AMÉRICO ANTONIO COSENTINO JÚNIOR

REDUÇÃO DA INDISPONIBILIDADE E DO CUSTO DOS SERVIÇOS DE TI UTILIZANDO INTELIGÊNCIA ARTIFICIAL NO GERENCIAMENTO DE INCIDENTES

Monografia apresentada ao Programa de Educação Continuada da Escola Politécnica da Universidade de São Paulo, para obtenção do título de Especialista, pelo Programa de MBA USP Tecnologias Digitais e Inovação Sustentável.

SÃO PAULO

2020

AMÉRICO ANTONIO COSENTINO JÚNIOR

REDUÇÃO DA INDISPONIBILIDADE E DO CUSTO DOS SERVIÇOS DE TI UTILIZANDO INTELIGÊNCIA ARTIFICIAL NO GERENCIAMENTO DE INCIDENTES

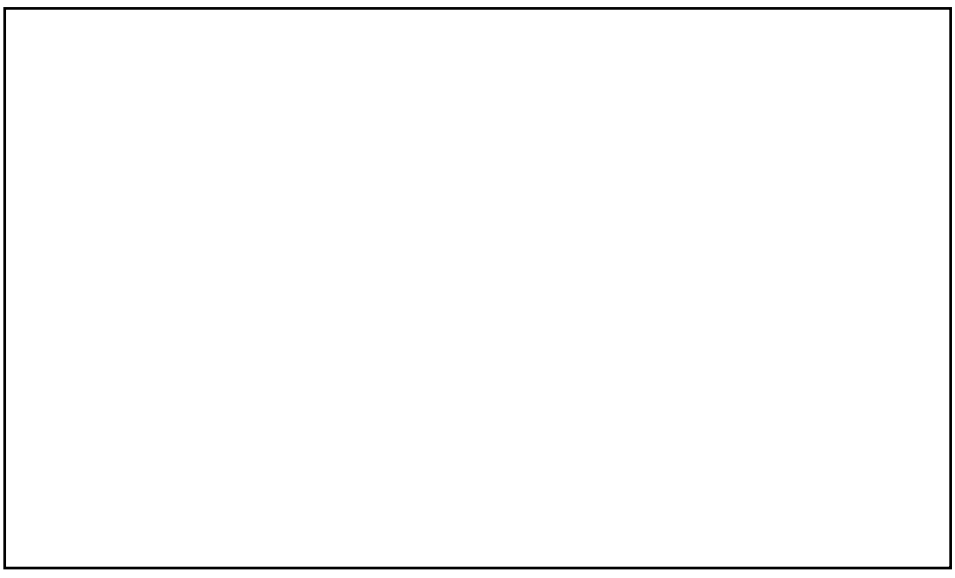
Monografia apresentada ao Programa de Educação Continuada da Escola Politécnica da Universidade de São Paulo, para obtenção do título de Especialista, pelo Programa de MBA USP Tecnologias Digitais e Inovação Sustentável.

Orientador: Prof. Dr. André Aguiar Santana

SÃO PAULO

2020

FICHA CATALOGRÁFICA



AGRADECIMENTOS

À minha esposa, Tereza, e aos meus filhos, Daniel e Bianca, pelo apoio na conclusão deste programa.

Aos meus pais, Sr. Américo e Sra. Wanda, por todo esforço e dedicação para que concluísse mais esta etapa na minha formação.

Ao professor André Aguiar Santana pela disponibilidade e colaboração para realização dessa monografia.

RESUMO

As organizações sofrem ao garantir a disponibilidade e a qualidade dos serviços de Tecnologia da Informação (TI) em um ambiente de mudanças recorrentes. Mudanças que causam o crescimento dos números de incidentes atendidos pelas áreas de suporte à TI e, consequentemente, aumentam a indisponibilidade e os custos dos serviços. O presente estudo propõe a utilização de Inteligência Artificial aplicando algoritmos de aprendizado de máquina no fluxo de Gerenciamento de Incidentes (GI) para redução da indisponibilidade e custo dos serviços de TI. Baseado em pesquisas bibliográficas, foi criado um método para classificação de cumprimento de acordo de nível de serviço, *service-level agreement* (SLA) em inglês, a partir dos dados iniciais do incidente para facilitar a priorização do incidente. O método foi aplicado em um conjunto de dados de atendimento real e atingiu a acurácia de aproximadamente 81,2%. Demonstrando que é possível substituir a classificação manual de priorização utilizando o modelo criado para redução de horas de equipe de suporte e de tempo médio de reparo. Dessa maneira, reduzindo o custo e aumentando a disponibilidade dos serviços de TI.

Palavras-chave: Redução de Indisponibilidade, Redução de Custo, Serviços de TI, Inteligência Artificial, Gerenciamento de Incidente.

ABSTRACT

Organizations provide by ensuring the availability and quality of Information Technology (IT) services in an environment of recurring changes. Changes that cause the increase in the number of incidents attended by the IT support areas and, consequently, increase the unavailability and costs of services. This study proposes the use of Artificial Intelligence by applying machine learning algorithms in the Incident Management (IM) flow to reduce the unavailability and cost of IT services. Based on bibliographic research, a method for classifying compliance with a service level agreement (SLA) was created from the initial incident data to facilitate prioritization of the incident. The method was applied to a set of real care data and reached an accuracy of approximately 81.2%. Demonstrating that it is possible to replace a manual prioritization classification using the model created to reduce hours of support staff and average repair time. Thus, the cost and increased availability of IT services.

Keywords: Downtime Reduction, Cost Reduction, IT Services, Artificial Intelligence, Incident Management.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE GRÁFICOS

SUMÁRIO

1 INTRODUÇÃO

Conforme Kotter (2015), as organizações sofrem ao acompanhar o ritmo acelerado das transformações globais, consequentemente, seus serviços de Tecnologia da Informação (TI) são afetados por essas mudanças e inovações. Desta maneira, é crescente a preocupação das organizações com a velocidade das inovações, pois, a relação com seus clientes depende da concepção de novas formas digitais de interação. (WEILL; WOERNER, 2019)

As organizações, alinhadas com este novo tipo de relação com os clientes, precisam garantir a disponibilidade e a qualidade dos serviços oferecidos neste ambiente volátil e, ainda, otimizar os custos das áreas de TI responsáveis pela sustentação dos serviços, deste modo, tornando a gestão dos serviços um dos principais processos para a sobrevivência do negócio.

Diante desse cenário, o presente estudo propõe reduzir o tempo de indisponibilidade dos serviços de TI e os custos das equipes pertencentes ao processo de Gerenciamento de Incidente (GI), utilizando Inteligência Artificial para possíveis pontos de automação no fluxo deste processo.

O *Service Desk* - ou Central de Serviços, em português – é uma área participante do GI e deve ser o único ponto de contato entre o provedor de serviços e o usuário, gerenciando incidentes, atendendo solicitações e comunicando os usuários. Esta área é dividida em níveis de equipes de suporte, Suporte Nível 1 (N1), Suporte Nível (N2), até a equipe Nível *n,* onde *n* representa o número máximo de níveis estipulado pela organização.

A equipe de Suporte N1 é responsável pelo pronto atendimento e resolução dos incidentes reportados. Caso o N1 não consiga resolver o problema por conta própria, deve escalonar o incidente de acordo com a categoria ao time de Suporte N2, assim, sucessivamente até a resolução do problema ou atingir o último nível de suporte. Este fluxo de atendimento deve cumprir as expectativas e obrigações mínimas do processo de GI são definidos contratos entre o provedor e o cliente do serviço de TI, chamados de Acordo de Nível de Serviço, ou *Service Level Agreement* (SLA) em inglês.

Em busca de facilitar o atendimento essas equipes de suporte utilizam ferramentas especializadas para o registro e tratamento dos incidentes, onde são apresentadas informações básicas da ocorrência como a data e hora do incidente, prioridade, impacto, histórico de ações em superações, até a data e hora de resolução.

Para atingir a finalidade do estudo, será criado um método para auxiliar o Suporte N1 na classificação e priorização do incidente utilizando os dados históricos, em busca de reduzir o tempo de atendimento, consequentemente, diminuindo o custo da equipe e aumentando a disponibilidade dos serviços.

1.1 Motivação

O ambiente de Operações de TI, em que a mudança é recorrente, tornou-se complexo devido à grande quantidade de dados, plataformas híbridas e uma tendência de crescimento no número de incidentes em um cenário de mudanças recorrentes.

Conforme pesquisa da Forrester Consulting (2013), 19% dos entrevistados relataram que a cada 10 mudanças, mais de 4 causam incidentes e impactam de alguma forma os clientes e a operação do negócio. Outro número alarmante apontado na pesquisa, é que cerca de 31% do público entrevistado não sabe se essas mudanças causam indisponibilidade.

Dado esse cenário, existe uma necessidade de resolver esses incidentes em um curto espaço de tempo para restaurar a qualidade e a disponibilidade do negócio no menor tempo possível e cumprindo os acordos de nível de serviços.

Porém, a grande quantidade de informações simultâneas utilizando múltiplas ferramentas em modificação constante, transformam as tarefas das equipes de suporte complexas e cada vez mais lentas, logo, demonstrando uma grande oportunidade de:

I. Utilizar tecnologias relacionadas à inteligência artificial;

II. Apoiar a equipe do N1 com informações relevantes que reduzam o tempo de atendimento dos incidentes, diminuindo o tempo na classificação, priorização e direcionamento para o time de N2 corretamente.

1.2 Objetivo

O objetivo deste presente trabalho consiste em reduzir o tempo de indisponibilidade e o custo dos serviços de TI, aplicando algoritmos de Inteligência Artificial nos dados das ferramentas de Gerenciamento de Serviços de TI (GSTI).

Para atingir esse objetivo, foram gerados modelos utilizando aprendizado de máquina para classificar e priorizar o incidente de maneira automatizada para reduzir tempo de processos repetitivos e agilizar o atendimento do Suporte N1.

1.3 Justificativa

Com o avanço do poder de processamento computacional e do aumento exponencial de disponibilidade de dados, conforme demonstrado no estudo de Hilbert e Lopez (2011), a utilização dos métodos de aprendizado de máquina podem ser uma alternativa para ganhos relevantes no tempo médio de reparo (MTTR) – do inglês *mean time to recovery* -, auxiliando a equipe de Suporte N1 na classificação, priorização e direcionamento dos incidentes.

1.4 Metodologia

Neste estudo foi desenvolvido um método para classificar a possibilidade de cumprimento do SLA com algoritmo de classificação utilizando aprendizado de máquina, baseado nas pesquisas bibliográficas sobre mineração de dados, gerenciamento de incidentes no GSTI, métodos de aprendizado de máquina e inteligência artificial para a Operação de TI.

Para validação do método foi realizada a pesquisa exploratória dos dados de atendimento de uma equipe de Operação de TI e aplicada a metodologia desenvolvida em um conjunto de dados real.

1.5 Organização do trabalho

Além deste capítulo, a monografia está dividida em mais 4 capítulos da seguinte forma:

I. Capítulo 2: Revisão bibliográfica, realizadas as pesquisas bibliográficas sobre Gerenciamento de Incidente no GSTI, Inteligência Artificial em Operações de TI e Aprendizado de Máquina para Classificação de Incidentes.
II. Capítulo 3: Método para classificação do SLA, desenvolvido um método para identificar incidentes com possibilidade de cumprimento do SLA a partir dos dados iniciais do incidente.
III. Capítulo 4 : Estudo de caso, aplicado o método em um conjunto de dados real e demonstrado os seus resultados.
IV. Capítulo 5 : Conclusão, relatadas as contribuições do estudo, uma breve recapitulação geral do trabalho, seus resultados e um direcionamento para pesquisas futuras.

2 REVISÃO BIBLIOGRÁFICA

Neste estudo foram abordados os assuntos associados ao gerenciamento de incidente (GI) e a classificação dos mesmos, onde foi pesquisado o fluxo do problema desde a sua origem até seu encerramento e complementada com a revisão sobre aplicação de inteligência artificial e aprendizado de máquina na área.

Baseado no levantamento, para aprofundamento da pesquisa o presente capítulo foi dividido nos temas Gerenciamento de Incidentes no GSTI e Inteligência Artificial para Operações de TI.

2.1 Gerenciamento de Incidentes no GSTI

O Gerenciamento de Serviços de Tecnologia da Informação (GSTI) utiliza uma abordagem orientada a processos que tem como objetivo *"estabelecer, implementar, manter e melhorar continuamente um sistema de gerenciamento de serviços"*, conforme definição da ISO/IEC 20000 (2018, parte 1, 3.5, tradução nossa),

Uma das bibliotecas de práticas de TI mais utilizadas como referência pelas organizações de todos os tipos e tamanhos pelo todo mundo é a *Information Technology Infrasructure Library* (ITIL), traduzido livremente como Biblioteca de Infraestrutura de Tecnologia da Informação. (BMC, 2016)

Em 2019 a ITIL foi atualizada para a versão 4 (AGUTTER, 2019), onde descreve um conjunto de boas práticas direcionadas ao Gerenciamento de Serviços desde a sua demanda até a geração de valor, fluxo ilustrado na figura 1.

