UNIVERSIDADE FEDERAL DO AMAZONAS
INSTITUTO DE CIÊNCIAS EXATAS
CURSO DE BACHARELADO EM ESTATÍSTICA

CLASSIFICAÇÃO ESTATÍSTICA DE TIPOS DE FALHAS EM SINAIS DOCSIS 3.0 EM TESTES DE ROTEADORES WIRELESS

Vanessa Almeida de Lima

MANAUS
2020

CLASSIFICAÇÃO ESTATÍSTICA DE TIPOS DE FALHAS EM SINAIS DOCSIS 3.0 EM TESTES DE ROTEADORES WIRELESS

Vanessa Almeida de Lima

Monografia apresentada ao Curso de Estatística da Universidade Federal do Amazonas - UFAM, como parte dos requisitos necessários à obtenção do título de Bacharel em Estatística.

Orientador : Prof. Me. Diego da Silva Souza

MANAUS
2020

Lima, Vanessa Almeida de
— Classificação estatística de tipos de falhas em sinais DOCSIS 3.0 em testes de roteadores wireless / Vanessa Almeida de Lima, 2020. 32 f.: il.color; 31 cm

Orientador: Diego da Silva Souza
TCC de Graduação (Estatística) - Universidade Federal do Amazonas, Instituto de Ciências Exatas.

1. Classificação 2. Pacote rminer
3. Reconhecimento de padrões I. Souza, Diego da Silva,
II. Universidade Federal do Amazonas
III. Título

Agradecimentos

Agradeço a Deus, por todas as minhas conquistas e por ter me dado forças para superar os obstáculos.

Agradeço a minha família, em especial minha avó, Maria de Lourdes, meu namorado, amigos, colegas de graduação, corpo docente do ICE-UFAM e amigos do estágio. Todos em conjunto colaboraram direta ou indiretamente com amor, carinho, paciência durante meus momentos de desesperos, apoio e incentivo para aqueles momentos que tudo parece dar errado. Obrigada a todos vocês por ajudarem no meu desenvolvimento acadêmico, pelos ensinamentos compartilhados e pelas experiências vividas nesses últimos anos.

Resumo

A crescente competitividade nos sistemas industriais, faz com que detecção de falhas se torne cada vez mais importante. Neste trabalho é feito um experimento utilizando modelos de classificação de reconhecimento de padrões em dados de roteadores *wireless*, modelo TC7337, com intuito de encontrar um classificador que melhor se adeque a esses dados. Ao fim do trabalho, foi desenvolvido uma ferramenta interativa para o *software* R, através do pacote *Shiny* para facilitar esta classificação ao usuário.

Palavras-chave: Classificação, Pacote *rminer*, Reconhecimento de padrões.

Abstract

The increasing competitiveness in industrial systems makes fault detection increasingly important role. In this study, reflects on research an experiment using pattern recognition classification models on data from textit wireless routers, model TC7337, in order to find a classifier that best fits these data. At the end of the study, an interactive tool for textit software R was developed, through the textit Shiny package to facilitate this classification for the user.

Keywords: Classification, Pattern recognition, Rminer package.

Sumário

Lista de Figuras

1 Introdução

Com a expansão da tecnologia ocorrendo em ritmo acelerado, o mercado vem se tornando mais competitivo e as empresas em busca de se adaptarem a essas novas tecnologias e atender melhor o mercado consumidor, estão se tornando mais rigorosas em termos de ciência. A qualidade de um produto, é fundamental para uma empresa se manter em meio a tanta concorrência, sendo a detecção de falhas uma tarefa indispensável quando se trata de determinar a precisão na avaliação de um produto durante o processo de produção, pois quanto mais cedo uma falha for detectada, menor será o custo com desperdícios de tempo e capital, a fim de repara-la. Contudo, existem certos tipos de falhas difíceis de detectar devido à complexidade do produto, como por exemplo, a falha do próprio equipamento usado para realizar a detecção ou à falta de experiência do operador que irá realizar os testes.

Neste trabalho, o produto a ser estudado são aparelhos roteadores *wireless* modelo TC7337, produtos fabricados em uma empresa do Pólo de Produtos Elétricos e Eletrônicos da Zona Franca de Manaus.

1.1 Objetivos

1.1.1 Objetivo geral

Detectar qual a causa da falha na transmissão do sinal DOCSIS 3.0 do CMTS até os equipamentos de testes, a fim de identificar e reparar o problema pela equipe da engenharia e manutenção, de forma mais eficiente.

1.1.2 Objetivo específico

- Apresentar as definições de classificadores usuais, que serão utilizados na solução do problema;
- Utilizar os classificadores definidos nos dados referentes a detecção de falha no sinal;
- Comparar o desempenho entre os classificadores através de critérios de avaliação de classificadores;
- Desenvolver uma *webapp* para monitoramento do sinal no processo produtivo.

1.2 Organização da Monografia

Este trabalho está estruturado em 5 capítulos, incluindo este capítulo inicial o qual apresentou a problemática, justificativa e contribuições desse trabalho assim como os objetivos, geral e específicos. Os demais capítulos estão dispostos conforme especificado a

seguir: no Capítulo 2 se encontram as definições básicas de Reconhecimento de padrões; No Capítulo 3 está a revisão bibliográfica, onde são apresentadas os principais classificadores da literatura, incluindo os que foram selecionadas para implementação e comparação neste trabalho; No Capítulo 4, são apresentados os algoritmos que iremos utilizar para avaliação de desempenho dos classificadores; No Capítulo 5, são descritas as bases de dados que foram utilizadas nos testes, são apresentados os resultados dos testes e é realizada uma discussão sobre os resultados dos métodos avaliados. E por fim, no Capítulo 6 são feitas as conclusões finais deste trabalho.

2 Reconhecimento de padrões

Reconhecimento de padrões, trata-se da área de pesquisa que tem por objetivo a classificação de objetos em um número de categorias ou classes, estes objetos podem variar de acordo com a aplicação [Theodoridis and Koutroumbas, 2006]. Segundo Ross (2010) reconhecimento de padrões pode ser definido como um processo de identificação de estruturas em dados por comparações com estruturas conhecidas. Por sua vez, Duda et al. (2001) dizem que o reconhecimento de padrões visa construir uma representação mais simples de um conjunto de dados através de suas características mais relevantes, possibilitando sua partição em classes.

É necessário conceituar alguns elementos fundamentais no reconhecimento de padrões: característica, padrão, classe, vetor de características, matriz de características, classificação e similaridade. Característica pode ser definida como qualquer medida útil retirada de um objeto no processo de identificação do padrão. Padrão é descrito por um conjunto de atributos que determinam características básicas de um objeto, podendo ser definido como uma descrição quantitativa ou estrutural do objeto dentro de determinada classe ou categoria. Classe é um conjunto de padrões que compartilham propriedades em comum ao objeto de estudo. Característica é definida como um dado retirado de uma amostra com base numa escala de medição, que ocorre quando uma regra (função) associa um valor numérico ou simbólico a um atributo de um objeto [Tan et al., 2009]. Considerando um conjunto fixo de atributos, podemos organizar os objetos como coordenadas de pontos no espaço n-dimensional, sendo estes conhecidos como vetores de características.

2.0.1 Classificação

Existem duas principais categorias: supervisionada e não-supervisionada, dependendo do algoritmo que será aplicado. Ambos os casos demandam duas fases: a do treinamento e a da classificação [Moreira, 2003]. Supervisionada em que as classes são definidas *a priori*, que permitem identificar a classe de interesse e não-supervisionada são geradas de acordo com a similaridade dos padrões de treinamento, não existe informação *a priori* acerca das classes.

2.0.2 Similaridade

A similaridade é definida como uma condição de dois objetos similares, isso ocorre quando eles possuem valores semelhantes em algum atributo comum, ou seja, partilham características tais como forma, tamanho, valor, orientação, entre outros.

