores de ambas acurácias. Repete-se este processo k vezes, até que todas as partições tenham sido consideradas como base de teste ao menos uma vez, obtendo-se então k valores de acurácia na base de treinamento e k valores de acurácias na base de teste. A figura a seguir esboça um exemplo para $k = 4$ de como o conjunto de dados é particionado e como ocorre cada iteração.

Figura 3.1: Particionamento do conjunto de dados no método de validação cruzada



É realizado o cálculo da média das acurácias na base de treinamento e o cálculo da

média das acurácias na base de teste. Um modelo com bom ajuste apresenta altas taxas de acurácias média tanto na base de treinamento quanto na base de teste.

Ao ocorrer uma acurácia média na base de treinamento muito superior em relação à base de teste, possivelmente há presença de um *over-fit* causado pelo tamanho de amostra pequeno, havendo assim a necessidade de maior quantidade de dados. Outra fonte de causa de *over-fit* que deve ser observado é em relação ao vazamento de um ou mais atributos para com o atributo alvo. Caso o desempenho da acurácia em ambas as bases seja baixo, é necessário rever o modelo proposto bem como os atributos de entrada que estão sendo empregados para explicar a ocorrência do atributo alvo.

3.2 Análise ROC

Embora seja também um processo iterativo, a curva ROC, diferentemente da validação cruzada, tem como objetivo avaliar a sensibilidade e a especificidade do modelo proposto. Isso significa que além da acurácia, outras métricas de ajuste são utilizadas para avaliar performances de modelos de classificação. Apresenta-se métricas que serão utilizadas para construção da curva ROC, considerando que o atributo alvo possui apenas duas classes.

Denotaremos as duas classe por *positivo*(+) e *negativo*(-). Logo, define-se quatro tipos de ocorrência ao se tentar classificar instâncias cujas classes são:

- Verdadeiro positivo (VP): Número de instâncias positivas que foram classificadas corretamente;
- Verdadeiro Negativo (VN): Número de instâncias negativas que foram classificadas corretamente;
- Falso positivo (FP): Número de instâncias negativas que foram classificadas como positivo, portanto classificadas erradas;
- Falso negativo (FN): Número de instâncias positivas que foram classificadas como negativa, portanto classificas erradas.

Pode-se então apresentar duas métricas de desempenho a partir das ocorrências elencadas acima. Tais métricas serão utilizadas para realizar o *plot* da curva ROC.

- Sensibilidade: razão entre a quantidade de instâncias classificadas corretamente como positivas e a quantidade total de instâncias positivas.

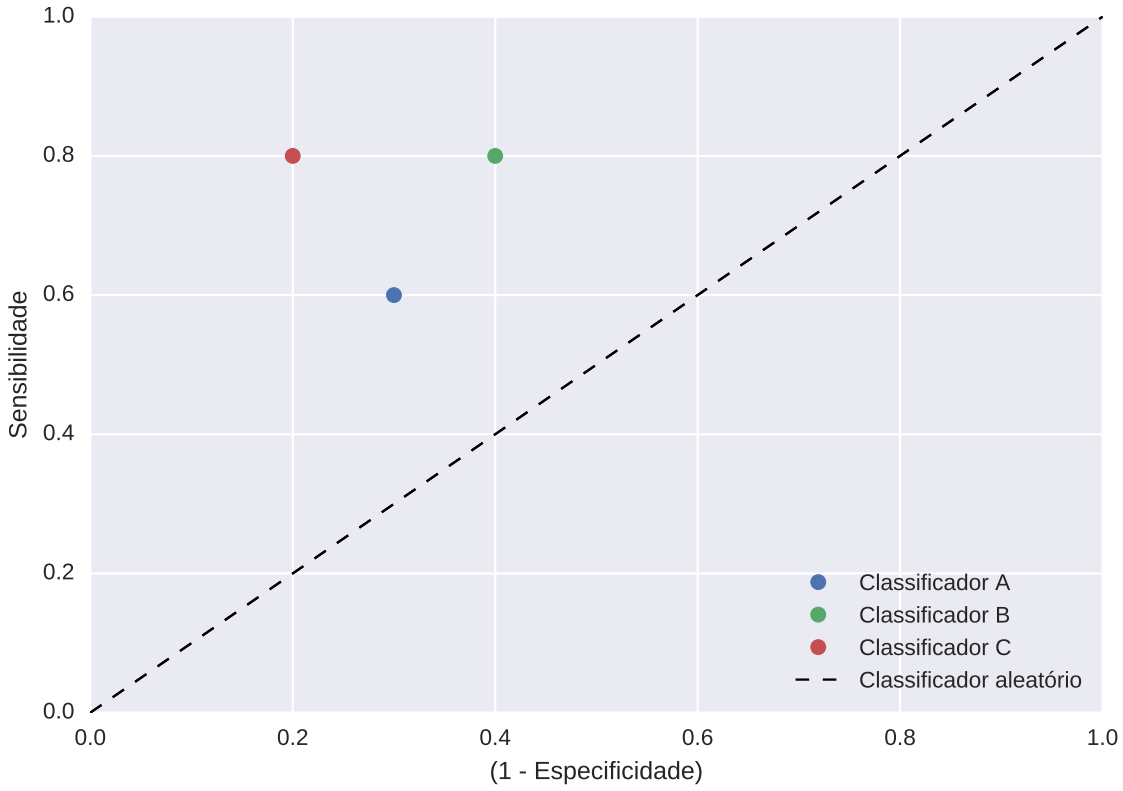
$$sens = \frac{VP}{VP + FN} \tag{3.1}$$

- Especificidade: razão entre a quantidade de instâncias classificadas corretamente como negativas e a quantidade total de instâncias negativas.

$$esp = \frac{VN}{VN + FP} \tag{3.2}$$

Quanto mais próximo do valor 1 essas métricas se aproximam, melhor será considerado o ajuste do modelo. Para análise ROC em questão, utiliza-se uma representação gráfica entre a sensibilidade e (1 - especificidade). A seguir uma figura que apresenta o desempenho de três classificados utilizando as métricas apresentadas.

