UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

BRUNO CORRÊA FEIL

Uma ferramenta de mineração de dados educacionais para o Moodle

Monografia apresentada como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação

Orientador: Prof. Dr. Leandro Krug Wives

Porto Alegre
2022

AGRADECIMENTOS

Agradeço ao meu orientador por toda a paciência e disponibilidade, a minha esposa por sempre me apoiar e a minha família por tudo.

RESUMO

O ensino a distância vem se tornando cada vez mais uma realidade. Com a sua chegada, surgem novos desafios, mas surgem também, muitas oportunidades através de ferramentas, antes não disponíveis, que podem ser úteis ao professor. Esta modalidade de ensino é realizada através de um ambiente virtual de ensino e aprendizagem (AVEA). Neste ambiente, cada ação do aluno/professor gera um log, uma informação. Estes logs podem ser analisados através de Mineração de Dados (MD) e novas informações, antes desconhecidas, podem surgir. Este trabalho de conclusão de curso foi desenvolvido com o intuito de criar uma ferramenta que auxilie o professor a obter informações a respeito dos seus alunos e seus ânimos em disciplinas ministradas via AVEA Moodle. Para isso, a ferramenta aqui proposta faz uso de ferramentas de MD como análises de correlação e de *Machine Learning* (aprendizado de máquina) para agrupamento e classificação de dados e disponibiliza as informações geradas através de visões intuitivas e de fácil entendimento para uso do professor.

Palavras-chave: Ciência da Computação. UFRGS. Mineração de Dados. Machine Learning. Moodle. Ambiente Virtual de Ensino e Aprendizagem.

A Moodle tool for mining educational data

ABSTRACT

Distance learning is increasingly becoming a reality. Thus, new challenges arrive to this teaching context, but also many opportunities through new tools that were previously unavailable to the teacher. This teaching modality is accomplished through virtual teaching and learning environment (VTLE). In this environment, every action of the student/teacher generates a log, a piece of information. These logs can be analyzed through Data Mining (DM) and new information, previously unknown, may emerge. This course completion work was developed with the intention of creating a tool that helps the teacher to obtain information about his students and their moods in Moodle courses. To accomplish this, the tool proposed here makes use of DM tools such as correlation analysis and Machine Learning for clustering and classifying data, and provides the generated information through intuitive and easy to understand views for the teacher's use.

Keywords: Computer Science, UFRGS, Data Mining, Machine Learning, Moodle, Virtual Teaching and Learning Environment.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AVEA	Ambiente Virtual de Ensino e Aprendizagem
API	Application Programming Interface
ER	Entidade Relacionamento
HTML	HyperText Markup Language
INF	Instituto de Informática da Universidade Federal do Rio Grande do Sul
ML	Machine Learning
REST	Representational State Transfer
UFRGS	Universidade Federal do Rio Grande do Sul
WB	Web Service
SGBD	Sistema de Gerenciamento de Banco de Dados

SUMÁRIO

1 INTRODUÇÃO

Com a chegada da pandemia do coronavírus em 2019, o ensino remoto foi se tornando cada vez mais importante. Com isso, alunos e professores tiveram que se reinventar para se adequar, às pressas, a essa nova realidade de aulas remotas. Além das preocupações "tradicionais" em ministrar uma disciplina, surgiram novos desafios para o professor, entre eles: como criar uma aula interessante para prender a atenção dos alunos em um ambiente virtual? Como fazer o aluno se interessar pelo conteúdo das aulas e não desistir?

Com a perda do contato "olho a olho" que se tinha no ensino presencial, ficou mais difícil para o professor entender o ânimo do aluno para participar da disciplina. Este trabalho de conclusão de curso tem como objetivo facilitar a vida do professor, no sentido de entender melhor o seu aluno, fornecendo alguns subsídios para que ele possa obter informações dos alunos com base nas suas ações na disciplina, e para que essas informações virem insumos para o desenvolvimento de suas futuras disciplinas. Para isso, disponibilizar uma ferramenta que gere visões sobre o comportamento dos alunos e que seja de fácil uso e entendimento.

O trabalho consiste em um *script* em Python que minera os dados de alunos de uma disciplina no Moodle utilizando algoritmos de *machine learning* (aprendizado de máquina) para realizar análises e disponibilizá-las para o professor.

O Moodle (*Modular Object-Oriented Dynamic Learning Environmen*t) é um software de apoio à aprendizagem executado em um ambiente virtual. Nele, os professores têm várias ferramentas para ministrar e conduzir uma disciplina, incluindo as opções de criar disciplinas; inserir alunos; carregar materiais; criar provas e tarefas de envio de arquivos; dar notas aos alunos; criar fóruns, entre outras. Já Mineração de dados é uma técnica para obter e encontrar anomalias, padrões e correlações entre os dados.

No Moodle, todas as ações de alunos e professores geram *logs*[1]. Esses *logs* podem ser acessados de diversas maneiras, inclusive via API (*Application Programming Interface*). Uma API é um conjunto de funções e padrões estabelecidos por um software. Utilizando essas funções e padrões, podemos obter dados de um sistema sem ter que conhecer detalhes da sua implementação. A partir da API do Moodle, foram obtidos os dados dos *logs* dos alunos.

Com esses dados, foram usados algoritmos de *machine learning* para realizar aná-

[1]Log é o registro de dados das atividades/eventos que ficam armazenados no computador.

lises. Os algoritmos utilizados foram:

- *K-médias*. Um algoritmo que é utilizado para agrupar dados semelhantes e, no contexto deste trabalho, foi utilizado para criar grupos de alunos semelhantes com base nas suas atividades na disciplina;
- Árvore de decisão. Um algoritmo de classificação de dados que cria uma árvore de decisão (um fluxograma sem ciclos) de acordo com o rótulo que se quer que o algoritmo faça a classificação. A decisão pelo rótulo alvo é feita através de uma sequência de testes dos valores de atributos. No contexto deste trabalho, o algoritmo foi utilizado para criar uma árvore, para a disciplina, com base no envio de tarefas em dia dos alunos.

Após a aplicação do algoritmo de agrupamento, com os grupos encontrados, foi realizada uma análise exploratória utilizando uma biblioteca em Python chamada de pandas_profiling. Essa biblioteca permite que seja realizada uma análise exploratória nos dados e a disponibiliza via arquivo HTML. Esse arquivo pode ser aberto em um navegador web qualquer. Com isso, o professor tem uma visão geral dos grupos encontrados, conseguindo traçar o perfil médio do aluno de cada grupo.

Além desses algoritmos de aprendizagem de máquina, também foi utilizado um algoritmo que gera o coeficiente de correlação de Spearman. Esse algoritmo serve para verificar se há alguma correlação entre os dados dos alunos.

Após todos os algoritmos aplicados, os dados de saída de cada análise são salvos em uma pasta e o professor conseguirá acessa-los e entender melhor como os alunos estão se comportando.

Todas essas ferramentas têm como intuito dar mais dados e possibilidades para o professor manter suas disciplinas sempre interessantes e os alunos sempre engajados.

Este trabalho está organizado conforme descrito a seguir. No capítulo 2 estão descritos os conceitos e tecnologias utilizados no desenvolvimento deste projeto. Este capítulo traz uma base teórica, detalhando conceitos como machine learning, correlação, mineração de dados e tecnologias como Moodle, Python e PostgreSQL. No capítulo 3 é mostrado um pipeline geral do projeto e as implementações dos algoritmos de correlação e machine learning utilizados pela ferramenta. O capítulo 4 traz um guia de uso ensinando, passo a passo, como utilizar a ferramenta para uma disciplina qualquer. O capítulo 5 descreve os resultados obtidos com a execução da ferramenta para uma disciplina real. No capítulo 6 são mencionados trabalhos relacionados, que analisam logs de alunos para

entender o seu comportamento e sucesso nas disciplinas. Por fim, o capítulo 7 apresenta as conclusões sobre o que se espera deste trabalho.

2 CONCEITOS E TECNOLOGIAS

Para um melhor entendimento sobre este trabalho, alguns conceitos e tecnologias precisam ser conhecidos. Este capítulo traz uma explicação sobre estes assuntos.

2.1 Conceitos

2.1.1 *Machine Learning*

Machine learning (ML) ou aprendizado de máquina abrange uma ampla gama de algoritmos e ferramentas de modelagem usadas para uma vasta gama de tarefas de processamento de dados (CARLEO et al., 2019). Estes algoritmos servem para resolver problemas de 3 classes de problemas de aprendizagem:

- Aprendizado supervisionado;
- Aprendizado não supervisionado;
- Aprendizado por reforço.

Para este trabalho, foram utilizadas duas dessas classes: aprendizado supervisionado e não supervisionado.

2.1.1.1 Aprendizado supervisionado

Este tipo de problema de aprendizado é resolvido através da construção de um modelo capaz de prever resultados futuros após ser treinado com base em dados passados. Para este treinamento, são usados pares de entrada/saída ou dados rotulados. O objetivo é produzir uma função que seja aproximada o suficiente para poder prever saídas quando novas entradas forem apresentadas ao modelo (Pratap Chandra Sen, Mahimarnab Hajra, Mitadru Ghosh, 2020).

2.1.1.2 Aprendizado não supervisionado

O aprendizado não supervisionado é uma classe de problemas de aprendizado onde os dados de entrada são obtidos, assim como no aprendizado supervisionado, mas não há rótulos disponíveis. O objetivo da aprendizagem é recuperar algumas informações básicas, e possivelmente não triviais, na estrutura no conjunto de dados. Um exemplo

típico de aprendizado não supervisionado é o agrupamento de dados em que os pontos de dados são atribuídos em grupos de tal forma que cada grupo tenha algumas propriedades comuns (CARLEO et al., 2019). Este, inclusive, é um dos tipos de aprendizado utilizado neste trabalho.

2.1.2 Correlação

Segundo Patrick Schober, Christa Boer, Lothar A Schwarte (2018) (tradução do autor):

> Correlação, em um sentido mais amplo, é uma medida de associação entre variáveis. Nos dados correlacionados, a mudança na magnitude de uma variável é associada com a mudança na magnitude de outra variável, tanto no mesmo sentido (correlação positiva) quanto no sentido oposto (correlação negativa).[1]

O resultado de uma análise de correlação é um coeficiente de correlação entre cada par de variável. Este coeficiente é um valor entre -1 e 1 (inclusive os extremos) e indicará o quão forte é a correlação entre o par calculado. Esta indicação, porém, não é unanimidade entre autores. A tabela 2.1 mostra algumas definições. Pode-se observar que as definições variam muito. Valores altos e baixos acabam em consenso pelos 3 autores, os valores "do meio" acabam divergindo.

Tabela 2.1: Interpretações dos coeficientes de correlação de Spearman

Coeficiente de Correlação		**Dancey & Reidy (Psicologia)**	**Quinnipiac University (Política)**	**Chan YH (Medicina)**
+1	-1	Perfeito	Perfeito	Perfeito
+0.9	-0.9	Forte	Muito forte	Muito forte
+0.8	-0.8	Forte	Muito forte	Muito forte
+0.7	-0.7	Forte	Muito forte	Moderado
+0.6	-0.6	Moderado	Forte	Moderado
+0.5	-0.5	Moderado	Forte	Razoável
+0.4	-0.4	Moderado	Forte	Razoável
+0.3	-0.3	Fraco	Moderado	Razoável
+0.2	-0.2	Fraco	Fraco	Baixo
+0.1	-0.1	Fraco	Insignificante	Baixo
0	0	Zero	Nenhum	Nenhum

Fonte: Elaborado por Haldun Akoglu (2018).

No contexto deste trabalho, foi considerada a interpretação de Dancey & Reidy porque parece ser a mais "balanceada" das 3 citadas, tendo, além dos extremos, 3 níveis

[1] Correlation in the broadest sense is a measure of an association between variables. In correlated data, the change in the magnitude of 1 variable is associated with a change in the magnitude of another variable, either in the same (positive correlation) or in the opposite (negative correlation) direction.

de correlação forte, 3 níveis de correlação moderada e 3 níveis de correlação fraca. A ideia é dar uma dica, para o professor, de ações que podem ser benéficas/maléficas se combinadas juntas. Como ação benéfica, por exemplo, correlação alta positiva entre a quantidade de interações no forum e a nota do aluno. Quanto mais interação nos foruns, maior tende a ser a nota final do aluno. Como ação maléfica, por exemplo, correlação alta negativa entre tarefas enviadas em atraso e as notas dos *quizzes*. Quanto mais envios em atraso, menor tende a ser a nota das provas do aluno, denotando uma relação inversa. Algumas informações interessantes podem ser tiradas estudando a correlação entre os dados.

2.1.2.1 Correlação de Spearman

Para este trabalho foi escolhido implementar a análise de correlação de Spearman. Esta escolha se dá porque a análise de correlação de Spearman não exige que as variáveis tenham uma distribuição normal, a análise de Pearson exige, conforme consta no trabalho de Jan Hauke, Tomasz Kossowski (2011). Para não precisar testar a normalidade dos dados toda vez que for gerar a análise, foi optado por utilizar o método de Spearman. Além disso, ele pode ser utilizado tanto para comparação de variáveis quantitativas quando qualitativas. Isso se dá porque o método não compara o valor da variável em si, mas cria um ranque para cada variável e compara os valores desses ranques para definir a correlação entre elas (Patrick Schober, Christa Boer, Lothar A Schwarte, 2018). Então caso se decida adicionar alguma variável qualitativa para a análise, não terá problema.

Uma distribuição de dados é chamada de distribuição normal quando a média, a moda e a mediana da distribuição são iguais.

Variáveis quantitativas são aquelas em que os valores são expressos em números, podendo ser discretas (conjunto finito/enumerável de valores possíveis) ou contínuas (conjunto infinito de valores possíveis). Variáveis qualitativas são aquelas cujos valores podem ser separados em diferentes categorias que se distinguem por alguma característica não numérica, podendo ser ordinal (quando existe uma ordem nos seus valores) ou nominal (quando uma ordem não pode ser estabelecida entre seus valores).

2.1.3 Mineração de dados

A mineração de dados é definida como a prática de examinar uma grande quantidade de dados para gerar novas informações (PRUENGKARN et al., 2017). Para encontrar padrões e tendências entre os dados e obter informações úteis, são utilizadas técnicas de mineração de dados. Neste trabalho, foram utilizadas as técnicas de classificação e agrupamento para realizar a mineração dos dados obtidos através dos logs dos alunos. Essas técnicas estão descritas nas subseções seguir.

2.1.3.1 Classificação

Algoritmos de classificação têm como objetivo classificar os dados, de acordo com características observadas por um supervisor, para obter previsões precisas em um grande conjunto de dados. Eles fazem parte do aprendizado de máquina supervisionado.

2.1.3.2 Agrupamento

Algoritmos de agrupamento buscam padrões existentes nos conjuntos de dados. Os dados então são agrupados de acordo com sua similaridade. Estudando a semelhança entre os padrões de cada grupo pode se chegar a conclusões interessantes. Muito utilizados para entender os perfis de cada cliente. No nosso contexto, utilizado para entender o perfil dos alunos. Informação relevante para o professor que poderá entender o comportamento de cada tipo de aluno (alunos animados, desanimados, com tendência ao desânimo).

Este tipo de algoritmo faz parte do aprendizado de máquina não supervisionado.

Segundo Dongkuan Xu, Yingjie Tian (2015), um agrupamento de dados pode ser definido por:

1. Instâncias, no mesmo grupo, devem ser o mais similares possíveis;
2. Instâncias, em grupos diferentes, devem ser o mais diferentes possíveis;
3. A medida para a similaridade e desigualdade deve ser clara e ter um sentido prático.

2.1.4 Algoritmo de Árvores de Decisão

Foi implementado o algoritmo de Árvore de Decisão como algoritmo de classificação. O motivo principal da escolha desse algoritmo é porque ele é de fácil execução e simples de entender e interpretar (é mais "visual") em comparação a outros algoritmos de classificação (Anuja Priyama, Abhijeeta, Rahul Guptaa, Anju Ratheeb, Saurabh Srivastavab, 2013). Consegue-se criar uma árvore de decisão e mostrar ela na tela, contendo o fluxo que os dados percorrem para cada classificação. Além disso, ele consegue lidar com variáveis quantitativas e qualitativas (adicionar novos campos nos dados, não requer alteração no algoritmo utilizado), usa um modelo de caixa aberta (as situações observáveis no modelo são facilmente explicadas por lógica booleana[2]), possui uma seleção de características integrada (atributos irrelevantes para o modelo, tendem a ser menos utilizados para a criação da árvore de decisão).

Segundo Carl Kingsford, Steven L Salzberg (2008) (tradução do autor):

> Uma árvore de decisão classifica os itens de dados fazendo uma série de perguntas sobre as características associadas a esses dados. Cada questão está contida em um nó da árvore, e cada nó interno aponta para um nó filho para cada resposta possível. As perguntas formam assim uma hierarquia, codificada como uma árvore. De forma simplificada, são feitas perguntas de sim ou não e cada nó interno tem um filho 'sim' e um filho 'não'. Um item é classificado em uma classe seguindo o caminho do nó mais alto, a raiz, para um nó sem filhos, uma folha, de acordo com as respostas que se aplicarem ao item em consideração. Será atribuída a classe do nó folha ao item em questão.[3]

Para entender melhor esta definição de (Carl Kingsford, Steven L Salzberg, 2008), a figura 2.1 foi criada. Nela, é mostrado o detalhe de cada nó da árvore de decisão. Os seus conceitos são os seguintes:

- Condição. Para cada registro é testada uma condição com base em um atributo dos dados. Caso o resultado do teste seja Verdadeiro, o registro continuará o fluxo indo para o nó à esquerda. Caso o resultado seja Falso, o registro irá para o nó da direita;
- Entropia. Conceito utilizado para construir a árvore, denota o grau de confusão do nó;
- Número de elementos. Quantidade de amostras que a árvore tem a partir do nó

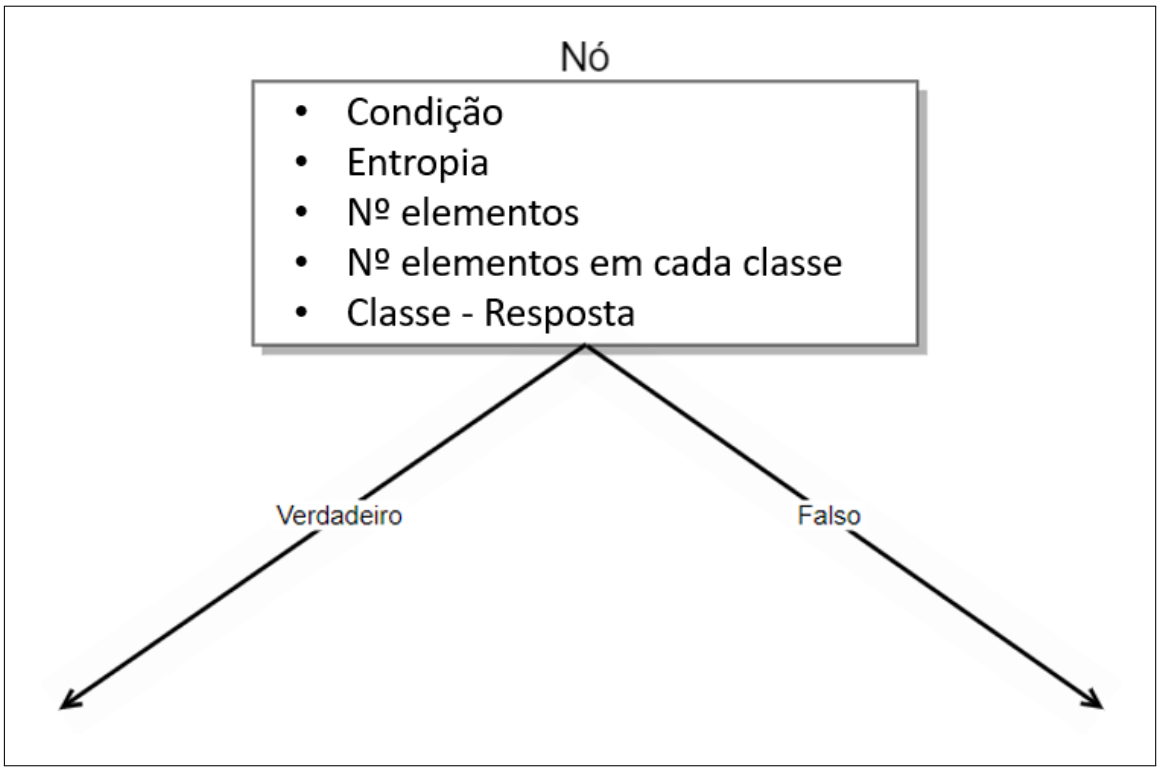
[2] A lógica Booleana é uma forma de representar a lógica através de equações matemáticas.