3 Algoritmos de Aprendizado de Máquinas

A seguir, são apresentadas brevemente algoritmos mais usuais que serão utilizados no presente trabalho. Como não existe uma forma ideal de utilizar o aprendizado de máquinas em todas as perspectivas, será necessário comparar o desempenho de forma que permita a avaliação de diferentes algoritmos.

3.1 Classificador *Naive Bayes*

Trata-se de um classificador probabilístico baseado no Teorema de Bayes, que pressupõe independência. Para simplificar, um classificador ingênuo (*naive*) de Bayes assume que a presença (ou ausência) de um determinado recurso de uma classe é não relacionado à presença (ou ausência) de qualquer outro recurso. Por exemplo, uma fruta pode ser considerada uma maçã se é vermelha, redonda e tem cerca de 10 cm de diâmetro. Mesmo que esses recursos dependam um do outro ou da existência de outras características, um classificador ingênuo de Bayes considera que todas essas propriedades contribuem independentemente para a probabilidade dessa fruta ser uma maçã [Babin et al., 2011].

Sejam A e B dois eventos associados ao experimento ε, segundo Meyer (1969) o teorema é dado por:

$$P(B_i|A) = \frac{P(A|B_i)P(B_i)}{\sum_{j=1}^{k} P(A|B_j)P(B_j)}, \qquad i = 1, \ldots, k. \tag{1}$$

O classificador *Naive Bayes* é construído utilizando dados de treinamento para estimar a probabilidade de um documento pertencer a uma classe:

$$P(A|B = b) = \prod_{j=1}^{k} P(A_j|B = b), \qquad j = 1, \ldots, k. \tag{2}$$

em que cada conjunto de atributos A = $A_1, A_2, \ldots, A_k$ consiste de k atributos.

3.1.1 *Naive Bayes Normal*

Métodos de classificação baseados em modelos normais predominam na prática estatística por causa de sua simplicidade e boa eficiência numa ampla variedade de problemas reais. Neste trabalho, um caso particular será considerado para o *Naive Bayes* na estimação das densidades marginais, empregando distribuições normais univariadas. A função de verossimilhança é dada por:

$$P(\mathbf{X}{=}\mathbf{x}|Y = y) = \prod_{j=1}^{p} \frac{1}{\sqrt{2\pi\sigma_{jy}^2}} \exp\left\{-\frac{1}{2\sigma_{jy}^2}\left(x_{jy} - \mu_{jy}\right)^2\right\}. \tag{3}$$

onde j = 1,2, ..., p e y = 1,2,..., k.

Independentemente das suas limitações em assumir independência entre os atributos ou variáveis, o classificador exibe bons desempenhos para muitos problemas reais. Uma das vantagens do classificador de Bayes é que ele só requer uma pequena quantidade de dados de treinamento para estimar os parâmetro (média e variância das variáveis) necessário para a classificação. Vale salientar que as variáveis independentes assumem apenas as variâncias das variáveis para cada classe que precisam ser determinadas.

3.2 *Linear discriminant analysis* - LDA

Por Sarda-Espinosaa et al.(2017) a análise discriminante linear (LDA) é um procedimento usado no *Machine Learning* para classificação ou redução de dimensionalidade. O objetivo é encontrar uma combinação linear dos recursos que captura a variabilidade nos dados de maneira que os diferentes níveis de classe possam ser separados por hiperplanos. Segundo, Ye et al.(2004), a lda clássica projeta os dados em um espaço vetorial de dimensão mais baixa, de modo que a razão entre a distância entre classes e a distância dentro da classe seja maximizada, atingindo a discriminação máxima. A projeção ideal (transformação) pode ser facilmente calculada aplicando a composição automática nas matrizes de dispersão.

Supondo que a classe y tem densidade $f_y(\mathbf{x})$ e probabilidade *apriori* de π_y.

$$P(Y = y|\mathbf{x}) = \frac{f_y(\mathbf{x})\pi_y}{\sum_j f_j(\mathbf{x})\pi_j} \tag{4}$$

onde $f_y(\mathbf{x})$ é a densidade da $normal_p(\boldsymbol{\mu}_y, \Sigma^*)$, $\Sigma^* = \frac{1}{k}\sum_{y=1}^{k}\Sigma_y$ e π_y é a probabilidade *a priori* da classe y.

Considere $\mathbf{X} \mid Y = w_v \sim \text{Normal}\left(\mu_v, \sum\right), v = 1, \ldots, r$. O classificador de Bayes com função de perda 0-1, é dado por:

$$d_{ADL}\left(\mathbf{x}\right) \;=\; \arg\max_{\Omega}\left\{\mathbf{x}'\sum{}^{-1}\mu_v - \frac{1}{2}\left(\mu_v^{'}\sum{}^{-1}\mu_v\right) + \log P\left(w_v\right)\right\}. \tag{5}$$

Esse técnica é intitulado como Análise Discriminante Linear. Dessa forma, a regra estimada é representada da seguinte maneira:

$$\hat{d}_{ADL}\left(\mathbf{x}\right) \;=\; \arg\max_{\Omega}\left\{\mathbf{x}'\hat{\sum}^{-1}\hat{\mu}_v - \frac{1}{2}\left(\hat{\mu}_v^{'}\hat{\sum}^{-1}\hat{\mu}_v\right) + \log \hat{P}\left(w_v\right)\right\}. \tag{6}$$

3.3 *Decision Tree*

Uma árvore de decisão (do inglês, *Decision Tree*) é uma estrutura de árvore do tipo fluxograma, em que cada nó interno é indicado por retângulos e os nós folha são denotado por ovais. Em cada nó, geralmente uma única variável é considerada e um ou mais limites são escolhidos usando uma determinada medida de qualidade de divisão ou impureza do

nó. Esses limites, representados pelos galhos da árvore, dividem o espaço de decisão para a variável considerada. Assim, uma instância pode ser classificada iniciando no nó raiz, analisando a variável especificada, seguindo o ramo apropriado e repetindo recursivamente até que uma folha seja alcançada [Sarda-Espinosaa et al., 2017].

3.4 *Conditional Inference Tree*

Árvore de inferência condicional, segundo Sarda-Espinosaa et al. (2017) é um possível algoritmo de árvore de decisão para divisão binária recursiva. Ele incorpora a estrutura em um ambiente estatístico bem definido, com base em testes de permutação, tentando distinguir entre melhorias significativas e insignificantes. Para Hothorn e Hornik (2015) é uma classe não paramétrica de árvores de regressão incorporando modelos de regressão estruturados em árvore em uma teoria bem definida dos procedimentos de inferência condicional.

3.5 *K-nearest neighbor*

O método dos K- Vizinhos mais Próximos (KNN, do inglês *k-Nearest Neighbor*) proposto por Fix e Hodges (1951), é um classificador de padrões no qual, está entre os métodos mais simples e de fácil implementação. O Classificador KNN é um método de aprendizado supervisionado do formato lazy, adotado por Aha et al. (1991) O conceito desse classificador consiste em encontrar os k padrões rotulados mais próximos do padrão não classificado e, baseado no rótulo desses padrões mais próximos, é tomada a decisão relativa à classe do padrão não rotulado. Os Classificadores da família KNN não necessitam de muito esforço durante o treinamento do estudo, em compensação, o custo computacional para rotular um novo padrão é consideravelmente alto, visto que, no pior dos casos, esse padrão deverá ser confrontado com todos os padrões contidos no conjunto de padrões de treinamento. O valor de k tem seu funcionamento no qual consiste em ser escolhido pelo usuário, com o objetivo de obter uma menor taxa de erro estimada de classificação. Usualmente, para evitar "empates", se aplica mais valores impares de k, sendo eles $k =$ 1,3,7 e n. Este procedimento, no entanto, não é imprescindível, podendo-se empregar valores pares de k, no qual n é o tamanho do conjunto de treino.

