UFRR

UNIVERSIDADE FEDERAL DE RORAIMA

PRÓ-REITORIA DE ENSINO E EXTENSÃO

DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

GENE CHARLES LIMA AGUIAR

UMA ABORDAGEM DATA-DRIVEN PARA PREDIÇÃO PRECOCE DA EVASÃO EM TURMAS DE PROGRAMAÇÃO QUE UTILIZAM JUÍZES ONLINE

Boa Vista - RR

2018

GENE CHARLES LIMA AGUIAR

UMA ABORDAGEM DATA-DRIVEN PARA PREDIÇÃO PRECOCE DA EVASÃO EM TURMAS DE PROGRAMAÇÃO QUE UTILIZAM JUÍZES ONLINE

Monografia de Graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Roraima como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Msc. Filipe Dwan Pereira

Boa Vista - RR

2018

GENE CHARLES LIMA AGUIAR

UMA ABORDAGEM DATA-DRIVEN PARA PREDIÇÃO PRECOCE DA EVASÃO EM TURMAS DE PROGRAMAÇÃO QUE UTILIZAM JUÍZES ONLINE

Monografia de Graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Roraima como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação.
Defendido em 12 de julho de 2018 e aprovado pela seguinte banca examinadora:

Prof. Msc. Filipe Dwan Pereira
Orientador / Curso de Ciência da Computação - UFRR

Prof. Dr. Luciano Ferreira Silva
Curso de Ciência da Computação - UFRR

Prof. Dr. Herbert Oliveira Rocha
Curso de Ciência da Computação - UFRR

Este trabalho é dedicado a toda minha família, por não medirem esforços em me apoiar em todos os momentos para chegar até esta etapa da minha vida.

AGRADECIMENTOS

Agradeço em primeiro lugar a Deus que iluminou o meu caminho, me guardou e me deu força durante esta caminhada.

Aos meus discipuladores Peres e Edilberto que com exemplo de vida me apoiaram em sempre seguir em frente e não desistir.

Agradeço também a todos os professores que me acompanharam durante a graduação, em especial ao Prof. Msc Filipe Dwan, por sempre estar dando seu máxima para me auxiliar, aconselhar e incentivar na realização deste trabalho.

Aos professores Luciano Ferreira Silva e Herbert Oliveira Rocha, por participarem da minha banca no trabalho de conclusão de curso.

"O temor do SENHOR é o princípio do saber,
mas os loucos desprezam a sabedoria e o ensino.
(Bíblia Sagrada, Provérbios 1, 4)

RESUMO

Muitas instituições de ensino superior vêm utilizando Juízes *Online* em turmas de Introdução à Programação de Computadores (IPC) para que haja um *feedback* mais rápido para os alunos e para diminuir a carga de trabalho do professor. Entretanto, mesmo com o uso dessa estratégia ainda existe um alto índice de evasão nas turmas de IPC. Neste sentido, muitas pesquisas vêm sendo conduzidas na área de *Learn Analitcs*, as quais usam os dados do processo de desenvolvimento do aluno em ambientes *online* e técnicas de aprendizagem de máquina e mineração de dados a fim de melhorar o processo de ensino e aprendizagem nessas turmas. Diante disso, o objetivo deste trabalho é desenvolver e validar um método para predição de evasão de alunos de turmas de IPC que usam Juízes *Online*, utilizando técnicas de aprendizagem de máquina. O modelo preditivo se mostrou satisfatório uma vez que sua acurácia alcançou um resultado de 80% usando um banco de dados balanceado e 88,63% em um banco de dados desbalanceado já nas primeiras duas semanas de aula na classificação se o aluno tem tendência a evadir ou não.

Palavras-chaves: Predição de Evasão, Aprendizagem de máquina, Análise de Aprendizagem, Juízes *Online*, Introdução à Programação de Computadores.

ABSTRACT

Many education institutions have been using Programming Online Judges (POJ) in CS1 classes to provide faster feedback for students and to reduce teacher workload. However, even with the use of this strategy, there is still a high level of avoidance in the CS1 classes. Research has been conducted in the area of Learn Analytics, which uses data from the student development process in online environments and Machine Learning (ML) techniques to improve the teaching and learning process in these classes. In this sense, the goal of this work is to develop and validate a method for predicting the student dropout rate from CS1 classes, which uses POJ. To do so, we employed the algorithm J48 and ensembles based on decision tree such as Random Forest. As a result, we achieved an accuracy of 80% using a balanced database and 88,63% in an unbalanced database. It is worth mentioning that the predictive model obtained this outcome being trained and tested with the data from the very first two weeks of CS1 classes.

Key-words: Prediction of Evasion, Machine Learning, Learning Analysis, Judges Online, Introduction to Computer Programming.