[3] A decision tree classifies data items by posing a series of questions about the features associated with the items. Each question is contained in a node, and every internal node points to one child node for each possible answer to its question. The questions thereby form a hierarchy, encoded as a tree. In the simplest form, we ask yes-or-no questions, and each internal node has a 'yes' child and a 'no' child. An item is sorted into a class by following the path from the topmost node, the root, to a node without children, a leaf, according to the answers that apply to the item under consideration. An item is assigned to the class that has been associated with the leaf it reaches.

corrente;

- Número de elementos em cada classe. Quantidade das amostras que estão em cada classe alvo. Neste caso do exemplo, demonstra quantos alunos enviaram 0, 1 e 2 tarefas em dia;
- Classe - Resposta. Classificação do registro. Quando o registro chegar em um nó folha (último nó, sem ramificação), ele será classificado com esta classe.

Figura 2.1: Detalhe de cada nó da árvore de decisão.



Fonte: Elaborado pelo autor

Há várias formas de criar uma árvore de decisão. Para o trabalho, foi utilizado o método de ganho de informação (entropia). O ganho de informação é a *informação mútua* entre a decisão do nó local (esquerda ou direita) e a saída preditiva (NOWOZIN, 2012). As variáveis são ranqueadas de acordo com o seu ganho de informação em relação à variável alvo. Esse ganho de informação é calculado usando o cálculo de entropia de cada variável. Quanto maior a entropia de um atributo, menos ganho de informação ele irá trazer. Após o ranqueamento das variáveis, a com maior ganho de informação será escolhida e virará a raiz da árvore (o ponto de partida do fluxo do dado até a classificação). Para cada ramificação da raiz, os atributos restantes serão novamente ranqueados de acordo com o seu ganho de informação. Esse cálculo continuará até que acabem os atributos, ou que a ramificação resulte em apenas uma classe (todos os alunos da disciplina

X com média de notas de tarefas maior do que 6, entregaram todas as tarefas em dia, por exemplo).

2.1.5 Algoritmo *K-médias*

O algoritmo de agrupamento utilizado para este trabalho foi o *K-médias*. Esta escolha se dá porque este algoritmo é rápido, robusto, de fácil compreensão e porque, após sua execução, todos os registros terão sido atribuídos a um *cluster* (grupo). Nenhum aluno ficará sem grupo.

O algoritmo *K-médias* é um algoritmo iterativo. Atribui cada ponto (registro) a um grupo cujo centro, também chamado de centroide, é o mais próximo. O centroide é a média de todos os pontos dentro do *cluster*, ou seja, as suas coordenadas são a média aritmética, para cada dimensão separadamente, de todos os pontos dentro do *cluster* (T. Soni Madhulatha, 2012). A execução do algoritmo funciona através dos 5 passos que estão descritos através da figura 2.2.

Figura 2.2: Pseudocódigo genérico do algoritmo K-médias

Passo 1: Recebe o número de grupos a ser criado e o conjunto de dados para agrupar como valores de entrada

Passo 2 Inicializa os primeiros K grupos

- Considera os primeiros K registros do conjunto de dados ou
- Considera uma amostra aleatória de K elementos

Passo 3: Calcula a média aritmética de cada cluster formado

Passo 4: K-médias atribui cada registro do conjunto de dados a apenas um dos grupos criados

- Cada registro é atribuído ao grupo mais próximo usando uma medida de distância (distância euclidiana, neste contexto)

Passo 5: K-médias recalcula a média aritmética de todos os clusters, considerando os elementos de cada cluster, e reatribui cada registro do conjunto de dados ao cluster mais próximo.

Fonte: Adaptado pelo autor a partir de Oyelade, O. J., Oladipupo, O. O., Obagbuwa, I. C. (2010)

Para execução do algoritmo, uma medida de similaridade entre os dados precisa ser definida. Ela definirá o quão "próximos" são os dados. Como no contexto do projeto estamos lidando com dados quantitativos, a medida de similaridade utilizada no algoritmo foi a distância euclidiana (medida na qual o algoritmo *K-médias* é baseado). Ela é definida

como a menor distância (linha reta) entre dois pontos e é denotada pela fórmula 2.1, onde n é o número de variáveis, e x_j e y_j são os valores da j-ésima variável dos registros x e y respectivamente (T. Soni Madhulatha, 2012).

$$d = \sqrt{\sum_{j=1}^{n} (x_j - y_j)^2} \tag{2.1}$$

2.2 Tecnologias

Nesta seção, é feita uma breve explicação sobre as tecnologias e ferramentas utilizadas para o desenvolvimento deste trabalho.

2.2.1 Moodle

O Moodle é uma plataforma de Ambiente Virtual de Ensino e Aprendizagem (AVEA) de *software* livre e código aberto. Esta plataforma é utilizada por alunos e professores como ferramenta de apoio ao ensino à distância em mais de 220 países. Como dito anteriormente, no Moodle, todas as ações dos alunos e professores geram logs. Estes logs são uma fonte muito rica de dados. Estes dados podem ser minerados trazendo informações úteis ao professor.

O Moodle por ser uma plataforma consolidada, amplamente utilizada, de fácil uso e obtenção de dados foi utilizada para este trabalho. Além disso, é onde a maioria das aulas da UFRGS são ministradas.

2.2.2 Linguagem de programação Python

A linguagem Python é uma linguagem de programação amplamente utilizada para análise e mineração de dados. Possui muitas bibliotecas[4] que auxiliam a criação de modelos de ML, visualizações de dados, operações matemáticas, configuração para conexão a bancos de dados externos. Além de ser uma linguagem de fácil entendimento (alto nível de abstração) e muito dinâmica (pode ser utilizada em vários contextos).

[4]Em programação, uma biblioteca é uma coleção de funções auxiliares, que não fazem parte do núcleo da linguagem, criadas para resolver um determinado tipo de problema.

Neste trabalho, onde os dados do Moodle são extraídos via API, depois consolidados via Banco de dados e analisados via aprendizado de máquina, a linguagem python foi essencial. Com ela, todos esses passos são feitos com facilidade, utilizando as muitas bibliotecas disponíveis.

A tabela 2.2 mostra as bibliotecas utilizadas e traz uma breve descrição sobre suas funcionalidades.

Tabela 2.2: Bibliotecas utilizadas.

Biblioteca	Descrição
pandas	Manuseio de DataFrames e função de correlação
psycopg2	Conexão com banco de dados PostgreSQL
seaborn	Visualização dos dados de correlação
matplotlib	Visualização dos dados de correlação e árvores de decisão
sklearn	Criação de modelos de ML para as análises de agrupamento e classificação
pandas_profiling	Análise exploratória

Fonte: Elaborado pelo autor

2.2.3 PostgreSQL

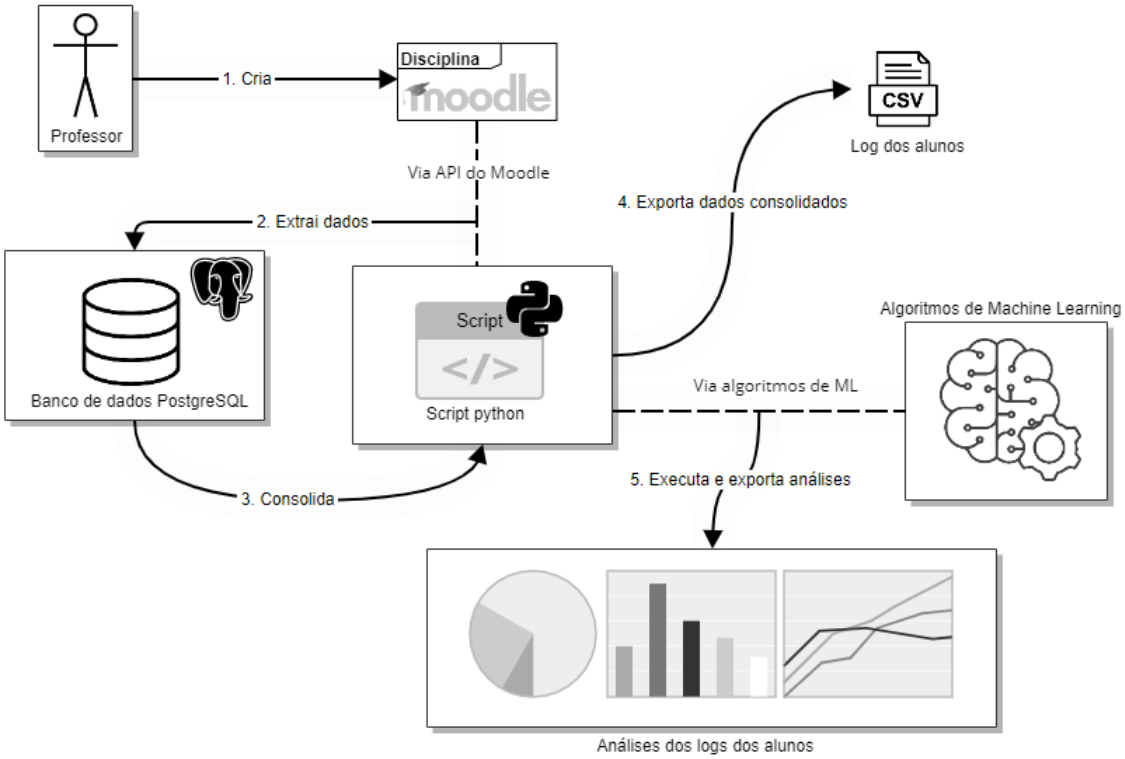
PostgreSQL é um sistema de gerenciamento de banco de dados (SGBD) objeto-relacional de código aberto. É um SGBD já consolidado, de fácil acesso, instalação e configuração (importante quando o professor for utilizar a ferramenta proposta por este trabalho) além de possuir uma comunidade grande. Apesar de este trabalho não exigir muitos recursos a nível de banco de dados, o SGBD PostgreSQL possui vários recursos disponíveis. Por estes motivos, este SGBD foi escolhido para este trabalho.

3 IMPLEMENTAÇÃO E ARQUITETURA

Este capitulo traz uma descrição genérica do processo de execução do projeto assim como os algoritmos implementados. O capítulo 4 contém um guia explicando como fazer para executar a ferramenta, as versões mínimas necessárias, as configurações e programas necessários. Os resultados da aplicação da ferramenta em uma disciplina real estão disponíveis no capítulo 5.

O projeto consiste em um *script* em Python que obtém os dados dos *logs* dos alunos de uma disciplina no Moodle, exporta-os para um banco de dados PostgreSQL (banco de dados relacional de código aberto), e executa funções que consolidam esses dados. Após a consolidação, importa os dados para executar algoritmos de ML e analisar correlação, exportando as análises para uso do professor. O *pipeline* da arquitetura do projeto se encontra na Figura 3.1. O código do *script* Python está disponível através do github do autor[1].

Figura 3.1: Pipeline do projeto



Fonte: Elaborado pelo autor.

[1] <https://github.com/brunofeil/TCC>

3.1 Criação da disciplina

O professor cria uma disciplina no Moodle e insere seus alunos. Para que os dados sejam exportados via API, é necessário que o Moodle a ser utilizado esteja com os *Web Services*[2] habilitados e que o professor tenha as permissões necessárias para acessar os dados via WB (*Web Services*). Um WB é um tipo de API que utiliza necessariamente a rede para realizar a comunicação.

3.2 Extração dos logs

Para conexão com a API do Moodle, foi utilizada uma biblioteca feita por Martin Vuk[3], a qual é feita utilizando o protocolo REST. A API retorna dados em formato JSON. As funções usadas para obtenção de dados se encontram na Tabela 3.1

Tabela 3.1: Tabela funções da API do Moodle utilizadas

Nome	**Entrada**	**Saída**
core_user_get_users_by_field	usuário	Informações do usuário.
core_enrol_get_users_courses	id_usuário	Cursos que o usuário participa.
core_enrol_get_enrolled_users	id_curso	Alunos que participam do curso.
mod_assign_get_assignments	id_curso	Tarefas do curso.
mod_assign_get_submissions	lista_tarefas	Envios de tarefas.
mod_forum_get_forums_by_courses	id_curso	Foruns do curso.
mod_forum_get_forum_discussions	id_forum	Discussões de foruns.
mod_forum_get_discussion_posts	id_discussão	Postagens das discussões.
mod_quiz_get_quizzes_by_courses	id_curso	Quizzes do curso.
mod_quiz_get_user_attempts	id_quiz, id_aluno	Tentativas do aluno para o quiz.
mod_quiz_get_user_best_grade	id_quiz, id_aluno	Melhor nota do aluno para o quiz.
gradereport_user_get_grade_items	id_curso, id_aluno	Notas do aluno.

Fonte: Adaptado pelo autor a partir da documentação da API do Moodle.

O *script* escrito em Python extrai os dados dos *logs* dos alunos de uma disciplina. Para tanto, quando executado, lista todas as disciplinas que o professor participa ou participou e ele deve escolher em qual será feita a análise. Para isso, o professor deve inserir o seu usuário do Moodle no código Python. A execução do *script* irá chamar a função 'core_user_get_users_by_field' da API do Moodle para obter as informações do usuário. A partir dessa informação, consegue-se chegar no id do usuário, usado para trazer todos

[2]Opção de configuração do Moodle. Explicação de onde habilitar os *Web Services* no Moodle disponível no capítulo 4.

[3]<https://github.com/mrcinv/moodle_api.py>

os cursos do usuário através da função ‘core_enrol_get_users_courses’. Após finalização dessas funções, os cursos do professor são mostrados na tela.

Com o curso escolhido, o *script* traz a informação dos alunos do curso chamando a função ‘core_enrol_get_enrolled_users’. Os alunos então são anonimizados usando uma função escrita por Jéferson Guimarães[4]. Para esse processo de anonimização, é criado um número inteiro aleatório para cada aluno. Posteriormente, esse número inteiro irá substituir o id do aluno.

O *script* irá executar as outras funções da API do Moodle para buscar os dados dos *logs* de cada aluno. Executando a função ‘mod_assign_get_assignments’ obtém-se as tarefas dos alunos, a partir delas, conseguimos as informações dos envios de cada tarefa com a função ‘mod_assign_get_submissions’. Para obter os dados das postagem em fóruns da disciplina, foi necessário executar a função ‘mod_forum_get_forums_by_courses’ que retorna todos os fóruns da disciplina. Com esse dado conseguimos trazer as discussões dos fóruns e as postagens dos alunos através das funções ‘mod_forum_get_forum_discussions’ e ‘mod_forum_get_discussion_posts’ respectivamente. A função que traz todos os *quizzes* da disciplina é a ‘mod_quiz_get_quizzes_by_courses’ e, após a sua execução, são trazidos os dados das tentativas do aluno para os *quizzes* com ‘mod_quiz_get_user_attempts’ e, para obter a melhor nota de cada aluno para cada *quiz*, usa-se ‘mod_quiz_get_user_best_grade’. A nota final da disciplina é retornada pela função ‘gradereport_user_get_grade_items’.

3.3 Consolidação dos dados

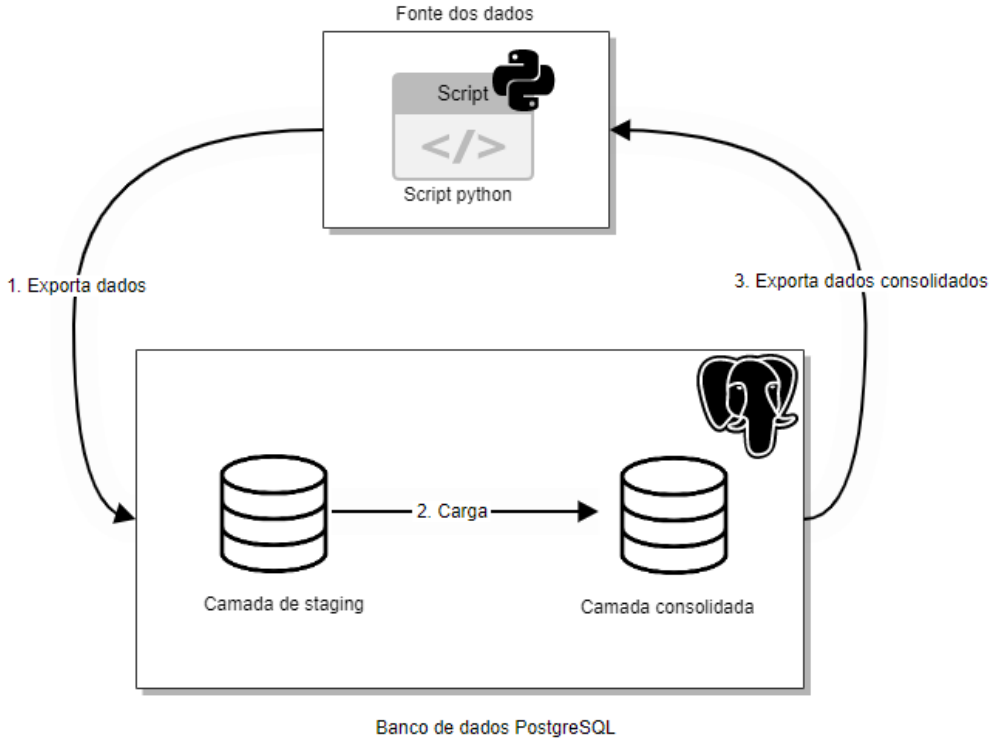
Após obter todos os *logs* dos alunos, é feita conexão com o banco de dados local PostgreSQL. Para a exportação dos dados no banco, foi criada uma camada de *staging* e uma camada consolidada. Segundo Shaker H. Ali El-Sappagh, Abdeltawab M. Ahmed Hendawi, Ali Hamed El Bastawissy (2011), a camada de *staging* contém todas as tabelas temporárias criadas durante o processo de extração ou resultantes de funções de transformação aplicadas. A exportação é feita seguindo o fluxo mostrado na Figura 3.2.

