UNIVERSIDADE FEDERAL DE SANTA CATARINA
CAMPUS ARARANGUÁ
TECNOLOGIAS DA INFORMAÇÃO E COMUNICAÇÃO

CARLOS MIGUEL CARVALHO DE SOUZA

ANÁLISE DE PADRÕES DE PROGRAMAÇÃO EM LOGS COLETADOS A PARTIR DO USO DA PLATAFORMA SCRATCH

Araranguá,2022

CARLOS MIGUEL CARVALHO DE SOUZA

ANÁLISE DE PADRÕES DE PROGRAMAÇÃO EM LOGS COLETADOS A PARTIR DO USO DA PLATAFORMA SCRATCH

Trabalho Conclusão do Curso de Graduação em Tecnologias da informação e Comunicação do Centro de Araranguá da Universidade Federal de Santa Catarina como requisito para a obtenção do título de Bacharel em Tecnologias da informação e comunicação Orientador: Prof. Cristian Cechinel.

Araranguá,2022

Ficha de identificação da obra elaborada pelo autor, através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

SOUZA, CARLOS MIGUEL CARVALHO DE
ANÁLISE DE PADRÕES DE PROGRAMAÇÃO EM LOGS COLETADOS A PARTIR DO USO DA PLATAFORMA SCRATCH / CARLOS MIGUEL CARVALHO DE SOUZA ; orientador, CRISTIAN CECHINEL, 2022.
60 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Campus Araranguá, Graduação em Tecnologias da Informação e Comunicação, Araranguá, 2022.

Inclui referências.

1. Tecnologias da Informação e Comunicação. I. CECHINEL, CRISTIAN. II. Universidade Federal de Santa Catarina. Graduação em Tecnologias da Informação e Comunicação. III. Título.

Carlos Miguel Carvalho de Souza

ANÁLISE DE PADRÕES DE PROGRAMAÇÃO EM LOGS COLETADOS A PARTIR DO USO DA PLATAFORMA SCRATCH

Este Trabalho Conclusão de Curso foi julgado adequado para obtenção do Título de ““Bacharel emTecnologias da Informação e Comunicação” e aprovado em sua forma final pelo Curso de Graduação em Tecnologias da Informação e Comunicação.

Araranguá, 03 de Agosto de 2022

Documento assinado digitalmente
Vilson Gruber
Data: 05/08/2022 09:39:30-0300
CPF: 175.317.788-07
Verifique as assinaturas em https://v.ufsc.br

Prof. Vilson Gruber, Dr.

Coordenadordo Curso

Banca Examinadora:

Documento assinado digitalmente
Cristian Cechinel
Data: 04/08/2022 20:02:54-0300
CPF: 454.628.449-72
Verifique as assinaturas em https://v.ufsc.br

Prof. Cristian Cechinel, Dr.

Orientador

Universidade Federal de Santa Catarina

Documento assinado digitalmente
Fernando Jose Spanhol
Data: 05/08/2022 09:26:17-0300
CPF: 642.656.419-20
Verifique as assinaturas em https://v.ufsc.br

Prof. Fernando José Spanhol, Dr.

Universidade Federal de Santa Catarina

Documento assinado digitalmente
EMANUEL MARQUES QUEIROGA
Data: 04/08/2022 20:05:11-0300
CPF: 013.743.590-84
Verifique as assinaturas em https://v.ufsc.br

Emanuel Marques Queiroga, Dr.

Instituto Federal Sul-rio-grandense

Este trabalho é dedicado aos meus amigos, familiares, professores e aos meus queridos pais.

AGRADECIMENTOS

Agradeço a todos que me guiaram até esse momento, especialmente meus pais Terezinha Clair Carvalho e Everaldo Balejo de Souza que sempre me apoiaram. Ao meu irmão, Daniel Elpidio Carvalho de Souza que sempre me incentivou e esteve ao meu lado. Agradecimento, também, aos meus amigos que sempre estiveram ao meu lado, especialmente Harlan Fabbris Boff que esteve comigo boa parte do meu ensino superior e médio técnico. A minha namorada e companheira Dayani Cristina Teixeira Maciel que sempre está ao meu lado. E, especialmente a Cristian Cechinel que acreditou em mim e nesse trabalho, além de todo apoio e incentivo que me deu como professor e orientador.

RESUMO

Este trabalho visa analisar os padrões dos dados de uma ferramenta de coleta de interações de alunos em projetos da plataforma Scratch realizada pela UFPEL, Universidade Federal de Pelotas, do Rio Grande do Sul, juntamente com o estudante Jorge Nachtigall. Foram estudados para realização deste trabalho métodos da mineração de dados como associação, agrupamento e classificação. Além disso, com os dados, da ferramenta de coleta de interações, disponibilizados em JSON(JavaScript Object Notation) pela UFPEL, foi importado as informações das coletas para o Python a fim de realizar a aplicação do K-means, algoritmo de agrupamento e, além disso, foi utilizada a análise de padrões desses dados com o WEKA, plataforma que permitiu realizar a aplicação de teste do k-means pelo ângulo das médias de coeficientes além de comprovar os dados do código desenvolvido para o algoritmo de clusterização. O objetivo deste estudo é promover o uso e estudo de técnicas de mineração de dados a fim de que futuramente esse projeto possa auxiliar professores que utilizam o Scratch em suas aulas para prever resultados e analisar os dados adquiridos das interações realizadas pela coleta de interações da ferramenta.

Palavras-chave: Mineração, K-Means, Scratch, Aluno, Agrupamento.

ABSTRACT

This work aims to analyze the data patterns of a tool for collecting student interactions in projects on the Scratch platform carried out by UFPEL, Universidade Federal de Pelotas, from Rio Grande do Sul, once with the student Jorge Nachtigall. The study was carried out using data mining methods such as association, grouping and classification. In addition, with the data from the interaction collection tool, made available in JSON by UFPEL, the application of K-means, clustering algorithm, was developed in Python and, after that, the analysis of patterns of these data was performed with WEKA, a platform that allowed testing other algorithms and confirming the response data of the developed code. The objective of this study is to promote the use of data mining techniques in order to, in future, help teachers who use Scratch in their classes to predict results and analyze the data acquired from interactions carried out by collecting the tool's interactions.

Keywords: Mineração, K-Means, Scratch, Aluno, Agrupamento.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

TCC – Trabalho de Conclusão de Curso

MIT - (Massachusetts Institute of Technology

UFPEL - Universidade Federal de Pelotas

KDD - Knowledge Discovery in Databases

JSON – JavaScript Object Notation

SUMÁRIO

1 INTRODUÇÃO

No século XXI, o âmbito educacional, de ensino básico, está sendo composto por alunos que cresceram após o surgimento de tecnologias como: a internet, os computadores modernos e os smartphones. O impacto dessa evolução tecnológica marcou a mudança de perfil no aprendizado dos alunos. Atualmente, o aluno tem características novas como: a capacidade de realizar mais tarefas ao mesmo tempo, postura mais ativa em relação a busca da informação, necessidade de motivação como feedbacks constantes (NAVEAVELA, 2018). Segundo dados levantados pelo fórum econômico mundial, 65% dos empregos dos estudantes de ensino básico, atualmente, ainda não existem (WEF,2018).

O ensino de programação, em todo mundo, tem aumentado sua demanda justamente por ser considerada um dos possíveis “empregos do futuro”, mas além disso pela capacidade de desenvolver o aluno em diversos aspectos. A aprendizagem de códigos já faz parte do currículo obrigatório em países como Austrália, Reino Unido, Estados Unidos e Alemanha. Os jovens dos países que utilizam a programação como disciplina obrigatória, já demonstram melhorias em habilidades como organização, compreensão de conceitos matemáticos, hábitos de pesquisa, resolução de problemas, comunicação e uso de tecnologia (IDOCODE, [201-?]).

Para auxiliar a aprendizagem de programação do estudante do segmento infantil e adolescente, a plataforma de ensino Scratch contém um modelo mais simplificado e visual de ensino permitindo a criação de jogos e projetos de forma facilitada (CTRLPLAY, [201-?]). O programa é adquirido de forma gratuita e possui uma linguagem baseada em blocos que ativa no jovem o interesse pela resolução dos projetos. Além disso, a ilustração em desenhos faz com que o entendimento dos códigos esteja mais próximo do público alvo. Desta forma, o interesse e a procura pelo conhecimento de programação no jovem iniciante é alcançado. Porém, em determinados casos o ensino de algoritmos pode se tornar complexo por sua cadeia de acontecimentos dentro dos códigos. A complexidade no início da aprendizagem pode desestimular o aluno e fazer com que ele perca o interesse na aprendizagem de programação.

No dia a dia pedagógico, utilizando plataformas práticas de ensino, como o Scratch, a coleta dos dados de interações dos alunos pode auxiliar o professor a identificar padrões de desempenho. O estudante Jorge Nachtigall da UFPEL, no seu TCC intitulado “Um Framework para Coleta de Interações em Projetos Scratch” detalhou a criação de uma plataforma de coleta de dados utilizada para minerar as interações dos alunos (VAZ JUNIOR,2019).

Dessa forma, caso o padrão de desempenho do discente seja negativo, é possível intervir, como professor, tentando auxiliar o aluno a encontrar, ou estimular, o seu estilo de aprendizagem e converter a situação problema. A computação, no geral, tem esta virtude de transformar dados cotidianos aglomerados em informações que organizadas auxiliam a ter um mapa comportamental sobre as interações ocorrentes. Desta forma, com tecnologias computacionais envolvendo *data-mining, big-data* e outros, é possível utilizar de dados cotidianos, utilizados em plataformas de ensino, como o Scratch, para auxiliar o professor encontrando os padrões das interações indicando possíveis resultados.

1.1 JUSTIFICATIVA

A plataforma de ensino de programação Scratch auxilia jovens do mundo inteiro a terem mais interesse pelo assunto e desenvolver suas habilidades lógicas. O programa possui ilustrações e uma linguagem própria de construção de blocos que se adequa ao público alvo. Porém, a utilização desse tipo de método de ensino, muitas vezes, faz com que o professor se depara com a situação de não ter como identificar o que cada aluno está passando naquele momento de aprendizagem.

Nesse contexto, para resolver a situação seria necessário realizar a coleta dos dados de todos os computadores obtendo as informações das relações entre aluno e computadores. Em seguida, além de obter as informações necessárias, a análise das informações a fim de saber como está o progresso de aprendizagem do aluno relacionando os dados de cada aluno para realizar a análise das interações.

Nesse sentido, o trabalho vem propor o estudo das práticas de mineração de dados para que no futuro seja possível interpretação dos dados de interação dos alunos utilizando a plataforma Scratch possibilitando uma melhor experiência de

ensino para o aluno e o professor. Tornando viável a interpretação detalhada do progresso de ensino do aluno.

1.2 OBJETIVOS

Nas seções abaixo estão descritos o objetivo geral e os objetivos específicos deste TCC.

1.2.1 Objetivo Geral

O trabalho tem como objetivo analisar as informações já resgatadas da ferramenta de coleta de interações de criada pelo Jorge Nachtigall da UFPEL do TCC intitulado “Um Framework para Coleta de Interações em Projetos Scratch” (VAZ JUNIOR,2019). Além disso, o trabalho busca estudar diferentes tecnologias de mineração de dados e analisar a aplicação das técnicas afim de verificar os resultados adquiridos.

1.2.2 Objetivos Específicos

- Estudar o que é a plataforma scratch, como ela contribui com o estudo de programação e trazer dados de utilização do programa no Brasil e no mundo;
- Estudar a coleta de dados realizada pela UFPEL, a fim de entender como interpretar as construções de bloco de código do Scratch;
- Estudar diferentes tecnologias de mineração de dados, principalmente utilizando a técnica de agrupamento com o algoritmo k-means e a plataforma WEKA, a fim de compreender como interpretar e analisar os dados da plataforma da UFPEL;
- Avaliar os resultados obtidos analisando a contribuição do estudo e do desenvolvimento;

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo irá descrever o objetivo da plataforma scratch, além de demonstrar o conceito de aprendizado de máquina e as diferenças sobre aprendizado supervisionado e não supervisionado, por fim será descrito o conceito da mineração de dados e também a demonstração de algumas técnicas.

2.1 PLATAFORMA SCRATCH

O Scratch é uma plataforma de ensino à programação de computadores voltado para o público jovem de 8 à 16 anos, projetado pelo grupo Lifelong Kindergarten na universidade do *MIT(Massachusetts Institute of Technology) e* é desenvolvido pela Scratch Foundation (SCRATCH BRASIL, [20--]). O objetivo da plataforma é elevar as capacidades de raciocínio, criatividade e colaboração, entre outras, com uma gama de recursos visuais que permitem o estudante a criar jogos, histórias e animações de maneira simplificada (SCRATCH, [20--]). A tecnologia é adquirida de forma gratuita e não possui fins lucrativos, sua comunidade tem cerca de 82 milhões de usuários e 92 milhões de projetos criados, sendo no brasil 1,35 milhões de usuários ativos, cerca de 1,76% do total no mundo (SCRATCH, [20--]).

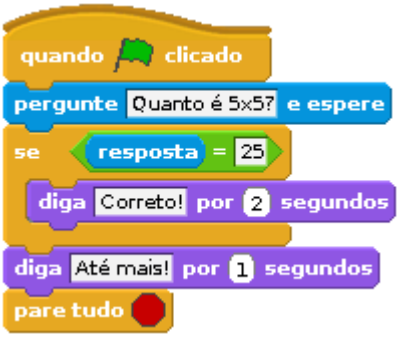
A seguir, Oliveira *et al.* (2014) detalha como é operacionalizado a linguagem de programação Scratch e quais habilidades são elevadas na utilização da plataforma de aprendizagem.

