Identificação de Fatores Críticos na Experiência do Usuário do IDEA: uma Abordagem para Seleção de Atributos

Alexandre Ramos Brasil
Pós-Grad. Data Science e Analytics
SENAI/CIMATEC
Salvador, Brasil
alexandre.brasil@aln.senaicimatec.edu.br

João Rafael Rosa da Silva Ribeiro
Pós-Grad. Data Science e Analytics
SENAI/CIMATEC
Salvador, Brasil
joao.r.ribeiro@aln.senaicimatec.edu.br

Nilmar Sousa Pereira
Pós-Grad. Data Science e Analytics
SENAI/CIMATEC
Salvador, Brasil
nilmar.pereira@aln.senaicimatec.edu.br

Éldman de Oliveira Nunes
Pós-Grad. Data Science e Analytics
SENAI/CIMATEC
Salvador, Brasil
eldman.nunes@fieb.org.br

Resumo*—Os avanços tecnológicos permitem que empresas entreguem softwares personalizados para um público exigente. E tem-se no monitoramento contínuo e análise de *logs*, importantes ferramentas para elevar o nível de satisfação do usuário. Este trabalho aplica técnicas de *feature selection* e *machine learning* para identificar atributos que influenciam na experiência do usuário em sistemas web. Os dados analisados são *logs* capturados pela ferramenta Dynatrace, gerados pelo sistema IDEA, inicialmente compostos por quatro tabelas, totalizando 106 atributos e 1.544.663 registros. Após análise exploratória e pré-processamento, obteve-se um conjunto de dados final com 57 atributos e 217.255 registros. Para seleção dos atributos, utilizou-se as abordagens Filtro, *Wrapper* e *Embedded*, baseadas em critérios estatísticos e preditivos. No Filtro, aplicou-se *ANOVA, Chi-square e Mutual-information*; no *Wrapper*, o método escolhido foi o *Reduce Feature Elimination* (RFE); e, na abordagem *Embedded*, foram utilizados *Decision Tree*, *Random Forest* e *Gradient Boosting*. Após o ranqueamento, quatro atributos mais significativos foram selecionados: *javascriptErrorCount*, *visuallyCompleteTime*, *requestErrorCount* e name. Esses atributos alcançaram um desempenho de 0.992657 de acurácia, 0.986963 de precisão, 0.994829 de recall e 0.990819 de F1. O trabalho demonstrou que quatro atributos foram suficientes para obter um desempenho excepcional na classificação, mesmo submetendo-os a outras amostras coletadas. Desta forma, comprovou-se que o método proposto identificou os principais pontos que contribuem para a experiência do usuário.

Palavras Chave*—Experiência do usuário, Monitoramento, Análise de *logs*, *Machine learning*, *feature selection

I. Introdução

A experiência do usuário (*User eXperience* - UX) é a percepção e resposta de uma pessoa ao usar um produto, sistema ou serviço, englobando fatores como usabilidade, acessibilidade e satisfação. O objetivo é proporcionar uma interação eficiente e agradável, alinhada às necessidades e expectativas dos usuários. A melhoria da UX pode aumentar a fidelidade e o engajamento do usuário [1].

Identificar os fatores de risco para a experiência do usuário apresenta desafios significativos, como a variabilidade nas necessidades e expectativas dos usuários, a complexidade dos contextos de uso e a subjetividade das percepções individuais. Além disso, a coleta e análise de dados relevantes pode ser dificultada por limitações tecnológicas e de recursos, enquanto a rápida evolução tecnológica exige adaptação constante. Estes desafios tornam a tarefa de identificar e mitigar riscos de UX complexa e multifacetada [2].

O método proposto por este trabalho foi estruturado em quatro módulos com o objetivo de identificar os fatores críticos que influenciam a experiência dos usuários do Sistema Integrado de Dados, Estatística e Atuação (IDEA), sendo este o sistema que processa os dados finalísticos do Ministério Público do Estado da Bahia. Para isso, foram extraídos informações da base do Dynatrace, que é uma ferramenta projetada para monitorar ambientes de TI, incluindo arquiteturas tradicionais baseadas em servidor, ambientes multi-nuvem, híbridos e microsserviços. A plataforma oferece, além de outras funções, observabilidade de aplicativos e infraestrutura em tempo real, segurança, análise de negócios e automação.

A análise foi feita por meio do monitoramento do IDEA e da utilização de técnicas de *machine learning*. Inicialmente, os dados de *log* foram coletados e pré-processados para remover ruídos e estruturar a informação relevante. Em seguida, aplicou-se algoritmos para seleção de atributos a fim de identificar os fatores críticos.

Este trabalho está organizado do seguinte modo. A seção II aborda o conteúdo teórico sobre os assuntos discutidos. A seção III expõe a metodologia adotada para a realização do trabalho. A seção IV trata dos experimentos realizados, análise dos resultados obtidos e comparação entre as técnicas usadas. Na última seção é apresentado a conclusão do trabalho, as principais contribuições, limitações e os trabalhos futuros.

II. Trabalhos Relacionados

A. Experiência do Usuário

A experiência do usuário (*User eXperience - UX*) é definida pela norma ISO 9241-11: 2018 [3] como as "percepções e respostas do usuário resultantes do uso ou da antecipação do uso de um sistema, produto ou serviço". Desta forma, a abordagem de satisfação está intrinsecamente ligada a suas emoções ao usar um sistema, suas crenças, preferências, graus de conforto resultantes do uso antes, durante e após este uso. Nota-se, portanto, que métricas de qualidade de *software*: eficiência e eficácia, vinculadas a capacidade de operacionalização, não avaliam completamente a usabilidade de um sistema se os níveis subjetivos de satisfação do usuário não estiverem atreladas ao objetivo específico relacionado àquele uso.

Em 2015, Wang e Wang [4] sugeriram que a satisfação do usuário fosse uma medida chave de sucesso do sistema computacional. Dentre os métodos para mensurar a experiência do usuário, o nível de satisfação no sistema IDEA é baseado nos tempos de resposta medidos pelo Apdex.

Segundo [5] o Apdex (*Application Performance Index*) é um padrão aberto que define um método para relatar, avaliar e classificar o desempenho do aplicativo por meio do tempo de resposta à requisição (*Request time - R*) quando comparado a um limite preestabelecido (*Threshold - T*). Este limite de tempo de resposta T é o tempo que foi necessário para que um ativo solicitado fosse devolvido ao usuário de forma satisfatória [6]. Para cada ação do usuário, por exemplo, uma requisição web a uma página de cadastro de processos no sistema IDEA, o índice Apdex classifica esta ação em três classes de resposta: Satisfatório, Tolerável e Frustrado.

a) ***Satisfatório****:* Essa classificação representa os usuários que tiveram uma capacidade de resposta satisfatória ou alta, podendo se concentrar totalmente em seu trabalho.

$$R <= T$$

b) ***Tolerável****:* Apesar das respostas se darem em um tempo distante do ideal (T), essa classificação representa usuários que experimentaram uma capacidade de resposta lenta, mas aceitável por não comprometer o desempenho do sistema.

$$R > T$$

c) ***Frustrado****:* Esta classificação representa os usuários que experimentaram uma capacidade de resposta inaceitável e acabam, muitas vezes, por desistir de usar o sistema.

$$R > (4xT)$$

Por fim, a experiência final do usuário será medido contabilizando todas as amostras do usuário durante a sessão, ou seja, desde o *logon* até sua desconexão deliberada ou abrupta. O Apdex Score mede o índice de satisfação deste usuário por meio de uma escala que varia entre 0 e 1 (0 = usuário não satisfeito, 1 = usuário totalmente satisfeito) em função de T que é dado pela equação (1):

$$\text{Apdex} = \frac{\text{satisfatório} + \text{tolerável}/2}{\text{total de amostras}} \quad (1)$$

Os valores obtidos da equação (1) se comparados a classificação padrão do Apdex descrita na documentação oficial e por *softwares* no mercado, seguem o referencial da tabela I:

Tabela I
Apdex Score - Faixa de classificação

[0,94 - 1]	Nível excelente de performance
[0,85 - 0,93]	Nível bom de performance
[0,70 - 0,84]	Nível aceitável de performance
[0,50 - 0,69]	Nível ruim de performance
[0,00 - 0,49]	Nível inaceitável de performance

Como cada sistema possui sua especificidade de tempo de resposta T, então é desejável que essas faixas de classificação, conforme tabela I, possam ser parametrizadas. Nota-se que esta métrica é centrada na sensibilidade do valor de T. Portanto, se o valor de T for muito baixo pode dar impressão que usuários podem estar insatisfeitos com a maioria das requisições. Em contrapartida, com um valor de T muito alto, pode oferecer uma medida alta de satisfação que não corresponde a realidade. O valor ideal de T, segundo *apdex.org*, reside no valor médio global ao longo do uso dos ativos na organização por diferentes usuários.

O Dynatrace que adota o Apdex como métrica de desempenho, classifica cada ação do usuário, como também a sessão deste usuário. A classificação da avaliação da experiência do usuário para cada ação foi usada neste artigo como atributo independente ou atributo alvo *(target)*, pois, o foco está na análise dos atributos que podem ter levado àquela classificação. Para este fim utilizou-se de técnicas de Seleção de Atributos (*Feature Selection*) para análise de relevância das dezenas de atributos para a classificação proposta.