A escolha de utilizar um banco de dados ao invés de consolidar os dados no python utilizando *DataFrames* se deu porque, caso o professor queira acessar os dados dos logs intermediários gerados pela ferramenta, ele consiga de forma mais fácil. Também por escalabilidade, caso se queira alterar o projeto para executar em todas as disciplinas de um professor ou em todas as disciplinas de um curso, os dados serão salvos em estruturas

[4]<https://github.com/JefersonFG/moodle-log-anonymizer>

consolidadas de fácil acesso e manutenção.

Figura 3.2: Fluxo do dado no banco de dados



Fonte: Elaborado pelo autor.

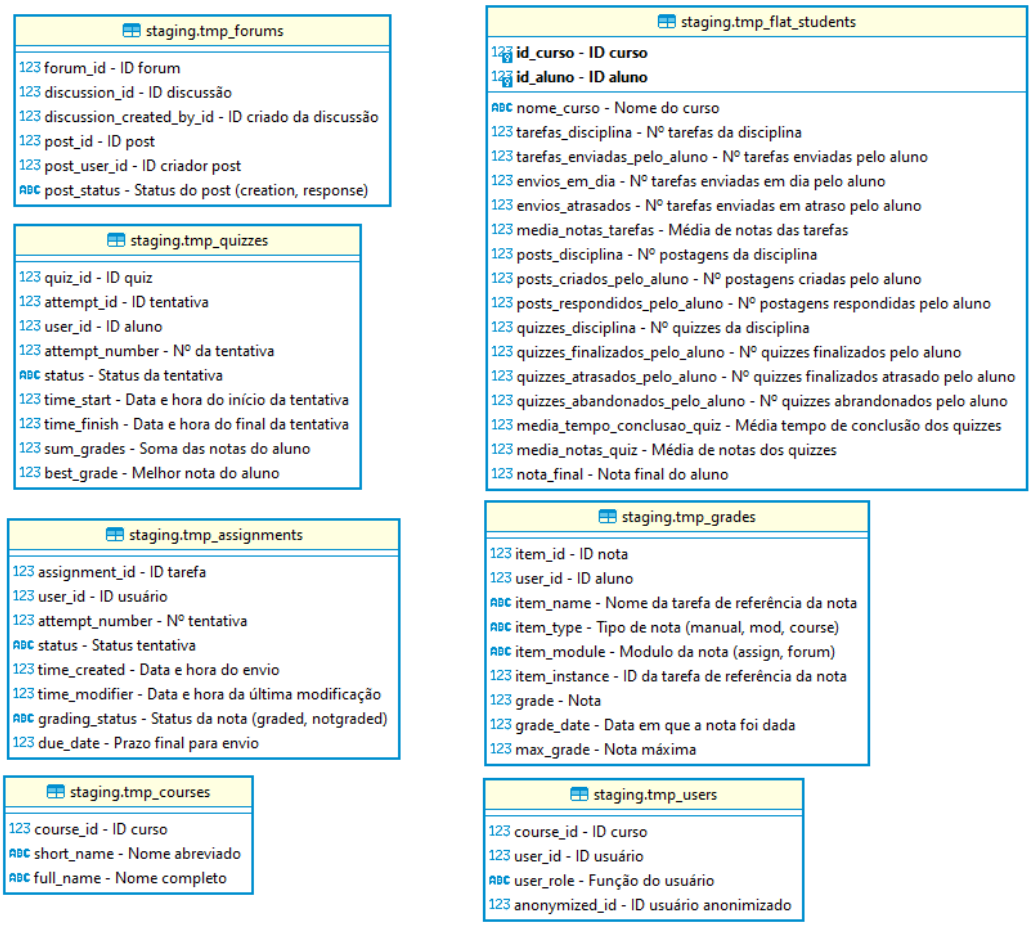
Para cada categoria de *log* (dados da disciplina, alunos, tarefas, fóruns, *quizzes* e notas) foi criada uma tabela e cada estrutura da camada de *staging* tem uma estrutura correspondente na camada agregada.

Foi escolhido criar uma camada intermediária no banco para que os dados sejam exportados e inseridos em sua totalidade. Como os *logs* dos alunos tendem a não crescer muito (devido ao tamanho de alunos em cada turma não ser muito alto) é feita uma carga completa do *log* de todos os alunos na camada intermediária.

O *script* irá deletar todos os registros das tabelas da camada de *staging* e irá inserir os dados coletados, via WS do Moodle, na tabela correspondente ao tipo de *log* (dados de alunos na tabela de alunos, tarefas na tabela de tarefas). O diagrama Entidade-Relacionamento (ER) da camada de *staging* está ilustrado na figura 3.3. A *procedure* (função no banco de dados) para carga de dados é chamada e a camada consolidada começa a ser populada. A *procedure* apaga todos os registros correspondentes que existirem na tabela consolidada e insere os novos. Então, quando o professor executar o *script* pela primeira vez, todas as tabelas estarão vazias e serão populadas. Caso o professor execute o *script* de novo, os registros que já existiam na camada consolidada serão atualizados e

os novos, caso existam, serão inseridos. O diagrama Entidade-Relacionamento da camada consolidada está ilustrado pela Figura 3.4

Figura 3.3: Diagrama ER da camada de staging



Fonte: Elaborado pelo autor.

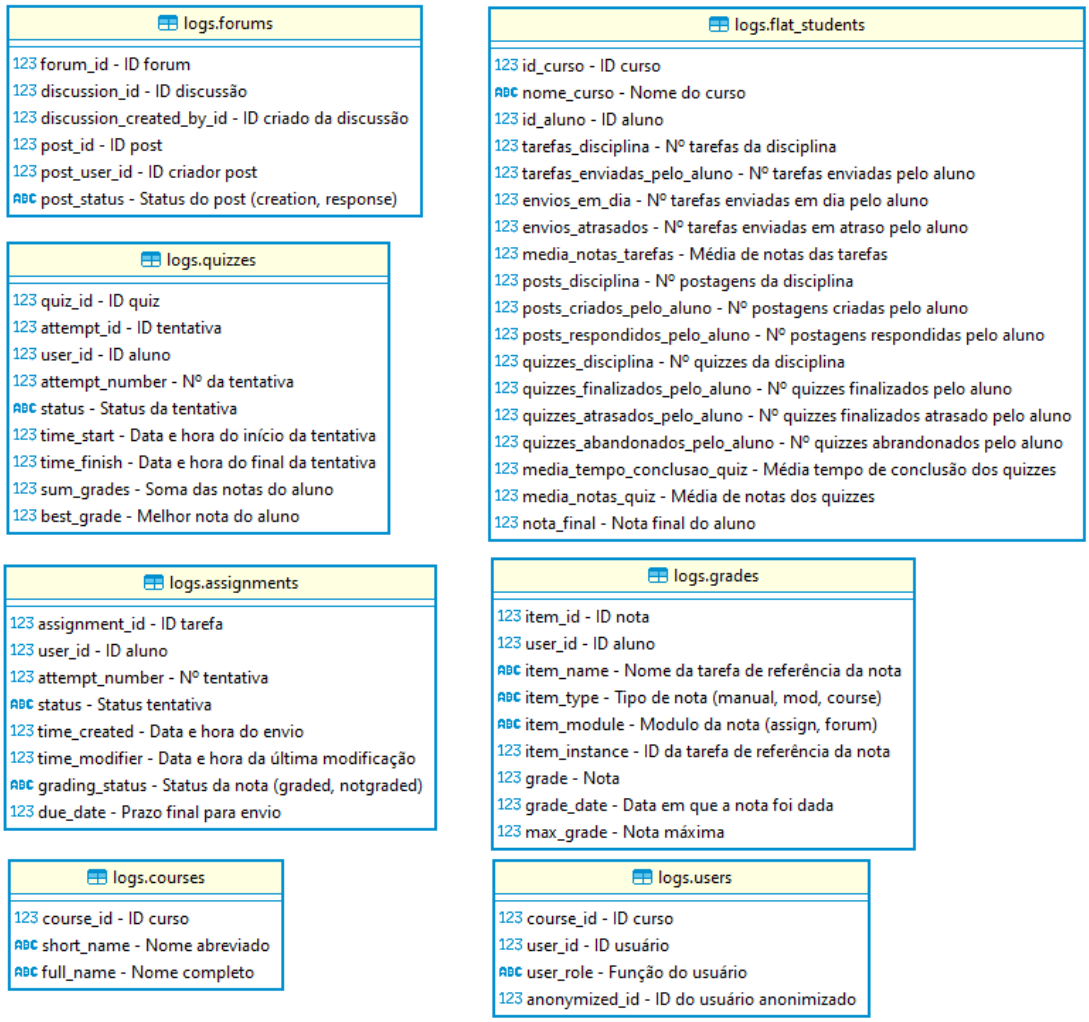
Para que os algoritmos de ML e correlação sejam aplicados são necessários um conjunto de dados agregados com todas as informações para as análises. Tendo esse ponto em vista, foi criada uma *procedure* que, após camada consolidada populada e atualizada, cria essa estrutura agregada juntando os dados da disciplina, do aluno, suas tarefas enviadas, sua participação nos fóruns, seus *quizzes* feitos e sua nota final. O processo é o mesmo de antes, deleta e insere na tabela da camada de *staging* e, logo após, atualiza/insere os registros que não estiverem na camada consolidada. A tabela com os dados agregados, resultante da execução da segunda *procedure*, está representada na Figura 3.4 pelo nome de 'logs.flat_students'.

Após tabela agregada pronta, seus dados são importados pelo *script* Python para serem realizadas as análises.

3.4 Exportação dos dados consolidados

Para que o professor tenha acesso a esses dados consolidados, antes de começar com as análises de ML e correlação, os dados são exportados para um arquivo no formato CSV. Esse arquivo também servirá como fonte de dados para as análises.

Figura 3.4: Diagrama ER da camada de staging



Fonte: Elaborado pelo autor.

3.5 Execução e exportação das análises

O *script* irá importar os dados, previamente exportados, colocando eles em uma estrutura chamada de $DataFrame$. Essa estrutura bidimensional é utilizada para armazenar dados alinhados de forma tabular em linhas e colunas. É semelhante a uma matriz, onde cada coluna pode possuir um tipo de dados diferente. Para importar o arquivo e

transformar em um $DataFrame$ foi utilizada a biblioteca 'pandas' do Python. Essa biblioteca permite manipular os dados de forma fácil e rápida utilizando $DataFrames$.

Após o $DataFrame$ ser populado, o próximo passo é executar as três análises propostas:

- Correlação de Spearman;
- Análise de agrupamento utilizando o algoritmo *K-médias++*;
- Análise de classificação utilizando o algoritmo de árvore de decisão.

Antes, porém, da execução de cada análise para sua otimização, é feito um pré-processamento nos dados. Pré-processamento de dados é um conjunto de atividades feitas antes da realização de alguma análise e envolve converter dados brutos em dados preparados. Em suma, é um processo que reduz a complexidade dos dados e oferece condições melhores para análises subsequentes (MALIK; GOYAL; SHARMA, 2010). Essas atividades incluem remover colunas sem dados ou com muitos dados faltando, preencher dados faltantes (quando houverem poucos), remover colunas com dados sem importância para análise.

3.5.1 Análise de correlação

A análise de correlação de Spearman compara todos os atributos (colunas) do conjunto de dados e traz a correlação de cada par. Além de conseguirmos trazer essa análise de uma forma bem visual com Python (utilizando a função 'heatmap' da biblioteca 'seaborn'), ter essa comparação entre cada par de atributos pode ser interessante para o professor. Alunos que entregarem as tarefas em dia tendem também a interagir mais no fórum, ou alunos que entregarem as tarefas em atraso tendem a abandonar os *quizzes* por exemplo.

Para calcular a correlação de Spearman, foi usado o método 'corr(method='spearman')' da biblioteca 'pandas'. A visualização, como mencionado acima, foi gerada com a função 'heatmap'. Esta função irá criar um mapa de calor, facilitando a visualização da análise.

O pré-processamento dos dados, para a execução do cálculo de correlação, foi feito seguindo os seguintes passos:

1. Retirar colunas sem importância para a análise (referentes à disciplina e aos alunos). Colunas retiradas:

- id_curso;
- nome_curso;
- id_aluno;
- tarefas_disciplina;
- posts_disciplina;
- quizzes_disciplina.

2. Retirar colunas sem dados ou com o mesmo dado em todas as linhas.

Com os dados pré-processados, o *script* Python irá gerar a análise e a sua visualização será salva em um arquivo de imagem no formato 'png' para maior facilidade de acesso.

3.5.2 Análise de agrupamento com *K-médias++*

O problema do algoritmo *K-médias* é a sua alta dependência da definição dos k pontos. Caso os pontos sejam definidos em locais "ruins" a análise fica comprometida. Por esse motivo, foi usado o algoritmo *K-médias++* neste projeto. Este algoritmo propõe uma forma específica de escolher os pontos iniciais (ARTHUR; VASSILVITSKII, 2006). Nele, o primeiro centroide é definido aleatoriamente e, a partir disso, os outros $k - 1$ pontos são escolhidos de forma que sejam o mais distantes possíveis dos centroides já definidos. Com isso, os k centroides tendem a ficar longe uns dos outros, aumentando as chances de selecionar centroides que se encontram em grupos diferentes.

A quantidade de grupos a serem gerados foi ajustada para 2 durante o desenvolvimento do projeto. Com isso tenta-se separar os alunos em desanimados e não desanimados.

Para executar o algoritmo deve-se primeiro pré-processar os dados e depois escolher a quantidade de grupos que se quer gerar. O pré-processamento dos dados foi realizado seguindo os seguintes passos:

1. Retirar colunas sem importância para a análise (referentes à disciplina e aos alunos). Colunas retiradas:

   - id_curso;
   - nome_curso;
   - id_aluno;

- tarefas_disciplina;
- posts_disciplina;
- quizzes_disciplina.

2. Retirar colunas sem dados ou com o mesmo dado em todas as linhas;
3. Preencher valores NaN ($NotaNumber$, quando um valor que deveria ser numérico consta como nulo). No nosso contexto, dois atributos podem conter NaN (os outros atributos são setados para 0 quando não houver valor):
   - media_tempo_conclusao_quiz: Como para esse atributo quanto menor o tempo de conclusão do quiz melhor (o aluno demorou menos tempo para fazer realizar a prova), foi considerado que valores nulos teriam o valor de o dobro do maior tempo de conclusão do quiz;
   - media_notas_quiz: Neste caso, quanto maior o valor da média de notas do quiz melhor, os NaN foram considerados como sendo 0.
4. Fazer a padronização dos dados. Utilizando a função 'preprocessing.StandardScaler' da biblioteca python 'sklearn'. Essa padronização de dados é feita para que valores maiores não sejam considerados mais importantes do que outros no cálculo do algoritmo.

Com os dados pré-processados, o algoritmo pode ser executado. Foi utilizada a função 'cluster.KMeans' da biblioteca 'sklearn' para a execução.

Com o algoritmo executado e os *clusters* definidos, os dados são exportados para um arquivo em formato 'xlsx'. O professor poderá olhar para todos os dados dos alunos de cada *cluster*. Além disso, para que o professor tenha mais uma análise disponível, para cada grupo de alunos, foi utilizada a função 'ProfileReport' da biblioteca 'pandas_profiling' para exportar esses dados em um arquivo no formato 'html'. Este arquivo interativo pode ser executado em qualquer navegador de internet e traz informações interessantes sobre cada variável (o tipo, a média de valores, a quantidade de valores faltantes, o percentual de valores distintos, a frequência de cada valor), a correlação entre as variáveis, a quantidade de registros duplicados. Traz uma série de análises que podem ser úteis quando o professor for tentar entender o perfil de cada grupo de alunos.

3.5.3 Análise de classificação por Árvore de decisão

Diferente do algoritmo visto na etapa anterior, onde não se tinha uma variável classe (de interesse) e o foco era buscar padrões, neste algoritmo agora, procuramos classificar os dados de acordo com uma variável de interesse. No contexto do trabalho, a variável de interesse é a envios_em_dia (quantidade de tarefas enviada dentro do prazo) e o objetivo do algoritmo é entender o que leva um aluno a enviar as tarefas em dia, de acordo com suas atividades na disciplina.

Para execução do algoritmo, os dados foram pré-processados conforme os seguintes passos:

1. Retirar colunas sem importância para a análise (referentes à disciplina e aos alunos). Colunas retiradas:
    - id_curso;
    - nome_curso;
    - id_aluno;
    - tarefas_disciplina;
    - posts_disciplina;
    - quizzes_disciplina.
2. Retirar colunas sem dados ou com o mesmo dado em todas as linhas;
3. Preencher valores NaN (*Not a Number*, quando um valor que deveria ser numérico consta como nulo). No nosso contexto, dois atributos podem conter NaN (os outros atributos são setados para 0 quando não houver valor):
    - media_tempo_conclusao_quiz: Como para esse atributo quanto menor o tempo de conclusão do quiz melhor (o aluno demorou menos tempo para fazer realizar a prova), foi considerado que valores nulos teriam o valor de o dobro do maior tempo de conclusão do quiz;
    - media_notas_quiz: Neste caso, quanto maior o valor da média de notas do quiz melhor, os NaN foram considerados como sendo 0.
4. Algumas disciplinas não permitem o envio de tarefas atrasadas. Com isso, o atributo 'tarefas_enviadas_pelo_aluno' será igual ao nosso atributo alvo 'envios_em_dia'. Caso isso aconteça, para que o atributo de tarefas enviadas pelo aluno não seja considerado o mais importante, ele será removido da análise.