> Deste modo são estimuladas a criatividade e a imaginação, não tratando o aprendiz apenas como usuário do software. As atividades são desenvolvidas a partir de blocos que se encaixam e são divididos em 8 categorias: Movimento, Aparência, Som, Caneta, Sensores, Controle, Operadores e Variáveis. Aprender a lógica de programação se torna mais intuitiva e visualmente mais agradável, pois o próprio ambiente é voltado para computação criativa e design (OLIVEIRA et al., 2014, p. 1528).

Nesse sentido, o Scratch tem como um de seus princípios fazer com que o discente se concentre na construção da lógica de programação, sem precisar estudar a "sintaxe", linguagem de algoritmo de alguma linguagem. Dessa maneira, a linguagem desenvolvida a partir de blocos trabalhará a criação do pensamento computacional dentro da aprendizagem do aluno, pois ele estará voltado a entender

a lógica de uma maneira descomplicada. A seguir será demonstrado como é construído dentro da plataforma os blocos de código.

Figura 1 - Exemplo básico da linguagem de blocos do Scratch



Fonte: IDOCODE ([201-?]),

A linguagem de blocos formaliza conceitos de condição, estruturas de repetição, funções e outros conceitos de programação de uma maneira simplificada. Para o iniciante, a motivação de ver seus códigos sendo operacionais em ilustrações e de fato funcionando tornam essa motivação um combustível para a busca desse conhecimento.

2.2 APRENDIZADO DE MÁQUINA

O aprendizado de máquina é um subcampo da inteligência artificial, área que trabalha a capacidade de dispositivos aprenderem por conta própria. No inglês o termo conhecido para o aprendizado de máquina é *Machine Learning.* Esse subcampo, tem o intuito de trabalhar com milhares de dados e interpretar seus padrões via treinamentos, utilizando modelos, provenientes de algoritmos de aprendizado (FREITAS, 2019).

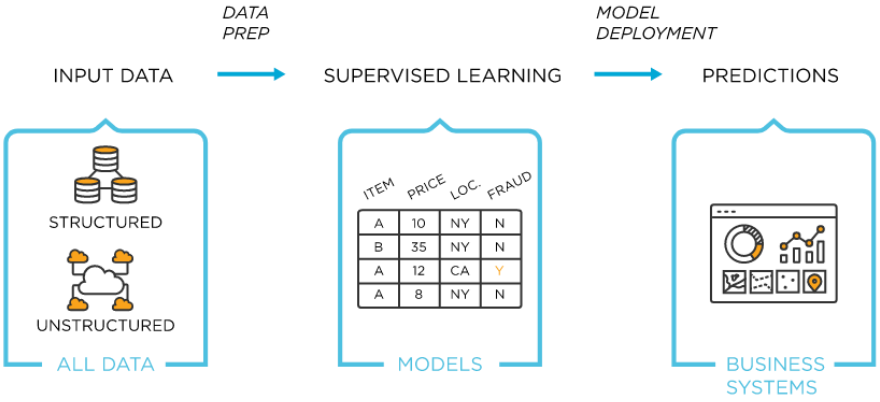
Para o *machine learning* interpretar padrões dos dados que serão apresentados, é necessário que haja um modelo que irá reconhecer as interações desses dados. O modelo de *machine learning* é um arquivo que foi treinado para entender certos padrões. O treinamento do conjunto de dados é realizado através de algoritmos que interpretam as relações entre os dados. Após isso, é possível analisar os dados com técnicas que poderão prever resultados (RADICH *et al.*,

2022). A seguir serão demonstradas duas categorias do aprendizado de máquina: aprendizado supervisionado e não supervisionado.

2.2.1 Aprendizado Supervisionado

A aprendizado supervisionado ocorre quando são passados valores que definem o padrão daquele objeto. Esse aprendizado, com o valor de referência, consegue rotular os padrões definidos para aquele objeto de referência fazendo com que o modelo seja ensinado. Dessa maneira, esses valores supervisionam os resultados de saída com base nos erros que já ocorreram (TECH, [201-?]). Ou seja, para utilizar esse modelo de aprendizado é necessário possuir dados anteriores de referência que definem o objeto em questão. A seguir, a Figura 2 demonstrará as etapas de preparação de dados, supervisão e previsão de resultados dentro do aprendizado supervisionado.

Figura 2 - Exemplo de aprendizado supervisionado

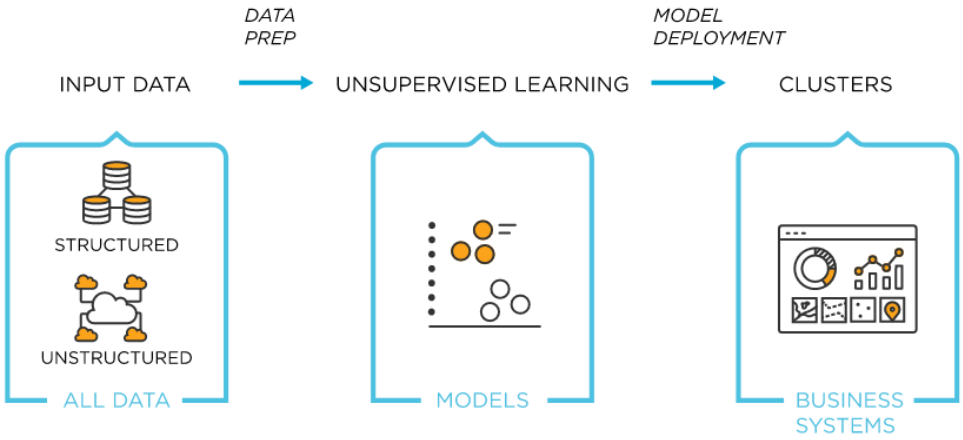


Fonte:TIBCO ([201-?]).

2.2.1 Aprendizado Não Supervisionado

O aprendizado não supervisionado, diferentemente da supervisionada, não possui valores rotulados para que a máquina guie seus resultados. Nesse sentido, a máquina irá processar e comparar dados sem referência de valores, por conta própria, ajustando e agrupando dados quando forem adequados. Esse aprendizado permite a exploração de um grande conjunto de dados desconhecido, possibilitando a análise e a segmentação desses dados (TIBCO, [201-?]). A seguir, a Figura 3 demonstrará as etapas de preparação de dados, análise de dados sem supervisão e visualização de resultados dentro do aprendizado não supervisionado.

Figura 3 - Exemplo de aprendizado não-supervisionada



Fonte: TIBCO ([201-?]).

2.3 MINERAÇÃO DE DADOS

A mineração de dados, de maneira simplificada, é a extração de um local que possui um volume elevado de dados. A mineração de dados é considerada uma das etapas da KDD (*Knowledge Discovery in Databases* ou Descoberta de Conhecimento nas Bases de Dados) neste trabalho foram utilizadas cinco etapas, que são: Limpeza dos dados, onde é eliminado inconstâncias; Seleção, definição das informações relevantes a serem coletadas; Transformação dos Dados, etapa onde os dados são adequados para seu formato; Mineração, etapa onde é realizada

o processo de aplicação de técnicas a fim de extrair um padrão de dados de seu interesse e visualização dos resultados, etapa onde é visualizado a extração dos dados (AMO, 2004).

3.3.1 Técnicas De Mineração De Dados

Como mencionado na seção anterior, a etapa da mineração de dados é o momento onde é realizado o processo da aplicação de uma técnica que irá se adaptar aos padrões que serão interessantes serem visualizados. Nesta seção, será apresentado a Análise preditiva e Análise descritiva e serão selecionadas algumas das técnicas que envolvem cada uma dessas análises e que possuem relevância dentro do estudo dos dados.

2.3.1.1 Análise Preditiva

Esta análise organiza os dados, a fim de prever os resultados futuros com base em probabilidades, decorrentes de relações anteriores. Ou seja, a predição de dados faz afirmações de resultados que ainda não ocorreram (PATH, 2020). A seguir será apresentada as técnicas de análise preditiva: classificação e clusterização.

2.3.1.1.1 Classificação

Esta técnica consiste em classificar um conjunto de dados(modelos/funções) que determinam valores, decorrentes do passado, relacionando os dados com a classe de objeto por conta do desempenho dos dados, por exemplo. Os modelos são predicados para uma classe que ainda não teria sido classificada (AMO, 2004).

> Por exemplo, suponha que o gerente do supermercado está interessado em descobrir que tipo de características de seus clientes os classificam em bom comprador ou mau comprador. Um modelo de classificação poderia incluir a seguinte regra: Clientes da faixa econômica B, com idade entre 50 e 60 são maus compradores.Em algumas aplicações, o usuário está mais interessado em predizer alguns valores ausentes em seus dados, em vez de

> descobrir classes de objetos. Isto ocorre sobretudo quando os valores que faltam são numéricos (AMO, 2004).

Na etapa da mineração de dados, a classificação inicia-se na predefinição de novos indícios de fatos ou objetos submetidos. A criação de novas classes necessita que apenas as principais sejam relevadas. A tarefa de classificar normalmente ocorre na comparação de um objeto com outro objeto que possa ter pertencido a classes definidas anteriormente. A comparação entre objetos ocorre na métrica das diferenças entre eles. Nesta técnica os três algoritmos mais utilizados são: Árvores de decisão, regressão e redes neurais (AMORIM, 2007).

2.3.1.1.2 Clusterização

A técnica de mineração de dados chamada Clusterização, diferentemente da técnica de classificação, não utiliza a pré-definição de classe. Os registros são agrupados pela semelhança entre suas características. Segundo Castanheira (2008), normalmente, a clusterização ocorre de maneira preliminar, a fim de segmentar um conjunto de dados com inúmeros padrões comportamentais, após o agrupamento, outra técnica classifica os dados já categorizados. Para Sandra de Amo (2004), um cluster é uma coleção de objetos que são semelhantes uns aos outros e diferentes aos objetos de outros grupos/clusters (AMO, 2004).

De acordo com StringFixer ([20--]), a clusterização é uma técnica de exploração de dados que analisa a estatística dos dados usada em diversas áreas como: reconhecimento de padrões, análise de imagens, recuperação de informações, bioinformática, compressão de dados, computação gráfica e aprendizado de máquina. Os clusters são interpretados de diversas formas, por conta disso existem dezenas de algoritmos publicados. A decisão por qual algoritmo de clusterização utilizar ocorrerá pela motivação de como serão agrupados os dados.

A fim de demonstrar algumas das opções de modelos existentes, como exemplo, serão descritos alguns dos típicos “modelos de cluster”: Cluster Baseado em conectividade: Agrupamento hierárquico, construído pela conectividade baseada

na distância entre os clusters; Agrupamento baseado em centroide: Mais conhecido como algoritmo k-médias, cada cluster possui um vetor médio único, um exemplo popular de algoritmo utilizado nesse modelo é o k-means; Agrupamento baseado em distribuição: Os clusters são modelados por balanceamento estatístico, com distribuições normais multivariadas utilizado pelo algoritmo de maximização de expectativa; Agrupamento baseado em densidade: Define as regiões mais densas ligadas ao espaço de dados dos cluster, em exemplo é possível utilizar o DBSCAN ÓPTICA para analisar as áreas mais densas de cluster (STRING FIXER,[2021]).

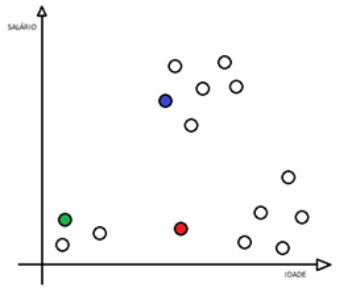
2.3.1.1.3.1 Algoritmo K-means

Esse algoritmo pertence a técnica de clusterização, esta técnica, citada anteriormente, tem a característica de não possuir rótulos/valores padrões de referência em seu aprendizado, por conta disso, é possível analisar que o k-means é um algoritmo de aprendizado não supervisionado. Ou seja, não existe a relação das características da classe a uma variável a ser prevista. Os clusters, como explicado anteriormente, são grupos de pontos com características similares ao seu cluster e diferente dos pontos de outros clusters. Esse algoritmo identifica através de alguns cálculos que serão introduzidos, principalmente, como é identificado o centro do clusters e os extremos (começo e término) de um agrupamento (GASPERI, 2020).

Primeiramente, baseado no estudo de Gasperi (2020), o algoritmo inicia-se escolhendo um centroide para o cluster, de maneira aleatória, normalmente começando dos extremos do grupo e após a repetição do algoritmo sendo deslocado até o centro. O próximo passo é comparar cada ponto ao centroide e definir por aproximação a qual cluster o ponto pertence. Após definido para quais clusters cada ponto pertence, é necessário realizar a média referente a posição do centro do grupo e assim definir um novo centroide. Dessa forma, o algoritmo k-means irá repetir o segundo processo e o último até encontrar o centroide perfeito e definir todos os grupos de maneira correta. Na figura 4 será ilustrado o primeiro passo do algoritmo, a inicialização dos centroides, já na 5, foi ilustrado o exemplo de cálculo da distância entre os centroides e por fim na figura 6 foi ilustrado o último

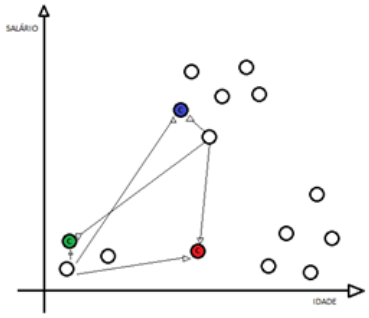
processo do fluxo, onde é demonstrado como é feito a média entre os pontos de cada cluster.

Figura 4 - Algoritmo K-means. Inicializando centroides



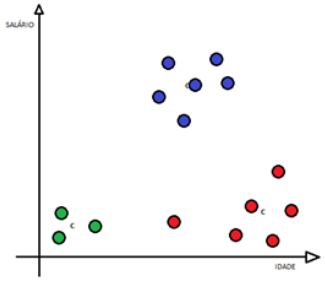
Fonte: GASPERI, (2020).