B. Seleção de Atributos

Dentro de uma estratégia de pré-processamento, a seleção de atributos se encarrega de lidar com grandes quantidades de atributos de entrada, buscando reduzir a dimensionalidade do conjunto de dados original para um subconjunto mais relevante, a fim de mitigar a 'maldição da dimensionalidade' (curse of dimensionality) [7]. Além disso, a partir da seleção de atributos é possível aumentar a interpretabilidade do modelo preditivo, diminuir a quantidade de memória e tempo de processamento dos modelos de aprendizado de máquina nos dispositivos computacionais e, em alguns casos, até otimizar a performance deste modelo. [8] [9]. Tal otimização decorre da resolução de problemas de redundância e correlação, da baixa colinearidade do atributo independente com o atributo dependente (*target class*) e do elevado *overfiting*, ou seja, de modelos treinados super ajustados aos dados de treinamento [10].

Outra estratégia para a redução da dimensionalidade é a extração de atributos, que consiste em combinar os atributos de entrada originais para produzir novos atributos. Essa abordagem pode ser verificada, por exemplo, nas técnicas PCA (Principal Component Analysis) e LDA (Linear Discriminant Analysis). Desta forma, a transformação dos atributos diminui a interpretatividade, o que afastaria a hipótese de verificação da relevância de apenas alguns atributos como determinísticos a experiências frustradas do usuário [11]. Em contrapartida, a seleção de atributos não modifica os dados originais, pois, seleciona um subconjunto de atributos a partir de modelos matemáticos cuja análise se baseia na correlação entre atributos.

Ainda conforme [11], existem três tipos de abordagens para a seleção de atributos – Filtros (*Filters*), *Wrappers* e *Embedded*. Será discutido nas próximas seções sobre cada uma das abordagens, suas vantagens e desvantagens e, para cada abordagem, alguns métodos amplamente aplicados e difundidos na literatura científica.

1) Abordagem Baseada em Filtros

A abordagem de seleção de atributos por Filtros não depende de um algoritmo de aprendizado de máquina [12]. Sua estratégia é mensurar, por meio de testes estatísticos e pontuações, a relevância individual ou combinatória dos atributos de análise com o atributo alvo que queremos prever.

Conforme o fluxograma, mostrado na figura 1, os atributos irrelevantes serão descartados e os atributos com as maiores pontuações serão mantidos. Essa seleção do número de atributos que são submetidos ao modelo é definido pelo usuário em tempo de execução.

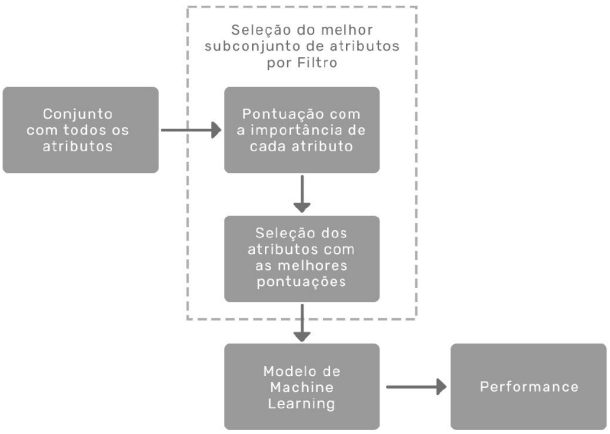

Figura 1. Fluxograma da abordagem Filtro. Fonte: [13]

Somente após subconjunto dos atributos com as melhores pontuações que os estimadores (modelos) de *machine learning* serão aplicados e a performance será mensurada.

Computacionalmente, os métodos dessa abordagem são de alto desempenho quanto ao tempo de processamento e podem resolver problemas de duplicação de atributos, *overfiting* e de correlação. São amplamente conhecidas e implementadas em inúmeras bibliotecas de *machine learning* [14].

A principal desvantagem do uso deste método é a falta de consideração das interações entre os atributos. Assim, os métodos de filtro podem deixar de capturar interações de dados que são importantes para a previsão [14].

Existe na literatura científica dezenas de métodos de filtragem que podem ser agrupados em função de sua aplicabilidade – classificação, regressão e clusterização – e pelas classes de filtro – univariado e multivariado [9]. Entretanto, partindo da necessidade de uma análise individualizada de cada atributo com objetivo de mensurar seus respectivos graus de importância, serão detalhadas três técnicas de filtragem: Análise de variância (ANOVA), Chi-quadrado(*Chi-Square*) e Informação mútua (*Mutual Information*).

a) Análise de Variância (ANOVA): Segundo [15], ANOVA é uma técnica estatística utilizada para avaliar padrões em dados complexos e variados entre 3 ou mais grupos, sendo as amostras independentes. O objetivo é inferir se as diferenças observadas são consequências do acaso ou se são estatisticamente significativas.

Além da simplicidade de entendimento e da facilidade de ser encontrado em muitas bibliotecas de machine learning, outra vantagem da utilização da ANOVA é a alta performance na comparação entre mais de dois grupos, o que torna viável aos experimentos submetidos neste artigo [15].

Para [16], uma limitação da ANOVA é que, para um maior grau de confiança nos resultados, exige-se que cada grupo siga uma distribuição normal e que haja homogeneidade de variâncias entre os grupos. Apesar da técnica demonstrar a existência de uma diferença significativa entre as medianas de pelo menos dois grupos, no entanto, ela não consegue explicar essa distinção. Outra desvantagem é que esta técnica é sensível a *outliers* que podem enviesar os resultados.

Para determinar a importância do atributo, calcula-se a razão F, que é obtida dividindo a variância entre os grupos pela variância dentro dos grupos. Um alto valor calculado para F sugere que a variância entre as médias dos grupos é grande em comparação com a variância dentro dos grupos, indicando que pelo menos uma média de grupo é diferente [16].

b) Chi-square (Chi-quadrado de Pearson): O teste *Chi-quadrado* utiliza a comparação entre frequências esperadas e observadas entre duas variáveis categóricas nominais ou ordinais para verificação de independência, ou seja, serve para avaliar quantitativamente a relação entre o resultado de um experimento e a distribuição esperada para o fenômeno. Além disso, também pode ser empregado para medir a aderência (*Goodness of fit*) e a homogeneidade. [17]

Diferente do ANOVA, este teste não é paramétrico, pois não depende de parâmetros como média, variância, desvio etc. Entretanto, há similaridade entre as técnicas, pois também utiliza o teste de hipótese, da significância e do grau de liberdade e uma tabela de contingência [18]. Sua fórmula é expressa pela equação (2):

$$\chi^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i} \tag{2}$$

Onde:

- i se refere a quantidade de classes
- O_ié o valor observado de uma determinada classe
- E_i é o valor esperado desta classe

Segundo [19], atributos que mostram uma forte relação de dependência com o alvo são considerados importantes para uma previsão. Por exemplo, para testar se o atributo "*name*" tem forte dependência com a experiência do usuário, deve-se rejeitar a hipótese nula, ou seja, existe uma associação significante entre os atributos.

A partir da tabela de distribuição e da escolha do nível de significância α, então se o valor de x^2 é superior ao valor Chi-quadrado crítico então está estatisticamente provado sua relação de dependência.

c) Mutual-Information: Técnica usada para medir o quanto de informação uma variável aleatória (X) está contida em outra variável aleatória (Y). O objetivo da informação mútua é selecionar atributos independentes com alta relação com o atributo dependente e com baixa redundância entre esses atributos independentes [20]. Sua fórmula é expressa pela equação (3):

$$I(X;Y) = H(X) - H(X|Y) \tag{3}$$

onde:

- $I(X;Y)$ é a informação mútua entre X e Y;
- $H(X)$ é a entropia de X;
- $H(X|Y)$ é a entropia condicional de X dado Y.

O valor da informação mútua I(X;Y) = 0 indica que X é completamente independente de Y, entretanto, caso I(X;Y) $>$ 0 indica a relevância de X em Y. Portanto, o objetivo da informação mútua é diminuir a entropia de X.

Quanto maior o I(X;Y) mais dependente a informação de X é de Y. Em contrapartida, a redundância entre atributos independentes X' e X" é indesejável à predição de um modelo de *machine learning*, ou seja, a informação mútua I(X';X") $>$ 0 indicaria forte dependência entre X' e X" quanto maior este valor. Consequentemente, a partir da relação de relevância e redundância a técnica deve "rankear" valores mais altos para atributos mais relevantes do conjunto original de dados [20].

2) Abordagem Baseada em Wrapper

Baseado na busca heurística de um subconjunto amostral das combinações possíveis, a abordagem *Wrapper*, diferente da abordagem por Filtro, utiliza-se de um modelo de aprendizado de máquina para a seleção dos atributos mais importantes a partir da performance deste modelo indutor, ou seja, a seleção destes atributos estão condicionados ao viés algorítmico do modelo (*machine learning bias*) [12], A figura 2 mostra o funcionamento desta abordagem.

Portanto, nesta abordagem, a busca pela seleção dos atributos é iterada para cada subconjunto e se encerra quando se atinge a performance otimizada do modelo específico, funcionando como uma caixa-preta [21] [22].