3.6 *Support Vector Machine*

O método de Máquinas de Vetores Suporte (SVM, do inglês *Support Vector Machines*) proposto por Vapnik e Cortes (1995), é uma generalização do algoritmo *Generalized Portrait* desenvolvido na Rússia nos anos sessenta. Este é um dos método de aprendizagem de máquina de maior sucesso, por apresentar resultados semelhantes ou, muitas vezes, superiores aos alcançados por demais algoritmos de aprendizagem, na qual visa a

proposição de técnicas de aprendizado que procuram a minimização não só do erro do treinamento, mas também da complexidade do modelo obtido, na qual resulta em um dos principais pontos fortes da SVM, sendo alta a capacidade de generalização. Seu método é empregado na classificação para dados lineares e não lineares e podendo da mesma forma, serem empregados para classificação de padrões e regressão linear. Na circunstância de padrões separáveis que podem aparecer no contexto de classificação de padrões a concepção principal de uma SVM é produzir um hiperplano como superfície de decisão de modo que a margem de separação entre exemplos positivos e negativos seja máxima. Dessa forma, a avaliação do principio indutivo é fundamentado no caso de que a taxa de erro de uma maquina de aprendizagem com relação a dados de teste é delimitada pela soma da taxa de erro de treinamento e por um termo que depende da dimensão de Vapnik - Chervonenkis (V-C). Para casos de padrões separáveis, o método SVM gera um valor de zero para o primeiro termo e minimiza o segundo termo [Haykin, 1995]. No entanto, este método tem ganhado considerável atenção, por ser um método no qual, se tornou uma das técnicas mais populares no sistemas de aprendizagem de máquinas, podendo ser implementado tanto em problemas de predição quanto de regressão.

3.6.1 *Kernel* SVM

A aplicabilidade da função *Kernel* é capacitar as operações para estarem devidamente habilitadas para serem executadas no espaço de entrada, deste modo, evitando assim, realizá-las no espaço de características de maior dimensão. Consequentemente, o produto interno não necessitará ser avaliado no espaço de características, livrando-se do problema da alta dimensionalidade. Contudo, o trabalho computacional é ainda relevante ao número de dados de treinamento e, para acarretar uma boa distribuição dos dados em problemas de alta dimensionalidade, dependem de um conjunto de treinamento de elevada cardinalidade.

3.6.2 Funções *Kernel* SVM

Estes resultados são baseados no espaço de reprodução de *Kernel* Hilbert. No qual, se tem sua representação por um produto interno no espaço de característica, sendo que tem um *kernel* equivalente ao espaço de entrada, na seguinte representação:

$$L\left(x_i, x_j\right) = \left(\varphi\left(\boldsymbol{x}_i\right) \cdot \varphi\left(\boldsymbol{x}_i\right)\right), \tag{7}$$

sendo que algumas condições devem ser satisfeitas. A primeira, se L é uma função definida positiva e simétrica, deve-se satisfazer as condições de Mercer:

$$\begin{cases} L\left(x_i, x_j\right) = \sum_{m=1}^{\infty} \alpha_m \varphi_m\left(\boldsymbol{x}_i\right) \varphi_m\left(\boldsymbol{x}_j\right) L & \alpha_m \geq 0 \\ \int\int \left(x_i, x_j\right) g\left(x_i\right) g\left(x_j\right) dx_i dx_i > 0 & g \in H_2 \\ \int g^2\left(x_i\right) dx < \infty \end{cases}$$

Em que H_2 representa o espaço de norma euclidiana.

3.6.3 Funções de base radial gaussiana

A Funções de base radial gaussiana (RBF) é dada pelo seguinte conjunto de regras de decisão:

$$f\left(\mathbf{x}\right) = sign\left(\sum_{i=1}^{N} a_i \, K\left(\mathbf{x}, \mathbf{x}_i\right) + b\right), \tag{8}$$

em que K $\left(\mathbf{x}, \mathbf{x}_i\right)$ necessita da distância $||\mathbf{x} - \mathbf{x}_i||$ entre dois vetores. A sua aplicabilidade da função Kernel RBF é uma operação monotônica definida positiva para qualquer σ fixado, e tende a zero quando $||\mathbf{x} - \mathbf{x}_i||$ vai para o infinito. A função mais comum deste

procedimento é:

$$L\left(\mathbf{x}, \mathbf{x}_i\right) = \exp\left(\frac{||\mathbf{x} - \mathbf{x}_i||^2}{2\sigma^2}\right), \tag{9}$$

3.6.4 Modelo KSVM utilizado

Para o ajuste do modelo de classificação KSVM foi utilizado os parâmetros pré-estabelecidos (C = 1, σ = "automatic"), em que o modo automático usa um intervalo de valores para o parâmetro "sigma" no kernel Gaussiano, a estimativa que é baseada nos dados a serem utilizados, para calcular um "bom" sigma. Utilizando a função KSVM do pacote *kernlab*, Karatzoglou et al. (2019), versão $0.9 - 29$, assim como o kernel RBF que é padrão dessa função.

3.7 *Random Forest*

Random Forest é uma aplicação que utiliza métodos que são eventualmente capazes de identificar variáveis na qual o modelo casual é desconhecido e capaz de lidar com problemas de dados com alta dimensionalidade. O algoritmo é fundamentado em árvores (classificação e regressão), que retrata a agregação de *Bootstrap*, e proporciona a redução da variância sem afetar muito o viés, além de apresentar atributos importantes como a sua flexibilidade, capacidade de trabalhar com um grande número de variáveis de entrada e sua simplicidade. Um dos principais exemplos é o *bagging* [Breiman,1996], no lugar em

que cultivar cada árvore uma seleção aleatória é feita a partir dos exemplos no conjunto de treinamento.

Outra aplicação, é a seleção por divisão aleatória na qual, em cada nó, a divisão é selecionada aleatoriamente entre os melhores splits do K [Dietterich, 2000]. Em Breiman (1996) é gerado um novo treinamento que define aleatoriamente as saídas no conjunto de treinamento original. Posteriormente, outra abordagem é selecionar o conjunto de treinamento de uma conjunto de pesos nos exemplos no conjunto de treinamento.

3.8 *Extreme Gradient Bosting*

Extreme Gradient Bosting - Xgboost é semelhante a árvore de decisão com o diferencial que utiliza um método de otimização de algoritmo. É um derivado eficiente da *Gradient Boosting Machine*(GBM) que tem sido uma ferramenta com características, devido a sua facilidade de uso, facilidade de paralelização e precisão preditiva impressionante. Conforme descrito por Chen e Guestrin (2016), o *Xgboost* é um conjunto de classificação e regressão Árvores (CART) $T_1(x_i, y_i) \dots T_k(x_i, y_i)$ onde x_i é o conjunto de descritores de treinamento associado a uma molécula para prever o rótulo da classe,y_i. Dado que um CART atribui uma pontuação real a cada licença (resultado ou alvo), as pontuações de previsão para CART individual são resumidas para obter a pontuação final e avaliadas através das funções aditivas K.

3.9 Regressão Logística

A regressão logística, segundo Johnson e Wichern (1992) é uma abordagem para classificação em que algumas ou todas as variáveis são qualitativas. Deixe a variável resposta Y ser 1 se a unidade de observação pertencer à população 1 e 0 se pertencer à população 2. Depois que uma função de regressão logística for estabelecida, usando conjuntos de treinamento para cada uma das duas populações, podemos proceder à classificação. É difícil incorporar *prioris* e custos na análise, portanto a regra de classificação se torna: Atribua **z** à população 1 se o razão de probabilidade estimada for maior que 1 ou

$$P(Y = y|\mathbf{x}) = exp(\hat{\beta}_0 + \hat{\beta}_1 z_1 + \cdots + \hat{\beta}_p z_k) > 1$$

Equivalentemente, temos a regra discriminante linear simples. Atribua **z** à população 1 se o razão de probabilidade estimada for maior que 0 ou

$$ln\frac{\hat{p}(z)}{1-\hat{p}(z)} = \hat{\beta}_0 + \hat{\beta}_1 z_1 + \cdots + \hat{\beta}_r z_r > 0$$

4 Avaliação de desempenho

Neste capítulo, são apresentadas brevemente algoritmos para avaliação de desempenho dos classificadores apresentados anteriormente. É útil avaliar o desempenho do modelo no conjunto de teste utilizando medições, pois analisar essas medidas, permite uma avaliação justa.

