UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO DE CIÊNCIAS, TECNOLOGIAS E SAÚDE DO CAMPUS ARARANGUÁ
CURSO DE GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

Caio Blumer Barrozo

Módulo Preditivo para Classificação de Estudantes em Risco de Reprovação

Araranguá
2022

Caio Blumer Barrozo

Módulo Preditivo para Classificação de Estudantes em Risco de Reprovação

Trabalho de Conclusão de Curso do Curso de Graduação em Engenharia de Computação do Centro de Ciências, Tecnologias e Saúde do Campus Araranguá da Universidade Federal de Santa Catarina para a obtenção do título de Bacharel em Engenharia de Computação.
Orientador: Prof. Cristian Cechinel, Dr.

Araranguá
2022

Caio Blumer Barrozo

Módulo Preditivo para Classificação de Estudantes em Risco de Reprovação

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do Título de Bacharel em Engenharia de Computação e aprovado em sua forma final pelo Curso de Graduação em Engenharia de Computação.

Araranguá, 23 de Março de 2022.

Documento assinado digitalmente
ANALUCIA SCHIAFFINO MORALES
Data: 25/03/2022 16:37:34-0300
CPF: 622.256.420-87
Verifique as assinaturas em https://v.ufsc.br

Prof[a]. Analucia Schiaffino Morales, Dra.
Coordenadora do Curso

Banca Examinadora:

Documento assinado digitalmente
Cristian Cechinel
Data: 25/03/2022 14:54:41-0300
CPF: 454.628.449-72
Verifique as assinaturas em https://v.ufsc.br

Prof. Cristian Cechinel, Dr.
Orientador

Documento assinado digitalmente
Anderson Luiz Fernandes Perez
Data: 25/03/2022 15:40:13-0300
CPF: 020.803.459-58
Verifique as assinaturas em https://v.ufsc.br

Prof. Anderson Luiz Fernandes Perez, Dr.
Avaliador
Universidade Federal de Santa Catarina

Documento assinado digitalmente
Alexandre Leopoldo Goncalves
Data: 25/03/2022 15:51:51-0300
CPF: 786.740.349-34
Verifique as assinaturas em https://v.ufsc.br

Prof. Alexandre Leopoldo Gonçalves, Dr.
Avaliador
Universidade Federal de Santa Catarina

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

Barrozo, Caio
Módulo Preditivo para Classificação de Estudantes em Risco de Reprovação / Caio Barrozo ; orientador, Cristian Cechinel, 2022.
49 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Campus Araranguá, Graduação em Engenharia de Computação, Araranguá, 2022.

Inclui referências.

1. Engenharia de Computação. 2. Moodle. 3. Plugin. 4. Modelo Preditivo. 5. Predição. I. Cechinel, Cristian . II. Universidade Federal de Santa Catarina. Graduação em Engenharia de Computação. III. Título.

AGRADECIMENTOS

O desenvolvimento e conclusão deste trabalho não seria possível sem a ajuda dos professores que me guiaram até aqui e da minha família por não me deixar desistir.

RESUMO

As taxas reprovação e evasão escolar são problemas que afligem cursos dentro das universidades causando um impacto negativo na jornada acadêmica do aluno, como perda de motivação e oportunidades. Identificar os estudantes que estão em risco de reprovação ou evasão a tempo de tomar uma medida que possa auxiliar a normalização do aluno na disciplina pode ser um desafio. Focando nisto, o trabalho propõem o desenvolvimento de um de um método, inserido dentro do *plugin* Moodle Analytics Dashboard, para classificação de estudantes em risco. O modelo preditivo foi criado com base em pesquisas publicadas que apontam a possibilidade de criar modelos tendo como base apenas as interações semanais dos alunos. O método foi avaliado em duas disciplinas e as predições tiveram uma acurácia média nas cinco primeiras semanas de 56.2% crescendo para 76.5% nas ultimas três semanas. A causa mais provável da baixa qualidade das predições é dada pelo problema de portabilidade de modelos visto que o treinamento foi realizado com disciplinas não similares. No geral o trabalho demonstra como o aprendizado de máquina pode auxiliar os alunos e traz ideias para trabalhos futuros.

Palavras-chave: Moodle. Plugin. Modelo Preditivo. Predição. MAD.

ABSTRACT

Failure and dropout rates are problems that afflict courses inside universities, causing a negative impact on the academic journey of the student, such as loss of motivation and opportunities. Identifying those students that are at risk of failure or dropping out with enough time to take actions that will help normalize the student's situation can be a challenge. With this in mind, this work proposes the development of a method, inside the Moodle Analytics Dashboard plugin, to classify students at risk. The predictive model was created with previously published studies that pointed to the possibility to create models with just the amount of weekly interactions of the student. The method was analized in two courses and the predictions achieved a medium accuracy in the first 5 weeks of 56.2% rising to 76.5% in the last three weeks. The most probable cause of the low quality of the predictions is due to a problem with model portability since it was trained with an unrelated course. In general, the work shows how machine learning can help students and brings new ideas for future works.

Keywords: : Moodle. Plugin. Modelo Preditivo. Predição. MAD.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

AdaBoost	*Adaptive Boosting*
EDM	*Educational Data Mining*
IBM	*International Business Machines*
KDD	*Knowledge Discovery in Databases*
LA	Learning Analytics
MAD	*Moodle Analytic Dashboard*
ROC	Curva Característica de Operação do Receptor
UFSC	Universidade Federal de Santa Catarina

SUMÁRIO

1 INTRODUÇÃO

O número de estudantes de engenharia que não conseguem passar do primeiro ano atinge incríveis 40% e entre aqueles que conseguem, 30% reprovam em disciplinas fundamentais (BOYLAN-ASHRAF; HAUGHERY, 2018). Um artigo analisou o resultado de 161 cursos de graduação em ciências da computação em 15 países, contemplando 51 instituições e encontrou uma média universal de aprovação quase idêntica de 67,7% (WATSON; LI, 2014).

A identificação precoce de estudantes em risco de reprovação ou evasão permite que uma ajuda ou intervenção seja fornecida para tentar normalizar a situação do acadêmico. Uma revisão sistemática da literatura encontrou uma redução, em média, de 45,6% para 32,6% quando há intervenção (RAMOS *et al.*, 2015).

Tendo em vista a existência de elevadas taxas de reprovação ou evasão em alguns aspectos da vida acadêmica dos estudantes de graduação, surge então a oportunidade de desenvolver um método que possa auxiliar na identificação precoce desses acadêmicos em risco. O trabalho adota a mesma abordagem de Macarini *et al.* (2019) que demonstrou a possibilidade de utilizar somente a quantidade de interações dos alunos para a criação de um modelo preditivo.

As interações são retiradas do Moodle, um ambiente virtual de aprendizagem, através do *plugin* Moodle Analytics Dashboard (MAD). O MAD é uma extensão, que vem sido desenvolvida para a plataforma Moodle na Universidade Federal de Santa Catarina (UFSC) auxiliando na visualização de dados e gestão estudantil.

Os dados gerados pelo Moodle são armazenados no banco de dados em uma tabela de *logs*, porém eles existem em uma forma difícil de extrair qualquer tipo de informação. Ferramentas que permitem uma visualização gráfica desses dados fornecem uma boa visão geral, mas podemos ir além e utiliza-los para criar um modelo preditivo.

1.1 OBJETIVOS

No desenvolvimento do trabalho foram dispostos objetivos que precisam ser satisfeitos para atingir a conclusão.

1.1.1 Objetivo geral

Este trabalho desenvolve um módulo preditivo expandindo as funcionalidades do MAD, utilizando técnicas de ciência de dados e aprendizado de máquina para a classificação de estudantes em risco de reprovação ou evasão.

1.1.2 Objetivos específicos

Para atingir a concretização do trabalho existe as seguintes especificidades:

- Identificar as principais características do Moodle que permitam realizar a predição de estudantes em risco de evasão
- Projetar comunicação entre o *plugin* MAD e o algoritmo de predição
- Garantir fácil usabilidade da ferramenta de predição

1.2 ORGANIZAÇÃO DO TEXTO

Os próximos tópicos foram estruturados da seguinte forma: no segundo capítulo será mencionado trabalhos que possuem semelhança ou pertencem a mesma área, discutindo as diferenças e escolhas de implementação.

O terceiro capítulo traz a base teórica para o acompanhamento deste trabalho, desde os passos iniciais de mineração de dados até os algoritmos de treinamento.

O quarto capítulo descreve o contexto do projeto e onde ele se encaixa.

No quinto capítulo se encontra a descrição sobre o desenvolvimento do projeto, tecnologias utilizadas e funcionalidades.

Por fim o capítulo seis apresenta os resultados e traz os trabalhos futuros para o aperfeiçoamento do projeto.

1.3 ESTADO DA ARTE

Para servir de fundação ao projeto, realizou-se uma revisão sistemática da literatura, pesquisando nas principais bases de dados científicas o que foi e o que está sendo discutido sobre o assunto, bem como os resultados encontrados pelos principais autores, pesquisadores que já criaram e treinaram modelos, utilizando os mais variados dados. Em um estudo inicial, as bases que se mostraram mais promissoras foram Research Gate e SciHub. Realizou-se então uma pesquisa pelas palavras-chave "*Prediction model AND students*". Realizou-se também uma pesquisa pelo termo "*Predicting student success in blended learning*", que retornou mais resultados relacionados.

1.3.1 Mineração de dados educacionais

Uma pesquisa sobre a metodologia de implementação para predição de estudantes em risco foi feita por Koutcheme *et al.* (2022). O artigo responde três questões, a primeira é que a inclusão dos dados de estudantes desistentes da semana anterior ajuda o modelo a identificar a tendência que estudantes inativos tendem a ficar inativos melhorando o modelo. A segunda questão é qual seria a melhor *feature* nos casos de inclusão e exclusão dos dados de estudantes desistentes, porém não houve uma variação substancial das *features* entre as abordagens. A terceira questão se refere a importância do contexto da disciplina e foi descoberta a necessidade de criar modelos preditivos específicos para o curso.

Romero e Ventura (2010) definem Mineração de dados educacionais, do inglês *Educational Data Mining* (EDM), como sendo uma aplicação de mineração de dados em dados educacionais com o objetivo de resolver problemas. O processo de EDM converte dados não processados em informação útil que pode ter um grande impacto nas praticas educacionais. O trabalho ainda descreve a previsão de performance estudantil como sendo um dos mais antigos e populares usos de mineração de dados em educação e que houve um crescimento exponencial no número de publicações.

Classificadores são geralmente validados utilizando validação cruzada onde parte do *dataset* é repetidamente e sistematicamente separado e utilizado para testar a qualidade do modelo (BAKER; INVENTADO, 2014). O autor também aponta que a acurácia, embora muito popular em outros campos, não é sensível a taxa-base e só deve ser usado em casos onde essas taxas são reportadas.

Uma pesquisa sobre 240 trabalhos de mineração de dados educacionais encontrou que classificação é a abordagem mais adotada com 42.15% de uso, seguido de clusterização com 26.86% e regressão com 15.29% (PEÑA-AYALA, 2014).

Após uma revisão de softwares e extensões Dondorf *et al.* (2019) definiu algumas características que um software de Learning Analytics (LA) deve possuir:

- O sistema deve ser fácil de usar.

- Visualização de dados importante devem ter fácil acesso.
- Visualizar como os resultados se alteram ao longo da semana.
- Comparar estatísticas com disciplinas anteriores.
- Apresentar uma relação entre notas e comportamento do aluno.

Zapparolli e Stiubiener (2017) juntou uma ferramenta de BI (*Business Intelligence*) com análise de dados educacionais para criar uma ferramenta que fornece visualizações mais genéricas. É possível filtrar pelo professor e visualizar todas as suas disciplinas e alunos, filtrar por cursos específicos e informações sobre acesso ao fórum.

1.3.2 *Dataset* e algoritmos

Adnan *et al.* (2021) realizou uma comparação da performance de diversos algoritmos levando em consideração a acurácia, precisão, suporte e *f-score*. O melhor algoritmo é selecionado e utilizado para treinar modelos ao longo da duração do curso. O algoritmo *random forest* apresentou o melhor comportamento com todas as métricas atingindo 80% próximo da metade do curso.

Cortez e Silva (2008), realizaram um estudo sobre a importância relativa de de 29 variáveis na construção de um modelo com objetivo de prever a reprovação de alunos no contexto de duas classes de ensino fundamental de duas escolas portuguesas. Descobriu-se que reprovações passadas possuíram o maior impacto com 21.8%, faltas apresentaram 9.4% e logo em seguida suporte extra-classe com 7.0%. Outros dados que também se mostraram relevantes foram consumo de álcool, desejo de cursar ensino superior, qualidade das relações familiares, profissão da mãe e idade do aluno.