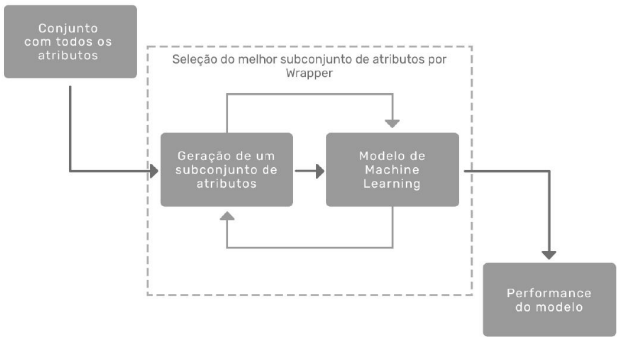

Figura 2. Fluxograma da abordagem Wrapper. Fonte: [13]

Experimentos já demonstraram que a performance usando esta abordagem otimiza a seleção dos atributos por ser associado a um algoritmo de modelagem real. Além disso, o modelo externamente adotado como indutor dos atributos pode ser diferente do efetivamente utilizado no processo de aprendizado, o que pode acarretar no aumento da precisão por possuir um bias de aprendizado distinto. Como desvantagem, a depender do tipo do modelo e do elevado número de atributos de análise o custo computacional e a complexidade é aumentada, haja vista que um classificador é gerado para cada iteração [14].

As abordagens de Filtro e *Wrapper* não se excluem como estratégia, pois, uma abordagem híbrida pode combinar sequencialmente métodos de Filtros para descartar variáveis redundantes e, logo após, de posse dos atributos mais significativos aplicar métodos

a) Reduce Feature Elimination (RFE): A técnica de RFE baseia-se na escolha de um estimador externo que, após inicialmente treinado com todos os atributos do dataset, será responsável pela classificação de atributos mais importantes para os menos importantes por meio dos pesos que este estimador atribui a cada atributo preditor. Recursivamente, os atributos menos importantes serão eliminados e um novo ciclo de treinamento deste estimador novamente classifica os atributos mais importantes até que o total desejado de atributos sejam alcançados [23].

Uma das vantagens no uso do RFE é a capacidade de manipular com conjunto de dados de alta dimensionalidade. Também podem lidar com interações entre os atributos. Por fim, sua independência com o estimador, podendo ser qualquer outro modelo supervisionado.

Como limitação, o RFE pode ter um custo computacional elevado ou até inviável para grandes conjuntos de dados. Também não é a melhor abordagem para conjuntos de dados com muita correlação entre os atributos. E, por fim, não lida bem com ruídos ou atributos mais irrelevantes [24].

Uma análise relevante para modelos de classificação multi classes ao utilizar a técnica do RFE reside em *datasets* com alto desbalanceamento de uma classe frente às demais, podendo levar o modelo treinado a priorizar atributos que con-

duzam a uma classificação da classe com maior "facilidade" de predição, ou seja, em uma taxa de erro menor [21].

3) Abordagem Baseada em Embedded

A terminologia *Embedded* deriva-se do processo que incorpora a seleção dos atributos ao treinamento do modelo em paralelo, conforme visualizado na figura 3. Assim como a abordagem *Wrapper*, a seleção de atributos é condicionada a escolha do modelo [10].

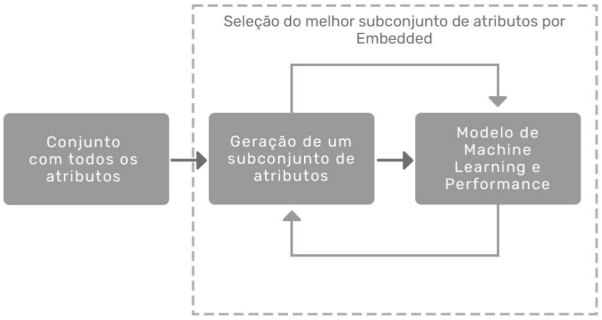

Figura 3. Fluxograma da abordagem Embedded. Fonte: [13]

Conforme o fluxograma da figura 3, esta abordagem condiciona o subconjunto ideal de atributos (*feature importance*) ao próprio treinamento e, consequentemente, descarta os demais atributos. Naturalmente, o custo computacional é o mesmo custo de treinamento do modelo.

a) Árvore de Decisão: É um algoritmo de aprendizado de máquina supervisionado largamente conhecido para problemas de classificação e regressão. Pode ser entendida como uma estrutura de dados definida recursivamente como: i) nó-raiz (*root node*) que é um entre todos os atributos; ii) nós de decisão (*decison node*) que corresponde a um teste "se-então"sobre um atributo; iii) nós-folha (leaf nodes) que corresponde a classe alvo (*target class*) ou o valor resposta [25].

Assim como um fluxograma, cada nó de decisão está relacionada hierarquicamente a outro nó de decisão ou ao nó-folha. Em uma árvore de decisão, inicia-se o percurso partindo do nó-raiz, passando pelos nós de decisão até o nó-folha. Alguns questionamentos podem surgir ao definir a estratégia para construção da árvore: a) Visto que pode existir árvores de decisão completas sendo que nem todos os atributos foram selecionados como nós de decisão, então como escolher os atributos mais relevantes? b) Já árvores de decisão com complexidades e ramificações diferentes alcançam as mesmas classes alvo, qual é a mais representativa para a representação no mundo real?

Diante do exposto, para [26] a construção otimizada da árvore de decisão parte da boa escolha do nó raiz e dos nós de decisão posteriores. Esta escolha exigem medidas de avaliação que orientem a hierarquia entre os atributos e os valores que devem ser testados em cada nó de decisão. Problemas como esses podem ser mitigados por meio de cálculos matemáticos que avaliam o grau de distribuição dos dados nos atributos de acordo com a classe alvo. Dentre as mais famosas medidas de avaliação, entropia e ganho de informação se destacam na literatura científica. Partindo da entropia, o algoritmo confere o ganho de informação de cada atributo. Aquele que apresentar maior ganho de informação será a variável do primeiro nó da árvore.

Segundo [27], um problema a ser resolvido na construção da árvore de decisão é o *overfiting*. *Overfiting* ocorre sempre que uma árvore fica condicionada ao histórico de dados de treinamento, ou seja, suas ramificações e complexidade garantem 100% de adequação a classe alvo. Entretanto, quando esta mesma árvore treinada é submetida a dados novos de testes ou validação, as classes alvo preditas não correspondem ao esperados. Diante deste problema, estratégia de "Podas"(*Prunning*) remove as extensas ramificações das árvores evitando o ajuste excessivo aos dados de treinamento e a perda de generalização para novos dados.

b) ***Random Forest****:* Os algoritmos que implementam as estratégias de ensemble, agregam as previsões de vários estimadores, a fim de melhorar a generalização e robustez de um único estimador. Dentre estes algoritmos de aprendizado supervisionado, o Random Forest, tem ampla notoriedade na comunidade científica devido a alta aplicabilidade dada sua robustez, desempenho e baixo overfitting [28].

Neste artigo, o Random Forest foi selecionado como estimador por sua capacidade de calcular a importância de cada atributo (*feature importance*), a capacidade de processar grandes volumes de dados multivalorados e sua versatilidade para resolver tarefas de classificação.

O algoritmo de Random Forest constitui um conjunto de árvores de decisão geradas aleatoriamente por meio de técnicas de *bootstrap aggregation* ou *bagging* para seleção de amostras e na seleção de atributos 29.

Resumidamente, a técnica de bagging pode ser entendida como uma etapa de seleção aleatória de amostras para treino do modelo. Para cada amostragem, um subconjunto de atributos são selecionados aleatoriamente e as árvores são individualmente construídas sem podas, portanto, em sua máxima profundidade. Para problemas relacionados à classificação, o resultado predito por cada árvore de decisão será caracterizado pela moda, ou seja, pela maior ocorrência dentre as classes preditas [28] [30].

Nos experimentos, o grau de importância de cada atributo, implementado pelo Random Forest, foi medida por meio da métrica Importância de Gini(*Gini Importance*) ou Média da Redução de Impureza (*Mean Decrease Impurity*).

Para cada árvore de decisão, a Importância de Gini quantifica a redução na impureza de Gini obtida pela divisão dos nós com base em um atributo específico. A média de todas as árvores da floresta é a medida da importância do atributo. Ao final, as características que levam consistentemente a uma maior redução de impurezas em todo o conjunto recebem pontuações mais altas de Importância de Gini [31].

c) ***Gradient Boosting****:* Para [29]o algoritmo Gradient Boosting é uma técnica de boosting para problemas de classificação e regressão e, assim como o Random Forest,

está incluído no grupo de classificadores ensemble. Entretanto, diferentemente do *Random Forest*, nesta técnica de *Boosting* cada preditor fraco é treinado de forma sequencial e adaptativa visando diminuir iterativamente os erros de previsão do modelo anterior.

Ainda segundo [29], existe uma função de perda (*loss function*) que calcula o resíduo (*gradient*) de cada modelo fraco treinado. O resíduo é a função que difere o valor predito com o valor esperado. Em cada iteração, o algoritmo calcula o gradiente da função de perda em relação às previsões do conjunto atual, otimiza os pesos e então treina um novo modelo fraco para minimizar esse gradiente. O processo é repetido até o critério de parada.

A abordagem nesse artigo ao selecionar o *Gradient Boosting* deriva de sua grande robustez , pois, lida bem com grande volume de dados e com tipos diferentes de atributos, como numéricos e categóricos. Não só um modelo otimizado é gerado ao final do processo, como os scores dos atributos que levaram à predição final.

