UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Valdi Ferreira do Nascimento Júnior

Mineração de dados de crowdsourcing para investigar o uso de energia em dispositivos Android

Recife

2022

Valdi Ferreira do Nascimento Júnior

Mineração de dados de crowdsourcing para investigar o uso de energia em dispositivos Android

Trabalho apresentado ao Programa de Pós-graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Área de Concentração: Engenharia de Software e Linguagens de Programação

Orientador (a): Prof. Dr. Fernando José Castor de Lima Filho

Recife

2022

Catalogação na fonte
Bibliotecária Nataly Soares Leite Moro, CRB4-1722

N244m Nascimento Júnior, Valdi Ferreira do
Mineração de dados de crowdsourcing para investigar o uso de energia em dispositivos Android / Valdi Ferreira do Nascimento Júnior. – 2022.
92 f.: il., fig., tab.

Orientador: Fernando José Castor de Lima Filho.
Dissertação (Mestrado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2022.
Inclui referências.

1. Engenharia de software e linguagens de programação. 2. Mineração de dados. 3. Dados de crowdsourcing. 4. Mobile. 5. Android. I. Lima Filho, Fernando José Castor de (orientador). II. Título

005.1 CDD (23. ed.) UFPE - CCEN 2022 – 99

Valdi Ferreira do Nascimento Júnior

“Mineração de dados de crowdsourcing para investigar o uso de energia em dispositivos Android”

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação. Área de Concentração: Engenharia de Software e Linguagens de Programação

Aprovado em: 11/03/2022.

BANCA EXAMINADORA

_____

Prof. Dr. Vinicius Cardoso Garcia
Centro de Informática/UFPE

_____

Prof. Dr. Ivan do Carmo Machado
Departamento de Ciência da Computação/UFBA

_____

Prof. Dr. Fernando José Castor de Lima Filho
Centro de Informática/UFPE
(**Orientador**)

Dedico este trabalho a minha amada família, amigos e a todos que me apoiaram nessa jornada.

AGRADECIMENTOS

Primeiramente, agradeço a minha família por me dar todo amor e apoio incondicional, não apenas durante este trabalho, mas por toda a minha vida. Ao meu pai, Valdi Ferreira, por ser meu exemplo de honestidade, minha referência de perseverança e meu herói. À minha mãe, Maria Auxiliadora, por ser a pessoa mais amorosa do mundo, por me dedicar tanto carinho e afeto, me inspirando a sempre ser uma versão melhor de mim. À minha irmã, Vanessa Maria, por ser minha amiga e parceira de tantos momentos, por estar comigo no meu crescimento e nos desafios da vida. Aos meus queridos amigos que me acompanham desde a graduação, Victor Nunes, Vitória Maciel, Pedro Clericuzi e Neto Barbosa, vocês são o melhor time que eu poderia fazer parte, e ao meu amigo Jorge Delgado por todas as conversas e conselhos valiosos para essa jornada. À todos os professores que já tive, desde o começo da minha vida acadêmica, desde a base, passando pela graduação e na pós-graduação no CIn-UFPE, até esse momento, pois todos foram responsáveis por eu chegar até aqui. Em especial, agradeço ao professor Fernando Castor, meu orientador, cujo o apoio e conhecimento só podem ser resumidos como fantásticos. Por fim, agradeço à cada um que me apoiou, me dedicou uma palavra de incentivo, ou simplesmente torceu por mim. Agradeço por poder ter tantos e tanto pra dividir nesse momento.

RESUMO

Um dos fatores essenciais para o sucesso de uma aplicação ou modelo de dispositivo é o consumo de energia, que impacta diretamente na autonomia e na satisfação de pessoas usuárias de dispositivos móveis e seus aplicativos. Em geral, os estudos relacionados à área são realizados em laboratório, por meio de simulações, com um número limitado de dispositivos e aplicações. Uma perspectiva mais ampla se faz necessária diante da vasta heterogeneidade do contexto móvel, em especial do Android. Este estudo explora o uso de dados de dispositivos móveis, coletados através de colaboração coletiva (crowndsourcing), modelando seu comportamento energético para entender os fatores que mais impactam o seu consumo de energia. Foi analisada a viabilidade de desenvolver modelos a partir de técnicas tradicionais de Mineração de Dados e Aprendizado de Máquina utilizando dados de uso real de dispositivos móveis em larga escala, de modo a relacionar os diversos fatores do contexto móvel com o tempo de decaimento das baterias. Foram estudados dados dos 100 modelos mais populares e suas configurações, juntamente com os aplicativos e processos mais populares, presentes no banco de dados GreenHub, uma iniciativa colaborativa e voluntária para coletar dados para estudos de dispositivos móveis Android. Os pontos de dados coletados correspondem ao estado de diversos aspectos dos aparelhos sempre que ocorre uma alteração no nível das baterias, sendo o intervalo de tempo entre as alterações de decaimento, o alvo das técnicas utilizadas nesse estudo. Foram utilizados algoritmos de regressão baseados em árvore (Decision Tree, Random Forest e XGBoost), em conjunto com a abordagem SHAP (SHapley Additive exPlanations) a fim de estabelecer a capacidade preditiva e descritiva das técnicas. Os resultados indicam um grau de dificuldade elevado ao estudar as relações contidas nos dados, pois a precisão das predições varia bastante de acordo com o modelo de dispositivo a ser estudado, refletindo a heterogeneidade do contexto. Por exemplo, para o modelo mais popular da base, SM-G532M, a precisão da melhor predição, usando a técnica Decision Tree, foi de 37%. Por outro lado, para o segundo modelo mais popular, o A0001, a melhor precisão obtida, usando a técnica XGBoost, foi de 68%. Resultados como estes mostram que experimentos de laboratório para avaliar consumo de energia de dispositivos móveis têm poder limitado de representar situações do mundo real, dada a grande variabilidade de contextos em que estes podem ser usados. Dentre os aspectos mais impactantes para os modelos, fatores como temperatura, voltagem, uso da CPU e conexões de rede foram considerados mais relevantes que, por exemplo, os processos e aplicativos que estão em execução em dado momento.

Palavras-chaves: mineração de dados; dados de crowdsourcing; energia; mobile; Android.

ABSTRACT

One of the essential factors for the success of an application or device model is energy consumption, which directly impacts the autonomy and satisfaction of people who use mobile devices and their applications. In general, studies related to the area are carried out in the laboratory, through simulations, with a limited number of devices and applications. A broader perspective is needed given the vast heterogeneity of the mobile context, especially Android. This study explores the use of data from mobile devices, collected through crowdsourcing, modeling their energy behavior to understand the factors that most impact their energy consumption. The feasibility of developing models based on traditional Data Mining and Machine Learning techniques using data from real use of mobile devices on a large scale was analyzed, in order to relate the various factors of the mobile context with the battery decay time. Data from the 100 most popular models and their configurations were studied, along with the most popular applications and processes, present in the GreenHub database, a collaborative and voluntary initiative to collect data for Android mobile device studies. The data points collected correspond to the state of several aspects of the devices whenever there is a change in the battery level, with the time interval between the decay changes being the target of the techniques used in this study. Tree-based regression algorithms (Decision Tree, Random Forest and XGBoost) were used, together with the SHAP approach (SHapley Additive exPlanations) in order to establish the predictive and descriptive capacity of the techniques. The results indicate a high degree of difficulty when studying the relationships contained in the data, as the accuracy of the predictions varies greatly according to the device model to be studied, reflecting the heterogeneity of the context. For example, for the most popular model in the dataset, SM-G532M, the accuracy of the best prediction, using the Decision Tree technique, was 37%. On the other hand, for the second most popular model, the A0001, the best accuracy obtained using the XGBoost technique was 68%. Results like these show that laboratory experiments to assess energy consumption of mobile devices have limited power to represent real-world situations, given the wide variability of contexts in which they can be used. Among the most impacting aspects for the models, factors such as temperature, voltage, CPU usage and network connections were considered more relevant than, for example, the processes and applications that are running at any given time.

Keywords: data mining; crowdsourcing data; energy; mobile; Android.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Cada vez mais pessoas do mundo todo estão utilizando funcionalidades e consumindo serviços através de aplicativos em dispositivos móveis. O aumento significativo do acesso das pessoas a esta tecnologia, bem como o aumento da oferta e disponibilidade dos serviços de comunicação, fez com que o número de usuários ativos e conectados aumentasse nos últimos anos. De maneira conveniente e eficiente, os dispositivos e seus aplicativos atendem às necessidades de seus usuários independente de local e hora (DINH et al., 2013).

A qualidade é um fator essencial para que um aplicativo tenha sucesso nesse mercado cada vez mais competitivo. Os aplicativos devem explorar características específicas dessa tecnologia (ex. mobilidade, geolocalização, variedade de interação com o usuário) a fim de entregar serviços que atendam, e até mesmo superem, a expectativa dos usuários, ao mesmo tempo em que precisam lidar com fatores particulares ao ambiente como limites de capacidade de processamento e armazenamento, diversidade de protocolos e meios de acesso à Internet, variedade de plataformas e dispositivos, segurança e etc. (CORRAL; SILLITTI; SUCCI, 2015).

Um desses fatores é o consumo de energia, limitada pelo tempo de vida da bateria, que está fortemente ligado à usabilidade, e aos níveis de satisfação dos usuários. Essa afirmação foi feita no estudo que analisou cerca de 36 mil comentários relacionados a pelo menos 27 mil aplicativos da Google Play[1] mostrou que 18% destes possuíam comentários relacionados ao consumo energético. O estudo revelou que problemas relacionados à eficiência energética impactavam negativamente suas avaliações, igualmente entre aplicativos grátis e pagos, mostrando a relevância da eficiência energética para o sucesso de um aplicativo e a necessidade geral de soluções e de pesquisas para o problema (WILKE et al., 2013).

Ainda segundo Wilke et al. (2013), os usuários relacionam o alto consumo de energia à fatores como atividades em *background*, comportamento ineficiente do GPS, uso desnecessário da CPU, má utilização da memória RAM, problemas de sincronização, entre outros, indicando o quanto os usuários estão atentos ás relações de consumo de energia em seus aparelhos.

As relações apontadas pelos usuários se fazem presentes entre os desenvolvedores. O estudo realizado por Pinto, Castor e Liu (2014) analisou mais de 300 questões no Stack Overflow[2] relacionadas a consumo e eficiência energética. Essas questões contavam com interações de mais de 800 usuários e, dentre outras descobertas interessantes, apontou como causas de

[1] https://play.google.com/store/apps
[2] https://stackoverflow.com/

ineficiência energética basicamente as questões levantadas pelos usuários (o uso desnecessário de recursos, como memória e CPU, atividades em *background*, mau comportamento do GPS e sincronizações excessivas), reforçando ainda mais a relevância e a atenção que os aspectos envolvendo a duração do tempo de uso das baterias recebem.

Pramanik et al. (2019) destacaram que os smartphones tiveram um aumento significativo na sua capacidade ao longo dos anos, combinando vários componentes (processadores, processadores gráficos, moldem, etc.) de alto desempenho em um único chip e uma maior oferta de capacidade de armazenamento e memória. Além desses recursos de computação, os dispositivos apresentam sistemas operacionais mais eficientes e com mais funcionalidades, telas maiores e com mais resolução, múltiplas interfaces de conexão com suporte a operações diferentes comunicações sem fio simultaneamente, bem como diversos sensores internos e externos.

As pesquisas que buscam uma melhor capacidade e eficiência das baterias não estão acompanhando a demanda dos dispositivos, tornando a disponibilidade de energia nas baterias um recurso cada vez mais escasso e limitado. Pramanik, Pal e Choudhury (2019) também apontaram que a tendência é que essa lacuna aumente capacidade e avanços aumente, o que torna importante entender o padrões de consumo de energia de dispositivos otimizando o uso desse recurso fundamental. Por fim, os autores indicaram diversas razões para drenagem de energia nos dispositivos móveis, incluindo:

- Aplicativos em *background*
- Atualizações frequentes
- *Streaming* de música e vídeo
- Backup e sincronização automática de dados
- Notificações frequentes
- Uso de *widgets* [3] e outros elementos de interface

Além disso, Pramanik et al. (2019) citaram questões energéticas relacionadas ao uso da CPU, GPU, GPS, memória, tela, e todos os tipos de conexão, indicando que os problemas de consumo de energia no contexto móvel continuam basicamente os mesmos, se comparados aos

[3] Elementos de uma interface gráfica do usuário que exibe informações ou fornece uma maneira específica para um usuário interagir com um serviço através do sistema operacional ou aplicativo

levantamentos anteriores. Questões essas que, segundo a pesquisa mais recente, tendem a se agravar pois o desenvolvimento das baterias não acompanha o ritmo da demanda energética das demais tecnologias e serviços do contexto móvel.

1.1 MOTIVAÇÃO

Estudar essas relações é um desafio, dada as características do contexto fragmentado em que elas se encontram, em especial no Android. Segundo a Statcounter[4], é o sistema operacional móvel mais popular do mundo, apresentando diversas versões espalhadas por milhares de modelos. Usados para as mais diversas finalidades através de milhões de aplicativos e suas funcionalidades, por usuários com hábitos e culturas completamente diferentes.

Na busca por compreender melhor esse contexto vasto e heterogêneo, surgiu o GreenHub[5] uma base colaborativa de dados de uso real de dispositivos Android, anônima e em larga escala, capaz de registrar eventos a cada mudança no nível da bateria, diversos aspectos de funcionamento e uso do dispositivo. A base possui informações sobre sensores ativos, uso da memória e processador, temperatura e voltagem da bateria, processos do sistema, aplicativos, configurações de rede, modelos e fabricantes. (MATALONGA et al., 2019).

Os dados do GreenHub representam um passo importante para essa área de estudo, pela quantidade de dados, representatividade e pelas diversas informações relacionadas oferecendo oportunidade para aplicação de diferentes técnicas para organização, agregação e análise das informações. Os registros foram coletados a cada mudança de nível de bateria, junto com informações sobre a hora e data que aconteceram, o que possibilita estipular a duração entre os registros, sendo esse o tempo que as baterias levam pra descarregar em 1%, e analisar a influência das configurações, processos e aplicativos nestes intervalos.

1.2 OBJETIVOS

Este estudo exploratório tem como principal objetivo investigar a viabilidade de desenvolvimento de modelos a partir de técnicas tradicionais de Mineração de Dados e Aprendizado de Máquina para estudar as relações das características e processos dos dispositivos com o tempo no decaimento dos níveis de bateria em dispositivos Android, numa base de dados

[4] https://gs.statcounter.com/os-market-share/mobile/worldwide
[5] https://greenhubproject.org/

em larga escala. A avaliação da capacidade preditiva foi feita utilizando as métricas MAPE (Mean Absolute Percentage Error) e MAE (Mean Aboslute Error), enquanto a aplicação da abordagem SHAP, usando o método TreeExplainer, foi usada para avaliação descritiva. Os resultados foram estudados sob a perspectiva dos modelos de dispositivos mais populares da base, quantificando o impacto das configurações, dos processos e dos aplicativos nos modelos de dispositivo para estabelecer a viabilidade da abordagem escolhida. O objetivo deste trabalho é composto pela execução dos objetivos específicos:

- Investigar a capacidade preditiva dessas técnicas tradicionais entre os modelos de dispositivos
- Investigar a capacidade descritiva dos modelos de aprendizagem desenvolvidos
- Analisar o impacto das configurações dos dispositivos móveis no decaimento das baterias
- Analisar o impacto dos processos e aplicativos no decaimento das baterias

1.3 ESTRUTURA DA DISSERTAÇÃO

Este estudo está organizado em 5 capítulos. O Capítulo 2 contém o referencial teórico dos principais componentes deste trabalho e os trabalhos com abordagens semelhantes à proposta deste estudo. O Capítulo 3 descreve a aplicação da metodologia descrita na exploração dos dados para a descoberta do conhecimento e os trabalhos relacionados a esse tipo de modelagem. O Capítulo 4 apresenta a síntese e discussão dos principais resultados, sendo o Capítulo 5 designado para conclusão, ameaças a validade e apontamentos de trabalhos futuros.

2 FUNDAMENTAÇÃO

Este capítulo apresenta as bases e conceitos utilizados na realização deste trabalho. A seção GreenHub (Seção 2.1) apresenta o objeto de estudo deste trabalho, a base de dados do projeto GreenHub, seus principais componentes e características. Na seção Mineração de Repositório de Software (Seção 2.2) são abordados os principais conceitos desta área da mineração de dados e a metodologia usada no estudo. Na seção Aprendizado de Máquina (Seção 2.3) são feitas as definições das abordagens, termos, além da descrição das técnicas selecionadas. Ainda nesta seção, são discutidos elementos de interpretação dos resultados e como eles se encaixam no ciclo da descoberta do conhecimento. Por fim, na seção TreeExplainer (Seção 2.4), é apresentada a técnica utilizada para interpretação dos modelos, responsável por medir o impacto das variáveis independentes nas predições realizadas

2.1 GREENHUB PROJECT

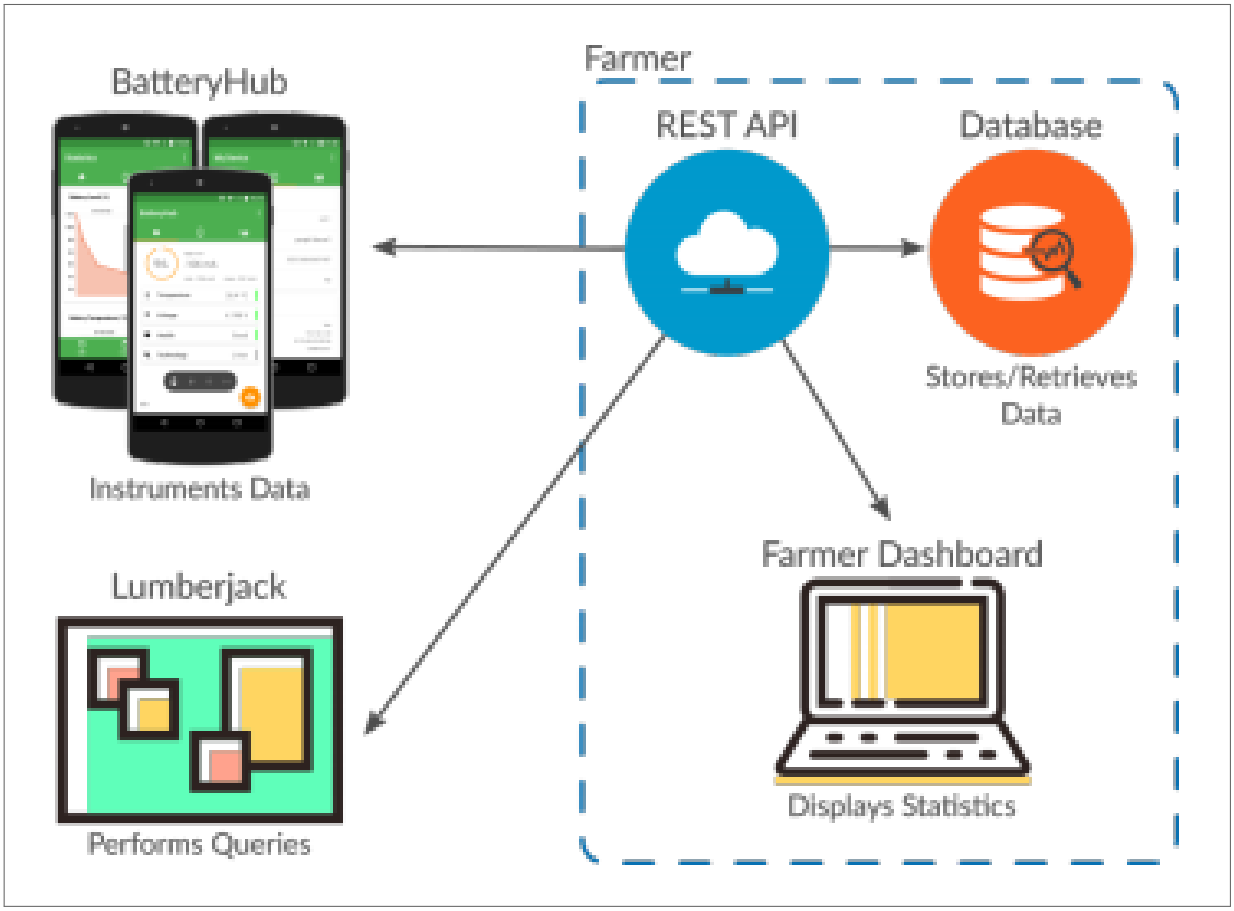
Visando dar suporte a novos processos e descobertas relacionadas ao consumo energético, surgiu o GreenHub[1], uma iniciativa para construção, através de colaboração coletiva (*crowd-sourcing*), de uma base de dados em larga escala obtidos por meio de um aplicativo móvel de coleta e processamento de dados relacionados a energia. Essa base é composta por dados relacionados ao uso de dispositivos móveis, seus sistemas, aplicativos e configurações, em uso real do dia-a-dia, para que sejam identificadas oportunidades para desenvolvimento de soluções e técnicas que melhorem o uso dos recursos energéticos dos dispositivos (MATALONGA et al., 2019).

A iniciativa multiplataforma é composta por três componentes: GreenHub BatteryHub, Farmer e o Lumberjack, conforme a Figura 1. O primeiro é o aplicativo para Android, disponível na Google Play Store[2], que monitora o sistema e identifica mudanças no nível da bateria, coletando uma amostra de diversos estados do dispositivo e do sistema. Essa amostra contém diversas informações do sistema como nível, temperatura e voltagem da bateria, uso de processamento e memória, estados dos componentes de rede, configurações habilitadas, processos entre outras. Todos os dados são coletados anonimamente, não sendo possível identificar nenhum usuário, ou associar qualquer um com os dados. O aplicativo oferece diversas funções

[1] https://greenhubproject.org/
[2] https://play.google.com/store/apps/details?id=com.hmatalonga.greenhub&hl=pt_BR&gl=US

para gerenciamento de energia, com gráficos interativos relacionados a diversos aspectos do uso do dispositivo, permitindo configurações de alerta baseado em temperatura, tendo como objetivo oferecer sugestões baseadas em perfis nas próximas versões.

Figura 1 – Arquitetura da Plataforma GreenHub



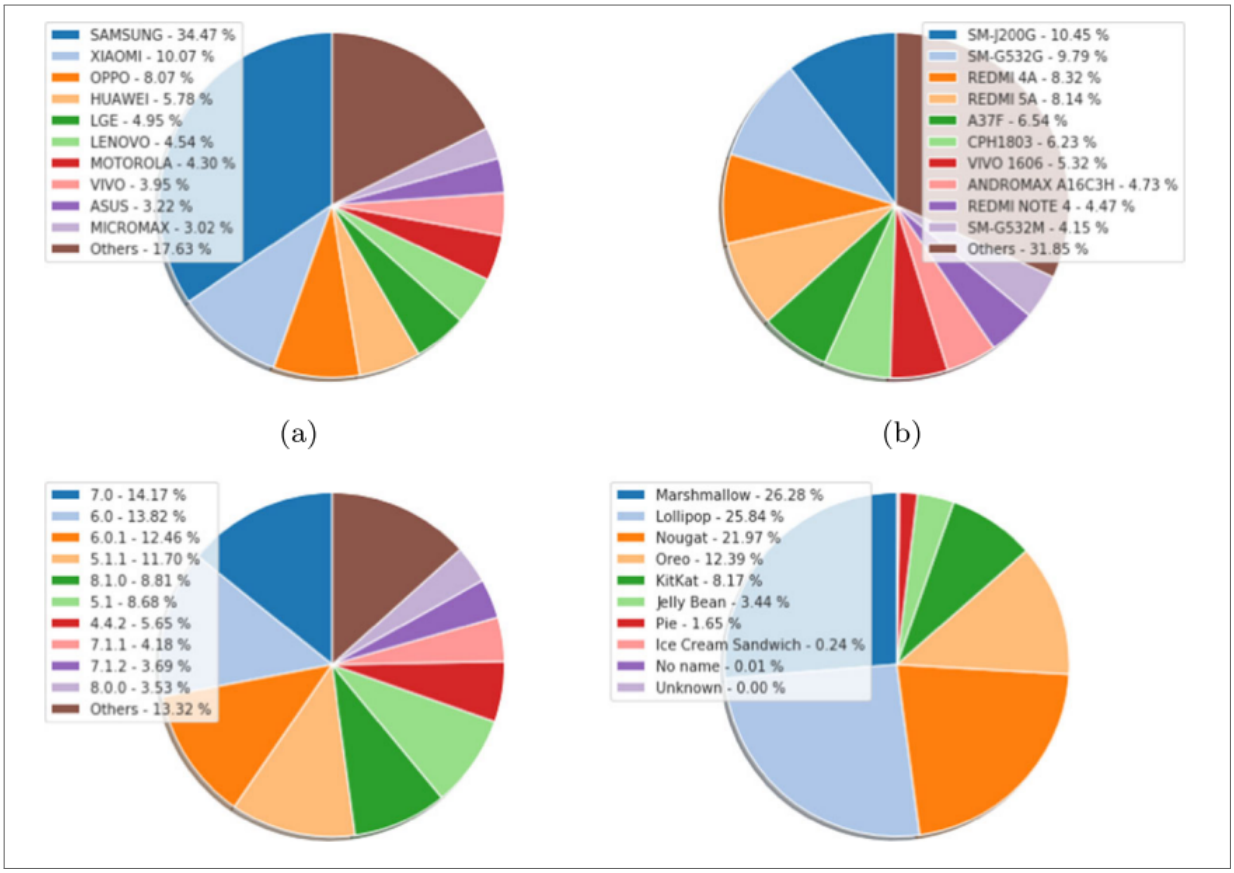
Fonte: (MATALONGA et al., 2019)

O Farmer é um aplicação servidora em PHP, responsável por armazenar os dados resultado das coleta apoiada por todos os usuários do GreenHub BatteryHub. As amostras são recebidas em JSON e mapeadas para um banco de dados relacional MariaDB. Periodicamente, o serviço gera uma versão completa do repositório, composta de arquivos CSV, contendo todos os dados da base. Além dessas funções, tem como objetivo oferecer uma interface com métricas, informações estatísticas e gráficos relacionados a todos os dados da base, além de uma API para acesso aos dados, também usada para comunicação entre dispositivos, através do GreenHub BatteryHub, e o servidor Farmer. Essa mesma API é usada pelo último componente da plataforma, o Lumberjack, uma interface de linha de comando que permite consultas flexíveis, sob demanda, aos dados da base.