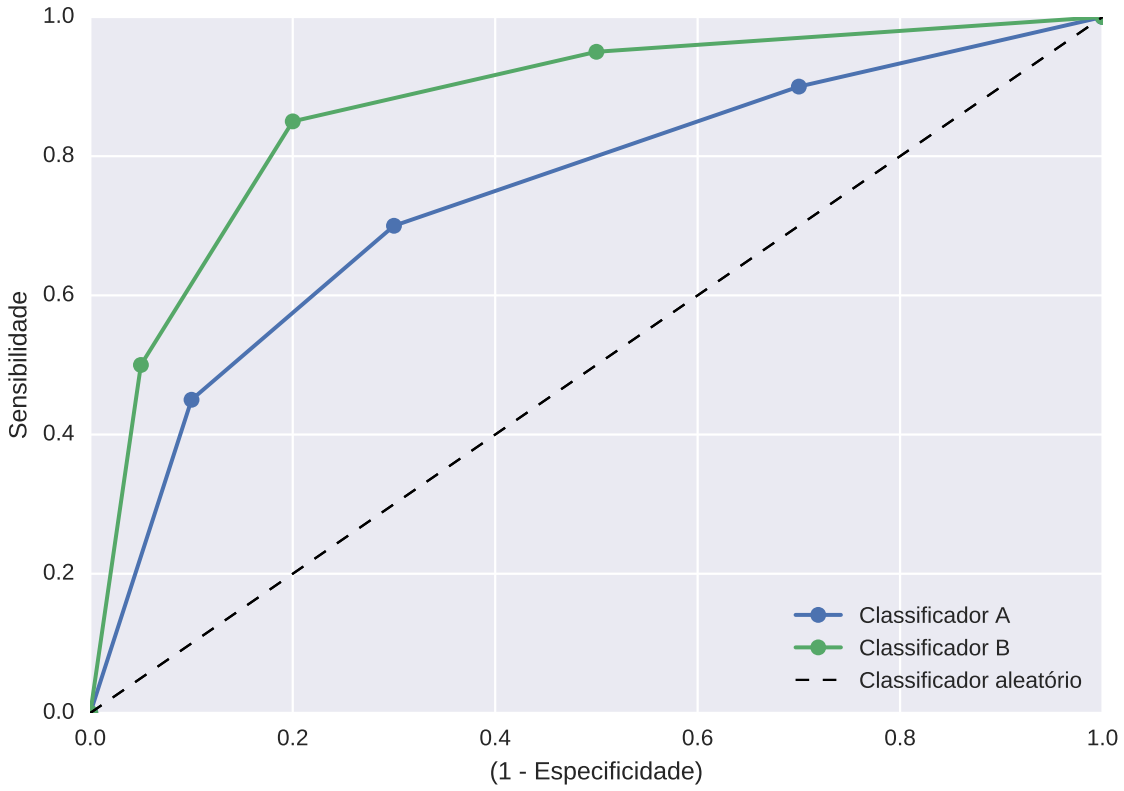
Figura 3.2: Gráfico de sensibilidade *versus* especificidade para diferentes modelos.



Analisando a figura , pode-se dizer que o classificador que apresenta a melhor performance é o classificador 'C', uma vez que é o que mais se aproxima do valor 1 em sensibilidade e possui o melhor valor de especificidade. Embora o classificador 'B' possua o mesmo valor de sensibilidade que o classificador 'C', sua especificidade é inferior.

Para traçar a curva ROC, é necessário que se crie diferentes cortes na probabilidade que o modelo atribui a cada classe, isto é, quando nos referimos a apenas duas classes, atribuindo uma probabilidade superior que 0,50 para uma determinada instância na classe (+), a mesma é classificada como (+). Consequentemente, quando a instância recebe uma probabilidade inferior que 0,50 para a classe (+), a mesma é considerada (-). Assim, altera-se este ponto de corte diversas vezes afim de medir quanto o modelo continua discriminando bem as duas classes. Espera-se que um bom modelo seja capaz de discriminar bem as classes observadas, isto é, atribuir uma probabilidade próxima de 1 para instâncias que de fato são (+) e probabilidade próximas a 0 para aquelas que de fato são (-). Após isso, obtém-se o seguinte gráfico:

Figura 3.3: Gráfico da curva ROC comparando dois modelos de classificação.



Existe uma clara diferença entre os desempenhos dos classificadores apresentados na figura 3.3, em que o classificador 'B' apresenta valores mais elevados de sensibilidade e especificidade para diferentes cortes. Assim, diz-se que o melhor classificador é sempre aquele cuja curva mais se aproxima do valor 1 em sensibilidade e do valor 0 para (1-especificidade), obtendo uma área a baixo da curva próxima a 1.

3.3 Considerações sobre métricas de desempenho

Ressalta-se a importância de avaliar cada métrica de desempenho de ajuste de modelos, nunca levando em consideração apenas uma delas isoladamente. Como visto na seção anterior, existem diversas métricas para verificar a qualidade do ajuste, havendo um propósito para cada métrica, servindo de auxílio para o pesquisador tomar a decisão de qual modelo utilizar para predições futuras.

Um grande erro que pode gerar resultados inesperados, é considerar apenas a acurácia do modelo como única métrica de ajuste a ser avaliada. Havendo casos em que a acurácia do modelo é 0,9 e ainda assim o mesmo pode não ser adequado para a situação. Uma vez que a classificação ocorre corretamente apenas para a classe (+) do atributo alvo. Ou seja, a especificidade do modelo não apresenta um bom desempenho e o modelo deve ser reconsiderado.

Capítulo 4

Aplicação ao Dota2

Esta seção aborda a aplicação dos métodos *Naïve*-Bayes e TAN para predição de partidas de Dota2. O método *Naïve*-Bayes utiliza a informação de quais personagens foram selecionados por cada equipe. Já o método TAN além dos personagens selecionados, considera-se o desempenho de cada um destes ao longo da partida realizada. Antes disso, é necessário expor de maneira introdutória como o jogo funciona e de que forma as informações contidas no mesmo foram coletadas, armazenadas e utilizadas no modelo proposto.

4.1 O jogo

Dota2 é um jogo eletrônico onde 10 jogadores se enfrentam numa arena de batalha. Cada jogador pode selecionar apenas 1 personagem dentre 113 disponíveis atualmente. Ou seja, dentre todos os personagens existentes, apenas 10 entram em jogo a cada partida realizada. Tais personagens possuem características distintas, o que torna o jogo diversificado e desafiador, pois os melhores jogadores devem compreender o funcionamento de todos os personagens, aprimorando assim estratégicas durante suas partidas (Dota2, 2016a).

O número de jogadores de Dota2 só tem aumentado dia após dia, da mesma forma que as premiações de seus torneios mundiais. Em agosto de 2016 foi realizada a 6ªedição do maior campeonato de e-sports no mundo, o *The International*, promovido pela companhia desenvolvedora do Dota2, *Valve*. A premiação total deste torneio foi de U$20.770.460,00 e a equipe que conquistou o grande prêmio levou U$9.139.002,00 para casa. Além da

quantia em dinheiro movimentada, o crescimento de pessoas que acompanham o jogo por meio de canais televisivos e sites na internet é nítido (Dota2, 2016b).

O objetivo fundamental do jogo é destruir a base da equipe adversária. Para isso, é necessário conquistar experiência e ouro (moeda do jogo), recursos que possibilitam aprimorar habilidades e equipamentos dos personagens, tornando-os mais fortes e poderosos.

Em jogos profissionais, a seleção dos personagens no início de cada partida é decisiva, traçando a estratégia que cada equipe utilizará para alcançar seu objetivo. Desta forma, dado que o primeiro personagem foi escolhido, é possível atribuir uma probabilidade de vitória para cada equipe. Isto é possível ao utilizar dados históricos de partidas contendo informações de quais personagens foram selecionados e a equipe que obteve vitória. Com isso, é possível criar um modelo para predição de partidas de Dota2 (Conley, 2013).

4.2 Coleta de dados

A desenvolvedora do jogo disponibiliza um modo que qualquer usuário de computador possa coletar dados de partidas histórias de seus servidores. Basta que o usuário crie uma comunicação com a API(*Application Programming Interface*) do jogo, requisitando por meio de linhas de comando quais dados deseja extrair.