LISTA DE ILUSTRAÇÕES

Figura 1 – Fórmula da Entropia 19
Figura 2 – Tipos de Aprendizagem de Máquina 20
Figura 3 – Árvore de Decisão 21
Figura 4 – Algoritmo de aprendizagem de Árvore de Decisão 23
Figura 5 – Rede Bayesiana de desistência proposta por especialista 31
Figura 6 – Gráfico com a média da precisão dos algoritmos 33
Figura 7 – Metodologia CRISP-DM adaptada para Evasão 35
Figura 8 – Arquitetura do Método Proposto 39
Figura 9 – Classificação de dados supervisionados 42
Figura 10 – Funcionamento das Métricas no Modelo Preditivo 43
Figura 11 – Árvore Sessão1 desbalanceada com método Cross-validation 47
Figura 12 – Árvore de Decisão no scikit-learn Balanceada 51
Figura 13 – Nodo da Árvore de Decisão 51
Figura 14 – Estrutura do Algoritmo no Python 52
Figura 15 – Parâmetros do método *GridSearch* 53
Figura 16 – Método *GridSearchCV* 53

LISTA DE TABELAS

Tabela 1 – Matriz de Coincidência 24
Tabela 2 – Apresentação da acurácia, recall dos positivos, precision dos positivos, recall dos negativos, precision dos negativos dos algoritmos de aprendizagem de máquina com melhores resultados (em negrito) 28
Tabela 3 – Avaliação dos Modelos Preditivos mais Precisos 28
Tabela 4 – Resultado do experimento 1 29
Tabela 5 – Resultado do experimento 2 29
Tabela 6 – Resultado do experimento 3 30
Tabela 7 – Resultados dos experimentos da Rede Bayesiana 32
Tabela 8 – Precisão para classificação de reprovado - sem discriminar o tipo de atributos 34
Tabela 9 – Resumo dos trabalhos relacionados 37
Tabela 10 – Período da disciplina 40
Tabela 11 – Modificações na Classe da Base de Dados 44
Tabela 12 – Resultados Sessão1 prevendo Sessão2 não-balanceada 45
Tabela 13 – Resultados Sessão1 não-balanceado 46
Tabela 14 – Resultados Matriz de Coincidência Sessão1 não-balanceado 47
Tabela 15 – Resultados Entre Sessões balanceadas 48
Tabela 16 – Resultados de algoritmos baseados em árvore de decisão 49
Tabela 17 – Parâmetros da Árvore de Decisão 50

LISTA DE ABREVIATURAS E SIGLAS

AB	AdaBoosting
AD	Árvore de Decisão
AM	Aprendizagem de Máquina
APE	Assistente de Previsão da Evasão
AVA	Ambientes Virtual de Aprendizagem
BN	BayesNet
CLEC	Curso de Licenciatura em Educação do Campo
CLPD	Curso de Licenciatura em Pedagogia
CRISP-DM	CRoss Industry Standard Process for Data Mining
DT	DecisionTable
FN	Falso Negativo
FP	Falso Positivo
INEP	Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira
IPC	Introdução à Programação de Computadores
JR	JRip
KDD	Knowledge Discovery in Database
KNN	K-Nearest Neighbours
LA	Learning Analytics
MP	MultilayerPerceptron
NB	NaiveBayes
OR	OneR
RF	RandomForest
SC	SimpleCart

SL	SimpleLogistic
STI	Sistema Tutor Inteligente
SVM	Support Vector Machine
UFAM	Universidade Federal do Amazonas
UFPEL	Universidade Federal de Pelotas
VN	Verdadeiro Negativo
VP	Verdadeiro Positivo
ZD	Zona de Dificuldade
ZE	Zona de Expertise

SUMÁRIO

1 INTRODUÇÃO 13
1.1 Objetivo geral 14
1.2 Objetivos Específicos 14
1.3 Organização do trabalho 14

2 FUNDAMENTAÇÃO TEÓRICA 16
2.1 Metodologia de aprendizagem híbrida 16
2.2 Learning Analytics 17
2.3 Técnicas de Pré-processamento 17
2.3.1 Atributos 18
2.3.2 Técnicas de Seleção de Atributos 18
2.3.2.1 Removendo atributos com baixa variância 19
2.3.2.2 Algoritmo Ganho de Informação 19
2.4 Aprendizagem de máquina Supervisionada 20
2.4.1 Árvore de Decisão 21
2.4.2 Floresta Aleatória 23
2.4.3 Técnicas de Avaliação do Modelo Preditivo (acurácia, recall, precision) 24
2.4.4 Ética e privacidade dos dados 25
2.4.4.1 Privacidade 25
2.4.4.2 Ética 25
2.5 Planejamento e Projeto dos Experimentos 26