Os autores também realizaram uma comparação entre diferentes algoritmos como Decision Tree (DT), Random Forest (RF), Neural Network (NN) and Support Vector Machine (SVM) e encontraram que os algoritmos DT e NN possuem uma precisão de 93% e 91% respectivamente.

Macarini *et al.* (2019) buscavam demonstrar se apenas a quantidade de interações dos alunos com o sistema de EAD permitiria gerar dados suficientes para prever sua aprovação ou reprovação no fim do semestre. Com o uso de algumas variáveis derivadas foi obtida uma melhora no modelo. A base de dados contendo quantidade, média e mediana foi a que apresentou melhores resultados, logo seguida pela base de dados com quantidade, média, mediana, semanas zeradas, fator de comprometimento e média das diferenças. O algoritmo que melhor se adequou foi o *Adaptive Boosting* (AdaBoost).

Kotsiantis, Pierrakeas e Pintelas (2004) geraram modelos utilizando inicialmente apenas dados demográficos dos estudantes. Os modelos mostraram uma precisão na faixa de 62% nas predições iniciais, excedendo 82% antes das avaliações finais, onde dados históricos do semestre foram utilizados.

Estudos mais recentes apontaram que o algoritmo Multi Layer Perceptron como promissor atingindo 72% de precisão logo no início do modelo e mostrando que a ocupação dos pais é um fator importante em alguns contextos. (MADNAIK, 2020);

É de extrema importância ressaltar que o levantamento de diversas variáveis como situações socioeconômicas dos estudantes, embora favorável para a criação do modelo preditivo nem sempre é possível, já que a privacidade deve ser levada a sério e trabalhar com dados pessoais pode causar problemas. Logo se existe uma solução elegante que gere resultados satisfatórios e ao mesmo tempo não trata com dados sensíveis é natural que ela seja escolhida, dessa maneira o modelo preditivo usado pelo módulo de predição foi criado com base em (MACARINI *et al.*, 2019), utilizando a quantidade de interações com o *moodle* em conjunto com variáveis derivadas. Quanto ao algorítimo, Não foi encontrada uma relação direta de qual algoritmo é superior, mas é possível inferir que algoritmos simples apresentam os melhores resultados e que os algoritmos vão estar atrelado as variáveis disponíveis. Como o AdaBoost provou-se superior quando considerando as iterações em conjunto de dados derivados, ele foi escolhido.

1.3.3 *Dashboards*

Um trabalho realizado por Santos *et al.* (2019) analisou cinco *dashboards* diferentes desenvolvidos para o moodle, eles são: Analytics graphs, GISMO, Completion Progress, Heatmap, Forum Graph. Todos eles são extensões que fornecem a funcionalidade de *dashboard*, uma tela onde dados são representados graficamente tornando mais fácil o entendimento.

- Analytics Graphs: Esta extensão fornece gráficos para visualização do perfil de cada estudante com o intuito de melhorar a qualidade de ensino e facilitar a comunicação entre docente e estudante. É proporcionado um gráfico com a distribuição de notas das avaliações, informações sobre quais alunos realizaram uma determinada tarefa junto com a possibilidade de entrar em contato e quantidade de interações dos estudantes.

- GISMO: Esta ferramenta possui um foco no rastreamento e monitoramento do estudante durante o decorrer da disciplina, um painel com informações sobre o desempenho em cada atividade e a quantidade de interações em recursos não avaliativos permite identificar para quais estudantes pode ser tomada uma ação interventiva.

- Completion Progress: Este *plugin* fornece uma visualização por aluno sobre o status das atividades, assim como quais possuem o menor índice de entrega ou menores notas. É possível identificar quais atividades possuem o pior desempenho e realizar ajustes necessários.

- Heatmap: Fornece a opção de visualizar quais tarefas requerem atenção do professor através da coloração atribuída.
- Forum Graph: Esta extensão gera uma visualização na forma de grafo dirigido das interações dos alunos e professores. O tamanho do nodo representa a quantidade de mensagens trocadas pelos usuários.

Outra ferramenta para visualização de dados no Moodle, porém já depreciado, é a primeira versão do MAD. Originalmente uma extensão criada para o navegador Google Chrome ela fornecia diversos tipos de visualizações como gráficos de dispersão, linhas, barras, pizza e mapa de calor e entre as métricas possuía número de participantes, número de páginas únicas, páginas únicas visitadas, atividades únicas, tempo médio de sessão, número de sessões e taxa de rejeição (DISTANTE *et al.*, 2020a).

2 FUNDAMENTAÇÃO TEÓRICA E TECNOLÓGICA

Neste capítulo é apresentado o conteúdo teórico do contexto onde o trabalho esta inserido como mineração de dados, classificadores, pré-processamento entre outros. Encerrando com a visão geral das tecnologias que foram utilizadas para o desenvolvimento do trabalho.

2.1 FUNDAMENTAÇÃO TEÓRICA

Como o objetivo é criar um módulo preditivo para a extensão MAD o trabalho toma como base teórica pesquisas já publicadas por outros acadêmicos, isso levou com que ao longo do trabalho algumas decisões fossem tomadas com base nesses estudos.

Primeiramente será visto uma apresentação sobre a mineração de dados já que foi o passo inicial no desenvolvimento do projeto, sem dados seria impossível construir um modelo preditivo ou até mesmo realizar uma predição. A etapa de predição ou classificação dos estudantes também sera abordada, seguida do pré-processamento. Por fim será dada vista aos algoritmos existentes com foco no escolhido, o AdaBoost.

2.1.1 *Moodle Analytic Dashboard* (MAD)

O Moodle possui um registro de acessos que contém as interações dos usuários nas disciplinas. Esses registros ficam armazenados na base de dados do moodle em uma tabela de *logs*, porém não existe uma forma viável de visualizar esses dados ou extrair facilmente qualquer tipo de informação útil.

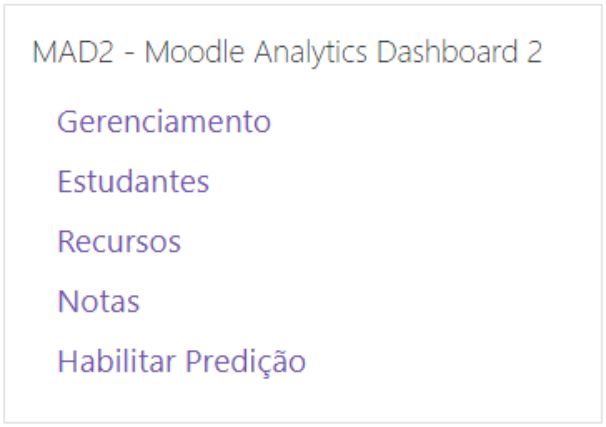
O *Moodle Analytic Dashboard* (MAD) é um *plugin* desenvolvido para o moodle que permite a visualização gráfica dos registros de acessos e outros aspectos das disciplinas. Além disso o professor é capaz de acompanhar de perto a trajetória acadêmica dos alunos e entender quais recursos educacionais estão sendo utilizados (FREITAS DOS SANTOS, 2021).

Após a instalação o MAD pode ser encontrado no menu lateral da página (Figura 1) onde cinco opções estão disponíveis.

2.1.1.1 Gerenciamento

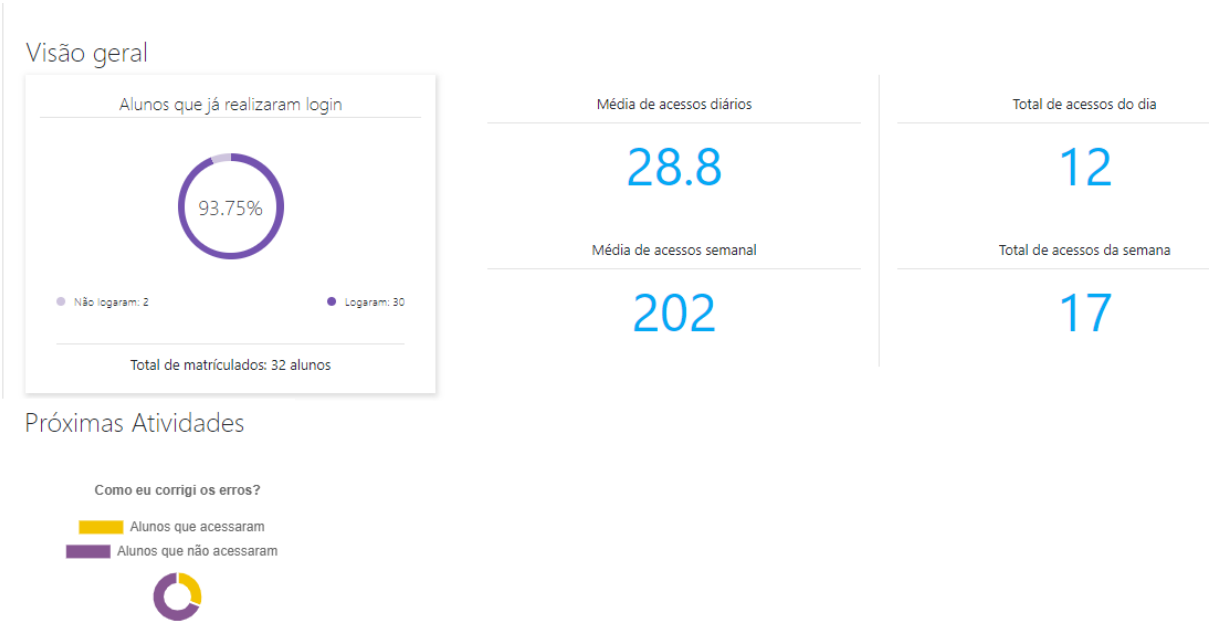
A tela de gerenciamento fornece a visualização sobre os dados de acesso dos alunos, nela é encontrada a informação de quantos alunos acessaram a disciplina, quantidade de acessos e a média geral de acessos. Também é possível ter uma ideia do engajamento das atividades através da quantidade de alunos que acessaram.

Figura 1 – Menu do *plugin*



Fonte: Figura do autor (2021)

Figura 2 – Tela de gerenciamento

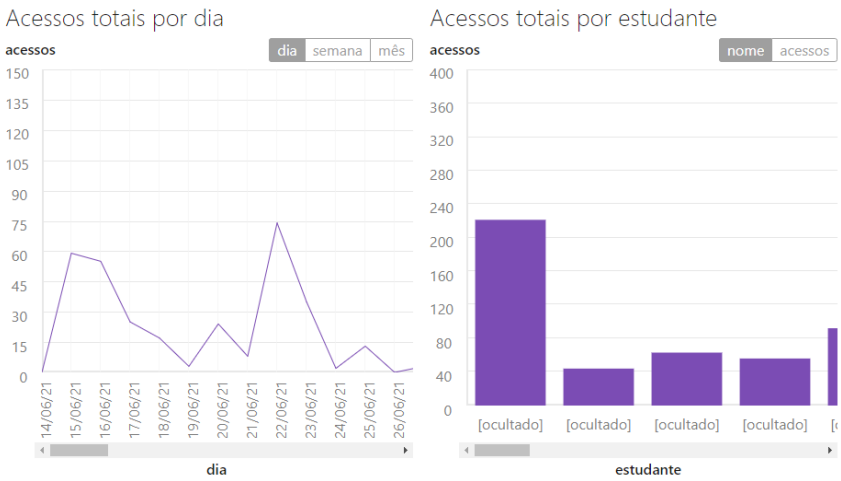


Fonte: Figura do autor (2021)

2.1.1.2 Estudantes

Nesta opção existe um gráfico de linhas com a informação temporal de acessos totais que pode ser alterado para diariamente, semanalmente ou mensalmente, ademais um gráfico de barras com a quantidade de acessos individuais dos alunos (Figura 3).