Segundo Pereira et al. (2021), a base atualmente é composta por amostras relacionadas a

mais de 11.600 modelos, distribuídos em mais de 1600 fabricantes, rodando com pelo menos 50 versões diferentes de Android. São mais de 700 milhões de processos registrados associados a mais de 23 milhões de amostras de eventos energéticos em dispositivos móveis oriundos de mais de 160 países, formada por uma comunidade de mais de 87 mil dispositivos com o GreenHub BatteryHub instalado. Conforme mostrado na Figura 2 esses dados refletem o real contexto fragmentado do sistema da Google, bem como da grande variedade de dispositivos e diversas das suas características.

Figura 2 – Distribuição dos dados dos dispositivos do GreenHub

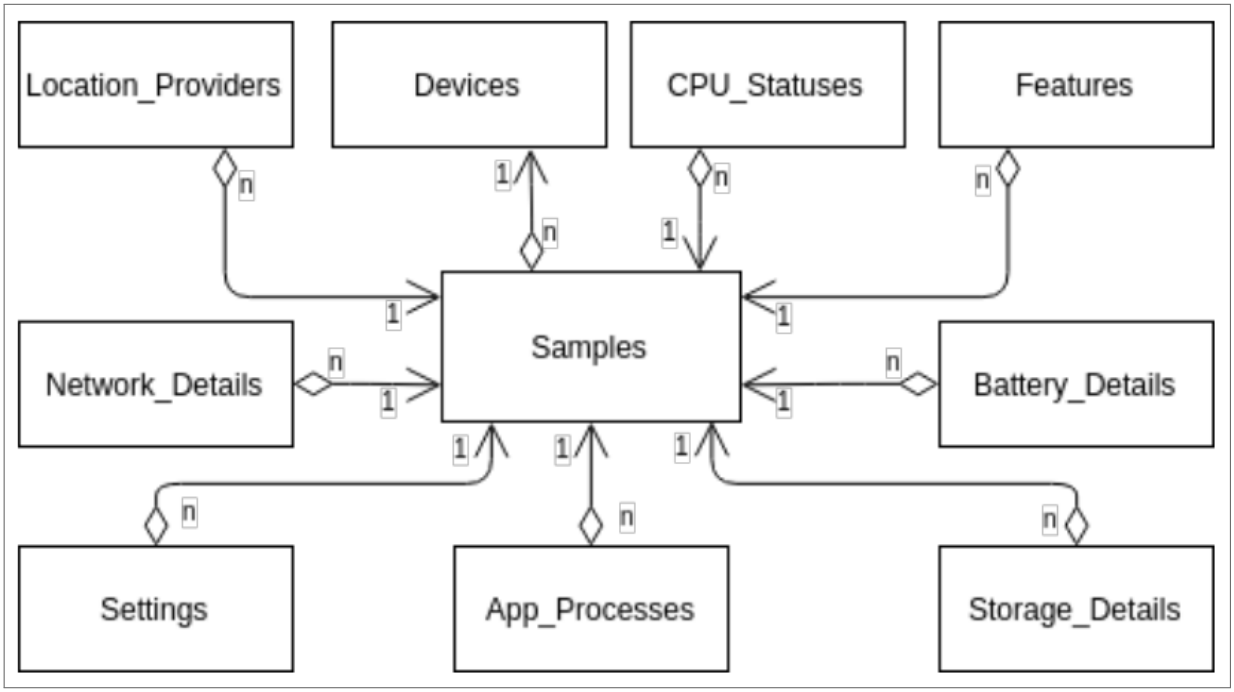


Fonte: (PEREIRA et al., 2021)

O estudo indica que mais de 50 versões do Android estão presentes nos dados, e que nesses ambientes mais de 740 milhões de processos foram executados, com uma taxa média de 31 processos registrados, em execução, para cada samples. Isso demonstram não apenas o vasta a quantidade de dados contidas na base, mas também as inúmeras características que capturadas do contexto Android. Os dados do são organizados em função dos samples (amostras), que agregam diversas informações do dispositivo, relacionadas conforme a Figura 3. Elas são coletadas pelo aplicativo no momento em que ocorre uma mudança no nível de bateria, geralmente em 1%, a menor granularidade detectável de mudança do nível da bateria,

podendo esta ser essa mudança negativa ou positiva. Para isso, utiliza informações de saídas de classes do sistema Android relacionadas a status de configuração além de processar arquivos de logs do kernel.

Figura 3 – Diagrama entidade relacionamento do GreenHub



Fonte: (MATALONGA et al., 2019)

A versão do banco de dados utilizada neste estudo está organizada em três tipos de arquivos. Os arquivos de samples são organizados conforme as Tabelas 1, 2, 3 e 4, agregando informações sobre configurações, sensores e estado do hardware, os de dispositivos, que conforme a Tabela 5 possui informações que caracterizam o dispositivo como modelo, fabricante e versão do sistema, e os processos, que informa quais aplicativos e processos do sistema estavam em execução naquele momento, conforme a Tabela 6.

O estudo também realizou uma análise para entender os impactos e tendências na bateria carga/descarga com base no uso do mundo real, considerando vários fatores inerentes de tal uso, explorando os dados para responder algumas questões sobre consumo de energia e destacando tendências do uso de energia sob diversas perspectivas. Para isso estabelece a métrica PPM (Percentage Per Minute) que identifica sequências de samples de bateria pertencentes ao mesmo dispositivo móvel que sigam na mesma direção (carregando ou descarregando), sendo a PPM é definida como a taxa de carga/descarga de um determinado período com base na quantidade de tempo dentro desse período e quanto o nível de bateria foi alterado. As relações entre as diversas características do contexto Android e as PPMs, evidenciadas

Tabela 1 – Detalhes dos dados de Sample do GreenHub - Gerais e da Bateria

Atributo	Tipo	Exemplo
Gerais		
timestamp	timestamp	2017-10-08 13:44:15
battery_state	varchar	"Discharging"
battery_level	decimal	82.00
timezone	vachar	"America/Chicago"
country_code	varchar	"us"
Bateria		
charger	varchar	"unplugged"
health	varchar	"Good"
voltage	decimal	04.05
temperature	decimal	28.50

Fonte: (MATALONGA et al., 2019)

Tabela 2 – Detalhes dos dados dos samples do GreenHub - CPU e configurações

Atributo	Tipo	Exemplo
CPU		
usage	decimal	0.03
up_time	int	408687
sleep_time	int	141369
Configurações		
screen_on	int	1
screen_brightness	int	61
roaming_enabled	int	0
bluetooth_enabled	int	0
location_enabled	int	1
power_saver_enabled	int	0
nfc_enabled	int	0
developer_mode	int	0

Fonte: (MATALONGA et al., 2019)

Tabela 3 – Detalhes dos dados dos samples do GreenHub - Memória e Armazenamento

Atributo	Tipo	Exemplo
Memória		
memory_active	int	49640
memory_inactive	int	515056
memory_free	int	1442060
memory_user	int	58688
Armazenamento		
free	int	3921
total	int	9634
free_system	int	637
total_system	int	3390

Fonte: (MATALONGA et al., 2019)

Tabela 4 – Detalhes dos dados de Sample do GreenHub - Conexões

Atributo	Tipo	Exemplo
Conexão		
network_status	varchar	"lte"
network_type	varchar	"MOBILE"
mobile_network_type	varchar	"lte"
mobile_data_status	varchar	"connected"
mobile_data_activity	varchar	"none"
wifi_status	varchar	"enabled"
wifi_signal_strength	varchar	-127
wifi_link_speed	int	-1

Fonte: (MATALONGA et al., 2019)

Tabela 5 – Detalhes dos dispositivos do GreenHub

Atributo	Tipo	Exemplo
id	int	1
model	varchar	"Nexus 5"
manufacturer	varchar	"LGE"
brand	varchar	"google"
os_version	varchar	"6.0.1"
is_root	int	0
created_at	timestamp	2017-10-09 09:04:00

Fonte: (MATALONGA et al., 2019)

Tabela 6 – Detalhes dos Processos do GreenHub

Atributo	Tipo	Exemplo
id	int	111
sample_id	int	3
name	varchar	"com.facebook.orca"
application_label	varchar	"Messenger"
is_system_app	int	0
version_name	varchar	"138.0.0.20.92"

Fonte: (MATALONGA et al., 2019)

através das análises qualitativas e quantitativas feitas por Pereira et al. (2021), demonstraram que o uso energético sofre influência de diversos fatores contidos nos dados, como as diferentes tendências de carga/descarga entre países, entre versões do sistema Android, entre aplicações mais populares, e etc. Durante as análises, muitas novas perguntas e caminhos de pesquisa (com foco no consumo de bateria de smartphones Android) surgiram, destacando a necessidade de novos estudos e as oportunidades que o GreenHub representa para essa área de estudo.

Conforme apontado por Matalonga et al. (2019), o banco de dados do GreenHub representa uma oportunidade para que estudos sobre consumo energético avancem sobre desafios encontrados na área, como a dificuldade em cobrir diversos cenários de uso real, nos quais o uso dos dispositivos pode estar inserido, levando em consideração a grande fragmentação do contexto relacionados aos sistemas Android. Nesta pesquisa os dados do GreenHub serão organizados e analisados pela perspectiva do intervalo de tempo de consumo de 1% de bateria, utilizando as metodologias e conceitos apresentados a seguir. Para isso, serão observados o impacto das variáveis no tempo de intervalo entre dois níveis de decaimento.

2.2 MINERAÇÃO DE DADOS EM REPOSITÓRIOS DE SOFTWARES

Projetos de software são grandes produtores de dados durante toda sua existência. De acordo com Hassan (2008) a necessidade de armazenar esses dados por diversas razões (backup, versionamento, rastreabilidade, testes e etc) tornaram os repositórios de software uma fonte valiosa para análises e descoberta de conhecimento. Esses repositórios podem ter informações referentes ao progresso do projeto, logs de execução, códigos, relatos de bugs, avaliações de usuários, entre outras, e o objetivo de minerar essas bases de dados é ampliar a

percepção sobre aspectos conhecidos e identificar novos que auxiliem em todas as etapas do processo, que muitas vezes não levam em consideração análises e medições mais profundas de dados armazenados em repositórios.

Na busca de geração de conhecimento sobre comportamento das aplicações e o consumo de energia, os repositórios de tempo de execução são fundamentais. Bases assim, como o GreenHub, podem ser usadas em busca de conhecimento relacionados a diversos fatores, identificando padrões na execução de sistemas de software e no funcionamento de dispositivos, que podem ser estudos em busca de novos conhecimentos. Além disso, o estudo de desvios desses padrões também costumam revelar oportunidades de desenvolvimento, auxiliar nas tomadas de decisão, ou identificar anomalias de execução.

Segundo Chaturvedi, Sing e Singh (2013), a exploração desses dados utiliza diversas ferramentas, desde aquelas de uso comum de mineração de dados, até o desenvolvimento de novas, incluindo protótipos e scripts específicos para o objetivo pretendido. Dentre as atividades reportadas neste estudo, a coleta, o preprocessamento, e pós processamento, são tidas como as mais importantes, podendo consumir muito tempo, devido a dificuldades de acesso e adequação do formato dos dados tanto para exploração, como para demais etapas e apresentação dos resultados, o que evidencia a importância da qualidade dos dados, facilitando e melhorando os resultados de todo o processo de descoberta de conhecimento.

A metodologia utilizada neste estudo de mineração de dados segue as bases do ciclo KDD (*Knowledge Discovery in Databases*), um processo interativo e iterativo (o KDD pode envolver significativos loops entre quaisquer dos passos descritos), com várias etapas que envolvem diversas tomadas de decisão, desde a compreensão do domínio, passando por todas as etapas de manipulação dos dados, e entrega de resultados (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996a).

O KDD é visto como uma metodologia fundamental na área de mineração de dados e ainda é bastante utilizado, servindo de base para metodologias mais recentes como SEMMA e CRISP-DM, que são vistas como implementações do KDD para contextos mais específicos (AZEVEDO; SANTOS, 2008). O SEMMA foi criado pela SAS e é um acronômo para *Sample, Explore, Modify, Model, and Assess* sendo usado como base para suas ferramentas de mineração, podendo servir como base par construção de outras aplicações de mineração (MATIGNON, 2007). O CRISP-DM foi desenvolvido por meio do esforço de um consórcio composto inicialmente por DaimlerChryrler, SPSS e NCR. CRISP-DM significa *CRoss-Industry Standard Process for Data Mining* e suas etapas possuem ênfase em processos industriais, voltadas para negócios (WIRTH;

Tabela 7 – Resumo das principais metodologias de mineração de dados

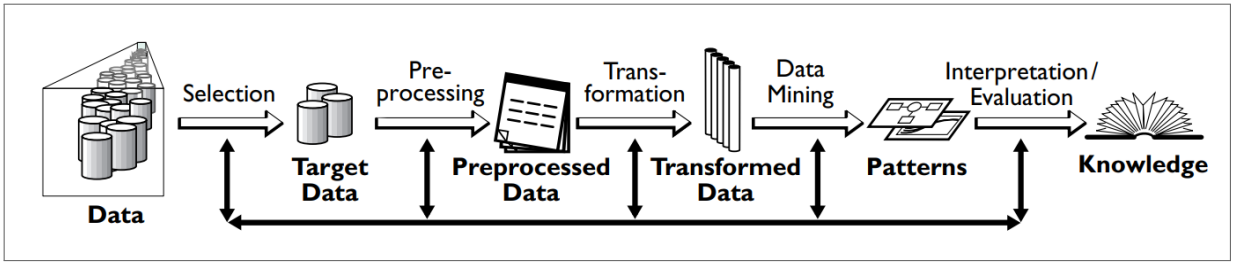
KDD	CRISP-DM	SEMMA
Compreensão do domínio da aplicação	Compreensão do negócio	———
Criação do conjunto alvo Limpeza de dados e pré-processamento	Compreensão dos dados	Amostragem Exploração
Redução dos dados e projeção	Preparação dos dados	Modificação
Escolha do método de mineração Análise e modelagem exploratória Mineração	Modelagem	Modelagem
Interpretação dos padrões minerados	Avaliação	Avaliação
Ação sobre conhecimento descoberto	Desenvolvimento	———

Fonte: (MATALONGA et al., 2019)

HIPP, 2000).

Segundo Shafique e Qaiser (2014), o KDD é visto como mais completo por apresentar passos bem determinados e precisos para cada fase do processo de descoberta do conhecimento, capazes de serem generalizados para diversas aplicações. O estudo realizado por eles faz um comparativo entre as três metodologias, comparando as etapas de cada uma conforme a Tabela 7. Sendo assim, a metodologia escolhida para este trabalho é o KDD, cujo ciclo é composto por 9 passos, está representado na Figura 4. São eles:

Figura 4 – Ciclo do processo KDD



Fonte: (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996b)

1. Compreensão do domínio da aplicação: levantamento de informações relevantes e identificação do objetivo do processo KDD.

2. Criação do conjunto alvo: Selecionar um conjunto de dados, ou focar em um subconjunto de variáveis ou amostras de dados, relevante para os objetivos, para as etapas seguintes.

3. Limpeza de dados e pré-processamento. Operações de coleta de informações para modelagem ou remoção de ruídos, tomadas de decisões sobre dados ausentes e mudanças

conhecidas.

4. Redução dos dados e projeção: Encontrar os dados mais representativos para o objetivo da atividade, sendo assim podem ser encontrados conjuntos menores, ou invariáveis, melhor represente os dados.

5. Escolha do método de mineração: Escolher um método de mineração de acordo com objetivo estabelecido na fase 1 do projeto. Sumarização, clustering, regressão, entre outros.

6. Análise e modelagem exploratória: Escolha do algoritmos e do métodos de mineração para serem usados no processo. Esta fase inclui a escolha dos modelos e os parâmetros podem ser apropriados combinando um método de mineração de dados específico com os critérios gerais do processo KDD.

7. Mineração: busca de padrões de interesse em uma forma representacional particular ou um conjunto de tais representações (incluindo regras de classificação ou árvores, regressão e agrupamento).

8. Interpretação dos padrões minerados: Esta etapa também pode envolver visualização dos padrões extraídos e modelos ou visualização dos dados dado o modelos extraídos. O processo pode refazer passos anteriores de acordo com conclusões e descobertas, nesta etapa.

9. Ação sobre conhecimento descoberto: usando o conhecimento diretamente, incorporando o conhecimento em outro sistema para adicionar novas funcionalidades, ou simplesmente documentando-a e relatá-la às partes interessadas. Esta fase inclui também a comparação com conhecimentos extraídos anteriormente.

Segundo Fayyad, Piatetsky-Shapiro e Smyth (1996a) o objetivo da descoberta de conhecimento pode ser dividido em dois: verificação de uma hipótese do usuário ou a descoberta autônoma de novos padrões, podendo esta ser usada para predição de comportamento de entidades do domínio e/ou descrição e apresentação desses padrões. A descoberta desses padrões envolve a modelagem das relações entre os dados observados para inferir conhecimento de interesse sendo importante a interpretação humana para melhor compreendê-las. A importância relativa da previsão e da descrição pode variar consideravelmente e seus objetivos podem ser alcançados usando uma variedade de métodos específicos de mineração de dados.

O estudo ainda estabelece que um algoritmo para mineração de dados possui três componentes básicos: representação do modelo, critério de avaliação e os métodos de pesquisa. A representação do modelo é a linguagem usada para descrever padrões detectáveis. Se a representação for muito limitada, nenhuma quantidade de tempo de treinamento ou de exemplos poderá produzir um modelo preciso para os dados. Observe que a aumentar de maneira indiscriminada a capacidade de representação para os modelos aumenta o perigo de que ele aprenda ruídos e peculiaridades daquele conjunto específico ao invés de uma regras capazes de serem generalizadas para dados não observados, o chamado *overfitting* (DIETTERICH, 1995).

Os critérios de avaliação (ou funções de ajuste) medem quão bem um padrão específico (um modelo e seus parâmetros) atende aos objetivos do processo KDD. Modelos preditivos são frequentemente julgados por sua precisão da previsão empírica em alguns conjuntos de teste. Modelos descritivos podem ser avaliados ao longo das dimensões da precisão preditiva, inovação ou avanço, utilidade e compreensibilidade do modelo montado.

Os métodos de pesquisa ocorrem após os dois outros componentes serem fixados, transformando o problema de mineração de dados em uma tarefa de otimização para encontrar os modelos e parâmetros da família selecionada que tenham melhor desempenho segundo os critérios de avaliação. Neste trabalho são utilizadas técnicas de processamento de dados e regressão baseadas em árvores, aplicadas aos dados do GreenHub para geração dos modelos seguindo a metologia KDD, por esta ser a mais flexível e apresentar etapas mais bem definidas, sendo base para as demais metodologias citadas nessa seção. A seção seguinte aborda os principais conceitos da área, algoritmo e análise dos modelos para geração de conhecimento a partir da base estudada.

2.3 APRENDIZADO DE MÁQUINA

As principais técnicas utilizadas para descoberta de conhecimento em um conjunto de dados estão relacionadas ao conceito de Aprendizado de Máquina (*Machine Learning*). O objetivo dessa área é estudar, projetar e melhorar modelos matemáticos que possam ser treinados (uma vez ou continuamente) com dados relacionados ao contexto, para inferir o futuro e tomar decisões sem conhecimento completo de todos os fatores externos. Para isso, o agente adota uma abordagem de aprendizagem estatística, determinando as distribuições de probabilidade, para calcular uma ação, decisão ou valor, com o mínimo de erro (BONACCORSO, 2017).

Essas técnicas se dividem em diversas abordagens, sendo as mais comuns:

- Supervisionadas: é caracterizada pelo conceito de supervisor, cuja principal tarefa é fornecer ao agente uma medida precisa de seu erro (diretamente comparável com os valores de saída). Esta função é fornecida por um conjunto de treinamento (entradas e saídas esperadas). A partir dessas informações, após cada iteração, se o algoritmo for flexível o suficiente e os elementos de dados forem coerentes, a precisão geral aumenta e a diferença entre o previsto e o valor esperado diminui. Os métodos supervisionados também variam de acordo com o objeto alvo para predição/descrição, podendo ser classificadores, ao mapear os dados em um conjunto alvo finito de classes, e regressores, capazes de mapear os dados em função de uma variável alvo de valor real.

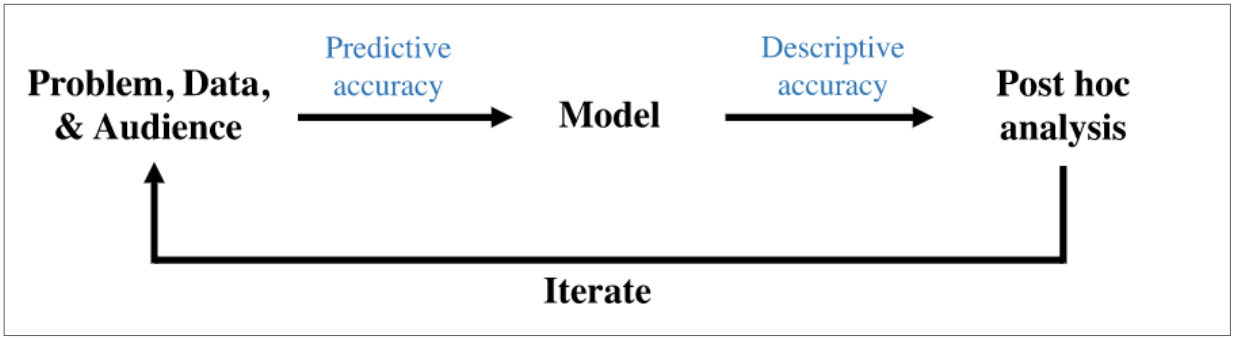
- Não-supervisionadas: Esta abordagem é baseada na ausência de qualquer supervisor, nem medidas de erro. O conjunto de dados não possuem uma saída esperada, cabendo à máquina, através do processo de aprendizado, descobrir como gerá-las. É muito utilizada quando é necessário aprender como um conjunto de elementos podem ser agrupados de acordo com sua similaridade (ou distância a medida).

- Aprendizagem por reforço: É baseada no feedback fornecido pelo ambiente, geralmente chamado de recompensa (às vezes, uma negativa é definida como uma penalidade) e que é útil para entender se uma determinada ação realizada em um estado é positiva ou não. A sequência de ações mais úteis é uma política que o agente tem que aprender, para ser capaz de tomar sempre a melhor decisão em termos de maior recompensa imediata e cumulativa. Este conceito é baseado na ideia de que um agente racional sempre persegue os objetivos que podem aumentar sua riqueza.

Modelos de Machine Learning também são capazes de produzir conhecimento sobre as relações e padrões contidos nos dados, que junto com outros diversos outros métodos (visualizações, equações matemáticas e etc.), são usados para interpretação dos resultados, sendo esta usada para avaliar os modelos, melhorando-os e tornando-os mais confiáveis. Murdoch et al. (2019) propuseram o *framework* PDR (*Predictive, Descritive, Relevant*) para interpretação em Machine Learning, definindo-a como sendo a extração de conhecimento relevante, promovendo ideias para determinada audiência, a partir dos relacionamentos contidos nos dados ou aprendidos pelos modelos de aprendizado de máquina, frequentemente resultando em guias, ações ou descobertas.

Segundo o estudo, no ciclo da descoberta do conhecimento as atividades relacionadas a interpretação ocorrem logo no início, pois é a partir das informações do domínio, dos dados disponíveis e do interesse dos *stakeholders*, que os dados a serem usados e os métodos para modelagem e análise dos resultados são escolhidos. Todas as etapas relacionadas a interpretação, são iterativas seguindo os princípios do ciclo de mineração de dados onde está inserida, podendo ser repetidas e melhoradas de acordo com a avaliação dos resultados de acordo com a Figura 5. No início do ciclo é definido um problema de domínio que se deseja compreender utilizando os dados e a natureza dele desempenha um papel na interpretabilidade, pois o contexto e os interessados são essenciais para determinar quais métodos usar.

Figura 5 – Ciclo de um projeto voltado para interpretação



Fonte: (MURDOCH et al., 2019)

Na fase de modelagem, com base no problema escolhido e nos dados coletados, o profissional então constrói um modelo preditivo. Nesta fase, o profissional processa, limpa e visualiza os dados, extraindo os recursos necessários e seleciona um modelo (ou vários modelos) para processar os dados. Os métodos são escolhidos entre os mais simples e fáceis de interpretar, e os mais complexos, que podem ter um melhor desempenho, observando os três critérios estabelecidos. Os métodos tem como objetivo fornecer uma base para compreensão das relações encontradas nos dados, sem perder a precisão das predições, ser estável quando submetida a pequenas variações nos dados ou modelos (MURDOCH et al., 2019). O *framework* estabelece diversas considerações relacionadas ao desenvolvimento dos modelos escolhidos, que auxiliam diretamente na satisfação da interpretabilidade, como esparsidade dos dados, simulabilidade do modelo, modularidade dos modelos, e engenharia das features.

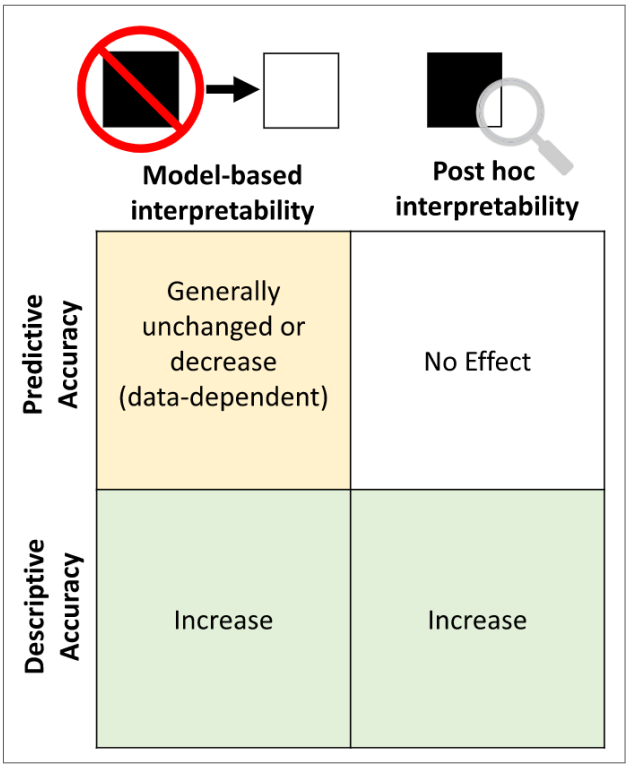
Na fase de análise *post hoc* do ciclo de vida de um processo de mineração, os modelos gerados, tendo sua capacidade preditiva estabelecida, são analisados em busca de respostas para a pergunta original, geralmente utilizando vários métodos de interpretação para extrair as informações. Estas podem então ser analisadas e exibidas através de análise de dados padrão,

como gráficos de dispersão e histogramas, estabelecendo assim a precisão descritiva do estudo (MURDOCH et al., 2019).