4.1 Taxa de erro

A taxa de erro de classificação é representado por uma estimativa para a probabilidade de má classificação. Como classificadores são construídos para minimizar a perda 0-1 (na qual perdemos uma unidade sempre que cometemos um erro de classificação), uma medida natural para avaliar um classificar é sua taxa de erro:

$$\text{Taxa de erro} = \frac{\text{Número de classificações errôneas}}{\text{Total de classificações}} \tag{10}$$

4.2 Validação cruzada

Segundo, Horta (2010), a validação cruzada é uma técnica de reamostragem que permite que todos os dados da base de dados sejam utilizados para treinamento e teste. Na validação cruzada os dados iniciais são particionados aleatoriamente em v subconjuntos $D_1, D_2, ..., D_v$ aproximadamente iguais em tamanho. Este procedimento é repetido v vezes de modo que cada partição seja usada para teste exatamente uma vez. O erro total é obtido pela soma dos erros de todas as v execuções.

4.3 Curva ROC e AUC

Receiver Operating Characteristics (ROC) e Area Under the Curve (AUC) são duas métricas utilizadas para medir o desempenho de classificadores. Um gráfico bidimensional de características de operação de receptor (ROC) é uma abordagem para exibir, organizar e selecionar classificadores com base em seus desempenhos, através da compensação entre taxa de positivo verdadeiros e taxa de positivos falsos dos classificadores. Curva ROC, pode ser definido como, taxa de positivos verdadeiros (do inglês, *true positive rate* ou TPR), desenhada no eixo y e taxa de negativos falsos (do inglês, *false positive rate* ou FPR), desenhada no eixo x, segue como calcular as taxas:

$$TPR = \frac{\text{Correções positivas do classificador}}{\text{Total positivos}}, \tag{11}$$

$$FPR = \frac{\text{Correções negativas do classificador}}{\text{Total negativas}}. \tag{12}$$

A área da curva ROC (AUC), é um número entre 0 a 1 que nos fornece outra forma de avaliar qual classificador, em média, é melhor. Se o modelo for perfeito, então sua área sob a curva ROC seria igual a 1. Para um modelo ser estritamente melhor do que outro terá que ter uma área maior sob a curva ROC.

A TPR também pode ser considerada como sensibilidade, verdadeiro positivo. Bem como existe a especificidade, verdadeiro negativo, conhecida como taxa de negativos verdadeiros (do inglês, *True negative rate* ou TNR), e a acurácia é a probabilidade de fornecer resultados corretos. Abaixo como calcular a TNR:

$$FPR = \frac{\text{Correções falso verdadeira do classificador}}{\text{Total negativas}} \tag{13}$$

5 Resultados

5.1 Banco de dados

O DOCSIS (do inglês, *Data Over Cable Service Interface Specification*) é um padrão internacional que inclui uma estrutura completa de comunicação para Internet e TV à cabo, multicamadas e bidirecional para transmissão de dados. Neste caso, como se trata de uma transmissão de cabo bidirecional, o provedor de serviço recebe e transmite informações, o canal de transmissão que o provedor envia para o cliente é chamado de *Downstream* e o canal que recebe informação é chamado de *Upstream*. O provedor controla e coordena essas informações através do CMTS (do inglês, *Cable Modem Termination System*), nele o provedor habilita os serviços de dados da alta velocidade e fornece cobertura para vários clientes finais, dependendo da sua configuração.

Tendo um conjunto de teste de vinte e cinco aparelhos roteadores *wireless* modelo TC7337, produtos fabricados em uma empresa do Pólo de Produtos Elétricos e Eletrônicos da Zona Franca de Manaus, foram realizados testes nas três antenas, em oito canais, totalizando 24 testes. Na Figura 1, um exemplo de roteador:

Figura 1: Exemplo de aparelho *wireless* em estudo.

As informações são coletadas pelo IQxel-160, um equipamento que realiza testes de calibração nos aparelhos, e são geradas por um software desenvolvido pela própria empresa, que fornece os *logs* e arquivos em formato de texto (.txt) que possuem um cabeçalho que contém alguns dados a respeito do aparelho: modelo, versão do log e data e hora de quando o teste foi realizado. Na Figura 2, segue processo desde o CMTS até os arquivos em .txt que serão utilizados neste trabalho. Na Figura 3 podemos ver um exemplo do cabeçalho que encontra-se nos arquivos .txt. Ao todo, foram coletados 874 *logs*, das seguintes falhas “Cabo Curto Blindagem - Conectado ao DUT (Device under test)”, “Cabo Desencapado Sem Blindagem - Conectado ao DUT” e “Blindagem Danificada”, bem como *logs* de equipamentos sem defeitos para ser a classe controle.

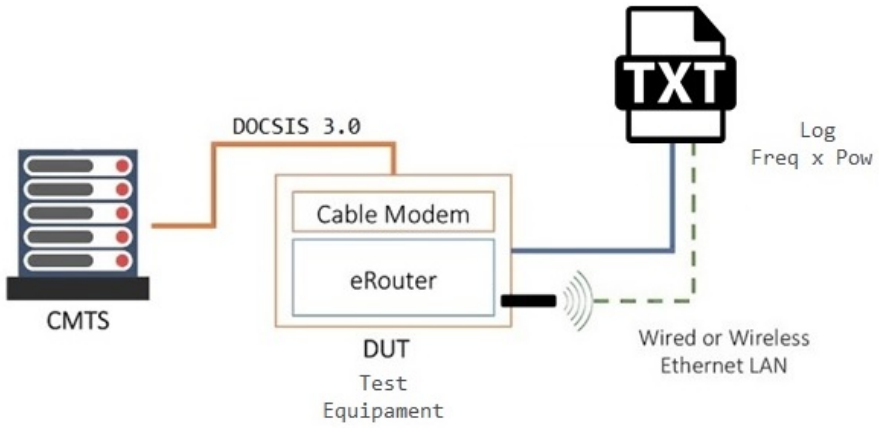


Figura 2: Fluxo de informações.

```
//*************************************************************************
//                DUT Setup file FGR1
//*************************************************************************
//
//     Copyright (c) 2016 All Rights Reserved
//
//*************************************************************************

Modelo TC7337
Date
2016-05-10-07:55:23
Ver 7.56.1
```

Figura 3: Exemplo de cabeçalho de um log.

5.2 Ferramenta Rminer

Segundo Li et al. (2013), o `rminer` é uma instrumento que fornece uma plataforma para fazer experimentos de comparação em algoritmos escolhidos. Uma biblioteca de código aberto para a linguagem de programação `R`, mais utilizado em tarefas de classificação oferece uma interface que pode ser usada para implementação de funções para mineração de dados, facilitando o uso desses algoritmos. Os usuários podem montar um próprio algoritmo ou selecionar dentre os 16 métodos de classificação e 18 métodos de regressão na biblioteca [Cortez, 2019]. No presente trabalho, foram utilizados os métodos de classificação citados no capítulo 3.