Figura 5 – Cálculo da distância entre os centroides



Fonte: GASPERI, (2020).

Figura 6 – Cálculo da média da distância entre os pontos



Fonte: GASPERI, (2020).

2.3.1.2 Análise Descritiva

Esta análise detalha o histórico de dados armazenados anteriormente, a fim de realizar estimativas e relatórios sobre os resultados que já ocorreram (PATH, 2020). A seguir será apresentada a técnica descritiva: Análise de regras de associação.

2.3.1.2.1 Associação

A técnica de associação, segundo Sandra de Amo (2004), é um padrão da forma X → Y, onde a ação do conjunto de valores X resulta no conjunto de valores Y. Na citação a seguir será mostrado um modelo de *machine learning* onde o conjunto de dados associa, em um mercado, o ato do cliente de comprar pão com o ato de comprar leite, resultando em uma contribuição significativa ao mercado.

> Consideremos por exemplo um supermercado. O seguinte padrão: Clientes que compram pão também compram leite representa uma regra de associação que repete um padrão de comportamento dos clientes do supermercado. Descobrir regras de associação entre produtos comprados por clientes numa mesma compra pode ser útil para melhorar a organização das prateleiras, facilitar (ou dificultar) as compras do usuário ou induzi-lo a comprar mais (AMO, 2004).

3 TECNOLOGIAS UTILIZADAS NO PROJETO

Neste capítulo será descrito sobre o framework utilizado para extração dos dados do Scratch pela UFPEL, além da linguagem de programação utilizada no projeto o Python, em seguida será apresentada as bibliotecas utilizadas dentro da linguagem para realizar a clusterização dos dados da ferramenta da UFPEL. Por fim, iremos apresentar também a ferramenta que auxiliou na confirmação de resultados das técnicas de mineração de dados WEKA.

3.1 FERRAMENTA DE COLETA DE INTERAÇÕES DE PROJETOS DO SCRATCH

De acordo com VAZ JUNIOR (2019), em seu TCC(Trabalho de Conclusão de curso) intitulado de “Um Framework para Coleta de Interações em Projetos Scratch” para a realização do projeto foi necessário modificar a plataforma Scratch inserindo um framework que coletava os dados dentro do programa. O trabalho apresentou um estudo sobre o funcionamento dos blocos internos do Scratch afim de capturar as interações dos alunos nas montagens de blocos e pré-visualização na ferramenta.

Desse modo, cada bloco é separado por suas funções, respectivamente definidas por: Motion: controle de movimentos dos elementos;Looks:controle visual de um elemento, altera a aparência do objeto; Sound: gerência do controle de som do elemento; Events: controle da inicialização de de uma ação codificada, reação de interação entre usuário e objeto; Sensing: gerência da ação de sensoriamento verificando qual usuário realizou a ação;Control: função que possibilita a gerência de condições e repetições de ações; Operators: Define os blocos de operadores(soma, subtração, divisão, multiplicação, operadores lógicos(verdadeiro ou falso), etc); Varibles: Define os blocos de variáveis gerência os valores contidos.

Na composição dos blocos do Scratch é transcrita a estrutura de texto JSON(JavaScript Object Notation), formato que representa um dado ou uma informação de maneira leve e objetiva, onde específica a representação da interação contida no bloco criado. Na figura 7 é possível observar que a estrutura inicia-se com

a chave identificadora “_blocks” e em seguida apresenta as características principais do bloco que são: "opcode": Demonstra qual bloco está sendo utilizado pelo formato "categoria_nomeDoBloco"; "inputs": Campo utilizado para guardar parâmetros; "fields": Guarda informações de blocos que possuem parâmetros especiais; "next": Campo que referência o próximo bloco, caso houver conexão • "parent": Campo faz referência ao bloco anterior, caso houver;"topLevel": Representa um valor verdadeiro ou falso identificando se é o início de um fluxo de execução de código; "x": define o valor da coordenada x do bloco no workspace;"y": define o valor da coordenada y do bloco no workspace.

Figura 7 - Estrutura de um bloco Scratch

Fonte: VAZ JUNIOR (2019).

Nesse sentido, para realizar a ferramenta de coleta de dados foi necessário, também, analisar os componentes internos do Scratch 3.0 para entender melhor o funcionamento da ferramenta. O estudo desses componentes viabilizou a aplicação do framework de coleta de interações, pois com esse entendimento foi possível captar e traduzir as informações trazidas externamente na plataforma. O primeiro componente estudado foi o scratch-gui, ele corresponde a toda interface

gráfica e dinâmica da ferramenta pois além de criar e executar projetos, com esse componente também é possível que haja a interação do usuário com os blocos.

Outro componente importante é o scratch-blocks, ele é responsável por todos mecanismos e funcionalidades de bloco do Scratch 3.0 desde a parte visual até o encaixe, funcionalidade e parâmetros envolvendo os blocos. Além disso, através do scratch-blocks é possível identificar as interações que estão sendo realizadas pelos alunos que estão utilizando a plataforma. Por fim, o componente scratch-vm é o responsável por executar os códigos representados nas montagens dos blocos realizando um trabalho em conjunto com o componente scratch-blocks. Dessa maneira, sempre que executado um evento pelo scratch-blocks o scratch-vm realiza os ajustes necessários.

Dessa forma, o trabalho de conclusão de VAZ JUNIOR (2019) através do último componente apresentado, scratch-vm, foi possível interpretar, principalmente, três eventos de interações de mudança de código que são eles: "BLOCK_DRAG_UPDATE": Evento ocasionado para cada interação de um usuário com um bloco presente no workspace. Podendo ser a adição, remoção ou remanejo do bloco no código;"TARGETS UPDATE": Evento ocasionado por interação do usuário dentro da área de pré-visualização, onde é possível testar o projeto e interagir com os atores; "PROJECT_START": Evento de execução do projeto, ilustrado por uma "bandeira verde" pelo Scratch.

A partir do entendimento dos componentes do Scratch 3.0 e dos eventos compostos nos blocos de código da ferramenta foi realizada uma versão modificada. A UFPEL, com o estudante Jorge, conseguiu realizar o desenvolvimento da modificação do Scratch 3.0 por conta da plataforma ter seu código aberto, a partir do repositório oficial do Github. A modificação do Scratch tornou possível realizar a coleta de todas interações dos usuários com a ferramenta, e da relação do código no momento da modificação. A alteração tornou possível a armazenação dos eventos emitidos pelo scratch-blocks, bloco responsável pela percepção das modificações feitas pelos usuários em seu local de trabalho na ferramenta. Nesse sentido, será citado pelas palavras do autor, como no componente scratch-vm foi implementada uma nova função no código-fonte, onde permitiu armazenar as informações de mudanças nos blocos para banco de dados do framework. No

quadro 1 será demonstrado a descrição das funções de entrada no banco de dados do framework.

> Desta maneira, foi adicionada uma nova função ao código de implementação da scratch-vm, onde sempre que o usuário adiciona, remove ou troca um bloco de lugar, bem como tenta executar seu programa, uma nova entrada no banco de dados do framework é gerada através de um servidor que recebe pacotes contendo a identificação deste usuário, o horário em que a ação ocorreu, o estado atual do código através do arquivo JSON mantido pela scratch-vm, e qual evento desencadeou esta nova entrada no banco de dados (JÚNIOR, 2019).

Quadro 1 - Descrição das funções de entrada no banco de dados do framework.

FUNÇÃO	Descrição
BLOCK_DRAG_UPDATE	Modificação de blocos
PROJECT_START	Tentativa de Execução do código a partir do aluno
TARGETS_UPDATE	Tentativa de interação com o autor da área de pré-visualização do projeto

Fonte: elaborado pelo autor (2022).

Para realizar os testes do framework, a UFPEL juntamente com Jorge e seus orientadores planejaram uma oficina. A atividade consistiu em desenvolver uma máquina do tempo utilizando a versão modificada da ferramenta Scratch com a captura das interações dos alunos. Para isso, foi disponibilizado um material contendo um projeto Scratch pré-montado com elementos básicos(atores, fantasias e painel) sendo eles: Uma chave, utilizada para ligar a máquina; quatro botões amarelos: responsáveis por navegar por épocas; Uma ignição: Utilizada com a chave para ligar a máquina do tempo; Luz vermelha: Identifica se está a máquina está ligada; Quatro cenários: responsáveis por simular épocas diferentes.

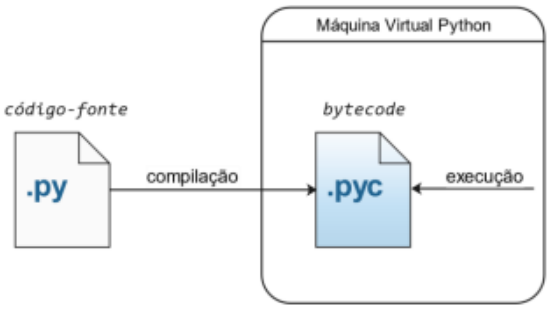
> Os alunos devem montar o painel da sua máquina do tempo e programar os elementos presentes neles de forma que, ao interagir com os botões, seja possível "viajar" entre diferentes épocas, mudando o cenário do seu projeto. Para alcançar este objetivo final, os alunos passam por diversas etapas de desenvolvimento, onde cada uma destas etapas consiste na implementação de novas funcionalidades para a máquina do tempo (VAZ JUNIOR,2019).

Dessa maneira foram elaboradas etapas com um material de apoio, contendo um fluxo de Scratch Cards, semelhante ao desenvolvido na ferramenta original, porém os cartões que resolvem a tarefa não estavam em ordem, fazendo com que o aluno tenha que resolver da melhor forma utilizando a dica dentro de seu projeto.

3.2 LINGUAGEM PYTHON

A linguagem de programação Python, teve como criador principal Guido Van Rossum que nos anos 80 trabalhava na criação da linguagem “ABC” juntamente com o SO (Sistema Operacional) “AMOEBA”. No meio tempo do desenvolvimento, Guido a fim de possuir uma linguagem de sintaxe semelhante a “ABC”, porém com acesso semelhante ao "AMOEBA" trabalhou até 1991 na conclusão da primeira versão do Python 1.0. Segundo Soubhia *et al.* (2019), a compilação/interpretação do Python é traduzida em bytecode, formato binário com instruções para o interpretador. Em seguida, o interpretador compila o código e armazena o bytecode em disco,e então analisa o código, convertendo-os para símbolos padrões, compila e executa na máquina virtual do Python (Virtual Machine) (SOUBHIA *et al.*, 2019).

Figura 8 - Estrutura da compilação/interpretação pelo Python



Fonte: DOC PLAYER, (2020).

O Python é multiplataforma funcionando em diversos sistemas operacionais como Linux, Windows e Mac além de ser multiparadigma suportando funções, imperatividade, modelo procedural e orientada a objeto (BORGES, 2014).

Dessa maneira, o Python tem atuação em diversas áreas como na indústria de games, sistemas web, análise de dados científicos etc. Porém a principal área de aplicação de crescimento da linguagem são as análises de bases de dados (Big Data), machine learning e inteligência artificial (DOCPLAYER, [201-?]).

3.3 BIBLIOTECAS E FERRAMENTAS UTILIZADAS

Nesta seção serão citadas as principais bibliotecas utilizadas na linguagem Python para realizar o estudo e a execução de técnicas de mineração de dados como o k-means.

3.3.1 NUMPY

Esta biblioteca é utilizada no Python, principalmente, para fazer operações matemáticas em Arrays Multidimensionais. O NumPy contém uma gama de funções e operações para realizar cálculos numéricos. As tarefas mais comuns para esta biblioteca são: Em modelos de machine learning para armazenar treinamentos e parâmetros dos modelos; No processamento de imagem auxiliando com seus Arrays Multidimensionais de números, além de conter funções de manipulação de imagem e Tarefas matemáticas como extrapolação, interpolação, diferenciação, além de realizar métodos para álgebra linear e, também para geração de números aleatórios (SANTIAGO JUNIOR, 2018).

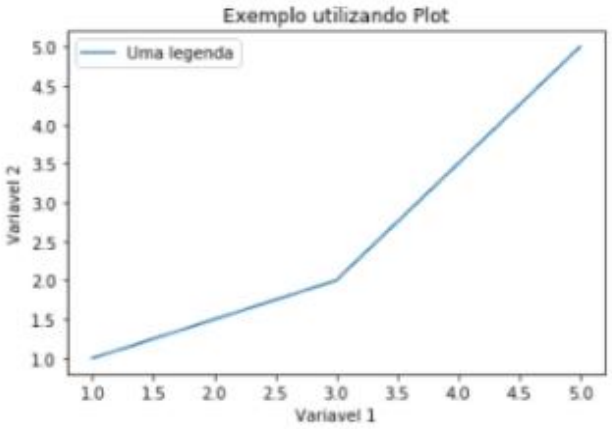
3.3.2 Matplotlib E O Pyplot

A biblioteca Matplotlib é utilizada para criação de aplicações de visualização de dados em Python. A ferramenta disponibiliza o serviço de criação de gráficos(2D) de pizza,barras,grafos, etc. O matplotlib oferece o pacote pyplot, essa interface permite a criação de gráficos de maneira simplificada e sem utilizar uma metodologia mais complexa para iniciantes como a Orientação Objeto (GEPAC, 2019). Em seguida, na Figura 8, será apresentado um exemplo de gráfico em linha, onde o código inicia-se com a importação do pacote pyplot da biblioteca matplotlib,

após isso, é escrito a definição da posição dos vértices do gráfico, depois, a atribuição das legendas verticais e horizontais e por fim a escrita de uma legenda interna. Dessa maneira é possível visualizar as possibilidades que esta ferramenta pode alcançar no âmbito de ilustração de imagens gráficas.