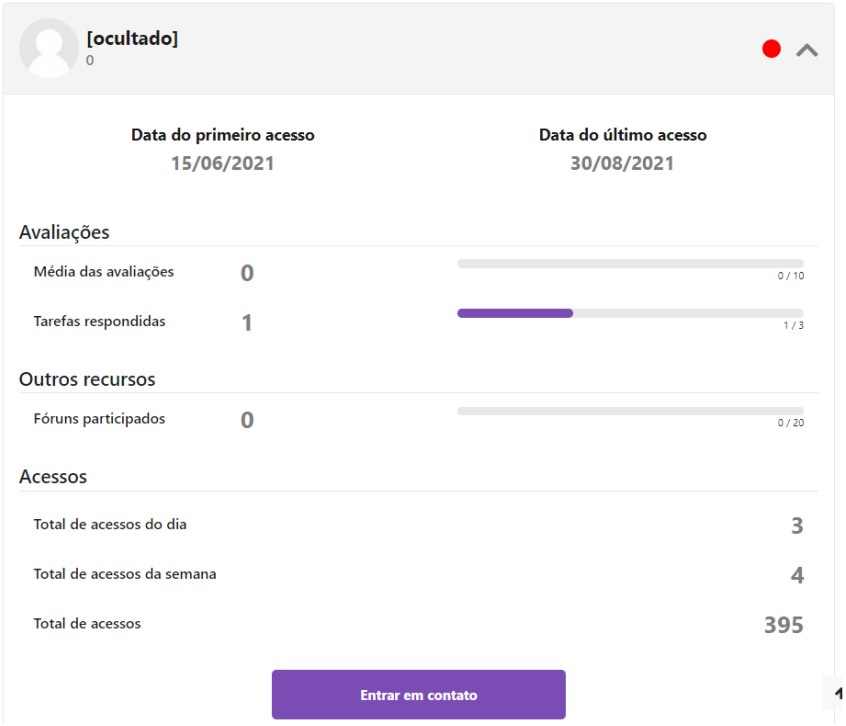
Figura 3 – Gráficos de acessos



Fonte: Figura do autor (2021)

Existe a opção de visualizar os dados individuais de um aluno e caso haja necessidade o botão entrar em contato facilita a comunicação com o estudante que pode estar tendo dificuldade. Um ponto vermelho no canto oposto ao nome do estudante é o indicador de risco que o modelo desenvolvido neste trabalho retorna da predição (Figura 4).

Figura 4 – Dados dos estudantes

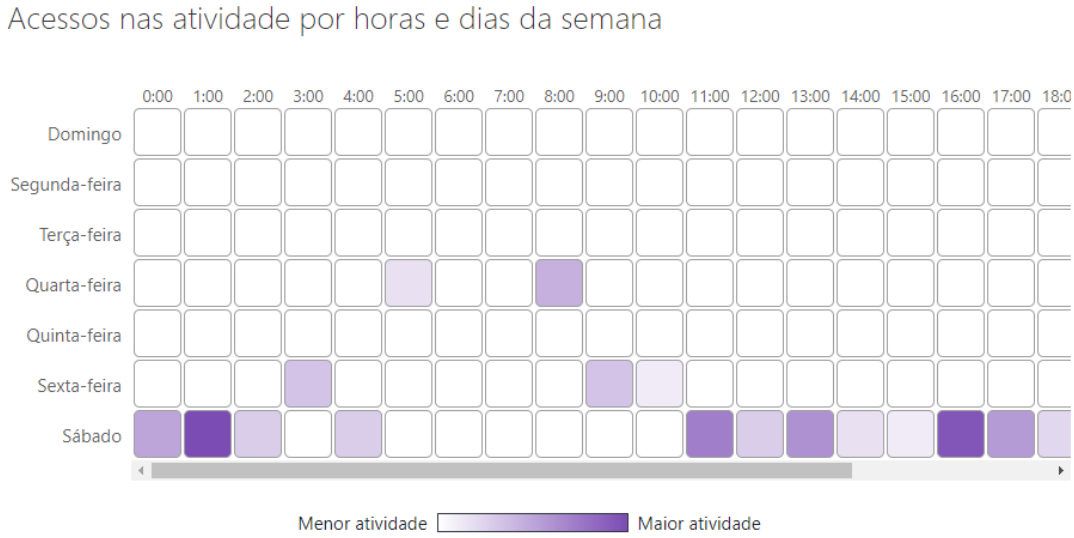


Fonte: Figura do autor (2021)

2.1.1.3 Recursos

A opção de recursos disponibiliza um gráfico de calor que possibilita saber as horas em que os alunos estão mais ativos na disciplina. Com essa informação o professor pode agendar atividades para os melhores horários, criando ainda mais oportunidades de gestão (Figura 5).

Figura 5 – Gráfico de calor



Fonte: Figura do autor (2021)

Esta tela também conta com a informação de quantos acessos obtiveram cada um dos recursos da disciplina, criando um comparativo dos quais são os mais populares. Esses recursos incluem tarefas, fórum, questionários, entre outros.

2.1.1.4 Predição

Como visto neste capítulo o MAD já permite a visualização e facilita a interpretação dos dados de acesso e uso dos recursos disponíveis no moodle da disciplina. O próximo passo mais natural é utilizar os dados disponíveis para a criação de um modelo preditivo que possa auxiliar o professor a identificar os indivíduos em risco de reprovação ou evasão. Portanto é nesse contexto que o trabalho se insere, como uma forma de complementar o que já foi desenvolvido na parte de visualização de dados através de um módulo preditivo.

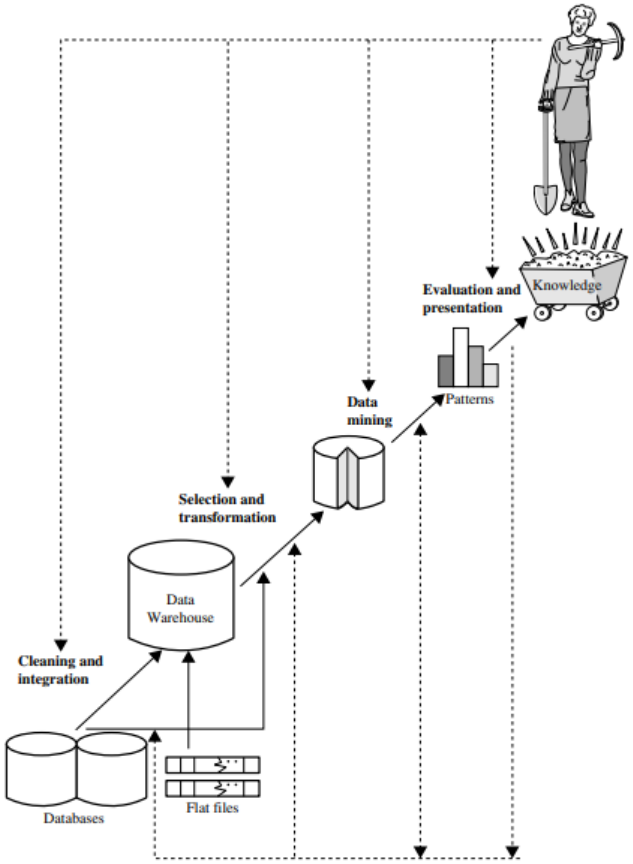
O módulo foi desenvolvido utilizando uma abordagem de contagem de interações que são extraídas da base de dados do moodle, separamos o *dataset* em *subsets* menores que representam as semanas do curso, após a fase de pré-processamento o treinando gera um *array* de modelos. O modelo referente a semana atual do curso é selecionado do *array* e utilizado na predição. A predição é executada automaticamente toda semana através da rotina de tarefas do próprio Moodle.

2.1.2 Mineração de dados

Com o mundo se tornando cada vez mais tecnológico a quantidade de dados que são gerados atingem um ponto onde é necessário técnicas especificas para extração e visualização de dados.

Em Data mining: concepts and techniques (2011) é definido que mineração de dados como sendo o processo de descobrimento de padrões e informação a partir de uma grande quantidade de dados, ou seja, mineração de dados pode ser vista como um passo essencial para o processo de *Knowledge Discovery in Databases* (KDD). A Figura 6 extraída do livro ilustra este processo.

Figura 6 – KDD



Fonte: Data mining: concepts and techniques (2011)

A mineração de dados possui diversas funcionalidades (HAN; KAMBER; PEI, 2011), como a caracterização e discriminação, mineração de padrões frequentes, associação e correlação, classificação e regressão, análise de *cluster* e análise de *outlier*.

Este trabalho utiliza classificação como abordagem.

2.1.3 Classificação

Segundo os autores Han, Kamber e Pei (2011), classificação é uma forma de analise de dados onde são extraídos modelos que descrevem uma classe importante de dados,

esses modelos são chamados de classificadores. A classificação funciona em dois passos, o primeiro é a aprendizagem ou treinamento onde o classificador é construído a partir de dados pré-selecionados. O segundo passo é a classificação ou predição, o classificador recebe dados e retorna o resultado.

No contexto de acurácia do modelo se simplesmente for utilizado os dados de treinamento para chegar em alguma estimativa ela seria, muito provavelmente, otimista. Isso acontece porque classificadores possuem uma tendência de causar um *overfit* dos dados, ou seja, a introdução de anomalias durante o treinamento que não estão presentes nos dados a serem classificados. Existem métricas que podem ser usadas quando os dados tendem a ser desbalanceados.Em Macarini *et al.* (2019) é utilizado a área embaixo da curva Curva Característica de Operação do Receptor (ROC) para descrever a qualidade do modelo.

Neste trabalho os classificadores são usados para identificar os alunos que estão em risco de reprovação ou evasão, isso é feito treinando o modelo classificador com a quantidade de interações que os alunos tiveram na disciplina.

2.1.4 Pré-processamento

As bases de dados são altamente suscetíveis a ruídos, ausência e inconsistência de dados devido ao tamanho e origem das inúmeras fontes. Dados de baixa qualidade vão levar a baixa qualidade dos resultados de mineração (HAN; KAMBER; PEI, 2011).

Existem técnicas de pré-processamento que podem melhorar a qualidade dos dados:

- Limpeza dos dados: Pode ser aplicado para remover ruído e corrigir inconsistências
- Integração de dados: Agrega dados de diferentes fontes em uma base coerente.
- Redução de dados: Reduz a quantidade de dados através de agregação, eliminando redundâncias ou clusterização.
- Transformação de dados: Pode ser aplicado quando dados são normalizados para caber em uma escala entre 0.0 e 1.0.

A criação de derivadas variáveis e a reestruturação dos dados para que se adéquem ao algoritmo escolhido também foi tratada como um passo do pré-processamento. Duas variáveis foram derivadas a partir da quantidade de acessos do estudantes: média e mediana.

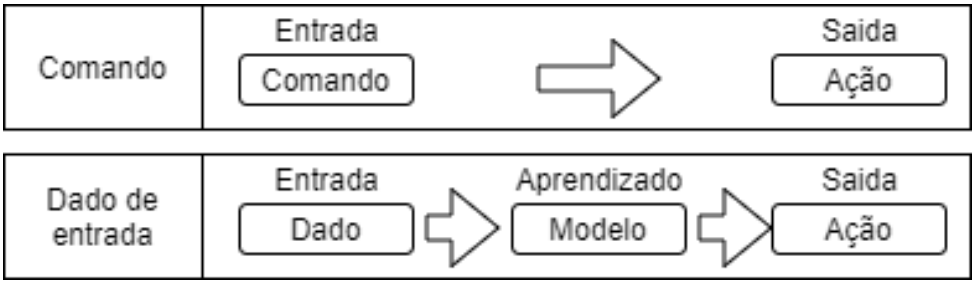
2.1.5 Aprendizado de máquina

Em 1959 a *International Business Machines* (IBM) publicou um artigo com autoria de Arthur Samuel que investigava o uso de aprendizado de máquina em um jogo de damas para saber se um computador pode ser programado de uma forma que ele aprenda a jogar

melhor do que a pessoa que escreveu o programa (SAMUEL, 1959). Embora não tenha sido o primeiro trabalho publicado a utilizar a expressão aprendizado de máquina ele é considerado o primeiro trabalho a utilizar o termo no mesmo contexto dos dias atuais.

Um ponto importante de aprendizado de máquina é o conceito de auto-aprendizagem, que envolve a aplicação de modelagem estatística para detectar padrões e melhorar a performance baseado em dados e informações empíricas. Um computador pode realizar uma tarefa a partir de um dado de entrada ao invés de um comando (WALKER, 2018). Decisões são geradas decifrando relações e padrões nos dados utilizando lógica

Figura 7 – Comparação entre comando e dado de entrada



Fonte: Figura adaptada do livro (WALKER, 2018)

probabilística, tentativa e erro e outras técnicas que envolva capacidade computacional. Existe uma relação onde o dado de saída depende do dado de entrada, mas não de regras pre-determinadas definidas por um programador. Embora as tarefas de inserir os dados no modelo, escolher um algoritmo e tratar os dados ainda existam, a máquina e o programador estão em camadas separadas quando comparados com a programação tradicional.