Figura 1 - Cadeia de Valor de Serviço na ITIL 4

Fonte: ITIL (2019)

O ITIL 4 foi dividido em 3 grupos de práticas de Gestão, são elas a Geral, Serviços e Técnica, totalizando 34 práticas para cumprir o objetivo da organização. O processo de Gerenciamento de Incidente (GI) faz parte deste ciclo de vida e encontra-se inserido nas práticas de Gerenciamento de Serviço, conforme a figura 2, esse processo garante que a operação normal do serviço seja restaurada o mais rápido possível e causando o menor impacto possível nos negócios.

Figura 2 - Práticas de Gerenciamento de Serviços na ITIL 4

Fonte: BMC (2019)

O incidente é uma interrupção não planejada ou uma redução da qualidade de um serviço de TI, são registrados e, na sequência, são atribuídas as prioridades com

base no impacto no negócio e na urgência da falha, etapas demonstradas na figura 3 no fluxo de GI. (AGUTTER, 2019; BMC, 2019)

Figura 3 - Fluxo de Gerenciamento de Incidente

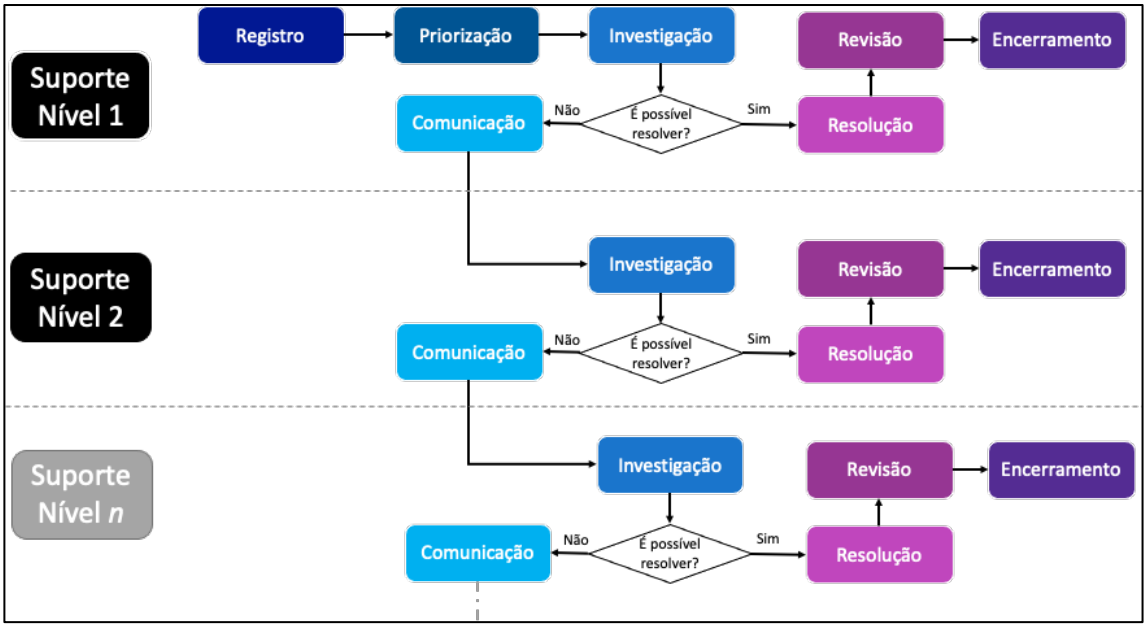
Fonte: BMC (2019)

Conforme o fluxo de GI, a equipe de Suporte N1 deve priorizar o registro e categorizá-lo e, após a investigação inicial, tem que resolvê-lo. Caso não possua a capacidade técnica para efetuar o diagnóstico ou a resolução, deve direcioná-lo ao Suporte N2 para que a equipe de especialistas possa investigá-lo, conforme a representação da figura 4, respeitando o acordo de nível de serviço.

A disponibilidade do negócio é uma das principais métricas utilizadas para garantir a efetividade do acordo de nível de serviço, ou *Service Level Agreement* (SLA) em inglês. O SLA é um contrato explícito ou implícito com seus usuários, com as consequências do cumprimento, ou não, dos objetivos contidos no mesmo. (MURPHY et al., 2016).

Conforme afirma Snow et al. (2010), a disponibilidade depende da confiabilidade e capacidade de manutenção, onde a confiabilidade é referida como Tempo Médio Entre Falhas (MTBF) – do inglês, *Mean Time Between Failures* - e a capacidade de manutenção como tempo médio para reparo (MTTR).

Figura 4 - Fluxo de Atendimento do *Service Desk*



Fonte: Elaborado pelo Autor

Desse modo, o *Service Desk* tem como objetivo a restauração da operação de maneira rápida e com a qualidade acordada no SLA. O fluxo de atendimento do incidente da área, representado pela figura 4, é um processo que pode possuir mais de uma equipe de suporte envolvida na resolução.

Portanto, o atendimento necessita de uma priorização adequada e uma classificação correta desde a sua origem para que não ocorra desperdício de tempo, assim, evitando penalidades financeiras, descontos de contratos, indisponibilidade de serviços ao cliente entre outras formas de prejuízos tanto para provedores de serviços internos quanto para externos.

2.2 Inteligência Artificial para Operação de TI

A Gartner (2016) criou o termo AIOps para o conceito de Inteligência Artificial (IA) para operações de TI, que surgiu como abreviação de *Algorithmic IT Operations* e, posteriormente, foi modificado para *Artificial Intelligence for IT Operations*.

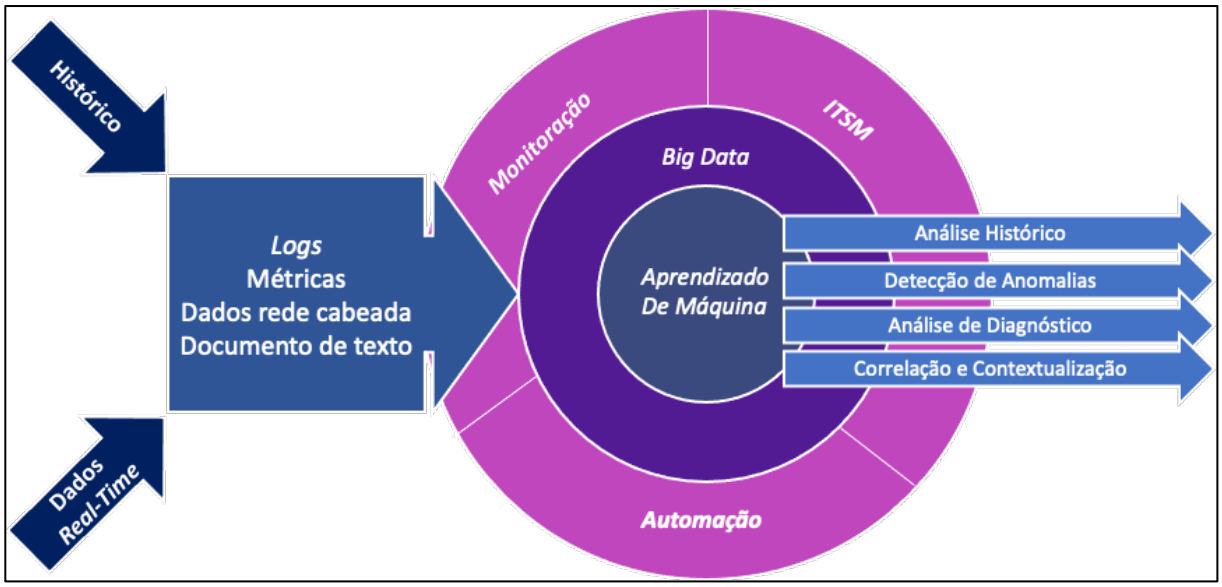
Definido pelo Vice-Presidente de Pesquisas da Gartner, o AIOPS é o processo que:

> [...] utiliza Big Data, aprendizado de máquinas e outras tecnologias avançadas para aprimorar direta e indiretamente as funções de Operação de TI (monitoramento,

> automação e central de atendimento)" onde "permitem o uso simultâneo de várias fontes de dados, métodos de coleta de dados, tecnologias analíticas. (em tempo real e profundas) e tecnologias de apresentação". (LERNER, 2017, tradução nossa)

Está representado na figura 5 o gerenciamento contínuo da área de operações no AIOps, onde espera-se que os dados históricos e gerados em tempo real sirvam como entradas do processo e que no final desse fluxo sejam obtidas as análises dos históricos e diagnósticos, detecções de anomalias, correlações e contextualizações dos dados inseridos utilizando aprendizado de máquina.

Figura 5 - AIOps - Gerenciamento contínuo de Operação de TI



Fonte: Gartner, Inc. (2018)

Segundo Dang et al. (2019), estamos nos primeiros passos do enfretamento do desafio de adotar soluções AIOps e no estudo foram identificadas 3 (três) lacunas:

I. Metodologias de inovação para criação de soluções
II. Necessidade de alterações na Engenharia de TI para suporte *AIOps*
III. Dificuldade em criar modelos de Aprendizado de Máquinas para *AIOps*

O uso de Inteligência Artificial para operação de TI é visto como um campo interdisciplinar e de inovação, por isso é fundamental a colaboração entre a área acadêmica e as organizações empresariais (DANG et al., 2019).

Com a implementação do AIOps é almejado garantir alta qualidade de serviço e satisfação do cliente, aumentando a produtividade da engenharia e reduzindo o custo operacional.

Diante desse panorama, faz-se necessária a utilização de dados compartilhados, aplicando métodos estatísticos e de aprendizado de máquina para interpretar os padrões comportamentais destas informações coletadas e gerar visões em tempo real para as equipes de Operações de TI.

Este ambiente proporciona aos desenvolvedores uma maior responsabilidade no monitoramento dos sistemas e, em contrapartida, ampliou as responsabilidades da Operação de TI em manter a integridade e interação dos sistemas envolvidos, facilitando a formação de equipes interdisciplinares onde as equipes de desenvolvimento compartilham a visão da operação e vice-versa.

2.2.1 Aprendizado de Máquina para Classificação de Incidentes

Conforme definição de Monard e Baranauskas (2003), o aprendizado de máquina:

> [...] é uma área de Inteligência Artificial cujo objetivo é o desenvolvimento de técnicas computacionais sobre o aprendizado bem como a construção de sistemas capazes de adquirir conhecimento de forma automática. (MONARD; BARANAUSKAS, 2003, p.32)

A Operação de TI pode utilizar os algoritmos de aprendizado de máquina para automatização dos procedimentos manuais e antecipação de possíveis problemas, assim, contribuindo para a redução de custo e do tempo de reparo de um incidente.

Para aplicar este tipo de algoritmo existe uma necessidade de utilização de uma linguagem de programação adequada. Onde o R[1], disponível como *software* livre, é uma excelente alternativa de linguagem para programação desses algoritmos. Uma linguagem voltada a análise, manipulação e visualização dos dados que possui um grande conjunto de bibliotecas para aplicação de técnicas estatísticas e gráficas, conforme definição dos projetistas da linguagem Ihaka e Gentleman (1996).

A classificação e priorização de incidentes é um assunto explorado cientificamente e no estudo realizado por Zuev et al.(2018), no qual foram utilizados

[1] R é uma linguagem de programação criada pelo Ross Ihaka e Robert Gentleman

algoritmos de aprendizado de máquina para prever uma possível violação de SLA para auxiliar na redução no tempo de tratamento de um incidente.

Ao associar o aprendizado de máquina e automatização da classificação do incidente, é possível atingir uma precisão maior que a classificação manual, conforme demonstrado por Silva et al. (2018).

2.3 Considerações

Baseado na pesquisa realizada sobre Gerenciamento de Incidentes (GI) e Inteligência Artificial (IA) em Operações TI, a automatização no fluxo de Gerenciamento de Incidentes utilizando IA traz benefícios como a redução do tempo médio de reparo, redução de erros de classificação no atendimento do *Service Desk* e melhorias nos indicadores de disponibilidade. Logo, impacta positivamente na disponibilidade dos serviços de TI e nos custos operacionais reduzindo a carga de trabalho dos times envolvidos.