Figura 9 - Inicialização de um gráfico de linha com o Pyplot

```
import matplotlib.pyplot as plt

# Definindo variáveis
x = [1, 3, 5]
y = [1, 2, 5]

# Criando um gráfico
plt.plot(x, y)

# Atribuindo um título ao gráfico
plt.title('Exemplo utilizando Plot')
plt.xlabel('Variavel 1')
plt.ylabel('Variavel 2')

# Atribuindo uma legenda
plt.plot(x, y, label = 'Uma legenda')
plt.legend()

# Exibindo o gráfico gerado
plt.show()
```



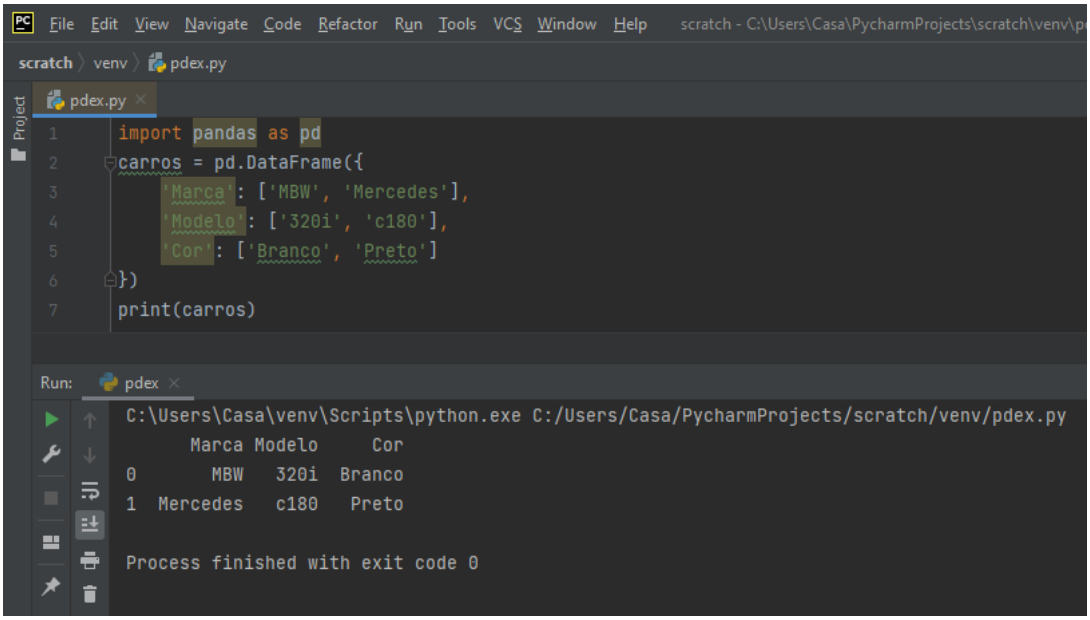
Fonte: LIMA (2018).

3.3.4 Pandas

A biblioteca pandas,derivada do nome *panel data*(painel de dados), foi desenvolvida em Python,com código aberto, a fim de realizar análises de dados. Sua base é construída com outras duas bibliotecas: a Matplotlib para visualização dos dados e o NumPy para operações matemáticas. Desta forma, o pandas permite que

sejam acessados muitos métodos dessas duas bibliotecas de maneira mais prática(com menos código). O Pandas apresenta dois objetos que armazenam dados o: Panda Series detém formato de lista, e DataFrame de estruturar tabular (COUTINHO, 2021). Em seguida, será demonstrado na Figura 9 a execução de um código de Data Frame feito em pandas.

Figura 10 - Execução de uma lista de DataFrame feita com pandas



Fonte: elaborado pelo autor (2022).

3.3.5 Scikit-Learn

A scikit-learn é uma biblioteca de código aberto desenvolvida em Python, para a aplicação de práticas de machine learning, com suporte para o aprendizado supervisionado e não supervisionado, podendo auxiliar na análise preditiva de dados. Esta ferramenta foi construída sobre os pacotes: NumPy, SciPy e Matplotlib e suas principais aplicações, são: Pré-processamento, Classificação, Regressão, Clusterização, Redução da dimensionalidade e Ajustar parâmetros (TECH, [201-?]).

Esta biblioteca oferece diversos algoritmos e modelos de machine learning, chamados de *estimators* (estimadores). É possível ajustar os dados dos estimadores utilizando o método fit(ajuste), normalmente utilizado com duas entradas: Matriz de amostras X, a quantidade de linhas na lista são o total de amostras e a quantidade correspondente às colunas são os recursos

(*n_samples*,*n_features*); Já, a outro entrada Y, corresponde a indicação dos índices da lista que correspondem os recursos das amostras, podendo indicar diretamente cada conjunto de amostras(classes) desses elementos (SCIKIT LEARN, [20--]).

Nesse caso, utilizando o método fit nessas entradas e introduzindo outro método chamado predict, é possível prever valores de novos dados sem precisar treinar novamente. Na figura à seguir será demonstrado um exemplo de aplicação estimadores, onde com a referência de dados importados é aplicado o método fit na matriz de amostras X (com seus valores de amostra e recurso) e Y apontando as classes de amostras correspondentes. Em seguida, é possível visualizar o conjunto de dados da primeira amostra (1,2,3), após aplicação do método predict tendo seus valores previstos para (4,5,6) (SCIKIT LEARN, [20--]).

Figura 11 - Exemplo simples de estimadores prevendo recursos de amostras

```
>>> from sklearn.ensemble import RandomForestClassifier
>>> clf = RandomForestClassifier(random_state=0)
>>> X = [[ 1,  2,  3],  # 2 samples, 3 features
...      [11, 12, 13]]
>>> y = [0, 1]  # classes of each sample
>>> clf.fit(X, y)
RandomForestClassifier(random_state=0)
```

```
>>> clf.predict(X)  # predict classes of the training data
array([0, 1])
>>> clf.predict([[4, 5, 6], [14, 15, 16]])  # predict classes of new data
array([0, 1])
```

Fonte: SCIKIT LEARN, ([20--]).

Com esta biblioteca é possível pré-processar os dados com um recurso chamado StandardScaler que oferece a opção de imputar ou transformar as colunas de amostras (Por exemplo: StandardScaler().fit(X).transform(X)), possibilitando posteriormente a utilização do modelo para predição desses dados, por exemplo. Realizando a combinação de transformadores e estimadores surgiu o recurso Pipelane, essa ferramenta possui recursos de segurança que evitam o vazamento de dados de treinamento. Por fim, para a avaliação de dados de modelos a biblioteca utiliza funções como train_test_split que divide conjuntos de dados e realiza testes,cross_validate que avalia dobras, realiza divisão de dados e utiliza

funções de classificação por pontuação, entre outros. Dessa maneira, foi demonstrado a capacidade de recursos que essa biblioteca pode realizar desde o pré-processamento, realizar previsão de dados, utilizar algoritmos de mineração de dados, avaliar modelos etc.

3.4 PYCHARM

O pycharm é um editor de código, principalmente da linguagem Python, com suporte para linguagens como javascript, kotlin e Coffe Script. Além da edição de códigos, o programa é uma IDE, podendo depurar, interpretar e utilizar outras ferramentas e bibliotecas de maneira facilitada. O pycharm pode ser encontrado no site da empresa JetBrains, instituição que detém os direitos do programa, podendo ser possível visualizar duas versões: uma freemium ou comunidade e outra premium (GARCIA, [202-?]).

3.5 WEKA

O WEKA (Waikato Environment for Knowledge Analysis) é um programa open source, desenvolvido em Java com o intuito de ser utilizado para mineração de dados. A sua interface, conhecida como Weka Explorer, permite realizar processos de mineração de dados com avaliação e comparação dos resultados com diversos algoritmos. Além disso, a plataforma possui recursos de pré-processamento dos dados relacionais (DEVMEDIA, 2012). O seu desenvolvimento ocorreu na Universidade de Waikato da Nova Zelândia, sendo de uso amigável e de processamento de dados rápido. Os principais tipos de algoritmo de mineração de dados que o WEKA possui são: regressão, classificação, associação, etc. (Sato, *et al*., 2013).

4 METODOLOGIA

Nesta seção será demonstrada as etapas que foram percorridas para geração da análise de padrões das interações dos alunos com a Plataforma Scratch. Para a realização do trabalho foram realizadas as seguintes etapas: Limpeza dos dados, Seleção, Transformação dos Dados, Mineração, Avaliação e Visualização. As principais etapas da análise dos padrões seguiram os métodos KDD (Knowledge Discovery in Databases ou Descoberta de Conhecimento nas Bases de Dados), citado da seção 2.3 do referencial teórico.

4.1 LIMPEZA DOS DADOS PARAS COLETA DAS INTERAÇÕES

A primeira etapa consistiu em eliminar as informações descartáveis do arquivo JSON enviado pela UFPEL, perante o objetivo da coleta, a fim de organizar e melhorar a semântica do arquivo para os próximos passos da análise. Primeiramente foram retirados caracteres indesejáveis do formato JSON. Logo em seguida, foram analisados a organização dos elementos (“event”,”state”,”time” e “xml”) do arquivo e retirado qualquer inconsistência. Desta forma, essa etapa preparou os dados para os próximos passos, principalmente para o seguinte, o da seleção, pois após feito a limpeza dos dados os elementos estavam prontos para serem manuseados de maneira aceitável.

4.2 SELEÇÃO

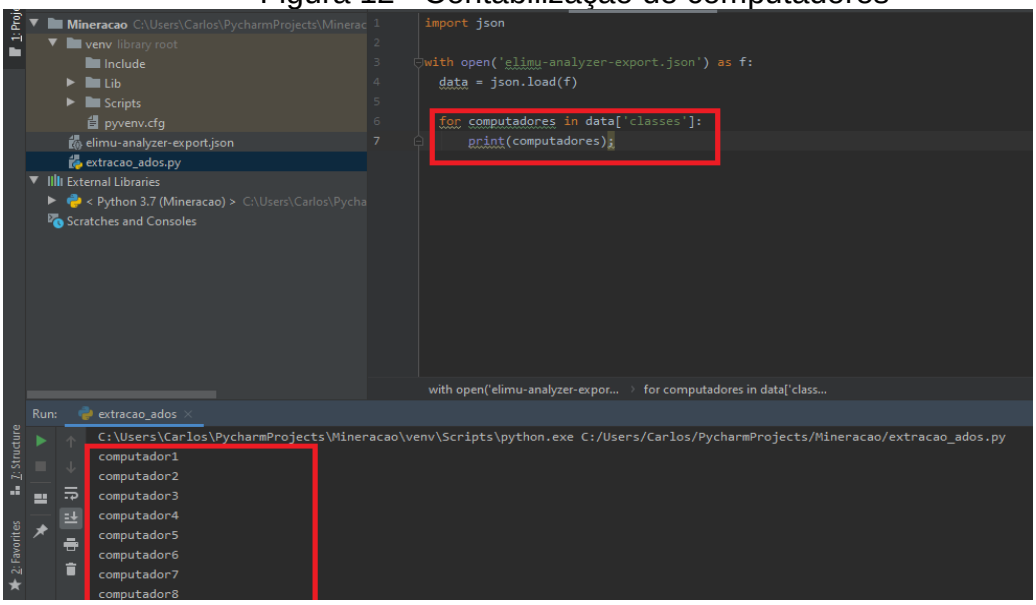
Esta etapa, conforme dito na seção 2.3, consiste em selecionar as informações mais relevantes a serem coletadas. Ao analisar o arquivo JSON e documentação, e ao decidir contabilizar quantidade de dos eventos foram selecionados as seguintes informações: "BLOCK_DRAG_UPDATE", citado na seção 3.1, onde sua composição pode ser interpretada por "ADD"(adição de bloco), "REMOVE"(remoção de bloco) e "DRAG"(reposição de bloco); “TARGETS_UPDATE”; ”PROJECT_START” e “PROJECT_RUN_STOP”. Após isso, foi feito um código em Python onde foi lido o arquivo JSON que apresentava os eventos citados e feito sua

contabilização.

4.3 TRANSFORMAÇÃO DOS DADOS

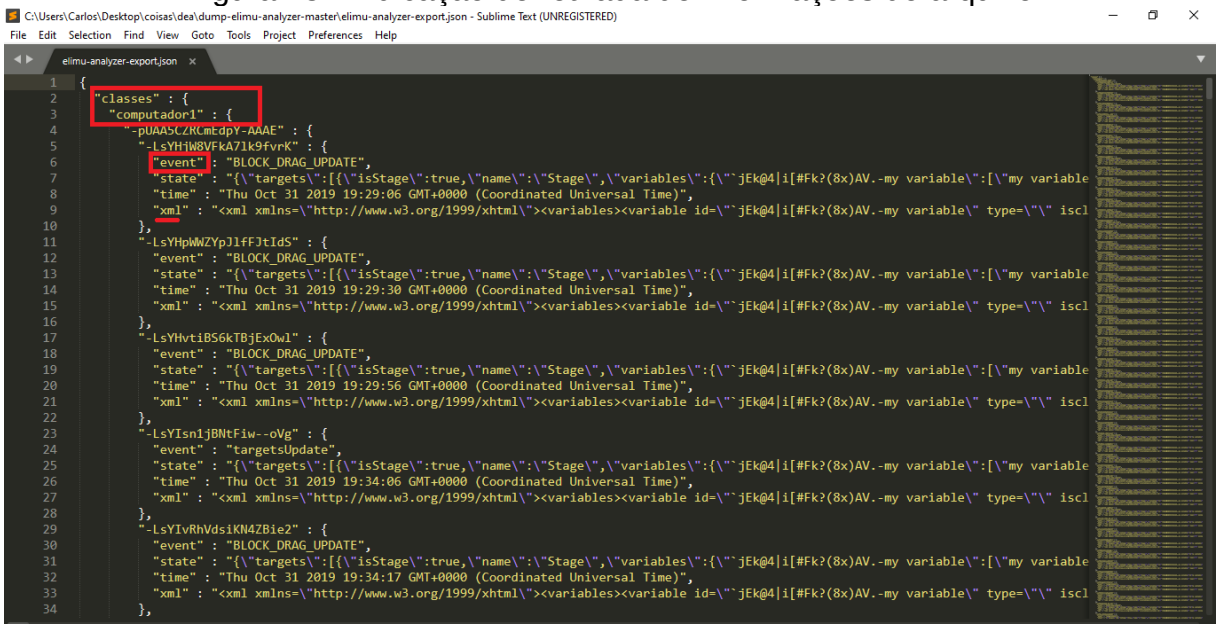
Esta etapa consistiu em transformar os dados para o formato adequado a fim de contabilizar, classificar e agrupar os dados do arquivo JSON. Elementos desnecessários que não fazem parte dos objetivos da mineração dos dados foram retirados, dessa maneira restando apenas as informações mais objetivas do arquivo. Com o auxílio da etapa anterior, a seleção, foi possível visualizar os dados mais importantes e dessa maneira retirar os dados desnecessários. Nesse sentido, um exemplo do que a transformação dos dados proporcionou foi a possível contabilização da quantidade de computadores do experimento, como na figura 10 demonstra a seguir. Além de retirar elementos destacados no JSON como o atributo “xml” que não oferecia nenhum dado de classificação dentro da análise, como a figura 12 demonstrará a seguir.