A maioria dos métodos de interpretação *post hoc* podem ser classificados como a nível de predição, quando buscam explicar e caracterizar predições individuais realizando assim uma interpretação local, e as abordagens relacionadas aos conjuntos de dados, estudando relações mais gerais, entre classes ou subconjunto de dados, sendo estas definidas como interpretações globais. Os métodos *post hoc* utilizam diversas técnicas para análise da importância das features e suas relações, tendências e outliers, além de visualizações gráficas.

Murdoch et al. (2019) sugerem que os métodos selecionados para modelagem dos dados e análises de resultados para o problemas de Machine Learning, sejam realizados observando três critérios: precisão preditiva, se um tem baixo desempenho em aprender relações subjacentes nos dados, é improvável que qualquer informação extraída do modelo seja precisa; a precisão descritiva, que se refere ao quanto aquele modelo é capaz de representar as relações aprendidas dos dados; e relevância, pois o modelo deve agregar novas ideias e perspectivas aos interessados. O impacto dos métodos escolhidos em cada fase segue o esquema da Figura 6, onde se destaca como a precisão descritiva é mais sensível às escolhas feitas em todas as fases.

Figura 6 – Impacto dos métodos de interpretação na precisão preditiva e descritiva



Fonte: (MURDOCH et al., 2019)

2.3.1 Regressores em Árvore

A análise de regressão é uma técnica de aprendizagem estatística bem conhecida usada para inferir a relação entre uma variável dependente (Y) e independentes $(x_1, x_2..., x_n)$, sendo (Y) também conhecida como variável alvo, resposta ou resultado, e as variáveis $(x_1, x_2..., x_n)$ como preditores, variáveis explicativas, covariáveis ou features. Estas análises visam estimar uma relação matemática, uma função (f), que possa explicar a variável alvo em termos dos preditores, usando as observações $(xi, Yi), i = 1, ..., n$, coletadas em n observações (PETER et al., 2018). Elas podem ser usadas para verificar a existência dessas relações, compreender e quantificar a natureza delas, além de fazer predições baseadas em novos valores de entrada das features.

É o processo de estimar o valor de um alvo contínuo como uma função de um ou mais preditores, um conjunto de parâmetros e uma medida de erro, também chamado de residual, que é a diferença entre o valor esperado e o valor previsto da variável dependente. O treinamento de um modelo de regressão tem como objetivo encontrar os valores dos parâmetros que minimizam uma medida do erro (SURAMPUDI, 2021).

Existem diferentes famílias de funções de regressão, e em Stulp e Sigaud (2015) elaboraram um estudo diversos tipos de algoritmos, demonstrando as relações entre eles e suas representações de função, oferecendo uma visão dos princípios de seu funcionamento interno, explicando acerca das diferentes abordagens e alguns de seus meta-parâmetros, valores usados como parâmetros de entrada para que o algoritmo tenha melhor desempenho no desenvolvimento dos modelos.

Esses algoritmos incluem diversos métodos clássicos, sendo os baseados em regressão linear e logística como os representantes mais populares de modelos paramétricos padrão, para lidar com essas tarefas. No entanto, em certas situações, esses métodos clássicos podem estar sujeitos a severas limitações. Em alguns casos onde existem um conjunto grande de preditores e muitas variáveis categóricas, a esparsidade torna difícil estimar a importância e dessas variáveis, e suas relações de maneira confiável, limitando-as a padrões restritos de associação (STROBL; MALLEY; TUTZ, 2009).

Razi e Athappilly (2005) realizam um estudo comparativo entre a capacidade preditiva de diferentes tipos de algoritmos aplicados em um mesmo conjunto de dados. Foram avaliados regressores não lineares, redes neurais e árvores de regressão, mostrando que os resultados são bastante parecidos. Dentre as conclusões, o estudo observa que árvores de decisão podem

escalar com facilidade, ao mesmo tempo que lidam muito bem com preditores categóricos e binários e variável alvo contínua.

2.3.2 Decision Tree

Uma árvore de decisão é um algoritmo de tomada de decisão que atribui uma probabilidade a cada uma das escolhas possíveis com base no contexto da decisão que é determinada por uma sequência de sobre o contexto, onde a i-ésima pergunta feita é determinada exclusivamente pelas respostas às perguntas anteriores. Cada pergunta feita pela árvore de decisão é representada por um nó da árvore e as possíveis respostas para essa pergunta estão associadas a ramos que emanam do nó. Cada nó define uma distribuição de probabilidade no espaço de decisões possíveis. Um nó no qual a árvore de decisão para de fazer perguntas é um nó folha que representam os estados únicos no problema (MAGERMAN, 1995).

A construção de uma árvore de regressão básica é baseada no particionamento binário recursivo dos dados em conjuntos retangulares de todas as maneiras possíveis. O objetivo é selecionar a divisão em que o custo da função escolhida para avaliar a precisão, como por exemplo a soma dos desvios ao quadrado da média de um grupo de entradas ou o desvio padrão, seja minimizada (XU et al., 2005).

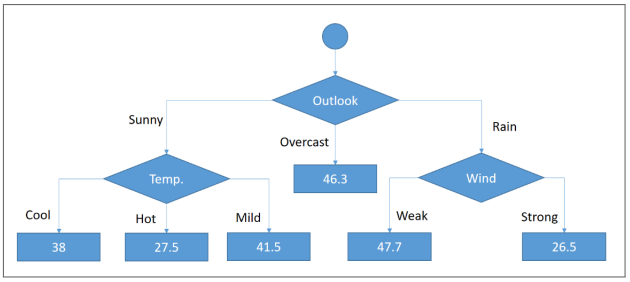
Figura 7 – Exemplo de dados - Decision Tree

Day	Outlook	Temp.	Humidity	Wind	Golf Players
1	Sunny	Hot	High	Weak	25
2	Sunny	Hot	High	Strong	30
3	Overcast	Hot	High	Weak	46
4	Rain	Mild	High	Weak	45
5	Rain	Cool	Normal	Weak	52
6	Rain	Cool	Normal	Strong	23
7	Overcast	Cool	Normal	Strong	43
8	Sunny	Mild	High	Weak	35
9	Sunny	Cool	Normal	Weak	38
10	Rain	Mild	Normal	Weak	46
11	Sunny	Mild	Normal	Strong	48
12	Overcast	Mild	High	Strong	52
13	Overcast	Hot	Normal	Weak	44
14	Rain	Mild	High	Strong	30

Fonte: (SERENGIL, 2018)

Loh (2011) fez um estudo comparativo entre diversas técnicas de árvores de regressão destacando que a complexidade total é compartilhada entre a estrutura de árvore e o conjunto de modelos de nós. A complexidade de uma estrutura de árvore geralmente diminui à medida que a complexidade dos modelos aumenta. Estruturas modulares de árvores podem ser usadas para produzir insights mas tendem a ter baixa precisão de previsão, a menos que os dados

Figura 8 – Forma Final - Decision Tree



Fonte: (SERENGIL, 2018)

sejam suficientemente informativos e abundantes para produzir uma árvore com muitos nós, observando que estruturas demasiadamente grandes, afetam a interpretação das relações.

Um dos algoritmos investigados nesse estudo é o Decision Tree, implementado pela biblioteca do scikit-learn para Python (LEARN, 2021) tem como objetivo criar um modelo que preveja o valor de uma variável de destino aprendendo regras de decisão simples inferidas a partir dos recursos de dados, podendo ser vista como uma aproximação constante por partes. A biblioteca Decision Tree do scikit-learn usa uma versão otimizada do algoritmo CART (BREIMAN et al., 1984), no entanto, a implementação do scikit-learn não suporta variáveis categóricas por enquanto, sendo necessário o uso de técnicas que as convertem em variáveis binárias. Sua implementação é muito semelhante ao C4.5 (QUINLAN, 2014), mas difere porque suporta variáveis de destino numéricas (regressão) e não computa conjuntos de regras. O CART constrói árvores binárias usando feature e o threshold que geram o maior ganho de informação em cada nó.

Árvores de decisão podem criar árvores supercomplexas que não generalizam bem os dados, produzindo overffitng. Poda (prunning), definir o número mínimo de amostras necessárias em um nó ou definir a profundidade máxima da árvore são necessários para evitar esse problema, que pode deteriorar a precisão das predições o que leva a uma menor capacidade de generalização (LEARN, 2021; XU et al., 2005).

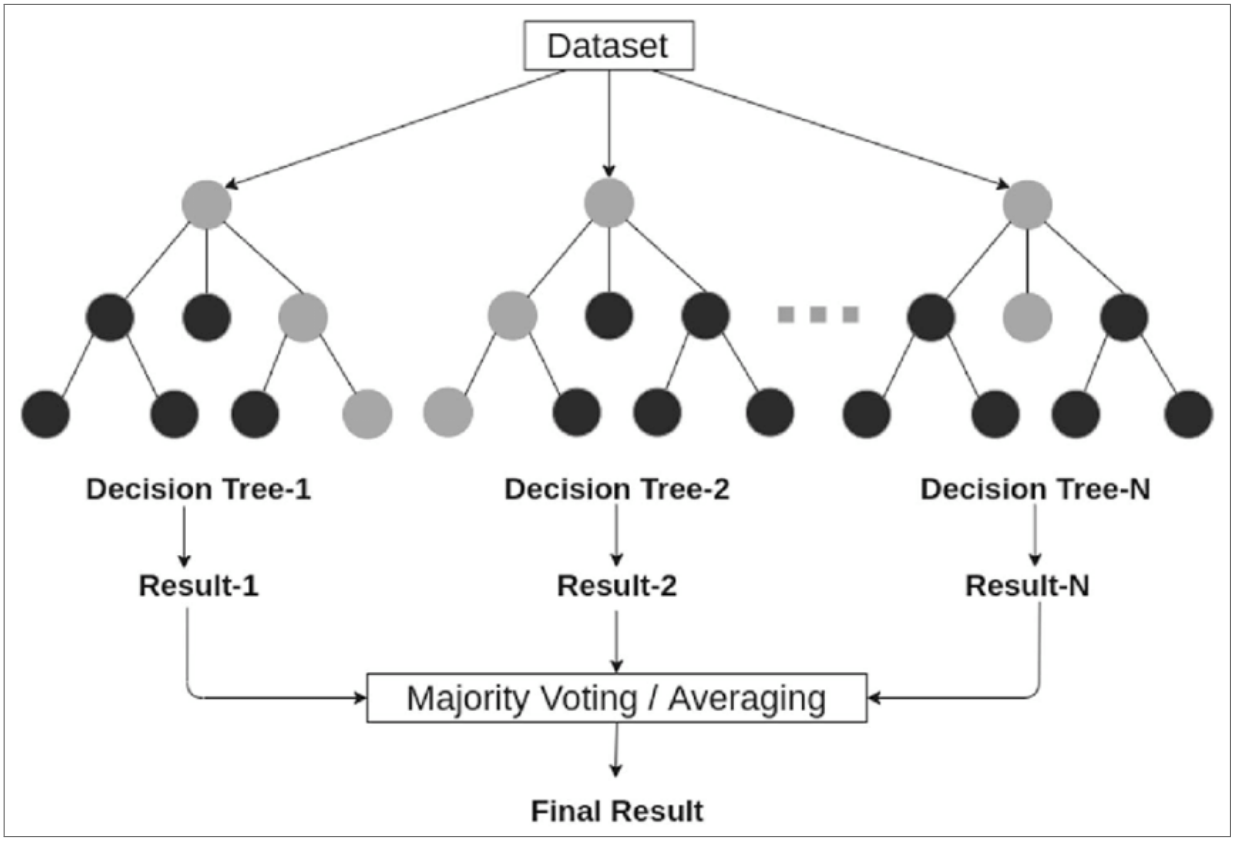
2.3.3 Random Forest

Em aprendizagem supervisionada, cada ponto de dado constitui um vetor de preditores e uma variável alvo e, a partir disso, se assume a existência de uma função que representa essas relações. Métodos de machine learning simples, como Decision Tree, funcionam procurando dentro do espaço dessas funções, também chamadas de hipóteses, uma única função que

melhor represente novos pontos. Técnicas de ensemble learning, constroem um conjunto de hipóteses (ensemble) cujos as saídas são agregadas para melhor aproximar o resultado (DIETTERICH et al., 2002). Em (DIETTERICH, 2000) são apresentadas diversas técnicas para serem utilizadas com o objetivo de agregar alguns tipos de algoritmos e resultados experimentais mostraram melhoras significativas utilizando-os em comparação aos modelos únicos.

Random Forest é um algoritmo de ensemble learning que utiliza uma combinação de árvores de predição onde cada árvore é construída a partir de valores de um vetor aleatório de preditores independente com a mesma distribuição para todas as árvores da floresta, onde o erro converge a um limite à medida que ela cresce, evitando o *overfiting* (BREIMAN, 2001). A utilização de diferentes amostras independentes do conjunto dos preditores para treinar diversos modelos agregando-os, é conhecida como Bagging (acrônimo para *bootstraping* e *agreggeting*)(BREIMAN, 1996). Nos classificadores, os modelos são agregados através de eleição, onde cada modelo tem um voto para eleger a mais classe mais popular ou médias de probabilidade, enquanto nos regressores é realizada uma média dos resultados.

Figura 9 – Random Forest - Example



Fonte: (SARKER, 2021)

Estimativas internas monitoram o erro, a força e a correlação e são usadas para mostrar

a resposta ao aumento do número de preditores a cada divisão, além de medir a importância delas (BREIMAN, 2001). O estudo demonstrou, através de experimento, um melhor desempenho preditivo do Random Forest, ao adicionar o fator aleatório às técnicas de *bagging*.

2.3.4 XGBoost

A partir das técnicas de ensemble learning surgiram outras técnicas que tinha como objetivo melhorar ainda mais as predições, utilizando a agregação de modelos, como o Boosting, que tem a ideia combinar diversos modelos baseando-se na dificuldade em de predição de determinado ponto, para atribuir mais ou menos peso de acordo com o erro (BROWNLEE, 2021). Boosting se assemelha em diversos pontos a técnica de Bagging, mas não envolve amostragem por bootstrap, ao invés disso as árvores crescem sequencialmente utilizando as informações (pesos) que as anteriores atribuíram ao conjunto de treino original, e atualizando-as ao terminar. Uma outra diferença, é que técnicas de boosting são elaboradas especificamente para algoritmos que suportem esse tipo de atribuição, como os algoritmos baseados em árvores de decisão.

A ideia fundamental de Boosting é combinar diversos modelos fracos, com previsões pouco melhores que palpites aleatórios em classificação ou que simplesmente obter a média em caso de regressores, para criar regressores mais eficientes e com melhor desempenho que um modelo único de alta precisão. Os modelos são construídos de maneira tendenciosa focando em fazer previsões mais precisas em linhas com peso maior até que o número atinja a quantidade de modelos estipulada. A contribuição de cada modelo para a previsão final é uma soma ponderada do desempenho de cada modelo, por exemplo, uma média ponderada ou voto ponderado (ZHANG; MA, 2012).

Diversos algoritmos surgiram baseados nessas idéias, sendo o Adaboost (FREUND; SCHAPIRE, 1997) considerado o primeiro a obter mais destaque, tendo diversos outros algoritmos sido desenvolvidos inspirados e a partir dele. Uma das mais usadas é a Gradient Boosting (FRIEDMAN, 2001) que propõe utilizar a minimização de alguma funções de perda, como a raiz dos erros, ao invés dos pesos das predições, como alvo de melhoria dos modelos subsequentes. Como mostrado por Friedman (2001), essa técnica combinada com árvores de decisão (Tree Boosting), é capaz de reestimar os parâmetros das folhas das árvores, para que as subsequentes minimizem o erro.

Embora tenha representado uma evolução em termos de predição, os algoritmos mais clás-

sicos de Gradiente Boosting, eram um pouco lentos, principalmente com conjunto de dados muito grandes, devido, em grande parte, à construção sequencial de seus modelos. A partir das ideias lançadas por esses algoritmos e necessidade de desempenho computacional, foi desenvolvido o XGBoost (CHEN; GUESTRIN, 2016), um sistema *open source* escalável de ensemble para Gradiente Boosting, baseado em árvores. O sistema é tido como bastante eficiente, tendo vencido, sozinho ou combinado com outros algoritmos, diversas competições ao longo dos anos, em predições de vendas, de comportamento, detecção de emoções, classificação de texto, entre outras.

O foco deste trabalho é analisar sob a perspectiva dos algoritmos de regressão baseados em árvore citados nesta seção (Decision Tree, Random Forest e XGBoost), o tempo de decaimento, em segundos, entre pares de registros sequências num mesmo dispositivo do GreenHub. Por fim, o impacto das variáveis, bem como as relações contidas nos modelos produzidos serão observados segundo os conceitos da precisão preditiva, descritiva e de relevância.

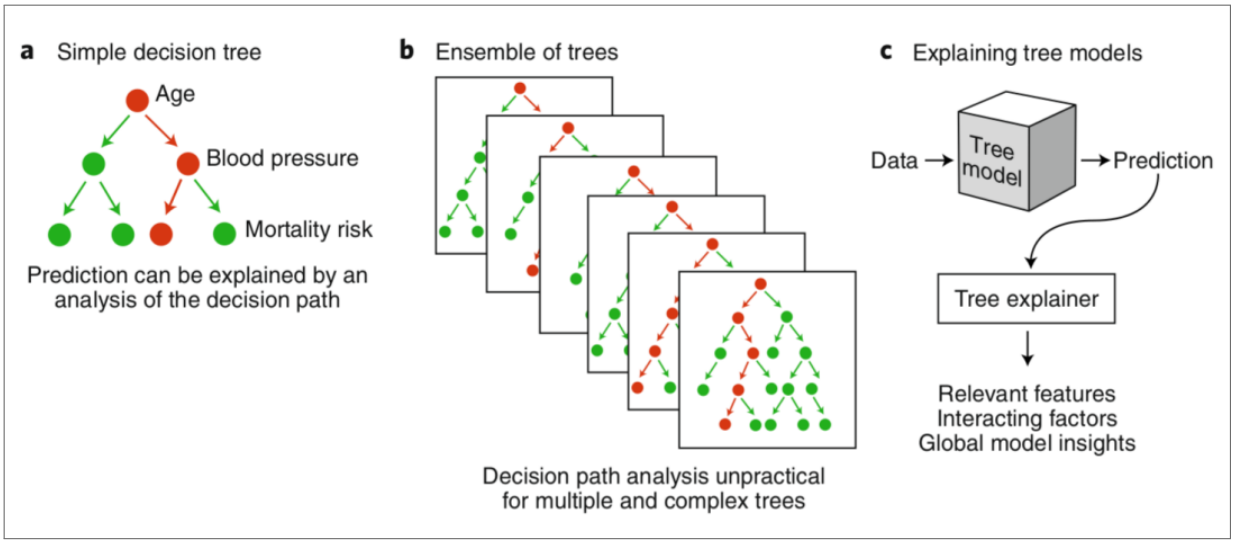
2.4 TREEEXPLAINER

Diversas pesquisas têm sido conduzidas com o objetivo de desenvolver técnicas que permitam explicar decisões individuais, a respeito de uma classificação ou regressão feita pelo algoritmo, também chamadas de interpretações locais, podendo essas serem agregadas em visões que ajudem a compreender as relações gerais do modelo, aumentando assim a confiabilidade e estabilidade das interpretações globais.

Os modelos baseados em árvore estão entre os algoritmos de aprendizado de máquina mais populares e bem-sucedidos na prática. Em desafios de machine learning, como os propostos na plataforma Kaggle, algoritmos como Random Forest e XGBoost estão sempre sendo usados nas soluções vencedoras, tendo este último participado de 17 vitórias, de 29 competições, em 2015 (CHEN; GUESTRIN, 2016). Diversas técnicas foram assimiladas por esses tipos de modelos, melhorando seu desempenho, atribuindo novas características e tornando-os mais robustos e versáteis, entretanto algumas dessas evoluções, aumentaram a complexidade da interpretação e das relações contidas nos modelos (SAMEK, 2020).

Com o objetivo de solucionar esses problemas, Lundberg et al. (2019) desenvolveram o TreeExplainer, um método prático baseado no modelo agnóstico baseado nos valores clássicos de Shapley da teoria dos jogos (SHAPLEY, 1953), capaz de explicar as decisões locais desses modelos em termos das entradas fornecidas. No método de Shapley esses valores são atribuídos

Figura 10 – Explicação para Algoritmos de Árvore



Fonte: (SAMEK, 2020)

conforme a contribuição de cada jogador de uma coalizão usada para composição do total.

Conforme explicado em (MOLNAR, 2022), o "jogo" é realizar a predição para uma única instância. O "ganho"é a predição real para esta instância menos a predição média para todas as instâncias. Os "jogadores" são os valores das variáveis independentes na instância que colaboram para predição de um determinado valor. O valor de Shapley de uma variável independente é sua contribuição marginal média em todas as possíveis, ou seja, a média de quanto ela afeta as previsões.

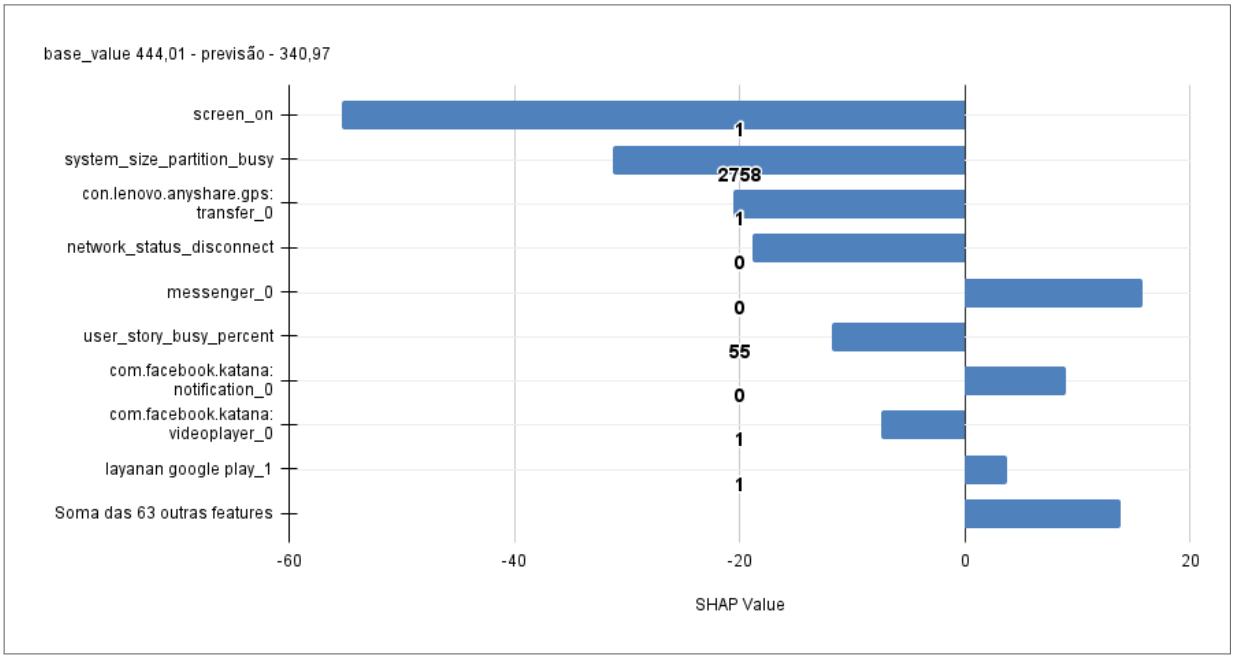
O modelo unificado proposto em (LUNDBERG; LEE, 2017), se baseia nessa teoria, e tem como objetivo unificar as abordagens de interpretação, utilizando as perspectivas dos modelos para explicar suas predições, apresentando uma estrutura unificada para interpretar previsões, SHAP (SHapley Additive exPlanations), que atribui a cada variável independente um valor de importância para uma previsão específica. A abordagem proposta neste estudo contempla três importantes características presentes na teoria original, não presente em outros modelos:

- Precisão Local (Local Accuracy): Ao aproximar o modelo original f para um entrada x, a precisão local requer que o modelo de explicação pelo menos corresponda à saída de f para o entrada simplificada x' (que corresponde à entrada original x).
- Falta (Missingness): Se as entradas simplificadas representarem a presença de variáveis independentes, então essa propriedade exige que os recursos ausentes na entrada original não tenham impacto.

- Consistência (Consistency): Se um modelo muda, de maneira que a contribuição de alguma variável independente aumenta ou permanece igual independentemente dos outras entradas, a atribuição dessa variável independente não deve diminuir

O TreeExplainer é baseado nesse modelo, permitindo o cálculo exato de explicações locais ideais para modelos baseados em árvores, estende as explicações locais para capturar diretamente as interações das variáveis independentes, fornecendo um novo conjunto de ferramentas para entender a estrutura do modelo global com base nas muitas explicações locais (LUNDBERG et al., 2019). A necessidade de explicar as previsões a partir de modelos de árvores é generalizada e é particularmente importante em muitas áreas, onde os padrões descobertos por um modelo são muitas vezes ainda mais importantes do que o desempenho de previsão do modelo.

Figura 11 – Exemplo de Interpretação Local



Fonte: (AUTOR, 2022)

A Figura 11 representa uma previsão realizada do tempo de consumo de um nível de bateria de um dispositivo. O valor base (base_value) representa a média de todas predições realizadas para o determinado conjunto de dados utilizado. Os SHAP Values indicados representam a contribuição marginal de cada variável independente para a diferença entre essa previsão (340,97) e a média (444,01).

Observando o gráfico é possível perceber que o fato da tela estar ligada contribui bastante para diminuição do tempo, uma vez que indica basicamente que o aparelho está em

uso, possivelmente com alguma interação. Além disso, um processo de vídeo do Facebook (com.facebook.katana:videoplayer_0) está em execução, estado indicado pelo valor 1 correspondente no centro do gráfico, contribuindo para a queda do tempo previsto.