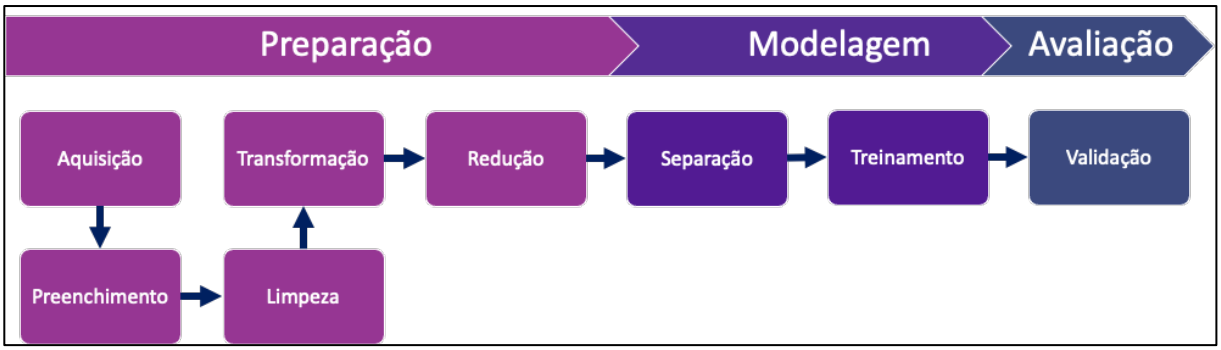
.

3 MÉTODO PARA CLASSIFICACAO DE CUMPRIMENTO DO SLA

Para contribuir com a redução do tempo de indisponibilidade dos negócios envolvidos, foi proposto um método para priorização do incidente de acordo com a possibilidade de cumprimento do SLA, criado baseado no histórico de atendimento da equipe de GI utilizando métodos estatísticos e aprendizado de máquina.

Baseado no processo CRISP-DM apresentado por Wirth e Hipp (2000), o método para classificação de incidente proposto neste estudo foi dividido em 3 fases que são a Preparação, a Modelagem e a Avaliação, ilustrado na figura 6. Estas fases serão desenvolvidas nas próximas seções do capítulo.

Figura 6 - Método para Classificação de Cumprimento de SLA



Fonte: Elaborado pelo Autor

3.1 Preparação

Nesta fase o objetivo é realizar a preparação dos dados do atendimento do incidente, partindo das informações registradas pela ferramenta de Gerenciamento de Serviços de TI utilizada pelas equipes de suporte.

3.1.1 Aquisição dos dados de incidente

Nesta primeira etapa de aquisição dos dados o objetivo é criar um conjunto de dados com os incidentes encerrados e seus respectivos dados de atendimento informados na ferramenta GSTI, a partir do histórico do *Service Desk*.

Partindo de um conjunto de dados formado pelo histórico das equipes de Suporte, foram elencados somente os atributos referentes às fases de Registro e Priorização do processo de GI, baseado em Agutter (2019). Foi agregado ao conjunto de dados o atributo que identifica o cumprimento do SLA, referente ao objetivo da classificação

- **Código do Incidente:** código identificador do incidente
- **Estado do Incidente:** descrição do estado do incidente.
- **Número de Reclassificação:** número de vezes que o incidente mudou o grupo.
- **Data Abertura:** data e hora de abertura do incidente.
- **Local:** local afetado.
- **Categoria:** categoria da falha.
- **Sintoma:** sintoma relatado pelo usuário sobre a disponibilidade do serviço.
- **Prioridade:** prioridade do incidente
- **Base de Conhecimento:** identificador de existência de base de conhecimento.
- **Mudança Relacionada:** identificador de mudança responsável pelo incidente.
- **Dentro do SLA:** identificador se o incidente cumpriu o SLA.

3.1.2 Preenchimento de valores ausentes

A presença de valores ausentes é muito comum no processo de aquisição de dados (GARCÍA et al., 2016), diante dessa constatação, na fase de preparação foi incluída no método a etapa de preenchimento de valores ausentes para os atributos “Sintoma”, “Prioridade”, “Número de Reclassificação”, “Base de Conhecimento” e “Mudança Relacionada”.

Vale ressaltar que ao preencher atributos utilizando técnicas de imputação no conjunto de dados, tem que ser considerada a possibilidade da quantidade de dados ajustados influenciar negativamente a estimativa dos resultados, tornando ineficiente a aplicação do método proposto.

3.1.2.1 Atributo “Sintoma”

Os registros que possuírem valores não previstos deverão ser substituídos pela literal “Desconhecido”. Este tratamento é fundamental para gerar o relacionamento com o atributo “Categoria” para criação do atributo de recorrência de incidente.

3.1.2.2 Atributo “Prioridade”

No preenchimento de valores inexistentes do atributo “Prioridade”, deverão ser selecionados os atributos com os mesmos valores de “Categoria”, “Sintoma” e “Localidade”. Logo após, os valores não preenchidos serão substituídos pelo valor mediano da seleção.

Caso não identifique um valor correspondente, o atributo “Prioridade” deverá ser mantido com o valor nulo.

3.1.2.3 Atributo “Número de Reclassificação”

No tratamento de atributo “Número de Reclassificação”, para os casos de conteúdos não preenchidos serão substituídos pelo valor mediano dos registros agrupados pelos campos “Categoria”, “Sintoma” e “Dentro do SLA”.

Na hipótese de não encontrar o valor, será preservado o valor nulo no atributo “Número de Reclassificação”.

3.1.2.4 Atributo “Base de Conhecimento”

Os registros que não possuírem valores válidos no atributo “Base de Conhecimento” serão preenchidos com o identificador referente ao valor booleano de *Falso*.

3.1.2.5 Atributo “Mudança Relacionada”

Uma vez que os valores do atributo “Mudança Relacionada” não estejam preenchidos, os registros terão estes substituídos pelo valor referente ao valor booleano de *Falso*.

3.1.3 Limpeza dos dados

Conforme García et al.(2016), a eliminação dos registros que não possuem dados é uma abordagem que raramente é benéfica, porém, o campo “Categoria” é necessário para formação de um novo atributo e o “Dentro do SLA” que é o atributo utilizado para validar o modelo.

Por essa razão, na etapa de limpeza de dados, serão removidos os registros que possuem os atributos “Categoria” e o indicador de “Dentro do SLA” com os valores ausentes.

3.1.4 Transformação dos dados

Posteriormente à limpeza dos dados, serão criados os campos de “Recorrência” e “Dia da Semana”. O campo “Recorrência” representa a quantidade de recorrências do incidente de acordo com a categoria e sua respectiva subcategoria. Foi criado baseado na que recorrência de problemas semelhantes o diagnóstico do sintoma, da categoria da falha e da solução tende a ser mais rápida.

Já o atributo “Dia da Semana” indica o dia da semana correspondente à data da abertura do chamado. Este atributo foi criado devido à relação entre o dia útil e o tempo médio de reparo, conforme Zuev et al.(2018).

Tabela 1 - Atributos do conjunto de dados pós transformação

Atributos	Tipo	Escala
Código do Incidente	Qualitativo	Nominal
Estado do Incidente	Qualitativo	Ordinal
Número de Reclassificação	Quantitativo	Intervalar
Data Abertura	Quantitativo	Intervalar
Local	Qualitativo	Nominal
Categoria	Qualitativo	Nominal
Sintoma	Qualitativo	Nominal
Prioridade	Qualitativo	Ordinal
Base de Conhecimento	Qualitativo	Nominal
Mudança Relacionada	Qualitativo	Nominal
Dentro do SLA	Qualitativo	Nominal
Recorrência	Quantitativo	Intervalar

Dia da Semana	Qualitativo	Ordinal

Fonte: Elaborado pelo autor

Deste modo, formando o novo conjunto de dados conforme a representação da tabela 1. Com os atributos definidos, foi aplicado o processo de discretização das variáveis, onde serão convertidos os valores dos campos contínuos em uma forma que possam utilizadas para cálculos numéricos.

3.1.5 Redução de dimensionalidade

Conforme Freitas(2003), o uso de atributos irrelevantes pode induzir resultados imprecisos e geram um desperdício de processamento e aumento do tempo de execução do modelo preditivo.

Nesta última etapa da Preparação o objetivo é reduzir a dimensionalidade do conjunto de dados aplicando métodos para retirada dos campos irrelevantes para atingir o objetivo do método. Deverão ser retirados os atributos de código do incidente, a classificação do estado do incidente e aplicar técnicas que buscam a remoção de variáveis altamente correlacionadas, caso existam, no conjunto de dados a ser estudado.

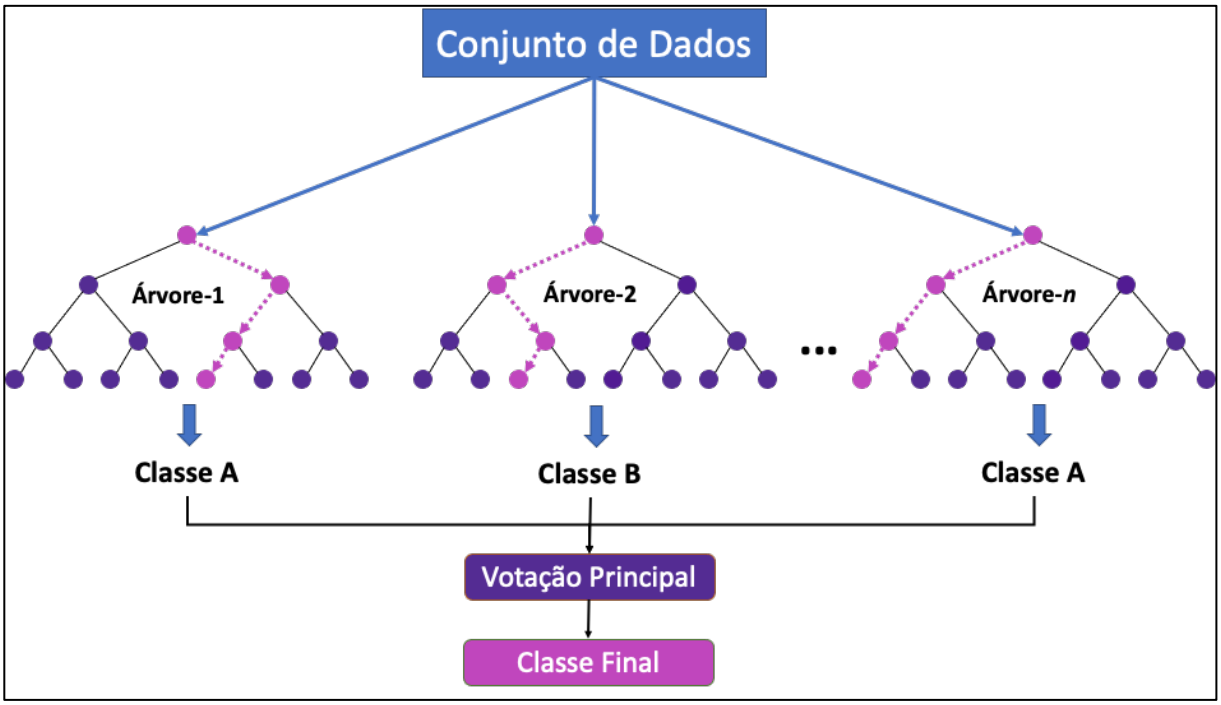
3.2 Modelagem

A Modelagem é a fase onde é selecionada a técnica que será aplicada nos dados preparados anteriormente e os seus parâmetros são calibrados para valores ideais. Para construção do algoritmo responsável pela modelagem será utilizado o *Random Forest* para classificar a possibilidade de cumprimento do SLA utilizando o conjunto de dados criado após a finalização da fase de preparação.

O classificador Floresta Aleatória (*Random Forest)* consiste em uma combinação de classificadores de árvores de decisão. Onde cada classificador é gerado utilizando um vetor aleatório a partir do conjunto de dados da entrada, assim, cada árvore lança um voto para a classe mais popular classificar este vetor utilizado na entrada. (BREIMAN, 2001)

Aplicando um método de aprendizagem em conjunto nas árvores de decisão para gerar conjuntos de treinamento diferentes e de baixa correlação para coleções de classificadores instáveis, chamado de *Bagging*, abreviação do inglês *Bootstrap aggregating*.

Figura 7 - Exemplo de funcionamento do *Random Forest*



Fonte: Breiman (2001)

Conforme a figura 7, classificador *Random Forest* é formado por N árvores, onde N representa o número de árvores de decisão a serem cultivadas, que é um parâmetro fornecido pelo usuário. Assim, para classificar um novo conjunto de dados, cada caso do conjunto de dados informado é passado para cada uma das N árvores. A floresta escolhe a classe final com o maior de número de votos, (BREIMAN, 2001).