Figura 12 - Contabilização de computadores



Fonte: Elaborado pelo autor (2022).

Figura 13 - Indicação de retirada de informações do arquivo



Fonte: Elaborado pelo autor (2020).

4.3 MINERAÇÃO

Nesta etapa foram aplicadas as técnicas e algoritmos de identificação de padrões de dados. Nesse sentido, após pesquisado qual seria o melhor método que se adequa para a análise de padrões no aprendizado de programação com Scratch e foi escolhida a técnica de agrupamento (clusterização), utilizando o algoritmo k-means, e detalhado a clusterização dos dados, realizando simulações com o programa WEKA. O motivo da escolha desta técnica e algoritmo foi justamente a separação de grupos de padrões de dados que juntamente com o machine learning possibilitarão a predição de dados e a classificação final das informações coletadas no futuro. Na seção 5.1 e 5.2 do desenvolvimento será detalhado o processo de aplicação do algoritmo k-means e a análise de clusterização feita no WEKA respectivamente.

4.4 AVALIAÇÃO

Nesta Etapa ocorreu a avaliação da aplicação dos padrões de agrupamento. Após o processamento das informações oferecidas pela UFPEL foi avaliado os resultados dos dados de interações da plataforma Scratch. Nesse

momento, foi visualizado, revisado e refatorado as informações coletadas a fim de manter a integridade dos dados e retestar o desenvolvimento da técnica de clusterização e do algoritmo utilizado. Além disso, a avaliação auxiliou na comparação das informações visualizadas na etapa posterior.

4.5 VISUALIZAÇÃO

Na última etapa da metodologia, foram escolhidos os métodos de visualização dos resultados apresentados. Ao escolher o algoritmo k-means, o grupo de informações traçadas em grafos automaticamente tornou-se um mecanismo de visualização dos resultados da comparação entre a semelhança das interações entre alunos. Além disso, o gráfico de silhueta do k-means demonstra também mostra os resultados da pontuação do ponto k(centroide) dentro da visualização dos resultados apresentados. Por fim, com a confirmação dos dados pelo programa WEKA foi possível visualizar os resultados específicos da clusterização envolvendo cada grupo e evento.

5 DESENVOLVIMENTO

Nesta seção, será demonstrado o processo de aplicação da clusterização com o algoritmo k-means, juntamente com o detalhamento do seu código-fonte e processamento das informações no programa WEKA. Em seguida, serão exibidos os resultados, por meio de tabelas e figuras que demonstrarão as informações retidas da análise de padrões no aprendizado de programação com a plataforma scratch.

5.1 PROCESSO DE APLICAÇÃO DA CLUSTERIZAÇÃO COM O ALGORITMO K-MEANS

Nesta Etapa ocorreu a avaliação da aplicação dos padrões de agrupamento com o algoritmo k-means envolvendo a biblioteca sklearn com a linguagem Python. Após o processamento das informações coletadas foi avaliado os resultados das interações dos alunos que utilizaram a plataforma Scratch como meio de ensino da programação. Nesse momento, foi visualizado, revisado e refatorado as informações coletadas a fim de manter a integridade dos dados e retestar o desenvolvimento da técnica de clusterização.

5.1.1 Levantamento das Interações

A primeira etapa do desenvolvimento, após os procedimentos de limpeza dos dados e de transformação, foi o levantamento das interações realizadas pelos alunos do experimento da UFPEL. Para visualizar os valores de total de interações, quantidade de computadores e discriminar a quantidade de interações por cada tipo de evento foi utilizada a linguagem Python, usada em todo projeto para levantar os dados de mineração e machine learning, juntamente com a biblioteca pandas. Essas tecnologias disponibilizam de maneira organizada visualizar o levantamento de interações realizadas por aluno. Em seguida será mostrada na figura à seguir, o código-fonte e o resultado do frame exibido pelo pandas.

Figura 14 – Levantamento de interações realizadas por aluno

```python
import json
import pandas as pd
#lido json disponibilizado pela UFPEL
with open('elimu-analyzer-export .json') as f:
    data=json.load(f)['classes']
#declaração de listas de computadores, totais de interações, e total de interações por eventos
computador=[]
totalinteracao=[]
targetsUpdate=[]
BLOCK_DRAG_UPDATE=[]
PROJECT_START=[]
PROJECT_RUN_STOP=[]
for computadores in data:
#adicionado na lista de computadores
    computador.append(computadores)
    for id in data[computadores]:...
#feito o frame do pandas que irá demonstrar visualmente o resultado dos totais de interações
frameTotalizador = pd.DataFrame({
    "Computadores": computador,
    "Interações": totalinteracao,
    "TARGETS_UPDATE":targetsUpdate,
    "DRAG_UPDATE":BLOCK_DRAG_UPDATE,
    "START": PROJECT_START,
    "RUN_STOP": PROJECT_RUN_STOP
})
#exibir o frame
print(frameTotalizador)
```

Resultado →

	Computadores	Interações	TARGETS_UPDATE	DRAG_UPDATE	START	RUN_STOP
0	computador1	177	139	34	2	2
1	computador2	197	142	44	5	6
2	computador3	192	91	88	10	3
3	computador4	697	175	385	67	70
4	computador5	157	60	79	11	7
5	computador6	277	138	120	7	12
6	computador7	149	20	119	1	9
7	computador8	168	74	61	22	11

Fonte: Elaborado pelo autor (2022).

O início deste código é marcado pela inicialização das bibliotecas: JSON que permite realizar a leitura do JSON da UFPEL; Pandas que realiza a demonstração de frame do resultado do código. Essas tecnologias disponibilizam de maneira organizada, visualizar o levantamento de interações realizadas por aluno. Neste algoritmo foram declaradas listas que ao percorrer os dados do JSON foram somando os contadores de cada respectivo computador. Por fim, foram unidas as listas de contadores com os levantamentos nos respectivos índices do frame realizado pelos pandas, o que possibilitou visualizar o resultado mencionado na imagem da figura 14.

5.1.2 Implementação do algoritmo k-means com os dados de interações

Nesta etapa foi implementado o algoritmo de clusterização k-means com os dados de interação do arquivo JSON da UFPEL. Inicialmente foram aproveitados os dados e o algoritmo que coletou as interações na etapa anterior. Além disso, foi utilizada de bibliotecas como: sklearn para realizar o K-Means, utilizando datasets com machine learn efetuando a silhueta com pontuação; O matplotlib para exibição de gráficos e dados matemáticos e numpy para cálculos e mapeamento da matriz que realiza a marcação das coordenadas dos grafos. Na figura 15, será demonstrado um trecho do código, em Python, onde é feita a inicialização dos

parâmetros necessários para realizar o agrupamento dos dados com k-means. Também, em seguida, foi feita a marcação das coordenadas com a relação entre total de interação e targetupdate. Foram realizadas as relações de coordenadas de grupos utilizando a comparação entre os outros os outros eventos nos Apêndices A, B e C envolvendo 3 clusters do algoritmo k-means. Além disso, foi demonstrado a variedade entre 2 e 3 clusters e exibindo a diferença entre eles. Por fim, foram definidos elementos que auxiliam na demonstração gráfica.

Figura 15 - Início do código do Algoritmo K-Means inter relacionando as coordenadas da figura com Dados totais de Interação e evento targetupdate da UFPEL

```
# Gerando o dataset com make_blobs
X, y = make_blobs(n_samples=len(computador),
                  n_features=2,
                  centers=50,
                  cluster_std=2,
                  center_box=([0,1000]),
                  shuffle=True,
                  random_state=100)
X= np.array([[totalinteracao[0],targetsUpdate[0]],[totalinteracao[1], targetsUpdate[1]],
range_n_clusters = [2,3,4]
for n_clusters in range_n_clusters:
    # Cria a divisão entre duas figuras
    fig, (ax1, ax2) = plt.subplots(1, 2)
    fig.set_size_inches(18, 7)
    # A primeira figura é a de silhueta
    ax1.set_xlim([-0.1, 1])
    # inserção visual da figura sobre espaçamento
    ax1.set_ylim([0, len(X) + (n_clusters + 1) * 10])
    # inicialização da clusterização com os número de cluster
    clusterer = KMeans(n_clusters=n_clusters, random_state=3)
    cluster_labels = clusterer.fit_predict(X)
    silhouette_avg = silhouette_score(X, cluster_labels)
    print("Total de clusters=", n_clusters,
          "Total da pontuação :", silhouette_avg)
```

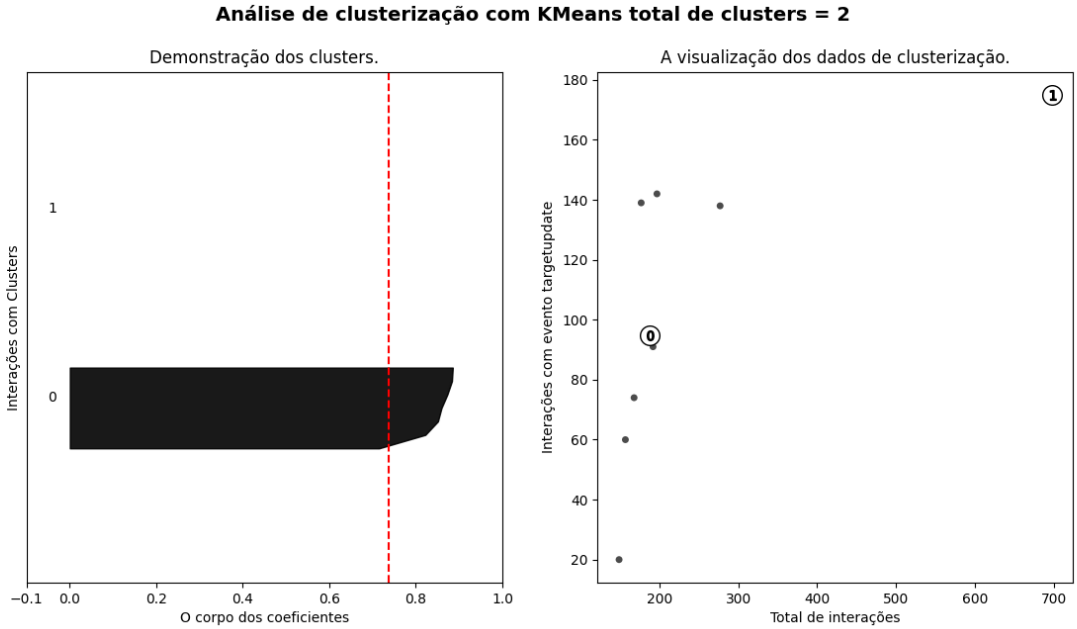
Fonte: Elaborado pelo autor (2022).

Após a execução do script envolvendo o algoritmo k-means com as interações, foi possível visualizar os resultados das figuras 16, 17 e 18 que conterão 2, 3 e 4 clusters, respectivamente. Nas figuras, foram demonstradas as coordenadas com dados totais na horizontal e os dados do evento targetupdate na vertical. É

possível observar que o cluster 1 está afastado do restante dos outros grafos pela maior quantidade de interações tanto totais como do evento targetupdate. Além disso, a pontuação da silhueta não é exibida em casos que ela é 1 por conta que a quantidade de elementos no cluster não é suficiente para realizar o cálculo.