No presente trabalho, criou-se um programa em Python para coletar os dados necessários. A coleta é realizada a cada 5 minutos por meio de um microcomputador que fica ligado 24 horas por dia. Como os dados obtidos são formato JSON, criou-se um banco de dados na estrutura de MongoDB (maiores detalhes em Hows et al. (2015)). Dentre as informações contidas em cada partida extraída, há a informação de quais personagens foram escolhidos e quem ganhou a partida. Com isso, é criada uma tabela no ambiente SQLite (maiores detalhes em Nield (2016)), estruturando assim a base de dados da maneira mais correta para realização da próxima etapa, isto é, a modelagem para predições futuras.

Tomou-se o cuidado de armazenar somente as partidas em que apenas os 12,9% dos melhores jogadores estavam presentes, isto é, só foram consideradas no estudo as partidas do melhores jogadores de Dota2. Tal ação é justificada pela busca de homogeneidade das partidas, evitando assim possíveis inconsistências na base de dados. Outra importante restrição considerada, utilizar apenas partidas com duração superior a 15 minutos, uma

vez que em média uma partida de Dota2 dura 25 minutos. Partidas que duram menos que 15 minutos podem ser caracterizadas por abandono de jogadores, o que deve ser evitado para realização da modelagem.

A partir de um volume de 650.000 partidas coletadas entre os dias 04 de agosto de 2016 e 06 de novembro de 2016, dá-se continuidade no estudo para implementação dos modelos propostos, utilizando as informações que correspondem a quais personagens foram selecionados e seus respectivos desempenhos em partida.

4.3 Modelagem

O primeiro passo para dar início ao processo de modelagem, é definir qual é o atributo alvo e qual a sua estrutura de dado. Como é desejável prever qual equipe vencerá partidas futuras, havendo ainda apenas duas equipes em cada partida, pode-se atribuir um valor booleano para representar esta informação, ou seja, quando a equipe 1 ganha, representa-se por 1, e quando a equipe 1 perde (consequentemente a equipe 2 ganha), representa-se por 0. Assim, definimos o atributo alvo y da seguinte maneira:

$$y = \begin{cases} 0, \text{ equipe 1 perdeu} \\ 1, \text{ equipe 1 ganhou} \end{cases}$$

Até a presente etapa não há diferenças entre os modelos *Naïve*-Bayes e TAN, uma vez que o propósito de ambos é o mesmo (prever qual equipe vencerá a partida). Entretanto, para definir os atributos de entrada cada modelo têm suas particularidades, assim, são apresentados em seções distintas. Dado que o modelo *Naïve*-Bayes é mais simples, seu desenvolvimento ocorre primeiramente, e seus resultados podem ser utilizados na elaboração do modelo TAN.

4.3.1 Modelagem *Naïve*-Bayes

O próximo desafio é representar matematicamente quais personagens participaram da partida e qual equipe o selecionou. No período em que as partidas foram coletadas, o número de personagens disponíveis era 112, e como são duas equipes em jogo, podemos criar um vetor com dimensão de 224. Assim, as posições de 1 a 112, representam quais personagens pertencem ou não à 1ª equipe, e as posições de 112 a 224 representam o

mesmo porém, para a 2ª equipe. Temos então a seguinte variável para representar todas essas informações:

$$\boldsymbol{V} = [\ \boldsymbol{v}_1 \ \vdots \ \boldsymbol{v}_2\ ]$$

Diz-se que $\boldsymbol{V}$ é matriz particionada de dimensão $2 \times 112 = 224$. E ainda temos que:

$$v_{pj} = \begin{cases} 0,\ j\text{-ésimo personagem ausente na } p\text{-ésima equipe} \\ 1,\ j\text{-ésimo personagem presente na } p\text{-ésima equipe} \end{cases} \tag{4.1}$$

onde $p = 1, 2$ e $j = 1, ..., 112$.

Em termos práticos, o vetor $\boldsymbol{v}$ será percorrido do índice 1 ao 224, embora acredita-se que a notação da maneira que é apresentada facilite o entendimento para o leitor.

Utilizando a equação 2.7 podemos reescreve-la da seguinte forma:

$$\hat{y} = arg\,max_y\ \ln p(y_i) + \sum_{p=1}^{2}\sum_{j=1}^{112} \ln p(v_{pj}|\theta_{ipj}, y_i) \tag{4.2}$$

Onde $i = 1, 2$.

É necessário definir como será a matriz dos parâmetros bem como a matriz dos estimadores. Seguiremos com o mesmo princípio utilizado para o vetor $\boldsymbol{x}$ para definir a matriz $\boldsymbol{\Theta}$:

$$\boldsymbol{\Theta} = \left[\ \boldsymbol{\Theta}_1 \ \vdots \ \boldsymbol{\Theta}_2\ \right]$$

Desta maneira, representamos por $\boldsymbol{\Theta}_1$ os parâmetros estimados referentes a 1ª equipe, lembrando que o mesmo é uma matriz, não um vetor, pois contêm os parâmetros das 2 classes do atributo alvo, isto é, para quando a equipe 1 ganha e para quando a equipe 1 perde. O mesmo é feito para a equipe 2, representada pela matriz $\boldsymbol{\Theta}_2$.

A seguir tem-se uma forma mais simples de definir a matriz dos parâmetros, utilizando a notação de como de fato é realizado em vias computacionais, levando apenas em consideração que são 224 atributos de entrada.

$$\boldsymbol{\Theta} = \begin{bmatrix} \theta_{1,1} & \cdots & \theta_{1,112} & \vdots & \theta_{1,113} & \cdots & \theta_{1,224} \\ \theta_{2,1} & \cdots & \theta_{2,112} & \vdots & \theta_{2,113} & \cdots & \theta_{2,224} \end{bmatrix}$$