3.2.1 Separação de dados

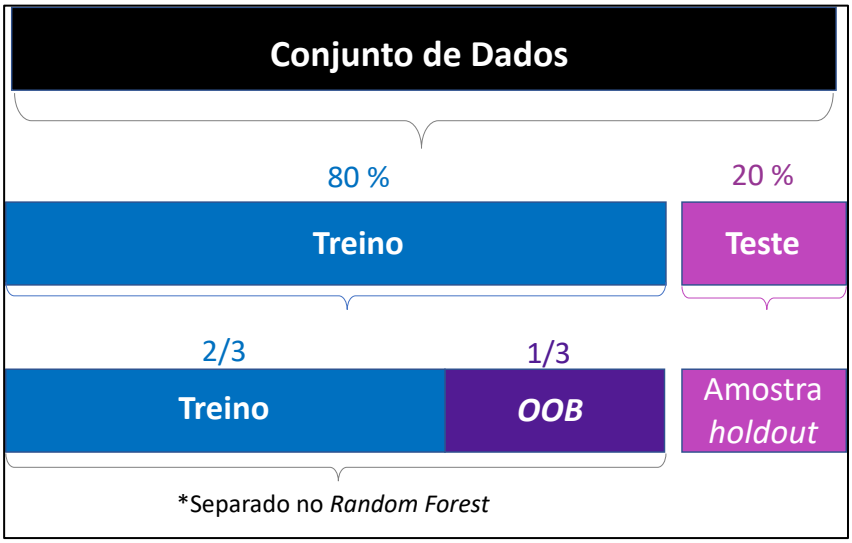
Com o conjunto de dados preparado o próximo passo é criar o classificador, porém, criar um classificador utilizando todos os dados levantados até pode ser útil, porém, é insuficiente para alcançar o resultado desejado.

Modelos sofisticados podem memorizar os dados de treinamento e causar o super-ajuste (do inglês: *overfitting*), assim, temos que separar os dados de

treinamento e teste em conjunto de dados similares. Neste estudo, serão divididos os dados em duas partes, treinamento em uma e teste na outra com o modelo treinado, técnica denominada de *hold-out* conforme YADAV e SHUKLA(2016).

Não foi incluída a criação de um conjunto para validação, pois, no algoritmo *Random Forest* aplicaremos o procedimento que faz uso de dois terços das amostras para treinamento. O terço restante, referido como na literatura como *out-of-bag* (OOB), será usado como conjunto de validação, conforme JAMES et al (2013), separação demonstrada na figura 8.

Figura 8 - Divisão de Treinamento e Teste



Fonte: Elaborado pelo autor

3.2.2 Treinamento

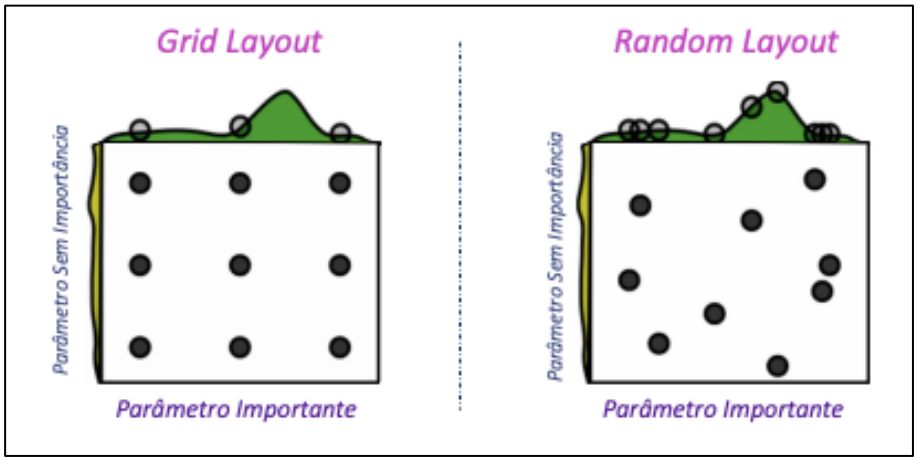
O Treinamento é a etapa do método onde algoritmo de aprendizado de máquina supervisionado gera o modelo de classificação, a partir dos dados separados na etapa anterior. Onde o termo modelo faz referência ao autômato gerado pela aplicação do algoritmo utilizando a base de treinamento.

Para um melhor resultado na aplicação do algoritmo *Random Forest* no conjunto de dados, recomenda-se o uso dos parâmetros otimizados, também chamados de hiperparâmetros de entrada.

Nesta etapa serão aplicadas técnicas para otimização dos hiperparâmetros utilizados na floresta aleatória, referente ao número de árvores (*ntree*) e ao número de atributos utilizados para construir cada árvore de decisão (*mtry*).

Na otimização do hiperparâmetro de número de árvores será aplicada as buscas randômicas, baseado no estudo de BERGSTRA e BENGIO (2012). Onde o estudo demonstrou uma eficiência maior do tipo de busca aleatória do que a busca manual ou a busca em grade (do inglês *Grid Search*) quando não é possível contemplar todos os valores da amostra.

Figura 9 - Comparação de métodos de Busca em Grade e Busca Aleatória



Fonte: Bergstra e Bengio(2012)

Conforme demonstrado na figura 9, onde BERGSTRA e BENGIO (2012) comparam o leiaute da busca em grade e a aleatória para otimização do hiperparâmetro, demonstrando uma possibilidade de aproximação maior na utilização da busca aleatória.

Para ajustar o número de atributos utilizados para construir cada árvore de decisão serão testados todos os valores possíveis devido ao baixo número (no máximo 11) de atributos do conjunto de dados.

Como o treinamento do modelo utilizando *Random Forest* é feito usando *out-of-bag* (*OOB*) para validação, torna a estimativa de erro do indicador importante para medir o erro médio de previsão do modelo, e permite que os hiperparâmetros sejam ajustados e validados de acordo com o método de otimização implementado.

3.3 Avaliação

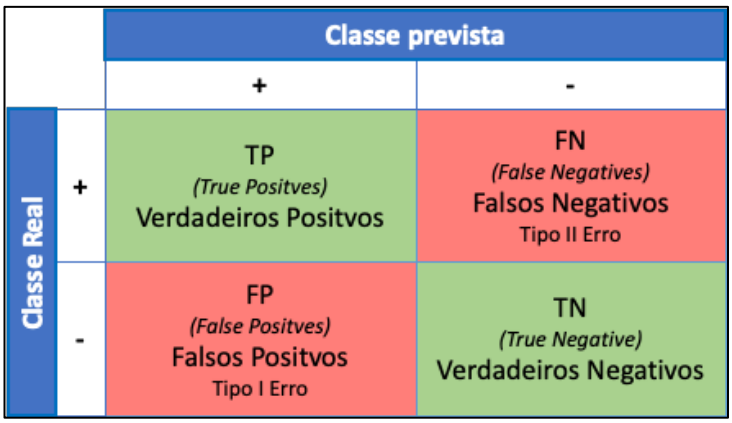
Na fase de Avaliação o objetivo é revisar o modelo construído de forma completa. Onde o resultado esperado após o término da fase, é uma decisão baseada na avaliação sobre o uso da apuração da mineração dos dados (WIRTH; HIPP, 2000).

3.3.1 Validação

Nesta etapa, serão avaliados se os resultados obtidos correspondem às expectativas do método, prevendo se um atendimento cumprirá ou não o SLA previsto somente com os dados iniciais do incidente.

Explorando os resultados obtidos a partir do conjunto de teste, conforme a matriz de confusão, exemplificada na figura 10.

Figura 10 - Estrutura da matriz de confusão



Fonte: Amidi e Amidi (2018)

Conforme Amidi e Amidi (2018), na tabela 2 estão descritas as principais métricas relacionadas à matriz como a acurácia, precisão, sensibilidade, especificidade e *F1 Score.*

Tabela 2 - Métricas de avaliação de desempenho

Métrica	***Fórmula***	***Interpretação***
Acurácia	$\frac{TP+TN}{TP+TN+FP+FN}$	Desempenho geral do modelo
Precisão	$\frac{TP}{TP+FP}$	Precisão das predições positivas
Sensibilidade	$\frac{TP}{TP+FN}$	Cobertura de amostra positiva real
Especificidade	$\frac{TN}{TN+FP}$	Cobertura de amostra negativa real
F1 Score	$\frac{2TP}{2TP+FP+FN}$	Métrica híbrida útil para classes desequilibradas

Fonte: Amidi e Amidi (2018)

É nesta fase que é definido se o modelo poderá ser implantado para prever o cumprimento do SLA, de acordo com os valores obtidos nas métricas apresentadas na avaliação e os objetivos do negócio

4 ESTUDO DE CASO

Para realizar o estudo de caso aplicando o método proposto, foi utilizada uma base de dados de atendimentos de chamados registrados no repositório de uma empresa de TI, com os dados anonimizados para fins de privacidade, publicado pela *University of California Irvine* por Amaral et al. (2018).

Já a análise exploratória inicial para entendimento e aplicação do modelo foram efetuadas utilizando a linguagem de programação R, uma linguagem voltada a análise, manipulação e visualização dos dados.

4.1 Análise do conjunto de dados

Antes da utilização do método proposto foi realizada a análise exploratória para entendimento do problema abordado e compreensão dos dados, buscando identificar padrões e valores extremos do conjunto selecionado para o estudo de caso.

Foram levantados 141 712 registros de eventos de atendimento referentes aos 24 985 incidentes registrados no período de 13 meses. Este conjunto de dados é composto por 1 identificador de caso, 1 identificador de estado, 32 atributos descritivos e 2 variáveis independentes, totalizando 36 atributos descritos no APÊNDICE A.

Na elaboração da análise foram selecionados somente os incidentes que possuíam evento de resolução (“*incidente_state*” = ”*Resolved*”), com o atributo categoria informado (“*category*” ≠ nulo), com a data e hora de resolução preenchida (“*resolved_at*” ≠ nulo) e indicador de SLA cumprido (“made_sla” ≠ nulo), reduzindo o público para 23 419 incidentes agrupados, onde foram apuradas as características principais do conjunto de dados e o seu relacionamento com o cumprimento do SLA.

Inicialmente foram calculadas as médias e medianas dos atributos de quantidade de reclassificações de incidentes, tempo de atendimento e recorrência da categoria do cancelamento do conjunto de dados, conforme a tabela 3.

Tabela 3 - Estatísticas básicas do conjunto de dados

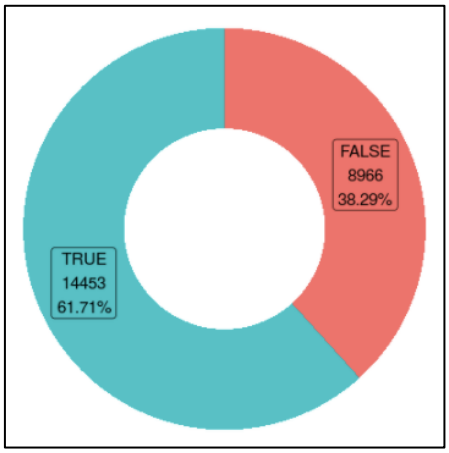
Atributo	***Média***	***Mediana***

Número de Reclassificação	1,004	0
Tempo de Atendimento	178h 43m	22h 15m
Recorrência	386,6	160

Fonte: Elaborado pelo autor

Seguindo, foi explorado o atributo “Dentro do SLA”, onde 61,7% dos incidentes foram atendidos conforme o acordado no SLA, representado no gráfico 1.

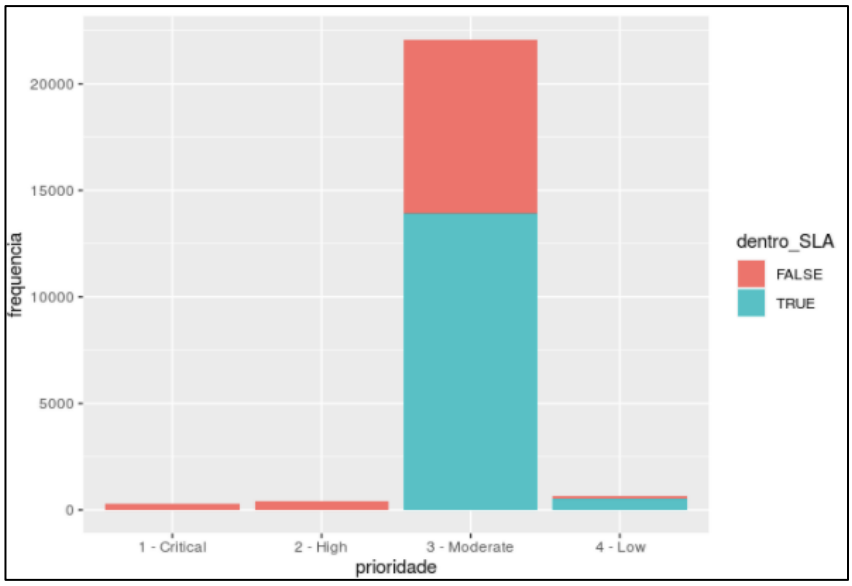
Gráfico 1 - Atendimentos que cumpriam o SLA previsto



Fonte: Elaborado pelo autor

Conforme o gráfico 2, a proporção de violação do SLA aumenta de acordo com a prioridade do incidente. Demonstrando uma ineficiência da área de *Service Desk* no atendimento, pois, conforme maior a criticidade do atendimento maior o número de violações do SLA.