Com o pré-processamento dos dados feito, o algoritmo será executado utilizando a biblioteca ‘sklearn’ e a função ‘tree.DecisionTreeClassifier’. a árvore de decisão resultante será salva em um arquivo no formato ‘png’ e estará disponível para o professor acessar. Essa árvore pode ajudar o professor a entender o que leva um aluno a entregar mais ou menos tarefas em dia em sua disciplina.

4 GUIA DE USO

Este capítulo tem como intuito mostrar as configurações e os passos a seguir para conseguir executar o projeto. A seção a seguir traz uma visão dos requisitos necessários para a execução, as próximas seções seguirão o pipeline do projeto, previamente demonstrado pela Figura 3.1.

4.1 Requisitos

Para conseguir executar a ferramenta é necessário ter um Moodle (pode ser local), um servidor de banco de dados PostgreSQL (pode ser local) e a linguagem python instalada. A tabela 4.1 mostra as versões mínimas e recomendadas necessárias para cada ferramenta.

Tabela 4.1: Versões necessárias e recomendadas para cada ferramenta.

Ferramenta	Versão mínima	Versão recomendada
Moodle	Moodle 2.3	Moodle 3.10 ou superior
PostgreSQL	PostgreSQL 9.6	PostgreSQL 11.2 ou superior
Python	Python 3	Python 3.7.1 ou superior

Fonte: Elaborado pelo autor

Qualquer Moodle (mantendo a versão mínima satisfeita) pode ser utilizado, desde que seja configurado da maneira correta (instruções nas seções a seguir) e que seu URL seja conhecido.

4.2 Configuração do Moodle

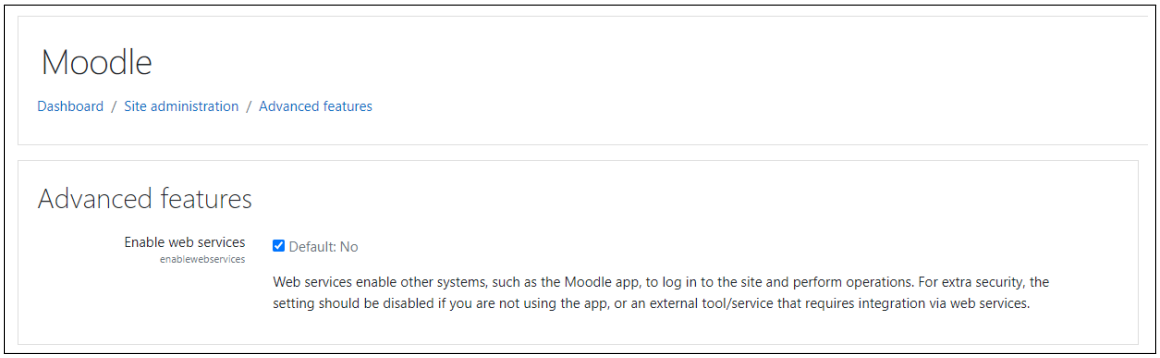
Para que o *script* Python consiga acessar a API do Moodle e obter os dados dos *logs* dos alunos, o Moodle precisa estar com os WB habilitados e permitir a execução das funções necessárias. Esta seção ajudará com essa parte. O site do Moodle também possui um tutorial que pode ser utilizado e está disponível na internet[1].

[1] <https://docs.moodle.org/400/en/Using_web_services>

4.2.1 Habilitando os *Web Services* no Moodle

Para habilitar os WB, deve-se navegar até “Site Administration” (Administração do site) - Advanced Features (Opções Avançadas) e marcar a opção “Enable web services” (habilitar serviços web). O caminho e a opção estão descritos na figura 4.1

Figura 4.1: Caminho no Moodle e opção para habilitar os WB



Fonte: Elaborado pelo autor a partir do site do Moodle

4.2.2 Habilitando protocolo REST

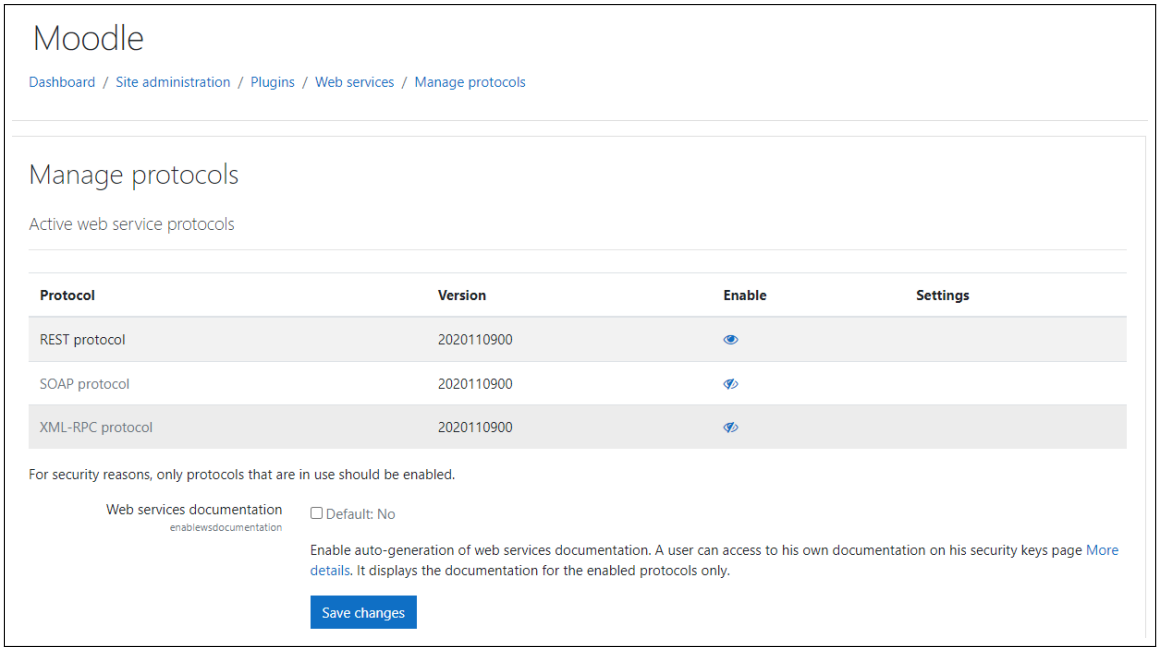
Além de habilitar os WB, o protocolo REST também precisa ser ser habilitado. Para isso, navegar até “Site Administration” (Administração do site) - “Plugins” - “Web services” (Serviços web) - “Manage protocols” (Gerenciar protocolos) e clicar no símbolo de olho na linha do protocolo REST. Para que as mudanças sejam salvas, clicar em “Save Changes” (Salvar mudanças). A opção marcada deve ficar como na figura 4.2.

4.2.3 Habilitando as funções da API

Para que o *script* chame a API do Moodle, deve-se conectar ao Moodle com as credenciais de um usuário que tenha acesso às funções que a API irá executar. A tabela 3.1 previamente mostrada, possui todas as funções que o usuário deverá ter acesso. Para habilitar acesso aos métodos deve-se criar um serviço externo no Moodle. Um serviço é um conjunto de funções que podem ser acessadas (via API) por um conjunto de usuários especificados. No nosso caso, vamos criar um serviço em que apenas o usuário do professor tenha acesso.

Para criar o serviço, deve-se navegar até “Site Administration” (Administração

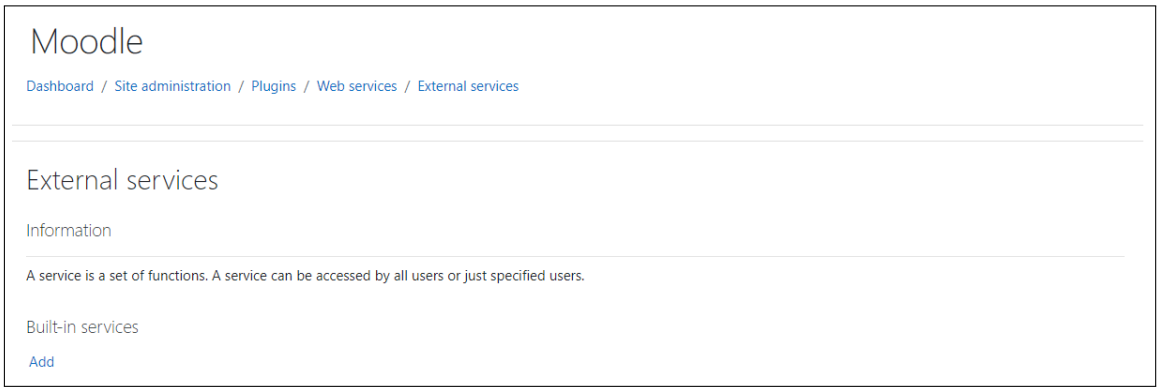
Figura 4.2: Caminho no Moodle e opção para habilitar o protocolo REST



Fonte: Elaborado pelo autor a partir do site do Moodle.

do site) - “Plugins” - “Web Services” (Serviços Web) - “External Services” (Serviços Externos) e clicar em “Add” (Adicionar). O Caminho e a opção estão descritos na figura 4.3.

Figura 4.3: Caminho no Moodle e opção para criar um serviço externo

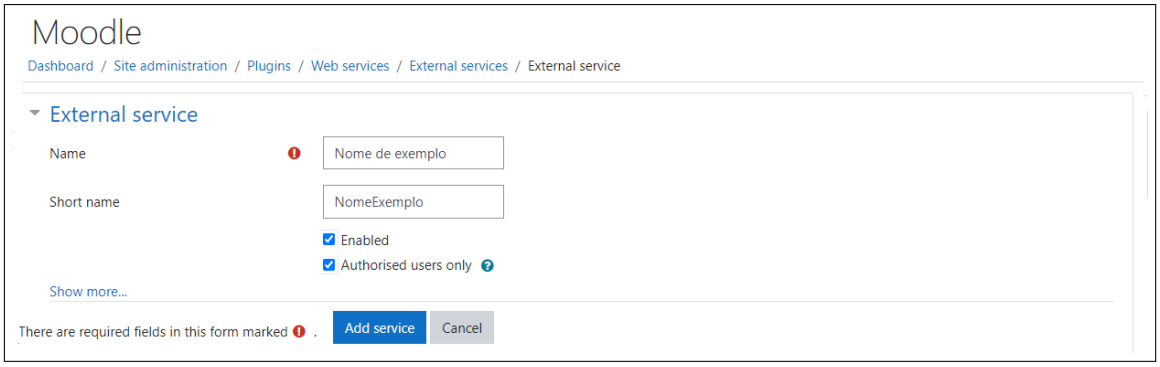


Fonte: Elaborado pelo autor a partir do site do Moodle.

Depois de clicar em “Add”, o próximo passo é escolher um nome para o serviço, marcar a opção “Enabled” (Habilitado) e também marcar a opção “Authorised users only” (Apenas usuários autorizados), marcando esta última opção, garantiremos que apenas o professor tenha acesso aos dados via API. Depois de preenchidos os campos, clicar em “Add Service” (Adicionar serviço). A página deverá ser preenchida de acordo com a figura 4.4.

Aparecerá uma página falando que o serviço não possui funções. Clicando em

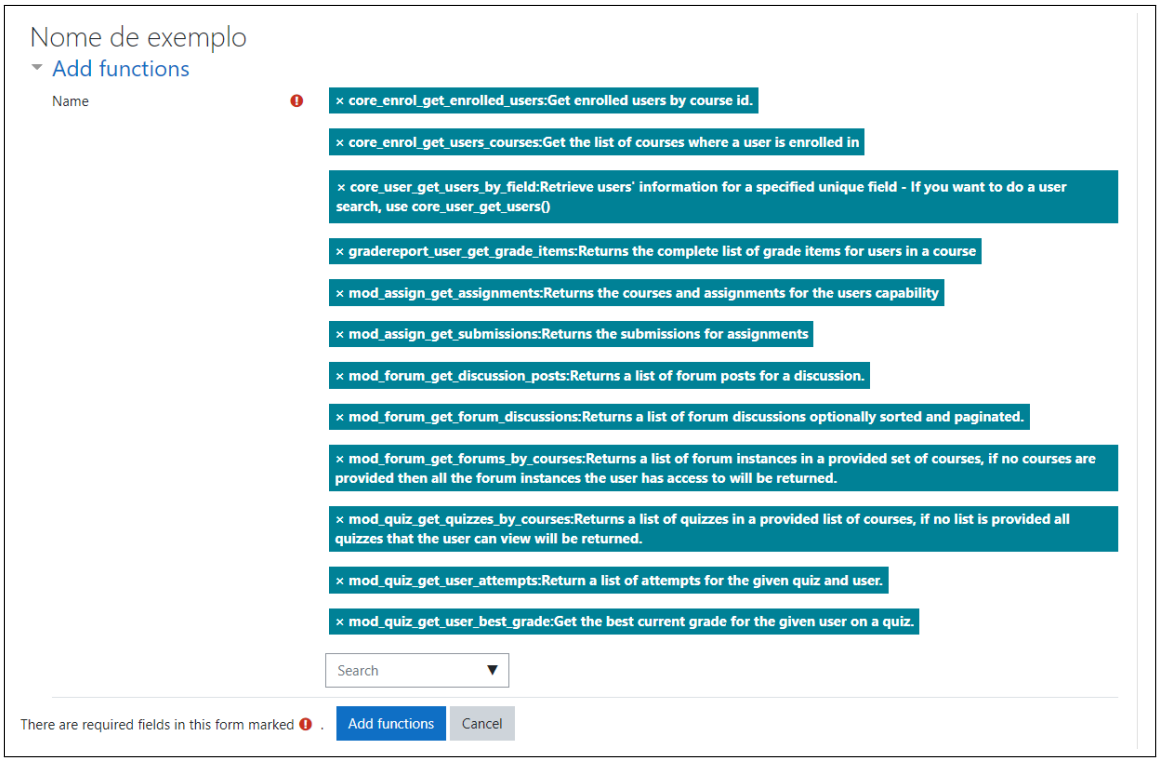
Figura 4.4: Caminho no Moodle e opção adicionar um serviço externo



Fonte: Elaborado pelo autor a partir do site do Moodle.

"Add functions" (Adicionar funções) o Moodle direcionará para outra página, nela poderemos colocar todas as funções da API que o *script* Python irá utilizar. No campo "Search" (Buscar) devem ser colocas as funções de acordo com a tabela 3.1. A página deverá ter o formato da figura 4.5 quando todas as funções forem adicionadas. Clicar em "Add Functions" (Adicionar funções).

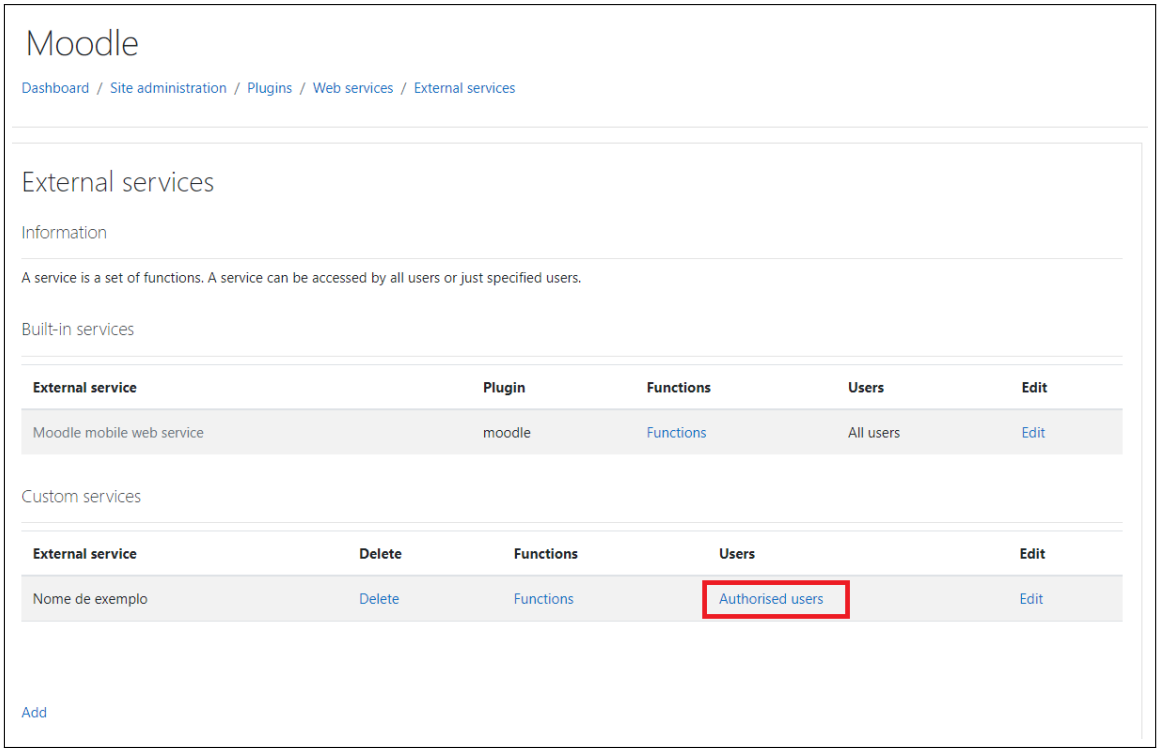
Figura 4.5: Funções a serem adicionadas no serviço externo criado



Fonte: Elaborado pelo autor a partir do site do Moodle.