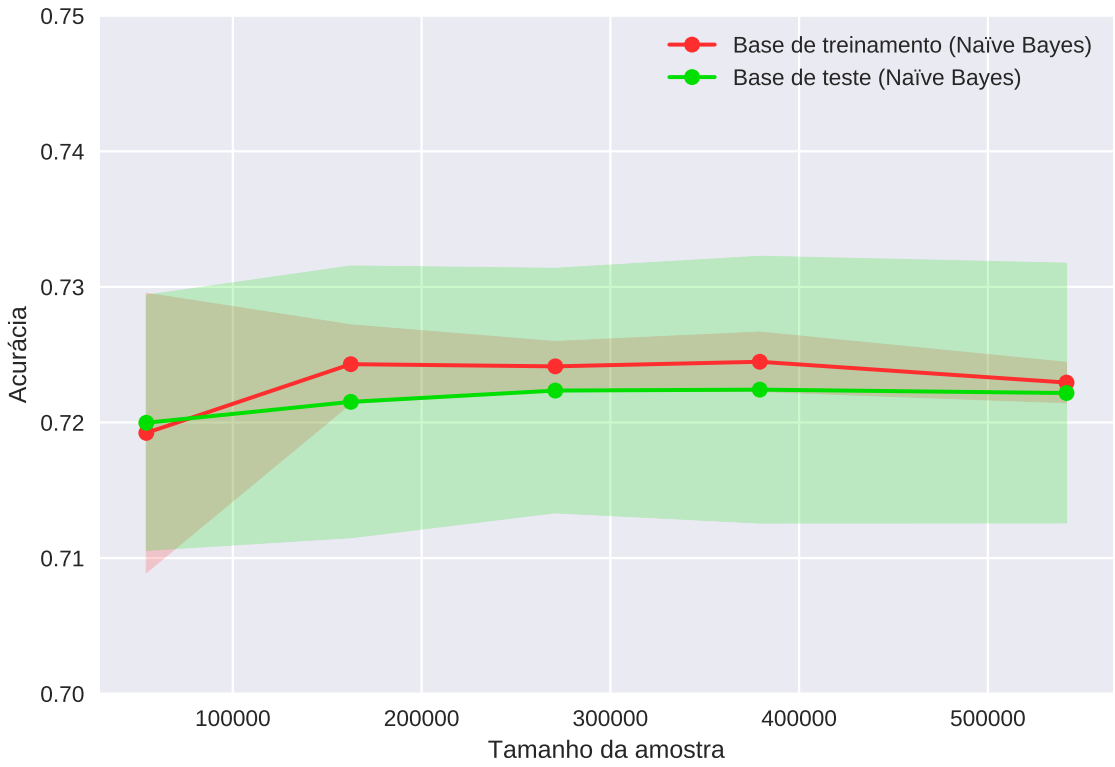
Seguiremos com esta notação para apresentar a matriz dos estimadores:

$$\hat{\boldsymbol{\Theta}} = \begin{bmatrix} \hat{\theta}_{1,1} & \cdots & \hat{\theta}_{1,112} & \vdots & \hat{\theta}_{1,113} & \cdots & \hat{\theta}_{1,224} \\ \hat{\theta}_{2,1} & \cdots & \hat{\theta}_{2,112} & \vdots & \hat{\theta}_{2,113} & \cdots & \hat{\theta}_{2,224} \end{bmatrix}$$

As bibliotecas *pandas* e *scikit-learn* foram essenciais para trabalhar com o conjunto de dados, bem como para realizar a estimação dos parâmetros, predições e validação do modelo ajustado (McKinney, 2013).

Para melhor verificar a adequabilidade do modelo proposto, utilizou-se o método de validação cruzada em cada tamanho de amostra pré definido com $k = 6$, isto é, para cada tamanho de amostra foram criadas 6 partições de forma aleatória e realizado o processo iterativo da validação cruzada. Temos a seguir uma figura que representa a performance do modelo tanto na base de treinamento quanto na base teste para diferentes tamanhos de amostra.

Figura 4.1: Curva de aprendizado para validação cruzada para o modelo *Naïve*-Bayes

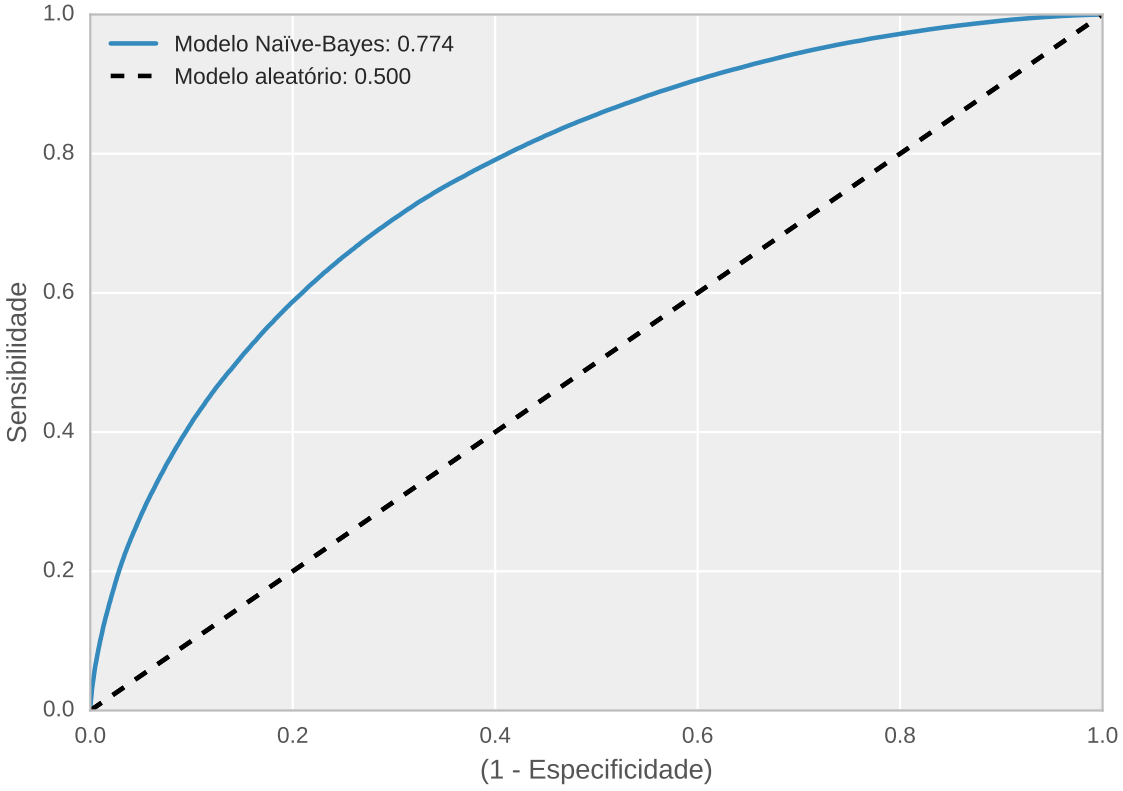


É notório que as taxas de acerto de ambos conjuntos de dados tendem a convergir, dado o aumento do tamanho amostral. Os valores exatos das acurácias em cada conjunto de dados para um tamanho de amostra 550.000, é 72,29% na base de treinamento e 72,21% na base teste. Logo, o modelo não possui problema de *over-fit* e sua real acurácia é igual a 72,21%. Isso significa que a cada 100 partidas futuras, é esperado que o modelo acerte 72

desfechos, isto é, quem ganhará a partida. Têm-se ainda confiança estatística nos valores apresentados pois as bandas de confiança para acurácia em cada tamanho de amostra são bem precisas, ou seja, os valores de acurácia não variam muito de iteração à iteração.

Como mencionado em seções passadas, só a acurácia pode não trazer todas informações relevantes sobre a qualidade do ajuste do modelo aos dados. Apresenta-se então a curva ROC do modelo.

Figura 4.2: Curva ROC para o modelo *Naïve*-Bayes para um tamanho de amostra 650.000



Ressalta-se ainda que a área abaixo da curva é igual a 0,7742. Sendo este valor satisfatório para o ajuste do modelo, tendo condições suficientes para utilizar o modelo ajustado em predições de jogos futuros. É importante relembrar que o jogo possui atualizações constantes e o balanceamento de personagens faz parte do desenvolvimento do jogo, ou seja, é necessário sempre ter dados atualizados e o modelo ajustado levando em consideração dados recentes.