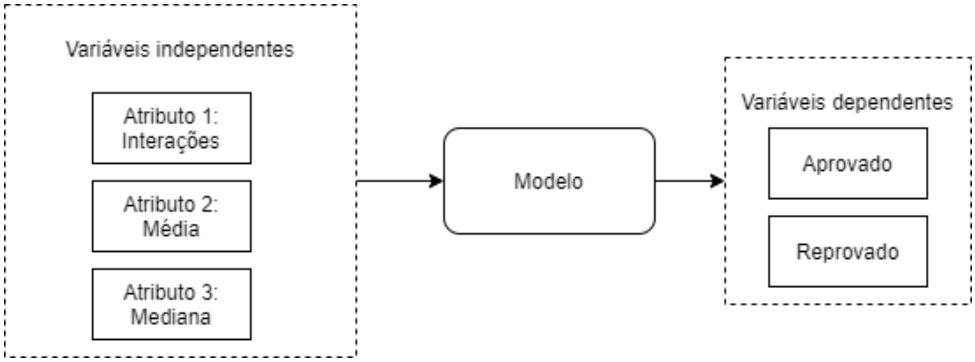
É muito comum na área de aprendizado de máquina separar o conjunto de dados em dados de treinamento e dados de teste. O primeiro conjunto é utilizado para treinar o modelo e uma vez treinado utilizar o segundo conjunto para testar a performance, isso se chama validação cruzada e é utilizada extensamente neste campo.

Em Walker (2018) é apresentada três categorias em aprendizado de máquina e como elas lidam com os dados de entrada e as variáveis de saída, estas categorias serão discutidas a seguir.

2.1.5.1 Aprendizado supervisionado

O aprendizado supervisionado consiste em encontrar padrões com base em um *dataset* rotulado e decodificar a relação entre as dados de entrada (variáveis independentes) e variáveis de saída (variáveis dependentes). Este tipo de treinamento requer fornecer a máquina dados contendo diferentes dados de entrada em conjunto da solução, como ambas entradas e saídas são conhecidas torna o *dataset* rotulado. O algoritmo irá descifrar os padrões existem entre as varáveis de entrar e saída e utilizar o que foi aprendido para fornecer as próximas previsões.

Figura 8 – Aprendizado supervisionado



Fonte: Figura adaptada do livro (WALKER, 2018)

A Figura 8 utiliza o cenário deste trabalho como exemplo, a previsão de se o estudante estaria em risco ou não analisando outros estudantes e a relação entre as varáveis independentes como, quantidade de interações, média das interações e mediana das interações e se o aluno foi aprovado. Dado que o algorítimo sabe o resultado que o aluno teve ele pode trabalhar retroativamente buscando alguma relação entre aprovação ou reprovação (variáveis dependentes) e interações (variáveis independentes). Quando o algoritmo descifra a relação desses dados um modelo é criado, representado por uma equação que produz um resultado baseado no que foi aprendido sobre as variáveis de entrada. Uma vez refinado e pronto, o modelo pode ser testado assim determinando sua qualidade.

Exemplos de algoritmos para aprendizado supervisionado incluem: árvores de decisão, KNN, redes neurais, regressão e máquina de vetores de suporte.

2.1.5.2 Aprendizado não supervisionado

No caso de aprendizado não supervisionado, as variáveis de saída não são rotuladas e as combinações entre variáveis dependentes e independentes são desconhecidas. Esse tipo de aprendizado foca em analisar as relações entre variáveis de entrada e descobrir padrões que podem ser utilizados para rotular as possíveis saídas.

A grande vantagem deste método é a capacidade de descobrir padrões ocultos nos dados e gera a oportunidade de realizar análises adicionais um vez que novos grupos são identificados. O exemplo dado por (WALKER, 2018) é uma empresa chamada *DataVisor*, que possui um modelo de negócio baseado em Aprendizado não supervisionado. A empresa protege cliente de fraudes online como, *spam*, avaliações falsas, aplicativos falsos e transações fraudulentas. DataVisor aplica esse tipo de aprendizado de máquina para detectar tipos de fraudes que ainda não foram classificadas. Nesse cenário a variável dependente é o evento de ataque (saída) e as variáveis independentes (entradas) são as variáveis preditivas de ataque. Exemplos desses dados de entradas podem ser:

- Um pedido extremamente grande de um usuário desconhecido.

- Um grande aumento no número de avaliações.
- Avaliações idênticas ou similares de usuários diferentes.
- Endereço de entrega suspeito.

2.1.5.3 Aprendizagem por reforço

A aprendizagem por reforço é a mais avançada categoria de aprendizado de máquina. Este tipo de aprendizado não segue os mesmo padrões dos vistos anteriormente, distinto dos outros o aprendizado por reforço constrói um modelo de predição através da retroalimentação de tentativa e erro aproveitando o discernimento das iterações passadas.

O objetivo é alcançar um alvo especifico (saída) através de inúmeras tentativas de uma vasta combinação de entradas e avaliando seu desempenho. Um modelo de aprendizagem por reforço possui critérios de performance bem definidos, no caso de veículos autônomos não causar um acidente possui uma pontuação positiva e no caso do jogo de xadrez não perder também é um critério positivo.

2.1.6 AdaBoost

Métodos ensemble combinam diversos algoritmos base para formar um único algoritmo preditivo otimizado. Por exemplo, um algoritmo típico de árvore de decisão para classificação pega diversos fatores, transformam eles em regras e dão um fator para cada um. Logo, ou uma decisão é tomada ou outro fator é considerado.

O resultado da árvore de decisão pode se tornar ambíguo quando muitos fatores são considerados. Surge então os métodos ensemble, ao invés de depender de uma única arvore de decisão para realizar a predição o método leva em consideração diversas árvores e agrega elas em um módulo preditivo.

Existem três tipos de métodos ensemble:

- Diminuir Variação (Bagging)
- Diminuir Bias (Boosting)
- Melhorar as previsões (Stacking)

É possível dividir os métodos ensemble em dois grupos:

- Aprendizado sequencial, onde diferentes modelos são gerados sequencialmente e os modelos sucessores aprendem com os erros dos modelos passados. O objetivo é explorar a dependência entre modelos dando dados rotulado erroneamente um peso maior (AdaBoost).
- Aprendizado paralelo, onde modelos base são gerados em paralelo. Isso explora a independência entre modelos criando uma media dos erros (Random Forest).

Os algoritmos de Boosting tentam criar um modelo preditivo a partir dos erros de diversos modelos mais fracos. Começamos criando um modelo dos dados de treinamento, depois tentamos reduzir o erro do modelo para criar um novo e assim por diante, sempre tentando corrigir os erros do modelo anterior. Boosting tenta reduzir a quantidade de bias que um modelo possui quando não consegue identificar tendências relevantes nos dados.

O AdaBoost é uma técnica extremamente popular que combina múltiplos classificadores fracos sequencialmente para criar um único classificador forte. O classificador pode ser árvores de decisão, regressão ou outros.

2.1.6.1 AdaBoost-SAMME

Este trabalho utiliza o algoritmo AdaBoost-SAMME para a criação do modelo preditivo, logo é importante entender como ele funciona.

O algoritmo SAMME amplia o algoritmo original para atender problemas com múltiplas classes. Porém o SAMME é reduzido ao algoritmo original do AdaBoost quando o problema possui apenas duas classes.

Supondo um conjunto de dados de treinamento $(x_i, c_i), ..., (x_n, c_n)$, onde a entrada (variável de predição) $x_i \in R_p$ e a saída (variável de resposta) c_i é qualitativa e assume valores em um conjunto finito $\{1, 2, ..., K\}$. K é o numero de classes. O objetivo é encontrar uma regra de classificação C(x), a partir do conjunto de treinamento, de modo que quando fornecido uma nova entrada x é atribuído um rótulo (label) c que pertence ao conjunto $\{1, 2, ..., K\}$. Assumindo que os dados de treinamento são amostras independentes e distribuídas igualmente através de uma distribuição probabilística desconhecida Prob(X, C). Então a taxa de erro na classificação C(x) é:

$$1 - \sum_{K}^{k=1} E_X \mathbb{I}_{c=(\mathbf{X})} Prob\left(C = k|\mathbf{X}\right) \tag{1}$$

A equação $C^*(x) = \arg\max_k Prob\left(C = k|X = x\right)$ irá minimizar o erro na classificação. Este classificador é conhecido como Bayes e a taxa de erro é chamada de erro de Bayes (ZHU *et al.*, 2006). O algoritmo AdaBoost é um processo iterativo que combina diversos classificadores fracos para aproximar o classificador Bayes $C^*(x)$. Começando com um peso nulo o AdaBoost cria um classificador, por exemplo árvore de decisão que introduz rótulos (labels). A idéia é aumentar o peso (*boosting*) no treinamento dos pontos de dados que foram classificados errado. Fazendo T(x) ser um classificador fraco que atribui um rótulo x, o algoritmo segue conforme a figura 9.

2.2 FUNDAMENTAÇÃO TECNOLÓGICA

Nesta seção, será dada vista as tecnologias empregadas no desenvolvimento do trabalho. Temos as tecnologias empregadas no lado do *plugin* como PHP, JavaScript, MySQL,

Figura 9 – Pseudo código AdaBoost

1. *Initialize the observation weights* $w_i = 1/n,\ i = 1, 2, \ldots, n$.
2. *For* $m = 1$ *to* M*:*
   (a) *Fit a classifier* $T^{(m)}(\boldsymbol{x})$ *to the training data using weights* w_i.
   (b) *Compute*
   $$err^{(m)} = \sum_{i=1}^{n} w_i \mathbb{I}\left(c_i \neq T^{(m)}(\boldsymbol{x}_i)\right) / \sum_{i=1}^{n} w_i.$$
   (c) *Compute*
   $$\alpha^{(m)} = \log \frac{1 - err^{(m)}}{err^{(m)}}.$$
   (d) *Set*
   $$w_i \leftarrow w_i \cdot \exp\left(\alpha^{(m)} \cdot \mathbb{I}\left(c_i \neq T^{(m)}(\boldsymbol{x}_i)\right)\right),\ i = 1, 2, \ldots, n.$$
   (e) *Re-normalize* w_i.
3. *Output*
   $$C(\boldsymbol{x}) = \arg\max_k \sum_{m=1}^{M} \alpha^{(m)} \cdot \mathbb{I}(T^{(m)}(\boldsymbol{x}) = k).$$

Fonte: Figura retirada de Zhu *et al.* (2006)

Moodle e também no modelo onde foi empregado Python em conjunto de bibliotecas que fornecem algumas funcionalidades.

2.2.1 *plugin*

Através do *plugin* MAD foi preciso extrair as informações da base de dados do Moodle, enviar para o módulo de predição em Python, receber os resultados e exibi-los. Para isso foi preciso utilizar algumas tecnologias que serão discutidas aqui.

2.2.1.1 PHP

PHP é uma linguagem de *script* que pode ser embutida no HTML. Ele possui um pouco das sintaxes que são encontradas nas linguagens C, Java e Perl com o adicional de recursos específicos do próprio PHP [1]. O grande objetivo desta linguagem é permitir desenvolvedores a escrever, de forma rápida, páginas na internet que sejam dinamicamente geradas. Porém existem outros usos que servem esse trabalho:

- Permite realizar funções em um sistema como criação, leitura, atualização e exclusão em arquivos.
- Adicionar, deletar e modificar elementos dentro de um banco de dados.

[1] https://www.php.net/manual/en/faq.general.php

- Integração nativa com o banco de dados MySQL.

2.2.1.2 JavaScript

JavaScript faz parte das três linguagens principais do desenvolvimento web. Enquanto HTML e CSS fornecem estrutura e estilo para a aplicação, JavaScript permite definir comportamentos e criar funcionalidades. Criado por Brendan Eich em 1995, foi desenvolvido para o navegador Netscape2 e devido ao crescimento acelerado da linguagem surgiu a necessidade de manter e gerência-la propriamente. Em 1997 a Netscape entregou a tarefa de especificação para para a ECMA, uma associação dedicada à padronização de sistemas de informação[2].

Em 2005 um artigo publicado por Jesse James Garrett introduziu o AJAX (Asynchronous JavaScript + XML) possibilitando a capacidade de realizar requisições assíncronas e é em cima dessas requisições que muitas funcionalidades do *plugin*, incluindo o módulo preditivo, são construídas.

2.2.1.3 SQL

SQL é uma linguagem utilizada para acessar e manipular dados que estão armazenados dentro de um banco de dados relacional. É particularmente útil quando existe a necessidade de lidar com dados estruturados, isto é, dados que possuem uma estrutura rígida e pensada previamente a inserção no banco de dados.

O SQL serve um propósito importante no desenvolvimento do projeto, ele é responsável pela extração dos dados de acesso que estão armazenados no banco de dados do Moodle em uma tabela de *logs*. Durante a extração o código SQL também organiza e agrupa os resultados, como uma forma inicial de preprocessamento dos dados que serão entregues ao algoritmo responsável pela predição.