5.2.1 Exemplo de aplicação

O exemplo abaixo, é apenas uma pequena demonstração de como funciona o pacote `rminer`. Na variável H escolhemos a proporção de divisão para o conjunto treinamento, neste caso 0.7. Depois, ajustamos um modelo supervisionado de mineração de dados para classificação, com os modelos ksvm (M), *task* = *prob* ou seja, classificação com probabilidades de saída (ou seja, a soma de todas saídas é igual a 1), obtemos sigma = 1,4795. Na função *predict*, obtemos uma previsão para cada classe. Após fazermos isso, é calculado a área sob a curva ROC, igual a 0.9993, concluindo que o ksvm é um bom modelo de classificação para esses dados.

```
> library(rminer)
> data(iris)
> H=holdout(iris$Species,ratio=0.70)
> M=fit(Species~.,iris[H$tr,],model="ksvm", task="prob")
> M
An object of class "model"
Slot "formula":
Species ~ .
Slot "model":
[1] "ksvm"
Slot "task":
[1] "prob"
Slot "mpar":
$kernel
[1] "rbfdot"
$kpar
$kpar$sigma
[1] 1.479543
$C
[1] 1
Slot "attributes":
[1] 1 2 3 4 5
Slot "scale":
[1] "default"
Slot "transform":
[1] "none"
Slot "created":
[1] "2020-02-19 20:40:18"
Slot "time":
```

```
elapsed
0.29
Slot "object":
Support Vector Machine object of class "ksvm"
SV type: C-svc  (classification)
parameter : cost C = 1
Gaussian Radial Basis kernel function.
Hyperparameter : sigma =  1.47954271717376
Number of Support Vectors : 62
Objective Function Value : -5.6833 -6.4247 -16.9615
Training error : 0.019048
Probability model included.
Slot "outindex":
[1] 5
Slot "levels":
[1] "setosa"     "versicolor" "virginica"
> P=predict(M,iris[H$ts,]) # classes
> head(P)
setosa  versicolor  virginica
[1,] 0.9794109 0.008915126 0.01167394
[2,] 0.9748116 0.010844586 0.01434381
[3,] 0.9350102 0.027142041 0.03784774
[4,] 0.9145935 0.035216638 0.05018986
[5,] 0.6556457 0.135214767 0.20913955
[6,] 0.8405008 0.064432939 0.09506623
> print(mmetric(iris$Species[H$ts],P,"AUC"))
setosa
0.9992593
```

5.3 Desenvolvimento da ferramenta

Com a finalidade de facilitar a classificação dos dados foi desenvolvido um *web app* interativo para o *software* `R`, com o auxílio do pacote `Shiny`, estruturado em três módulos: carregamento de dados, aprendizado e classificação.

No primeiro módulo "Carregamento de dados", Figura 4 é permitido o envio dos arquivos, para criação das classes de controle e de falhas para treinamento. Após o carregamento dos *logs*, podemos nomear a nova classe de defeitos.

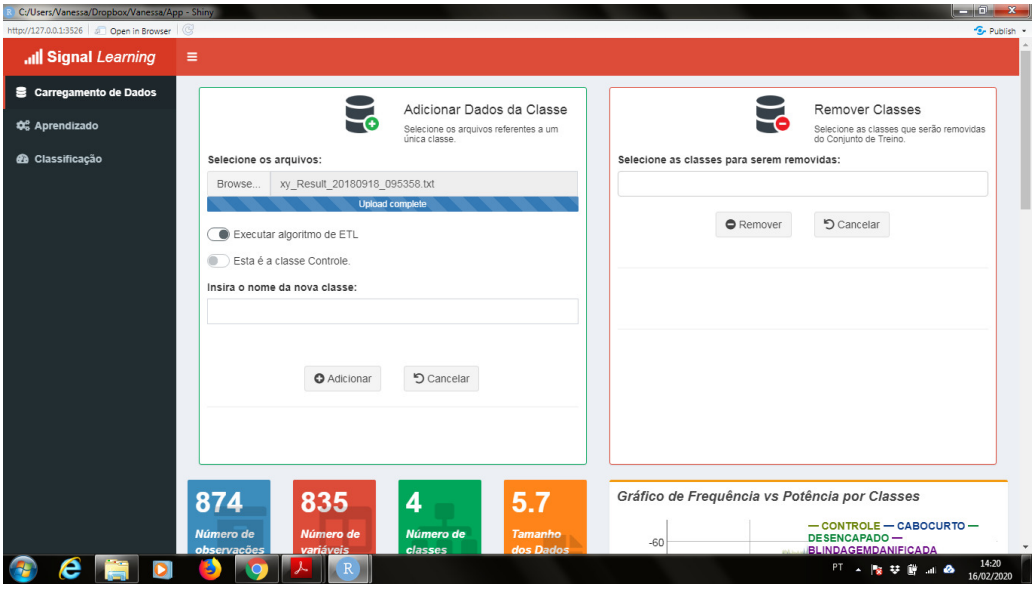


Figura 4: Módulo Carregamento de Dados - parte 1.

Logo abaixo, o *web app* permite que o usuário analise quais as principais diferenças entre a nova falha e as falhas já cadastradas no sistema. Informações gerais no canto superior esquerdo da Figura 5, informam ao usuário sobre as características dos *logs* analisados, além de dois gráficos que descrevem como a nova classe se adequa frente as classes já presentes no sistema; "Current Classes Pie-Chart" descreve a proporção entre os *logs* e em "Euclidian Similarity of classes" é possível determinar quais falhas possuem grande correlação. Em "frequência x potência" temos as classes sendo plotadas de acordo com o sinal que foi analisado no *upstream* e *downstream*.

Figura 5: Módulo Carregamento de Dados - parte 2.

No segundo módulo "Aprendizado", Figura 6, está dividido em duas etapas, "Aprendi-

zado", Figura 7, onde o usuário escolhe os classificadores. Desta forma, é possível ajustar alguns parâmetros a fim de determinar a melhor forma de aprendizagem do sistema.

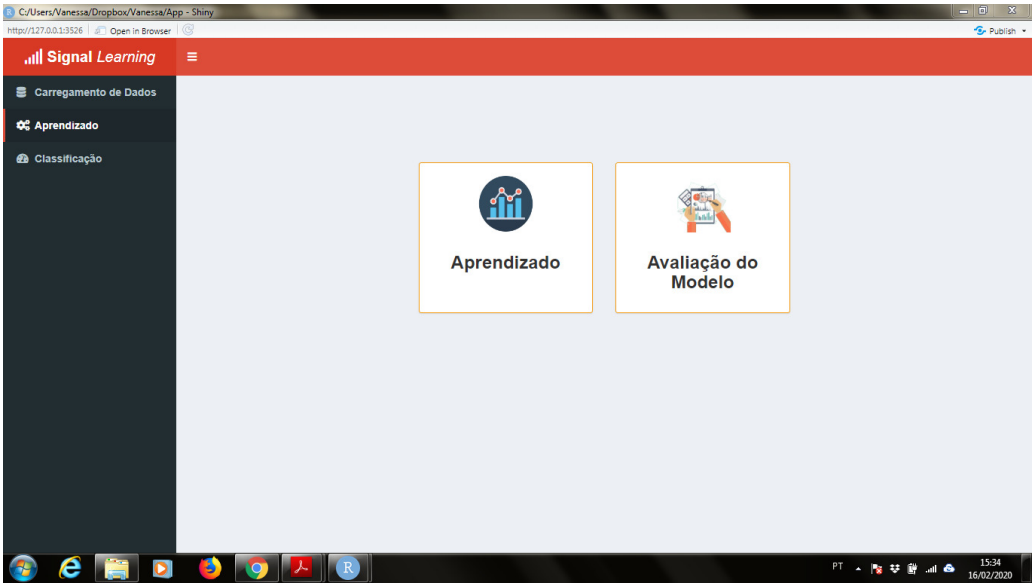


Figura 6: Módulo Aprendizado.