4.3.2 Modelagem *Modelagem Tree Augmented Naïve* Bayes (TAN)

Na busca por uma taxa de acertividade maior, elabora-se o modelo TAN levando em consideração a performance de cada personagem durante o jogo. Para tal, a quantidade de atributos de entrada irá aumentar significativamente. Além dos 224 atributos correspondentes a presença ou não do personagem no jogo, pode-se somar o 'ouro por minuto' e a 'experiência por minuto' de cada personagem. É sabido de antemão que ambas variáveis são altamente correlacionadas, assim é preferível fazer uso de apenas uma delas. Acredita-se que a 'experiência por minuto' pode trazer mais informação a respeito do desempenho do personagem, logo, tem-se mais 224 atributos de entrada condicionados à presença ou não do personagem.

Tais atributos são valores contínuos e não podem ser representados pela distribuição Bernoulli, assim adota-se a distribuição Gaussiana. Não é proposta outra distribuição de probabilidade contínua pois para um tamanho amostral suficientemente grande há convergência para distribuição Gaussiana. Acredita-se que o conjunto de dados satisfaz este tamanho.

A dificuldade central está em como definir as probabilidades condicionais do modelo TAN para o tipo de variável e problema proposto. Assim, é razoável pensar que só faz sentido observar o desempenho de um determinado personagem se ele está presente na partida. Ou seja, o desempenho do mesmo só ocorre e deve ser levado em consideração quando o mesmo está presente na partida. Pode-se então definir a seguinte distribuição de probabilidade para 'experiência por minuto':

$$f(w_{pj}|\mu_{ipj}, \sigma_{ipj}, v_{pj}, y_i) = \begin{cases} 1 & \text{, quando } v_{pj} = 0; \\ f(w_{pj}|\mu_{ipj}, \sigma_{ipj}, y_i) & \text{, quando } v_{pj} = 1; \end{cases} \tag{4.3}$$

onde w_{pj} corresponde ao 'experiência por minuto' do j-ésimo personagem na p-ésima equipe, v_{pj} possui mesma definição em (4.1). Logo, caso o personagem não esteja presente no jogo, w_{pj} não interfere no cálculo da probabilidade no momento da predição da partida. Já quando o mesmo está presente é atribuído a distribuição Gaussiana $w_{pj} \sim N(\mu_{ipj}, \sigma_{ipj})$.

Assim, basta estimar os parâmetros de w_{pj} apenas quando $v_{pj} = 1$, em cada classe de $\boldsymbol{y}$, isto é, para quando a equipe 1 ganha e para quando a equipe 1 perde. Em outras palavras, adequa-se uma distribuição de probabilidade para o desempenho de 'experiência

por minuto' para cada personagem quando o mesmo ganha e perde, dado que o mesmo participou da partida.

A partir de (2.19) é possível definir um modelo que represente bem o problema de predição de partidas de Dota2 levando em consideração o desempenho individual de cada personagem, a que equipe pertence e se de fato está presente na partida:

$$\hat{y} = arg\,max_y \ \ln p(y_i) + \sum_{p=1}^{2}\sum_{j=1}^{112} \ln p(v_{pj}|\theta_{ipj}, y_i) + \sum_{p=1}^{2}\sum_{j=1}^{112} \ln f(w_{pj}|\mu_{ipj}, \sigma_{ipj}, v_{pj}, y_i) \quad (4.4)$$

em que $\sum_{p=1}^{2}\sum_{j=1}^{112} \ln p(v_{pj}|\theta_{ipj}, y_i)$ corresponde à informação de presença ou não do personagem em cada equipe, o que é tratado da mesma maneira que em (4.2). E ainda $\sum_{p=1}^{2}\sum_{j=1}^{112} \ln f(w_{pj}|\mu_{ipj}, \sigma_{ipj}, v_{pj}, y_i)$ é a probabilidade condicional do desempenho do personagem levando em consideração a presença ou não do j-ésimo personagem na p-ésima equipe, definida anteriormente em (4.3).

Assim, para todos os personagens de Dota2 tem-se as seguintes matrizes de parâmetros de suas respectivas distribuições de probabilidade:

$$\boldsymbol{\mu} = \left[\begin{array}{c:c} \boldsymbol{\mu}_1 & \boldsymbol{\mu}_2 \end{array} \right] \ ; \ \boldsymbol{\Sigma} = \left[\begin{array}{c:c} \boldsymbol{\sigma}_1^2 & \boldsymbol{\sigma}_2^2 \end{array} \right]$$

onde $\boldsymbol{\mu}_1$ é matriz correspondente às médias de 'experiência por minuto' dos personagens quando estão presentes na equipe 1, em cada classe de $\boldsymbol{y}$, idem para $\boldsymbol{\mu}_2$, porém para equipe 2. Raciocínio similar ao que diz respeito de $\boldsymbol{\Sigma}$, mas nos referindo à variância.

Para facilitar o entendimento, podemos reescrever as matrizes anteriores da seguinte forma:

$$\boldsymbol{\mu} = \left[\begin{array}{cccc:cccc} \mu_{1,1} & \cdots & \mu_{1,112} & \vdots & \mu_{1,113} & \cdots & \mu_{1,224} \\ \mu_{2,1} & \cdots & \mu_{2,112} & \vdots & \mu_{2,113} & \cdots & \mu_{2,224} \end{array} \right];$$

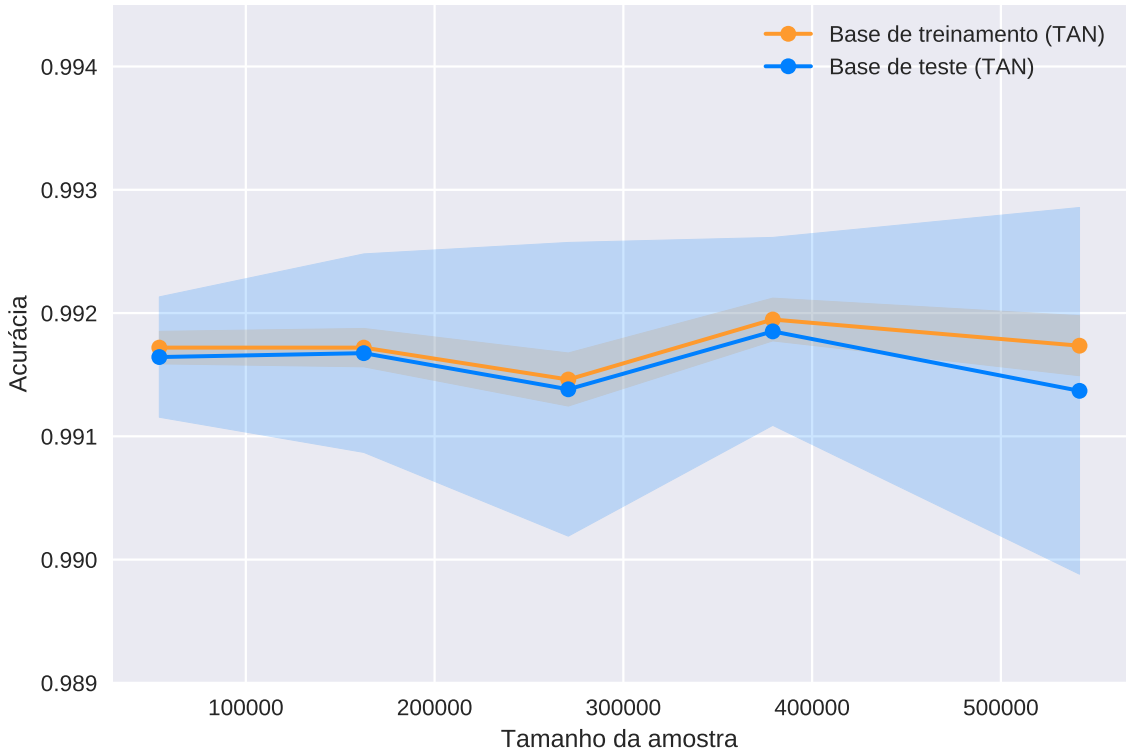
$$\boldsymbol{\Sigma} = \left[\begin{array}{cccc:cccc} \sigma^2_{1,1} & \cdots & \sigma^2_{1,112} & \vdots & \sigma^2_{1,113} & \cdots & \sigma^2_{1,224} \\ \sigma^2_{2,1} & \cdots & \sigma^2_{2,112} & \vdots & \sigma^2_{2,113} & \cdots & \sigma^2_{2,224} \end{array} \right],$$