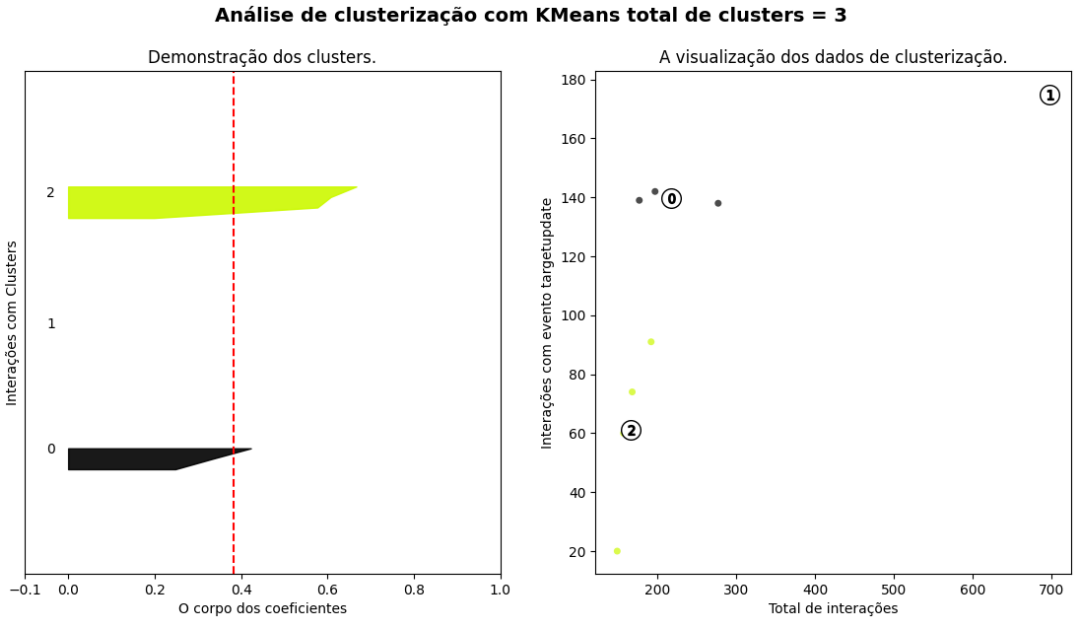
Na figura 16 é possível observar que os grupos são divididos entre o estudante com maior número de interações e o restante dos participantes. Nessa análise da silhueta, a pontuação dos clusters é de 0,7361 indicando que o agrupamento dos pontos de dados semelhantes entre si se encontra em um bom valor de k. Em observação, vale lembrar que o cluster 1 tem o valor de coeficiente em 1 já que ele se refere a apenas um dado e o cluster 0. Em contrapartida, na figura 17 é possível visualizar a separação do cluster 0 em 0 e 2, dessa maneira a pontuação da silhueta desce drasticamente para 0,3816 pontos indicando um número de k muito inferior ao anterior e também valores de coordenadas dos pontos muito diferentes um dos outros. Por fim, o cluster 3 é o que tem a menor pontuação com 0,33755, na figura 18 é possível observar que o anterior cluster 0 perde um ponto mais a direita que se transforma no cluster 2 e recebe outro ponto mais embaixo porém essa substituição decorrente da semelhança de distância entre pontos piorou a média entre todos os cluster causando um péssimo valor de k.

Figura 16 – Algoritmo K-Means contendo 2 clusters com Dados totais de Interação e evento targetupdate da UFPEL



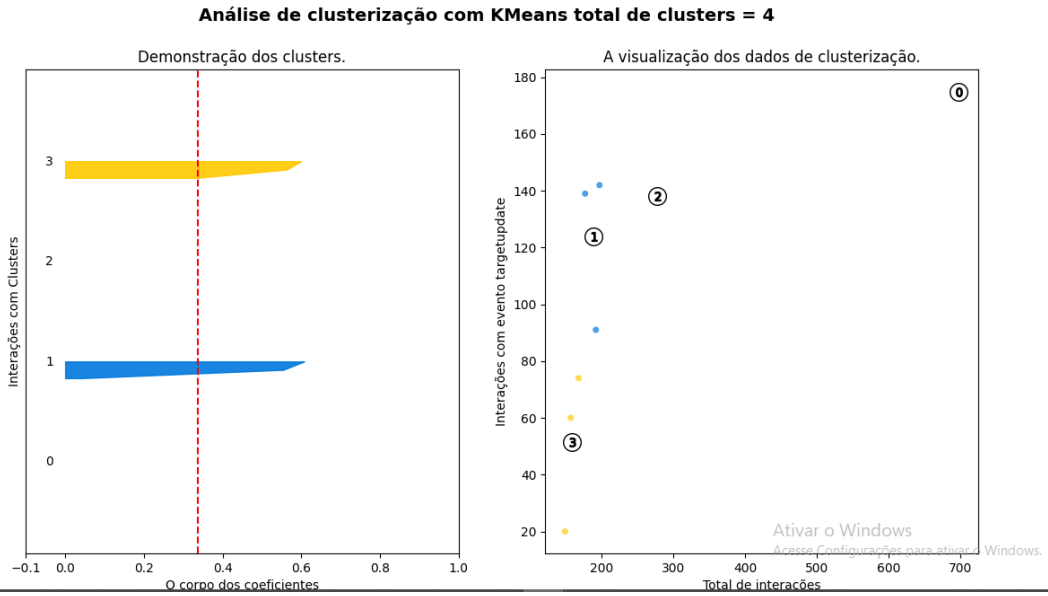
Fonte: Elaborado pelo autor (2022).

Figura 17 – Algoritmo K-Means contendo 3 clusters com Dados totais de Interação e evento targetupdate da UFPEL



Fonte: Elaborado pelo autor (2022).

Figura 18 – Algoritmo K-Means contendo 4 clusters com Dados totais de Interação e evento targetupdate da UFPEL



Fonte: Elaborado pelo autor (2022).

5.2 ANÁLISE DOS RESULTADOS DE CLUSTERIZAÇÃO DOS DADOS

Nesta etapa foram analisados os resultados da clusterização com a aplicação do programa WEKA. Nesse sentido, será exibido de maneira ilustrativa os resultados da clusterização das interações dos alunos mostrando um comparativo quantitativo das interações coletadas em formato de tabela.

5.2.1 Aplicação do programa WEKA

Nesta etapa, foi realizada a aplicação do programa WEKA para a análise dos padrões dos dados executando a clusterização realizada pelo algoritmo k-means, utilizando os dados do JSON da UFPEL. Nas próximas imagens será observado um padrão de informações semelhantes, sendo o primeiro dado a coluna (Attribute) identificando os eventos do scratch. Em seguida, é possível identificar a coluna que representa a quantidade de dados listados (8) e a relação da média dos centroides totais dos clusters (grupos). As próximas colunas referem-se aos clusters e a média de centroides de cada grupo relacionando os eventos do scratch, sendo o início de sua numeração a partir do número zero, exemplo: Cluster 0,1,2,3 etc.

Na figura 19, foram analisadas as quantidades de 2 clusters, nela foram divididos igualmente os 8 computadores do experimento, em 4 unidades para cada grupo. No evento "Target UPDATE" o valor médio dos centroides foi de 104,875, maior que o do Cluster 0(82,25) e menor que o do Cluster 1(127,5). Já no evento "BLOCK DRAG UPDATE" o valor médio foi de 116,25 sendo menor do que o do Cluster 0(161) e maior que o do Cluster 1(71,5).

Para o evento "PROJECT RUN START" o valor de média foi 15,625 sendo menor do que o Cluster 0(25,25) e maior do que o Cluster 1(6). Por fim, no evento "PROJECT RUN STOP" o valor médio foi de 15 sendo ele menor do que o do Cluster 0(24,25) e maior que o do Cluster 1(5,75). Dessa forma, foi visto que o Cluster 1 esteve apenas uma vez acima da média das interações com o evento "TARGET UPDATE", já o Cluster 1 esteve acima das médias nos eventos “BLOCK_DRAG_UPDATE”,“PROJECT_RUN_START” e “PROJECT_RUN_STOP”.

Figura 19 – Clusterização de dados realizado pelo WEKA com 2 clusters.

```
Missing values globally replaced with mean/mode

Final cluster centroids:
                                Cluster#
Attribute            Full Data          0          1
                         (8.0)      (4.0)      (4.0)
=================================================
TARGETS_UPDATE         104.875      82.25      127.5
BLOCK_DRAG_UPDATE       116.25        161       71.5
PROJECT_RUN_START       15.625      25.25          6
PROJECT_RUN_STOP            15      24.25       5.75



Time taken to build model (full training data) : 0 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       4 ( 50%)
1       4 ( 50%)
```

Fonte: Elaborado pelo autor (2022).

Na figura 20, foram analisadas as quantidades de 3 clusters, nessa análise foram divididos os 8 computadores do experimento em: 4 unidades para o Cluster 0; 3 unidades para o Cluster 1 e uma unidade para o Cluster 2. No evento "Target UPDATE" o valor médio dos centroides foi de 104,875, maior que o do Cluster 0(61,25), menor que o do Cluster 1(139,66) e menor que o Cluster2(175).Já no evento "BLOCK DRAG UPDATE" o valor médio foi de 116,25 sendo maior do que o do Cluster 0(86,75), maior que o do Cluster 1(66) e menor do que o Cluster 2(385).

Para o evento "PROJECT RUN START" o valor de média foi 15,625 sendo maior do que o Cluster 0(11) maior do que o Cluster 1(4,66667) e menor do que o Cluster 2(67). Por fim, no evento "PROJECT RUN STOP" o valor médio foi de 15 sendo ele maior do que o do Cluster 0(7,5), maior que o do Cluster 1(6,6667) e menor do que o Cluster 2(70). Dessa forma, foi visto que o Cluster 0 esteve em todas as médias abaixo, já o Cluster 1 esteve acima apenas no evento “BLOCK_DRAG_UPDATE” e o cluster 2 esteve acima do valor médio de coeficientes em todos os eventos.

Figura 20 – Clusterização de dados realizado pelo WEKA com 3 clusters.

```
Missing values globally replaced with mean/mode

Final cluster centroids:
                                Cluster#
Attribute            Full Data          0          1          2
                         (8.0)      (4.0)      (3.0)      (1.0)
============================================================
TARGETS_UPDATE         104.875      61.25   139.6667        175
BLOCK_DRAG_UPDATE       116.25      86.75         66        385
PROJECT_RUN_START       15.625         11     4.6667         67
PROJECT_RUN_STOP            15        7.5     6.6667         70



Time taken to build model (full training data) : 0 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       4 ( 50%)
1       3 ( 38%)
2       1 ( 13%)
```

Fonte: Elaborado pelo autor (2022).

Na figura 21, foram analisadas as quantidades de 4 clusters, nessa análise foram divididos os 8 computadores do experimento em: 4 unidades para o Cluster 0; uma unidade para o Cluster 1; uma unidade para o Cluster 2 e 2 unidades para o Cluster 3. No evento "Target UPDATE" o valor médio dos centroides foi de 104,875, maior que o do Cluster 0(61,25), menor que o do Cluster 1(138), menor que o Cluster2(175) e Cluster 3(140,5). Já no evento "BLOCK DRAG UPDATE" o valor médio foi de 116,25 sendo maior do que o do Cluster 0(86,75), menor que o do Cluster 1(120), menor do que o Cluster 2(185) e maior do que o do Cluster 3(39).

Para o evento "PROJECT RUN START" o valor de média foi 15,625 sendo maior do que o Cluster 0(11), maior do que o Cluster 1(7), menor do que o Cluster 2(67) e maior que o do Cluster 3(3,5). Por fim, no evento "PROJECT RUN STOP" o valor médio foi de 15 sendo ele maior do que o do Cluster 0(7,5), maior que o do Cluster 1(12), menor do que o Cluster 2(70) e maior que o do Cluster 3(4). Dessa forma, foi visto que o Cluster 0 esteve em todas as médias abaixo, já o Cluster 1 esteve acima no evento "TARGET UPDATE" e “BLOCK DRAG UPDATE”;

O cluster 2 esteve acima do valor médio de coeficientes em todos os eventos e o Cluster 3 apenas esteve acima no evento “TARGET UPDATE”.

Figura 21 – Clusterização de dados realizado pelo WEKA com 4 clusters.

```
Missing values globally replaced with mean/mode

Final cluster centroids:
                                Cluster#
Attribute            Full Data          0          1          2          3
                         (8.0)      (4.0)      (1.0)      (1.0)      (2.0)
=============================================================================
TARGETS_UPDATE         104.875      61.25        138        175      140.5
BLOCK_DRAG_UPDATE      116.25       86.75        120        385         39
PROJECT_RUN_START      15.625          11          7         67        3.5
PROJECT_RUN_STOP           15         7.5         12         70          4



Time taken to build model (full training data) : 0 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       4 ( 50%)
1       1 ( 13%)
2       1 ( 13%)
3       2 ( 25%)
```

Fonte: Elaborado pelo autor (2022).

6 CONCLUSÃO

Este trabalho possibilitou a análise dos padrões das interações dos dados com a plataforma scratch. Os dados disponibilizados pela UFPEL, no framework desenvolvido pelo estudante Jorge, possibilitaram a coleta do histórico de interações de 8 computadores que foram analisadas dentro desse trabalho de conclusão de curso. Com isso, foi possível aplicar o agrupamento das informações geradas, visualmente apresentadas pelo algoritmo k-means, e também detalhar essa clusterização pela plataforma WEKA. Além disso, foi possível entender as construções de bloco de código do Scratch.

Foi possível entender diferentes tecnologias de mineração de dados, a fim de compreender como interpretar e analisar os dados das interações disponibilizadas. Ao realizar esse estudo, foi possível escolher a análise de grupos/clusterização com o algoritmo k-means. Entretanto, por conta da pouca quantidade de dados, o trabalho realizado foi limitado, principalmente não havendo a medição da qualidade dos dados por conta da quantidade de dados analisados.

Futuramente, este trabalho terá sua continuidade para a ampliação dos dados e também na avaliação da qualidade e quantidade dos dados. Os conceitos apresentados irão contribuir futuramente para inicialização e desenvolvimento de novos integrantes do projeto. Dessa forma, esse material servirá como guia para outros participantes, além de disponibilizar esse estudo, principalmente, para os estudantes do curso de Tecnologias da Informação e Comunicação que não possuem esse material dentro da grade curricular e tem o desejo de estudar essa área.

REFERÊNCIAS

AMO, Sandra de. Técnicas de Mineração de Dados. **Jornada de Atualizaçao em Informatica**, jul. 2004. Disponível em: https://www.researchgate.net/publication/260300816_Tecnicas_de_Mineracao_de_Dados. Acesso em: 29 dez. 2021.