III. Método

Os dados que originaram o dataset deste trabalho foram extraídos dos *logs* coletados pelo Dynatrace via API (*Application Programming Interface*). Esses dados são gerados a partir da interação do usuário com o sistema. Onde informações variadas são coletadas; da localização, provedor de internet e sistema operacional às páginas requisitadas, tempo de respostas e erros obtidos.

Os dados e informações disponibilizados pelo Dynatrace para monitorar a experiência do usuário podem ser complexos e nem sempre estruturados para uma análise direta. Isto ocorre porque o Dynatrace, com o intuito de proteger informações sensíveis e garantir a segurança dos dados empresariais, e suas próprias regras de negócio, oculta ou torna anônima certos detalhes específicos. Dessa forma, os *logs* não contêm chaves de referência claras ou dados organizados de maneira imediata para extração e análise. O que implica desafios adicionais para a coleta e o processamento de informações de maneira eficiente e precisa, exigindo métodos complementares de tratamento e estruturação dos dados antes de sua análise detalhada.

A. Dataset

O Dynatrace disponibiliza quatro tabelas para consulta e extração: userAction, userEvent, userError e userSession. Sendo essa última a principal, por trazer dados generalistas e informações relacionadas ao usuário. Porém, apenas é permitido a extração de userAction, userEvent e userError uma por vez e se unidas a userSession. E, por isso, daqui para frente as entidades serão mencionadas com seus nomes originais, entendendo que userSession já faz parte de seus contextos. A documentação e o dicionário de dados estão disponíveis em [32].

Com intuito de selecionar uma base de dados adequada, optou-se por limitar o período de extração em uma semana. Assim, um conjunto de dados com dimensão de 506,9mb foi coletado de 18 a 24 de fevereiro de 2024. Posteriormente, para replicar o fluxo de trabalho desenvolvido, mais duas extrações de períodos distintos foram testadas e submetidas ao processo do modelo proposto, obtendo resultados semelhantes.

Após análise inicial, identificou-se que as entidades com maior significância para este estudo foram userAction e userError, e descartou-se a entidade userEvent. Com isso, iniciou-se um processo de simplificação das estruturas, com eliminação e criação de atributos para acomodar as regras negociais e dar sentido ao dataset.

B. Análise Exploratória de Dados (AED)

No presente trabalho, a AED foi conduzida com várias etapas específicas. Inicialmente, estudou-se os dados extraídos, que consistiam em quatro tabelas distintas. Durante essa análise, identificou-se a ausência de chaves estrangeiras, o que dificultou a junção (*joins*) das entidades. Consequentemente, houve a necessidade de reorganizar os dados para facilitar futuras análises. Com isso, pode-se identificar dados nulos, valores ausentes e definir estratégias de preenchimento; e também atributos a serem convertidos ou categorizados, além de colunas repetidas, que precisaram ser tratadas para assegurar a qualidade e a integridade do dataset. Essas etapas foram cruciais para a preparação dos dados, garantindo uma base sólida para as análises subsequentes e a construção de modelos mais precisos e confiáveis.

C. Modelo Proposto

Esta seção apresenta o modelo proposto para a identificação de fatores críticos na experiência do usuário do IDEA, utilizando técnicas de seleção de atributo e aprendizado de máquina. O modelo definido é composto por quatro módulos principais, cada um responsável por uma etapa crucial do processo de análise e processamento dos dados: Módulo 1 - Filtragem dos dados; Módulo 2 - Pré-processamento; Módulo 3 - Seleção de Atributos e Módulo 4 - Avaliação, como podem observar na figura 4.

A seguir detalha-se cada um desses módulos, explicando suas funções e métodos aplicados.

1) Módulo 1 - Filtragem dos dados

Neste módulo, focou-se na limpeza dos dados brutos coletados. Este passo é essencial para garantir que o conjunto de dados utilizado nas etapas subsequentes esteja livre de inconsistências, valores ausentes e ruídos que possam comprometer a qualidade da análise.

Após o descarte das entidades userSession e userEvent, mantiveram-se as tabelas userAction e userError, selecionadas por sua maior relevância para este estudo. Inicialmente, a tabela userAction possuía 106 atributos e 1.544.663 registros, enquanto a tabela userError continha 63 atributos e 121.982 registros. Para reduzir o escopo da análise, foram extraídos apenas os acessos oriundos do estado da Bahia. A decisão de focar na Bahia se deve à presença de acessos de outros estados e países em quantidades insuficientes para formar padrões claros para análise.

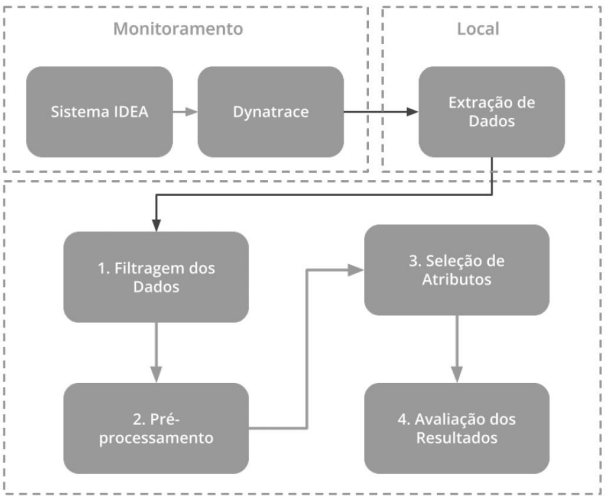

Figura 4. Fluxograma da Estrutura do Método. Fonte: Autor

Com os primeiros estudos, identificou-se a necessidade de criar a coluna Tipo_erro para categorizar os registros em userError, além das colunas Erro_Sys, Erro_Null e Erro_Outros para contabilizar erros por usuário na tabela userAction. Os erros foram agrupados em tipos específicos, contabilizados por usuário e, ao final, os resultados foram mesclados com a tabela userAction.

Na exclusão de colunas, algumas estratégias foram adotadas: colunas com mais de 95% de registros nulos não foram aproveitadas; colunas do tipo inteiro, com a finalidade de contagem, que estavam zeradas em sua totalidade, foram descartadas; colunas categóricas, mas que possuíam apenas uma categoria, foram excluídas; colunas boleanas ou com identificadores irrelevantes também não foram utilizadas no trabalho.

Nesta etapa, o empenho foi em eliminar, juntar tabelas e excluir colunas. Além de aplicar filtros para diminuir a dimensionalidade dos registros e tornar o escopo mais próximo do negócio. O trabalho executado pode ser visualizado na síntese exposta na figura 5.

2) Módulo 2 - Pré-processamento

O pré-processamento envolve a transformação dos dados limpos em um formato adequado para a análise. Esta etapa inclui a normalização, padronização e outras técnicas de preparação dos dados para otimizar o desempenho dos algoritmos de aprendizado de máquina.

O dataset obtido ao término da etapa anterior, possui 57 atributos e 217255 registros já classificados por meio da coluna escolhida como target deste trabalho: apdexCategory. Este atributo possui as seguintes classes: satisfied; tolerating; frustrated. Após a definição da variável alvo e suas classes, pontos relevantes foram analisados:

a) ***Distribuição Normal****:* é uma das distribuições de probabilidade mais importantes e utilizadas em estatísticas e *machine learning*. O Shapiro-Wilk, teste estatístico para verificar a normalidade, mostrou que nenhum dos atributos deste dataset está em uma distribuição normal. Isso implica em maiores desafios, principalmente porque muitos métodos estatísticos e de *machine learning* assumem a normalidade dos dados. Com isso, será necessário aplicar transformações nos dados para aproximá-los de uma distribuição normal.

b) ***Matriz de Correlação****:* Inicialmente, mapeou-se as classes satisfied, tolerating e frustrated para 1, 2 e 3, respectivamente. Isso se deve à necessidade de conversão para obtenção dos coeficientes de correlação. Neste trabalho, além de uma análise generalista com todos os atributos, utilizou-se a matriz de correlação para identificar aqueles que possuem coeficiente de correlação superior a 0.33 com o target. Para esses, aplicou-se individualmente ferramentas como histograma e boxplot para um estudo mais detalhado.

c) ***Conversão de Atributos e Preenchimento de Valores****:* Atributos do tipo object foram convertidos para o tipo category. Verificou-se o percentual de atributos nulos e dados faltantes, constatando-se um número considerável de valores faltantes ou nulos, com destaque para as colunas Erro_Null, Erro_Outros e Erro_Sys. Para essas colunas, preencher com zeros foi a estratégia escolhida. Para as demais, optou-se por preencher com a média.

d) ***Base de Treino e Teste****:* O dataset foi separado mantendo um percentual de 80% para treino e 20% para teste.

e) ***Pipeline e Transformadores****:* foram criados 4 *pipelines* para acomodar transformadores e funções de pré-processamento, de acordo com a tabela II:

Tabela II
PIPELINES E DETALHES DA COMPOSIÇÃO

Nome	Atributos	Descrição
pipeline_media	Números Reais	Preenche os atributos ausentes com a média, depois normaliza.
pipeline_zero	Números Inteiros	Preenche os atributos ausentes com zero, depois normaliza.
pipeline_moda	Categóricos	Preenche atributos ausentes com a moda, depois substituí o valor pela sua ocorrência
pipeline_preprocessor	Todos	Aplica os demais pipelines por meio do tipo do atributo

Esta estrutura cria um processo automatizado para receber uma base de dados e realizar as transformações e processamentos necessários para preparação do dataset final.