em que $\mu_{1,1}$ é a média de desempenho do 1º personagem na equipe 1 quando o mesmo vence partidas e $\mu_{1,113}$ é a média de desempenho do 1º personagem na equipe 2 quando o mesmo vence partidas. Consequentemente, $\mu_{2,1}$ é a média de desempenho do 1ºpersonagem

na equipe 1 quando o mesmo perde partidas. O mesmo pode ser extendido à variância: $\sigma^2_{1,1}$ é a variância do 1º personagem na equipe 1 quando vence.

Para o modelo proposto não foi possível utilizar uma biblioteca computacional que já o tivesse implementado, assim o próprio autor desenvolveu-o computacionalmente, disponibilizando em seu GitHub (`www.github.com/teocalvo`). O código foi desenvolvido na linguagem de programação Python utilizando a biblioteca *numpy* para facilidades em formas matriciais e estimação dos parâmetros de forma eficiente. Abaixo tem-se os resultados obtidos com o ajuste do modelo utilizando a validação cruzada com $k = 6$.

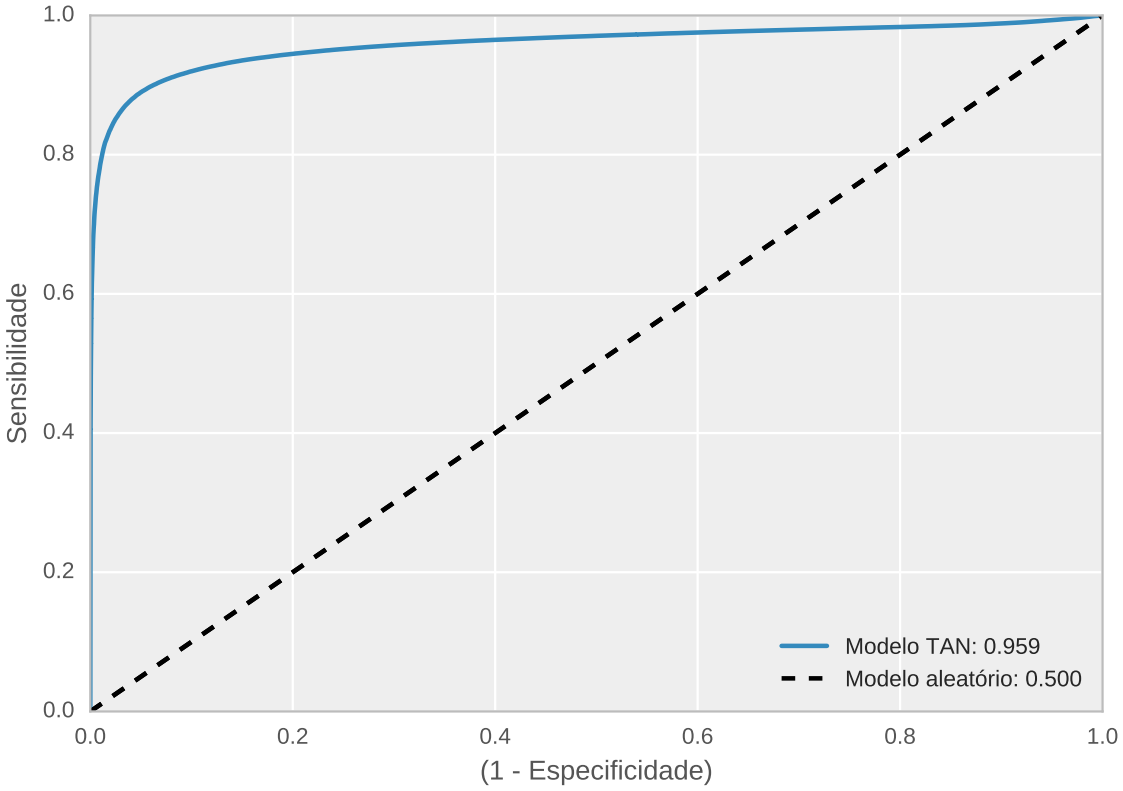
Figura 4.3: Curva de aprendizado para validação cruzada para o modelo TAN



É notório que a performance é surpreendentemente alta, em que para uma amostra de tamanho 650.000 na base de treinamento a média da acurácia é de 99,17% e para base de teste a média da acurácia é de 99,16%. Não se pode dizer que houve *over-fit* uma vez que ambas as bases em que o modelo foi testado apresentaram altíssimas taxas de acurácia, ou seja, há convicção de que o modelo foi devidamente ajustado. O próximo passo é analisar a curva ROC verificando se o modelo de fato está adequado e pode ser

utilizado na predição de partidas futuras.

Figura 4.4: Curva ROC para o modelo TAN para um tamanho de amostra 650.000



Novamente, a performance do modelo é surpreendente. Sua curva ROC apresenta comportamento muitíssimo satisfatório, uma vez que a sensibilidade e especificidade estão em níveis de alto padrão. A área abaixo da curva corresponde ao valor 0,959.