2.2.1.4 Moodle

O Moodle é uma plataforma de aprendizagem onde é possível criar ambientes customizados para cada disciplina. Ele permite a adição de *plugins* ou extensões que introduzem novas funcionalidades dependendo da necessidade do usuário.

Esses *plugins* são desenvolvidos pela própria comunidade de usuários tornando o Moodle colaborativo e aprimorado pela comunidade.

2.2.2 Predição

Uma vez que os dados de interações foram extraídos dos *logs* através do *plugin*, o algoritmo de predição desenvolvido na linguagem Python precisa acessa-los e iniciar os próximos passos para que o treinamento ou predição possam ocorrer. A linguagem Python

[2] https://www.springboard.com/blog/data-science/history-of-javascript

foi escolhida por possuir bibliotecas que tornam o trabalho mais simples e elas também serão discutidas nessa seção.

2.2.2.1 Python

Python é uma linguagem interpretada e orientada a objeto que incorpora módulos, exceções, digitação dinâmica, estrutura de dados dinâmicas de alto nível e classes. Possui suporte para múltiplos paradigmas de programação além da orientação a objeto, como procedural e funcional. Python é uma linguagem poderosa e ao mesmo tempo mantém uma sintaxe clara, possuindo bibliotecas que abrangem muitas áreas inclusive aprendizado de máquina e ciência de dados[3].

2.2.2.2 Pandas

Pandas é uma biblioteca para Python que provê estrutura de dados rápidas, flexíveis e expressivas capazes de trabalhar com dados relacionado ou rotulados de forma fácil e intuitiva. Criado para ser um bloco fundamental de alto nível para realizar trabalhos práticos em análise de dados.Pandas possui duas estruturas de dados primárias, series (1-Dimensão) e DataFrame (2-Dimensões) que lidam com a maior parte dos usos típicos.[4]

2.2.2.3 Scikit-learn

Scikit-learn é uma biblioteca em Python que contém diversos algoritmos de aprendizagem supervisionados e não supervisionados. Construída em cima de tecnologias como Numpy, Pandas e Matplotlib o Scikit-learn trabalha muito bem na área de ciência de dados[5]. As funcionalidades dessa biblioteca incluem:

- Regressão, incluindo regressão logística e linear.
- Classificação, incluindo AdaBoost.
- Clusterização, incluindo K-Means e K-Means++.
- Seleção de modelos.
- Preprocessamento.

2.2.2.4 Joblib

Quando o Scikit-learn treina um modelo é conveniente ter uma forma de persistir-lo para futuras predições, sem a necessidade de realizar o treinamento novamente. O Joblib é escolhido por ser mais eficiente quando trabalhamos em objetos que contêm grande

[3] https://docs.python.org/3/faq/general.html
[4] https://pandas.pydata.org/docs/getting_started/overview.html
[5] https://www.codecademy.com/article/scikit-learn

quantidades de *arrays* da biblioteca Numpy, o que ocorre frequentemente quando usamos scikit-learn [6].

[6] https://scikit-learn.org/stable/modules/model_persistence.html

3 TRABALHOS CORRELATOS

Neste capitulo serão apresentado alguns trabalhos que realizaram pesquisa semelhantes envolvendo treinamento de modelos preditivos para o problema de evasão estudantil.

Konstantinidis e Grafton (2013) buscaram utilizar o Excel, editor de planilhas produzido pela Microsoft, para analisar os dados presentes nos *logs* do Moodle. Foram extraídos o total de páginas visualizadas dentro de uma disciplina específica, quantidade de usuário únicos em uma disciplina, quantidade de interações, número total de páginas que constituem a estrutura da disciplina, endereço IP do usuário que realizou a interação e a média da duração de um usuário dentro do espaço da disciplina. Com as informações extraídas colocadas dentro do Excel foi possível criar uma representação gráfica dos dados ao longo do tempo, permitindo que professores possam compreender o comportamentos dos alunos.

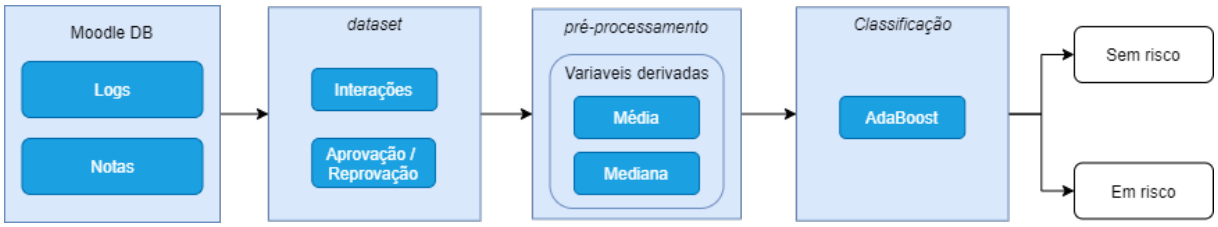
Distante *et al.* (2020b) desenvolveram o MILA uma ferramenta analítica de dados de aprendizagem que busca melhorar o processo de ensino nos ambientes virtuais da Universidade de Roma Unitelma Sapienza. Além de visualizações gráficas dos dados, foi treinado um modelo utilizando informações do último acesso de um estudante e atividades que foram realizadas para prever risco de abandono da disciplina, também é possível configurar parâmetros e qualquer outro tipo de modelo pode ser implementado.

Sáiz-Manzanares *et al.* (2020) utilizou SPSS v.24 um software de analise estatística com dados extraídos dos *logs* para aplicar técnicas de predição e classificação para identificar estudantes em risco. Uma técnica de clusterização encontrou três *clusters* com relação a frequência de acesso dos estudantes aos recursos do Moodle. A ferramenta proporcionou ótimos resultados acadêmicos nas turmas com taxas de sucesso e performance entre 95% e 100%.

4 MÉTODO PROPOSTO

Neste capitulo serão descrito os métodos utilizados para alcançar o objetivo do trabalho que possui como intenção a criação de um módulo preditivo composto de modelos para cada semana da disciplina utilizando somente as interações do usuário no Moodle.

Figura 10 – Visão geral do método



Fonte: Figura do autor (2021)

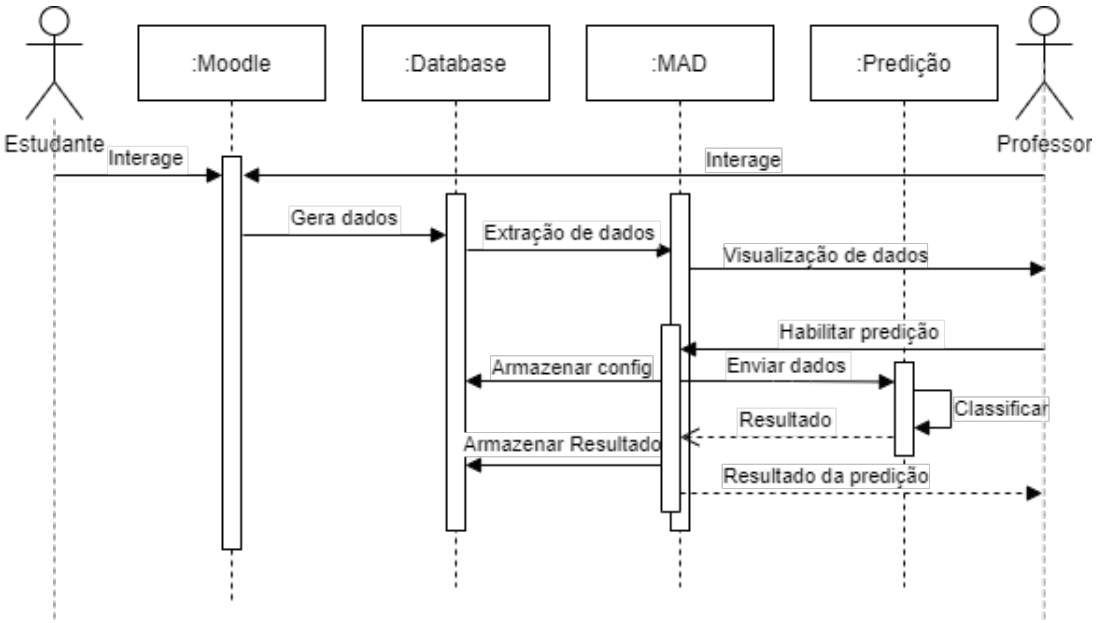
Iniciando pelo banco de dados do Moodle onde é preciso acessar as tabelas que contém os dados de interações e notas dos cursos anteriores. As notas e interações de uma disciplina anterior são usadas para treinar um modelo e as interações da disciplina atual na qual o *plugin* esta ativado são usadas na predição. O inicio do preprocessamento também ocorre durante essa fase de extração dos dados para tornar a fase subsequente mais fácil.

O *dataset* é resultado da fase de extração dos dados e consiste de três arquivos no formato JSON (JavaScript Object Notation). Dois arquivos são informações sobre interações, um dos arquivos se refere a um curso anterior e o outro a predição atual, o terceiro arquivo são as notas do curso anterior.

A fase de preprocessamento recebe o *dataset* e prepara os dados para serem utilizados pelo modelo, tanto no treinamento quanto na predição. Com o modelo criado o resultado da predição deve retornar para o *plugin* onde será armazenado e posteriormente exibido para o professor.

O diagrama de sequência Figura 11 exemplifica como os dois atores, estudantes e professor, interagem com os objetos do sistema. Após habilitar a predição não há nenhum passo adicional e tudo corre automaticamente. O módulo foi construído com o intuito de não precisar de nenhuma intervenção do usuário além da ativação, isso torna extremamente fácil o uso e diminui os motivos pelos quais um usuário não queira utiliza-lo.

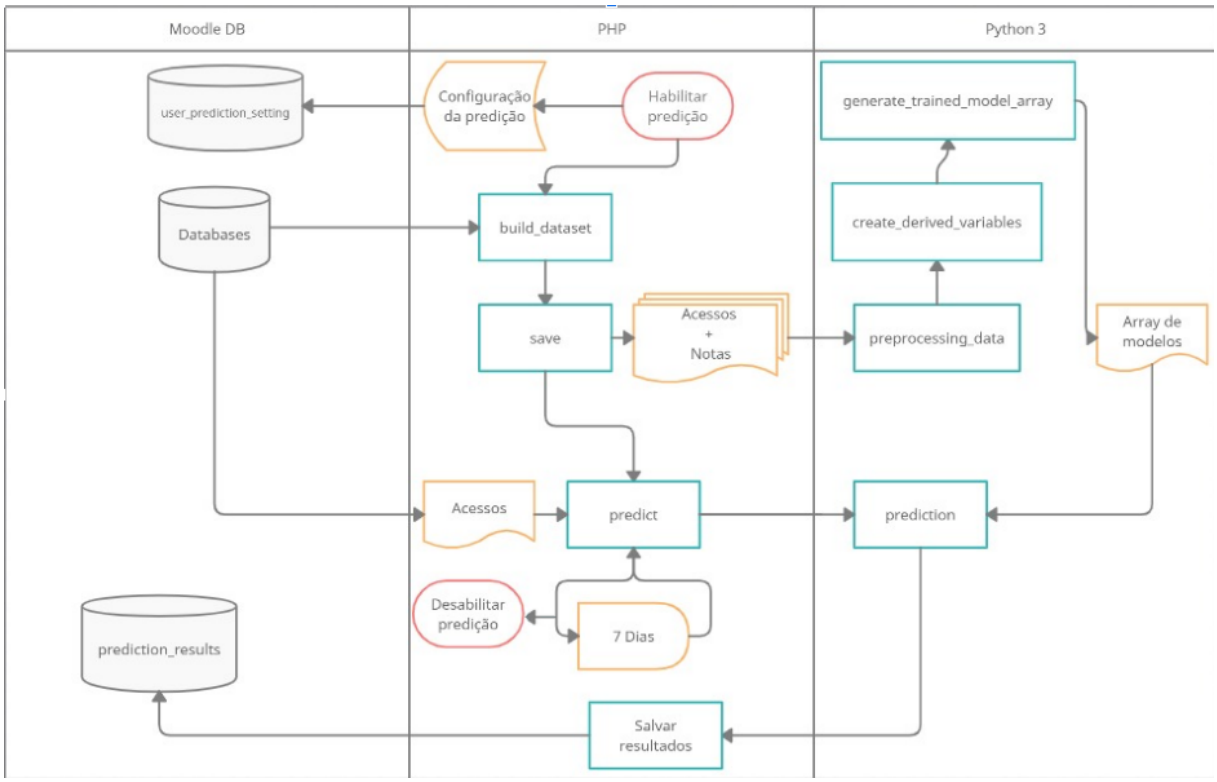
Figura 11 – Diagrama de sequência



Fonte: Figura do autor (2021)

O fluxograma da Figura 12 fornece uma visão geral de como o banco de dados, *plugin* em PHP e a predição em Python trabalham juntos. A integração desses sistemas foi uma estágio importante no desenvolvimento do trabalho.