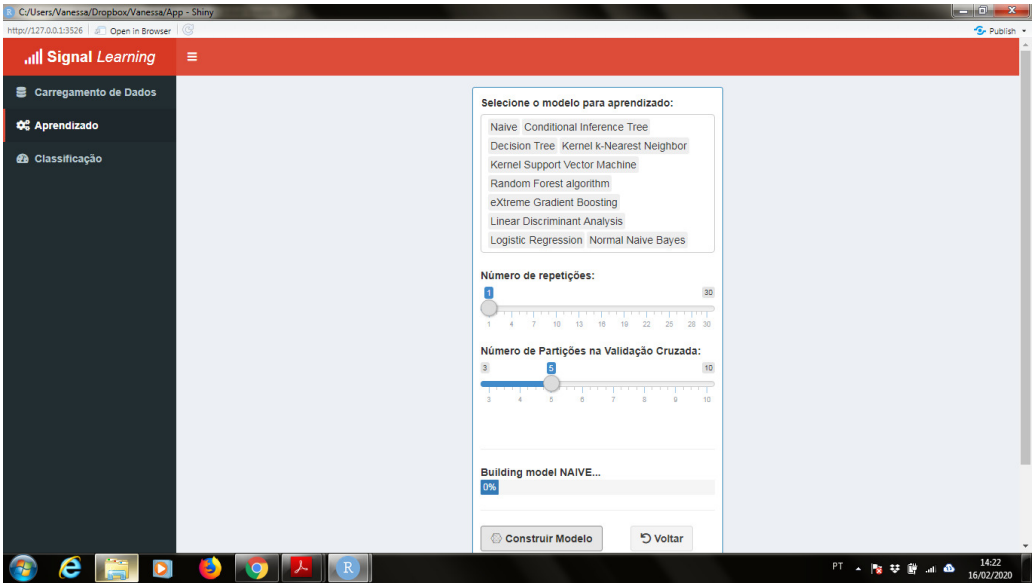


Figura 7: Módulo Aprendizado - Etapa Aprendizado.

Após o processo de aprendizagem é possível avaliar o desempenho dos testes por classificador estatístico, com base métricas como curva ROC, área sob a curva ROC e acurácia, conforme Figura 8, e na tabela de métricas, conforme Figura 9, todas contidas na segunda etapa que é "Avaliação do Modelo", e assim pode-se escolher qual classificador deseja utilizar para prever os casos de falha.

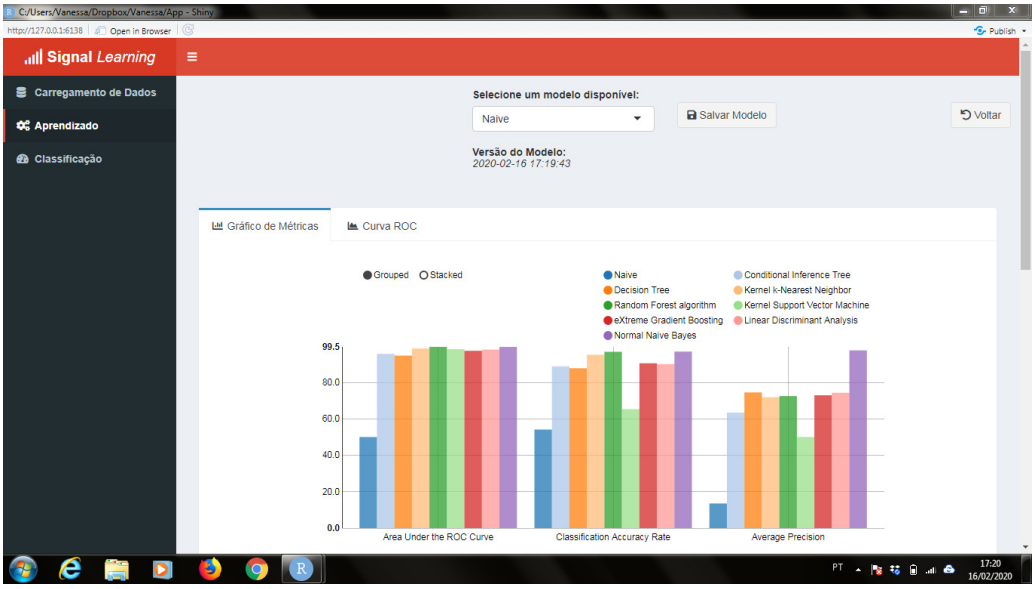
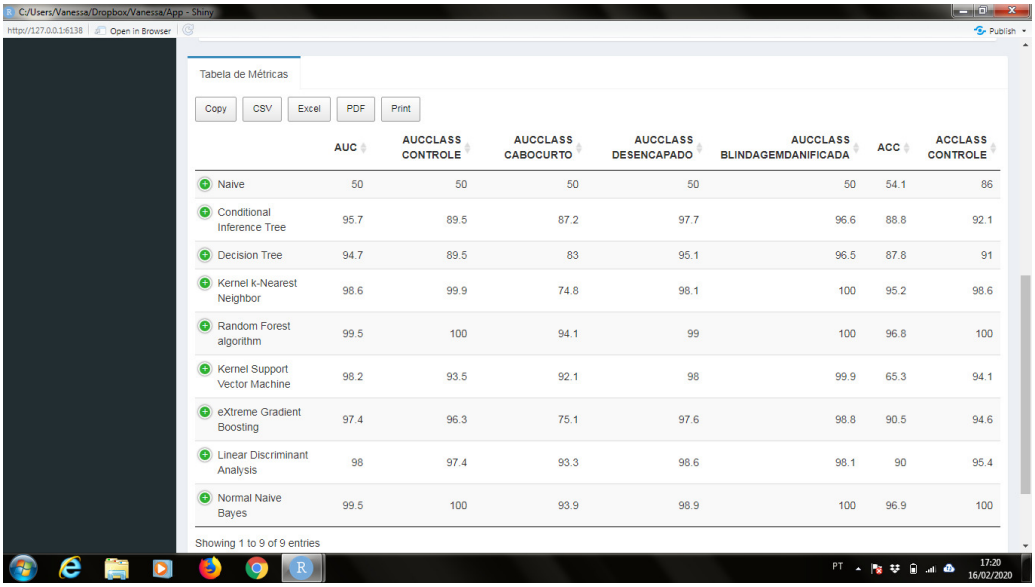


Figura 8: Módulo Aprendizado - Etapa Avaliação do Modelo (Gráfico de métricas).



	AUC	AUCCLASS CONTROLE	AUCCLASS CABOCURTO	AUCCLASS DESENCAPADO	AUCCLASS BLINDAGEMDANIFICADA	ACC	ACCLASS CONTROLE
Naive	50	50	50	50	50	54.1	86
Conditional Inference Tree	95.7	89.5	87.2	97.7	96.6	88.8	92.1
Decision Tree	94.7	89.5	83	95.1	96.5	87.8	91
Kernel k-Nearest Neighbor	98.6	99.9	74.8	98.1	100	95.2	98.6
Random Forest algorithm	99.5	100	94.1	99	100	96.8	100
Kernel Support Vector Machine	98.2	93.5	92.1	98	99.9	65.3	94.1
eXtreme Gradient Boosting	97.4	96.3	75.1	97.6	98.8	90.5	94.6
Linear Discriminant Analysis	98	97.4	93.3	98.6	98.1	90	95.4
Normal Naive Bayes	99.5	100	93.9	98.9	100	96.9	100

Figura 9: Módulo Aprendizado - Etapa Avaliação do Modelo (Tabela de métricas).

No terceiro módulo “Classificação”, Figura 10, é permitido o envio dos arquivos para classificação de acordo com o modelo escolhido no módulo “Aprendizado”. Ainda em fase de construção, conforme a Figura 11, após o carregamento de dados no terceiro módulo, teremos a previsão em qual classes melhor se adequam, de acordo com as classes criadas no primeiro módulo.