AMORIM, Thiago. **Conceitos, técnicas, ferramentas e aplicações de Mineração de Dados para gerar conhecimento a partir de bases de dados**. Centro de Informática UFPE. 2007. 50 f. TCC (Graduação) – Curso de Ciência da Computação, Faculdade de Educação, Universidade Federal de Pernambuco. Disponível em: <https://www.cin.ufpe.br/~tg/2006-2/tmas.pdf>. Acesso em: 30 dez. 2021.

Borges, L. E. **Python para desenvolvedores**. São Paulo SP: Novatec, v. 1, n. 1, p. 15, 2014.

CASTANHEIRA, Luciana Gomes. **Aplicação de Técnicas de Mineração de dados em problemas de classificação de padrões**. 2008. 95 f. Dissertação (Pós-Graduação) - Curso de Engenharia Elétrica, Universidade Federal de Minas Gerais, Belo Horizonte, 2008. Disponível em: https://repositorio.ufmg.br/bitstream/1843/BUOS-8CDFQK/1/luciana_gomes_castanheira.pdf. Acesso em: 03 jan. 2022.

COUTINHO, Thiago. **O que é a biblioteca Pandas?** 2021. Disponível em: https://www.voitto.com.br/blog/artigo/biblioteca-pandas. Acesso em: 17 jan. 2022.

CTRLPLAY (Brasil). **Scratch**: o que é e para que serve? [201-?]. Disponível em: https://ctrlplay.com.br/o-que-e-e-para-que-serve-o-scratch/#:~:text=Como%20falamos%20acima%2C%20o%20Scratch,pe%C3%A7as%2C%20n%C3%B3s%20criamos%20um%20programa. Acesso em: 25 dez. 2021.

DEVMEDIA. **Mineração de dados no MySQL com a ferramenta Weka**. 2012. Disponível em: https://www.devmedia.com.br/mineracao-de-dados-no-mysql-com-a-ferramenta-weka/26360. Acesso em: 19 jan. 2022.

DOCPLAYER. **PROGRAMAÇÃO EM PYTHON.** Direção Concursos, [201-?]. Disponível em: https://docplayer.com.br/197705938-Programacao-em-Python-conhecimentos-de-informatica-parte-ii-para-escriturario-do-banco-do-brasil-1-de-105.html. Acesso em: 10 jan. 2022.

FREITAS, Tainá. **Os três tipos de aprendizado no machine learning, um ramo da inteligência artificial**. 2019. Disponível em: https://www.startse.com/noticia/nova-economia/machine-learning-inteligencia-artificial-aprendizado/. Acesso em: 04 jan. 2022.

GARCIA, Joaquim. **PyCharm, um IDE poderoso para criar programas com Python**. [201-?]. Disponível em: https://www.linuxadictos.com/pt/pycharm-un-potente-ide-para-crear-programas-con-Python.html. Acesso em: 18 jan. 2022.

GASPERI, Guilherme Zimmer de. **Clusterização de dados K-Means na biblioteca scikit-learn**. 2020. Disponível em: https://micreiros.com/clusterizacao-de-dados-k-means-na-biblioteca-scikit-learn/. Acesso em: 05 jan. 2022.

GEPAC. **Introdução ao matplotlib**. 2019. Disponível em: https://gepac.github.io/2019-05-17-intro-matplotlib/. Acesso em: 23 jan. 2022.

IDOCODE. **Entenda como nossos alunos conseguiram desempenhar melhor na escola com a ajuda da programação** [201-?]. Disponível em: https://idocode.com.br/blog/programacao/entenda-como-nossos-alunos-conseguiram-desempenhar-melhor-na-escola-com-ajuda-da-programacao/. Acesso em: 29 nov. 2021.

IDOCODE. **Programação em blocos**: aprendendo de maneira divertida. aprendendo de maneira divertida. [201-?]. Disponível em: https://idocode.com.br/blog/programacao/programacao-em-blocos/. Acesso em: 29 dez. 2021.

LIMA, Stefani. **Visualização de dados em Python**: Matplotlib. 2018. Disponível em: https://king.host/blog/2018/03/visualizacao-de-dados-matplotlib/. Acesso em: 23 jan. 2022.

NAVEAVELA. **Qual o perfil do aluno do Século XXI**, 2018. Disponível em: https://naveavela.com.br/qual-e-o-perfil-do-aluno-do-seculo-xxi/. Acesso em: 16 dez. 2021.

OLIVEIRA, Millena Lauyse Silva de *et al*. **Ensino de lógica de programação no ensino fundamental utilizando o Scratch**: um relato de experiência. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 34., 2014, Garanhuns: Csbc, 2014. p. 1525-1534. Disponível em: https://sol.sbc.org.br/index.php/wei/article/view/10978/10848. Acesso em: 29 dez. 2021.

PATH. **DIFERENÇA ENTRE ANÁLISE PREDITIVA E ANÁLISE DESCRITIVA**. 2020. Disponível em: https://path.com.br/noticias/diferenca-entre-analise-preditiva-e-analise-descritiva/. Acesso em: 30 dez. 2021.

RADICH, Quinn *et al*. **O que é um modelo de machine learning?** 2022. Disponível em: https://docs.microsoft.com/pt-br/windows/ai/windows-ml/what-is-a-machine-learning-model. Acesso em: 10 jun. 2022.

RATZ, Arthur V. **Classifying Data Using Artificial Intelligence K-Means Clustering Algorithm**. 2020. Disponível em:

https://www.codeproject.com/Articles/5256294/Classifying-Data-Using-Artificial-Intelligence-K-M. Acesso em: 03 jan. 2022.

SANTIAGO JUNIOR, Luiz. **Entendendo a biblioteca NumPy**. 2018. Disponível em: https://medium.com/ensina-ai/entendendo-a-biblioteca-numpy-4858fde63355. Acesso em: 20 jan. 2022.

SATO, Luciane Yumie *et al*. Análise comparativa de algoritmos de árvore de decisão do sistema WEKA para classificação do uso e cobertura da terra. **Impe**, Foz do Iguaçu, Pr, p. 1-9, abr. 2013. Disponível em: https://www.researchgate.net/publication/261913754_Analise_comparativa_de_algoritmos_de_arvore_de_decisao_do_sistema_WEKA_para_classificacao_do_uso_e_cobertura_da_terra. Acesso em: 19 jan. 2022.

SCIKIT LEARN. **Fitting and predicting: estimator basics**. [20--]. Disponível em: https://scikit-learn.org/stable/getting_started.html. Acesso em: 30 jan. 2022.

SCRATCH. **Acerca do Scratch**. [20--]. Disponível em: https://scratch.mit.edu/about. Acesso em: 29 dez. 2021.

SCRATCH. **Scratch Statistics**. [20--]. Disponível em: https://scratch.mit.edu/statistics/. Acesso em: 29 dez. 2021.

SCRATCHBRASIL, **O que é Scratch?** [20--]. Disponível em: https://scratchbrasil.org.br/o-que-e-scratch. Acesso em: 29 dez. 2021.

SOUBHIA, Ana Luisa *et al*. **Python 101**. Cachoeira do Sul: UFSM, 2019. 64 p. Disponível em: https://www.ufsm.br/app/uploads/sites/679/2019/08/Apostila_Python_v_1.pdf. Acesso em: 10 jan. 2022.

STRINGFIXER. **Análise de cluster**. [20--]. Disponível em: https://stringfixer.com/pt/Cluster_Analysis. Acesso em: 04 jan. 2022.

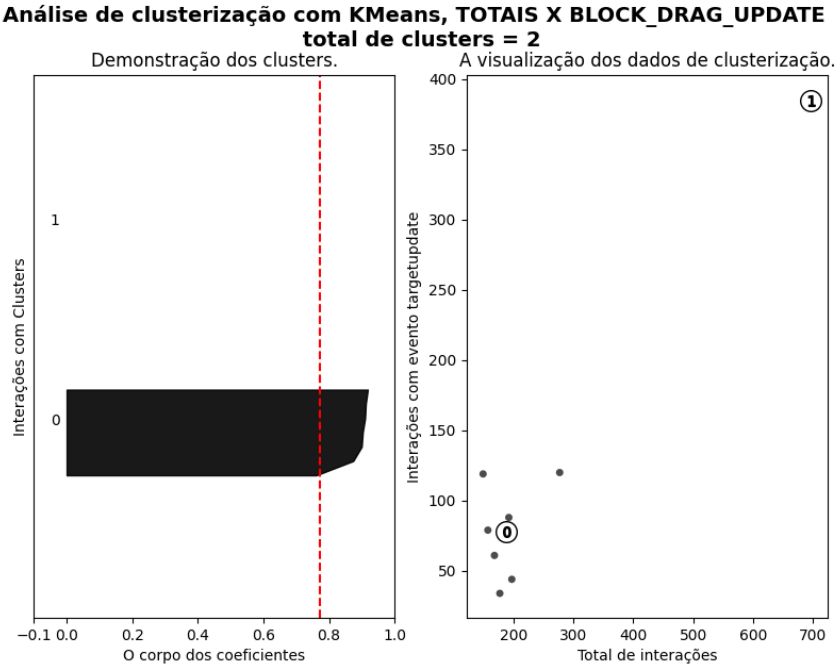
TECH, DIDÁTICA. **Aprendizado Supervisionado ou Não Supervisionado**. [201-?]. Disponível em: https://didatica.tech/aprendizado-supervisionado-ou-nao-supervisionado/. Acesso em: 04 jan. 2022.

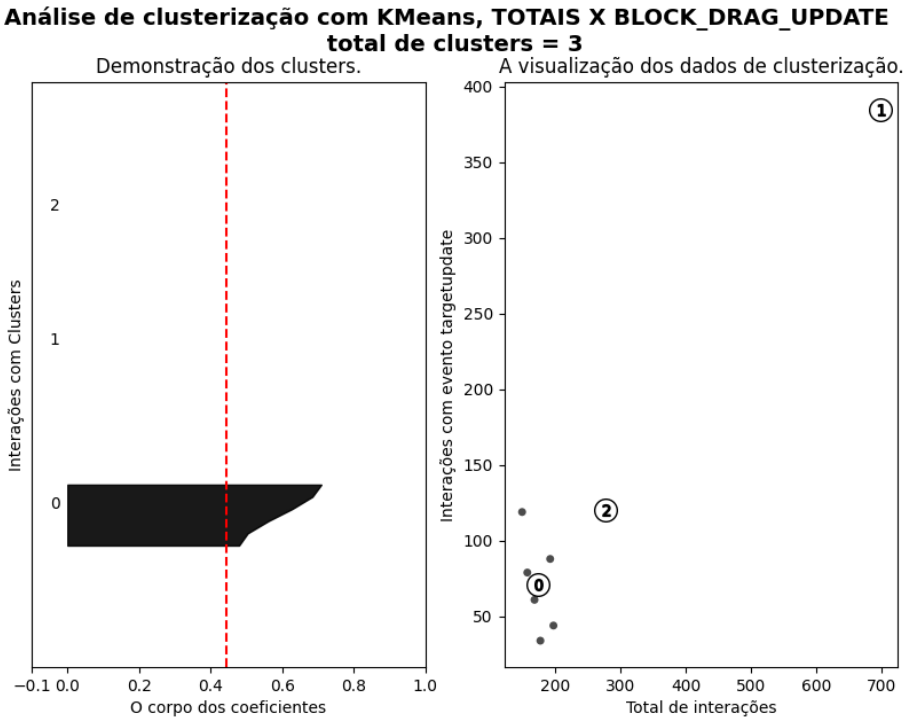
TIBCO. **What is Supervised Learning?** [201-?]. Disponível em: https://www.tibco.com/reference-center/what-is-supervised-learning. Acesso em: 04 jan. 2022.

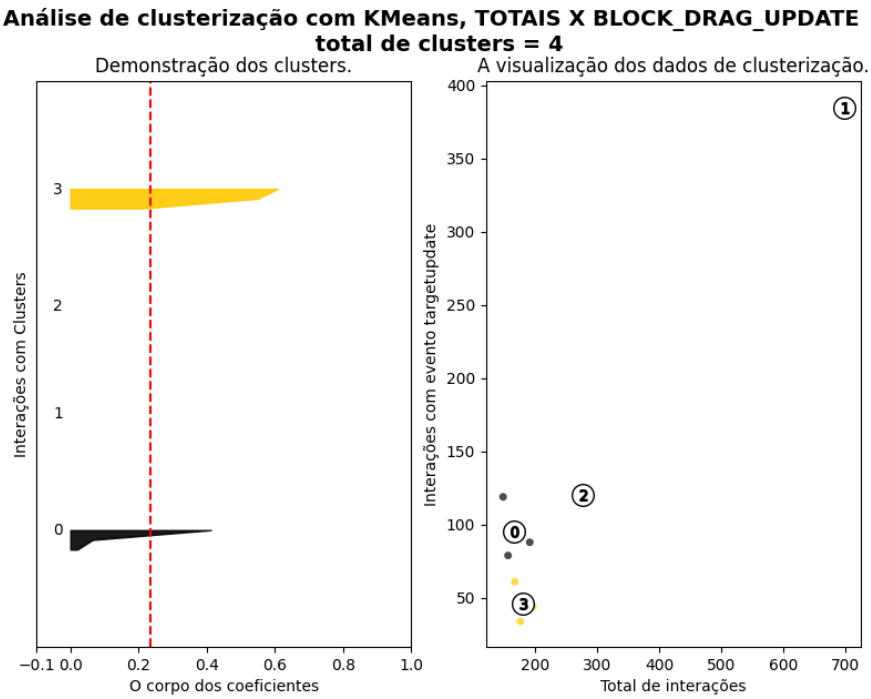
VAZ JUNIOR, J. L. N. **Um Framework para Coleta de Interações em Projetos Scratch**. 2019. 52 f. TCC (Graduação) – Curso de Ciências da Computação, Faculdade de Educação, Universidade Federal de Pelotas, Pelotas, 2019. Disponível em: https://sol.sbc.org.br/index.php/sbie/article/view/12842. Acesso em: 29 dez. 2021.