O gráfico ainda aponta que o celular esta conectado (network_status_disconnect = 0), fazendo a previsão cair ainda mais em relação a média. Por fim, as notificações e o Messenger, estarem desabilitados (ou não estarem em execução) contribui positivamente para o tempo em relação a média prevista, bem como a soma das demais variáveis independentes.

Este trabalho busca analisar o impacto global das principais variáveis independentes contidas no GreenHub para cada modelo de regressão, senda esta a média absoluta das contribuições em todas as predições realizadas pelos algoritmos de regressão, observando a precisão, a capacidade descritiva dos modelos e a relevância dos resultados.

2.5 TRABALHOS RELACIONADOS

No estudo realizado por Zhao et al. (2011), o contexto dos dispositivos móveis foi utilizado para predição de consumo de energia, utilizando para isso as relações entre esses estados dos componentes (CPU, configurações da rede e da tela) identificados como sendo os principais consumidores de energia e a taxa de descarga da bateria. Foram realizados diversos experimentos em laboratório, utilizando diversos tipos de aplicativos em diferentes cargar de funcionamento em 40 cenários diferentes. Foram realizadas diversas regressões para criação de modelos de predição que também quantificassem a relação desses componentes com o consumo. Embora o estudo tenha relatado modelos com precisão acima de 90%, os componentes considerados não refletem o contexto dos dispositivos, nem de suas aplicações e processos. Além disso, por ter sido realizado em laboratório, com número reduzido de cenários, reflete de maneira muito limitada a heterogeneidade dos contextos do universo móvel, tão pouco o uso em condições reais. Essa limitação, além da baixa quantidade de dados, são fatores constantes nesse tipo de estudo.

Com o objetivo de analisar anomalias no consumo de energia de dispositivos móveis, Oliner et al. (2013) utilizaram uma base de dados colaborativa para inferir uma especificação (uso de energia esperado) para determinado aplicativo, considerando os desvios dessa inferência como anomalias. Estas foram classificadas em dois tipos: hog, quando o uso de uma aplicação eleva o consumo acima da média normal do dispositivo, e bugs, quando o app em um cliente apresenta consumo acima da média de suas outras instâncias nos demais aparelhos. Diversos

aplicativos com comportamentos anômalos foram detectados com alta taxa de precisão, e recomendações feitas a partir das descobertas foram capazes de estender o tempo de duração da energia em cerca de 10%. Por focar no comportamento do usuário no uso das aplicações, o estudo não traz uma análise da relação entre os diversos componentes e processos, como comportamento energético e possíveis anomalias, aspecto interessante para desenvolvedores e fabricantes compreenderem melhor sua relação com o contexto.

Guo, Wang e Chen (2017) introduziram uma nova abordagem de cálculo de consumo energético de um aplicativos utilizando dados colaborativos em larga escala, que obteve uma taxa de erro inferior a 10%, quando comparada com a simulação em laboratório. Além disso, analisou as relações entre diferentes tipos de aplicativos e padrões no consumo de bateria, além de hábitos energéticos dos dispositivos. O estudo fez diversas descobertas interessantes, quantificando consumo de energia de aplicativos em foreground por categoria, descobrindo comportamento energético dos usuários e detectando diferenças inesperadas de consumo entre estados do dispositivo. Entretanto, o trabalho limita os aspectos de hardware apenas à tela estar ligada ou não, o que pouco reflete a diversidade de dispositivos e suas configurações. Embora colete dados sobre todos os processos, apenas aqueles em foreground são considerados, o que limita a capacidade de estudar as diversas relações existentes num meio tão heterogêneo e dinâmico.

Com o objetivo de analisar aplicativos em busca de *bugs* energéticos, Gao et al. (2017) desenvolveram uma ferramenta, chamada E-Android, capaz de monitorar esses eventos colaterais, sendo capaz de detectar ataques com o de esgotar as baterias. Para os experimentos, o sistema proposto foi implementado no *framework* do Android, atingindo mais precisão na detecção de ataques com seu modelo. Esse tipo de coleta de dados e modelagem, embora mais eficiente na comparação com abordagens semelhantes, requer uma configuração mais especializada do que a a instalação de um aplicativo, não estando disponível para maioria dos usuários.

No estudo realizado por Alawnah e Sagahyroon (2017), foi proposta uma abordagem para modelar o consumo de energia sob a perspectiva do usuário, para isso criando uma aplicativo de monitoramento para capturar logs do sistema Android, semelhante ao BatteryHub, aplicativo usado para construção do GreenHub. O aplicativo capturou informações relacionadas a diversos fatores como brilho da tela, conectividade com a rede, manipulação de dados e serviços de mensagens, utilizando algumas abordagens de Redes Neurais para modelar as relações entre esses componentes, mostrando assim a eficiência desse tipo de abordagem. Muito embora

tenha utilizado dados de estudo real, o estudo acabou limitando sua coleta a 10 smartphones, e utilizando apenas configurações de hardware, sem os processos ou aplicativos, o que pode limitar um pouco a generalização dos resultados.

Nucci et al. (2017) desenvolveram a PETrA (*Power Estimation Tool for Android*), ferramenta desenvolvida para estimar o consumo energético de aplicativos com granularidade suficiente para analisar métodos. O estudo faz uma ampla gama de experimentos em diversos aplicativos, em alguns modelos selecionados, alcançando excelentes resultados de precisão em comparação com abordagens semelhantes, sem a utilização de hardware externos. Contudo não considera sensores e outros componentes de hardware, não estimando com a mesma precisão aplicativos que utilizam esses recursos mais intensamente.

yan2019modeling realizaram um estudo focado na relação dos diversos tipo de conectividade 4G de dispositivos Android com o consumo de bateria. O estudo considerou não apenas a tecnologia da conexão, mas classificou diversas topologias utilizadas para o fornecimento de serviços de rede. Foram analisados fatores relacionados ao tráfego, sinal, duração das transferências, processamento e consumo. O estudo utiliza ferramentas para medir a energia consumida com o objetivo de modelá-la em função das conexões, em certas condições de uso de aplicativos e serviços. O estudo possui as limitações de um experimento controlado, e considera variáveis isoladas dentro do contexto móvel, porém trás uma perspectiva interessante sobre um importante aspecto desses dispositivos.

Em seu trabalho, Neto et al. (2021) desenvolveram, através de experimentos, um modelo para consumo de energia baseado em padrões de interação do usuário, analisando os dados de maneira automática. Para isso consideraram a potência utilizada (calculada a partir da corrente e da voltagem) em intervalos de tempo de 1 segundo, como sua unidade de medida. Utilizando diversas APIs do Android para capturar as informações relacionadas a criação da variável do estudo e aos aspectos mais relevantes para o consumo de energia, de maneira parecida com o que faz o BatteryHub, usado na construção do GreenHub. Analisando os dados através de técnicas de Aprendizagem de Máquina e Redes Neurais, desenvolveram uma metodologia para criação automática desses modelos. Embora tenham estabelecido uma metodologia bastante robusta, com resultados interessantes, capaz de analisar o comportamento energético dos dispositivos com os dados fornecidos pelo Android, não faz uso de uma base de dados que agregue esses resultados, como o GreenHub.

Ding, Wang e Wang (2021) propuseram uma abordagem de modelagem chamada E-Sub, com o objetivo de relacionar o comportamento do usuário com o consumo energético dos

dispositivos. Utilizando mecanismos como agregação de aplicativos, decomposição de uso e camadas que identificassem padrões nesses usos, através de técnicas de clustering, para lidar com a alta dimensionalidade e dispersão dos dados. Analisando dados de uma base em larga escala que ranqueia cada aplicativo segundo o consumo seu energia diário, dividindo o tal diário do dispositivo entre os componentes de hardware, os pesquisadores identificaram diversos padrões de comportamento do usuário que foram relacionados ao consumo energético. O trabalho propõe uma abordagem diferente de modelagem da usada neste trabalho, alcançando bons resultados em relação a outras técnicas semelhantes, apresentando diversas perspectivas para os dados analisados.

Pereira et al. (2021) exploraram a base GreenHub, de maneira qualitativa e descritiva, demonstrando o quanto seus dados são representativos em relação a as diversas características de contextos do ecossistema móvel. A partir disso, analisou as diferenças de tendências de comportamento energético entre localidades, configurações, fabricantes, modelos e aplicativos mais populares. Os resultados apresentam uma gama muito interessante de descobertas, sendo possível identificar variações de comportamento através de diversos contextos. Entretanto, a relação entre os diversos atributos estudados e seu impacto no consumo energético não foram mensurados.

Em (DUNN; MINGARDI; ZHUO, 2021), foi realizada uma comparação entre modelos baseados em árvores (CART, Optimal Trees e XGBoost) através de diversos experimentos, utilizando a abordagem SHAP para identificar corretamente o subconjunto relevante de variáveis através dos vários experimentos. O estudo aponta que SHAP e XGBoost, em relação aos outros conjuntos, subestimam consistentemente a importância de características e atribuir importância significativa a características irrelevantes, o que pode levar ao abandono de um número de recursos que são úteis ou, alternativamente, manter recursos irrelevantes no modelo e atribuir importância significativa ao explicar o modelo.

Na comparação entre os trabalhos realizados, é possível notar que maior parte dos trabalhos selecionados para esta seção modelam o consumo de energia a partir de poucos dispositivos, com dados gerados em situações simuladas de uso o que torna difícil generalizar os resultados, dada a vasta heterogeneidade do contexto móvel, em especial do Android. O trabalho realizado nessa pesquisa tem como objetivo viabilizar a exploração de dados que representem esse aspecto da diversidade, podendo inclusive gerar direcionamentos para para essas pesquisas.

Algumas das pesquisas também utilizam um número limitado de variáveis no estudo, observando de maneira isolada determinados aspectos do hardware. Embora isso melhore os

resultados, é inegável que considerar mais fatores durante as pesquisas, especialmente quando atuam em conjunto, torna a descoberta mais próxima da dinâmica dos dispositivos móveis durante o uso. Alguns desses trabalhos abordam grandes bases de dados como o GreenHub, fazendo uso de outras técnicas de Aprendizado de Máquina. Um estudo se faz necessário para comparação entre as técnicas aplicadas em uma mesma base de dados, não apenas avaliando o desempenho, mas observando em que elas podem se complementar.

2.6 CONSIDERAÇÕES

Neste capítulo foram apresentados os conceitos fundamentais utilizados neste trabalho. Para mineração da base de dados GreenHub será utilizada a metodologia KDD para descoberta do conhecimento, com as técnicas de regressão baseadas em árvore (Decision Tree, Random Forest e XGBoost) para realização das predições e criação dos modelos, posteriormente submetidos ao TreeExplainer, para análise do impacto das fatores presentes na base no intervalo de tempo de decaimento dos níveis de bateria.

3 METODOLOGIA

Neste capítulo serão relatadas todas as atividades executadas para desenvolvimento da pesquisa, seguindo as etapas descritas no processo KDD, que possui passos bem definidos capazes de serem generalizados para os mais diversos tipos de problemas na descoberta do conhecimento. O objetivo do processo é utilizar métodos de regressão em árvore (Decision Tree, Random Forest e XGBoost), juntamente com o TreeExplainer, na exploração de dados do GreenHub, respondendo às seguintes perguntas:

PP1. É possível utilizar técnicas tradicionais de aprendizado de máquina para prever o uso de bateria de dispositivos móveis a partir de dados de *crowdsourcing*?

PP2. Quais são os componentes mais impactantes para os modelos?

As etapas do processo de descoberta do conhecimento do KDD, já descritas nesse trabalho, bem como todas as decisões, estão descritas neste capítulo, e foram realizadas neste projeto seguindo os seguintes passos:

1. Compreensão do domínio da aplicação: Os dados do GreenHub são um registro do estado de diversas atividades dos sistema Android dos dispositivos colaboradores a cada mudança do nível de bateria.

2. Criação do conjunto alvo: A variável alvo é o intervalo de tempo entre duas sequências de decaimento de 1% de bateria. A partir disso foram estudados o impacto de todos os estados (configurações, aplicativos e processos) nas previsões realizadas.

3. Limpeza de dados e pré-processamento. Os arquivos contendo os dados foram processados e selecionados seguindo os seguintes critérios:

    - Inclusão de dados de descarga da bateria (*Discharging* no campo *battery_state*).
    - Inclusão de registros que formavam pares para criação da variável alvo com diferença de nível de bateria de 1%.
    - Exclusão de registros que não possuíam ao menos um processo relacionado.
    - Exclusão de registros que não possuíam dados de configuração relacionados.
    - Exclusão de registros que não possuíam registro subsequente para criação da variável alvo.

- Exclusão de registros que formavam variáveis alvo abaixo de 1 segundo.
- Exclusão de registros que formavam variáveis alvo cujo valor foi detectado como *outlier*.

4. Redução dos dados e projeção: Foram selecionados os dados dos 100 dispositivos mais populares da base, juntamente com os processos e aplicativos mais populares limitados a no máximo 25 de cada.

5. Escolha do método de mineração: Devido a natureza contínua da variável alvo e das características variadas das features, foram selecionados algoritmos regressores baseados em árvore.

6. Análise e modelagem exploratória: Foram selecionados algoritmos de árvore de diferentes abordagens e complexidade: Decision Tree, Random Forest e XGBoost. Os modelos resultantes foram submetidos ao TreeExplainer a fim de explorar suas capacidades descritivas.

7. Mineração: Os modelos foram submetidos ao TreeExplainer e foram observadas as features de maior importância para os modelos, bem como o comportamento de a relação de algumas delas com as predições.

8. Interpretação dos padrões minerados: Foram produzidos gráficos e tabelas para sumarizar e apresentar os resultados encontrados.

9. Ação sobre conhecimento descoberto: Foram destacados alguns dos conhecimentos encontrados, além de discutida a capacidade dos modelos de encontrar conhecimento, lançando bases para pesquisas futuras mais profundas e sob diferentes perspectivas.

Todas etapas foram realizadas utilizando a linguagem Python, estando os *scripts* desenvolvidos para execução de todas as etapas, bem como os arquivos com todos os resultados, se encontram no repositório da pesquisa[1].

3.1 COMPREENSÃO DO DOMÍNIO DA APLICAÇÃO

O método consiste em processar os dados para gerar modelos de regressão a partir dos dados obtidos sobre variações de energia. Esses modelos devem identificar e quantificar a

[1] https://github.com/https://github.com/valdiferreira/mining_discharging_greenhub

influência de parâmetros, configurações e processos (variáveis independentes), no tempo de consumo de energia (variável dependente) em determinadas condições de uso real, como o tipo de conexão escolhida e/ou qual aplicativo em uso, entre outras. Sendo assim podemos identificar padrões que elevam ou diminuem o gasto energético, além de comparar o desempenho de modelos regressores através de métricas de avaliação.

A base utilizada como fonte de dados conta com um aplicativo cliente que capta, com autorização do usuário, diversas informações relacionadas às configurações do dispositivo, aplicativos e processos do sistema. Esses dados são coletados cada vez que o nível de bateria varia em 1%, juntamente com a data e hora, com precisão de segundos, em que a variação ocorreu. Essas informações são referentes ao estado de diversas configurações, como status de conexões, GPS, modo de economia de bateria, uso da tela entre outras, além de processos referentes à aplicativos e serviços, que estejam rodando no momento da variação. Essas informações são registradas e enviadas, totalmente anônimas, para um servidor que as armazena em uma base de dados.

Para processar os dados coletados pelo BatteryHub, foi utilizada a versão do dataset que está disponível em formato CSV (*comma-separated values*). O dataset inclui três tipos de arquivos. Arquivos do primeiro tipo, chamados arquivos de **amostras** (samples), contém informações gerais, de diversas condições e configurações dos dispositivos relacionadas a cada aumento (carregamentos) ou diminuição (descarregamento) da energia na bateria de cada dispositivo com o aplicativo instalado, durante o uso. Cada observação nos arquivos de amostras é identificado por um id único e inclui o id do seu dispositivo no qual a observação foi coletada.

O segundo tipo de arquivo encontrado na base se refere a todos os **processos** que estavam sendo executados no momento em que cada alteração do nível de energia registrada ocorreu. Como era de se esperar em dispositivos com sistema operacional, a cada observação estão associados diversos processos, cada qual identificado por um id, com nome e/ou pacote ao qual pertence. Cada linha em um arquivo de processos inclui o id da observação à qual se refere. O último tipo de arquivo a ser mencionado se refere aos **dispositivos** participantes da base que são caracterizados pelo modelo e fábrica, além de um id que é referenciado por todas as observações nos arquivos de amostras que foram coletadas nesses dispositivos.

O primeiro passo foi obter toda a base disponível no portal do projeto GreenHub[2], organizada em diretórios de arquivos comprimidos: amostras (dataset-samples - 1,6 GB), processos (dataset-app_processes - 10,5 GB), e dispositivos (dataset-devices - 3,5 MB). Ao serem des-

[2] https://greenhubproject.org/

compactados, estes arquivos passam a ocupar 11.1 GB, 125,8 GB, e 17,6 MB respectivamente. Os registros foram gerados em ordem cronológica e estão assim distribuídos, sendo 165 arquivos de samples, 5.912 de processos e 2 de dispositivos.

3.2 CRIAÇÃO DO CONJUNTO ALVO

Os samples seguem uma estrutura parecida com a mostrada na Figura 12, onde são listados em sequência, com todas as informações relacionadas. Para análise do consumo de energia realizada, foram selecionadas apenas as amostras onde a bateria não tinha qualquer tipo de carregamento de bateria (seja por carregador elétrico ou conexão USB), para compor os intervalos usados na geração dos modelos.

Figura 12 – Exemplo de sample

battery state	battery level	network status	timestamp	bluetooth enabled	location enabled
Discharging	0.79	disconnected	2018-07-08 13:38:10	0	1
Charging	0.80	WIFI	2018-07-08 13:44:15	0	1
Discharging	0.79	disconnected	2018-07-08 13:49:53	0	1
Discharging	0.78	disconnected	2018-07-08 14:12:15	0	1

Fonte: (PEREIRA et al., 2021)

A seleção utilizou o campo *battery_state*, selecionando todas as entradas que continham o valor *Discharging* indicando que não havia conexão com fonte de energia externa, restringindo, assim, possíveis influências destas, conforme a Tabela 8. O tipo e os valores originais do campo id foram trocados por valores genéricos para permitir melhor compreensão.

Para fins deste estudo, as sequências de samples foram organizadas em pares consecutivos

Tabela 8 – Samples de Discharging do dispositivo 15040

id	timestamp	battery_state	battery_level	timezone	country_code
S1	2018-09-06 09:46:02	Discharging	100.0	America/Manaus	br
S2	2018-09-06 09:47:00	Discharging	100.0	America/Manaus	br
S3	2018-09-06 09:51:01	Discharging	99.0	America/Manaus	br
S4	2018-09-06 10:04:03	Discharging	98.0	America/Manaus	br
S5	2018-09-06 10:22:06	Discharging	97.0	America/Manaus	br

Fonte: (AUTOR, 2022)

Tabela 9 – Tempo de consumo samples - dispositivo 15040

id	battery_state	battery_level	timezone	country_code	consume_time
S2	Discharging	100	America/Manaus	br	241
S3	Discharging	99	America/Manaus	br	782
S4	Discharging	98	America/Manaus	br	1083

Fonte: (AUTOR, 2022)

de descarga, pertencentes a um mesmo dispositivo, permitindo identificar a variação de tempo a cada vez que 1% de bateria era utilizado. Os tempos de origem de cada amostra se encontram registrados no formato "YYYY-MM-dd HH:MM:SS" (ex: "2020-02-03 10:35:03"), e a duração entre eles foi obtido pela diferença entre os tempos (t2-t1) dos pares, permitindo obter o intervalo com precisão de segundos, sendo este atribuído a amostra correspondente a t1, como sua variável dependente.

Utilizando o exemplo da Tabela 8, vimos que apesar do primeiro sample apresentar o valor de Discharging no campo de estado da bateria, ele não possui um campo subsequente com uma diferença de 1% estado, portanto não sendo considerado para fins deste estudo. Sendo os samples subsequentes válidos para criação da variável alvo, podemos tomar como exemplo S2 e S3 e seus campos timestamp como t2 e t3. A partir disso, é calculada a diferença entre t3 (2018-09-06 09:51:01) e t2 (2018-09-06 09:47:00), obtendo no exemplo, o resultado de 241 segundos, atribuído a S2 como sendo sua variável dependente, como sendo o tempo que levou para consumir 1% de bateria sendo o mesmo é feito para S3 e S4 resultando na Tabela 9.

3.3 LIMPEZA DE DADOS E PRÉ-PROCESSAMENTO

Para manipulação dos dados, foi utilizada a biblioteca Pandas[3] v1.3.4, por apresentar versatilidade e simplicidade, bem como diversos recursos úteis ao propósito, como leitura e escritas de arquivos CSV, incluindo a leitura em blocos, útil para processar arquivos muito grandes, execução de funções nos dados, realização de consultas, entre outros.

As etapas de preprocessamento foram planejadas para aproveitar a arquitetura dos arquivos, criando uma organização lógica que melhor atendesse às necessidades da pesquisa através de paralelismo, utilizando o objeto Pool da biblioteca multiprocessing[4] do Python. Sendo assim,

[3] https://pandas.pydata.org/
[4] https://docs.python.org/3/library/multiprocessing.html

o tempo de seleção, organização e transformação dos dados foi reduzido drasticamente, e os recursos computacionais limitados, frente ao volume de dados, puderam ser melhor utilizados.

Em seguida, os arquivos de samples foram agrupados por dispositivos, e gravados em arquivos próprios, um para cada dispositivo, sendo utilizado multiprocessamento, para a seleção dessas informações através dos arquivos. A cada arquivo eram identificados os dispositivos e colocados em uma lista a qual o pool de processos teria acesso. Cada processo retirava um identificador da lista e percorria todos os arquivos selecionando os registros correspondentes, para compor um arquivo relacionado exclusivamente com este dispositivo. Caso fosse detectado que o arquivo do mesmo já havia sido gerado, o processo passava para o próximo, até que todos tivessem seus samples selecionados e separados.

Essa organização facilitou criação da variável alvo, intervalos da variação de dois níveis distintos de bateria, a ordenação e identificação de sequências de consumo, a verificação das etapas, o agrupamento de informações, e possibilitou a maior flexibilização dos critérios de análise, que podem ser feitos de maneira individual, pareada, por modelo e etc. Para obter os intervalos de consumo, as amostras organizadas por dispositivo foram ordenadas cronologicamente utilizando os tempos (t) em que foram gerados pelo aplicativo em cada dispositivo, permitindo identificar sequências de decaimento constituídos por observações onde os nível de bateria diminui em 1% em relação ao anterior.

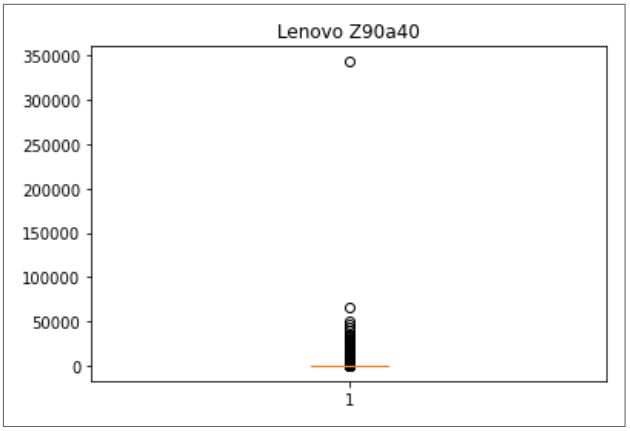
Originalmente, cada linha do arquivo de processos representava um processo (serviço, aplicativo ou outra rotina do sistema) em execução em um dispositivo no momento em que uma observação foi coletada. Além do próprio id, os processos também são identificados por uma chave estrangeira (sample_id), que foi usada para agrupá-los segundo suas amostras de origem. Os processos foram então organizados por dispositivo, utilizando os identificadores dos arquivos de amostras por dispositivo, em arquivos de processos por dispositivos. Essa organização é importante para manter as informações coesas sob a mesma perspectiva (a dos dispositivos), e para que a cada etapa e seus resultados sejam verificados.

3.4 REDUÇÃO DOS DADOS E PROJEÇÃO

Como dito anteriormente, os dados utilizados constituem os samples de descarga das baterias, que representam 33% das amostras. Os arquivos de dispositivos, foram agrupados em arquivos de modelos de dispositivos, visando manter as particularidades dos hardwares o mais agrupadas possível.

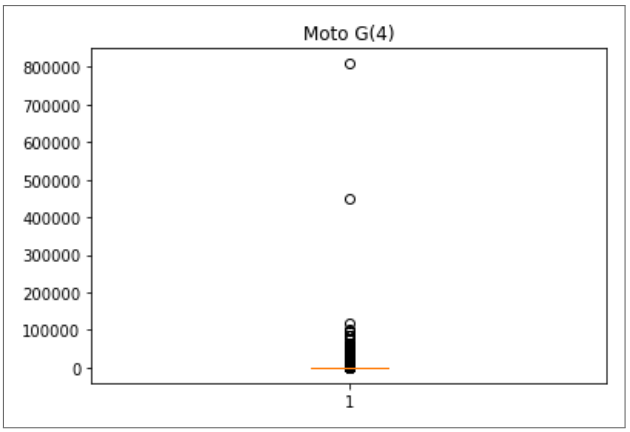
Houve também a necessidade de identificar outliers na nossa variável alvo, o tempo de consumo, sendo para isso utilizado o método de Boxplot Ajustado (*Adjusted Bloxplot*) para dados que apresentam curvas não-normais, como o caso da nossa variável alvo, na detecção de eliminação de outliers (HUBERT; VANDERVIEREN, 2008). O método ajusta os dados de acordo com a assimetria de uma distribuição, baseada na diferença mediana em escala da metade esquerda e direita da distribuição, portanto, não se baseando no terceiro momento como a assimetria clássica (BRYS; HUBERT; STRUYF, 2004). Nas Figuras 13, 14 e 15 um exemplo dos dados antes de serem submetidos ao ajuste e nas Figuras 16, 17 e 18 o resultado da transformação.