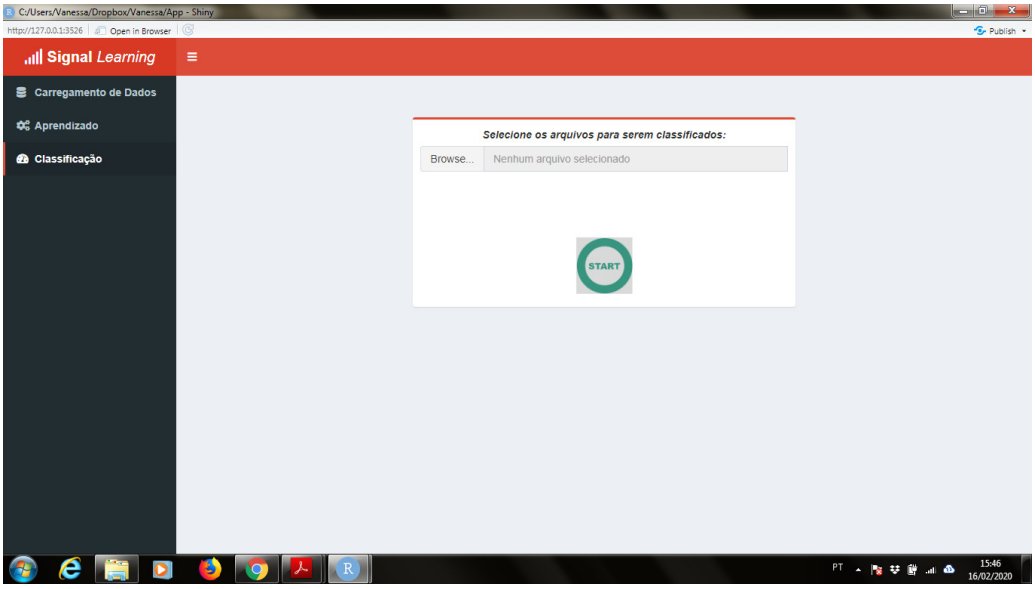


Figura 10: Módulo Classificação.

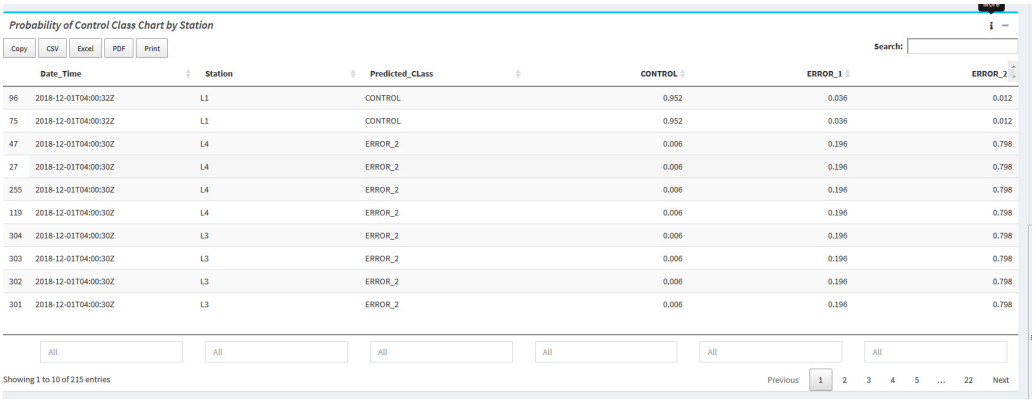


Figura 11: Módulo Classificação - Resultado.

5.4 Avaliação do modelo

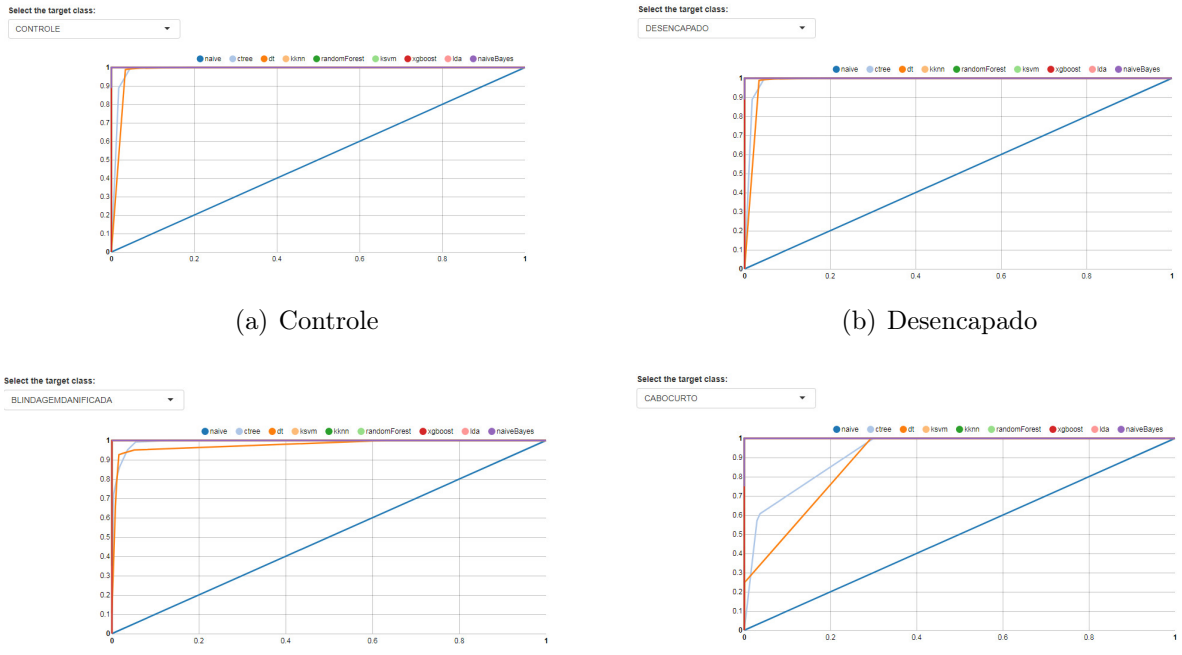
Um bom modelo de classificação, deve estar localizado o mais próximo quanto possível do vértice superior esquerdo do gráfico, conforme Figura 12, pode-se ver que o modelo *Naive* executa suposições aleatórias, pois se iguala a 0,5 em todos os gráficos e na área sob a curva ROC é igual a 50, conforme Tabela 1. Como pela curva ROC é difícil enxergar com precisão, de acordo com a análise dos resultados obtidos da área da curva ROC, apesar de que para as falhas "Cabo Curto" e "Cabo Desencapado" o *Random Forest* é considerado o melhor, pode-se constatar que para as falhas selecionadas é equivalente utilizar o *Random Forest algorithm* ou *Normal Naive Bayes*.

Na Tabela 2, percebe-se que o classificador com maior acurácia para os dados em geral, é o *Normal Naive Bayes*, seguido do *Random Forest*. A acurácia é a proximidade de um resultado com o seu valor de referência real. Nas Tabelas 3 e 4, pode-se ver as

taxas de sensibilidade e especificidade, respectivamente. Nesse caso, o modelo *Naive* é o menos capaz de identificar corretamente as falhas, já o *Random Forest* e *Normal Naive Bayes* são mais capazes de identificar corretamente tanto as falhas quanto as não falhas. Considerando que todos os modelos são capazes de identificar corretamente os roteadores que não possuem falhas.

	AUC	AUC CONTROLE	AUC CABO CURTO	AUC DESENCAPADO	AUC BLINDAGEM DANIFICADA
Naive	50	50	50	50	50
Conditional Inference Tree	95,7	89,5	87,2	97,7	96,6
Decision Tree	94,7	89,5	83	95,1	96,5
Kernel k-Nearest Neighbor	98,6	99,9	74,8	98,1	100
Random Forest algorithm	99,5	100	94,1	99	100
Kernel Support Vector Machine	98,2	93,5	92,1	98	99,9
eXtreme Gradient Boosting	97,4	96,3	75,1	97,6	98,8
Linear Discriminant Analysis	98	97,4	93,3	98,6	98,1
Normal Naive Bayes	99,5	100	93,9	98,9	100

Tabela 1: Resultados da área da curva ROC em %.