3) Módulo 3 - Seleção de Atributos

Nesta seção, aborda-se o processo de seleção de atributos do *dataset* utilizando diversas técnicas de seleção. O objetivo é identificar os atributos mais descritivos e relevantes, comparando os resultados obtidos de diferentes experimentos. As técnicas empregadas são categorizadas em três abordagens principais: Filtros, *Wrapper* e *Embedded*.

a) ***Abordagem de Filtros****:* As técnicas de seleção de atributos baseadas em filtros são utilizadas para avaliar a relevância de cada atributo de forma independente do modelo preditivo. As técnicas experimentadas nesta abordagem incluem ANOVA, Chi-squared e Mutual-information.

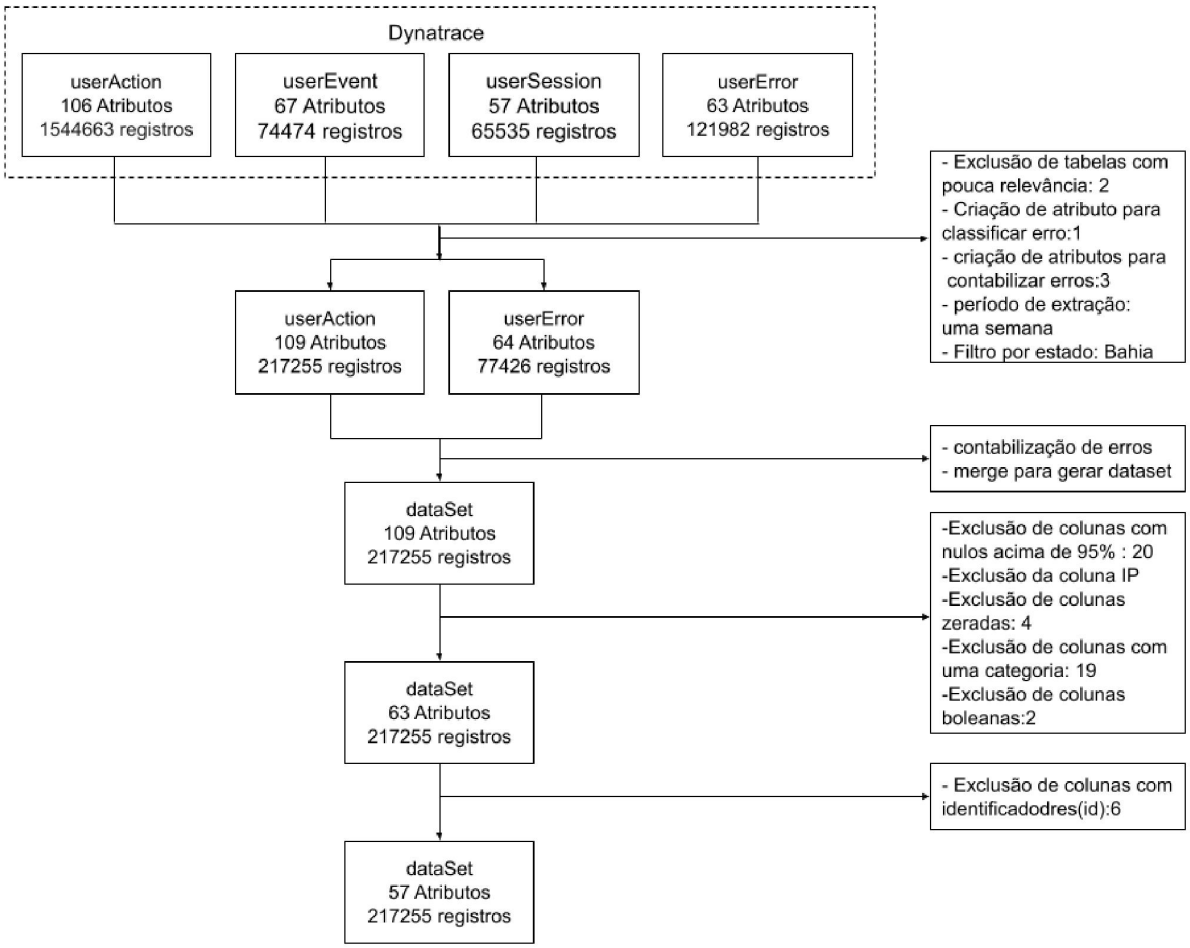

Figura 5. Fluxograma da Filtragem dos dados. Fonte: Autor

b) ***Abordagem Wrapper****:* Na abordagem Wrapper, os atributos são selecionados considerando a performance de um modelo preditivo específico. O método escolhido foi o RFE (Recursive Feature Elimination).

c) ***Abordagem Embedded*** *:* Na abordagem Embedded, a seleção de atributos é realizada durante o processo de treinamento do modelo. Foram escolhidos três modelos para esta abordagem: Decision Tree; Random Forest e Gradient Boosting

d) ***Ferramentas Utilizadas****:* Os experimentos foram conduzidos utilizando o Google Collaboratory, a linguagem Python (versão 3.10.12) e a biblioteca SciKit-Learn (versão 1.2.2). Essas ferramentas foram essenciais para a execução e análise dos resultados obtidos em cada técnica de seleção de atributos.

4) Módulo 4 - Avaliação

Nesta seção, o objetivo é avaliar a eficácia dos modelos desenvolvidos para a seleção de atributos e a melhoria da experiência do usuário. Para isso, utilizou-se diversas ferramentas e bibliotecas que auxiliaram no processo de análise e visualização dos resultados obtidos.

A biblioteca Pycaret (versão 3.3.2) foi uma das principais ferramentas utilizadas para conduzir experimentos de *machine learning*. PyCaret é uma biblioteca de código aberto em Python que facilita a experimentação de *machine learning*, oferecendo uma abordagem simplificada para a criação e comparação de modelos. O *PyCaret* foi utilizado para automatizar a seleção de modelos, a validação cruzada e a comparação de desempenho entre diferentes algoritmos. Essa biblioteca permitiu uma avaliação rápida e eficiente dos modelos, proporcionando insights sobre a precisão, *recall*, *F1-score* e outras métricas relevantes.

Além disso, Matplotlib foi amplamente utilizada para a visualização dos dados e dos resultados dos experimentos. Matplotlib é uma biblioteca de plotagem em 2D que permite a criação de gráficos e figuras de alta qualidade. Utilizando esta biblioteca, foi possível criar histogramas, boxplots, matrizes de correlação e outros tipos de gráficos que auxiliaram na interpretação dos dados e dos resultados dos modelos. A

visualização gráfica foi essencial para identificar padrões, tendências e outliers nos dados, bem como para comparar o desempenho dos modelos de forma clara e intuitiva.

Para garantir a comparabilidade dos resultados, aplicou-se diversas técnicas de normalização aos dados antes da avaliação. O *Min-Max Scaling* foi utilizado na normalização para ajustar a escala dos atributos, garantindo que todos os dados estivessem na mesma faixa de valores. Essa etapa foi crucial para evitar que atributos com diferentes escalas influenciassem desproporcionalmente o desempenho dos modelos. Com a normalização, foi possível assegurar que os resultados refletissem de maneira justa a eficácia dos modelos na seleção de atributos e na melhoria da experiência do usuário.

Ao final deste módulo, as ferramentas e técnicas mencionadas permitiram uma avaliação abrangente e detalhada dos modelos desenvolvidos. A utilização de PyCaret facilitou a experimentação e a comparação entre diferentes algoritmos, enquanto Matplotlib proporcionou uma visualização clara dos resultados. As técnicas de normalização asseguraram a integridade e a comparabilidade dos dados, contribuindo para a robustez das conclusões alcançadas

IV. Experimentos

Esta seção trata dos experimentos realizados e dos resultados obtidos, comparando o tempo de execução, a relevância dos dez maiores atributos e por fim, destacando a quantidade de atributos que tiveram uma relevância menor que 0,1%, sendo assim, classificados como irrelevantes.

A. Abordagem Baseada em Filtros

1) ANOVA

Essa foi a primeira técnica experimentada. O processamento da base de treinamento levou cerca de 0,59 segundos. A figura 6 mostra o percentual de relevância dos atributos.

Com 30% de relevância, o atributo *javascriptErrorCount* ficou em primeiro lugar, seguido do *requestErrorCount* com 15%,em terceiro ficaram empatados os atributos name e totalErrorCount, com 6% cada. Para esta técnica, os erros de JavaScript são os mais relevantes na determinação da experiência do usuário,, além disso, 14 atributos foram identificados como irrelevantes receberam um percentual menor que 0,1%.

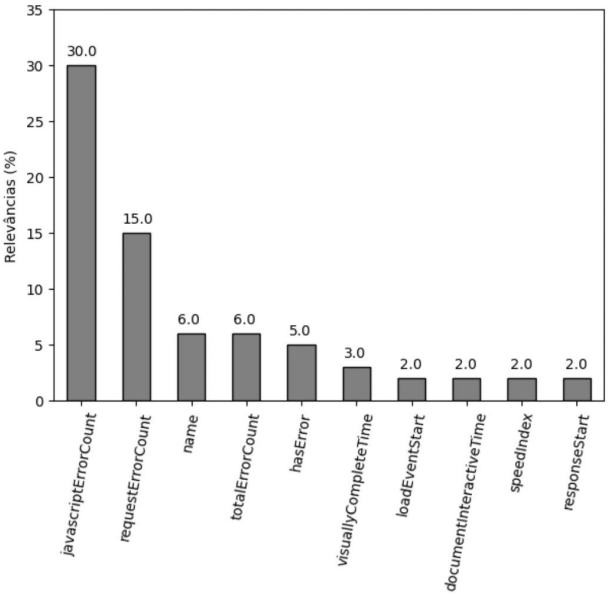

Figura 6. Resultado do Experimento 1 com ANOVA. Fonte: Autor.