Figura 13 – Boxplot - Lenovo Z90a40



Fonte: (AUTOR, 2022)

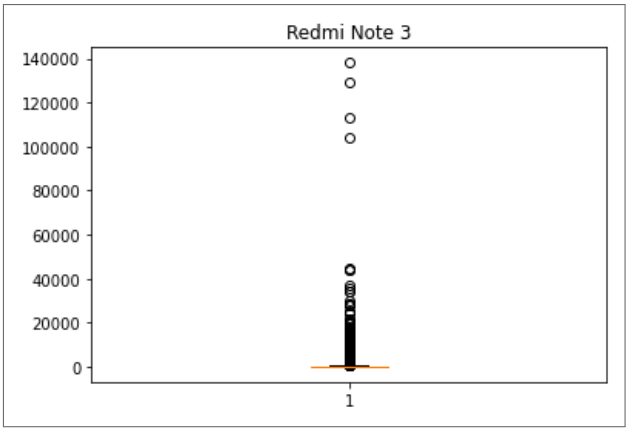
Figura 14 – Boxplot - Moto G4



Fonte: (AUTOR, 2022)

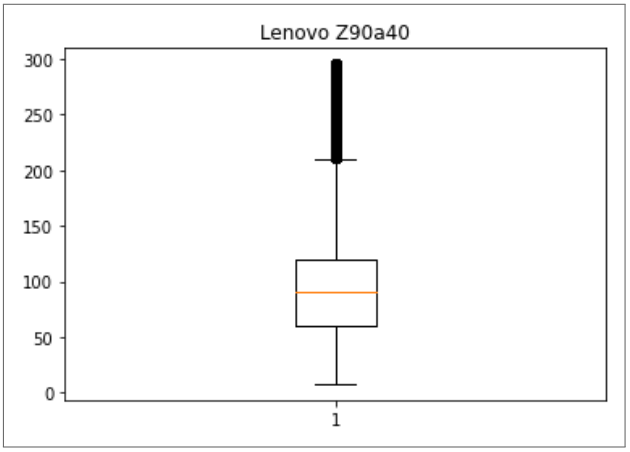
Selecionar os processos e modelos de dispositivos que seriam utilizados no estudo também foi necessário, para diminuir a esparsidade dos dados e tornar o estudo viável diante dos recursos computacionais disponíveis. Por isso, foram usados os 25 processos do sistema e

Figura 15 – Boxplot - Redmi Note 3



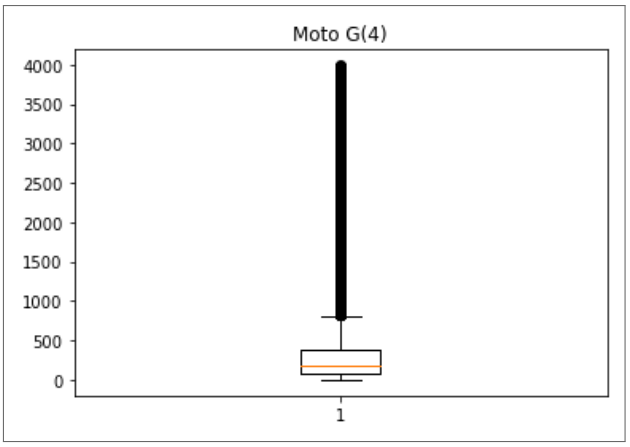
Fonte: (AUTOR, 2022)

Figura 16 – Boxplot Ajustado - Lenovo Z90a40



Fonte: (AUTOR, 2022)

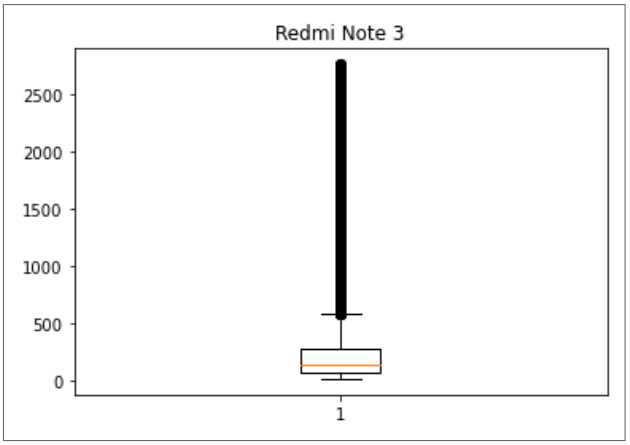
Figura 17 – Boxplot Ajustado - Moto G4



Fonte: (AUTOR, 2022)

os 25 processos de aplicativos mais populosos (relacionados ao maior número de samples) foram utilizados e os 100 modelos com melhores resultados iniciais de regressão segundo a porcentagem de média de erros absolutos, para cada algoritmo desenvolvido nesse estudo.

Figura 18 – Boxplot Ajustado - Redmi Note 3



Fonte: (AUTOR, 2022)

Tabela 10 – Armazenamento samples - dispositivo 15040

id	free	total	free_system	total_system	consume_time
S2	9693.0	24425.0	1502.0	3905.0	241
S3	9693.0	24425.0	1502.0	3905.0	782
S4	9693.0	24425.0	1502.0	3905.0	1083

Fonte: (AUTOR, 2022)

A fim de identificar com mais precisão o significado de algumas features, e identificar oportunidades de tornar os dados mais compreensíveis e relevantes para o modelo e interessados, foi realizada uma pesquisa de como as informações são extraídas pelo BatteryHub. Algumas dessas informações foram retiradas diretamente do arquivo proc.txt do kernel do Android (BOWDEN et al.,), processadas e inseridas no GreenHub. São dados sobre armazenamento (Tabela 10) e memória (Tabela 11).

A tabela de armazenamento, contém dados referentes a capacidade de armazenamento do dispositivo separadas por uso. As colunas free e total, se referem a memória interna do dispositivo usada pelo usuário para arquivos (fotos, documentos e etc) e os aplicativos instalados por ele. enquanto total_system e free_system é o total e que restou disponível da memória usada pelo sistema operacional e aplicativos nativos. Elas foram então construídas da seguinte maneira:

- user_storage_busy_percent: A porcentagem de espaço da memória interna ocupada pelo usuário
- system_size_partition_busy: O tamanho do espaço ocupado pelo sistema e aplicativos

Tabela 11 – Memória samples - dispositivo 15040

id	memory_active	memory_inactive	memory_free	memory_user	consume_time
S2	1640	1397900	2947584	143024	241
S3	1640	1361804	2947584	184240	782
S4	1664	1409752	2947584	108080	1083

Fonte: (AUTOR, 2022)

Tabela 12 – Dados de Network - dispositivo 15040

id	network_status	network_type	mobile_network_type	wifi_status
S1	WIFI	WIFI	hspa	enabled
S2	disconnected	WIFI	hspa	enabled
S3	WIFI	WIFI	hspa	enabled

Fonte: (AUTOR, 2022)

nativos

A ocupação da memória RAM também foi estimada baseada nos dados fornecidos pelo GreenHub, seguindo as especificações da documentação, conforme Tabela 11 que as descreve da seguinte maneira:

- memory_active: Memória utilizada que não pode ser reivindicada, a menos que seja extremamente necessário.
- memory_inactive: Memória recém utilizada que pode ser reivindicada para outros propósitos.
- memory_free: O tamanho total da memória
- memory_user: Informação da memória extraída de MemFree, sendo esta a soma da HighFree e da LowFree.

Para fins deste trabalho, foi criado o campo ram_busy_percent, considerando a porcentagem da memory_free ocupada pela soma da memory_active com a memory_inactive.

Durante a exploração dos dados foi notado que os dados de conexão, conforme exemplo da Tabela 12, apresentavam detalhamentos que estavam em função de um determinado campo principal, o network_status. Sendo assim com o objetivo de reduzir um pouco o espaço amos-

Tabela 13 – Exemplo de Arquivo de Processos

sample_id	id	name	application_label	is_system_app
S1	49336526	com.sec.android.app.soundalive	SoundAlive	1
S1	49336526	com.android.phone	CSC	1
S2	49336526	com.sec.android.inputmethod	Teclado Samsung	1
S3	49336526	com.facebook.katana	Facebook	0

Fonte: (AUTOR, 2022)

tral, e reduzir assim a necessidade de computação, apenas ele foi considerado no conjunto final de dados.

A maioria das features têm nomes autoexplicativos, porém faz se necessário comentários adicionais sobre algumas features:

- up_time: tempo total desde o último boot, incluindo tempo de de baixíssimas atividades do sistema, como deep sleep.
- sleep_time: tempo total, desde o último boot, que o aplicativo entrou no modo deep sleep, estado onde o processador está na mais baixa frequência.
- voltage: unidade de medida em volts.
- temperature: unidade de medida em graus centígrados.
- usage: porcentagem de uso da CPU
- health: espécie de auto diagnóstico da bateria oferecido pelo Android

Os processos inicialmente constituíam arquivos com uma lista imensa de processos (entre aplicativos e processos do sistema) conforme a Tabela 13. Cada linha continha um processo, identificado por nome, pacote, versão e se uma flag binária (is_system_app)que indicava se era nativo, ou não do sistema.

Esses processos foram agrupados por samples, cada um deles se tornou uma coluna binária, que indicava ou não sua presença naquele sample à medida que outros samples com outros processos agrupados eram adicionados. Novos processos (ou aplicativos) são adicionados adicionados como novas colunas, enquanto a presença ou não dos já existentes, era indicada na sua respectiva coluna, conforme o exemplo na Tabela 14. A coluna is_system_app foi incorporada ao final do nome da coluna, permitindo assim, sua diferenciação.

Tabela 14 – Processos nos samples

id	SoundAlive_1	CSC_1	Teclado Samsung_1	Facebook_0	Fotos_1
S1	1	1	0	0	0
S2	0	0	1	0	0
S3	0	0	0	1	0

Fonte: (AUTOR, 2022)

Essa organização de samples e processos, facilitou sua agregação em função dos dispositivos através das chaves id e sample_id, respectivamente, sendo assim, cada um passou destes passou a ter um arquivo próprio, com todas as informações de configurações consideradas e processos e aplicativos mais populares da base. Em seguida, foram agrupados em função de seus modelos, a fim de isolar o máximo possível particularidades, não presente nos dados, que pudessem interferir nas predições.

3.5 ESCOLHA DO MÉTODO DE MINERAÇÃO

Todos os métodos escolhidos para esse estudo (Decision Tree, Random Forest, XGboost) são baseados em árvores devido a sua versatilidade dessa estrutura em lidar com diferentes tipo de dados (categóricos, binários e numéricos), necessitarem de pouca (ou nenhuma) transformações nos dados (como normalização), e serem utilizados em conjunto com diversas técnicas, produzindo resultados e níveis de interpretações e predições diferentes.

Os passos seguintes, Análise e modelagem exploratória, Mineração, Interpretação dos padrões minerados e Ação sobre conhecimento descoberto, serão melhor explicados no capítulo seguinte, por apresentarem características mais experimentais. Além disso, são apresentados os resultados e realizada uma comparação entre as técnicas de regressão a partir deles.

4 EXPLORAÇÃO E RESULTADOS

4.1 ANÁLISE E MODELAGEM EXPLORATÓRIA

Tendo sido determinado todo o conjunto de dados habilitados para o estudo, foi necessário tornar computacionalmente viável os experimentos, tendo em vista que, segundo os critérios usados neste estudo, foram gerados dados válidos de mais de 24 mil dispositivos distribuídos em mais de 2 mil modelos, com milhares de processos e aplicativos, o que demonstra a riqueza e o potencial a ser explorado nos dados do GreenHub.

Os algoritmos de regressão escolhidos Decision Tree[1] e Random Forest[2] do scikit-learn, e o XGBoost[3] foram executados utilizando scripts da linguagem Python. O objetivo inicial era processar todas entradas dos modelos de dispositivos da base e, por esse motivo, a divisão entre os conjuntos de validação, treino e teste, a estimativa dos outliers e toda a fase de pré-processamento foi feita utilizando todos os samples dos modelos de dispositivos. Entretanto, com o objetivo de tornar a exploração possível, dentro das limitações de processamento, as visões dos dados para os experimentos foi elaborada da seguinte forma:

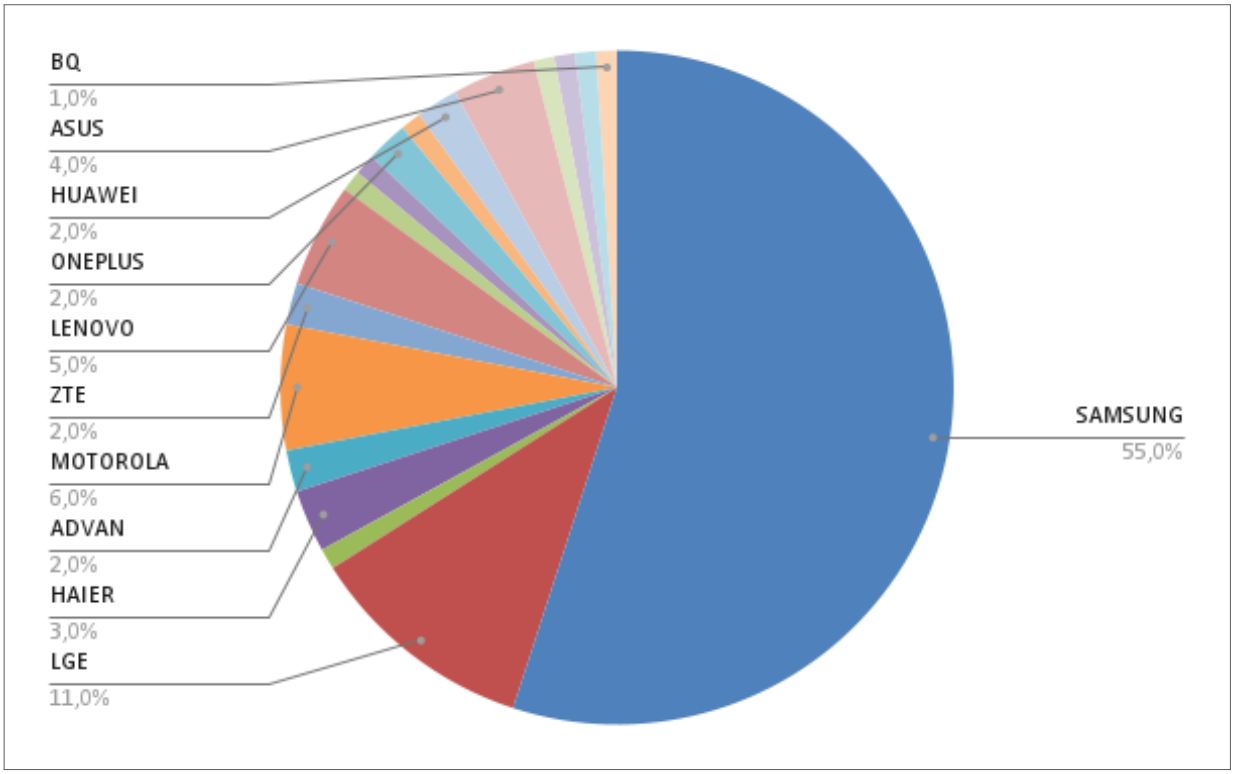
Critério para seleção dos dados:

- Dados provenientes dos 100 modelos de dispositivo mais populares da base (com maior quantidade de samples relacionados
- Dados relacionados aos 25 aplicativos mais populares da base (presentes na maior quantidade de samples)
- Dados relacionados aos 25 processos do sistema mais populares da base (presentes na maior quantidade de samples)
- os samples foram ordenados cronologicamente
- os dados de cada modelo de dispositivo foram divididos, em conjunto de treino e testes, na proporção de 0.8 e 0.2 respectivamente
- amostragem aleatória dos conjuntos (segundo parâmetro random=1, da função sample da biblioteca Pandas)

[1] https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html
[2] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html
[3] https://xgboost.readthedocs.io/en/stable/python/python_intro.html

Seguindo a divisão de conjunto de validação, treino e teste, conforme definido por Ripley (2007), foram amostrados 1500, 10000 e 2500 entradas respectivamente, para cada modelo de dispositivo, totalizando 150 mil linhas para ajuste, 1 milhão de linhas para treino e 250 mil linhas para teste. Os dados de treino e teste retirados dessa maneira estão caracterizados de acordo com fabricante (Figura 19, fuso horário (Figura 20 e Figura 21) e versões do sistema operacional (Figura 22 e Figura 23). As figuras mostram que as amostras coletadas refletem a heterogeneidade do contexto Android contida no GreenHub.

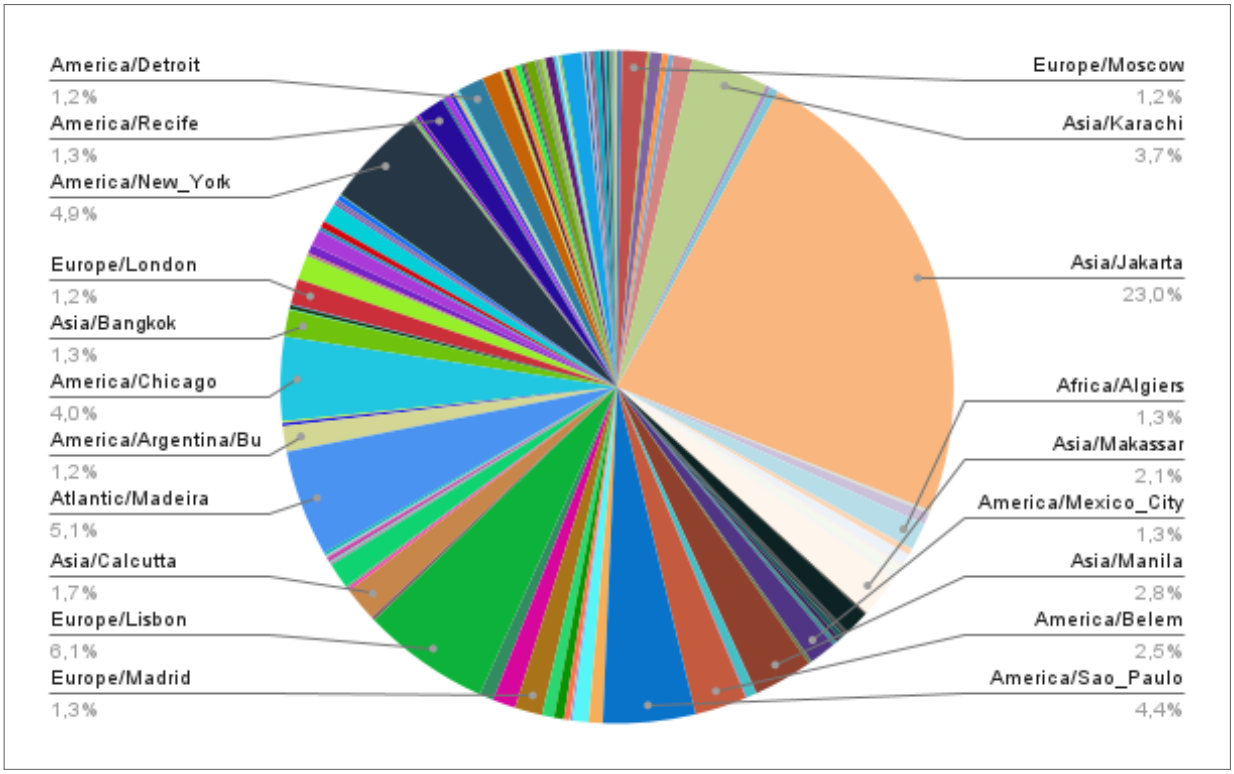
Figura 19 – Distribuição de Fabricantes de Treino e Teste



Fonte: (AUTOR, 2022)

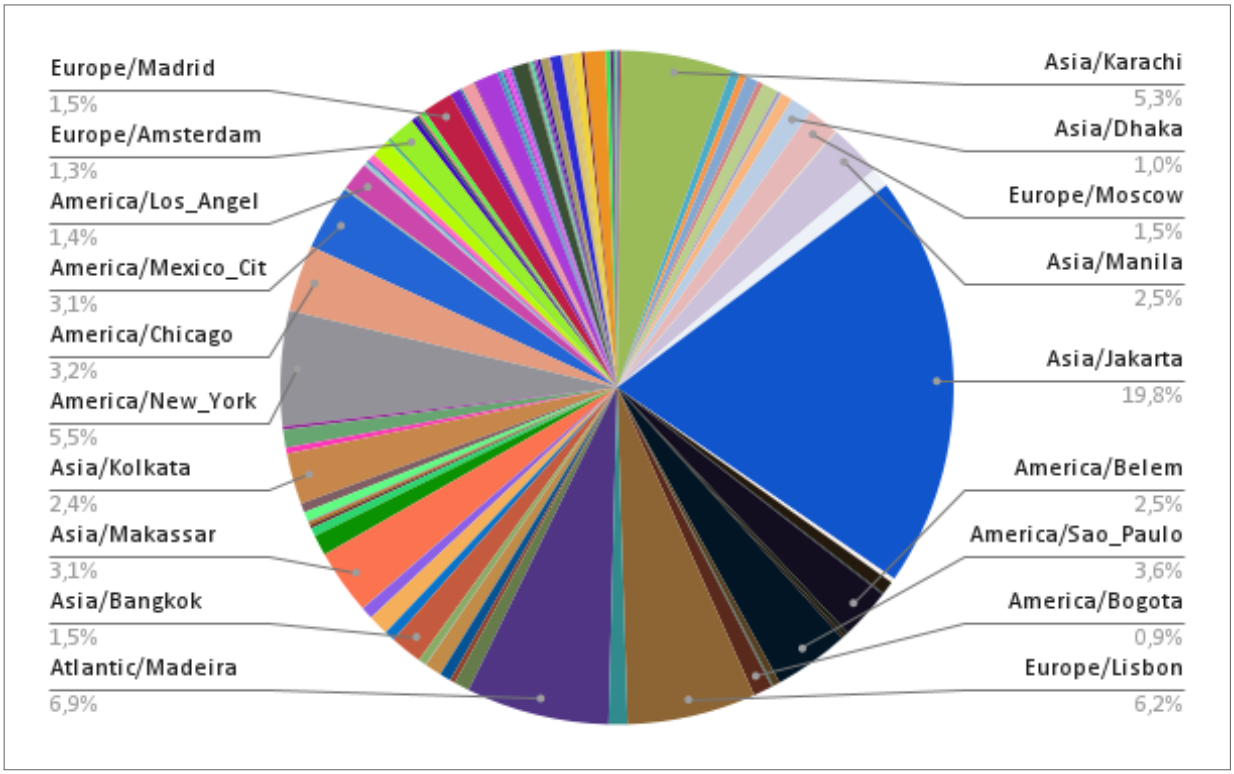
Nesta etapa, os regressores correspondentes aos 100 modelos de dispositivos mais populares tiveram seus meta-parâmetros otimizados, através de uma técnica chamada Random Search (BERGSTRA; BENGIO, 2012). Assim como outras semelhantes, essa estratégia envolve o estabelecimento de um espaço de busca, volume onde cada dimensão representa um meta-parâmetro (ou hiper-parâmetro) e cada ponto é um vetor com um valor para cada parâmetro. O objetivo é encontrar o vetor que torne o algoritmo mais preciso, minimizando os erros. Ela se diferencia por testar e comparar um número limitado de combinações aleatórias, o que possibilita encontrar valores ótimos com um custo computacional menor, em relação a outras técnicas.