(a) Controle

(b) Desencapado

(c) Blindagem danificada

(d) Cabo curto

Figura 12: Curva ROC das falhas de acordo com os classificadores.

	ACC	AC CONTROLE	AC CABOCURTO	AC DESENCAPADO	AC BLINDAGEM DANIFICADA
Naive	54,1	86	96,8	71,3	54,1
Conditional Inference Tree	88,8	92,1	96,8	95,5	93,1
Decision Tree	87,8	91	96,8	94,4	93,4
Kernel k-Nearest Neighbor	95,2	98,6	96,6	96,6	98,6
Random Forest algorithm	96,8	100	96,8	96,8	100
Kernel Support Vector Machine	65,3	94,1	71,3	71,3	94,1
eXtreme Gradient Boosting	90,5	94,6	95,9	95,4	95,1
Linear Discriminant Analysis	90	95,4	94,6	94,6	95,4
Normal Naive Bayes	96,9	100	96,9	96,9	100

Tabela 2: Resultados da acurácia em %.

	TPR CONTROLE	TPR CABO CURTO	TPR DESENCAPADO	TPR BLINDAGEM DANIFICADA
Naive	0	0	0	100
Conditional Inference Tree	70,5	0	98,4	93,7
Decision Tree	73	3,6	92,8	93,9
Kernel k-Nearest Neighbor	90,2	0	99,2	100
Random Forest algorithm	100	0	100	100
Kernel Support Vector Machine	57,4	100	0	100
eXtreme Gradient Boosting	72,1	10,7	95,2	97,5
Linear Discriminant Analysis	88,5	39,3	88	94,5
Normal Naive Bayes	100	3,6	100	100

Tabela 3: Resultados para TPR de cada classificador em %.

	TNR CONTROLE	TNR CABO CURTO	TNR DESENCAPADO	TNR BLINDAGEM DANIFICADA
Naive	100	100	100	0
Conditional Inference Tree	95,6	100	94,4	92,5
Decision Tree	93,9	99,9	95	92,8
Kernel k-Nearest Neighbor	100	99,8	95,5	97
Random Forest algorithm	100	100	95,5	100
Kernel Support Vector Machine	100	70,3	100	87
eXtreme Gradient Boosting	98,3	98,7	95,5	92,3
Linear Discriminant Analysis	96,5	96,5	97,3	96,5
Normal Naive Bayes	100	100	95,7	100

Tabela 4: Resultados para FPR de cada classificador em %.

6 Conclusão

Ao longo deste trabalho, o leitor foi levado a alguns conceitos para definições dos classificadores, em que propôs-se utilizá-los para detectar as falhas no sinal de aparelhos roteadores *wireless*, comparar o desempenho dentre eles, assim como desenvolver uma ferramenta para esse monitoramento. Após a utilização do pacote `rminer` para classificação e avaliação dos modelos em estudo, chegou-se a uma escolha consistente e notou-se que os melhores classificadores a serem utilizados nos dados desse experimento são *Random Forest* e *Naive Bayes Normal*, já o modelo *Naive* apresentou resultados não tão bons. Além disso também foi desenvolvida uma ferramenta, *web app*, que possibilita a visualização e seleção do melhor classificador para os dados que o usuário escolher, de maneira fácil e útil para quem o utilizar. A etapa de monitoramento ainda precisa ser melhorada, mas nas demais etapas a ferramenta cumpriu o desejado.

Referências

[Aha et al., 1991] AHA, David; KIBLER, Dennis; ALBERT, Marc. Instance-Based Learning Algorithms, Machine Learning, Volume 6, Issue 1, pp 37–66, 1991.

[Babin et al., 2011] BABIN, Gilbert; STANOEVSKA-SLABEVA, Katarina; KROPF, Peter.E-Technologies: Transformation in a Connected World: 5th International Conference, MCETECH 2011, Les Diablerets, Switzerland, January 23-26, 2011, Revised Selected Papers, Springer Science & Business Media, 2011.

[Breiman,1996] BREIMAN, L. Bagging Predictors. Machine Learning, 1996. v. 24, n. 2, p. 123–140.

[Chen e Guestrin,2016] CHEN, Tianqi; GUESTRIN, Carlos. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Association for Computing Machinery, New York, NY, USA, 785–794,2016.

[Cortez, 2019] CORTEZ, Paulo. Package "rminer", CRAN R Project, 2019.

[Dietterich, 2000] DIETTERICH, Thomas G. An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting and Randomization, Machine Learning 1-22.

[Duda et al., 2001] DUDA, Richard O.; HART, Peter E.; STORK, David G. Pattern classification. 2nd ed. New York: John Wiley e Sons, c2001.

[Fix e Hodges,1951] E. Fix; J.L. Hodges (1951): An Important Contribution to Nonparametric Discriminant Analysis and Density Estimation: Commentary on Fix and Hodges,1951.

[Haykin, 1995] HAYKIN, Simon. Kalman filtering and neural networks. John Wiley & Sons, Pag 349, 2001.

[Horta, 2010] HORTA, Rui Américo Mathiasi. Uma metodologia de mineração de dados para a previsão de insolvência de empresas brasileiras de capital aberto/ Rui Américo Mathiasi Horta. – Rio de Janeiro: UFRJ/COPPE, 2010.

[Hothorn e Hornik, 2015] HOTHORN, Torsten; HORNIK; Kurt. ctree : Conditional Inference Trees, 2015.

[Johnson e Wichern, 1992] JONHSON, Richard A.;WICHERN, Dean W. Applied Multivariate Statistical Analysis. Englewood Cliffs, N.J.: Prentice Hall, 1992.

[Karatzoglou et al., 2019] KARATZOGLOU, Alexandros; SMOLA, Alex; HORNIK, Kurt. Package "kernlab", CRAN R Project, 2019.

[Li et al., 2013] RUIXAN, Li; HUAQING, Li; Wei, Wang; XIAOPU, Ma; XIWU, Gu. RMiner: A Tool Set for Role Mining, in Proceedings of the 18th ACM symposium on Access control models and technologies (SACMAT'13). Association for Computing Machinery, New York, NY, USA, 193–196, 2013.

[Meyer,1969] MEYER, Paul L. Probabilidade - Aplicações à Estatística, JC Livros Técnicos e científicos, 2ª edição, pag 49, 1969.

[Moreira, 2003] MOREIRA, Maurício A. Fundamentos de Sensoriamento Remoto e Metodologias de Aplicação. 1st ed. São José dos Campos : Instituto Nacional de Pesquisas Espaciais (1NPF.), 2001.

[Ross, 2010] ROSS, Timothy J.Fuzzy logic with engineering applications. 3rd ed. University of New Mexico, USA, 2010.

[Sarda-Espinosaa et al., 2017] SARDA-ESPINOSAA, Alexis; SUBBIAHA, Subanatarajan; BARTZ-BIELSTEINB, Thomas. Conditional inference trees for knowledge extraction from motor health condition data. Engineering Applications of Artificial Intelligence,pag 26–37, 2017.

[Tan et al., 2009] TAN, Pang-Ning; STEINBACH, Michael; KUMAR, Vipin. Introdução ao DATAMINING Mineração de Dados. Rio de Janeiro: Editora Ciência Moderna Ltda, 2009

[Theodoridis and Koutroumbas, 2006] THEODORIDIS, Sergios; KOUTROUMBAS, Konstantinos. Pattern recognition. 3nd ed. Amsterdam;: Boston: Academic Press, 2006.

[Vapnik e Cortes, 1995] CORTES, Corinna; VAPNIK, Vladimir. Support-vector networks. Machine Learning,Volume 20, Issue 3, 273–297, 1995.

[Ye et al., 2004] YE, Jieping; JANARDAN, Ravi; LI, Qi. Two-Dimensional Linear Discriminant Analysis, 2004.