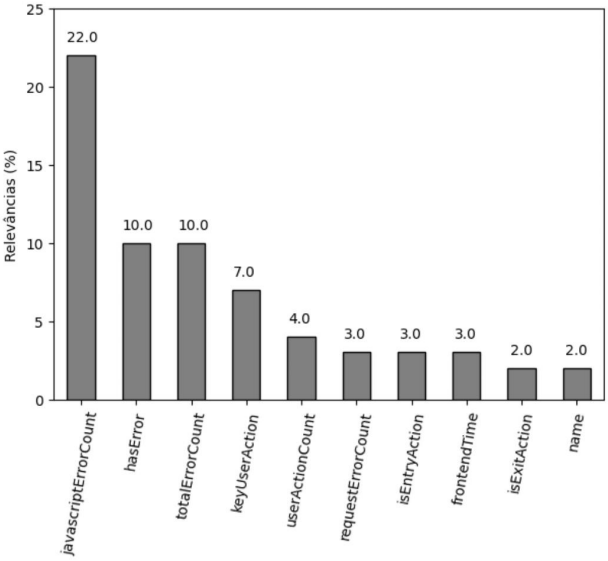

Figura 7. Resultado do Experimento 2 com Chi-Square. Fonte: Autor.

2) Chi-Square

O segundo exeprimento foi realizado com um tempo de processamento de 2,23 segundos. O gráfico da figura 7 exibe os atributos mais relevantes para o *Chi-Square*.

Os atributos *javascriptErrorCount*, *requestErrorCount* e *name* reduziram sua relevância, sendo que o *javascriptErrorCount* saiu de 30% para 22% mantendo a liderança, o *requestErrorCount* caiu de 15% para 3% de relevância e o *name* foi para a decima posição com 2%.

A segunda posição foi ocupada pelos atributos *hasError* e *totalErrorCount* ambos com 10% de relevância. Novos atributos aparecem no gráfico: *keyUserAction* (7%), *userActionCount* (4%), *isEntryAction* (3%), *frontendTime* (3%) e *isExitAction* (2%).

Os erros de JavaScript continuam sendo o fator determinante para a experiência do usuário. Assim como no experimento anterior, 14 atributos foram identificados como irrelevantes.

3) Mutual Information

O último experimento para abordagem baseada em filtro levou cerca de 3 minutos. O gráfico da figura 8 mostra os resultados desse experimento.

Pela primeira vez, o atributo visuallyCompleteTime ocupou o primeiro lugar, com 8% de relevância, seguido por duration

e speedIndex, ambos empatados com 7%. Em terceiro lugar, o atributo name apareceu com 6% de relevância..

O mutual information identificou que o tempo é o fator mais relevante para determinar a experiência do usuário, sendo que o tempo necessário para exibir o conteúdo da página na tela do dispositivo é o mais significativo.

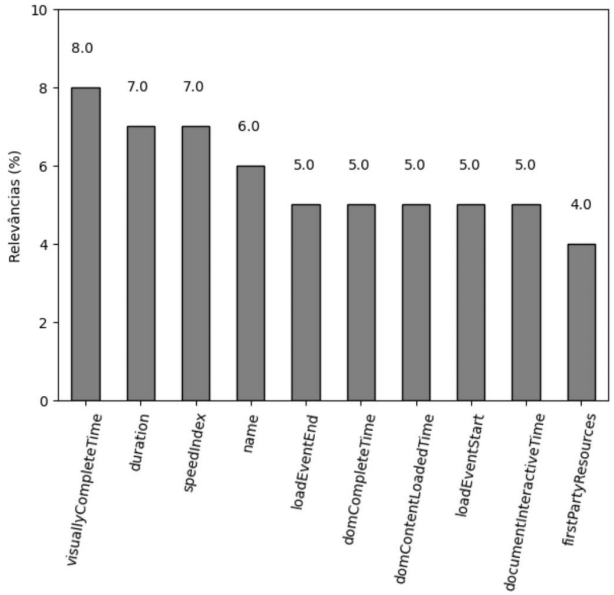

Figura 8. Resultado do Experimento 3 com Multual-Information. Fonte: Autor.

B. Abordagem Baseada em Wapper

1) RFE (Recursive Feature Elimination)

Neste trabalho, experimentou-se apenas o algoritmo *Recursive Feature Elimination (RFE)*, com a abordagem *Wapper*, devido ao tempo de processamento significativamente mais longo associado à natureza dessa abordagem.

O atributo name foi o mais importante, com 13% de relevância, seguido por requestErrorCount, com 11%. Em terceiro lugar, ficaram empatados os atributos visuallyCompleteTime e javascriptErrorCount, ambos com 7% de relevância.

O tempo de processamento levou cerca de 40 minutos. Desta vez o atributo *name* responsável por informar o recurso que esta sendo acessado pelo usuário foi o fator mais relevante para a experiência do usuário.

C. Abordagem Baseada em Embedded

1) Decision Tree

Esse foi o primeiro experimento com a abordagem *Embedded*. A execução do treinamento levou 11 segundos. O gráfico na figura 10 exibe os atributos mais importantes para o modelo.

Novamente o atributo *visuallyCompleteTime* surge como o mais relevante, tendo 43% de relevância, seguido pelos atributos *requestErrorCount* e *javascriptErrorCount* com 29% e 26% respectivamente.

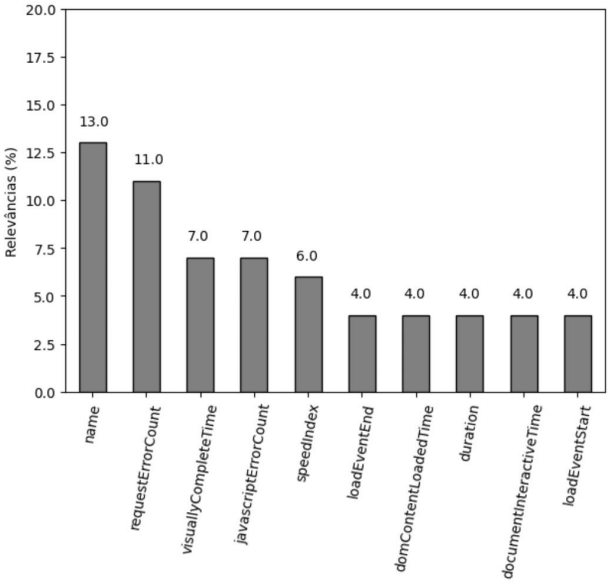

Figura 9. Resultado do Experimento 4 com RFE. Fonte: Autor.

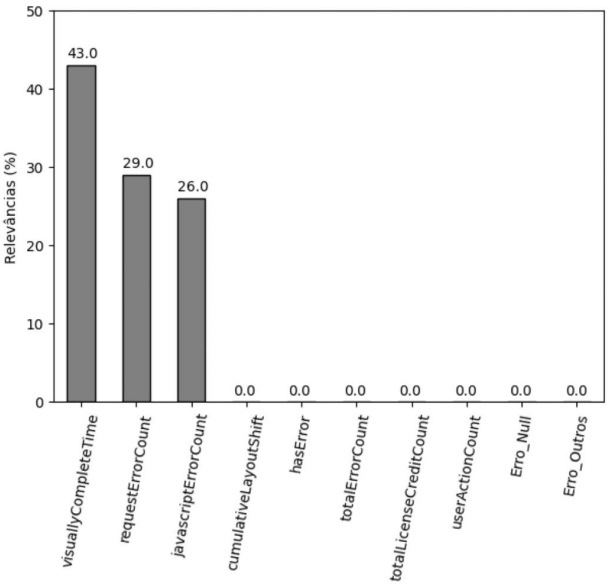

Figura 10. Resultado do Experimento 5 com Decision Tree. Fonte: Autor.

Para esta técnica, o tempo de resposta é o fator crítico. Além disso, o modelo identificou que apenas três atributos são necessários para classificar a experiência do usuário: o tempo de resposta e os erros de requisição e de JavaScript.

2) Random Forest

O segundo modelo da abordagem *embedded* foi o *Random Forest*. Seu tempo de treinamento durou cerca de 2 minutos e a figura 11 apresenta o resultado obtido

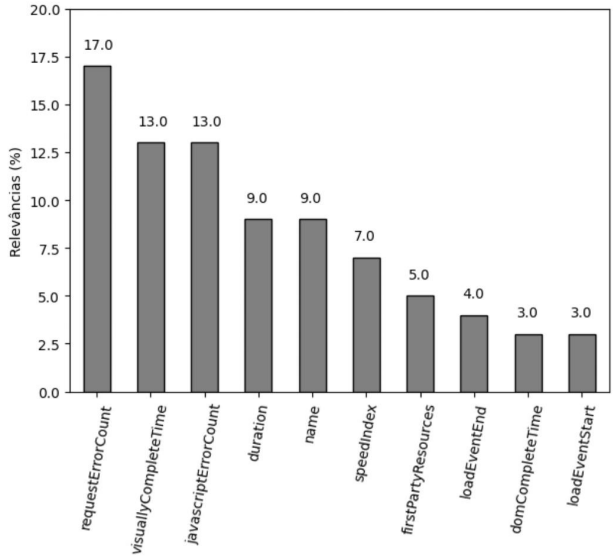

Figura 11. Resultado do Experimento 6 com Random Forest: Fonte: Autor.