Figura 20 – Distribuição do fuso horário do Conjunto de Treino



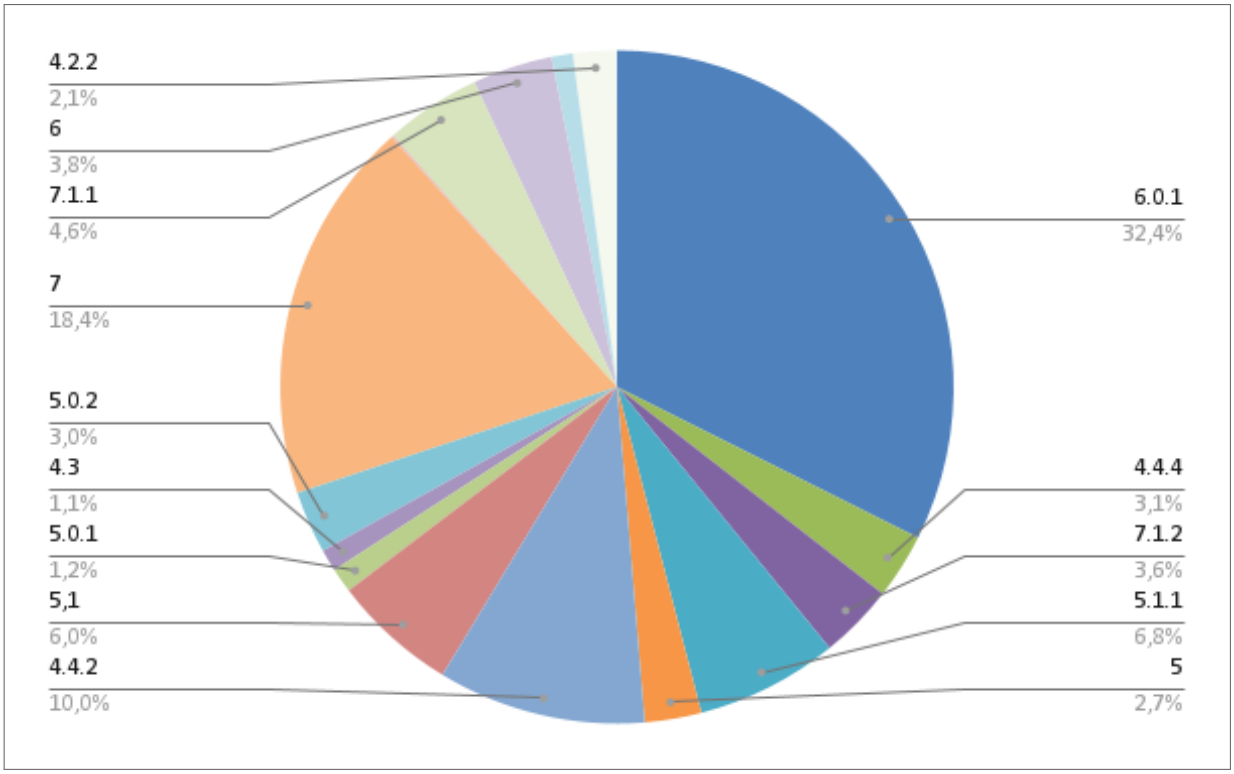
Fonte: (AUTOR, 2022)

Figura 21 – Distribuição do fuso horário do Conjunto de Teste



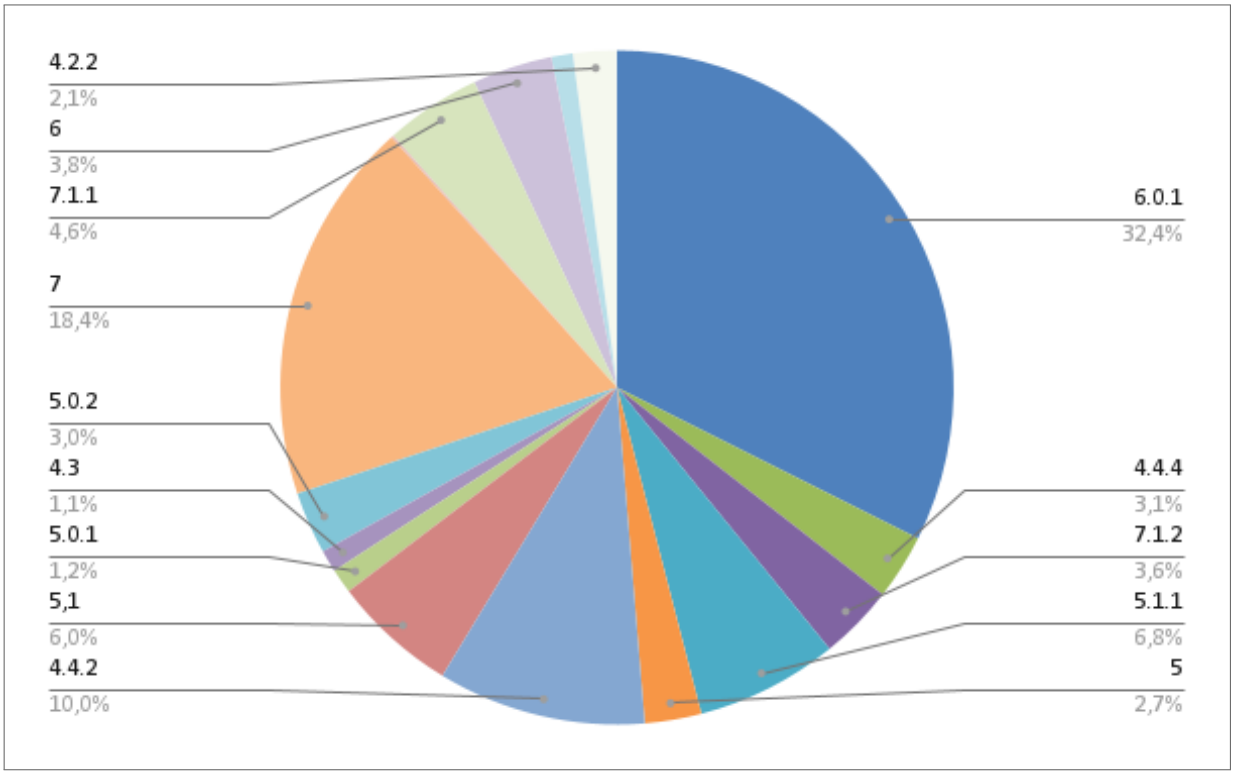
Fonte: (AUTOR, 2022)

Figura 22 – Distribuição de Versões do Sistema Operacional do Conjunto de Treino



Fonte: (AUTOR, 2022)

Figura 23 – Distribuição de Versões do Sistema Operacional do Conjunto de Teste



Fonte: (AUTOR, 2022)

Tabela 15 – Meta-paramêtros ajustados - Decision Tree

Atributo	Valores
splitter	best, random
max_deapth	{5, 6, 7, 8, 9, 10}
min_samples_split	{2, 3, 4, 5, 6, 7, 8, 9, 10}
min_samples_leaf	{2, 3, 4, 5, 6, 7, 8, 9, 10}
min_impurity_decrease	{0 , 0.5, 1}
criterion	{"mse", "mae"}

Fonte: (AUTOR, 2022)

Tabela 16 – Meta-paramêtros ajustados - Random Forest

Atributo	Valores
n_estimatores	{50, 100, 150, 250, 300}
max_depth	{ 5, 6, 7, 8, 9, 10}
max_features	{auto', 'sqrt', 'log2'}
criterion	{mae', 'mse'}
min_samples_split	{ 2, 3, 4, 5, 6, 7, 8, 9, 10}
min_impurity_decrease	{0, 0.05, 0.1}
bootstrap	{True, False}

Fonte: (AUTOR, 2022)

A fim de comparar de maneira mais próxima o desempenho dos regressores, e equilibrar a exigência computacional para os experimentos, foram utilizados os mesmos valores para conjuntos de meta-parâmetros semelhantes entre os regressores conforme as Tabelas 15, 16 e 17. Com o objetivo de melhorar o desempenho das predições para investigá-las e tornar viável computacionalmente o estudo, para cada um dos 100 modelos de dispositivo foi gerado um modelo de regressão de cada um dos algoritmos estudados neste trabalho.

As opções utilizadas para ajustes dos algoritmos de regressão se encontram descritas nas tabelas abaixo e foram determinadas de acordo com as características dos dados e quantidade dos dados selecionados, além da viabilidade computacional. As opções escolhidas, bem como o conjunto de valores explorado, pode variar de acordo com o objetivo de cada estudo, sendo, geralmente, de caráter fundamentalmente empírico, e estão listadas na Tabela 15, Tabela 16 e na Tabela 17.

Dentro do espaço amostral de metaparâmetros utilizado foram feitas 60 iterações(combinações), número suficiente para achar pelo menos uma das combinações ótimas (WEIRAN, 2019). Para

Tabela 17 – Meta-paramêtros ajustados - XGBoost

Atributo	Valores
n_estimatores	{50, 100, 150, 250, 300}
max_depth	{ 5, 6, 7, 8, 9, 10}
min_child_weight	{ 2, 3, 4, 5, 6, 7, 8, 9, 10}
objective	{eg:squarederror', 'reg:squaredlogerror'}
eta	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6}
sub_sample	{0.5, 0.6, 0.7, 0.8, 0.9, 1}

Fonte: (AUTOR, 2022)

qualquer distribuição em um espaço amostral com um máximo finito, o máximo de 60 observações aleatórias está dentro dos 5% superiores do máximo verdadeiro, com 95% de probabilidade $(1-0,05)^n$. Se a região próxima da ótima de metaparâmetros ocupa pelo menos 5% da superfície da busca, então 60 tentativas aleatórias encontrarão essa região com alta probabilidade (ZHENG, 2015).

4.2 MINERAÇÃO

Os algoritmos foram avaliados por modelo de dispositivo, sendo utilizada a porcentagem média do erro absoluto (Mean absolute percentage error - MAPE) dada pela equação:

$$MAPE = \frac{100\%}{n} \sum_{i=1}^{n} \left| \frac{A_i - F_i}{A_i} \right|$$

onde:

- n: n
- A_i: Valor real
- F_i: Valor previsto
- $\sum$: somatório

e a média de erro absoluto médio (Mean Absolute Error - MAE) dada pela equação:

$$MAE = \frac{1}{n} \sum_{t=1}^{n} |A_i - F_i|$$

por serem menos sensíveis a *outliers* que outras métricas. Os resultados onde cada um dos três algoritmos usados teve seus melhores resultados, encontram-se descritos nas seções

seguintes. Também estão descritos a média do conjunto teste, o desvio padrão, e os valores máximos e mínimos das amostras. Todos os algoritmos e demais códigos usados para obtenção dos resultados da exploração foram escritos na linguagem Python, utilizando as bibliotecas de regressão Decision Tree[4] e Random Forest[5] do scikit-learn[6] e XGBoost[7], e da abordagem SHAP[8] utilizando a ferramenta TreeExplainer da biblioteca, e estão localizados no repositório do projeto[9] assim como todos os resultados apresentados ou discutidos nas seções seguintes.

4.3 INTERPRETAÇÃO DOS PADRÕES MINERADOS

Essa seção tem como objetivo destacar e comentar as descobertas mais relevantes de todo o ciclo KDD realizado neste trabalho, primeiramente analisando o desempenho dos algoritmos de regressão em relação a precisão das predições e o uso associado ao TreeExplainer para precisão descritiva, considerando a interpretação global dos resultados.

Dentro de cada previsão a contribuição de cada variável independente na diferença entre a previsão real e a previsão média é o valor estimado de Shapley, medido pelo TreeExplainer, sendo este o impacto dos fatores nos modelos. Todos os resultados estão acompanhados do valor base das predições (base_vale) para efeito de dimensão. Todos os valores estão em função de segundos, unidade de medida da variável alvo.

As imagens contendo o sumário dos impactos desta seção mostram a relação das com a média em cada previsão realizada. Os valores mais altos dos fatores apresentam o tom mais avermelhado, os mais baixos em azul e os intermediários em degradê, mas semelhante ao roxo. As variáveis independentes com valores binários apresentam apenas as cores vermelho e azul, indicando presença ou não naquela previsão, respectivamente. Os valores contidos nessas imagens indicam a contribuição em segundos, unidade de medida da variável alvo deste trabalho, dessa variável sobre a diferença entre a média das previsões e a previsão realizada nesse ponto. As demais fazem comparação de como o modelo muda sua percepção em relação aos a presença de um dos fatores.

[4] https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html
[5] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
[6] https://scikit-learn.org/stable/index.html
[7] https://xgboost.readthedocs.io/en/stable/python/python_intro.html
[8] https://github.com/slundberg/shap
[9] https://github.com/valdiferreira/mining$_d ischarging_g reenhub$

4.3.1 Algoritmos baseados em árvores

Nesta seção são apresentados uma parte dos resultados para discussão. O desempenho preditivo dos algoritmos foi comparado utilizando a métrica MAPE, e se encontram aqui representados por aqueles com obtiveram menos de 50% nos resultados. Os modelos de dispositivo estão distribuídos pelo algoritmo que obteve o melhor desempenho em seus dados, conforme as Tabelas 18, 19 e 20.

Tabela 18 – Decision Tree - Modelos de dispositivo

Model	MAPE	MAE	Média	Desvio	Mínimo	Máximo
POLYTRON R2407	11,21%	25,96	56,46	141,48	29	1823
SM-J111F	32,96%	148,22	386,07	292,77	95	1484
SAMSUNG-SM-G935A	33,93%	59,69	159,95	114,90	56	850
SM-J500M	34,68%	119,49	294,75	248,76	67	1583
SAMSUNG-SM-G925A	35,61%	229,02	356,82	615,82	73	5656
SM-G920F	37,24%	65,18	156,69	150,98	50	1210
SM-G355H	38,61%	150,30	249,50	496,02	42	3908
GT-I9500	38,75%	213,55	328,17	751,22	31	6016
SM-G900V	40,96%	79,43	173,57	215,03	34	2644
XT1080	42,92%	100,00	227,19	245,59	64	1699
SM-G532F	44,71%	127,42	303,11	252,39	66	1604
SM-G130H	46,66%	41,63	113,77	110,01	10	863
SM-G900F	47,35%	79,59	151,28	250,11	37	2542
ASUS_T00J	47,65%	80,78	147,24	191,98	23	1224

Fonte: (AUTOR, 2022)

O resultado dessas tabelas mostra um certo equilíbrio entre as técnicas, tendo o XGBoost obtido melhor resultado em 42 modelos de dispositivos, o Random Forest em 28 e o Decision Tree em 30, o que mostra que embora o Random Forest seja considerado uma evolução do Decision Tree, este último ainda deve ser considerado, tendo em vista que pode obter bons resultados, frente à um processamento muito menor. O desempenho descritivo observado pela quantidade de valores diferentes de zero que a abordagem SHAP, através do TreeExplainer, foi capaz de capturar. Em média, cada modelo de dispositivo teve a contribuição de 63 variáveis independentes detectadas pela abordagem, entre configurações, processos do sistema e aplicativos.

Embora tenha obtido o melhor resultado geral, o desempenho do XGBoost, foi o menor

Tabela 19 – Random Forest - Modelos de dispositivo

Model	MAPE	MAE	Média	Desvio	Mínimo	Máximo
A75	9,65%	18,41	48,56	87,50	30	1248
SM-G925T	15,47%	16,42	112,17	37,26	87	4272
SM-S727VL	25,90%	136,81	355,96	339,21	20	669
i5E	27,12%	129,19	325,49	277,75	23	635
HUAWEI Y541-U02	33,81%	36,24	69,48	86,17	122	2150
SM-G610M	34,37%	134,83	383,72	227,20	35	812
SM-G532G	34,54%	161,16	374,83	405,58	14	134
QMobile i2 PRO	35,54%	144,76	178,02	393,43	114	7668
SM-J510FN	37,14%	275,44	486,73	846,10	10	697
SM-G532M	37,30%	170,51	361,48	376,22	33	1734
SM-J250F	37,39%	83,34	255,59	127,31	23	5500
SM-N916L	39,91%	18,29	63,30	24,06	30	640
LG-M250	40,78%	118,25	288,05	267,92	107	3769
SM-G530H	43,72%	143,06	239,05	347,24	71	838
ASUS_T00F	44,00%	148,92	221,65	502,42	16	181
SM-G920V	46,05%	80,55	164,94	148,52	13	2511
SM-G610F	49,31%	185,14	386,07	432,63	57	4482

Fonte: (AUTOR, 2022)

Tabela 20 – XGboost - Modelos de dispositivos

Model	MAPE	MAE	Média	Desvio	Mínimo	Máximo
SM-N920K	29,55%	25,40	121,37	47,29	9	242
Lenovo A2020a40	29,55%	139,06	167,39	204,50	1	1350
SM-G935F	30,02%	159,35	284,25	473,14	114	6759
GT-I9301I	30,02%	281,80	359,66	752,92	3	14446
SM-G925F	34,21%	183,42	287,61	461,11	68	3938
ONE E1003	34,21%	534,00	637,96	863,39	1	9867
SM-N910F	35,74%	217,20	324,47	624,93	64	7235
GT-I9192	35,74%	461,60	483,03	1362,98	1	23279
SM-J105B	37,78%	293,00	417,17	740,58	111	5300
LG-M150	37,78%	112,38	151,24	205,20	1	1460
Moto E (4)	44,23%	211,18	332,57	399,54	75	4350
SM-G930F	46,28%	502,69	622,07	1165,87	82	10626
SM-N950F	46,30%	333,20	457,06	646,83	83	8999

Fonte: (AUTOR, 2022)

na explicabilidade quando submetido ao TreeExplainer. Em média, das 63 variáveis apenas 17 tiveram valores não nulos computados. Isso não significa que as variáveis independentes não foram consideradas, mas que o conjunto pode precisar de ajustes adicionais para aumentar seu desempenho nesse quesito, pela dificuldade em capturar essas informações conforme apontado por Dunn, Mingardi e Zhuo (2021). Em relação ao Decision Tree, o TreeExplainer conseguiu capturar em média 21 variáveis independentes, novamente demonstrando a relevância de se considerar algoritmos mais simples nos estudos. Por fim, o Random Forest foi o que obteve melhor desempenho explicativo, tendo em média contribuições de 46 variáveis quantificadas pelo TreeExplainer.

Numa comparação direta, foi considerado o modelo SM-G935 onde o XGBoost obteve melhor precisão preditiva segundo a MAPE (30%), mas apenas a variável tela foi capturada pelo TreeExplainer de um total de 75. O mesmo modelo de dispositivo gerou 51,5% de precisão na sua avaliação com o Random Forest, tendo 61 variáveis computadas, enquanto o Decision Tree obteve 57,5%, com 36 do total de 75.

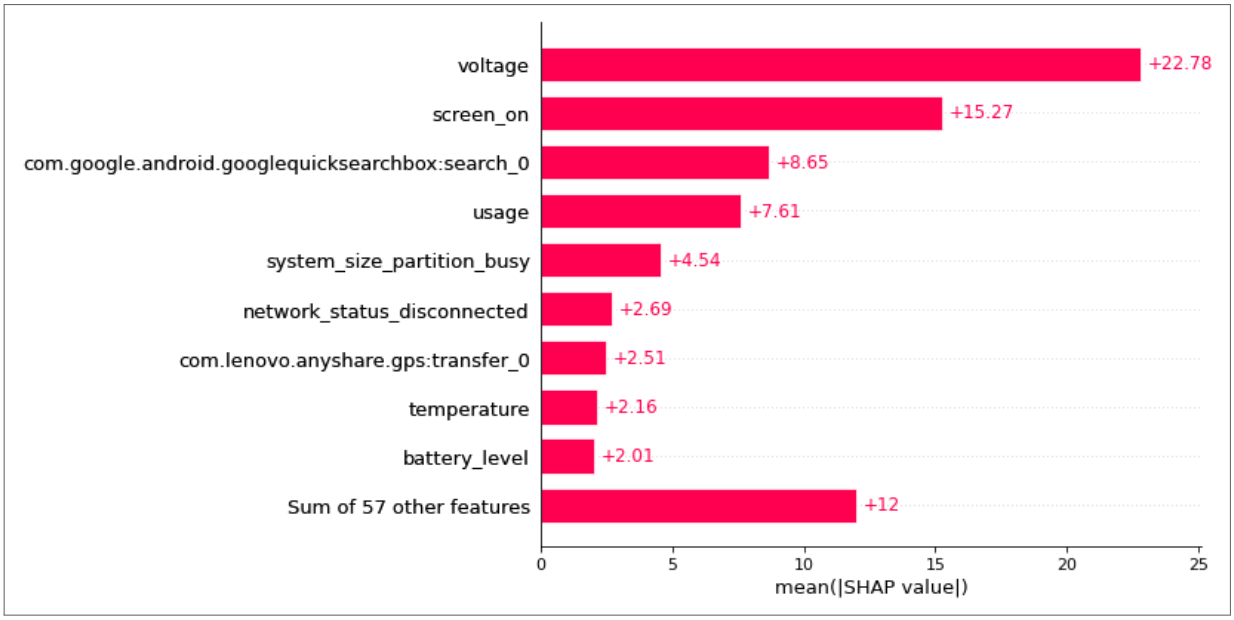
Para comparação de como os modelos foram compreendidos pelo TreeExplainer foi selecionado o modelo SM-N910H, por apresentar uma boa quantidade de variáveis independentes com contribuições. As Figuras 24, 25 e 26 mostram como os algoritmos consideram as variáveis independentes para construção dos modelos. Entre as variáveis independentes de maior impacto, aparecem as três variáveis independentes relacionadas a sensores de hardware: voltagem da bateria (voltage), temperatura (temperature) e porcentagem de uso da CPU (usage), indicando o maior peso de questões relacionadas ao hardware nesse modelo de dispositivo.

A voltagem estar em destaque pode indicar problemas de alto consumo de bateria nesse modelo ligados ao envelhecimento, pois o SM-N910H (Samsung Galaxy Note 4) é um modelo antigo, lançado no final de 2014. Além disso, o aplicativo de compartilhamento de arquivos SHAREit (sharei_0 e com.lenovo.anyshare.gps_0) foi avaliado pelo Random Forest e pelo XGBoost, juntamente com o TreeExplainer, como o aplicativo com maior impacto no tempo de consumo de energia.

As Figuras 27 e 28 mostram como a presença desse processo contribui na diferença entre a média das previsões e valor real de cada uma delas. Os pontos em vermelho indicam um maior valor da variável (nesse caso 1, o aplicativo está sendo executado) e os azuis indicam um menor valor (nesse caso 0, o aplicativo não está sendo executado) e os valores indicam de quanto foi essa contribuição, em segundos.

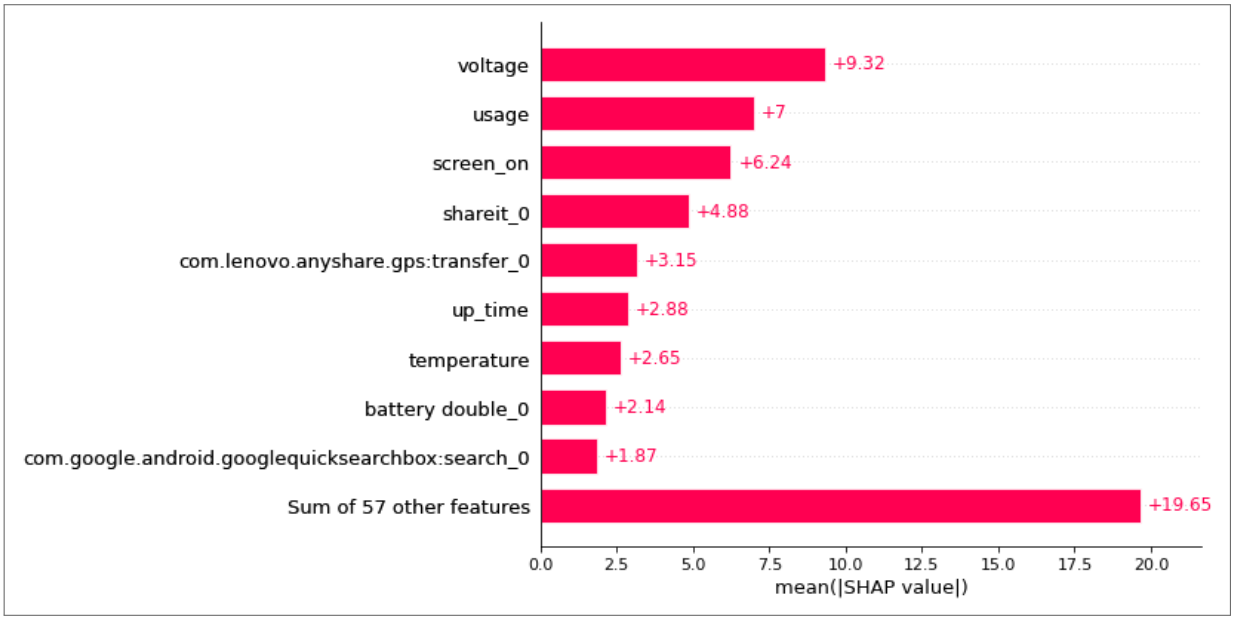
Os resultados mostram variação da precisão preditiva e descritiva das técnicas na análise

Figura 24 – Decision Tree SHAP Values - SM-N910H



Fonte: (AUTOR, 2022)

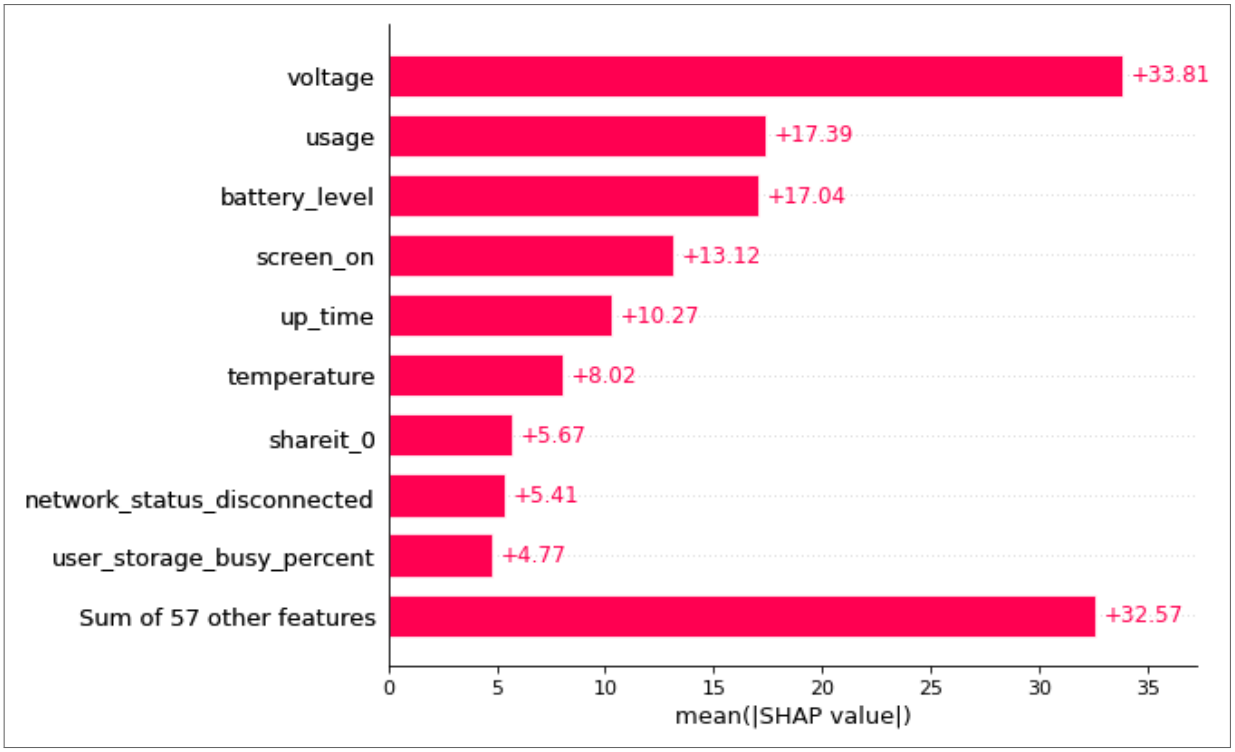
Figura 25 – Random Forest SHAP Values - SM-N910H



Fonte: (AUTOR, 2022)

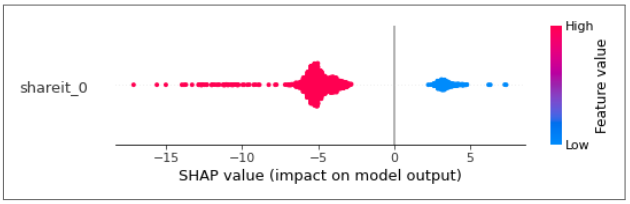
dos dados de cada modelo de dispositivo do GreenHub. Além disso, demonstram a flexibilidade na escolha do conjunto das técnicas dependendo de fatores particulares de cada algoritmo de regressão, como complexidade, recursos computacionais e tempo de execução, e dos objetivos da descoberta de padrões.

Figura 26 – XGBoost SHAP Values - SM-N910H



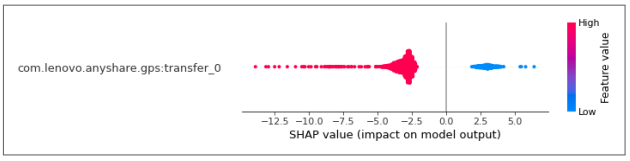
Fonte: (AUTOR, 2022)

Figura 27 – Sumário do impacto nas predições do aplicativo SHAREit (shareit_0) - SM-N910H



Fonte: (AUTOR, 2022)

Figura 28 – Sumário do impacto nas predições do processo exp_lenovo_anyshare_gps_0 - SM-N910H



Fonte: (AUTOR, 2022)

4.3.2 Configurações

Nesta seção são observadas as relações das principais configurações com as previsões realizadas pelos algoritmos de regressão em alguns modelos de dispositivos. Nas tabelas, o valor base_value representa a média das previsões e os valores correspondem de cada variáveis independentes a média absoluta da contribuição desses valores, sobre a média, na previsão final, conforme visto na Seção 2.4, é a contribuição para a diferença entre a previsão real e a

previsão média (valor de Shapley estimado), medido pelo TreeExplainer.