WEF, World Economic Forum. The Future of Jobs: Employment, Skills and Workforce Strategy for the Fourth Industrial Revolution. **Global Challenge Insight Report.** 2016. Disponível em: https://www3.weforum.org/docs/WEF_Future_of_Jobs.pdf. Acesso em: 29 nov. 2021.

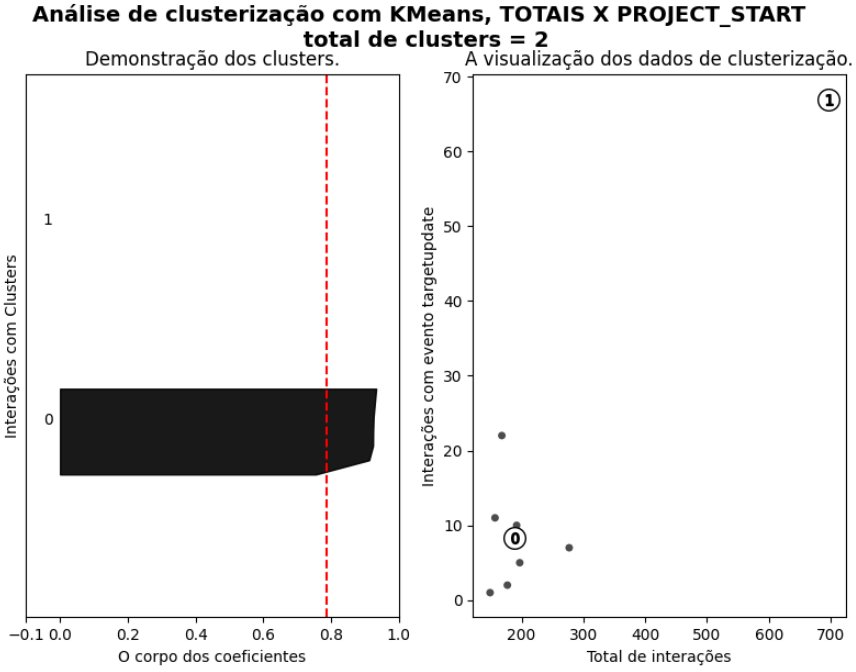
APÊNDICE A - CLUSTERIZAÇÃO ENTRE TOTAL E DRAG UPDATE

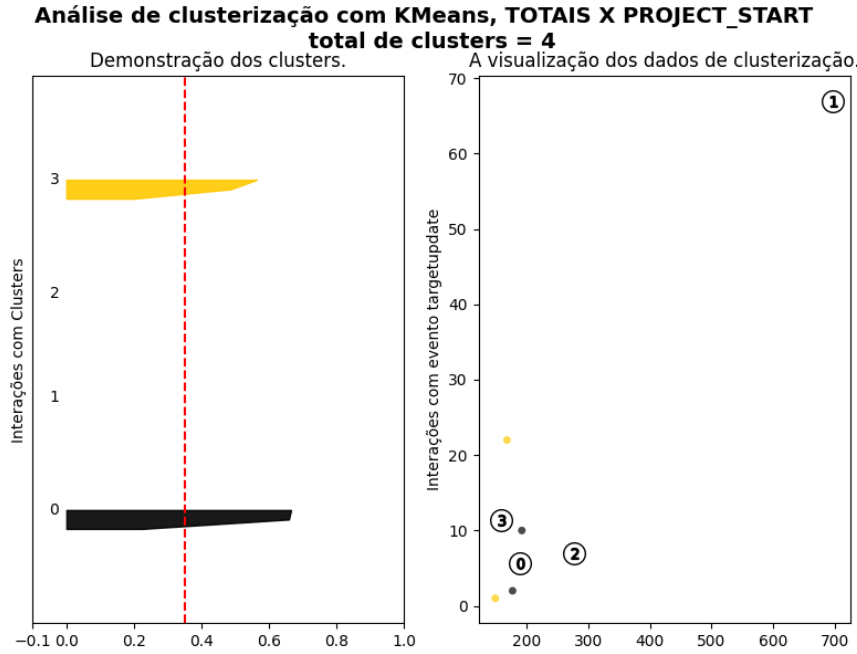


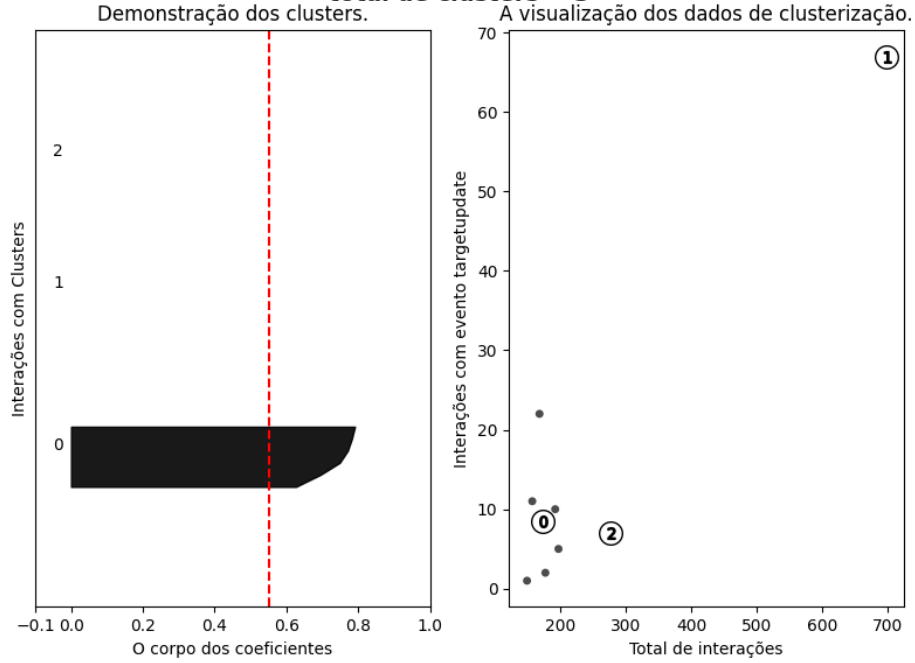




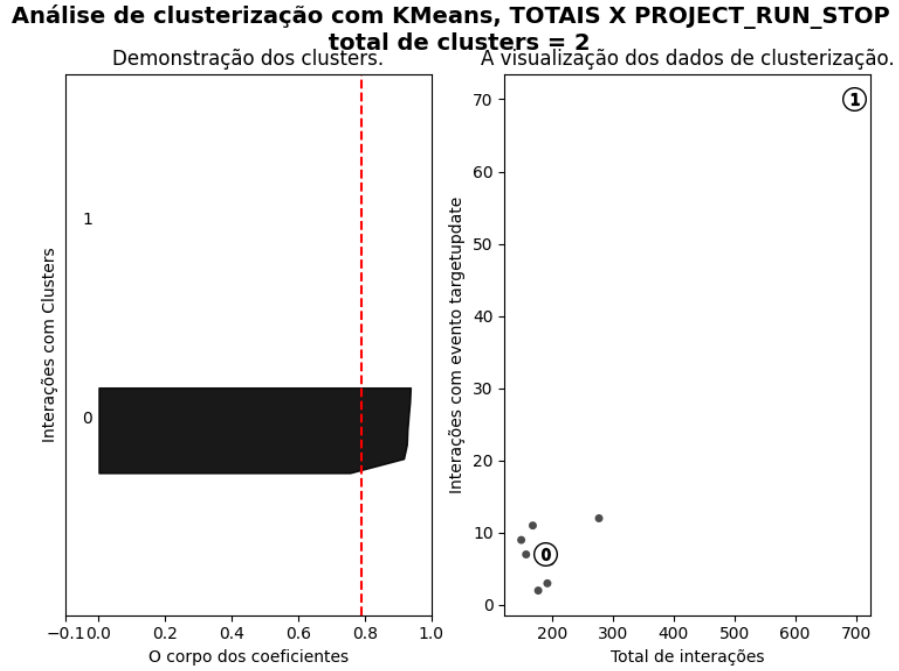
APÊNDICE B - CLUSTERIZAÇÃO ENTRE TOTAL E PROJECT_START

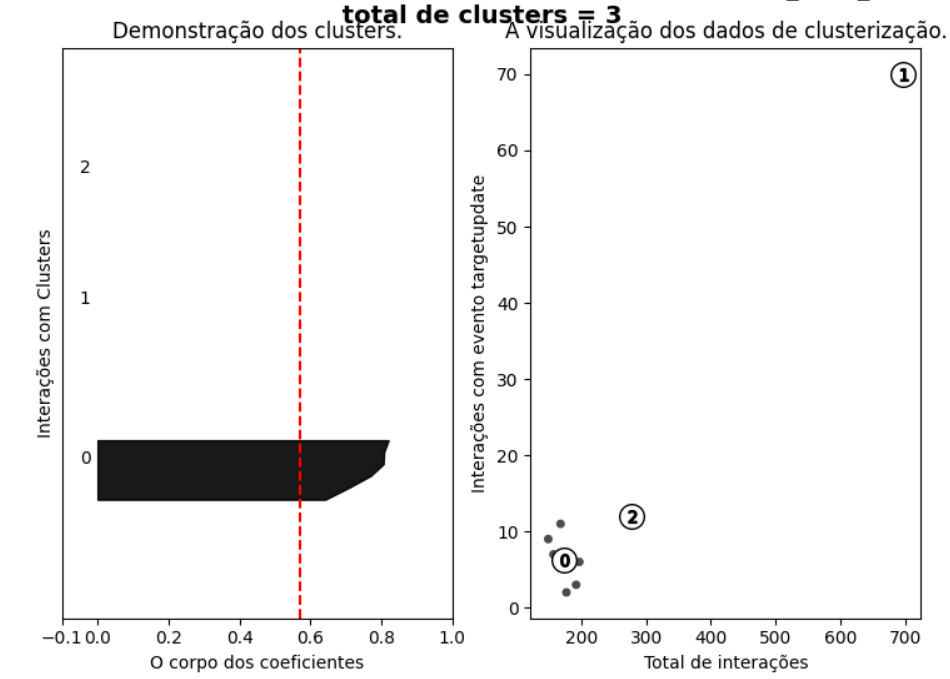


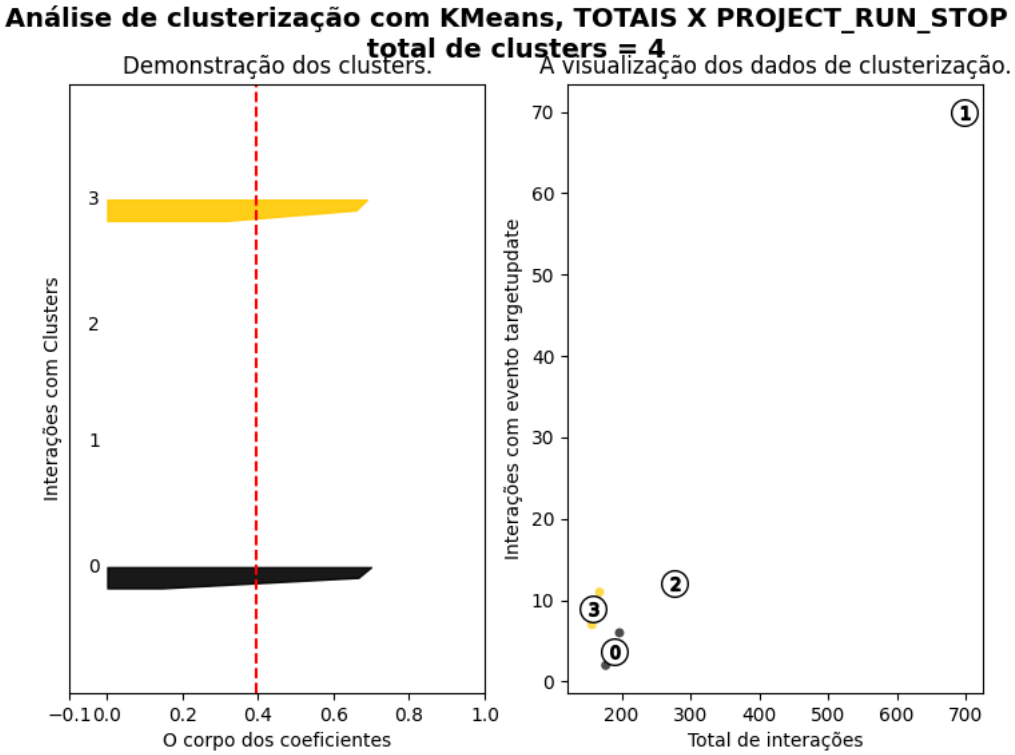

Análise de clusterização com KMeans, TOTAIS X PROJECT_START
total de clusters = 4
Demonstração dos clusters.
A visualização dos dados de clusterização.
Interações com Clusters
Interações com evento targetupdate
3
2
1
0
−0.1 0.0 0.2 0.4 0.6 0.8 1.0
0 10 20 30 40 50 60 70
200 300 400 500 600 700


Análise de clusterização com KMeans, TOTAIS X PROJECT_START
total de clusters = 3
Demonstração dos clusters.
A visualização dos dados de clusterização.
Interações com Clusters
Interações com evento targetupdate
2
1
0
−0.1 0.0 0.2 0.4 0.6 0.8 1.0
O corpo dos coeficientes
0 10 20 30 40 50 60 70
200 300 400 500 600 700
Total de interações

APÊNDICE C - CLUSTERIZAÇÃO ENTRE TOTAL E PROJECT_RUN_STOP






Análise de clusterização com KMeans, TOTAIS X PROJECT_RUN_STOP
total de clusters = 4
Demonstração dos clusters.
A visualização dos dados de clusterização.
Interações com Clusters
O corpo dos coeficientes
Interações com evento targetupdate
Total de interações