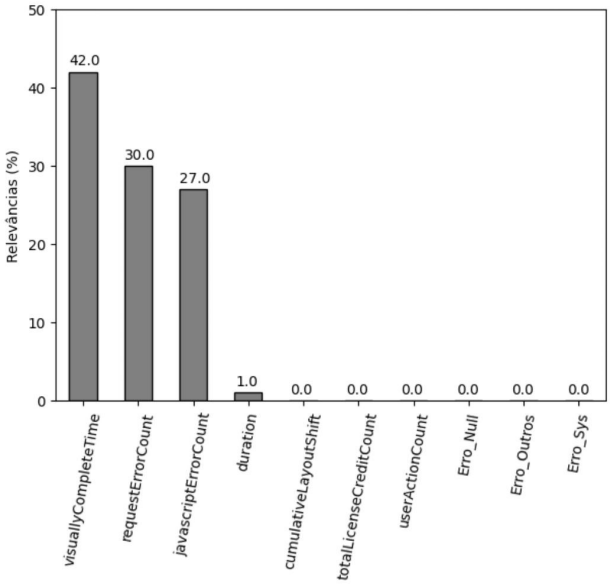

Figura 12. Resultado do Experimento 7 com *Gradient Boosting*. Fonte: Autor.

Pela primeira vez, com 17% de relevância, o atributo *requestErrorCount* aparece como sendo o atributos mais relevante, seguido pelos *visuallyCompleteTime* e *javascriptErrorCount*, ambos com 13% de relevância e entre os 56 atributos presentes no dataset 39 ficaram com o percentual abaixo de 0,1%.

Para o modelo, o fator principal na determinação da experiência do usuário são os erros de requisição, seguidos de perto pelo tempo de resposta.

3) Gradient Boosting

Com o tempo de 19 minutos para treinamento o resultado final é exibido na figura 12. Mais uma vez, o atributo *visuallyCompleteTime* aparece em primeiro lugar com 43% de relevância, seguido pelos atributos *requestErrorCount* e *javascriptErrorCount* com 30% e 27% de relevância respectivamente. O resultado foi semelhante ao modelo *Decision Tree*, também foram identificados 51 atributos como irrelevantes.

D. Análise Comparativa das Técnicas

Após a execução dos experimentos, as relevâncias obtidas para cada atributo foram normalizadas e somadas, e por fim, um novo ranqueamento foi gerado a partir da relevância total dos atributos.

A tabela III lista as relevâncias de cada técnica e a relevância total, para os primeiros dez atributos do ranque. O atributo visuallyCompleteTime (19,32%) é o mais importante, seguido pelos atributos javascriptErrorCount (16,65%), requestErrorCount (15,47%) e name (5,14%). Além disso, trinta e três atributos ficaram abaixo de 0,1% de relevância.

Com a lista de atributos ranqueados pela relevância total, iniciou-se o processo final de escolha dos atributos, para essa nova etapa, o modelo *Random Forest* foi selecionado.

Novas bases de treinamento foram geradas. A base inicial contém o primeiro atributo da lista e a cada iteração o próximo atributo será adicionado. O desempenho do modelo *Random Forest* foi avaliado em cada dataset gerado e a figura 13 apresenta o gráfico com a variação das métricas para as 6 primeiras iterações.

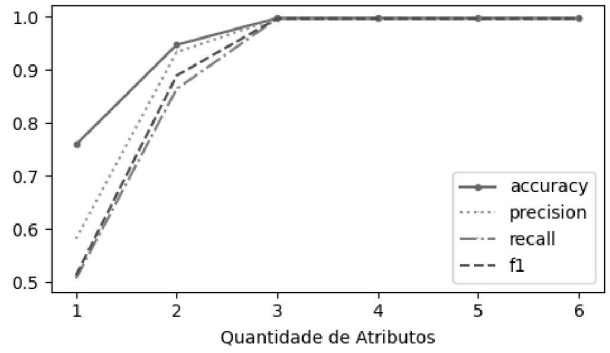

Figura 13. Desempenho do *RandomForest* para as bases geradas. Fonte: Autor.

Observa-se que a partir do terceiro atributo mais descritivo, o modelo apresentou um ótimo desempenho, sendo: **0,997375** de **acurácia**, **0,996516** de **precisão**, **0,996612** de **recall** e **0,996564** no **f1**, com uma leve melhora até o 5º atributo de onde se manteve constante ou com redução de desempenho. Para a seleção final dos atributos, este trabalho adotou os 4 primeiros atributos do ranque: *javascriptErrorCount*, *requestErrorCount*, *visuallyCompleteTime*, name, essa decisão foi tomada consultando o especialista do negócio, que identificou a relevância do atributo name, responsável por identificar o recurso que está sendo acessado pelo usuário do sistema.

Tabela III
ANÁLISE COMPARATIVA DAS TÉCNICAS

Atributos	Técnicas							Relevância Total
	ANOVA	Chi-Square	Mutual-Information	RFE	Decision Tree	Random Forest	Gradient Bossting	
visuallyCompleteTime	3%	2%	8%	7%	43%	13%	42%	19,32%
javascriptErrorCount	30%	22%	3%	7%	26%	13%	27%	16,65%
requestErrorCount	15%	3%	2%	11%	29%	17%	30%	15,47%
name	6%	2%	6%	13%	0%	9%	0%	5,14%
duration	2%	2%	7%	4%	0%	9%	1%	3,94%
speedIndex	2%	2%	7%	6%	0%	7%	0%	3,78%
hasError	5%	10%	1%	3%	0%	1%	0%	2,71%
loadEventEnd	2%	2%	5%	4%	0%	4%	0%	2,57%
totalErrorCount	6%	10%	1%	2%	0%	1%	0%	2,54%
loadEventStart	2%	2%	5%	4%	0%	3%	0%	2,41%

A tabela IV mostra o desempenho do modelo Random Forest com e sem a utilização de técnicas de seleção de atributos. O melhor desempenho foi obtido sem a utilização de técnicas de seleção de atributos. No entanto, entre as técnicas de seleção abordadas, o método proposto neste trabalho apresentou o melhor desempenho.

V. CONCLUSÃO

Este trabalho teve como objetivo identificar os fatores críticos que determinam a qualidade da experiência do usuário no sistema IDEA através da aplicação de técnicas avançadas de aprendizado de máquina para seleção de atributos. Inicialmente, foram realizados procedimentos que envolveram a coleta de dados de monitoramento, seu pré-processamento, a execução de experimentos utilizando 7 técnicas distintas de seleção de atributos, e, por fim, a análise dos resultados obtidos.

Um modelo composto por quatro módulos foi desenvolvido para ordenar as etapas e separar responsabilidades no processo de análise e processamento dos dados. Os resultados demonstraram a eficácia dessas técnicas ao reduzir a dimensionalidade dos dados de 56 para apenas 4 atributos mais descritivos, mantendo uma precisão na previsibilidade da classificação da experiência do usuário com métricas superiores a 98%.

Este estudo contribui significativamente ao destacar a viabilidade de identificar fatores críticos para a experiência do usuário através da análise de dados de monitoramento. Além disso, possibilita a verificação conjunta do resultado de variados métodos de seleção de atributos. Isso não apenas permite um direcionamento mais eficiente para a melhoria desses fatores no sistema IDEA, mas também ressalta a importância crucial do uso de diversas técnicas de aprendizado de máquina na tomada de decisões estratégicas.

No entanto, é importante reconhecer as limitações deste estudo, que se restringiu à análise de apenas 7 técnicas de seleção de atributos e a um período específico de monitoramento. Para estudos futuros, recomenda-se a expansão deste trabalho para incluir uma análise mais abrangente, explorando um conjunto mais diversificado de técnicas de aprendizado de máquina e estendendo o período de monitoramento para capturar interações de usuários ao longo de períodos mais longos, como 3 ou 6 meses.

Em suma, este trabalho não apenas oferece insights valiosos para a otimização da experiência do usuário no sistema IDEA, mas também abre novas possibilidades de pesquisa e desenvolvimento futuro no campo da análise de dados para melhorias contínuas na experiência do usuário em sistemas semelhantes.

REFERÊNCIAS

[1] M. Rosala and R. Krause, *User Experience Careers*. Nielsen Norman Group, 2nd ed. ed., 2020.

[2] J. J. Garrett, *The Elements of User Experience: User-Centered Design for the Web and Beyond*. New Riders, 2011.

[3] Internacional Organization for Standardization, Genebra, Suiça, *Ergonomics of human-system interaction. Part 11: Usability: Definitions and concepts.*, 2018. ISO.

[4] G.WANG and Y.WANG, *User satisfaction based dynamic priority assignment algorithm for internet of things*. IEEE, 2015.