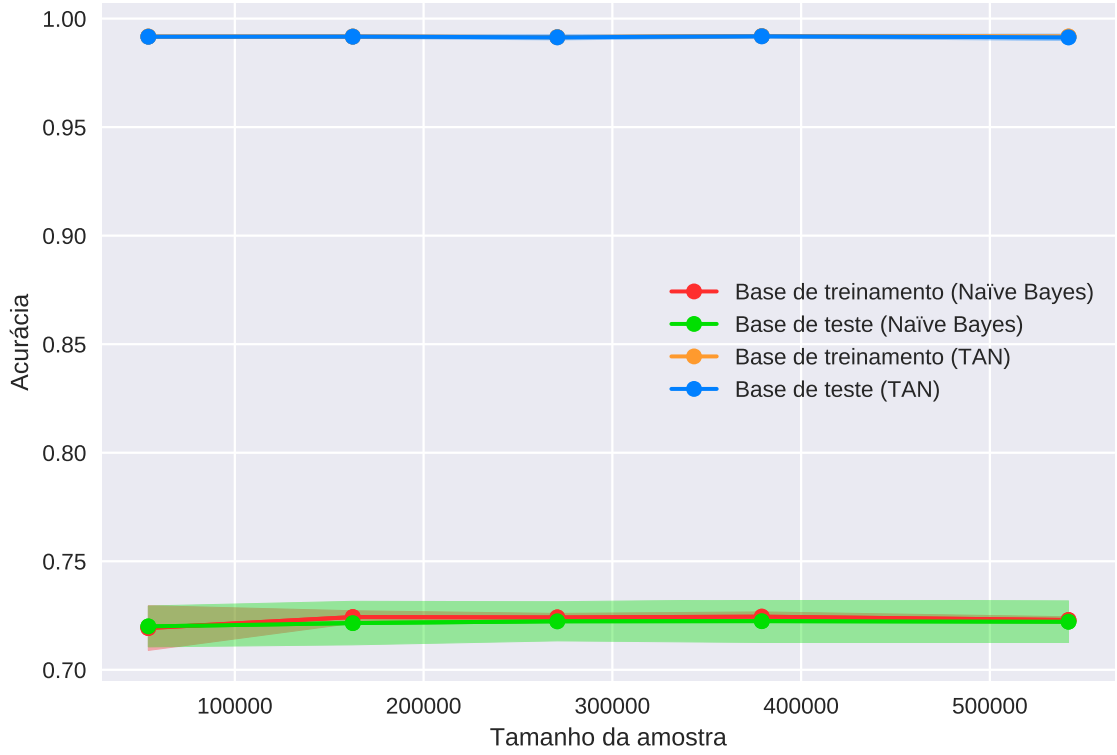
4.3.3 Comparação entre modelos

Os modelos foram apresentados de forma isolada, explorando as peculiaridades individuais e a aplicabilidade em Dota2. Busca-se levantar alguns pontos relevantes entre os modelos desenvolvidos, tais como: métricas de validação de ajuste, performance computacional, complexidade de implementação e aplicação prática.

O primeiro critério de validação a ser utilizado é a validação cruzada, em que é realizado a curva de aprendizado para os dois modelos no mesmo cenário. Isto é, em um mesmo conjunto de dados avalia-se a performance individual de cada modelo para dife-

rentes tamanhos de mostras, acompanhando assim qual se destacará de forma positiva. A imagem a seguir mostra a diferença significativa que existe entre a performance dos dois modelos propostos. O modelo *Naïve*-Bayes que utiliza apenas as informações referentes a quais personagens participaram de cada partida têm um desempenho muito inferior do que o modelo TAN, que além das informações de presença dos personagens considera também o desempenho individual de cada um deles.

Figura 4.5: Curva de aprendizado comparando os dois modelos

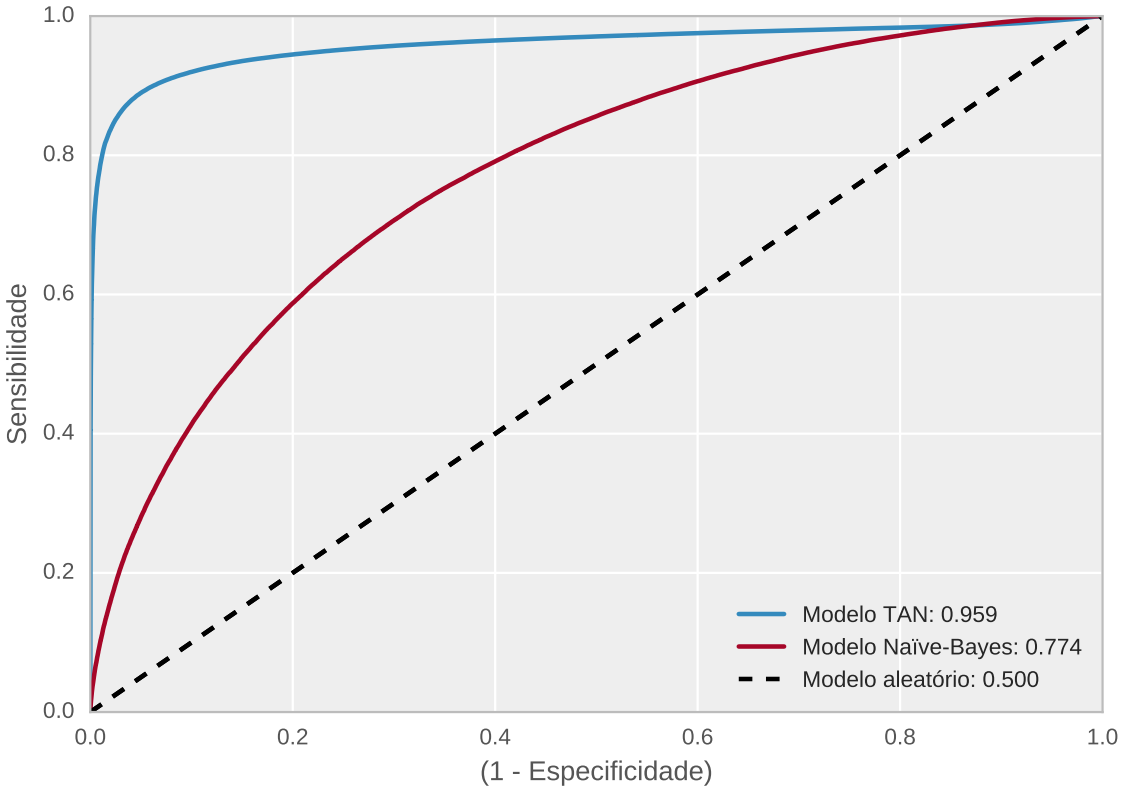


As curvas próximas ao valor 1.00 no eixo da "acurácia" são referentes ao desempenho do modelo TAN, isto é, uma performance muito satisfatória e acima (significativamente) da performance do modelo *Naïve*-Bayes, cuja é representada pelas curvas próximas ao valor 0,70 no eixo da "acurácia". Outra importante constatação a ser mencionada, é em relação à variabilidade da acurácia dos modelos propostos durante os testes realizados, em que para o modelo TAN mal se observa as bandas de confianças geradas com 95% de confiança estatística, ou seja, sua variabilidade é tão pequena que seus intervalos de confiança são extremamente precisos. O mesmo não se pode dizer a respeito do modelo

Naïve-Bayes, em que se observa seus intervalos de confiança no gráfico, embora esteja em um nível de validação dentro do aceitável. Assim, utilizando a validação cruzada como critério de decisão, tem-se, sem dúvida, que o modelo TAN deve ser adotado para predição de partidas futuras dado seu ótimo desempenho de acurácia.