4.3.2.1 Configurações - battery_level

Umas dos impactos mais importantes a serem observadas é a influência do nível de bateria (battery_level), usado na geração da variável alvo deste estudo. Segundo Jeon et al. (2021), é uma preocupação fundamental para os usuários de dispositivos móveis estimarem o tempo de descarga, sendo usada pelo sistema para determinar o uso de funções, tal qual a economia de energia. Em seus experimentos detectou o consumo não proporcional entre os níveis de bateria, chegando a propor uma abordagem mais precisa para medição.

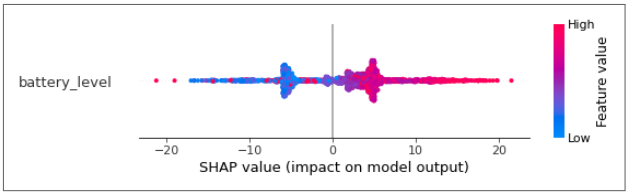
A Tabela 21 contém o valor médio das previsões (base_value) e o impacto médio do nível de bateria sobre este, e as Figuras 29, 30, 31 e 32 mostram a relação dessa variável com as previsões realizadas em alguns dos dispositivos. Quanto mais azul o ponto no gráfico, menor o nível de bateria, quanto mais vermelho, mais alto é o valor do nível. O gráfico mostra a distribuição do impacto dos dados dessa variável nas previsões finais, indicando que há uma aceleração de descarga, redução no tempo da previsão, nos níveis mais baixos de bateria.

Tabela 21 – Impacto absoluto médio (Mean(|SHAP Value|)) do nível de bateria no tempo de consumo

Atributo	Valor	Atributo	Valor
Modelo	SM-G925T	Model	MS45S
base_value	102,571	base_value	59,094
battery_level	9,590	battery_level	6,126
Modelo	GT-I9300	Model	VS501
base_value	277,345	base_value	128,482
battery_level	5,451	battery_level	15,196

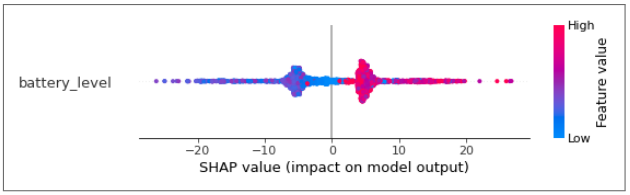
Fonte: (AUTOR, 2022)

Figura 29 – Sumário do impacto nas predições do nível de bateria - GT-I9300



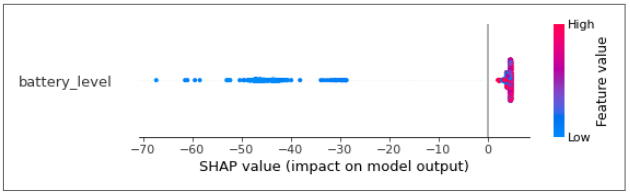
Fonte: (AUTOR, 2022)

Figura 30 – Sumário do impacto nas predições do nível de bateria - MS45S



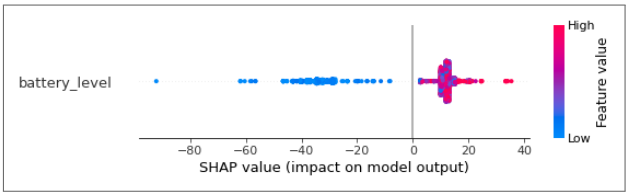
Fonte: (AUTOR, 2022)

Figura 31 – Sumário do impacto nas predições do nível de bateria - SM-G925T



Fonte: (AUTOR, 2022)

Figura 32 – Sumário do impacto nas predições do nível de bateria - VS501



Fonte: (AUTOR, 2022)

4.3.2.2 Configurações - bluetooth_enabled

Czurak et al. (2018) realizaram um estudo experimental onde observaram que a principal redução do consumo de energia se dá através da otimização dos algoritmos que controlam a busca e o consumo de recursos pelo bluetooth sendo esta específica de cada caso, dependendo da arquitetura usada e do desenvolvimento da aplicação. A utilização de conexões bluetooth nos dispositivos móveis vem crescendo à medida que cada vez mais periféricos (centrais multimídia, televisores, ar condicionados e etc.) passam a ser controlados por aplicativos. A Tabela 22 mostra o impacto médio do uso do Bluetooth sobre a média das predições, indicando uma média de contribuição mais constante entre os modelos de dispositivos em comparação com outros componentes de hardware.

4.3.2.3 Configurações - network_status

Estar conectado a uma rede, e assim a internet, é fundamental para o uso pleno e satisfatório de qualquer dispositivo móvel, sendo assim esse é um fator importante para a usabilidade

Tabela 22 – Impacto absoluto médio (Mean(|SHAP Value|)) do bluetooth no tempo de consumo

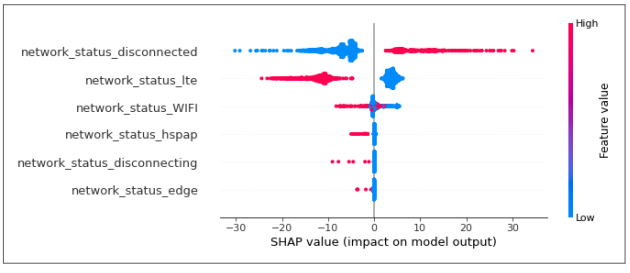
Atributo	Valor	Atributo	Valor
Modelo	SM-G903F	Model	SM-G610M
base_value	295,231	base_value	309,688
bluetooth_enabled	1,614	bluetooth_enabled	1,682
Modelo	SM-G950F	Model	SM-G610F
base_value	340,740	base_value	442,844
bluetooth_enabled	1,210	bluetooth_enabled	1,248

Fonte: (AUTOR, 2022)

de um modelo ou aplicativo. (METRI et al., 2012) observaram experimentalmente, que o uso de Wi-Fi é mais eficiente que o 3G em termos energéticos e que diferentes atividades na rede (tamanho do pacote e intervalo entre pacotes enviados/recebidos) afetam diretamente o consumo de energia e finalmente a vida útil da bateria.

Uma grande diversidade de estados, existentes na base, foi capturada conforme os exemplos da Tabela 23. Todos os estados disponíveis foram considerados neste estudo exploratório, o que acarretou em alguns vieses, como sendo o estado Desconectado (network_status_disconnected) o mais influente, funcionando como uma espécie de contraponto acumulador de todos os outros. A Figura 33 traz mais detalhes de como essas contribuições nas previsões finais, a partir da média, foi compreendida pelo modelo de regressão, indicando, conforme (METRI et al., 2012) observaram, um tempo de descarga menor para uso de conexões móveis.

Figura 33 – Sumário do impacto nas predições dos estados da rede no consumo de energia - SM-G532MT



Fonte: (AUTOR, 2022)

4.3.2.4 Configurações - location_enabled

A utilização do GPS é tida como uma das principais causas de drenagem da energia das baterias, segundo Kjaergaard (2010), um conceito geral por trás de muitos métodos de

Tabela 23 – Impacto absoluto médio (Mean(|SHAP Value|)) dos estados da rede

Atributo	Valor	Atributo	Valor
Modelo	SM-S727VL	Modelo	SM-G950F
base_value	261,780	base_value	340,740
disconnected	1,421	disconnected	7,660
disconnecting	0,124	disconnecting	0,053
evdo_a	0,023	edge	0,032
lte	7,176	gprs	0,001
WIFI	2,552	hspap	0,152
Modelo	SM-G532MT	hsupa	0,015
base_value	367,363	lte	7,541
status_0	0,000	utms	0,024
disconnected	6,852	WIFI	0,901
disconnecting	0,018	Modelo	SM-J111F
edge	0,030	base_value	246,203
gprs	0,011	disconnected	47,703
hsdpa	0,258	hspap	0,232
hspap	0,511	WIFI	26,305
hsupa	0,111	Model	GT-I9500
lte	0,047	base_value	245,877
utms	0,026	disconnected	18,468
WIFI	3,290	hspap	14,731

Fonte: (AUTOR, 2022)

economia de energia é usar uma precisão mais alta apenas quando necessário, devendo ser utilizadas diferentes técnicas dependendo da necessidade de precisão e atualização da posição e o uso consciente por parte dos serviços que utilizam esse recurso.

Conforme mostra a Tabela 24, o uso do GPS tem uma contribuição considerável nas previsões finais nos intervalos de descarga. Sendo assim, um ponto importante para o gerenciamento de energia para aplicativos e processos independente de fabricante ou modelo de dispositivo compreendendo como este se relaciona com os demais recursos do sistema, identificando situações de consumo elevado quando utilizada, ou identificando tecnologias mais econômicas dependendo do modelo de dispositivo.

Tabela 24 – Impacto absoluto médio (Mean(|SHAP Value|)) da localização habilitada nos dispositivos

Atributo	Valor	Atributo	Valor
Modelo	SM-S727VL	Model	i5E
base_value	261,780	base_value	261,756
location_enabled	6,085	location_enabled	2,733
Modelo	SM-G532MT	Model	SM-G610F
base_value	367,363	base_value	442,844
location_enabled	5,097	location_enabled	1,692

Fonte: (AUTOR, 2022)

4.3.2.5 Configurações - os_version

Sistemas operacionais são programas que atuam como uma interface entre o hardware do sistema e o usuário, sendo responsáveis por gerenciar toda a comunicação entre processos, aplicativos e componentes do hardware. Atualizações constantes fazem parte da rotina desses sistemas, aumentando a segurança, corrigindo falhas, e trazendo novas funcionalidades para desenvolvedores e usuários. Entretanto, outro fator que parece estar associado a diferentes versões do sistema é o consumo de energia. Na Tabela 25, as versões 6.0.1 e 7 do Android apresentam impactos parecidos, porém distintos, em dois modelos diferentes indicando que é possível capturar as diferenças no consumo para cada versão, presente na base, que for instalada em cada modelo de dispositivo.

Tabela 25 – Impacto absoluto médio (Mean(|SHAP Value|)) das versões dos sistema operacional

Atributo	Valor	Atributo	Valor
Model	SM-G610M	Model	SM-G610F
base_value	309,688	base_value	442,844
os_version_6.0.1	3,319	os_version_6.0.1	1,930
os_version_7	2,609	os_version_7	1,238
Model	SM-G920V	Model	SM-J510FN
base_value	136,311	base_value	374,939
os_version_5.1.1	0,419	os_version_6.0.1	5,082
os_version_6.0.1	0,059	os_version_7.1.1	1,250
os_version_7	0,503		

Fonte: (AUTOR, 2022)

4.3.2.6 Configurações - power_saver_enabled

O impacto da função de economia de energia nas previsões também foi medido pelo modelo de regressão, sendo uma configuração considerada importante para manutenção do consumo de energia. A Tabela 26 mostra a média do impacto desse modo dessa configuração em relação às predições.

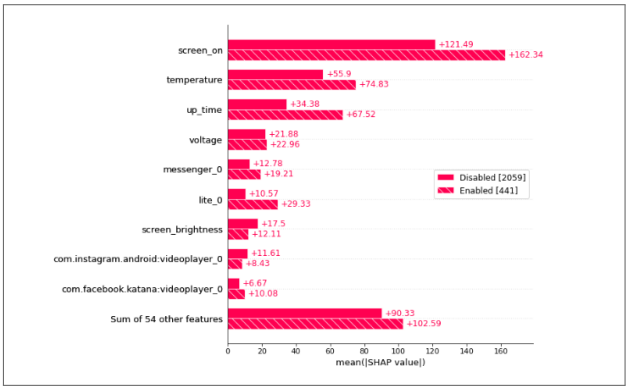
Tabela 26 – Impacto absoluto médio (Mean(|SHAP Value|)) da economia de energia no tempo de consumo

Atributo	Valor	Atributo	Valor
Modelo	SM-S727VL	Model	SM-G532MT
base_value	261,780	base_value	367,363
power_saver_enabled	4,971	power_saver_enabled	4,783
Modelo	SM-G950F	Model	SM-G950F
base_value	167,636	base_value	340,740
power_saver_enabled	2,349	power_saver_enabled	2,011

Fonte: (AUTOR, 2022)

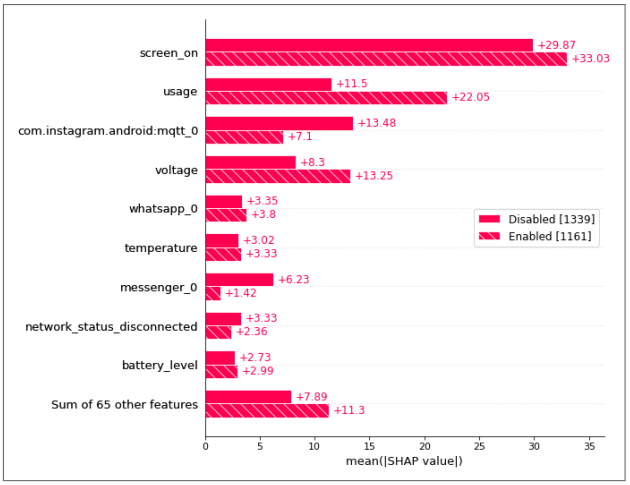
As Figuras 34, 35, 36 e 37 mostram a mudança de impacto das variáveis independentes quando o modo de economia esta ativo no sistema. É possível observar que esse modo do sistema Android atua bastante nos componentes de hardware, e que os aplicativos e processos não apresentam diferença significativa a sua contribuição sob essas condições.

Figura 34 – Comparação entre estados do modo economia de energia - SM-G532MT



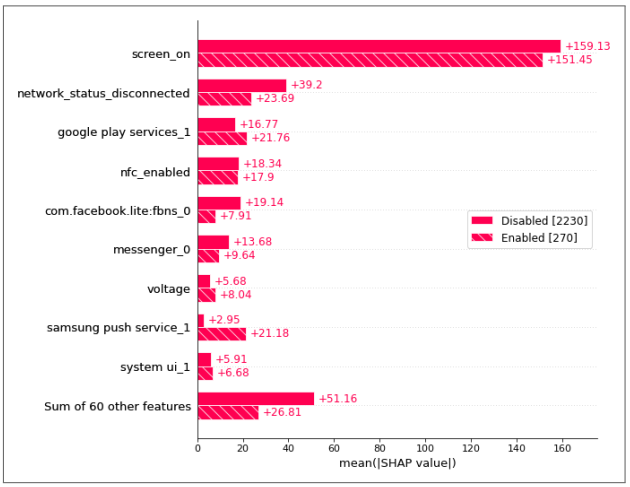
Fonte: (AUTOR, 2022)

Figura 35 – Comparação entre estados do modo economia de energia - SM-G928F



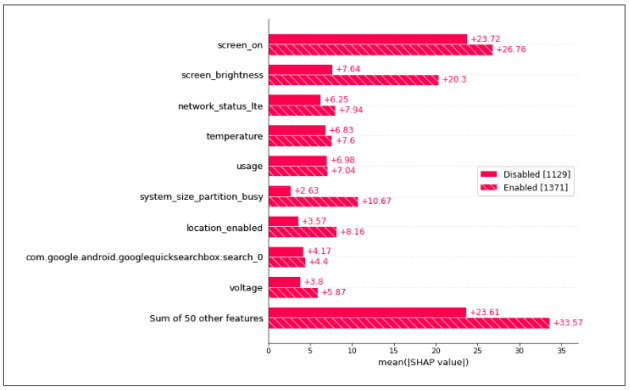
Fonte: (AUTOR, 2022)

Figura 36 – Comparação entre estados do modo economia de energia - SM-G950F



Fonte: (AUTOR, 2022)

Figura 37 – Comparação entre estados do modo economia de energia - SM-S727VL



Fonte: (AUTOR, 2022)

4.3.2.7 Configurações - screen_on/screen_brightness

A utilização da tela (screen_on) é um fator de grande impacto em todos os modelos de regressão para todos os dispositivos, pois pode significar apenas o uso geral do aparelho ou

não, agregando em uma variável binária diversos fatores. Por isso, na Tabela 27 o uso foi associado ao brilho da tela, indicando como este impacta no tempo de consumo de energia, abrindo possibilidades para comparação de diferentes tecnologias de tela, sistemas de cores, tamanho e outras informações que puderem ser agregadas através da ficha técnica dos modelos de dispositivos ou detectadas na base.

Tabela 27 – Impacto absoluto médio (Mean(|SHAP Value|)) do uso da tela

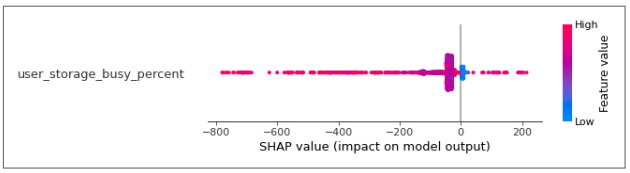
Atributo	Valor	Atributo	Valor
Model	SM-G610M	Model	SM-G610F
base_value	248,604	base_value	442,844
screen_brightness	10,690	screen_brightness	7,344
screen_on	33,648	screen_on	54,362
Model	GT-I9300	Model	Redmi Note 3
base_value	277,345	base_value	155,426
screen_brightness	6,528	screen_brightness	3,925
screen_on	57,504	screen_on	24,249

Fonte: (AUTOR, 2022)

4.3.2.8 Configurações - user_storage_busy_percent

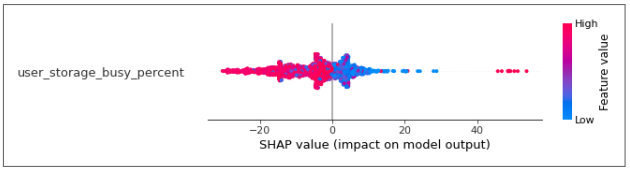
O armazenamento de dados do usuário também foi detectado como item de grande influência no tempo de decaimento das baterias. As Figuras 38, 39, 40, e 41 mostram como os dados do usuários, incluindo os aplicativos baixados e seus dados armazenados, influenciam no consumo de energia. Uma porcentagem maior de ocupação desse armazenamento (caracterizada pelos pontos mais em vermelho nas imagens) tem a tendência de reduzir o tempo de descarga prevista. A Tabela 28 mostra a contribuição média dessa variável sobre o resultado final das previsões, a partir da média.

Figura 38 – Sumário do impacto nas predições do armazenamento do usuário - GT-I9300



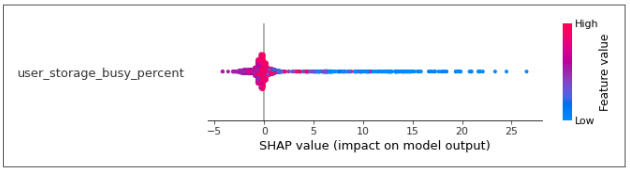
Fonte: (AUTOR, 2022)

Figura 39 – Sumário do impacto nas predições do armazenamento do usuário - SM-G532MT



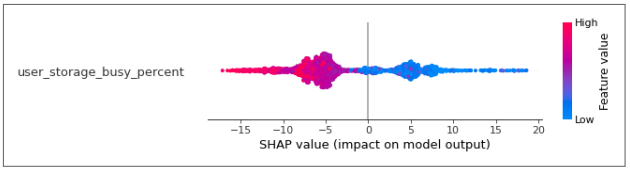
Fonte: (AUTOR, 2022)

Figura 40 – Sumário do impacto nas predições do armazenamento do usuário - Lenovo A1000



Fonte: (AUTOR, 2022)

Figura 41 – Sumário do impacto nas predições do armazenamento do usuário - XT1080



Fonte: (AUTOR, 2022)

Tabela 28 – Impacto absoluto médio (Mean(|SHAP Value|)) do armazenamento do usuário no tempo de consumo

Atributo	Valor	Atributo	Valor
Modelo	GT-I9300	Model	SM-G532MT
base_value	277,345	base_value	367,363
user_storage_busy_percent	30,845	user_storage_busy_percent	7,997
Modelo	Lenovo A1000	Model	XT1080
base_value	26,599	base_value	202,726
user_storage_busy_percent	1,625	user_storage_busy_percent	6,171

Fonte: (AUTOR, 2022)

4.3.2.9 Configurações - ram_busy_percent

A média da contribuição absoluta para as previsões da porcentagem de memória RAM utilizada está indicada na Tabela 29. Conforme mostra o exemplo, os resultados podem variar dependendo de cada modelo de dispositivo, o que indica alguma influência de outros fatores de hardware.

As Figuras 42, 43, 44 e 45 mostram que o maior uso da memória RAM impacta positi-

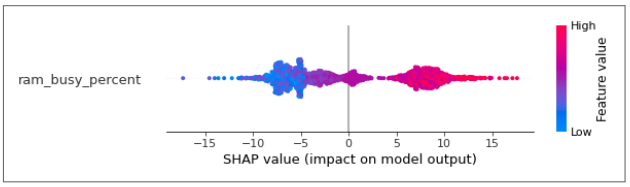
vamente o tempo de descarga. Uma das possibilidades é que a manipulação de dados nesse tipo de memória pode indicar uma carga energética menor em relação a operações de escrita e leitura no armazenamento.

Tabela 29 – Impacto absoluto médio (Mean(|SHAP Value|)) do uso da memória RAM no tempo de consumo

Atributo	Valor	Atributo	Valor
Modelo	LG-M250	Model	Redmi Note 3
base_value	251,938	base_value	155,426
ram_busy_percent	6,167	ram_busy_percent	1,487
Modelo	SM-G950F	Model	SM-N900T
base_value	340,740	base_value	186,994
ram_busy_percent	9,323	ram_busy_percent	1,408

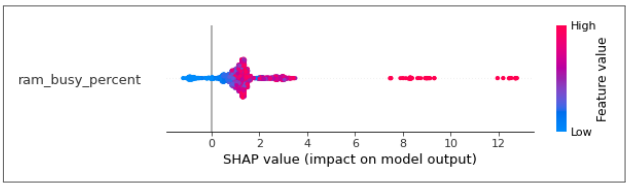
Fonte: (AUTOR, 2022)

Figura 42 – Sumário do impacto nas predições do uso da memória RAM energia - LG-M250



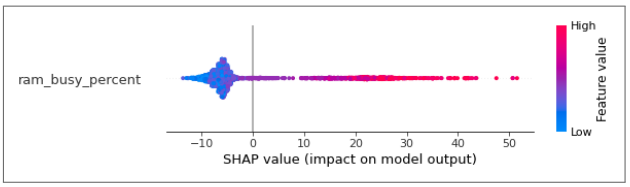
Fonte: (AUTOR, 2022)

Figura 43 – Sumário do impacto nas predições do uso da memória RAM energia - Readmi 3



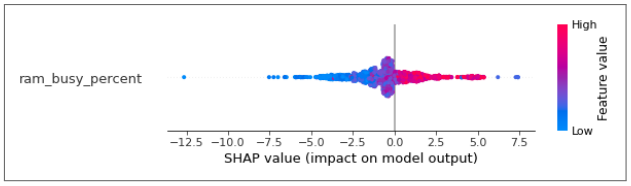
Fonte: (AUTOR, 2022)

Figura 44 – Sumário do impacto nas predições do uso da memória RAM energia - SM-G950F



Fonte: (AUTOR, 2022)

Figura 45 – Sumário do impacto nas predições do uso da memória RAM energia - SM-N900T



Fonte: (AUTOR, 2022)

4.3.2.10 Configurações - temperature/usage/voltage

Os sensores têm papel fundamental para detectar problemas e condições do uso que possam diminuir ou aumentar o tempo de descarga no uso dos dispositivos. Conforme a Tabela 30, em comparação com as outras variáveis independentes, os sensores tem um impacto absoluto médio maior sobre as previsões.

Tabela 30 – Impacto absoluto médio (Mean(|SHAP Value|)) dos sensores (temperatura, voltagem e uso de CPU

Atributo	Valor	Atributo	Valor
Model	ASUS_Z007	Model	SM-G7102
base_value	126,064	base_value	419,697
temperature	34,380	temperature	130,279
usage	17,846	usage	0,000
voltage	44,629	voltage	73,579
Model	GT-I9300	Model	LG-M250
base_value	277,345	base_value	251,938
temperature	19,729	temperature	16,519
usage	41,182	usage	11,612
voltage	17,276	voltage	9,681

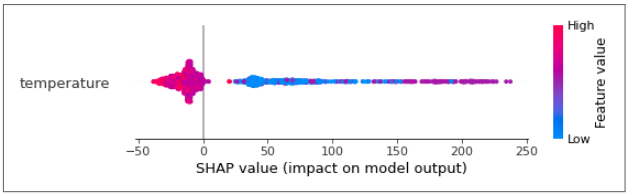
Fonte: (AUTOR, 2022)

As Figuras 46, 47, 48 e 49, dão a dimensão do impacto da temperatura para as previsões, indicando o quanto os modelos de dispositivos podem ser sensíveis a condições externas, ou quanto o fato de produzirem calor acelera a descarga da bateria.