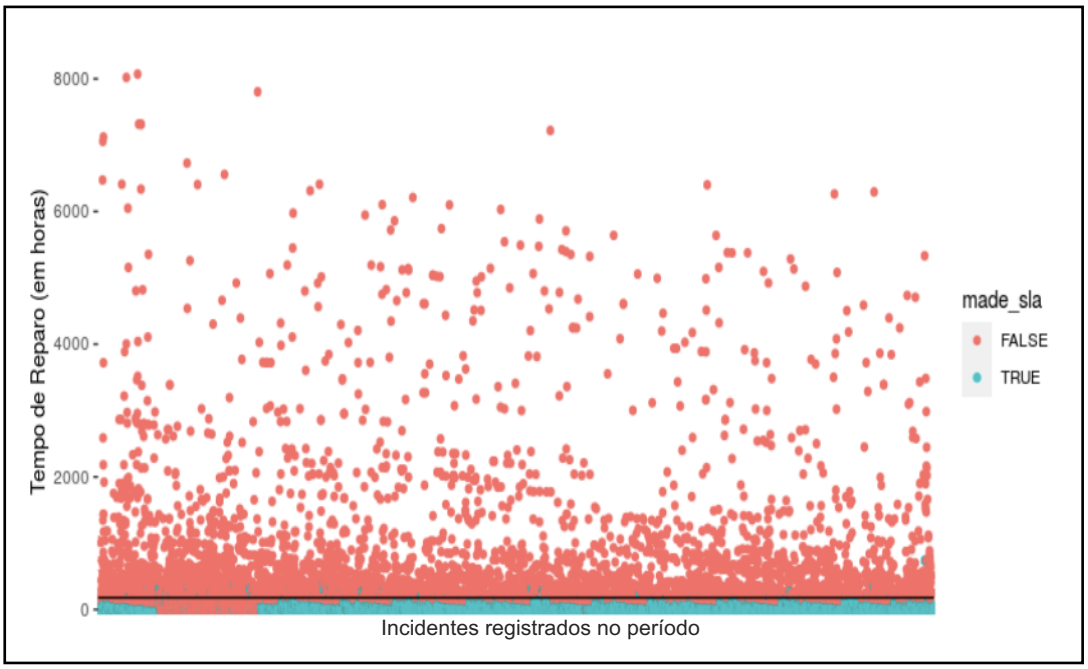
Gráfico 2 - Atendimentos que cumpriram o SLA agrupados por prioridade



Fonte: Elaborado pelo autor

No gráfico 3, onde cada ponto representa um incidente no período estudado e sua coloração representa se houve, ou não, o cumprimento de SLA. Os pontos vermelhos representam a violação do acordo e os verde o cumprimento do mesmo, demonstrando uma correlação forte com o tempo de reparo do incidente, pois, quanto maior o tempo maior o número de pontos vermelhos.

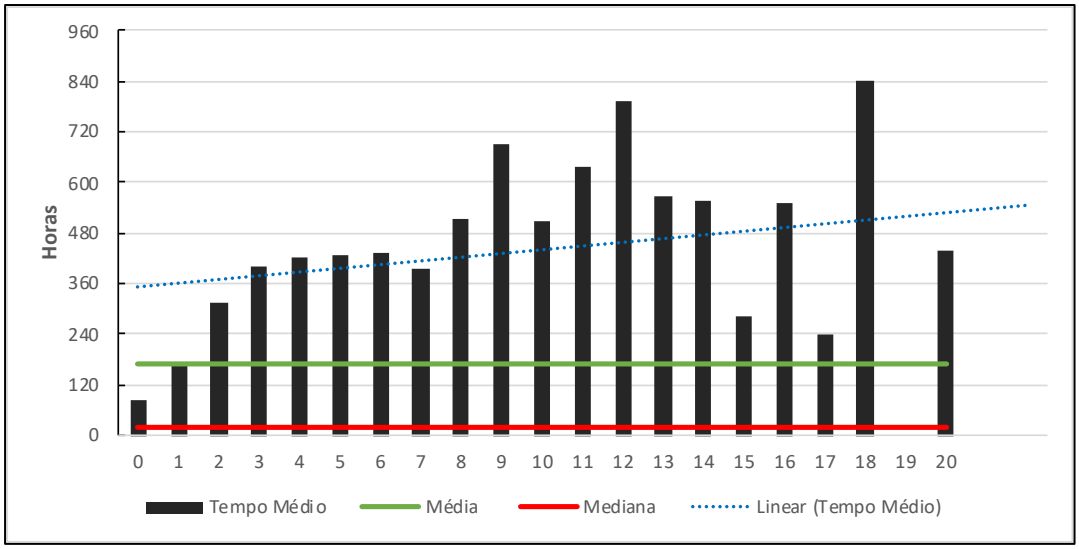
Gráfico 3 - Tempo de atendimento por Incidente



Fonte: Elaborado pelo autor

Com a constatação de que o tempo influencia no cumprimento do SLA, foi investigado outros fatores que influenciam no tempo de reparo. Foi colocada em evidência a variável de reclassificação do atendimento e buscamos a sua correlação com o tempo médio de reparo. Demonstrado no gráfico 4, conforme representado pela linha de tendência linear (pontilhada em azul) o crescimento do tempo diretamente relacionado com o aumento no número de reclassificações do incidente, representado no eixo horizontal.

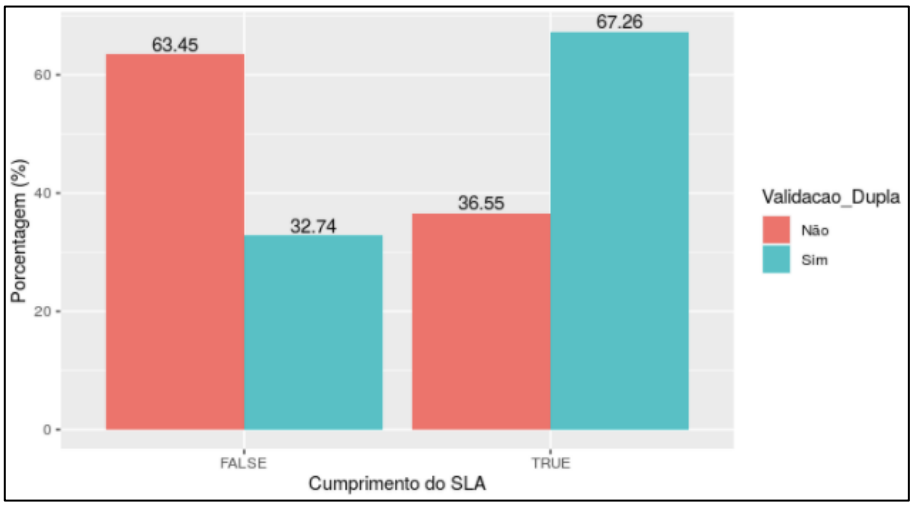
Gráfico 4 - MTTR por quantidade de reclassificação de incidente



Fonte: Elaborado pelo autor

Outro ponto analisado, foi o percentual de atendimentos que não tiveram dupla validação de prioridade, onde em 63,4% dos 4 227 incidentes nessa condição não cumpriram o SLA. Já os 19 192 atendimentos que foram duplamente validados só não cumpriram o SLA em 32,7% dos casos, conforme demonstrado no gráfico 5, assim, confirmando a importância da priorização correta para cumprimento do SLA.

Gráfico 5 - Comparativo cumprimento de SLA por validação de prioridade



Fonte: Elaborado pelo autor

Fundamentado na análise exploratória, foi evidenciado que os tempos de atendimento deste conjunto de dados afetam os SLAs e ocorrem com incidentes de todas as prioridades. Também, que a priorização efetuada corretamente tem um impacto no cumprimento do SLA do *Service Desk*.

Com o término da análise foi adotado o método para Classificação de Cumprimento do SLA no conjunto de dados explorado, onde serão descritos os resultados na seção seguinte deste capítulo.

4.2 Aplicação do Método

Nesta seção serão aplicadas as 3 fases do Método para Classificação de Cumprimento de SLA e validados os resultados obtidos. O algoritmo desenvolvido na linguagem R para essa aplicação do Método está representado no APÊNDICE B (etapa de Preparação, Modelagem e Avaliação).

4.2.1 Preparação

Como ação inicial no conjunto de dados, cumprindo a primeira etapa de Aquisição, formamos um novo conjunto de dados contendo os campos relacionados ao incidente como de categoria, sintoma, prioridade, indicador se existe base de conhecimento, localidade do incidente e o atributo que classifica o cumprimento do SLA. Foi incluído o campo de subcategoria presente no conjunto de dados e associado ao campo de categoria.

Na etapa de Transformação, com o novo conjunto de dados gerado foram criados 2 (dois) atributos, o primeiro que indica o número de vezes que o incidente recorreu no período informado no conjunto de dados, chamado de recorrência, e o segundo o atributo que corresponde ao dia da semana que ocorreu o incidente denominado “dia_semana”.

Após o preenchimento dos valores ausentes foi efetuada a limpeza dos registros que possuíam os atributos categoria e indicador de cumprimento do SLA inválidos, diminuindo a quantidade de linhas de 24 985 para 24 975.

Iniciando a etapa de Transformação dos dados, os campos "category", "subcategory", "u_symptom", "priority", "location" , "knowledge" e "made_sla" foram discretizados para podermos trabalhar com os valores numéricos na execução do algoritmo de aprendizado de máquina.

Com os atributos transformados em numéricos, utilizamos a função genérica da biblioteca do R para efetuar a centralização e o escalonamento das variáveis baseado nos valores estatísticos do conjunto dos dados.

Finalizando a fase de preparação foi reduzido o conjunto de dados, onde foram retiradas as colunas de status do incidente e seu respectivo código, para não influenciar incorretamente na etapa de modelagem.

4.2.2 Modelagem

Seguindo o método proposto no estudo, na fase de modelagem foi utilizado o *Random Forest* – ou Florestas Aleatórias (tradução nossa) – da biblioteca *randomForest* do R.

A partir dos dados obtidos da fase de preparação, foi iniciada a etapa de Separação dos Dados onde 80% (19 980) dos registros do conjunto de dados preparado foram separados para treinamento do modelo e o restante (4 995) disponibilizado para teste do modelo na etapa de avaliação.

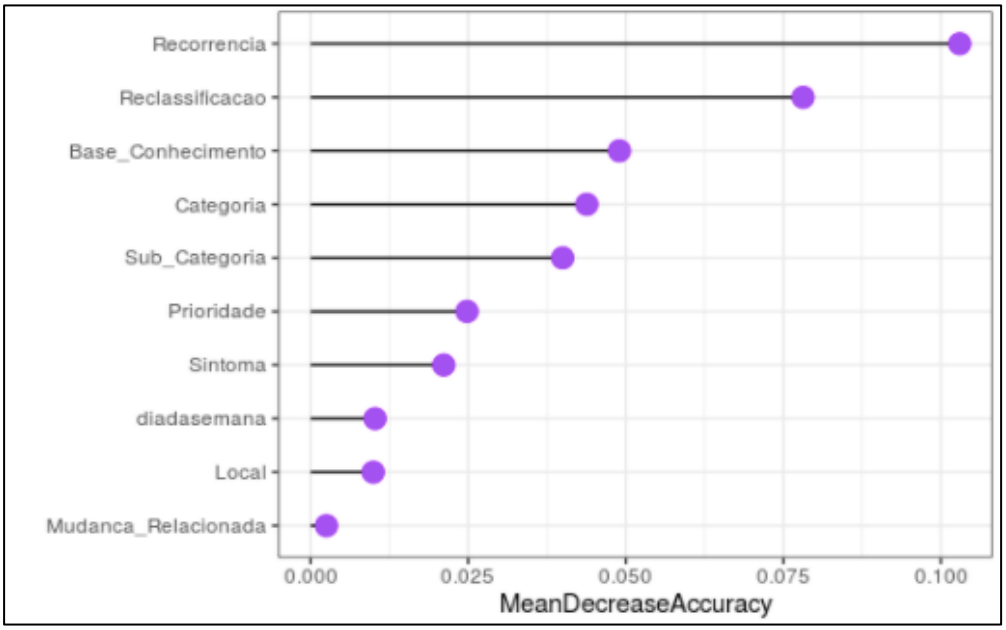
Após esta separação foi validada a similaridade entre os conjuntos de dados de treino e teste. Aplicando a similaridade por cosseno, medida pelo valor do cosseno do ângulo compreendido entre os vetores comparados, entre os vetores formados pela média e pelos valores medianos dos atributos. Confirmando a distribuição similar dos 2 (dois) conjuntos de dados.