Não limitando-nos à validação cruzada, faz-se questão de comparar as duas curvas ROC dos modelos propostos, adquirindo maior confiança de qual modelo deve ser adotado para ser colocado em produção.

Figura 4.6: Curvas ROC para os modelos *Naïve*-Bayes e TAN



A figura 4.6 apresenta a diferença entre as duas curvas ROC dos modelos propostos, onde fica evidente a diferença entre os dois modelos. O modelo TAN apresenta uma área abaixo de sua curva no valor de 0,959. Já o modelo *Naïve*-Bayes apresente uma área abaixo da curva ROC com valor igual a 0,774. Ambas se mostram satisfatória, ainda sim, a diferença entre os dois modelos é grande, mais uma vez o modelo TAN se mostra com melhor qualidade de ajuste.

Desta forma, avaliando as métricas apresentadas há confiança suficiente para dizer

que a performance do modelo TAN é superior que a do modelo *Naïve*-Bayes. Mas, vale lembrar que o modelo TAN possui um nível de complexidade superior, bem como sua implementação não foi contemplada por nenhum pacote computacional, havendo necessidade do autor desenvolver seu próprio modelo computacional. Na prática, operacionalmente, não seria viável demandar tanto tempo para obter um nível de sofisticação tão elevado para o modelo TAN, isto é, dado o tempo de desenvolvimento o mesmo não seria tão rentável em relação à um modelo que estivesse implementado e fornecesse resultados satisfatórios, mesmo que não tão altos. A vantagem do presente desenvolvimento é que a implementação realizada pode ser utilizada em qualquer outro tipo de problema que envolva classificação, isto é, o modelo foi implementado de tal forma que não se restringe ao problema de predições de partidas de Dota2.

A questão da implementação computacional nos remete a outro ponto: desempenho computacional do modelo. Haja visto que a implementação partiu do autor do trabalho, sendo sua experiência e formação na área da estatística, haverá *bugs* potenciais no código gerado, bem como ineficiências de processos que deverão ser aperfeiçoados. A biblioteca *scikit-learn* tem sido utilizada amplamente pelo mercado em grandes projetos, logo, seu código é devidamente otimizado e aprimorado continuamente. Deve-se então considerar que o modelo *Naïve*-Bayes está codificado de forma mais robusta que o modelo TAN, proposto pelo autor.

A última consideração a ser mencionada a respeito dos modelos propostos, consiste na aplicabilidade de cada um. O modelo *Naïve*-Bayes foi construído a partir da premissa que se sabe quais personagens compõem a partida em questão. É fato que este dado é observado ao início da partida, ou seja, a partir do instante que antecede o início da partida sabe-se exatamente quais personagens entraram no campo de batalha bem como a qual equipe pertence. Tais informações são *inputs* para o modelo realizar suas predições. Já para o modelo TAN, leva-se em consideração as informações referentes ao desempenho de cada personagem ao longo da partida, mas este fato só pode ser observado ao longo de cada instante que decorre a partida. Assim, a priori não se sabe qual *input* fornecer para o modelo realizar suas predições de antemão, isto é, antes da partida iniciar. Logo, para se ter determinado nível de confiabilidade do modelo TAN, é necessário aguardar alguns instantes para que os personagens tenham seus desempenhos desenvolvidos.

Capítulo 5

Considerações finais

Foi possível apresentar o modelo *Naïve*-Bayes tanto de maneira teórica quanto aplicada, constatando que seu desenvolvimento embora simples, é eficaz para solução do problema proposto, isto é, predição de partidas de Dota2. Ao comparar os resultados obtidos por Conley (2013), obteve-se taxas de acurácia similares ao utilizar outra técnica de modelagem (Regressão Logística), porém, *Naïve*-Bayes possui um custo computacional menor e conceitos teóricos simplificados.

Em busca de taxas de assertividade maiores, propôs-se um modelo com maior nível de sofisticação, TAN. Tal modelo considera não apenas a presença dos personagens em jogo mas também o desempenho de cada um individualmente. As métricas de validação de modelos mostram que há diferença significativa entre os modelos propostos, sendo o TAN o modelo com melhor performance em acurácia e desempenho na curva ROC.

Para utilizar os modelos propostos em produção, isto é, acompanhar partidas futuras de Dota2 utilizando as predições do modelo, o autor criou um *script* em Python que a cada 10 segundos obtêm os dados de uma determinada partida online e prediz qual equipe tem maior chance de vitória dado os personagens escolhidos e os respectivos desempenhos em "experiência por minuto" de cada personagem.

Entende-se que o autor alcançou seu objetivo em modelar estatisticamente partidas de Dota2 buscando prevê-las com determinada exatidão. Em trabalhos futuros, pretende-se utilizar dados passados no nível de jogador para obter informações a priori dos desempenhos do mesmo, assim consolida-se maiores certezas antes do início de cada partida. Embora seja um problema que possui soluções voltadas à área de computação e não estatística, há este interesse para aprimorar os resultados obtidos no presente trabalho.

Referências Bibliográficas

Bolfarine, H. (2010). *Introdução à inferência estatística*. SBM, Rio de Janeiro, 2a edição.

Conley, K. (2013). How Does He Saw Me? A Recommendation Engine for Picking Heroes in Dota 2. Technical report, Stanford University.

Dota2 (2016a). Disponível em: `<http://www.dota2.com>`. Último Acesso: 12 de abril de 2016.

Dota2 (2016b). The International 2016. Disponível em: `<http://www.dota2.com/international/overview>`. Último Acesso: 29 de outubro de 2016.

Dougherty, J. et al. (1995). Supervised and Unsupervised discretization of continuous features. *Machine Learning Proc. 12th International Conference*.

Downey, A. B. (2012). *Think Bayes: Bayesian Statistics Made Simple*. Green Tea Press.

EsportsEarnings (2016). Disponível em: `<http://www.esportsearnings.com>`. Último Acesso: 12 de abril de 2016.

Faceki, K. et al. (2015). *Inteligência Artificial: Uma Abordagem de Aprendizagem de Máquina*. LTC.

Friedman, N. et al. (1997). Bayesian network classifiers. *Machine Learning*, 29:131–163.

Hows, D. et al. (2015). *Introdução ao MongoDB*. Novatec, São Paulo.

McKinney, W. (2013). *Python for Data Analysis*. O'Reilly Media.

Nield, T. (2016). *Introdução à linguagem SQL: abordagem prática para iniciantes*. Novatec, São Paulo.

O'Higgins, N. (2011). *MongoDB and Python*. O'Reilly Media.

Sheldon, R. (2010). *Probabilidade: Um curso moderno com aplicações*. Bookman, Porto Alegre, 8a edição.

Summerfield, M. (2012). *Programação em Python 3: uma introdução completa à linguagem Python*. Alta Books, Rio de Janeiro.

Zhang, H. (2004). The Optimality of Naive Bayes. *University of New Brunswick*.

Zhang, H. and Ling, C. X. (2001). Learnability of augmented Naive Bayes in nominal domains. *Proceedings of the Eighteenth International Conference on Machine Learning*.