Figura 12 – Diagrama de sequência do sistema



Fonte: Figura do autor (2021)

4.1 CONJUNTO DE DADOS

4.1.1 Interações

Para a criação do *dataset* que será usado no treinamento dos modelos foi necessário extrair as notas finais e quantidade de interações dos alunos em uma disciplina já concluída que foi selecionada pelo professor, para isso foi preciso acessar o histórico ou *log* já presente no sistema do Moodle.

A tabela de *logs* em questão é a *mdl_logstore_standard_log* e como mostra a Figura 13 ela possui uma chave primária id e quatro chaves estrangeiras: contextid, contextinstanceid, userid e courseid. O atributo courseid serve para indicar qual disciplina ou curso aquela linha de *log* se refere, logo é preciso filtrar somente pelo curso que desejamos extrair as interações. Também realizamos uma operação do SQL chamada "INNER JOIN" no userid para juntar informações da tabela *mdl_role_assignments* que contêm os cargos existentes e conseguir identificar quais usuário são estudantes.

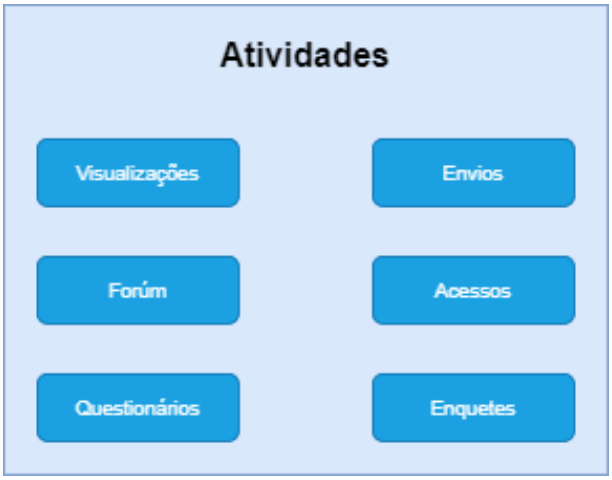
Figura 13 – Tabela de *logs*

logstore_standard_log	[table]
id	bigint[19]
eventname	varchar[255]
component	varchar[100]
action	varchar[100]
target	varchar[100]
objecttable	varchar[50]
objectid	bigint[19]
crud	varchar[1]
edulevel	bit[1]
contextid	bigint[19]
contextlevel	bigint[19]
contextinstanceid	bigint[19]
userid	bigint[19]
courseid	bigint[19]
relateduserid	bigint[19]
anonymous	bit[1]
other	longtext[2147483647]
timecreated	bigint[19]
origin	varchar[10]
ip	varchar[45]
realuserid	bigint[19]
< 1	0 >

Fonte: Figura do autor (2021)

A Figura 14 mostra algumas das atividades que são filtradas.

Figura 14 – Atividades filtradas



Fonte: Figura do autor (2021)

Será feita a contagem de interações com cada tarefa para cada semana, por exemplo se um aluno acessou o fórum duas vezes isso gera duas interações que vão ser somadas com todas as outras. O Moodle utiliza *unixtime*, isto é, quantidade de segundos desde 1 de janeiro de 1970, precisamos converter esta data, agrupar por semanas e realizar uma contagem.

As interações e notas que serão utilizadas no treinamento dos modelos são armazenadas no diretório do *plugin*. A estrutura do diretório contém uma pasta separada para cada curso que possui a predição ativa, dentro é encontrado dois arquivos um de interações e o outro com as notas finais dos alunos. A árvore de diretório é exemplificada a seguir:

Figura 15 – Árvore de diretório

```
prediction
└── datasets
    ├── dataset_1
    │   ├── interações_treino_1.json
    │   ├── notas_treino_1.json
    │   └── interações_predição_1.json
    └── dataset_2
        ├── interações_treino_1.json
        ├── notas_treino_1.json
        └── interações_predição_1.json
```

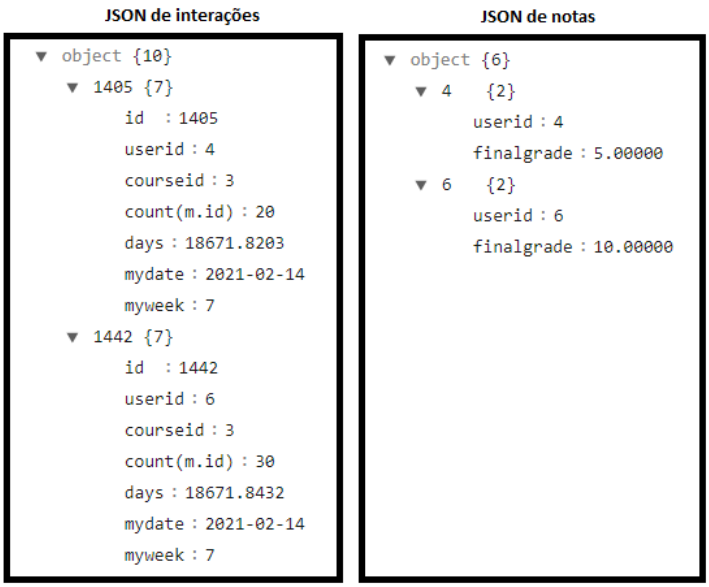
Fonte: Figura do autor (2021)

4.1.2 Arquivos JSON

Os arquivos criados a partir da extração de dados do moodle podem ser separados em dois tipo, interações e notas. As notas em conjunto com as interações são retiradas de

uma disciplina anterior e são utilizadas somente no treinamento, outro arquivo contêm as interações do curso que desejamos prever. O conteúdo dos arquivos é como segue abaixo.

Figura 16 – Conteúdo JSON



Fonte: Figura do autor (2021)

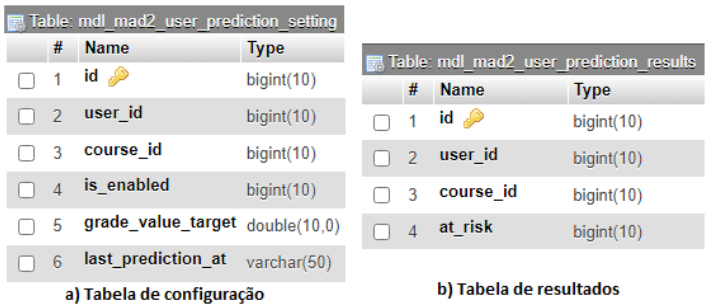
Os campos mais importantes são:

- **userid**: Número identificador do usuário no sistema Moodle.
- **courseid**: Número identificador da disciplina no sistema Moodle.
- **count(m.id)**: Quantidade de interações.
- **myweek**: Número da semana.
- **finalgrade**: Nota final do aluno na disciplina.

4.1.3 Configurações

Para o funcionamento do módulo preditivo dentro do MAD foi preciso criar duas tabelas(Figura 17) que são geradas quando o *plugin* é instalado e ficam juntas as tabelas do Moodle. Elas podem ser identificadas na base de dados pelo uso do prefixo "mdl_mad2".

Figura 17 – Tabelas MAD



Fonte: Figura do autor (2021)

A primeira tabela chamada mdl_mad2_user_prediction_setting contêm informações sobre a configuração da predição em cada uma das disciplinas onde o *plugin* esta sendo usado. Nela estão armazenados as seguintes informações:

- **userid**: número identificador do usuário no sistema Moodle.
- **courseid**: número identificador da disciplina no sistema Moodle.
- **is_enabled**: se a predição esta ativada ou desativada.
- **grade_value_target**: informação da nota de corte para o treinamento.
- **last_prediction_at**: data da última predição.

A segunda tabela possui os resultados das predições para cada um dos alunos:

- **userid**: número identificador do usuário no sistema Moodle.

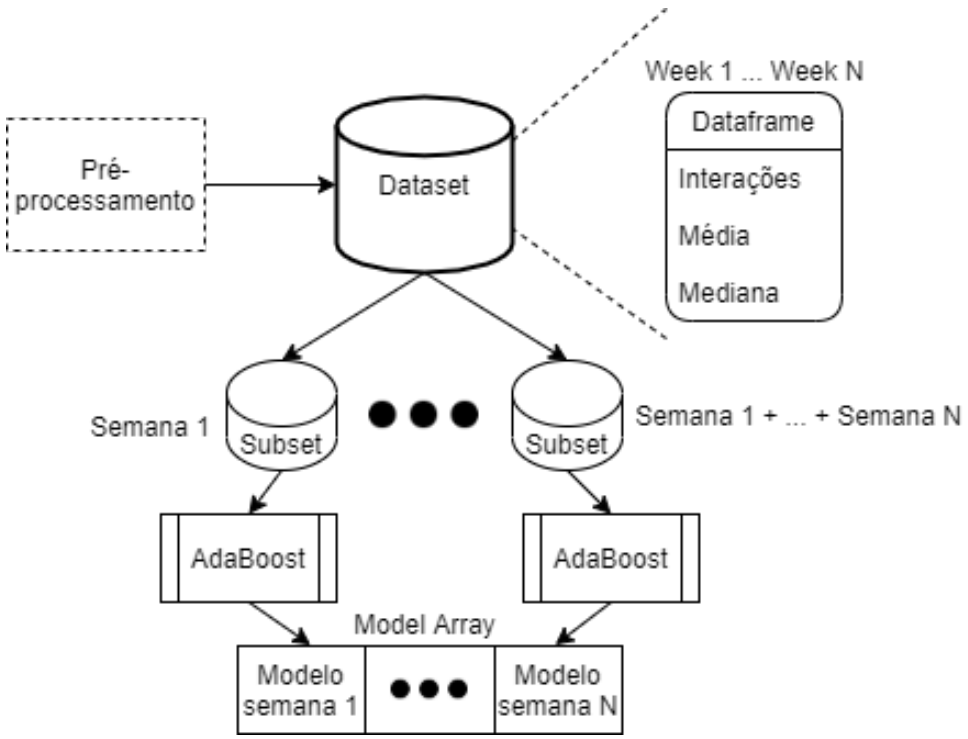
- **courseid**: número identificador da disciplina no sistema Moodle.
- **at_risk**: se o aluno esta ou não em risco.

4.2 MODELOS

A criação do *array* de modelos que será utilizado na predição semanal envolve utilizar os dados extraídos e as bibliotecas *scikit-learn*, *pandas*, *joblib* e *numpy* para treinar um classificador para cada semana.

O *dataset* é separado em *subsets* que representam as semanas de aula. Cada um dos *subsets* é utilizado no treinamento para a criação do *array* de modelos (Figura 18). Também é preciso gerar variáveis derivadas das quantidades de interações, essas variáveis vão tornar o modelo mais robusto.

Figura 18 – Arquitetura do modelo



Fonte: Figura do autor (2021)

Os dados de interações e notas da disciplina que foi escolhida para treinar o modelo são lidos dos arquivos JSON pelo script Python. O primeiro passo é transpor o dataframe em que o JSON foi carregado e excluir as colunas que não serão usadas, por fim juntamos as notas com as interações resultando em um único *dataframe* (Figura 19).

Figura 19 – Dataframe exemplo

	userid	finalgrade	week-0	week-1	week-2
0	4	10	20.0	0.0	0.0
1	10	10	0.0	22.0	4.0
2	6	10	30.0	20.0	32.0
3	9	3	8.0	0.0	0.0
4	8	0	0.0	0.0	0.0
5	7	5	18.0	20.0	25.0

Fonte: Figura do autor (2021)

4.2.1 Pré-processamento

O pré-processamento é a etapa onde as variáveis derivadas são criadas e os dados são ajustados para serem alimentados ao classificador AdaBoost.

O *one-hot-encoding* é utilizado nos campo das notas, transformando a coluna em variáveis binárias (0 ou 1) que vão indicar se o estudante esta ou não em risco de acordo com a nota de corte fornecida quando a predição foi habilitada. Em seguida criamos um *array* dos dados de interação de forma que a primeira posição tenha dados somente da primeira semana, a segunda posição possua dados da primeira e da segunda semana e assim por diante. Com o *array* gerado é iniciada a criação dos dados derivados para cada posição (semana).