Agora, para que o professor tenha acesso ao serviço criado, clicar em "Authorised users" (Usuários autorizados) na página de serviços externos como mostrado em vermelho pela figura 4.6

Figura 4.6: Opção a ser selecionada para adicionar um usuário ao serviço externo criado



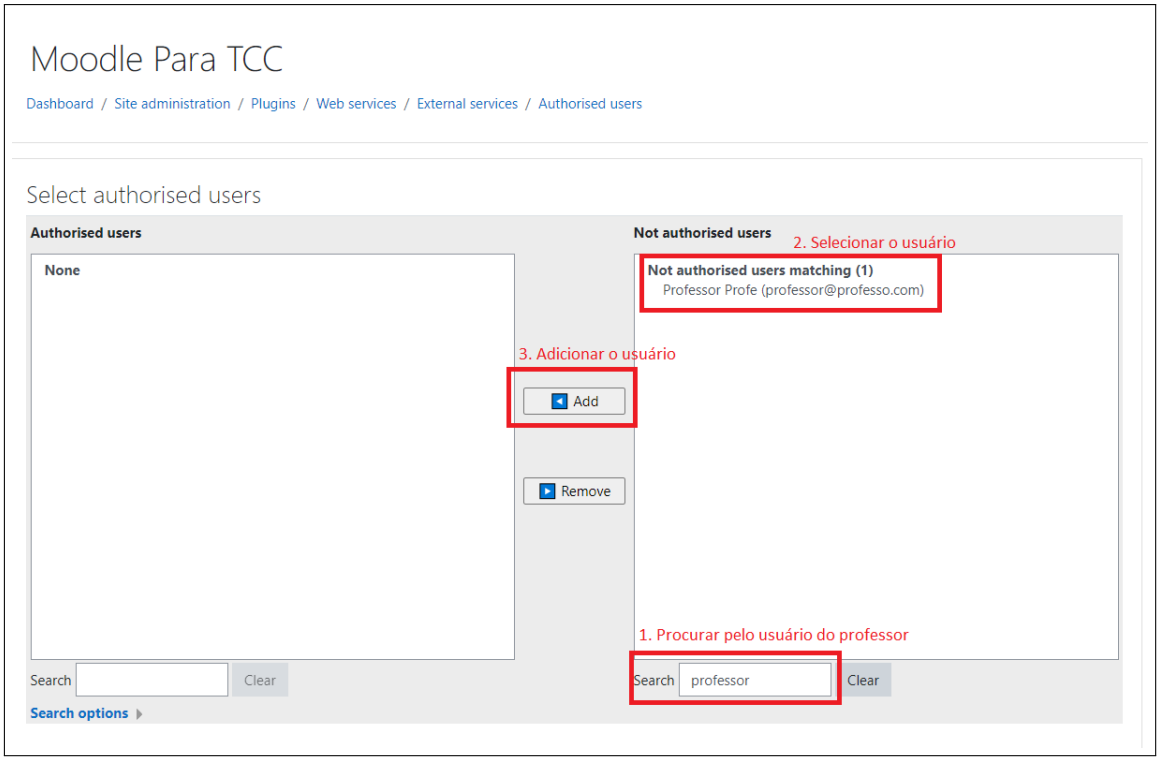
Fonte: Elaborado pelo autor a partir do site do Moodle.

Na nova página, o quadro à direita mostrará todos os usuários sem acesso ao serviço e, à esquerda, todos com acesso. Como o serviço foi recém criado, não haverá nenhum usuário no quadro da esquerda. Para adicionar o professor, selecionar o seu usuário no quadro da direita. Caso ele não apareça no quadro, deve ser feita uma busca pelo seu usuário usando o campo "Search" (Buscar). Após usuário encontrado e selecionado, clicar no botão "Add" (Adicionar) para que o professor seja adicionado à lista de usuários com acesso ao serviço. Esse processo está demonstrado na figura 4.7.

4.2.4 Criando um *token* de acesso

Um *token*, no contexto deste trabalho, é uma chave, vinculada a um usuário específico, usada para se conectar com a API do Moodle. O intuito do *token* é não precisar utilizar todas as credenciais do usuário na hora de fazer a conexão com o Moodle. Para criar essa chave, deve-se navegar até "Site administration" (Administração do site) - "Plugins" - "Web services" (Serviços web) - "Manage tokens" (Gerenciar Tokens) e clicar em "Add" (Adicionar). A figura 4.8 demonstra o caminho completo no Moodle e o visual da página.

Figura 4.7: Passo a passo para adicionar um usuário ao serviço externo criado



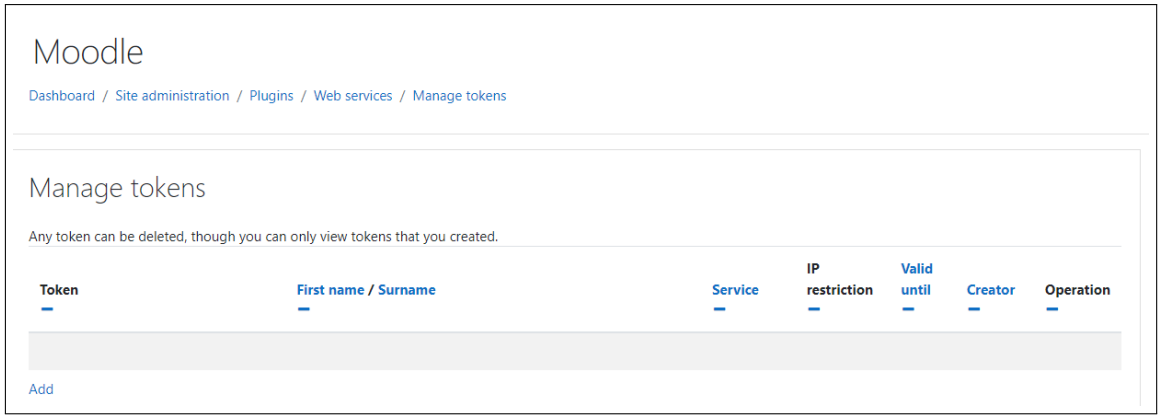
Fonte: Elaborado pelo autor a partir do site do Moodle.

Na próxima página, o *token* será vinculado a um usuário. Como este trabalho é voltado para uso do professor, o usuário a ser vinculado ao *token* deve ser o usuário do professor.

Na página, colocar o usuário escolhido utilizando o campo com “Search” (Buscar) escrito dentro. Após usuário selecionado, na opção “Service” (Serviço), escolher o serviço criado na etapa anterior e clicar em “Save Changes” (Salvar mudanças). A página deverá estar com os campos preenchidos como na figura 4.9.

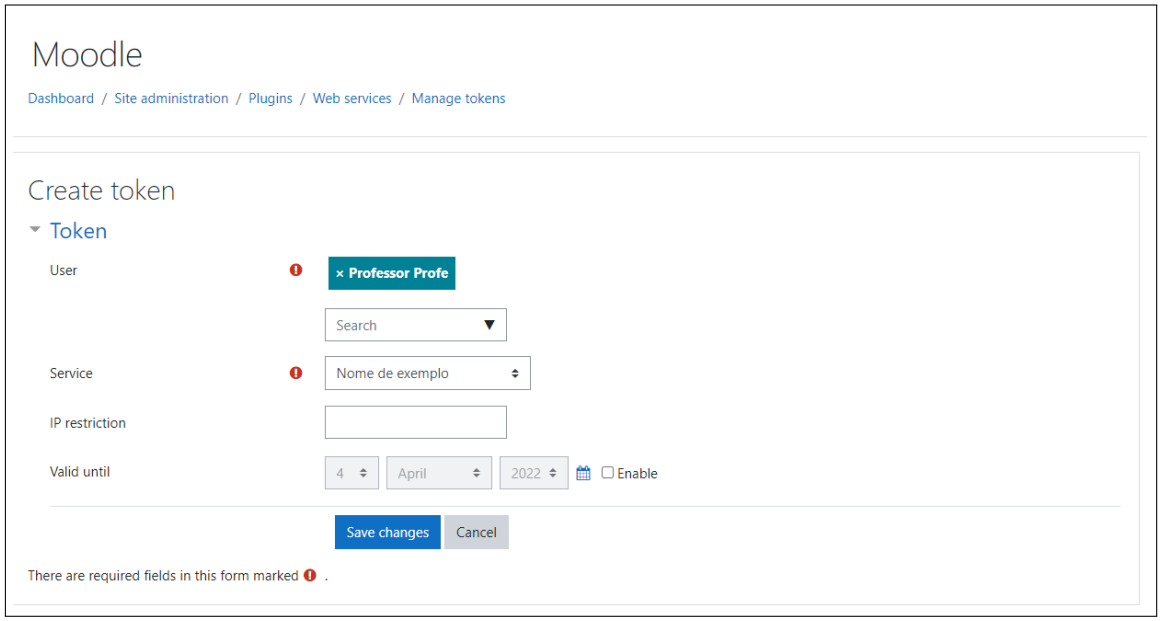
Com o *token* criado, para se conectar ao Moodle via *script*, será necessário o uso do seu número. Para encontrar o número do *token* criado, navegar até a página de gerenciamento de *tokens* como demonstrado pela figura 4.10. O número estará abaixo da coluna *Token*. No exemplo da figura 4.10, ele está marcado em vermelho (e foi truncado por privacidade).

Figura 4.8: Página do Moodle para adicionar novo token



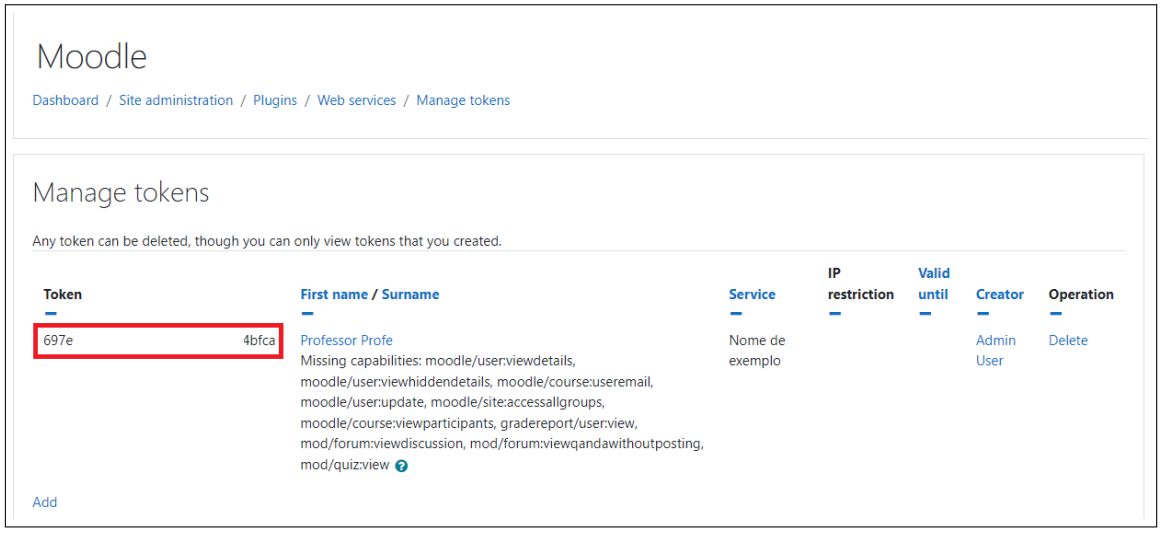
Fonte: Elaborado pelo autor a partir do site do Moodle.

Figura 4.9: Adicionando um novo *token*, configuração da página



Fonte: Elaborado pelo autor a partir do site do Moodle.

Figura 4.10: Exemplo de *token* gerado

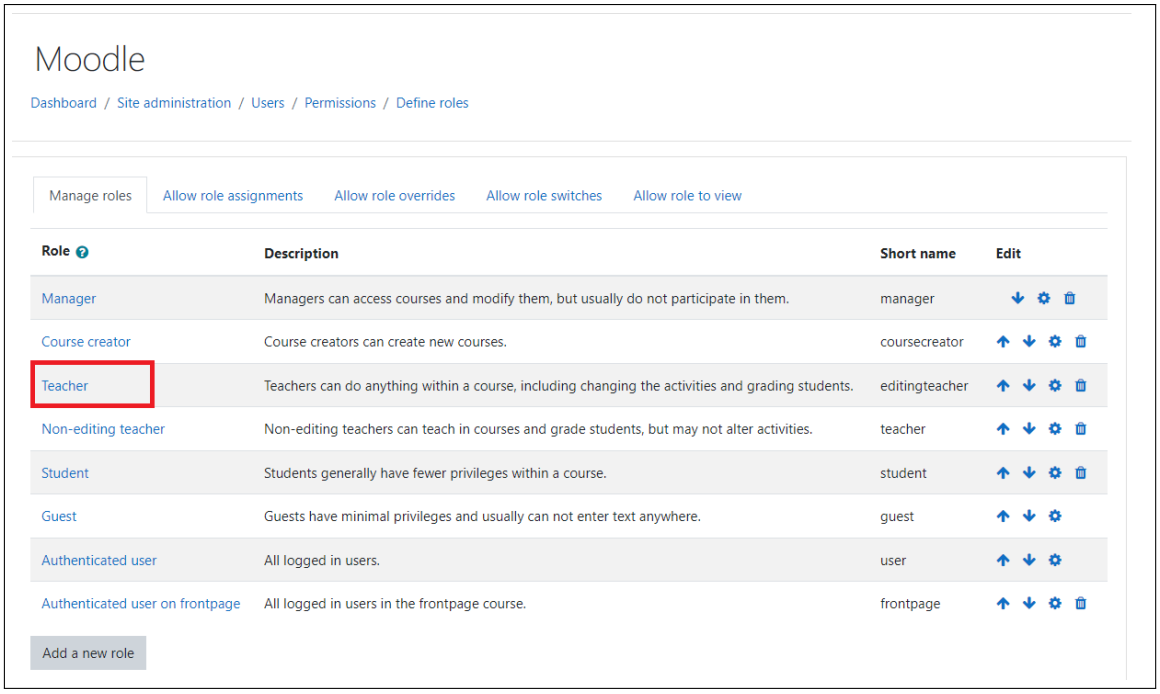


Fonte: Elaborado pelo autor a partir do site do Moodle.

4.2.5 Adicionando permissão à função de Professor

A função de Professor no Moodle ainda precisa de uma permissão para que o usuário professor consiga acessar o serviço criado. Para isso, navegar até “Site administration” (Administração do site) - “Users” (Usuários) - “Permissions” (Permissões) - “Define roles” (Definir funções) e selecionar “Teacher” (professor). A figura 4.11 mostra o caminho e a opção a ser selecionada (em vermelho).

Figura 4.11: Caminho e opção a para adicionar uma permissão à função Professor



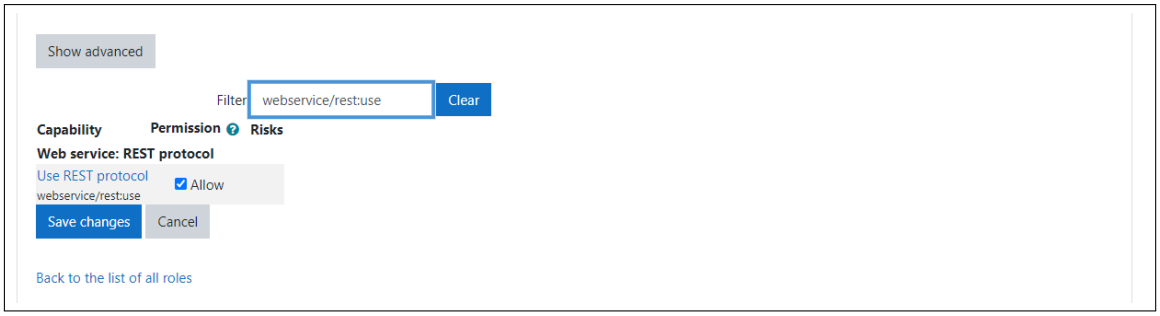
Fonte: Elaborado pelo autor a partir do site do Moodle.

Clicar em ‘ ´Edit” (Editar). Na parte de baixo da página irá aparecer uma tabela com as competências, permissões e riscos. Logo acima da tabela existe um campo de filtro. Neste campo, escrever “webservice/rest:use”, marcar a caixa permitindo este acesso e clicando em “Save Changes” (Salvar mudanças). A página com a opção deve ser como na figura 4.12. Marcando esta permissão, um usuário com função de Professor conseguirá acessar o serviço do Moodle que foi criado.

4.3 Configuração do Python

A linguagem de programação Python assim como suas bibliotecas precisam ser instaladas. A versão do Python que foi utilizada para criação deste projeto foi “Python

Figura 4.12: Opção a ser marcada permitindo acesso ao protocolo REST



Fonte: Elaborado pelo autor a partir do site do Moodle.

3.7.1", mas qualquer outra versão do Python 3 conseguirá realizar a análise. Um passo a passo fácil para instalação está disponível na internet[2]. É recomendado fazer a instalação usando o instalador completo com as opções padrões e deve-se marcar a opção para adicionar o Python ao "PATH" na hora da instalação.

A lista completa de bibliotecas (assim como suas versões utilizadas durante o desenvolvimento do projeto) se encontram na tabela 4.2.

Tabela 4.2: Bibliotecas do python e suas verões

Biblioteca	**Versão**
pandas	1.3.5
pandas-profiling	3.1.0
scikit-learn	1.0.2
psycopg2	2.9.1
seaborn	0.11.2
matplotlib	3.5.1
matplotlib-inline	0.1.3

Fonte: Elaborado pelo autor.

Um arquivo com os requisitos chamado de "requirements.txt" foi disponibilizado junto com o projeto. Usando um comando na linha de comando, o usuário consegue instalar todas as bibliotecas para executar o *script* em Python. Para isso, abra o *prompt* de comando (Vá em menu iniciar e digite 'cmd' na busca), navegue, via linha de comando, até a pasta onde está o arquivo de requisitos e digite o seguinte comando:

```
pip install -r requirements.txt
```

[2]<https://docs.python.org/pt-br/3/using/windows.html#the-full-installer>

Para mais informações sobre como abrir o prompt de comando e navegar entre pastas, o site oficial da microsoft[3] possui um manual muito útil.

4.4 Configuração do PostgreSQL

Para consolidação dos dados, é preciso ter um banco de dados postgreSQL rodando. Um tutorial para instalação do postgreSQL está disponível através do manual oficial deste SGBD[4]. Utilize o tutorial até criar o banco de dados. Durante a instalação, será pedido que se crie uma senha para o super usuário, a porta onde ocorrerá a conexão com o servidor e, na criação do banco, o nome para o banco. Essas informações serão utilizadas depois, na hora de realizar a conexão do banco de dados com o Python.

Feita a instalação, é necessário que um arquivo contendo as estruturas e funções do banco de dados seja importado. Para isso, acessar o github do autor[5] e fazer *download* do arquivo "dump-postgres". Após *download* do arquivo de *backup*, importar no seu PostgreSQL. Para isso, pode-se usar a ferramenta pgAdmin (disponível na internet[6]. A ferramenta pgAdmin é de fácil uso e a importação do arquivo com as estruturas e funções pode ser feito em poucos passos.

O manual[7] da ferramenta pgAdmin traz um tutorial para fazer a importação de arquivos de backup. Os passos deste tutorial estão descritos a seguir.

4.4.1 Importando arquivo de backup

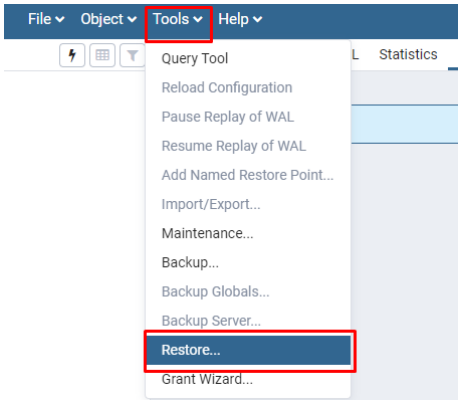
Com o pgAdmin aberto, ir até a aba "Tools" (ferramentas) e clicar na opção "Restore" (restaurar) como mostrado na figura 4.13.