[5] P. Sevcik, "The apdex users group." https://www.apdex.org/. Acessado em: 12-jul-2024.

[6] Z. Stadnik, W. e Nowak, "The impact of web pages' load time on the conversion rate of an e-commerce platform," *Springer, Cham*, 2018.

[7] H. D. Lee, "Seleção de atributos importantes para a extração de conhecimento de bases de dados," *https://doi.org/10.11606/D.55.2000.tde-15032002-113112*, 2000.

[8] B. Remeseiro and V. Bolon-Canedo, "A review of feature selection methods in medical applications," *Computers in Biology and Medicine vol. 112, p. 103375*, 2019.

[9] Z. Z. Salem Alelyani and H. Liu, "A dilemma in assessing stability of feature selection algorithms," *https://doi.org/10.1109/hpcc.2011.99*, 2011.

[10] N. Pudjihartono, T. Fadason, A. W. Kempa-Liehr, and J. M. O'Sullivan, "A review of feature selection methods for machine learning-based disease risk prediction," *Frontiers in Bioinformatics*, 2022.

[11] e. N. B. A. Jovic, K. Brkic, "A review of feature selection methods with applications," *Electronics and Microelectronics (MIPRO)*, 2015.

[12] M. Sebban and R. Nock, "A hybrid filter/wrapper approach of feature selection using information theory," *Pattern Recognition, vol. 35, no. 4, pp. 835–846*, 2002.

[13] F. J. Rui and A. Azevedo, "Seleção de atributos na previsão de insolvência: aplicação e avaliação usando dados brasileiros recentes," *RAM. Revista de Administração Mackenzie, vol. 15, no. 1, pp. 125–151*, 2014.

[14] M. B. e. B. P. Saroj Biswas, "Review on feature selection and classification using neurofuzzy approaches," *International Journal of Applied Evolutionary Computation*, 2016.

Tabela IV
DESEMPENHO DO MODELO RANDOM FOREST (K=4)

Abordagem	Técnicas de Seleção	Acurácia	Precisão	Recall	F1
-	Todos os Atributos	**0.9928332010231984**	**0.9930197845009662**	**0.9928332010231984**	**0.9928736866292552**
Filtro	ANOVA	0.9926347358207639	0.9928094083040079	0.9926347358207639	0.992673725784158
	Chi-Squared	0.8150304313310399	0.8089872853317023	0.8150304313310399	0.737838403373689
	Mutual-Information	0.8679985886918938	0.8643982596965424	0.8679985886918938	0.864636277195319
Wapper	RFE	0.9509349916203581	0.9507224678451435	0.9509349916203581	0.9478284043420763
Embedded	Decision Tree	0.9926347358207639	0.9928094083040079	0.9926347358207639	0.992673725784158
	Random Forest	0.9926347358207639	0.9928094083040079	0.9926347358207639	0.992673725784158
	Gradient Boosting	0.9925024256858075	0.9926875497980894	0.9925024256858075	0.9925434442345337
-	Método Propósto	**0.9926567875099233**	**0.9928316971316643**	**0.9926567875099233**	**0.9926957846576149**

[15] O. I. N. Omer Fadl Elssied and A. H. Osman, "A novel feature selection based on one-way anova f-test for e-mail spam classification," *Research Journal of Applied Sciences, Engineering and Technology, vol. 7, no. 3, pp. 625–638*, Jan. 2014.

[16] P. K. Anwla, "Introduction to anova for statistics and data science (with covid-19 case study using python)." https://www.analyticsvidhya.com/. Acessado em: 31-jul-2024.

[17] A. Chug, "Chi-square test in machine learning." https://www.geeksforgeeks.org/ml-chi-square-test-for-feature-selection/. Acessado em: 01-ago-2024.

[18] M. Ventura, "Teste qui-quadrado: Uma ferramenta essencial para análise de dados categóricos." https://medium.com/@matheusventura/teste-qui-quadrado-a95cb5ba8c66. Acessado em: 01-ago-2024.

[19] L. A.-H. Amani Abdo, Rasha Mostafa, "An optimized hybrid approach for feature selection based on chi-square and particle swarm optimization algorithms," *https://doi.org/10.3390/data9020020*, 2024.

[20] X. W. H. Zhou and R. Zhu, "Feature selection based on mutual information with correlation coefficient," *Applied Intelligence, vol. 52, no. 5, pp. 5457–5474*, 2021.

[21] K. J. M. Kuhn and Springer, "Science business media, applied predictive modeling," *New York: Springer*, 2016.

[22] R. N.-K. C. X.-S. Y. Douglas Rodrigues, Luís Pereira, "A wrapper approach for feature selection based on bat algorithm and optimum-path forest," *Expert Systems With Applications, vol. 41, no. 5, pp. 2250–2258*, 2014.

[23] Scikit-learn.org, "Feature selection." https://scikit-learn.org/stable/modules/feature$_s election.html$, 2019.

[24] Y.-L. K. Samo, "5 reasons why you should never use recursive feature elimination." https://blog.kxy.ai/why-you-should-stop-using-recursive-feature-elimination/index.html. Acessado em: 01-ago-2024.

[25] P. C. Madan Somvanshi, Shital Tambade, "A review of machine learning techniques using decision tree and support vector machine," *https://doi.org/10.1109/ICCUBEA.2016.7860040*, 2016.

[26] S. P. e. B. A. Arundhati Navada, Aamir Nizam Ansari, "Overview of use of decision tree algorithms in machine learning," *https://doi.org/10.1109/ICSGRC.2011.5991826*, 2011.

[27] Scikit-learn.org, "Decision trees." https://scikit-learn.org/stable/modules/tree.html, 2019.

[28] S. R. Marteletto, "Técnicas de seleção de atributos através de random forests: um estudo de caso para detecção de tendências em séries temporais financeiras," *https://doi.org/10.11606/d.100.2022.tde-08032023-134543*, 2023.

[29] G. M. Candice Bentéjac, Anna Csörgő, "A comparative analysis of gradient boosting algorithms," *https://doi.org/10.1007/s10462-020-09896-5*, 2021.

[30] F. Augusto, "Random forest multiclasse: a diagnostic study of mathematical learning difficulties," *https://doi.org/10.11606/d.59.2024.tde-08032024-073933*, 2024.

[31] Scikit-learn.org, "Random forest classifier." t.ly/URLt7, 2018.

[32] D. LLC, "Dynatrace documentation," 2024. Acessado em: 12-jul-2024.

Sistema FIEB | SENAI CIMATEC
PELO FUTURO DA INOVAÇÃO

CENTRO UNIVERSITÁRIO SENAI CIMATEC
ESPECIALIZAÇÃO EM DATA SCIENCE & ANALYTICS

ATA DE APRESENTAÇÃO DE PROJETO FINAL DE CURSO

Ata de apresentação do Projeto Final de Curso, "**Identificação de Fatores Críticos na Experiência do Usuário do IDEA: uma Abordagem para Seleção de Atributos"**, submetido pelo aluno **Nilmar Sousa Pereira**, como parte dos requisitos para obtenção do Certificado de **Especialista em Data Science & Analytics** pelo Centro Universitário SENAI CIMATEC, às 18h40 do dia 15 de Julho de 2024. Reuniu-se remotamente pela plataforma Google Meet, a Banca Examinadora designada pelo Prof Dr. Éldman de Oliveira Nunes – Orientador, constituída pelo Prof Dr. Éldman de Oliveira Nunes e Prof MSc Braian Varjão Gama Bispo. O Orientador deu início aos trabalhos com as devidas orientações, e a exposição foi realizada pelo estudante dentro do prazo de tempo estabelecido. Ao final da apresentação a banca reuniu-se atribuindo a seguinte nota: **9,4** (nove pontos e quatro décimos).

A banca de avaliadores decidiu pela:

(X) Aprovação do trabalho
Caberá ao aluno apresentar em no máximo em 30 (trinta) dias a contar da data de assinatura desta Ata, uma cópia do trabalho em PDF, constando as considerações pontuadas pela banca. A Ata de Apresentação do Projeto Final de Curso deve ser digitalizada e inserida na terceira página do TCC ou como anexo do artigo.

( ) Reprovação do trabalho
O aluno terá que se matricular novamente no TCC – Trabalho de Conclusão de Curso e ser submetido a uma banca avaliadora no semestre seguinte**.**

As ações consequentes ao status de Aprovação deverão obedecer ao prazo proposto acima sob pena do parecer final ser modificado para o status de Reprovado automaticamente e sem possibilidade de recurso.

Para constar, lavrou-se a presente ata que vai assinada por todos os membros da Banca. Por estarem cientes de suas obrigações estão de acordo com os termos desse documento:

Salvador, 15 de Julho de 2024.

gov.br
Documento assinado digitalmente
ELDMAN DE OLIVEIRA NUNES
Data: 16/07/2024 08:26:04-0300
Verifique em https://validar.iti.gov.br

Éldman de Oliveira Nunes
Professor Orientador

Electronically signed by:
Braian Varjão Gama Bispo
CPF: ***.291.145-**
Date: 7/24/2024 8:10:52 PM -03:00

Braian Varjão Gama Bispo
Membro da banca

Assinado eletronicamente por:
Patricia Freitas Tourinho
CPF: ***.733.265-**
Data: 19/07/2024 15:35:31 -03:00

Patricia Freitas Tourinho
Coordenadora de Pós-Graduação *Lato Sensu*