Existem duas derivações que serão feitas a partir da contagem de interações, elas são: média e mediana.

$$contagem_i = \sum_{j=1}^{7} x_j \tag{2}$$

$$media_i = \frac{\sum_{j=1}^{7} x_j}{7} \tag{3}$$

$$mediana_i = \begin{cases} amostras_{\frac{n}{2}}, & \text{se } n \text{ for impar} \\ \frac{amostras_{\frac{n}{2}} + amostras_{(\frac{n}{2})+1}}{2}, & \text{se } n \text{ for par} \end{cases} \tag{4}$$

A Equação (1) é a soma das interações de cada dia j em cada semana i. A Equação (2) é a média das interações na semana i, ela é calculada pela contagem das interações dividida por sete dias da semana. A Equação (3) representa a mediana do vetor de interações, onde n é o tamanho do vetor, se n for par a mediana é calculada pela média dos dois valores no meio do vetor.

O pré-processamento termina com a criação de um *array* onde cada posição refere-se a cada semana da disciplina e contém as interações e variáveis derivadas daquela semana

e de todas as anteriores. O dataframe contendo os dados derivados pode ser visto na Figura 20.

Figura 20 – Dataframe exemplo

	week-0	week-1	mean-0	median-0	mean-1	median-1
0	20.0	0.0	2.857143	20.0	0.000000	10.0
1	0.0	22.0	0.000000	0.0	3.142857	11.0
2	30.0	20.0	4.285714	30.0	2.857143	25.0
3	8.0	0.0	1.142857	8.0	0.000000	4.0
4	0.0	0.0	0.000000	0.0	0.000000	0.0
5	18.0	20.0	2.571429	18.0	2.857143	19.0

Fonte: Figura do autor (2021)

4.2.2 Treinamento

Para a geração dos modelos o algoritmo escolhido foi o AdaBoost contido na biblioteca *scikit-learn*. Para utilizá-lo é preciso de *features* que serão as interações e dados derivados, também é necessário o status de aprovado ou reprovado.

Nesta etapa é feito o treinamento dos modelos, para isso cada posição do *array* onde estão contidas as *features* é introduzido ao classificador AdaBoost em conjunto com o estado de aprovado ou reprovado. Cada treinamento gera um modelo que representa aquela semana, armazenando cada resultado em um outro *array* e finalmente criando o *array* de modelos.

O classificador AdaBoost é inicializado com os seguintes parâmetros:

- **base_estimator = None**: O estimador base a partir do qual o ensemble será criado. Caso o valor inserido seja None o classificador por árvore de decisão é inicializado para o *boosting*.
- **n_estimators = 50**: Número máximo de estimadores no qual o processo de *boosting* é terminado. No caso de um *fit* perfeito, o processo é terminado antes.
- **learning_rate = 1.0**: Peso aplicado pra o classificador em cada iteração do *boosting*. Um peso maior aumenta a contribuição de cada classificador, existe uma troca entre a taxa de aprendizagem e a quantidade de estimadores.
- **algorithm = SAMME.R**: O Algoritmo SAMME.R foi desenvolvido para convergir mais rápido que o SAMME.

- **random_state = None**: Controla a aleatoriedade dado ao estimador base em cada iteração do *boosting*. Só é utilizado quando o estimador possui um *random_state*.

4.2.3 Persistência dos modelos

A biblioteca *joblib* possui duas funções que beneficiam muito esse trabalho, uma delas é a *dump* que permite salvar qualquer variável em um arquivo com extensão .joblib e a outra é a função *load* que é capaz de ler esses arquivos e armazenar a variável de volta na memória para ser utilizada.

Não seria viável treinar o modelo cada vez que uma predição é solicitada, a solução adotada foi salvar o *array* de modelos gerado em um arquivo no diretório do *plugin*. A biblioteca *joblib* que permite armazenar o *array* em um arquivo que pode ser facilmente carregado durante uma predição. A persistência do modelo é uma etapa importante já que reduz drasticamente a quantidade de processamento melhorando o desempenho.

4.3 PREDIÇÃO

A predição segue muito de perto os mesmos passos do treinamento, a principal diferença é que os dados são extraídos da disciplina atual toda semana já que o objetivo é classificar se o estudante está ou não em risco antes do fim da disciplina.

Os dados são pré-processados da mesma forma, mas antes da predição ser realizada o *array* de modelos é resgatado e o modelo da semana atual é carregado em uma variável. O módulo *joblib* mencionado anteriormente lida com esse processo. Essa seleção não é difícil já que cada posição refere-se ao número da semana, por exemplo se a semana atual é a segunda semana basta acessar a segunda posição.

Com as interações e o modelo o AdaBoost possui tudo necessário para a classificação de cada estudante, o resultado da predição é então armazenado no banco de dados para que possamos exibir para o usuário.

4.3.1 Automação

Para que a predição ocorra de forma automática, uma ferramenta do próprio Moodle permite a criação de tarefas agendadas que são executadas toda vez que o *script* cron do moodle é iniciado. O cron é um programador de tarefas do próprio sistema operacional e de acordo com a documentação de instalação do Moodle é recomendado executar a cada minuto para que ele funcione de forma adequada.

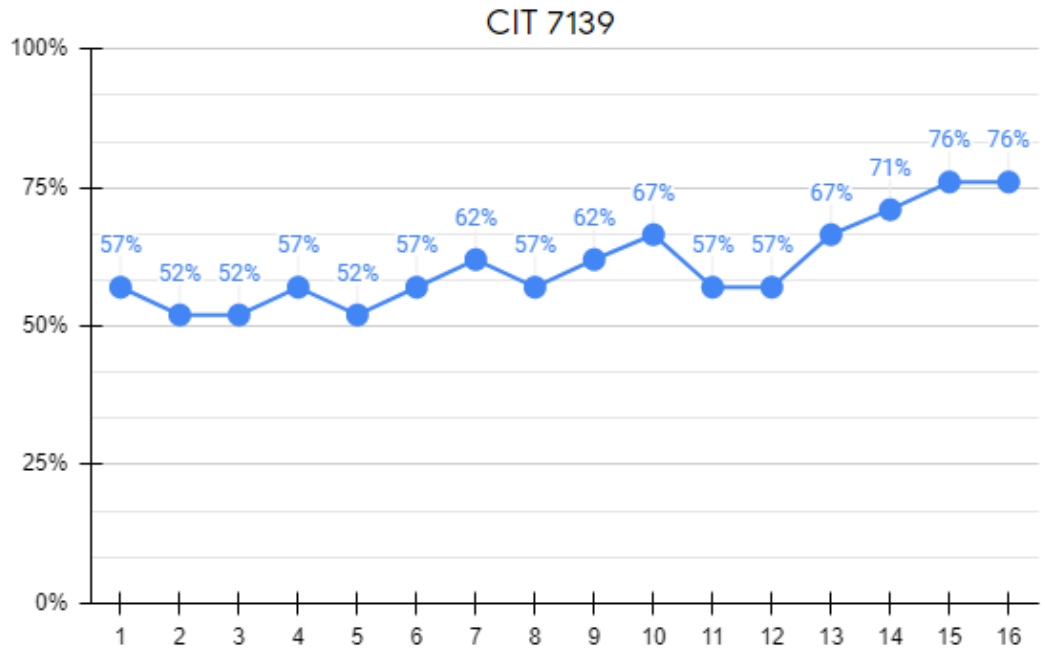
Uma subclasse de */core/task/scheduled_task* é criada e um registro é colocado no arquivo *tasks.php* dentro do *plugin*. A classe possui a tarefa a ser executada e o registro contém informação sobre a periodicidade da execução.

5 RESULTADOS

O *plugin* em conjunto com o módulo preditivo foram testados em duas disciplinas remotas do semestre 2021/1 na UFSC, como não foi possível instala-lo na universidade um servidor próprio foi criado para testar o projeto. Os alunos foram cadastrados e passaram a utilizar o Moodle neste novo servidor. É importante reforçar que não foi possível conseguir acesso a uma disciplina do semestre anterior para realizar o treinamento, logo o modelo foi treinado utilizando dados de uma disciplina de Algoritmos e Programação semipresencial de 2017.

A acurácia ao longo da semana pode ser vista nos gráficos da Figura 21 e Figura 22. Embora os valores obtidos não tenham sido excelentes uma tendência de melhora no modelo ao longo das semanas pode ser observada.

Figura 21 – Acurácia na disciplina Programação em Computadores



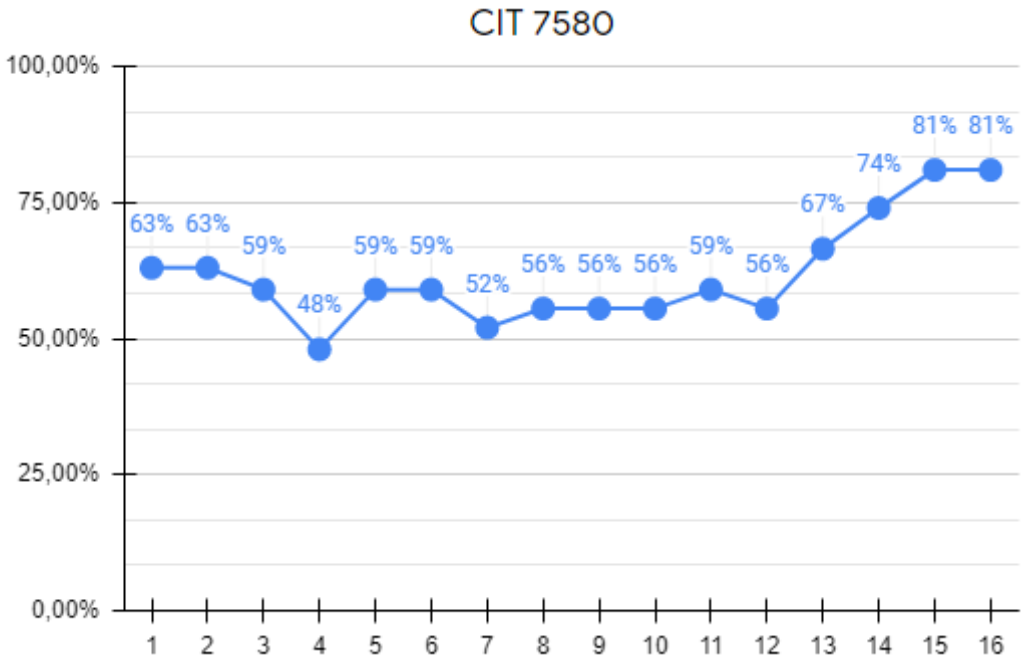
Fonte: Figura do autor (2021)

A portabilidade de modelos foi o tópico da pesquisa de Arroyo-Barrigüete *et al.* (2021) que encontrou um declínio na performance dos modelos treinado com uma disciplina e utilizados em outra, conforme as características entre as disciplinas diminui. Esse comportamento foi observado no teste do *plugin* e o fato da acurácia ter sido melhor na disciplina de Algoritmos e Programação que possui uma maior semelhança com a utilizada no treinamento, corrobora que existe um declínio no desempenho dos modelos ao utilizar os mesmos para conjuntos de dados de contextos diferentes do conjunto de treino.

A usabilidade foi uma parte importante do desenvolvimento e é refletida no menu para habilitar a predição (Figura 23).