Na janela aberta, na opção "Filename" (nome do arquivo) deverá clicar na opção mostrada pela figura 4.14.

Feito isso, outra janela irá abrir trazendo a visão de um explorador de arquivos. Nele, navegar até a pasta em que foi feito o *download* do arquivo de *backup*. Caso o arquivo não esteja aparecendo na tela, clicar na opção "Format" (formato) no canto inferior

[3]<https://docs.microsoft.com/pt-br/windows-server/administration/windows-commands/windows-commands>
[4]<https://www.postgresql.org/docs/14/index.html>
[5]<https://github.com/brunofeil/TCC>
[6]<https://www.pgadmin.org/download/>
[7]<https://www.pgadmin.org/docs/pgadmin4/development/restore_dialog.html>

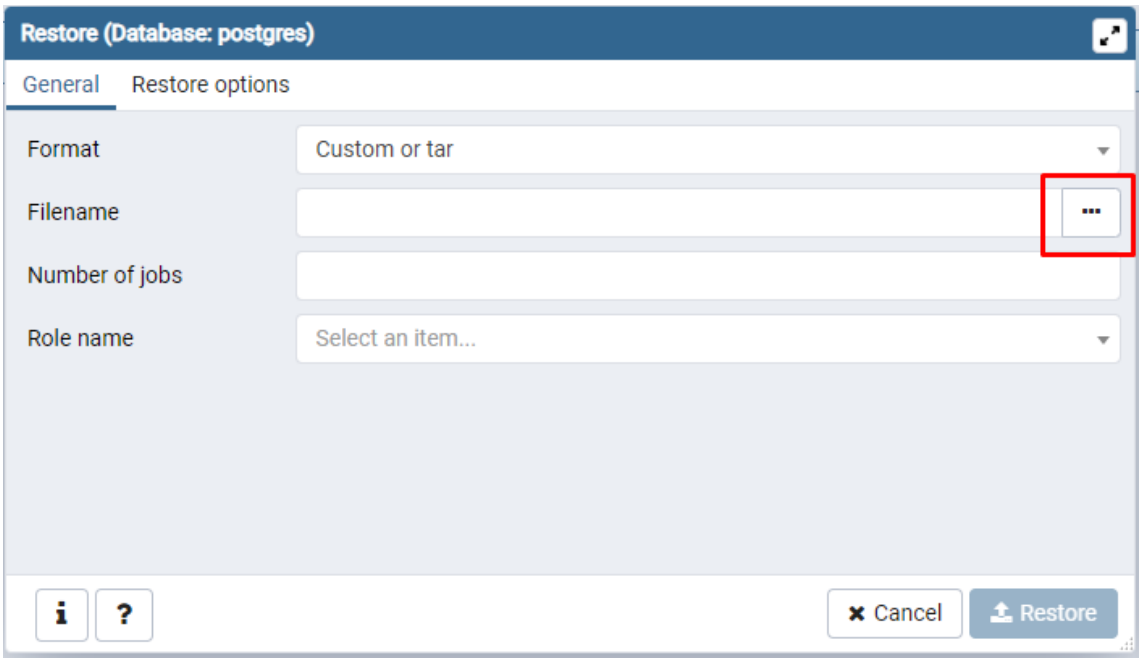
Figura 4.13: Começo da importação do arquivo de estruturas e funções do banco de dados



Fonte: Elaborado pelo autor.

esquerdo e escolher a opção “All files” (todos os arquivos). Com isso, todos os arquivos, não só os com os formatos especificados, aparecerão no explorador. Selecionar o arquivo de *backup* e clicar em “Select” (selecionar) para selecionar o arquivo a ser importado. A figura 4.15 demonstra este processo.

Figura 4.14: Opção a marcar para selecionar o arquivo a ser importado

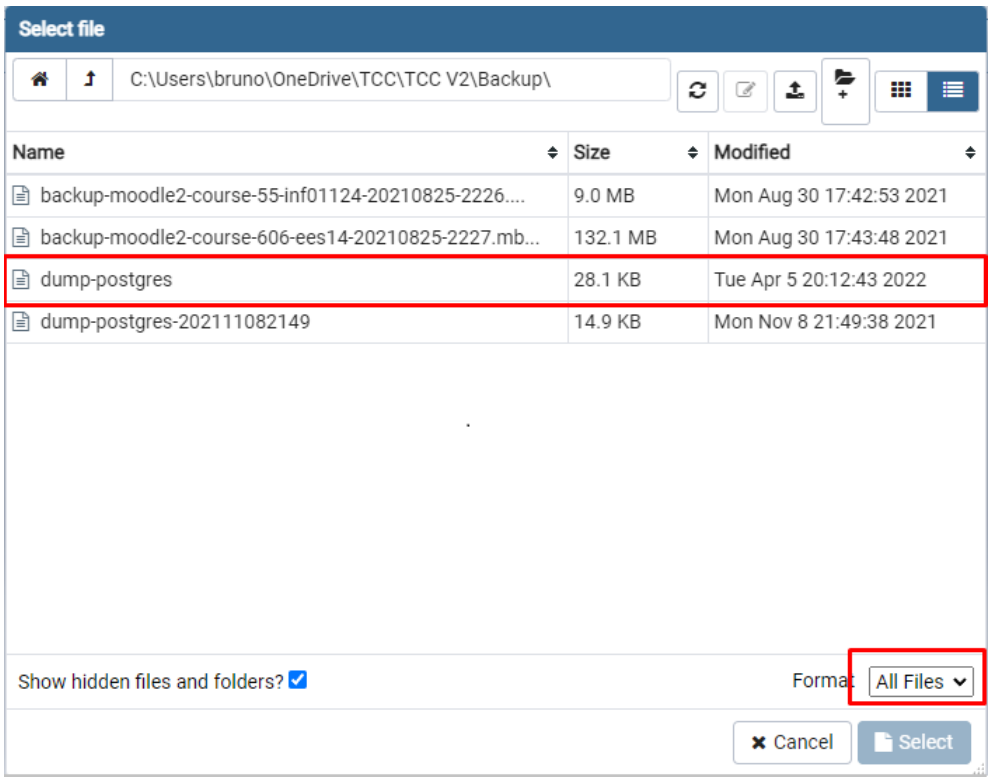


Fonte: Elaborado pelo autor.

Com o arquivo selecionado, clicar em “Restore” e o *backup* começará a ser importado. Como são poucos dados e o arquivo é pequeno, este processo não deverá levar muito tempo.

Agora o banco de dados está configurado e contém todas as estruturas (tabelas) e funções necessárias. Com isso, só falta realizar algumas mudanças no *script* python para

Figura 4.15: Selecionando o arquivo a ser importado



Fonte: Elaborado pelo autor.

que a ferramenta possa ser executada com sucesso.

4.5 Mudança no *script* Python

Para execução do projeto, algumas mudanças precisam ser feitas no *script* Python. A primeira mudança será colocar as informações para conexão com o banco de dados. Para isso, alterar as seguintes linhas do código:

```
PGHOST = ''
PGDATABASE = ''
PGUSER = ''
PGPASSWORD = ''
PORT = ''
```

A Tabela 4.3 descreve o que deve ser colocado em cada variável.

Tabela 4.3: Variáveis referentes ao banco de dados a serem alteradas

Variável	Descrição
PGHOST	Host do banco de dados, se for local, utilizar 'localhost'.
PGDATABASE	Nome do banco de dados.
PGUSER	Nome do usuário (pode ser usado o super usuário).
PGPASSWORD	Senha do banco de dados.
PORT	Porta para conexão.

Fonte: Elaborado pelo autor.

As próximas alterações serão referentes à conexão com o Moodle. Alterar as seguintes variáveis:

```
moodle_api.URL = ''
moodle_api.KEY = ''
user_name = ''
```

A Tabela 4.4 descreve o que deve ser colocado em cada variável.

Tabela 4.4: Variáveis referentes ao Moodle a serem alteradas

Variável	Descrição
moodle_api.URL	URL do Moodle
moodle_api.KEY	Token criado na figura 4.9
user_name	Usuário do Moodle

Fonte: Elaborado pelo autor.

4.6 Execução do *script*

O *script* Python pode ser executado via linha de comando. Para isso, abra o *prompt* de comando, navegue até a pasta "main" onde está o arquivo executa_ferramenta.py e digite o seguinte comando:

```
> python .\executa_ferramenta.py
```

Depois do comando executado, o *script* listará todas as disciplinas que o usuário (colocado no código) participa e uma delas deverá ser escolhida para realizar a análise.

5 VALIDAÇÃO E RESULTADOS

O projeto foi testado em uma disciplina de graduação do curso de Ciência da Computação da UFRGS, ministrada em 2014. A disciplina conta com 57 alunos e não teve *quizzes* feitos pelo Moodle, então esse dado não estava disponível para análise. Houve uma dificuldade para executar a extração dos dados via Moodle institucional da UFRGS. Por algum parâmetro de segurança, algumas funções da API não puderam ser executadas com o usuário de aluno do autor. Sendo assim, o projeto foi testado localmente, importando a disciplina em uma instalação local do Moodle e criando um usuário com a função de professor.

Todos os passos do capítulo 3 foram seguidos e os resultados obtidos em cada algoritmo estão nas próximas seções.

5.1 Correlação

O resultado obtido a partir da análise de correlação está presente na Figura 5.1. A figura mostra os níveis de correlação entre os pares de variáveis através de cores e valores. As cores refletem os valores. Quanto mais próximo de roxo, maior é a correlação (escala positiva), quanto mais próximo de azul, menor a correlação (escala negativa). Quanto aos valores, quanto mais próximo de 1, maior a correlação, quanto mais próximo de -1, menor. Caso o número seja 0, a correlação é nula.

Para interpretar o gráfico, deve-se selecionar as duas variáveis que se quer ver a correlação. O nível de correlação estará na intersecção entre as estas variáveis.

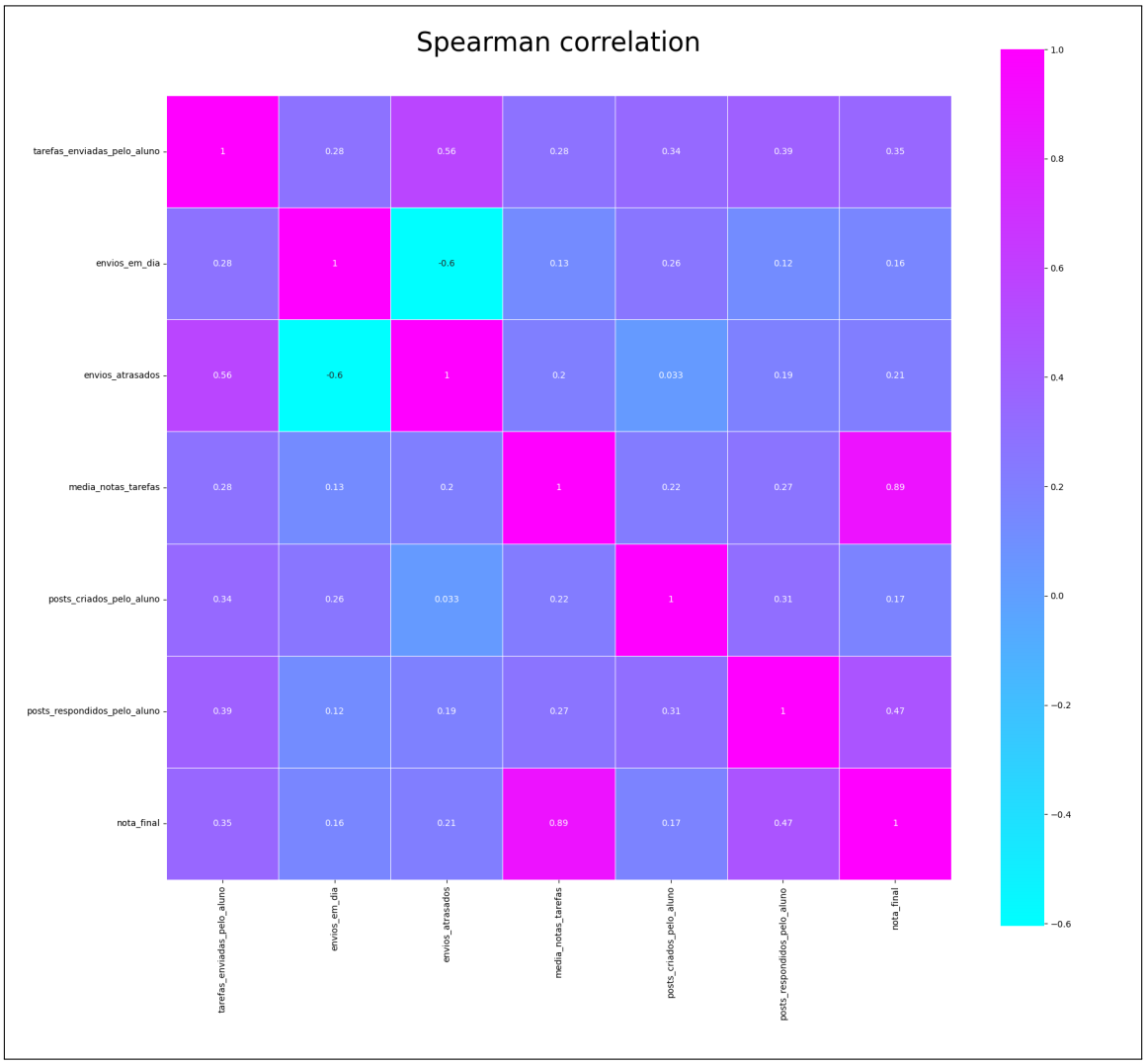
Nota-se uma forte correlação entre a nota final e a média de notas das tarefas (coeficiente de 0.89). Como as notas das tarefas compõem parte da nota final da disciplina, esta é uma correlação que parece óbvia, assim como a correlação negativa entre envio de tarefas dentro e fora do prazo (coeficiente de -0.6). Quanto mais tarefas forem enviadas fora do prazo, menos tarefas são enviadas dentro do prazo, e vice-versa.

Algumas informações interessante que se pode descobrir com essa visualização estão descritas a seguir. Considerando o parâmetro de Dancey & Reidy (visto anteriormente no capítulo 2, tabela 2.1), temos uma correlação média entre a nota final e a quantidade de *posts* respondidos pelos alunos (coeficiente de 0.47). Podemos dizer que alunos mais engajados, que interagem nos fóruns, tendem a ter uma nota maior. Há também uma correlação fraca/moderada entre a quantidade de *posts* respondidos pelos aluno com

a quantidade de tarefas enviadas (coeficiente 0.39). O engajamento dos alunos também se refrete na quantidade de tarefas enviadas.

É uma análise simples que traz informações interessantes quando aplicada a dados reais, como neste caso.

Figura 5.1: Correlação dos *logs* dos alunos da disciplina de graduação do curso de Ciência da Computação.



Fonte: Elaborado pelo autor.

5.2 Agrupamento

A análise de agrupamento é separada em duas etapas:

1. Execução do algoritmo *K-Médias++* para realizar o agrupamento, criando dois grupos de alunos;

2. Com os grupos formados pela etapa 1, realizar uma análise exploratória em cada grupo.

5.2.1 Algoritmo *K-Médias++*

Para execução do algoritmo de agrupamento, foi escolhido criar dois agrupamentos. Este número foi escolhido porque o intuito é encontrar alunos desanimados e estudar o seu comportamento. Para isso, separando os dados em dois grupo, conseguimos separar os alunos desanimados dos não desanimado. Também foi testado utilizar 3 grupos, mas os resultados foram iguais aos de dois grupos. O intuito era achar alunos animados, desanimados e com tendência ao desânimo, mas nenhum aluno acabou indo para o terceiro agrupamento. Por isso, foi decidido continuar a análise com dois grupos apenas.

Dos 57 alunos, 7 acabaram indo para primeiro grupo, mostrado pela tabela 5.1, e os outros 50 alunos foram considerados no segundo grupo, alguns deles mostrados pela tabela 5.2.

Todos os alunos do primeiro grupo reprovaram enquanto que todos os alunos do segundo grupo foram aprovados na disciplina. É interessante notar que os alunos do grupo 1 não só reprovaram, como parecem ter desistido ou nem tentaram. Dos sete alunos deste grupo, apenas um enviou as tarefas, e apenas um interagiu no fórum. No segundo grupo de alunos, todos entregaram as tarefas e quase todos interagiram no fórum.

Tabela 5.1: Alunos agrupados no grupo 1.

Tarefas enviadas	Nota tarefas	Posts criados	Posts respondidos	Nota final	Grupo
0	0	0	0	0	1
0	0	0	11	0	1
0	0	0	0	0	1
0	0	0	0	3.33	1
0	0	0	0	0	1
0	0	0	0	0	1
2	0	0	0	1.47	1

Fonte: Elaborado pelo autor.

Tabela 5.2: Alunos agrupados no grupo 2.

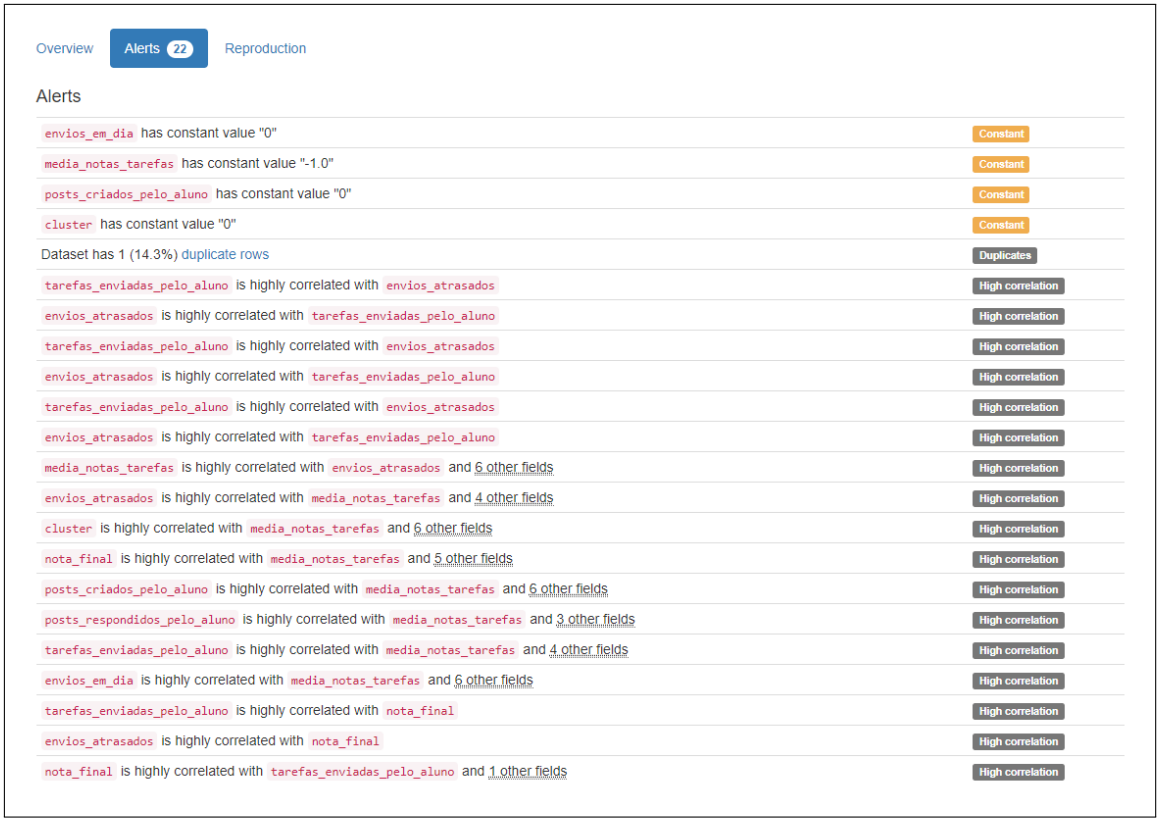
Tarefas enviadas	Nota tarefas	Posts criados	Posts respondidos	Nota final	Grupo
2	5.75	0	3	6.25	2
2	8.65	0	2	7.99	2
2	9.5	4	6	9.08	2
...					