4.3.3 Processos do Sistema

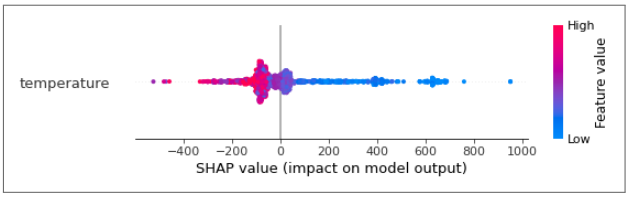
A maior parte dos aplicativos do Google comumente utilizados nos dispositivos móveis são tidos como processos do sistema (identificados dentre as variáveis independentes neste estudo com um "_1"ao final dos nomes) e por isso surgem entre os mais populares da base

Figura 46 – Sumário do impacto nas predições da temperatura no consumo de energia - ASUS_Z007



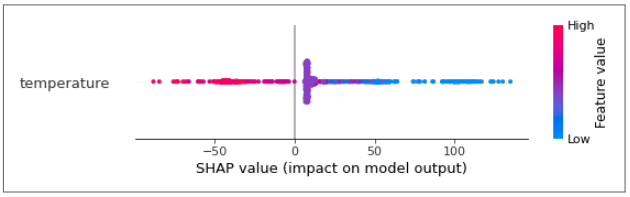
Fonte: (AUTOR, 2022)

Figura 47 – Sumário do impacto nas predições da temperatura no consumo de energia - SM-G7102



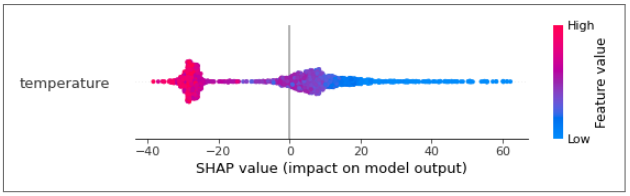
Fonte: (AUTOR, 2022)

Figura 48 – Sumário do impacto nas predições da temperatura no consumo de energia - GT-I9300



Fonte: (AUTOR, 2022)

Figura 49 – Sumário do impacto nas predições da temperatura no consumo de energia - LG-M250



Fonte: (AUTOR, 2022)

segundo esse critério. Na Tabela 31 podemos ver uma comparação entre esses processos em alguns dispositivos. Dentre esses processos, o Google Play Services (google play services_1) é o que possui maior influência nas previsões, pois é usado para atualizar apps do Google e apps do Google Play, fornecendo funcionalidades essenciais, como autenticação de serviços, sincronização, pesquisas off-line, mapas mais imersivos e aprimoramento de experiências de jogo.

Tabela 31 – Impacto absoluto médio (Mean(|SHAP Value|)) de processos do sistema

Atributo	Valor	Atributo	Valor
Model	SM-G610F	Model	SM-J250F
base_value	442,844	base_value	233,213
google play music_1	8,144	google play music_1	1,641
google play services_1	4,464	google play services_1	5,208
google play store_1	0,595	google play store_1	0,337
Model	SM-N900T	Model	GT-I9300
base_value	186,994	base_value	277,345
google play music_1	0,879	google play music_1	0,746
google play services_1	5,315	google play services_1	4,430
google play store_1	4,097	google play store_1	2,017

Fonte: (AUTOR, 2022)

4.3.4 Aplicativos

Decisões na construção de um aplicativo impactam diretamente o consumo de recursos de um dispositivo móvel. Conforme Pereira et al. (2017) apontaram após um estudo experimental com 27 linguagens, que o uso de diferentes linguagens na construção de um software influenciam diretamente no consumo de recursos energéticos. Avaliar processos em uso real e em larga escala sob essa perspectiva pode relacionar essas decisões a muitas outras configurações e recursos.

De maneira semelhante aos processos do sistema operacional, é possível avaliar cada processo ligado a um aplicativo (ou pacote) de todos os aplicativos presentes na base GreenHub, e quantificar sua importância para os modelos de regressão, comparando-os dentro de uma mesma categoria/assunto de interesse, conforme a Tabela 32. Pelos resultados é perceptível que os aplicativos ligados ao Facebook contribuem mais para o resultado final das previsões entre os mais populares na categoria de processos de aplicativos que reproduzem vídeo.

Tabela 32 – Impacto absoluto médio (Mean(|SHAP Value|)) de aplicativos - videoplayers

Atributo	Valor	Atributo	Valor
Model	SM-S727VL	Model	LG-M250
base_value	261,780	base_value	251,938
facebook.katana	0,567	facebook.katana	2,256
facebook.orca	0,703	facebook.orca	1,070
instagram.android	0,475	instagram.android	0,370
youtube_1	2,146	youtube_1	0,753
Model	Redmi Note 3	Model	XT1080
base_value	155,426	base_value	202,726
facebook.katana	0,199	facebook.katana	0,598
facebook.orca	0,707	facebook.orca	0,308
instagram.android	0,132	instagram.android	0,558
youtube_1	0,011	youtube_1	0,137

Fonte: (AUTOR, 2022)

4.4 AÇÃO SOBRE O CONHECIMENTO DESCOBERTO

Nesta última etapa do processo KDD as descobertas, principais observações e apontamentos, relevantes para os interessados são relatadas e disponibilizadas para os interessados. Sendo assim, foram levantadas algumas possibilidades de uso do tipo de aplicação do processo de mineração de dados proposto neste trabalho.

A partir de um trabalho de mineração de dados como esse, um pesquisador pode estudar características semelhantes dentro de grupos, pois um esquema novo de cores terá pesos diferentes entre dispositivos diferentes, por exemplo. Um desenvolvedor pode estar interessado em comparar o desempenho energético, e de outros componentes, do seu aplicativo com concorrentes da mesma categoria e/ou entre atualizações do seu próprio aplicativo, como o caso do sistema operacional destacado nesse trabalho.

Conforme mostram alguns estudos citados nas seções anteriores, pesquisas relacionadas ao consumo energético são realizadas considerando diversas perspectivas do contexto móvel (configurações, sensores, tipos de conexão, arquitetura de aplicativos e etc.), presentes também neste estudo, o que torna a metodologia e os indicadores que ela produz uma fonte valiosa para descoberta de conhecimento relacionada a todas essas áreas, estimulando o surgimento de novas abordagens baseadas em dados de uso real e em larga escala, agregando mais conhecimento às descobertas realizadas em experimentos de laboratório com uma quan-

tidade limitada de cenários e de modelos de dispositivos, frente a heterogeneidade do contexto Android, podendo inclusive direcioná-las a partir das relações observadas em uma maior escala.

Além disso, o destaque da influência de alguns sensores pode direcionar a atenção do desenvolvimento para o uso mais específico dos recursos, pois em vários dispositivos determinada características pesam mais que em outros, e para garantir que economia de bateria em determinados modelos, é necessário olhar para esses fatores que pesam mais neles, como por exemplo, não sobrecarregar a CPU em certos modelos mais sensíveis.

Os usuários também podem se beneficiar a partir de estudos como esse, pois fatores de consumo relacionados hardware (ex: tela) podem apresentar padrões de maior ou menor consumo, auxiliando na escolha e no uso mais eficiente do aparelho, além de poder comparar aplicativos desempenho de aplicativos semelhantes, e incorporar padrões de uso mais energeticamente saudáveis para seus aparelhos e/ou desmistificando percepções equivocadas.

Dentro do projeto GreenHub, esse trabalho buscou manter as mesmas considerações e caracterizações das questões abordadas em (PEREIRA et al., 2021), buscando ser uma extensão direta do trabalho realizado, conforme seus apontamentos e considerações.

5 CONCLUSÃO

5.1 CONSIDERAÇÕES FINAIS

Este trabalho exploratório teve como objetivo investigar a viabilidade de desenvolvimento de modelos a partir de técnicas tradicionais de Mineração de Dados e Aprendizado de Máquina para estudar as relações das características e processos dos dispositivos com o tempo no decaimento dos níveis de bateria em dispositivos Android, numa base de dados em larga escala. Para isso, os modelos regressores deveriam ser capazes de prever o tempo levado para consumir o equivalente a 1% da bateria dos modelos de dispositivo analisados. Além disso, seriam submetidos a uma abordagem explicativa de como esses resultados foram construídos, estimando o impacto das variáveis independentes utilizadas, sendo estás divididas em três grupos: configurações, processos e aplicativos.

Os algoritmos de regressão Decision Tree, Random Forest e XGBoost, foram escolhidos para serem avaliados neste estudo, todos baseados em árvores de decisão. Essa escolha foi feita pela maior simplicidade no processo de tratamento de dados para esses tipos de modelos e a versatilidade para aplicação de diferentes técnicas como boosting, bagging e bootstraping. A escolha também foi feita pensando na análise de suas capacidades explicativas, utilizando a abordagem SHAP, através de sua aplicação prática, o TreeExplainer.

Os 100 modelos de dispositivos mais populares da base foram estudados, justamente com os processos e aplicativos mais populares. Os regressores foram treinados utilizando um total de 1 milhão de entradas para treino e 250 mil para testes. Os três algoritmos tiveram bons resultados, cada um deles sendo capazes de prever o tempo de consumo de 1% de bateria com erros médios abaixo de 15% nos melhores casos, e melhores capacidade preditiva em número parecido de modelos de dispositivos. O XGBoost obteve melhor MAPE em 42 modelos de dispositivos, enquanto o Decision Tree e o Random Forest, 30 e 28 respectivamente. Isso mostra que as técnicas podem se alternar, sendo escolhidas de acordo com a capacidade computacional disponível, simplicidade de implementação, ajuste de parâmetros, e objetivo da mineração.

Além da capacidade preditiva, a precisão descritiva dos algoritmos foi analisada. Para que a capacidade de uma estrutura mais simplificada fosse comparada com uma mais complexa, foi utilizada uma abordagem comum aos três algoritmos, o TreeExplainer. Explicando de maneira simples, ele estima uma previsão média local (individual de cada ponto) e estabelece como

cada variável independente interfere na diferença entre a predição média e a final. A média das contribuições marginais absolutas foi utilizada nesta pesquisa para estimar o impacto da dos fatores nos modelos, ou seja, sua relevância média entre todas as predições realizadas, sendo essa a precisão descritiva global usada para medir seu impacto nos modelos de regressão.

Na capacidade descritiva, o Random Forest demonstrou ter o melhor desempenho, segundo a abordagem do estudo, tendo o TreeRxplainer quantificado um número médio maior de variáveis quando aplicado neste algoritmos, enquanto o XGBoost apresentou dificuldade em ter seus critérios de decisão avaliados. Embora tenha apresentando uma capacidade preditiva menor que o Random Forest, o Decision Tree teve um custo computacional muito menor que os outros dois, o suficiente para ter seu custo benefício avaliado para uso.

Ainda na sessão de resultados, foi possível quantificar o impacto das configurações, processo e aplicativos, mesmo numa análise inicial exploratória, o que indica a capacidade de captura dessas contribuições pela metodologia escolhida, e sua relevância para os interessados em compreender melhor a importância deles para o consumo energético. Foi possível observar que as informações dos sensores e configurações possuem maior influência sobre a atividade energética dos dispositivos e que a maior carga de trabalho no modo de economia é do sistema no gerenciamento do uso do hardware, pois o impacto médio das configurações são mais sensíveis a essa configuração.

5.2 LIMITAÇÕES DO ESTUDO

Um trabalho de mineração de dados costuma ter um caráter bastante empírico pois, apesar de todo o embasamento teórico existente, muitas etapas se repetem de acordo com resultados e escolhas dos interessados. Sendo assim, esse trabalho apresenta diversos pontos que podem ser apontados como oportunidades de melhoria em trabalhos futuros, ou revisões.

O método de detecção e remoção de outliers poderia ser menos conservador com os dados, eliminando assim a necessidade de fazer previsões com tempo de decaimento muito baixo, que podem indicar problemas na bateria e dificultar a produção das métricas, mas que foram considerados neste estudo pelo seu objetivo investigativo e viés exploratórios das técnicas.

A variação dos tamanhos das baterias, e da idade de cada aparelho, faz o tempo das baterias variarem dentro de um mesmo cenário, pois teria que ter a media de descargas totais para normalizar a variável alvo. Espera-se que por ser um processo em larga escala esse impacto seja reduzido.

Muitos modelos de regressão obtiveram erros que podem ser considerados como ameaça a confiança da metodologia quando aplicada a certas configurações de cenários, como foi o caso de alguns modelos de dispositivos neste estudo, limitação essa que se deve principalmente a capacidade reduzida que os dados tem de representar o comportamento dos aparelhos em sua totalidade, em especial os relacionados à características dos hardwares.

Por fim, embora tenha processado uma quantidade grande de dados, esse estudo analisou apenas uma fração do GreenHub para medir capacidade de produzir conhecimento utilizando a metodologia proposta.

5.3 TRABALHOS FUTUROS

Com tudo que foi considerado na seção anterior, e as interessantes descobertas e oportunidades para a realização de novas análises no GreenHub e em outras bases semelhantes, muitas outras pesquisas podem ser realizadas, entre elas:

Trabalhos Futuros:

- Estudos com seleção de samples por critérios: Avaliar categorias inteiras de aplicativos, ou apenas aplicativos presentes em certas configurações.
- Estudar variações de impacto em cenários específicos.
- Estudar por particularidades de sensores: dispositivos com velocidades maiores ou menores de decaimento de um mesmo modelo, ou que esquentem mais ou menos que a média do modelo.
- Estudo das interações entre as features: compreender como elas interagiram para produzir os resultados locais e globais que possam a vir a ser observados.

REFERÊNCIAS

ALAWNAH, S.; SAGAHYROON, A. Modeling of smartphones' power using neural networks. *EURASIP Journal on Embedded Systems*, Springer, v. 2017, n. 1, p. 1–11, 2017.

AZEVEDO, A.; SANTOS, M. F. Kdd, semma and crisp-dm: a parallel overview. *IADS-DM*, 2008.

BERGSTRA, J.; BENGIO, Y. Random search for hyper-parameter optimization. *Journal of machine learning research*, v. 13, n. 2, 2012.

BONACCORSO, G. *Machine learning algorithms*. [S.l.]: Packt Publishing Ltd, 2017.

BOWDEN, T.; BAUER, B.; NERIN, J.; FENG, S. S.; SEIBOLD, S. *Documentation/filesystems/proc.txt - kernel/MSM - git at google*. Google Git. Disponível em: <https://android.googlesource.com/kernel/msm/+/android-msm-flo-3.4-kitkat-mr1/Documentation/filesystems/proc.txt>.

BREIMAN, L. Bagging predictors. *Machine learning*, Springer, v. 24, n. 2, p. 123–140, 1996.

BREIMAN, L. Random forests. *Machine learning*, Springer, v. 45, n. 1, p. 5–32, 2001.

BREIMAN, L.; FRIEDMAN, J.; STONE, C. J.; OLSHEN, R. A. *Classification and regression trees*. [S.l.]: CRC press, 1984.

BROWNLEE, J. *Essence of boosting ensembles for machine learning*. 2021. Disponível em: <https://machinelearningmastery.com/essence-of-boosting-ensembles-for-machine-learning/>.

BRYS, G.; HUBERT, M.; STRUYF, A. A robust measure of skewness. *Journal of Computational and Graphical Statistics*, Taylor & Francis, v. 13, n. 4, p. 996–1017, 2004.

CHATURVEDI, K. K.; SING, V.; SINGH, P. Tools in mining software repositories. In: IEEE. *2013 13th International Conference on Computational Science and Its Applications*. [S.l.], 2013. p. 89–98.

CHEN, T.; GUESTRIN, C. Xgboost: A scalable tree boosting system. In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. [S.l.: s.n.], 2016. p. 785–794.

CORRAL, L.; SILLITTI, A.; SUCCI, G. Software assurance practices for mobile applications. *Computing*, Springer, v. 97, n. 10, p. 1001–1022, 2015.

CZURAK, P.; MAJ, C.; SZERMER, M.; ZABIEROWSKI, W. Impact of bluetooth low energy on energy consumption in android os. In: IEEE. *2018 XIV-th International Conference on Perspective Technologies and Methods in MEMS Design (MEMSTECH)*. [S.l.], 2018. p. 255–258.

DIETTERICH, T. Overfitting and undercomputing in machine learning. *ACM computing surveys (CSUR)*, ACM New York, NY, USA, v. 27, n. 3, p. 326–327, 1995.

DIETTERICH, T. G. Ensemble methods in machine learning. In: SPRINGER. *International workshop on multiple classifier systems*. [S.l.], 2000. p. 1–15.

DIETTERICH, T. G. et al. Ensemble learning. *The handbook of brain theory and neural networks*, MIT press Cambridge, Massachusetts, v. 2, n. 1, p. 110–125, 2002.

DING, M.; WANG, T.; WANG, X. Establishing smartphone user behavior model based on energy consumption data. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, ACM New York, NY, v. 16, n. 2, p. 1–40, 2021.

DINH, H. T.; LEE, C.; NIYATO, D.; WANG, P. A survey of mobile cloud computing: architecture, applications, and approaches. *Wireless communications and mobile computing*, Wiley Online Library, v. 13, n. 18, p. 1587–1611, 2013.

DUNN, J.; MINGARDI, L.; ZHUO, Y. D. Comparing interpretability and explainability for feature selection. *arXiv preprint arXiv:2105.05328*, 2021.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. From data mining to knowledge discovery in databases. *AI magazine*, v. 17, n. 3, p. 37–37, 1996.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. The kdd process for extracting useful knowledge from volumes of data. *Communications of the ACM*, ACM New York, NY, USA, v. 39, n. 11, p. 27–34, 1996.

FREUND, Y.; SCHAPIRE, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, Elsevier, v. 55, n. 1, p. 119–139, 1997.

FRIEDMAN, J. H. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, JSTOR, p. 1189–1232, 2001.

GAO, X.; LIU, D.; LIU, D.; WANG, H.; STAVROU, A. E-android: A new energy profiling tool for smartphones. In: IEEE. *2017 IEEE 37th international conference on distributed computing systems (ICDCS)*. [S.l.], 2017. p. 492–502.

GUO, Y.; WANG, C.; CHEN, X. Understanding application-battery interactions on smartphones: A large-scale empirical study. *IEEE Access*, IEEE, v. 5, p. 13387–13400, 2017.

HASSAN, A. E. The road ahead for mining software repositories. In: IEEE. *2008 Frontiers of Software Maintenance*. [S.l.], 2008. p. 48–57.

HUBERT, M.; VANDERVIEREN, E. An adjusted boxplot for skewed distributions. *Computational statistics & data analysis*, Elsevier, v. 52, n. 12, p. 5186–5201, 2008.

JEON, S.; KIM, D.; AHN, J.; HA, R.; CHA, H. Revisiting the battery level indicator of mobile devices. *Design Automation for Embedded Systems*, Springer, v. 25, n. 1, p. 65–85, 2021.

KJAERGAARD, M. Location-based services on mobile phones: Minimizing power consumption. *IEEE Pervasive Computing*, IEEE, v. 11, n. 1, p. 67–73, 2010.

LEARN scikit. *Decision Trees*. 2021. Disponível em: <https://https://scikit-learn.org/stable/modules/tree.html>. Acesso em: 30 jan. 2022.

LOH, W.-Y. Classification and regression trees. *Wiley interdisciplinary reviews: data mining and knowledge discovery*, Wiley Online Library, v. 1, n. 1, p. 14–23, 2011.

LUNDBERG, S. M.; ERION, G.; CHEN, H.; DEGRAVE, A.; PRUTKIN, J. M.; NAIR, B.; KATZ, R.; HIMMELFARB, J.; BANSAL, N.; LEE, S.-I. Explainable ai for trees: From local explanations to global understanding. *arXiv preprint arXiv:1905.04610*, 2019.

LUNDBERG, S. M.; LEE, S.-I. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, v. 30, 2017.

MAGERMAN, D. M. Statistical decision-tree models for parsing. *arXiv preprint cmp-lg/9504030*, 1995.

MATALONGA, H.; CABRAL, B.; CASTOR, F.; COUTO, M.; PEREIRA, R.; SOUSA, S. M. de; FERNANDES, J. P. Greenhub farmer: real-world data for android energy mining. In: IEEE. *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*. [S.l.], 2019. p. 171–175.

MATIGNON, R. *Data mining using SAS enterprise miner*. [S.l.]: John Wiley & Sons, 2007.

METRI, G.; AGRAWAL, A.; PERI, R.; SHI, W. What is eating up battery life on my smartphone: A case study. In: IEEE. *2012 International Conference on Energy Aware Computing*. [S.l.], 2012. p. 1–6.

MOLNAR, C. *Interpretable machine learning*. 2022. Disponível em: <https://christophm.github.io/interpretable-ml-book/shapley.html#shapley>.

MURDOCH, W. J.; SINGH, C.; KUMBIER, K.; ABBASI-ASL, R.; YU, B. Definitions, methods, and applications in interpretable machine learning. *Proceedings of the National Academy of Sciences*, National Acad Sciences, v. 116, n. 44, p. 22071–22080, 2019.

NETO, A. S. B.; FARIAS, F.; MIALARET, M. A. T.; CARTAXO, B.; LIMA, P. A.; MACIEL, P. Building energy consumption models based on smartphone user's usage patterns. *Knowledge-Based Systems*, Elsevier, v. 213, p. 106680, 2021.

NUCCI, D. D.; PALOMBA, F.; PROTA, A.; PANICHELLA, A.; ZAIDMAN, A.; LUCIA, A. D. Software-based energy profiling of android apps: Simple, efficient and reliable? In: IEEE. *2017 IEEE 24th international conference on software analysis, evolution and reengineering (SANER)*. [S.l.], 2017. p. 103–114.

OLINER, A. J.; IYER, A. P.; STOICA, I.; LAGERSPETZ, E.; TARKOMA, S. Carat: Collaborative energy diagnosis for mobile devices. In: *Proceedings of the 11th ACM conference on embedded networked sensor systems*. [S.l.: s.n.], 2013. p. 1–14.

PEREIRA, R.; COUTO, M.; RIBEIRO, F.; RUA, R.; CUNHA, J.; FERNANDES, J. P.; SARAIVA, J. Energy efficiency across programming languages: how do energy, time, and memory relate? In: *Proceedings of the 10th ACM SIGPLAN International Conference on Software Language Engineering*. [S.l.: s.n.], 2017. p. 256–267.

PEREIRA, R.; MATALONGA, H.; COUTO, M.; CASTOR, F.; CABRAL, B.; CARVALHO, P.; SOUSA, S. M. de; FERNANDES, J. P. Greenhub: a large-scale collaborative dataset to battery consumption analysis of android devices. *Empirical Software Engineering*, Springer, v. 26, n. 3, p. 1–55, 2021.

PETER, S. C.; DHANJAL, J. K.; MALIK, V.; RADHAKRISHNAN, N.; JAYAKANTHAN, M.; SUNDAR, D.; SUNDAR, D.; JAYAKANTHAN, M. Encyclopedia of bioinformatics and computational biology. *Ranganathan, S., Grib-skov, M., Nakai, K., Schönbach, C., Eds*, p. 661–676, 2018.

PINTO, G.; CASTOR, F.; LIU, Y. D. Mining questions about software energy consumption. In: *Proceedings of the 11th Working Conference on Mining Software Repositories*. [S.l.: s.n.], 2014. p. 22–31.

PRAMANIK, P. K. D.; PAL, S.; CHOUDHURY, P. Green and sustainable high-performance computing with smartphone crowd computing. *Scalable Computing: Practice and Experience*, v. 20, n. 2, p. 259–284, 2019.

PRAMANIK, P. K. D.; SINHABABU, N.; MUKHERJEE, B.; PADMANABAN, S.; MAITY, A.; UPADHYAYA, B. K.; HOLM-NIELSEN, J. B.; CHOUDHURY, P. Power consumption analysis, measurement, management, and issues: A state-of-the-art review of smartphone battery and energy usage. *IEEE Access*, IEEE, v. 7, p. 182113–182172, 2019.

QUINLAN, J. R. *C4. 5: programs for machine learning*. [S.l.]: Elsevier, 2014.

RAZI, M. A.; ATHAPPILLY, K. A comparative predictive analysis of neural networks (nns), nonlinear regression and classification and regression tree (cart) models. *Expert systems with applications*, Elsevier, v. 29, n. 1, p. 65–74, 2005.

RIPLEY, B. D. *Pattern recognition and neural networks*. [S.l.]: Cambridge university press, 2007.

SAMEK, W. Learning with explainable trees. *Nature Machine Intelligence*, Nature Publishing Group, v. 2, n. 1, p. 16–17, 2020.

SARKER, I. H. Data science and analytics: an overview from data-driven smart computing, decision-making and applications perspective. *SN Computer Science*, Springer, v. 2, n. 5, p. 1–22, 2021.

SERENGIL, S. *A step by step regression tree example*. 2018. Disponível em: <https://sefiks.com/2018/08/28/a-step-by-step-regression-decision-tree-example/>.

SHAFIQUE, U.; QAISER, H. A comparative study of data mining process models (kdd, crisp-dm and semma). *International Journal of Innovation and Scientific Research*, Citeseer, v. 12, n. 1, p. 217–222, 2014.

SHAPLEY, L. S. A value for n-person games. *Contributions to the Theory of Games*, p. 307—-317, 1953.

STROBL, C.; MALLEY, J.; TUTZ, G. An introduction to recursive partitioning: rationale, application, and characteristics of classification and regression trees, bagging, and random forests. *Psychological methods*, American Psychological Association, v. 14, n. 4, p. 323, 2009.

STULP, F.; SIGAUD, O. Many regression algorithms, one unified model: A review. *Neural Networks*, Elsevier, v. 69, p. 60–79, 2015.

SURAMPUDI, S. *Oracle Data Mining API Guide, 19c*. 2021. Disponível em: <https://docs.oracle.com/en/database/oracle/oracle-database/19/dmapi/index.html>. Acesso em: 21 dez. 2021.

WEIRAN, S. *Hyper parameter tuning with randomised grid search*. Towards Data Science, 2019. Disponível em: <https://towardsdatascience.com/hyper-parameter-tuning-with-randomised-grid-search-54f865d27926#:~:text=Randomised\%20Grid\%20Search\%20is\%20a\%20great\%20alternative\%20to\%20Grid\%20Search&text=However\%2C\%20how\%20many\%20iterations\%20are,found\%2C\%20regardless\%20of\%20grid\%20size.>

WILKE, C.; RICHLY, S.; GÖTZ, S.; PIECHNICK, C.; ASSMANN, U. Energy consumption and efficiency in mobile applications: A user feedback study. In: IEEE. *2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing*. [S.l.], 2013. p. 134–141.

WIRTH, R.; HIPP, J. Crisp-dm: Towards a standard process model for data mining. In: MANCHESTER. *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*. [S.l.], 2000. v. 1, p. 29–40.

XU, M.; WATANACHATURAPORN, P.; VARSHNEY, P. K.; ARORA, M. K. Decision tree regression for soft classification of remote sensing data. *Remote Sensing of Environment*, Elsevier, v. 97, n. 3, p. 322–336, 2005.

ZHANG, C.; MA, Y. *Ensemble machine learning: methods and applications*. [S.l.]: Springer, 2012.

ZHAO, X.; GUO, Y.; FENG, Q.; CHEN, X. A system context-aware approach for battery lifetime prediction in smart phones. In: *Proceedings of the 2011 ACM Symposium on Applied Computing*. [S.l.: s.n.], 2011. p. 641–646.

ZHENG, A. *How to evaluate machine learning models: Hyperparameter tuning*. 2015. Disponível em: <https://web.archive.org/web/20160701182750/http://blog.dato.com/how-to-evaluate-machine-learning-models-part-4-hyperparameter-tuning>.