Como procedimento inicial do Treinamento, foi utilizado o algoritmo de Floresta Aleatória utilizando um número elevado de árvores (1 000) e o número de atributos padrão para cada árvore, representado pelo valor inteiro da raiz quadrada da quantidade de atributos do conjunto, para iniciar os ajustes dos hiperparâmetros do algoritmo.

Inicialmente foram extraídas as informações referentes à importância dos atributos para a construção do modelo. No gráfico 6, aponta a variável “Recorrencia” como a mais importante do modelo gerado, demonstrando que quando a variável é

retirada da árvore maior é a redução média da precisão da floresta, coeficiente representado pelo eixo *MeanDecreaseAccuracy*.

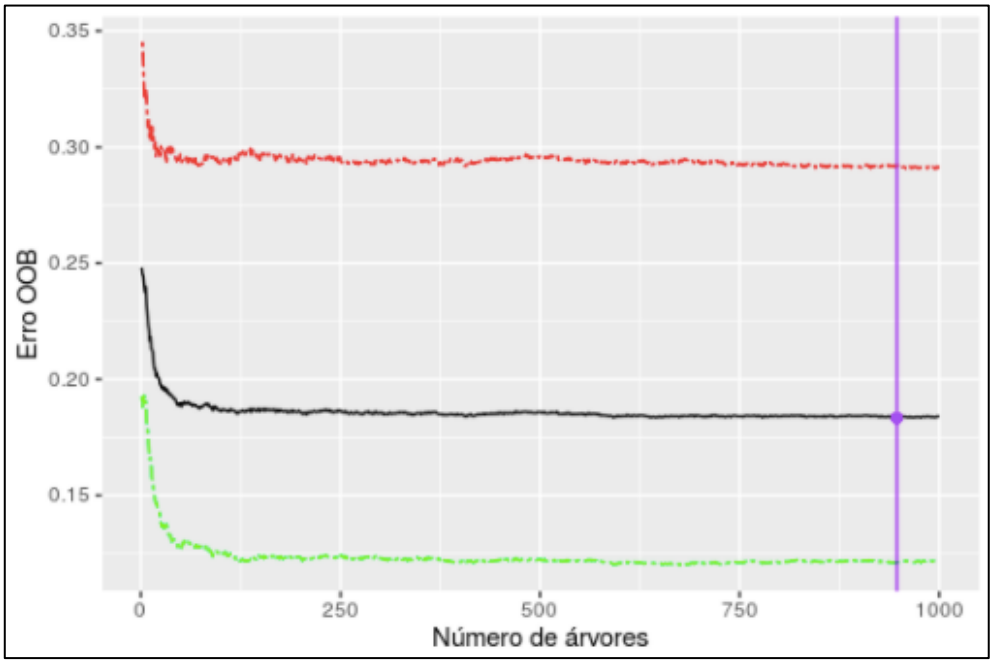
Gráfico 6 - Importância dos atributos no modelo



Fonte: Elaborado pelo autor

O Erro OOB gerado pela execução do algoritmo, apontou a melhor acurácia para a quantidade de 947 árvores (ntree) atingindo um percentual de erro de 18,3%, representado no gráfico 7 pela reta roxa.

Gráfico 7 - Taxa de erro do OOB por número de árvores utilizadas no modelo

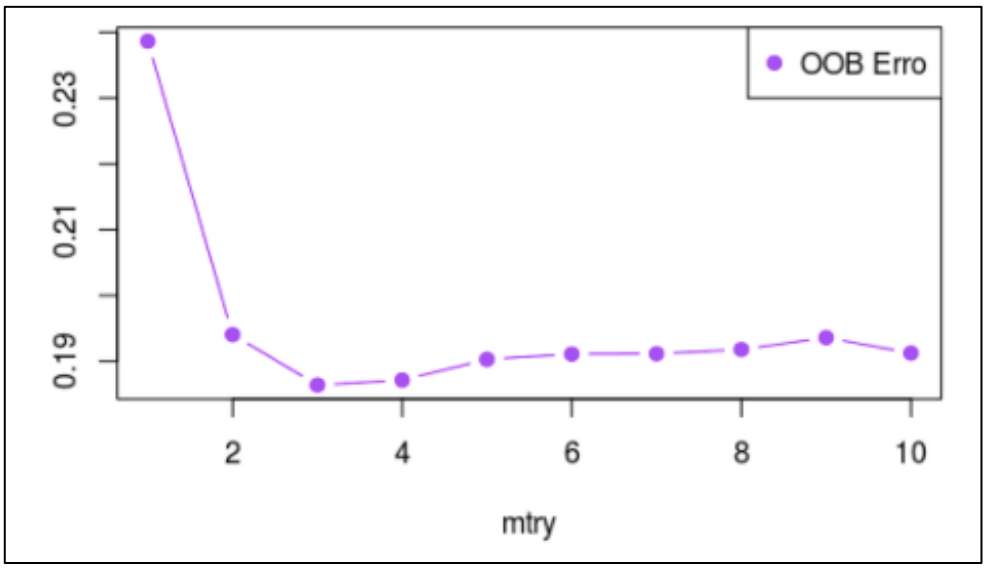


Fonte: Elaborado pelo autor

Neste gráfico também estão representadas a taxa de erro de classificação das variáveis positivas (verde) e as negativas em (vermelho), contribuindo para a compreensão que o modelo começa a estabilizar próximo ao número de 125 árvores utilizadas na floresta.

Após o entendimento de algumas características do modelo, foi iniciada a busca pelo melhor valor do *mtry*, efetuando o teste de todos os possíveis parâmetros para o conjunto de 10 (dez) atributos. Conforme representação no gráfico 8, o *mtry* igual a 3 (três) foi diagnosticado como o valor de melhor desempenho utilizando o Erro do OOB como métrica de avaliação.

Gráfico 8 - Taxa de erro por OOB por número de variáveis utilizadas no modelo

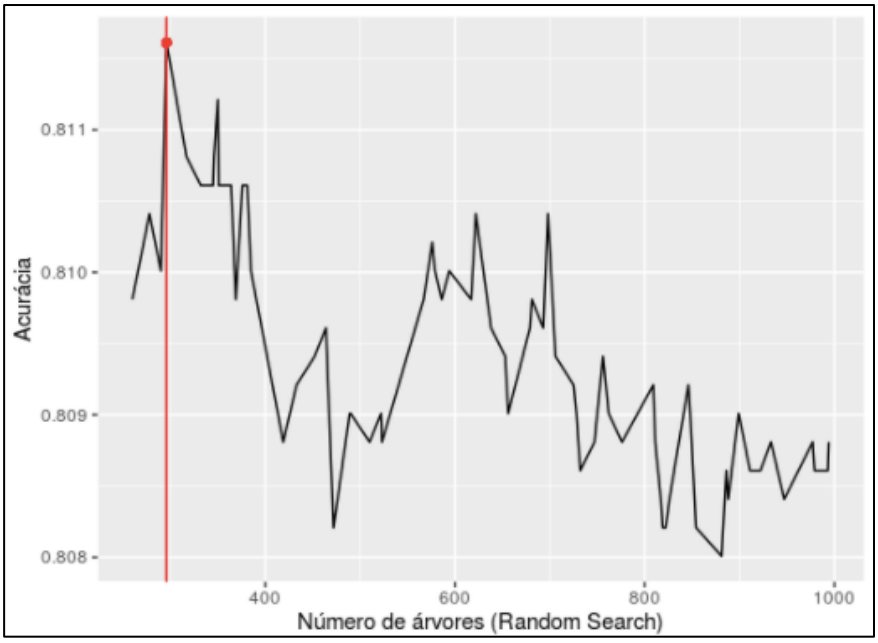


Fonte: Elaborado pelo autor

Já para otimizar de número de árvores da Floresta Aleatória a partir do conjunto de dados utilizados para treinamento, foi aplicada a busca aleatória (*random_search no R*).

Aplicado o algoritmo de busca em 10% da amostra das árvores que obtiveram uma taxa de erro OOB inferior ao 3º quartil da amostra (18,58%), totalizando 74 interações de busca aleatória.

Gráfico 9 - Acurácia na busca aleatória utilizando o conjunto de teste



Fonte: Elaborado pelo autor

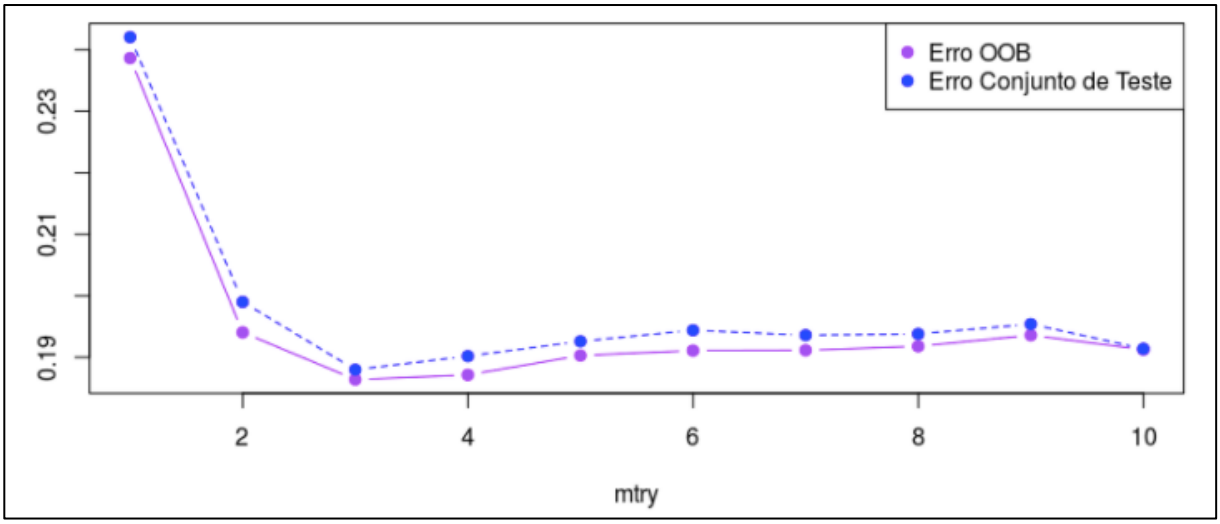
Utilizando o conjunto de teste na busca aleatória, o melhor parâmetro apurado foi de 296 árvores atingindo a melhor taxa de desempenho geral (acurácia de 81,16%) dentre as quantidades apresentadas pela função de busca aleatória, conforme a representação no gráfico 9.

4.2.3 Avaliação

Conforme a fase de modelagem, ajustamos os hiperparâmetros o *mtry* para 3 atributos utilizados para construir cada árvore de decisão e *ntree* foi ajustado para 296 árvores utilizadas no modelo utilizando o *Random Forest*.

A partir dos resultados obtidos pelo modelo foi avaliada se a utilização do parâmetro *mtry*, provando no gráfico 10 que a otimização utilizando a base de treinamento está de acordo com a validação empregando o conjunto de teste, onde o mtry igual a 3 é o melhor valor, também, para o conjunto de teste.

Gráfico 10 - Erros OOB *versus* erros obtidos no teste



Fonte: Elaborado pelo autor

Ao aplicar o modelo com os parâmetros propostos, foi atingida a acurácia de 81,16%, aproximando-se do erro calculado pela validação via Erro *Out-Of-Bag* no treinamento, onde atingiu 18,44% (ou 81,56% de acurácia).

Como a acurácia representa o desempenho geral, montamos a matriz de confusão (tabela 4) para aprofundar a análise preditiva do método utilizando como classe positiva o conteúdo do atributo “Dentro do SLA” igual a “1”.

Tabela 4 - Matriz Confusão do resultado do modelo gerado

		Prevista	
		1	**0**
Real	**1**	**2 754**	411
	0	530	**1 300**

Fonte: Elaborado pelo autor

Destacados na tabela 4, estão os verdadeiros positivos e verdadeiros negativos obtidos usando o conjunto de teste foram geradas as métricas apresentadas na tabela 5. Conforme tabela, o modelo apresentou um número significante de acertos na predição positiva onde a métrica de precisão atingiu 83,8% e a de sensibilidade 87%.

Tabela 5 - Métricas de avaliação de desempenho do modelo gerado

Métrica	Valor
Acurácia	81,16%
Precisão	83,86%
Sensibilidade	87,01%
Especificidade	71,04%
F1 Score	85,41%

Fonte: Elaborado pelo autor

Já a Especificidade, demonstra a taxa que a classe negativa (“0”) é avaliada como negativa corretamente em 71% dos casos encontrados no conjunto de teste. Isto significa que o modelo indica que um incidente não cumprirá o SLA acerta em 71% das vezes no conjunto de testes utilizados.

Outra métrica importante avaliada no conjunto de dados foi o *F1 Score* onde apresenta uma taxa de 85,4% ao utilizar a classe “1” como positiva demonstrando um ótimo indicador para aplicação do modelo, o melhor dentre todos os modelos testados via busca aleatória.