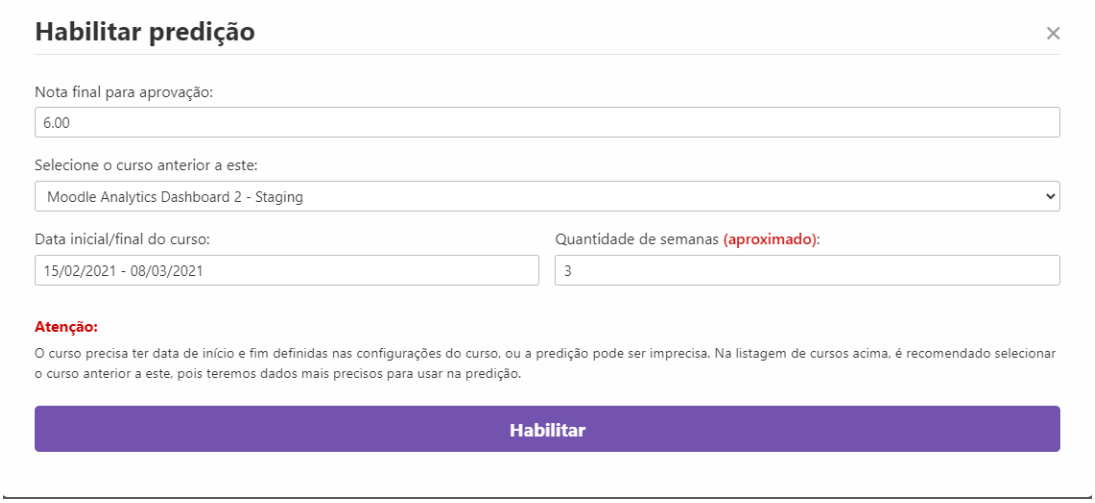
Para habilitar e gerar o *array* de modelos treinado, basta selecionar um curso

Figura 22 – Acurácia na disciplina Algoritmos e Programação



Fonte: Figura do autor (2021)

Figura 23 – Menu da predição



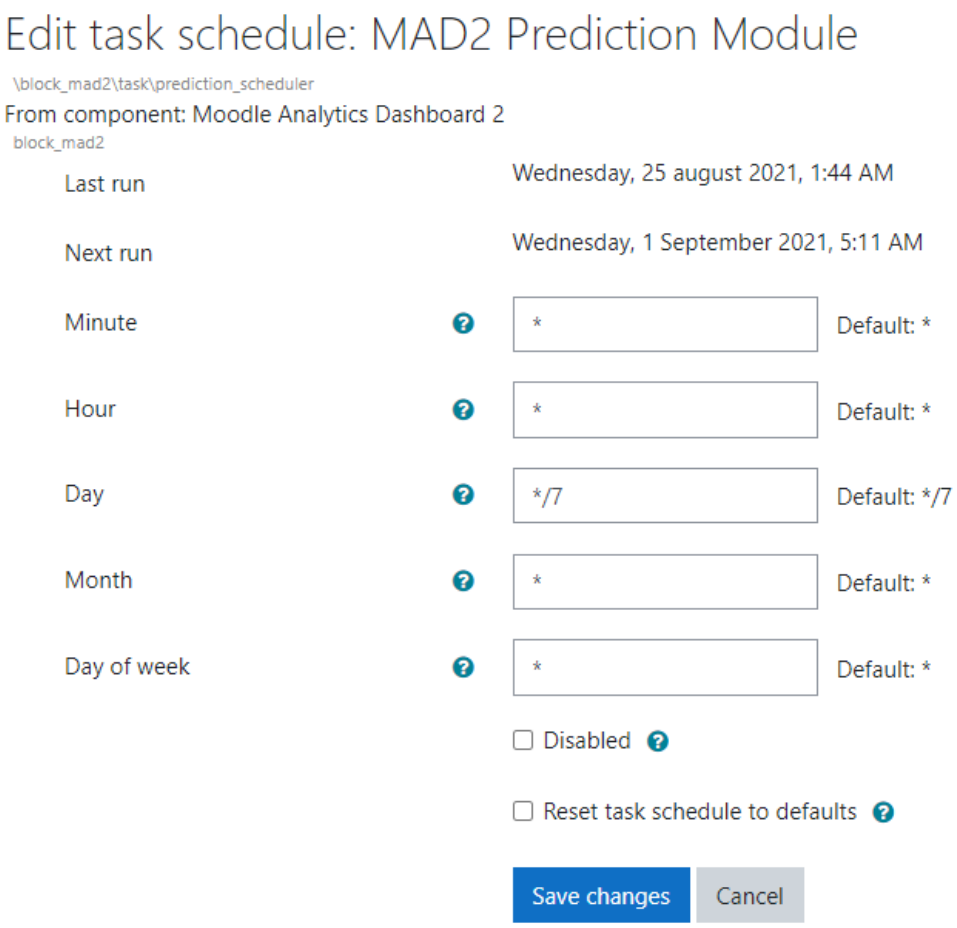
Fonte: Figura do autor (2021)

anterior como base e preencher a nota mínima de aprovação. As datas de início e fim são preenchidas automaticamente com base na disciplina selecionada, a quantidade de semanas indica o tamanho do modelo, em outras palavras, a quantidade de semanas que o curso de treinamento tinha disponível para gerar o modelo.

Se o moodle foi configurado corretamente durante a sua instalação, a predição irá ocorrer automaticamente toda a semana. Esta periodicidade pode ser alterada nas configuração de tarefas agendadas do moodle (Figura 24). Porém dada a natureza semanal na qual o modelo foi treinado não é recomendado tentar realizar predições diárias, por

exemplo.

Figura 24 – Menu da predição



Fonte: Figura do autor (2021)

Os resultados das predições para cada aluno são encontrados na opção "Estudantes"do MAD e são representados por um ponto onde a cor verde se refere a "Sem risco"e a cor vermelha se refere a "Em risco"(Figura 25).

Figura 25 – Resultado da predição

Fonte: Figura do autor (2021)

6 CONSIDERAÇÕES FINAIS

O trabalho complementa o *plugin* MAD adicionando a funcionalidade de predição de uma maneira intuitiva e de fácil usabilidade. O módulo preditivo em conjunto das funcionalidades do *plugin* permite que docentes acompanhem a trajetória acadêmica dos estudantes e identifiquem quais estão em risco de reprovação ou evasão.

Um maior entendimento sobre o efeito que diversos aspectos da vida acadêmica possuem sobre os alunos pode trazer diversos benefícios para o aluno. A identificação precoce permite a tomada de uma ação para que o estudante possa se recuperar e evitar uma reprovação na disciplina. As áreas que causam maiores dificuldades podem ser revisadas no planejamento de aula, melhorando a qualidade de ensino em ambos curto e longo prazo.

6.1 TRABALHOS FUTUROS

Após a criação do produto viável mínimo que foi colocado em testes, surgiram alguns tópicos de melhorias que podem ser feitas ao módulo preditivo existente.

O primeiro ponto é o desacoplamento do algoritmo em *python* que é utilizado nas fases de pré-processamento, treinamento e predição para uma API própria. Assim é aproveitada todas as partes de comunicação entre o Moodle e o MAD que foram desenvolvidas, mas o servidor do Moodle não vai mais possuir o custo computacional do módulo preditivo.

Em segundo lugar, uma vez que o módulo preditivo existe isolado na forma de uma API, novas alternativas de modelos preditivos treinados com outros algoritmos podem ser criados. Utilizando outros dados ou implementando técnicas completamente diferentes da que existe hoje.

É preciso adicionar uma forma de informar a qualidade do modelo treinado para o usuário.

Por último, o módulo preditivo pode ser ofertado como um produto a parte. já que existirá um custo oriundo da decisão de desacoplar o módulo em uma API. O usuário poderá decidir se possui interesse ou não em adquirir esta funcionalidade.

REFERÊNCIAS

ADNAN, Muhammad *et al.* Predicting at-Risk Students at Different Percentages of Course Length for Early Intervention Using Machine Learning Models. **IEEE Access**, v. 9, p. 7519–7539, 2021. DOI: 10.1109/ACCESS.2021.3049446.

ARROYO-BARRIGÜETE, Jose Luis *et al.* Portability of Predictive Academic Performance Models: An Empirical Sensitivity Analysis. **Mathematics**, v. 9, n. 8, 2021. ISSN 2227-7390. DOI: 10.3390/math9080870. Disponível em: https://www.mdpi.com/2227-7390/9/8/870.

BAKER, Ryan; INVENTADO, Paul. Educational Data Mining and Learning Analytics. *In:* [*S.l.*: *s.n.*], mai. 2014. P. 61–75. ISBN 978-1-4614-3304-0. DOI: 10.1007/978-1-4614-3305-7_4.

BOYLAN-ASHRAF, Peggy C.; HAUGHERY, John R. Failure Rates in Engineering: Does It Have to Do with Class Size? *In:*_____. Salt Lake City, UT. **2018 ASEE Annual Conference & Exposition**. [*S.l.*]: ASEE, jun. 2018.

CORTEZ, Paulo; SILVA, Alice. Using data mining to predict secondary school student performance. **EUROSIS**, jan. 2008.

DISTANTE, Damiano *et al.* MILA: A SCORM-Compliant Interactive Learning Analytics Tool for Moodle. *In:* 2020 IEEE 20th International Conference on Advanced Learning Technologies (ICALT). [*S.l.*: *s.n.*], 2020. P. 169–171. DOI: 10.1109/ICALT49669.2020.00056.

_____._____. P. 169–171, 2020. DOI: 10.1109/ICALT49669.2020.00056.

DONDORF, Thomas *et al.* LEARNING ANALYTICS SOFTWARE IMPLEMENTATION FOR THE MOODLE LEARNING MANAGEMENT SYSTEM. *In:* p. 6957–6964. DOI: 10.21125/iceri.2019.1655.

FREITAS DOS SANTOS, Mateus de. **Implementação front-end de um bloco moodle para predição de acadêmicos em risco**. 2021. F. 58. Tecnologias da Informação e Comunicação, Universidade Federal de Santa Catarina, São Paulo.

HAN, Jiawei; KAMBER, Micheline; PEI, Jian. **Data mining: concepts and techniques**. [*S.l.*]: Morgan Kaufmann Publishers, 2011.

KONSTANTINIDIS, Andreas; GRAFTON, Cat. Using Excel macros to analyse Moodle logs, 2013.

KOTSIANTIS, Sotiris; PIERRAKEAS, Christos; PINTELAS, P. PREDICTING STUDENTS'PERFORMANCE IN DISTANCE LEARNING USING MACHINE

LEARNING TECHNIQUES. **Applied Artificial Intelligence**, v. 18, p. 411–426, jan. 2004.

KOUTCHEME, Charles *et al.* Methodological Considerations for Predicting At-Risk Students. *In:* AUSTRALASIAN Computing Education Conference. Virtual Event, Australia: Association for Computing Machinery, 2022. (ACE '22), p. 105–113. DOI: 10.1145/3511861.3511873. Disponível em: https://doi.org/10.1145/3511861.3511873.

MACARINI, Luiz *et al.* Predicting Students Success in Blended Learning-Evaluating Different Interactions Inside Learning Management Systems. **Applied Sciences**, v. 9, p. 5523, dez. 2019. DOI: 10.3390/app9245523.

MADNAIK, Sandeep Subhash. Predicting students' performance by learning analytics, 2020.

PEÑA-AYALA, Alejandro. Educational data mining: A survey and a data mining-based analysis of recent works. **Expert Systems with Applications**, v. 41, 4, Part 1, p. 1432–1462, 2014. ISSN 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2013.08.042. Disponível em: https://www.sciencedirect.com/science/article/pii/S0957417413006635.

RAMOS, Vinicius *et al.* A Comparação da Realidade Mundial do Ensino de Programação para Iniciantes com a Realidade Nacional: Revisão sistemática da literatura em eventos brasileiros. **Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação - SBIE)**, v. 26, n. 1, p. 318, 2015. ISSN 2316-6533. DOI: 10.5753/cbie.sbie.2015.318. Disponível em: http://br-ie.org/pub/index.php/sbie/article/view/5178.

ROMERO, Cristóbal; VENTURA, Sebastian. Educational Data Mining: A Review of the State of the Art. **Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on**, v. 40, p. 601–618, dez. 2010. DOI: 10.1109/TSMCC.2010.2053532.

SÁIZ-MANZANARES *et al.* Monitoring Students at the University: Design and Application of a Moodle Plugin. **Applied Sciences**, v. 10, n. 10, 2020. ISSN 2076-3417. DOI: 10.3390/app10103469. Disponível em: https://www.mdpi.com/2076-3417/10/10/3469.

SAMUEL, A. L. Some Studies in Machine Learning Using the Game of Checkers. **IBM Journal of Research and Development**, v. 3, n. 3, p. 210–229, 1959. DOI: 10.1147/rd.33.0210.

SANTOS, Jose *et al.* Estudo comparativo de plugins Moodle para Análise e Acompanhamento da Aprendizagem. *In:* p. 189. DOI: 10.5753/cbie.sbie.2019.189.

WALKER, Jonathan S. **Machine Learning for Beginners: Your Ultimate Guide To Machine Learning For Absolute Beginners, Machine Learning Guide, Scikit-Learn, Deep Learning, TensorFlow, Data Analytics, Python, Data Science**. North Charleston, SC, USA: CreateSpace Independent Publishing Platform, 2018. ISBN 1973962853.

WATSON, Christopher; LI, Frederick W.B. Failure Rates in Introductory Programming Revisited. *In:* PROCEEDINGS of the 2014 Conference on Innovation & Technology in Computer Science Education. Uppsala, Sweden: Association for Computing Machinery, 2014. (ITiCSE '14), p. 39–44. DOI: 10.1145/2591708.2591749. Disponível em: https://doi.org/10.1145/2591708.2591749.

ZAPPAROLLI, Luciana; STIUBIENER, Itana. FAG – a management support tool with BI techniques to assist teachers in the virtual learning environment Moodle. **Advances in Science, Technology and Engineering Systems Journal**, v. 2, p. 587–597, jun. 2017. DOI: 10.25046/aj020375.

ZHU, Ji *et al.* Multi-class AdaBoost. **Statistics and its interface**, v. 2, fev. 2006. DOI: 10.4310/SII.2009.v2.n3.a8.