Fonte: Elaborado pelo autor.

5.2.2 Análise exploratória nos grupos

O próximo passo é analisar esses dois grupos de alunos e entender o comportamento de quem desiste da disciplina e de quem não desiste. Para começar essa análise, é executada uma função que traz um conjunto de informações para uma análise exploratória.

Figura 5.2: Análise de variáveis do grupo 1.

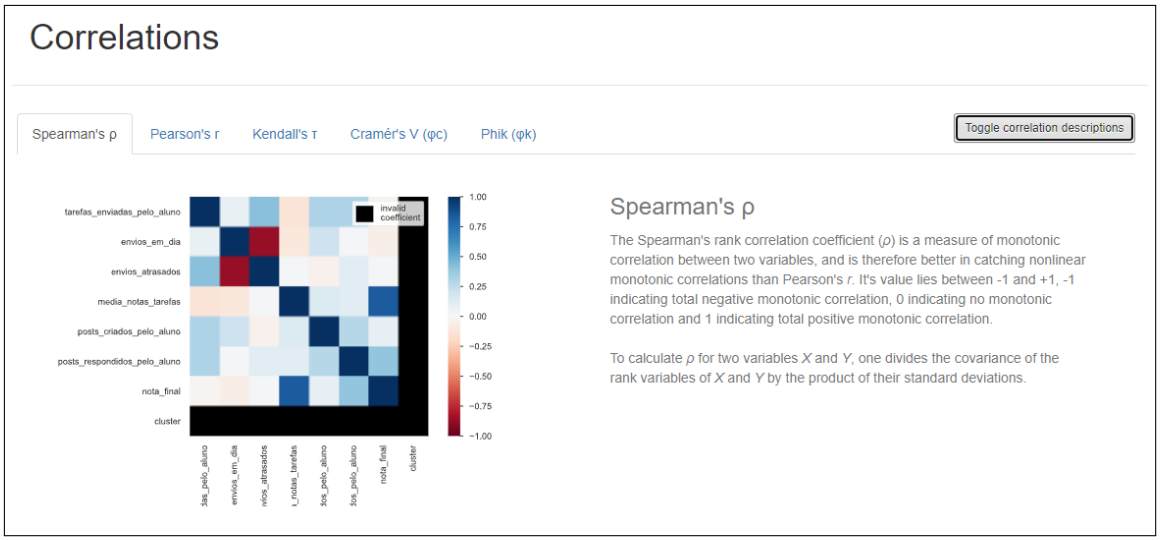


Fonte: Elaborado pelo autor.

Uma dessas informações é em relação aos valores das variáveis. A Figura 5.2 mostra uma parte dessa análise realizada no primeiro grupo de alunos e traz a quantidade

de variáveis constantes (com um valor só para todos os registros), a quantidade de registros duplicados e a correlação entre as variáveis. Neste caso, vê-se que a quantidade de envios de tarefas em dia, as notas das tarefas e a quantidade de *posts* criados pelos alunos é zero em todos esses registros. Ou seja, alunos que reprovam/desistem do curso, tendem a não enviar as tarefas no prazo e nem a criar postagens no fórum. Sobre a nota das tarefas, está constando valor o de -1 porque esse foi o valor padrão que foi setado para tarefas sem nota. A correlação entre as variáveis é mostrada através de vários gráficos, com uma descrição de cada tipo de correlação, como mostrado na figura 5.3.

Figura 5.3: Correlações entre variáveis do grupo 2.

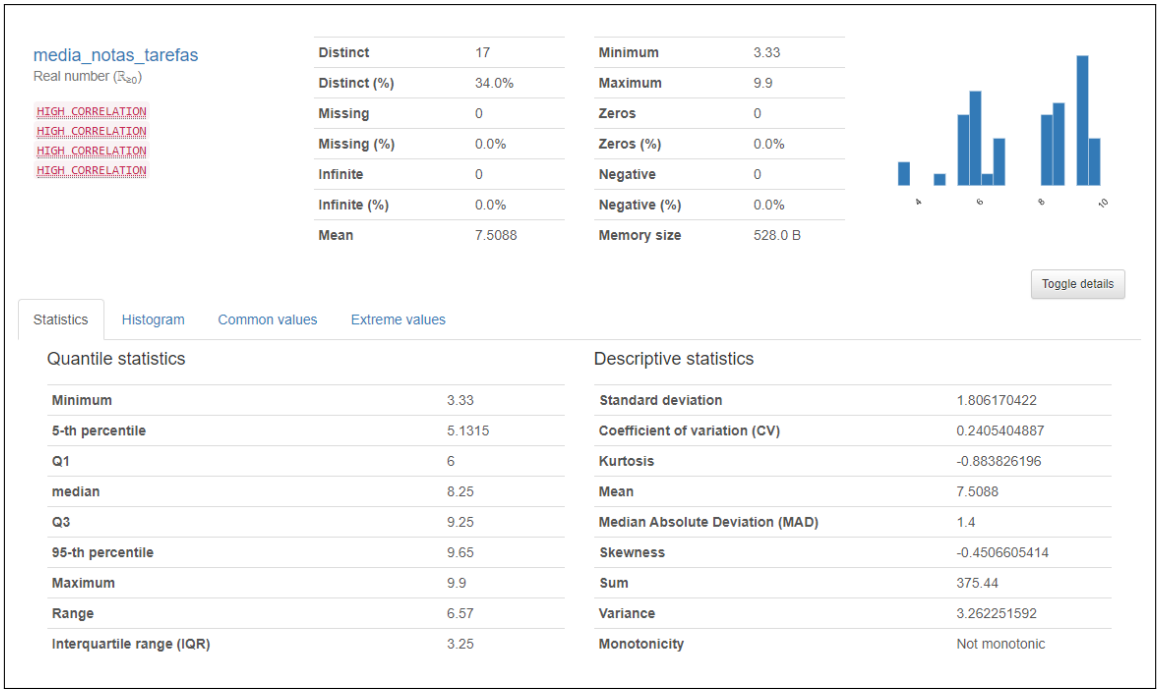


Fonte: Elaborado pelo autor.

Essa análise também traz informações referentes à distribuição de cada variável, seu tipo, quantidade de valores distintos, faltantes, a média, mediana, frequência de valores. A Figura 5.4 exemplifica o resultado obtido para a variável de notas das tarefas, realizada no segundo grupo de alunos. Observa-se que alguns alunos ficaram com uma nota menor do que 6 (média da disciplina) nas tarefas, e, mesmo assim, foram aprovados (estão no grupo 2). Esse fato reforça a hipótese de que os alunos do primeiro grupo desistiram do curso. Os alunos que não desistiram, aprovaram mesmo não tendo ido muito bem nas tarefas.

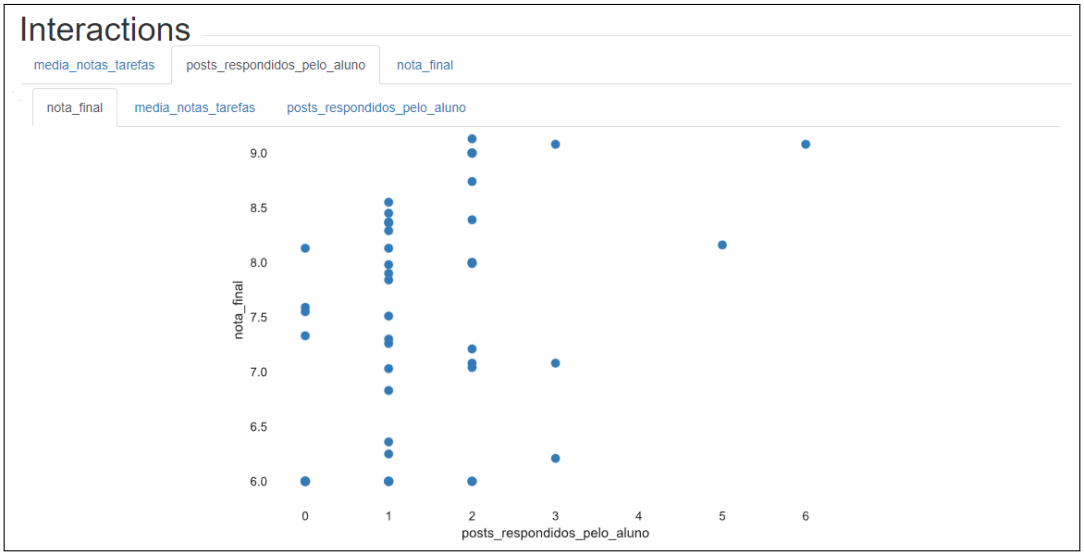
Outra análise interessante é a de interações entre os campos. A biblioteca pandas_profiling disponibiliza também esses dados, tudo no mesmo arquivo 'html'. A figura 5.5 ilustra a interação entre a notas final e a quantidade de *posts* respondidos. Traçando uma regressão linear, nota-se a tendência de alunos que estão mais engajados terem notas maiores, como mostra a figura 5.6. Reforçando a ideia que já vimos na seção anterior, quando calculamos a correlação entre essas variáveis.

Figura 5.4: Análise da variável de notas das tarefas do grupo 2.



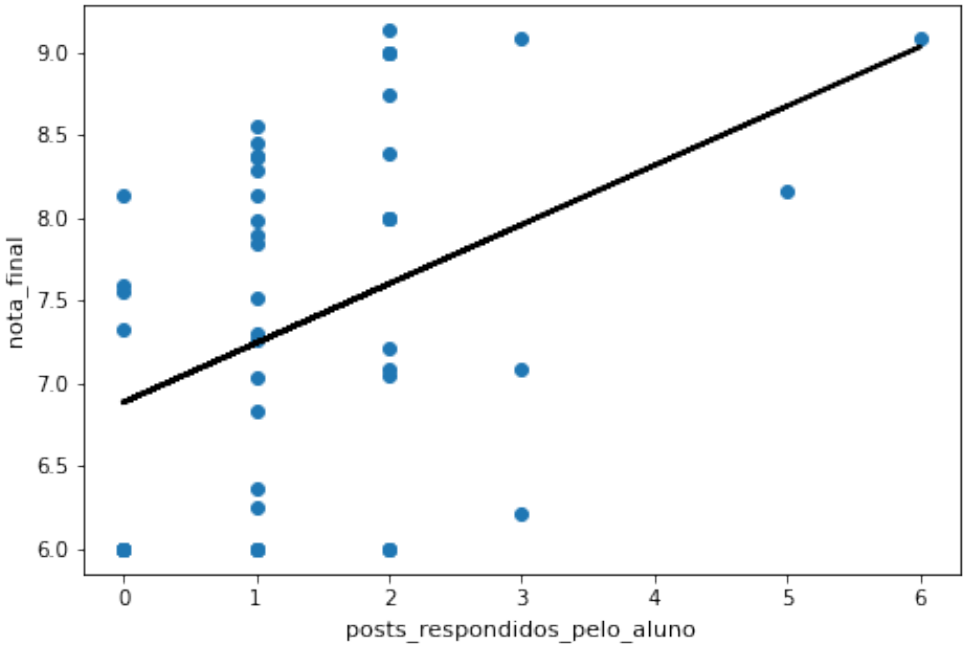
Fonte: Elaborado pelo autor.

Figura 5.5: Interações entre variáveis do grupo 2.



Fonte: Elaborado pelo autor.

Figura 5.6: Tendência entre variáveis do grupo 2.



Fonte: Elaborado pelo autor.

Com esses dados, o professor conseguirá entender um pouco melhor cada grupo de alunos, como se comportaram aqueles alunos que desistiram/reprovaram e como isso se refletiu em suas atividades (alunos que reprovaram não enviaram nenhuma tarefa e não criaram nenhum post como foi visto na figura 5.2) e também como se comportaram aqueles alunos que não desistiram/aprovaram e como isso se refletiu em suas atividades (alunos mais engajados tendem a ter uma nota maior como foi visto na figura 5.6).

5.3 Classificação

Como resultado do algoritmo de classificação, foi criada uma árvore de decisão. O campo tido como alvo foi envios_em_dia (quantidade de tarefas enviada dentro do prazo estipulado). Para este campo, foram observados os valores mostrados pela tabela 5.3.

Tabela 5.3: Observações da classe alvo do algoritmo de classificação para a disciplina.

Classe alvo - envios_em_dia
0 Envios de tarefas em dia
1 Envio de tarefa em dia
2 Envios de tarefas em dia

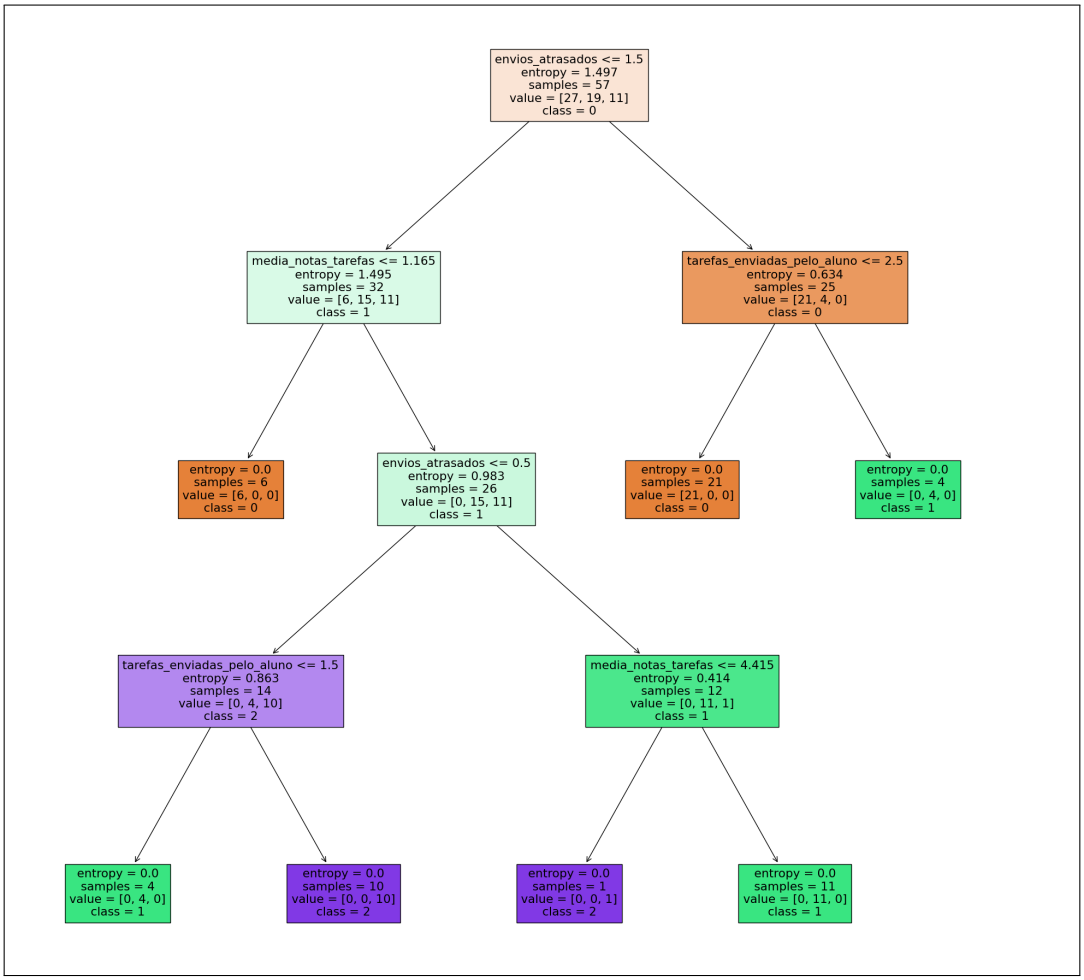
Fonte: Elaborado pelo autor

A árvore criada está ilustrada pela figura 5.7. O nó raiz da árvore (o primeiro, de

cima para baixo) contém uma condição com o atributo mais importante para a análise. Neste caso, esse campo foi envios_atrasados. Como visto anteriormente, o campo de envios atrasados tem uma forte correlação negativa com o envio de tarefas em dia (alvo da nossa classificação). Dado este fato, é compreensível que envios_atrasados seja a raiz da árvore, porque, a partir dele, todos os registros serão separados em dois, uma parte irá para a esquerda da árvore, caso a condição seja verdadeira, e outra parte irá para a direita da árvore, caso contrário.

Todos os alunos que tiveram 0 ou 1 envio atrasado (32 alunos), serão levados para à esquerda e todos que tiveram 2 envios atrasados (25 alunos) irão para à direita. Após essa ramificação, os outros campos considerados foram a média das notas das tarefas (o que também faz sentido, quem envia as tarefas atrasado pode perder algum percentual da nota pelo atraso e quem envia em dia, não perde) e a quantidade total de envios do aluno (alunos que enviam todas as tarefas tendem a enviar mais tarefas em dia do que quem não envia todas as tarefas).

Figura 5.7: Árvore de decisão criada.



Fonte: Elaborado pelo autor

Para a criação da árvore de decisão para esta disciplina foram utilizados apenas 3 campos:

- envios_atrasados;
- media_notas_tarefas;
- tarefas_enviadas_pelo_aluno.