No término da fase de Avaliação, foi concluído que o Método para Classificação de Cumprimento de SLA pode ser utilizado para predizer se o SLA será cumprido. Isso porque, as métricas apresentam valores aceitáveis perante a análise exploratória realizada.

5 CONCLUSÃO

Seguindo a proposta de reduzir o tempo de indisponibilidade e o custo dos serviços de TI foram pesquisados os temas de Gerenciamento de Incidentes, Inteligência Artificial na Operação de TI e sua aplicação na classificação dos incidentes.

De acordo com a pesquisa, foram apurados que os problemas de classificação dos incidentes afetam negativamente o tempo de disponibilidade dos serviços de TI e geram desperdícios de recursos financeiros nas organizações. Foi demonstrado que aplicar algoritmos de inteligência artificial é uma solução efetiva para alcançar resultados satisfatórios na redução do tempo médio de reparo, refletindo positivamente no custo e nos indicadores de disponibilidade.

Para atingir o objetivo do estudo foi desenvolvido o Método para Classificação de Cumprimento de SLA, um importante indicador de disponibilidade, através do uso de aprendizado de máquina para acelerar a priorização e, consequentemente, a resolução do incidente.

O método foi dividido em Preparação, Modelagem e Avaliação e no final das 3 fases espera-se um modelo que poderá ser aplicado nos dados iniciais do atendimento para predição do cumprimento do SLA, auxiliando o suporte N1 na etapa de priorização do incidente.

A partir de um conjunto de dados real foi aplicado o método proposto, quando foram obtidos resultados satisfatórios na predição de uma possível violação de SLA, atingindo uma acurácia de 81,16% do modelo gerado.

Este resultado possibilita a substituição da priorização manual do incidente efetuada pelos analistas do Suporte N1 pela priorização automatizada, onde o modelo gerado utiliza os dados das ferramentas de gerenciamento de serviços. Desta maneira, reduz o número de horas gastas do *Service Desk* com a atividade e, consequentemente, o tempo médio de reparo substituindo um processo manual por um automatizado, demonstrando que o método auxilia na redução do custo e aumenta a disponibilidade dos serviços de TI.

5.1 Trabalhos Futuros

Como trabalho futuro, será implantado o método criado e automatizado o processo de Priorização em uma empresa real para medir as reduções de custo e o aumento da disponibilidade dos serviços de TI.

Focando na melhoria do método proposto, iniciar o estudo de inserção de uma etapa para análise de registro de erro (*log*) do incidente e associar a categorização do incidente e a utilização da classificação do grupo responsável pela solução aplicando algoritmos de classificação utilizando aprendizado de máquina. Dessa maneira, enriquece o conjunto de dados para diminuir as taxas de falso positivos e negativos do modelo.

A melhoria proposta tem o objetivo de caracterizar o que não deverá ser resolvido pelo primeiro nível diminuindo as horas gastas com este tipo de trabalho e melhorando a qualidade do direcionamento evitando uma atribuição incorreta, contribuindo para redução do tempo médio de reparo, consequentemente, reduzindo custos e aumentando a disponibilidade dos serviços de TI.

REFERÊNCIAS BIBLIOGRÁFICA

AGUTTER, C. **ITIL Foundation Essentials ITIL 4 Edition - The ultimate revision guide**, second edition. 2019.

AMARAL, C. A. L. et al. Enhancing Completion Time Prediction Through Attribute Selection. **Proceedings of the 15th International Conference on Advanced Information Technologies for Management (AITM 2018),** Revised Selected Papers Lecture Notes in Business Information Processing, v. 346, pp. 3-23, 2019.

AMIDI, A.; AMIDI, S. **Stanford – CS 229 – Aprendizado de Máquina.** Dicas e truques de aprendizado de máquina. Acessado em: 2018-10-13. Disponível em: <https://stanford.edu/~shervine/l/pt/teaching/cs-229/dicas-truques-aprendizado-maquina>. Acesso em 18 nov. 2020

BERGSTRA, J., BENGIO, Y.: Random search for hyper-parameter optimization. **Journal of Machine Learning Research 13,** 281–305, 2012.

BMC. **Service Management Blog.** Incident Management in ITIL 4. 14 mai. 2019. Disponível em: <https://www.bmc.com/blogs/itil-incident-management/>. Acesso em 22 nov. 2020.

BMC. **Service Management Blog.** ITIL 4 Management Practices. 08 mai. 2019. Disponível em: <https://www.bmc.com/blogs/itil-management-practices/>. Acesso em 22 nov. 2020.

BREIMAN, L. **Random forests. Machine Learning**, 45(1):5–32, 2001.

DANG, Y. et al. AIOps: Real-world challenges and research innovations. **PROCEEDINGS - 2019 IEEE/ACM 41st INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING: COMPANION**, 2019. ICSE - Companion 2019, 2019. 4–5.

FREITAS, A. A. A survey of evolutionary algorithms for data mining and knowledge discovery. **Advances in evolutionary computing.** [S.l.]: Springer, 2003. p. 819–845.

FORRESTER CONSULTING. **IT Speed: The Crisis and The Savior Of The Enterprise,** Cambridge, USA, 2013. 22p.

GARCÍA et al. Big data preprocessing: methods and prospects, **Big Data Analytics**, Granada, 2016, 1-9.

GUPTA, R. et al. Automating ITSM incident management process. **5TH INTERNATIONAL CONFERENCE ON AUTONOMIC COMPUTING**, 2008, Chicago, USA. ICAC 2008, Chicago, 2008. 141-150.

HILBERT, M.; LÓPEZ, P. The world's technological capacity to store, communicate, and compute information. **Science**, 2011, 332(6025): 60-65.

IHAKA, R.; GENTLEMAN, R. R: A Language for Data Analysis and Graphics. **Journal of Computational and Graphical Statistics**, Vol. 5, USA, 1996, 299-314.

ISO/IEC 20000-1:2005 – Information Technology - Service Management - Part 1: Specification, ISO/IEC, Dec. 2005.

JAMES, G. et al. **An Introduction to Statistical Learning – with Applications in R, volume 103 of Springer Texts in Statistics**. Springer, New York, 2013.

KOTTER, J.P. **Acelere: Tenha agilidade estratégica num mundo em constante transformação**. Ed.1. São Paulo: HSM, 2015. 208p.

LERNER, A. AIOps Plataforms. 9 ago. 2017. Disponível em: <https://blogs.gartner.com/andrew-lerner/2017/08/09/aiops-platforms/>. Acesso em 28 abr. 2020.

MONARD, M.C.; BARANAUSKAS, J.A., 2003. **Conceitos sobre aprendizado de máquina. Sistemas inteligentes - Fundamentos e aplicações**, 1(1), p.32.

MURPHY, N. R. et al. **Site Reliability Engineering.** Ed. Kindle. Sebastopol: O'Reilley Media, 2016.

SERVICENOW. **ServiceNow Product Documentation.** Apresenta documentação dos serviços fornecidos pela ServiceNow. Disponível em: <https://docs.servicenow.com> . Acesso em: 29 out. 2020.

SILVA, S.; PEREIRA, R.; RIBEIRO, R. Machine learning in incident categorization automation, **Iberian Conference on Information Systems and Technologies**, CISTI 2018, Caceres, 2018, 1-6.

SNOW, A. P.; WECKMAN, G. R.; GUPTA, V.; Meeting SLA Availability Guarantees through Engineering Margin, **2010 Ninth International Conference on Networks**, Menuires, 2010, pp. 331-336, doi: 10.1109/ICN.2010.59.

WEILL, P.; WOERNER, S.L. **Qual o seu Modelo Digital de Negócio ?.** Ed.1. São Paulo: M.Books, 2019. 254p.

WIRTH, R.; HIPP, J. CRISP-DM: Towards a standard process model for data mining. **Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining**, pp. 29-39. London, UK: Springer-Verlag, 2000.

YADAV, S.; SHUKLA, S. "Analysis of k-Fold Cross-Validation over Hold-Out Validation on Colossal Datasets for Quality Classification," **2016 IEEE 6th International**

Conference on Advanced Computing (IACC), Bhimavaram, 2016, pp. 78-83, doi: 10.1109/IACC.2016.25.

ZUEV, D.; KALISTRATOV, A.; ZUEV, A. Machine Learning in IT Service Management. **Procedia Computer Science 2018,** 2018, Prague, Czech Republic. BICA 2018, Prague, 2018. 675-679.

APÊNDICE A - DESCRIÇÃO DO CONJUNTO DE DADOS ESTUDADO

Segue o Quadro 1 que descreve os atributos do conjunto de dados obtidos para o estudo.

Tabela 6 - Descrição dos Campos do Conjunto de Dados

Atributo	Descrição
number	Identificador de Incidentes
incidente_state	Estados que controlam as transições do processo de GI, da abertura até o fechamento do caso
active	Booleano que mostra se o registro está ativo ou fechado
reassignment_count	Nº de vezes que o incidente mudou o grupo ou os analistas de suporte
reopen_count	Nº de vezes que a resolução do incidente foi rejeitada pelo solicitante
sys_mod_count	Nº de atualizações de incidentes até aquele momento
made_sla	Booleano que mostra se o incidente excedeu o SLA de destino
caller_id	Identificador do usuário afetado
opened_by	Identificador do usuário que relatou o incidente
opened_at	Data e hora de abertura do usuário incidente
sys_created_by	Identificador do usuário que registrou o incidente
sys_created_at	Data e hora da criação do sistema de incidentes
sys_updated_by	Identificador do usuário que atualizou o incidente e gerou o registro de log atual
sys_updated_at	Data e hora da atualização do sistema de incidentes
contact_type	Variável categórica que mostra por que meios o incidente foi relatado
location	Identificador da localização do local afetado
category	Descrição de primeiro nível do serviço afetado
subcategory	Descrição de segundo nível do serviço afetado (relacionada à descrição de primeiro nível, isto é, à categoria)
u_symptom	Descrição da percepção do usuário sobre a disponibilidade do serviço
cmdb_ci	Item de configuração usado para relatar o item afetado
impact	Descrição do impacto causado pelo incidente

urgency	Descrição da urgência informada pelo usuário para a resolução do incidente
priority	Valor calculado com base no produto de 'impacto' e 'urgência'
assignment_group	Identificador do grupo de suporte responsável pelo incidente
assigned_to	Identificador do usuário responsável pelo incidente
knowledge	Booleano que mostra se um documento da base de conhecimento foi usado para resolver o incidente
u_priority_confirmation	Booleano que mostra se o campo de prioridade foi verificado duas vezes
notify	Variável categórica que mostra se as notificações foram geradas para o incidente
problem_id	Identificador do problema associado ao incidente
rfc	Identificador da solicitação de mudança associada ao incidente
vendor	Identificador do fornecedor responsável pelo incidente
caused_by	Identificador da RFC responsável pelo incidente
closed_code	Identificador da resolução do incidente
resolved_by	Identificador do usuário que resolveu o incidente
resolved_at	Data e hora da resolução do usuário incidente
closed_at	Data e hora de fechamento do usuário incidente

Fonte: *ServiceNow*™ (2020)