Todos os outros campos como nota final, quantidade de *posts* criados e respondidos pelos alunos, não tiveram um ganho de informação bom o suficiente para serem considerados na análise. Para classificar a quantidade de envios de tarefas em dia do aluno, basta olhar para a árvore e ir seguindo o caminho, conforme as condições dos nós. A tabela 5.4 traz alguns exemplos de alunos fictícios sendo classificados.

Tabela 5.4: Classificação de alunos fictícios usando a árvore de decisão criada.

Envios atrasados	**Tarefas enviadas**	**Nota das tarefas**	**Classificação - Envios em dia**
1	2	1.05	0
2	3	4.5	1
0	2	7.65	2

Fonte: Elaborado pelo autor

6 TRABALHOS RELACIONADOS

Este trabalho é focado na análise dos logs (via mineração de dados) dos alunos em disciplinas ministradas no Moodle (ou com o seu auxílio). A mineração de dados é um campo promissor para a exploração de dados em ambientes educacionais (ZHANG; GHANDOUR; SHESTAK, 2020). Com isso, outros trabalhos foram estudados e se relacionam a este trazendo visões semelhantes além de pontos interessantes a serem somados em futuros projetos.

6.1 Um plugin do tipo report para a identificação do risco de evasão na educação superior a distância que usa técnicas de visualização de dados

Proposto em Brito, Medeiros e Bezerra (2019), o objetivo primário do trabalho é desenvolver um plug-in que apresente alunos em risco de evasão. Para selecionar esses alunos, usa indicadores cognitivos, sociais e comportamentais através de dados do Moodle e mostra esses dados na forma de relatórios e gráficos interativos.

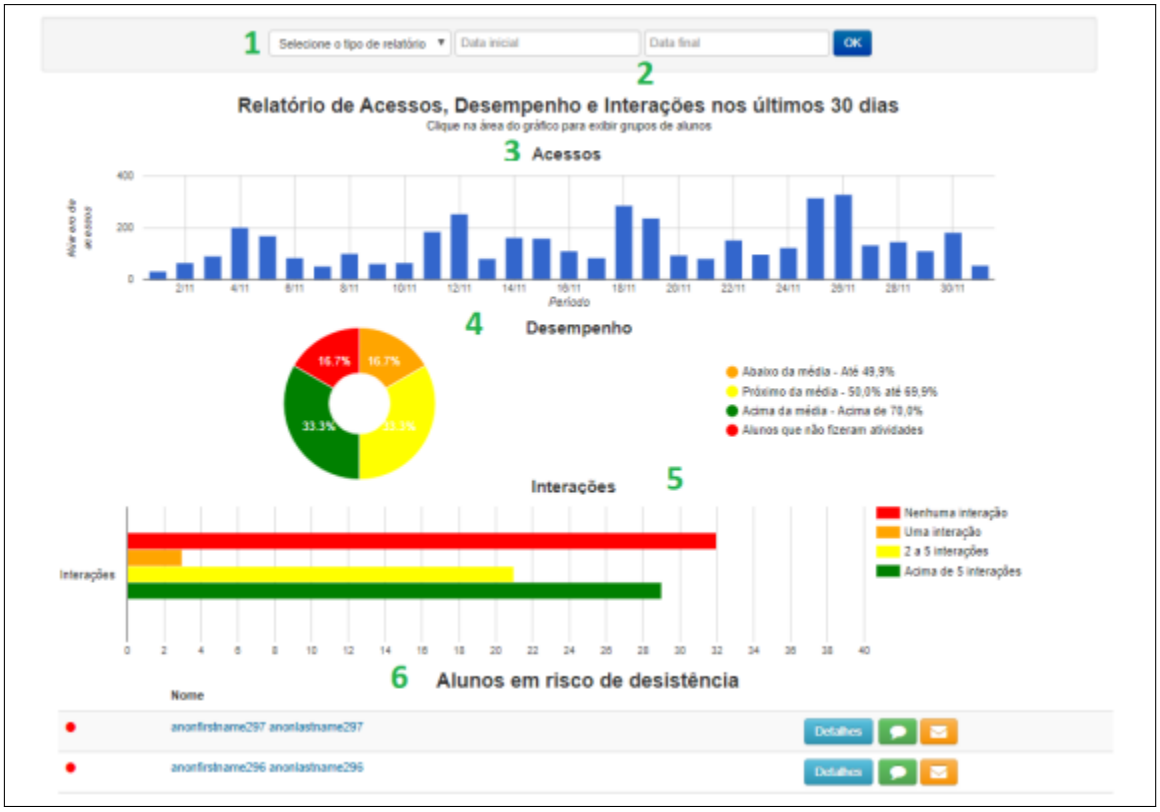
Integrado ao Moodle, traz 3 visões: acessos ao Moodle, desempenho em atividades e interações nos fóruns de discussões. Traz também um relatório com alunos em risco de desistência. Todos os painéis são interativos e possuem filtro de data. A interface do usuário do plug-in está disponível na figura 6.1.

Essa ferramenta é incorporada ao Moodle, trazendo maior facilidade para o professor/tutor. Além disso, possui uma interface amigável e de fácil uso. Porém, não faz uso de ML para gerar análises e não é dinâmico ao trazer os alunos em risco (as configurações de alunos em risco são fixas), diferente deste trabalho que faz análises utilizando algoritmos de ML para cada disciplina, trazendo assim, uma visão mais dinâmica dos alunos.

6.2 Análise do comportamento e sucesso do aluno com base nos logs do Moodle

O trabalho proposto por Kadoić e Oreški (2018) consiste em analisar os logs de uma disciplina real com o intuito de detectar frequências e encontrar padrões nas escolhas das atividades dos alunos. O objetivo deste trabalho é encontrar o impacto de certas atividades na nota final.

Figura 6.1: Interface do usuário do Plug-in



Fonte: Elaborado por Brito, Medeiros e Bezerra (2019)

O estudo encontra uma correlação positiva e estatisticamente significativa entre as notas dos alunos e a abertura de arquivos da disciplina. Também é mostrado que alunos que interagem nos fóruns tendem a abrir mais arquivos. A tabela 6.1 mostra essa correlação entre as variáveis.

Tabela 6.1: Correlação entre variáveis.

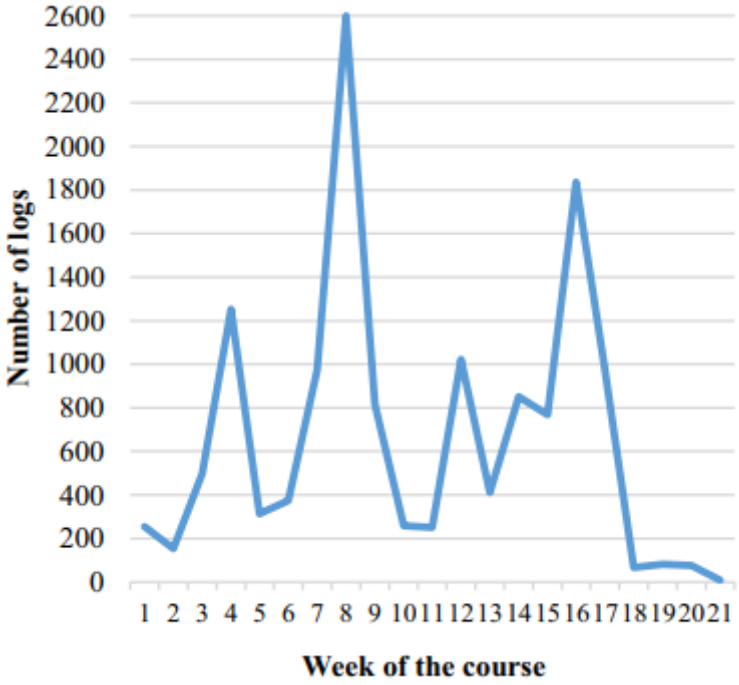
Variável	Nota	Arquivos acessados	Fóruns acessados	Links acessados	Tarefas enviadas
Nota	-	0.35	0.18	-0.01	0.08
Arquivos acessados	0.35	-	0.41	0.05	0.19
Fóruns acessados	0.18	0.41	-	0.13	0.38
Links acessados	-0.01	0.05	0.13	-	0.09
Tarefas enviadas	0.08	0.19	0.38	0.09	-

Fonte: Elaborado por Kadoić e Oreški (2018)

O estudo também mostra que, para a específica disciplina estudada, os alunos tendem a serem mais ativos no Moodle em semanas de provas e perto de datas de entregas de atividades. A figura 6.2 mostra o número de logs e gerados por semana. A 8ª e a 16ª

semanas são as semanas em que as provas ocorreram. Pode-se observar um aumento muito grande no número de logs gerados nessas duas semanas.

Figura 6.2: Distribuição dos logs pelas semanas.



Fonte: Elaborado por Kadoić e Oreški (2018)

É interessante notar também que o número de logs gerados na semana da primeira prova é muito maior do que na da segunda prova. Talvez alunos que desistiram do curso tenham desistido após a primeira prova. Um ponto interessante de estudo, não explorado pelo trabalho, seria analisar o comportamento dos alunos antes da primeira prova e após ela, para tentar entender o comportamento dos alunos desistentes.

Este estudo é muito focado na nota final do aluno. Alunos que desistem tendem a reprovar, mas conseguimos analisar o comportamento deles antes da reprovação, como acabamos de ver. O meu trabalho não tem este enfoque na nota final do aluno.

6.3 Usando Learning Analytics para prever o desempenho dos alunos Moodle

Em Zhang, Ghandour e Shestak (2020), o trabalho feito tem como objetivo analisar os dados obtidos via Moodle, melhorar o processo de aprendizado e reduzir o número de alunos com baixo desempenho. Além disso, o estudo é focado em identificar até que ponto os dados individuais obtidos nos logs de atividades são parâmetros confiáveis do sucesso acadêmico dos alunos. Para isso, aplica uma série de análises nos logs dos alunos,

focando nos seguintes dados:

- Quantidade de *quizzes* aprovados ou reprovados;
- Quantidade de mensagens lidas ou respondidas no forum;
- Tempo passado em tarefas, *quizzes* e forum;
- Nota final do aluno

Os resultados são parecidos com o trabalho visto anteriormente (Kadoić e Oreški (2018)). Foi encontrada uma correlação positiva e estatisticamente significativa entre as notas e o número de aberturas de arquivos da disciplina. Outro resultado parecido foi a maior quantidade de acessos ao Moodle nas semanas de provas. O mesmo padrão é encontrado no presente estudo: maior quantidade de logs gerados na semana da primeira prova do que na da segunda.

As análises feitas se assemelham muito às análises do artigo anterior. Com isso, as limitações acabam sendo as mesmas: muito foco na nota final do aluno.

6.4 Análise comparativa

Para comparar os trabalhos de forma mais visual, foi elaborada a tabela 6.2. Nela estão contidas as funcionalidades de cada trabalho.

Tabela 6.2: Funcionalidades entre os trabalhos estudados e o do autor.

Funcionalidade	Brito, Medeiros e Bezzera	Kadoić e Oreški	Zhang, Ghandour e Shestak	Do Autor
Plug-in integrado ao Moodle	X			
Gráficos interativos	X			X
Filtro por data	X			
Análise de correlação		X	X	X
Análise de logs por semana		X	X	
Análise de logs por gênero		X	X	
Análise por árvore de decisão				X
Análise de agrupamento				X

Fonte: Elaborado pelo autor

7 CONCLUSÕES

O intuito do projeto era disponibilizar algumas ferramentas para que o professor pudesse obter informações sobre seus alunos a fim de entender o seu comportamento. O objetivo principal era facilitar a vida do professor a respeito de insumos para o desenvolvimento de suas futuras disciplinas. Baseado nos resultados obtidos no capítulo 5, é possível afirmar que o objetivo foi atingido. O projeto disponibiliza tais ferramentas através da mineração de dados dos *logs* dos alunos de cursos no Moodle.

Há certas limitações no projeto, como a quantidade de configurações prévias a serem feitas para execução do *script*. Este projeto surgiu com o intuito de ser de fácil uso, mas, por necessitar de muitas configurações prévias, acabou não atingindo este objetivo completamente. Esse é justamente um ponto que pode ser melhorado no futuro, facilitar a configuração para quem não é da área de informática. Para isso, ao invés de ter que configurar um banco de dados, que tudo seja feito via *script*: a extração e consolidação dos dados também. Tirando, assim, esta etapa da configuração prévia.

Como não se conseguiu testar o projeto via Moodle institucional da UFRGS, uma ideia que surgiu durante o projeto foi testar via Moodle do Instituto de Informática da UFRGS. Esta ideia, porém, não saiu do papel, tornando este outro ponto como ajuste futuro, testar a execução do script no Moodle do INF.

Há outros *logs* que podem ser utilizados na análise, como por exemplo a quantidade de acessos no Moodle, nota dos fóruns. Gráficos de dispersão junto com alguma regressão linear pode ser utilizado junto com a informação de correlação entre as variáveis. A ampliação da análise para conter mais *logs* e gráficos de dispersão ficam como trabalho futuro.

Outro ponto interessante de trabalho futuro é realizar as análises deste trabalho em uma disciplina em andamento para ir entendendo o comportamento dos alunos durante o curso da disciplina. Verificar se o que foi encontrado no trabalho de Kadoić e Oreški (2018), referente a análise semanal dos alunos, pode ser observado em outras disciplinas.

Em suma, este trabalho propõe uma ferramenta que utiliza técnicas de aprendizado de máquina para entender o ânimo dos alunos, achar semelhanças entre eles e correlacionar as suas ações durante a disciplina com as suas notas, trazendo novas percepções sobre as ações do dia a dia do aluno e como isso se reflete no seu desempenho. Disponibilizando essas informações através de gráficos e visões intuitivas e de fácil compreensão.

REFERÊNCIAS

Anuja Priyama, Abhijeeta, Rahul Guptaa, Anju Ratheeb, Saurabh Srivastavab. Comparative Analysis of Decision Tree Classification Algorithms. **International Journal of Current Engineering and Technology**, v. 3, n. 2, p. 334–337, 6 2013. ISSN 2277 - 4106. Available from Internet: <http://inpressco.com/comparative-analysis-of-decision-tree-classification-algorithms/>.

ARTHUR, D.; VASSILVITSKII, S. **k-means++: The Advantages of Careful Seeding**. [S.l.], 2006. Available from Internet: <http://ilpubs.stanford.edu:8090/778/>.

BRITO, M.; MEDEIROS, F.; BEZERRA, E. A report-type plugin to indicate dropout risk in the virtual learning environment moodle. In: **2019 IEEE 19th International Conference on Advanced Learning Technologies (ICALT)**. [S.l.: s.n.], 2019. v. 2161-377X, p. 127–128.

Carl Kingsford, Steven L Salzberg. What are decision trees? **Nature Biotechnology**, v. 26, n. 9, p. 1011–1013, 9 2008. ISSN 1546-1696. Available from Internet: <https://doi.org/10.1038/nbt0908-1011>.

CARLEO, G. et al. Machine learning and the physical sciences. **Reviews of Modern Physics**, American Physical Society (APS), v. 91, n. 4, dec 2019. Available from Internet: <https://doi.org/10.1103%2Frevmodphys.91.045002>.

Dongkuan Xu, Yingjie Tian. A Comprehensive Survey of Clustering Algorithms. **Annals of Data Science**, v. 2, n. 2, p. 165–193, 6 2015. ISSN 2198-5804. Available from Internet: <https://doi.org/10.1007/s40745-015-0040-1>.

Haldun Akoglu. User's guide to correlation coefficients. **Turkish Journal of Emergency Medicine**, v. 18, n. 3, p. 91–93, 9 2018. ISSN 2452-2473. Available from Internet: <https://doi.org/10.1016/j.tjem.2018.08.001>.

Jan Hauke, Tomasz Kossowski. Comparison of values of Pearson's and Spearman's correlation coefficient on the same sets of data. **Quaestiones Geographicae**, v. 30, n. 2, p. 87–93, 4 2011. ISSN 0137-477X. Available from Internet: <https://doi.org/10.2478/v10117-011-0021-1>.

KADOIć, N.; OREšKI, D. Analysis of student behavior and success based on logs in moodle. In: **2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)**. [S.l.: s.n.], 2018. p. 0654–0659.

MALIK, J. S.; GOYAL, P.; SHARMA, M. A. K. A comprehensive approach towards data preprocessing techniques & association rules. In: . [S.l.: s.n.], 2010.

NOWOZIN, S. **Improved Information Gain Estimates for Decision Tree Induction**. arXiv, 2012. Available from Internet: <https://arxiv.org/abs/1206.4620>.

Oyelade, O. J., Oladipupo, O. O., Obagbuwa, I. C. Application of k Means Clustering algorithm for prediction of Students Academic Performance. **International Journal of Computer Science & Information Security**, v. 7, n. 1, p. 292–295, 1 2010. ISSN 1947-5500. Available from Internet: <https://arxiv.org/abs/1002.2425>.

Patrick Schober, Christa Boer, Lothar A Schwarte. Correlation Coefficients: Appropriate Use and Interpretation. **Anesthesia & Analgesia**, v. 126, n. 5, p. 1763–1768, 5 2018. ISSN 1526-7598. Available from Internet: <https://doi.org/10.1213/ane.0000000000002864>.

Pratap Chandra Sen, Mahimarnab Hajra, Mitadru Ghosh. Supervised Classification Algorithms in Machine Learning: A Survey and Review. In: MANDAL, J. K.; BHATTACHARYA, D. (Ed.). **Emerging Technology in Modelling and Graphics**. Singapore: Springer Singapore, 2020. p. 99–111. ISBN 978-981-13-7403-6.

PRUENGKARN, R. et al. A review of data mining techniques and applications. **Journal of Advanced Computational Intelligence and Intelligent Informatics**, v. 21, n. 1, p. 31–48, 2017.

Shaker H. Ali El-Sappagh, Abdeltawab M. Ahmed Hendawi, Ali Hamed El Bastawissy. A proposed model for data warehouse ETL processes. **Journal of King Saud University – Computer and Information Sciences**, v. 23, n. 2, p. 91–104, 7 2011. ISSN 1319-1578. Available from Internet: <https://doi.org/10.1016/j.jksuci.2011.05.005>.

T. Soni Madhulatha. AN OVERVIEW ON CLUSTERING METHODS . **IOSR Journal of Engineering**, v. 2, n. 4, p. 719–725, 4 2012. ISSN 2250-3021. Available from Internet: <https://doi.org/10.48550/arXiv.1205.1117>.

ZHANG, Y.; GHANDOUR, A.; SHESTAK, V. Using learning analytics to predict students performance in moodle lms. **International Journal of Emerging Technologies in Learning (iJET)**, v. 15, n. 20, p. pp. 102–115, Oct. 2020. Available from Internet: <https://online-journals.org/index.php/i-jet/article/view/15915>.