APÊNDICE B - ALGORITMO DE PREPARAÇÃO, MODELAGEM E AVALIAÇÃO

```
dfIncidentes  = read.csv("atendimentos.csv",
                         header=TRUE,
                         sep=",",
                         dec = ".")

intlinhas = nrow(dfIncidentes)
msgDisplay = paste("Linhas dfIncidentes:",
as.character(intlinhas))
print(msgDisplay)

#Selecionar os incidentes com Status fechado e agrupado por
número de incidente
dfAtendimentos <- dfIncidentes[dfIncidentes$incident_state ==
"Closed",]

intlinhas = nrow(dfAtendimentos)
msgDisplay = paste("Linhas dfAtendimentos antes do Pré-
Processamento:", as.character(intlinhas))
print(msgDisplay)

# 1 - Pré-Processamento

# 1.1 - Aquisição de dados do Incidente
dfAquisição <- data.frame(
                         dfAtendimentos$number,
                         dfAtendimentos$incident_state,
                         dfAtendimentos$reassignment_count,
                         dfAtendimentos$opened_at,
                         dfAtendimentos$location,
                         dfAtendimentos$category,
                         dfAtendimentos$subcategory,
                         dfAtendimentos$u_symptom,
                         dfAtendimentos$priority,
                         dfAtendimentos$knowledge,
                         dfAtendimentos$rfc,
                         dfAtendimentos$made_sla
                 )

lstCampos_dfAquisicao <- list(
                             "Cod_Incidente",
                             "Estado_Incidente",
                             "Reclassificacao",
                             "Data_Abertura",
                             "Local",
                             "Categoria",
                             "Sub_Categoria",
                             "Sintoma",
```

```
                                "Prioridade",
                                "Base_Conhecimento",
                                "Mudanca_Relacionada",
                                "Dentro_SLA"
                            )

lenLista = length(lstCampos_dfAquisicao)

i = 0
for (campo in lstCampos_dfAquisicao){
  i = i + 1
  names(dfAquisição)[i] = campo
}

# 1.2 -  Preenchimento de Valores Ausentes

# Atributo Sintoma
dfAquisição$Sintoma[dfAquisição$Sintoma == "?"] =
"Desconhecido"

# Atributo Prioridade
dfVlrAusentes_Prioridade <- dfAquisição[dfAquisição$Prioridade
== '?',]

# Atributo Reclassificação
dfVlrAusentes_Reclassificacao <-
dfAquisição[dfAquisição$Reclassificacao == '?',]

# Atributo Base de Conhecimento
dfAquisição$Base_Conhecimento[dfAquisição$Base_Conhecimento ==
"?"] = "false"

# Atributo Mudança Relacionada
dfAquisição$Mudanca_Relacionada[dfAquisição$Mudanca_Relacionad
a != "?"] = 1
dfAquisição$Mudanca_Relacionada[dfAquisição$Mudanca_Relacionad
a == "?"] = 0


# 1.3 Limpeza dos Dados
dfAquisição <- dfAquisição[dfAquisição$Categoria != '?',]
dfAquisição <- dfAquisição[dfAquisição$Sub_Categoria != '?',]
dfAquisição <- dfAquisição[dfAquisição$Dentro_SLA != '?',]

# 1.4 Transformação dos Dados

# Binarização e Discretização
dfAquisição$Data_Abertura <-
strptime(dfAquisição$Data_Abertura, "%d/%m/%Y %H:%M")
dfAquisição$Base_Conhecimento <-
as.logical(dfAquisição$Base_Conhecimento)
```

```
dfAquisição$Dentro_SLA <- as.logical(dfAquisição$Dentro_SLA)
dfAquisição$Categoria <- as.factor(dfAquisição$Categoria)
dfAquisição$Sub_Categoria <-
as.factor(dfAquisição$Sub_Categoria)
dfAquisição$Sintoma <- as.factor(dfAquisição$Sintoma)
dfAquisição$Local <- as.factor(dfAquisição$Local)
dfAquisição$Prioridade <- as.factor(dfAquisição$Prioridade)

# Criando Campos (Data Integration)
dfSeqErros <- data.frame(dfAtendimentos$category,
                         dfAtendimentos$subcategory,
                         0)

names(dfSeqErros)[1] <- "Categoria"
names(dfSeqErros)[2] <- "Sub_Categoria"
names(dfSeqErros)[3] <- "Recorrencia"

dfSeqErros$Categoria = as.character(dfSeqErros$Categoria)
dfSeqErros$Sub_Categoria =
as.character(dfSeqErros$Sub_Categoria)

dfSeqErros <- dfSeqErros[with(dfSeqErros, order(Categoria,
Sub_Categoria)), ]

dfSeqErros <- dfSeqErros[!duplicated(dfSeqErros), ]

row.names(dfSeqErros) <- NULL

for ( i in 1:nrow(dfAquisição)) {
  categoria = as.character(dfAquisição[i, "Categoria"])
  subcategoria = as.character(dfAquisição[i, "Sub_Categoria"])
  indices <- which(dfSeqErros$Categoria == categoria &
dfSeqErros$Sub_Categoria == subcategoria)
  dfAquisição[i, "Recorrencia"] = dfSeqErros[indices,
"Recorrencia"]
  dfAquisição[i, "causaconcat"] = paste(categoria,
subcategoria)
  dfSeqErros[indices, "Recorrencia"] = dfAquisição[i,
"Recorrencia"] + 1
}

dfAquisição$causaconcat <- as.factor(dfAquisição$causaconcat)
dfAquisição$diadasemana <-
as.factor(weekdays(dfAquisição$Data_Abertura))

# Conversão para númerico
dfAquisição_Numeric = dfAquisição
tempDentro_SLA <- as.numeric(dfAquisição_Numeric$Dentro_SLA)
dfAquisição_Numeric$Dentro_SLA <- NULL

dfAquisição_Numeric$Cod_Incidente <- NULL
```

```
dfAquisição_Numeric$Estado_Incidente <- NULL

for (i in 1:12 ){
  dfAquisição_Numeric[,i] <-
as.numeric(dfAquisição_Numeric[,i])
}

dfAquisição_Numeric$Cod_Incidente <- NULL
dfAquisição_Numeric$Estado_Incidente <- NULL

# Escalonamento de varíaveis
dfAquisição_Numeric_Scale =
as.data.frame(scale(dfAquisição_Numeric))

# Passando o atributo classificador para última coluna
dfAquisição_Numeric_Scale$Dentro_SLA <- tempDentro_SLA

library(caTools)
library(caret)
library(randomForest)
library(dplyr)
library(stylo)
library(ggplot2)
library(paramtest)

rf_RdmSch <- function(iter, N, base_testeNum) {
  set.seed(10)
  ModeloRandomRF = randomForest(
    formula = Dentro_SLA ~ .,
    data=base_treinamentoNum,
    ntree=N,
    mtry = 3,
    importance=FALSE
  )

  predBuscaRF<-predict(ModeloRandomRF,newdata =
base_testeNum[-11])
  tblMatriz_confusao = table(base_testeNum[, 11], predBuscaRF)
  cnfMatrizRF <- confusionMatrix(tblMatriz_confusao,
positive='1')

  return(t(c ( cnfMatrizRF$positive,
               cnfMatrizRF[["table"]][1,1],
               cnfMatrizRF[["table"]][1,2],
               cnfMatrizRF[["table"]][2,1],
               cnfMatrizRF[["table"]][2,2],
               cnfMatrizRF$overall[1],
               cnfMatrizRF$overall[2],
               cnfMatrizRF$overall[3],
               cnfMatrizRF$overall[4],
               cnfMatrizRF$byClass[1],
```

```
                cnfMatrizRF$byClass[2],
                cnfMatrizRF$byClass[3],
                cnfMatrizRF$byClass[4],
                cnfMatrizRF$byClass[5],
                cnfMatrizRF$byClass[6],
                cnfMatrizRF$byClass[7],
                cnfMatrizRF$byClass[8],
                cnfMatrizRF$byClass[9],
                cnfMatrizRF$byClass[10],
                cnfMatrizRF$byClass[11]
  ))
  )
}

dblSplitRatio = 0.8
set.seed(10)

divisao = sample.split(dfAquisição_Numeric_Scale$Dentro_SLA,
SplitRatio = dblSplitRatio)

base_treinamentoNum = subset(dfAquisição_Numeric_Scale,
divisao==TRUE)
medianaTreino = apply(base_treinamentoNum,2,median)
mediaTreino = apply(base_treinamentoNum,2,mean)

base_testeNum = subset(dfAquisição_Numeric_Scale,
divisao==FALSE)
medianaTeste = apply(base_testeNum,2,median)
mediaTeste = apply(base_testeNum,2,mean)


MedianaTT <- rbind(medianaTreino, medianaTeste)
MediaTT <- rbind(mediaTreino, mediaTeste)

#Similaridade por Cosseno
coefCosineMedianaTT <- dist.cosine(MedianaTT)
coefCosineMediaTT <- dist.cosine(MediaTT)

base_treinamentoNum$Dentro_SLA <-
as.factor(base_treinamentoNum$Dentro_SLA)
base_testeNum$Dentro_SLA <-
as.factor(base_testeNum$Dentro_SLA)

Modelo = randomForest(
  formula = Dentro_SLA ~ .,
  data=base_treinamentoNum,
  ntree=1000,
  importance=TRUE
)

dfErrRate <- as.data.frame(Modelo[["err.rate"]])
```

```
dfErrRate <- cbind(ntree = rownames(dfErrRate), dfErrRate)
dfErrRate$ntree <- as.numeric(dfErrRate$ntree)
rownames(dfErrRate) <- 1:nrow(dfErrRate)

names(dfErrRate)[3] <- "Falso"
names(dfErrRate)[4] <- "Verdadeiro"

xintcptOOB = which.min(dfErrRate$OOB)
xintcptErr0 = which.min(dfErrRate$Falso)
xintcptErr1 = which.min(dfErrRate$Verdadeiro)

ggplot(dfErrRate, aes(x=ntree)) +
  geom_line(aes(y = OOB), color = "black") +
  geom_line(aes(y = Falso), color="red", linetype="twodash") +
  geom_line(aes(y = Verdadeiro), color="green",
linetype="twodash") +
  geom_point(aes(x=xintcptOOB, y=), colour = "purple") +
  geom_vline(xintercept = xintcptOOB, colour = "purple") +
  labs(y="Erro OOB", x = "Número de árvores")

quartil_OOB_Err = quantile(dfErrRate$OOB)
dfErrRate = dfErrRate[dfErrRate$OOB < quartil_OOB_Err[4],]
summary(dfErrRate)

ValoresImportancia <- Modelo[["importance"]][,3]

dfImportancia <- data.frame(
  MeanDecreaseAccuracy = ValoresImportancia
)

dfImportancia <- cbind(Atributos = rownames(dfImportancia),
dfImportancia)
dfImportancia <-
dfImportancia[order(dfImportancia$MeanDecreaseAccuracy,decreas
ing=TRUE),]
rownames(dfImportancia) <- 1:nrow(dfImportancia)

dfImportancia %>%
  arrange(MeanDecreaseAccuracy) %>%
  mutate(Atributos=factor(Atributos, levels=Atributos)) %>%
  ggplot( aes(x=Atributos, y=MeanDecreaseAccuracy)) +
  geom_segment( aes(xend=Atributos, yend=0)) +
  geom_point( size=4, color="purple") +
  coord_flip() +
  theme_bw() +
  xlab("")

plot(Modelo)

oob.err<-double(10)
```

```
test.err<-double(10)

for (var_mtry in 1:10) {
  ModeloMtry = randomForest(
    formula = Dentro_SLA ~ .,
    data=base_treinamentoNum,
    ntree=200,
    mtry=var_mtry
  )
  oob.err[var_mtry] = ModeloMtry[["err.rate"]][200,1]

  pred<-predict(ModeloMtry,newdata = base_testeNum[-11])
  tblMatriz_confusao = table(base_testeNum[, 11], pred)
  cnfMatrizRF <- confusionMatrix(tblMatriz_confusao,
positive='1')
  test.err[var_mtry] = 1 -
cnfMatrizRF[["overall"]][["Accuracy"]]
}

previsoes = predict(modelo, newdata = base_testeNum[-11])
tblMatriz_confusao = table(base_testeNum[, 11], previsoes)
cnfMatrizRF <- confusionMatrix(tblMatriz_confusao,
positive='1')
cnfMatrizRF

matplot(1:10 , cbind(oob.err), pch=19 ,
col=c("purple"),type="b",ylab="",xlab="mtry")
legend("topright",legend=c("OOB Erro"),pch=19,
col=c("purple"))


vNtreTreino <- as.vector(dfErrRate$ntree)
qtdTests <- (length(vNtreTreino) * 0.10) %/% 1

power_sim <- random_search(rf_RdmSch,
                           params=list(N=vNtreTreino),
                           n.iter=1,
                           n.sample = qtdTests,
                           base_testeNum = base_testeNum
                           )

dfCnfMatrizRF <- data.frame()
for (i in 1:length(power_sim[["results"]])){
  dfNew <- as.data.frame(power_sim[["results"]][[i]])
  dfCnfMatrizRF <- rbind(dfCnfMatrizRF, dfNew)
}
ntreeRandomSearch <- as.vector(power_sim[["tests"]][["N"]])
dfCnfMatrizRF$ntreeRandomSearch <- ntreeRandomSearch
```

```
dfCnfMatrizRF <-
dfCnfMatrizRF[order(dfCnfMatrizRF$ntreeRandomSearch),]

for (i in 2:length(colnames(dfCnfMatrizRF))){
  dfCnfMatrizRF[,i] <- as.numeric(dfCnfMatrizRF[,i])
}
melhorNtree <-
dfCnfMatrizRF[which.max(dfCnfMatrizRF$Accuracy),21]

ggplot(dfCnfMatrizRF, aes(x=ntreeRandomSearch)) +
  geom_line(aes(y = Accuracy), color = "black") +
  geom_point(aes(x=melhorNtree, y=max(Accuracy)), colour =
"red") +
  geom_vline(xintercept = melhorNtree, colour = "red") +
  labs(y="Acurácia", x = "Número de árvores (Random Search)")

matplot(1:10 , cbind(oob.err,test.err), pch=19 ,
col=c("purple","blue"),type="b",ylab="",xlab="mtry")
legend("topright",legend=c("Erro OOB","Erro Conjunto de
Teste"),pch=19, col=c("purple","blue"))
```