3 TRABALHOS RELACIONADOS 27
3.1 Predição de Zona de Aprendizagem de Alunos de Introdução à Programação em Ambientes de Correção Automática de Código 27
3.2 Previsão de Estudantes com Risco de Evasão Utilizando Técnicas de Mineração de Dados 28
3.3 Um Assistente de Predição de Evasão aplicado a uma disciplina Introdutória do curso de Ciência da Computação 30
3.4 Predição de desempenho de alunos do primeiro período baseado nas notas de ingresso utilizando métodos de aprendizagem de máquina 32
3.5 Modelagem e Predição de Reprovação de Acadêmicos de Cursos de Educação a Distância a partir da Contagem de Interações 33

3.6 UMA VISÃO DO FUTURO: PREVISÃO DE EVASÃO EM CURSOS DE GRADUAÇÃO PRESENCIAIS DE UNIVERSIDADES PÚBLICAS: O CASO DO CURSO DE ZOOTECNIA 35
3.7 Considerações dos Trabalhos Relacionados 35

4 MÉTODO PROPOSTO 38
4.1 Arquitetura 38
4.2 Descrição da Base de Dados 39
4.3 Construção do Modelo Preditivo 42
4.4 Tecnologias Utilizadas 44

5 RESULTADOS EXPERIMENTAIS 45
5.1 Execução do Experimento I 45
5.2 Execução do Experimento II 47
5.3 Execução do Experimento III 48
5.4 Execução do Experimento IV 49
5.5 Detalhes de Implementação 52

6 CONSIDERAÇÕES FINAIS 55
6.1 Limitações e ameaças à validade 55
6.2 Trabalhos Futuros 56

REFERÊNCIAS 57

1 INTRODUÇÃO

O problema de evasão e retenção nos cursos de graduação das universidades públicas brasileiras assola os governos e inúmeras instituições (MANHÃES et al., 2011), conforme censo da Educação de Ensino Superior realizado pelo Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (INEP) (SOUZA et al., 2016). Diante dessa situação o Governo Federal estipulou uma meta para elevação da taxa de conclusão dos cursos de graduação presenciais para noventa por cento (MANHÃES et al., 2011).

A disciplina de Introdução à Programação de Computadores (IPC) é ministrada por dezenas de cursos em diversas universidades, uma vez que a programação é uma habilidade necessária para diversas competências imprescindíveis para o mercado (ALVES, 2012). Entretanto essas disciplinas possuem um alto índice de evasão (ROCHA et al., 2010).

Uma maneira de lidar com esse problema de forma parcial é utilizando o avanço da tecnologia na educação. Para exemplificar, muitas instituições vêm utilizando a metodologia de aprendizagem híbrida (BACICH et al., 2015), que utiliza sistemas *online* juntamente com o ensino tradicional. Os Juízes *Online* são um exemplo da aplicação desse método e cada vez mais as instituições de ensino vêm implantando esses sistemas de correção automática nas universidades, visto que ameniza a carga de trabalho do professor e gera um *feedback* mais rápido para o aluno (PAES et al., 2013).

Os Juízes *Online* apresentam maior flexibilidade temporal e espacial aos estudantes, entretanto as taxas de abandono das disciplinas de IPC ainda é um problema estudado por vários pesquisadores (MARTINS et al., 2012). Nesse sentido, várias pesquisas vêm sendo conduzidas na área de *learning analytics*, utilizando dados coletados à medida que eles vão resolvendo os exercícios nesses ambientes de correção automática de código. Tais dados são utilizados juntamente com técnicas de aprendizagem de máquina e mineração de dados para melhorar o processo de ensino e aprendizagem nessas disciplinas.

Nesse mesmo sentido, a presente pesquisa pré-processou esses registros provindos do processo de codificação dos alunos ao resolver os exercícios em Juízes *Online*, sendo usados como atributos do modelo preditivo e inferindo com antecedência se um aluno corre risco de evadir na disciplina ou não. Foram utilizados algoritmos baseados em árvores de decisão, especificamente o JR8 e a Floresta aleatória, uma vez que eles apresentaram os melhores resultados em estudos que foram conduzidos em cenários similares, como os de Manhães et al. (2011), Detoni et al. (2015), Martins et al. (2012), Brito et al. (2014) e KANTORSKI et al. (2015).

Como resultado, o modelo preditivo construído obteve uma acurácia de 80% em uma

base de dados balanceada e 88,63% em uma base de dados não balanceada utilizando os dados das duas primeiras semanas de aula. Isso significa que com o emprego da abordagem apresentada neste estudo, o professor e a instituição de ensino têm a informação, ainda no início da disciplina, se o aluno pode evadir ou não. Além disso, houve uma redução de dimensionalidade dos atributos de 22 para 5 atributos. Isso significa que o método torna-se mais simples, mantendo uma acurácia desejável. Por fim, percebeu-se que alunos menos aplicados nas listas de exercícios, que acessam poucas vezes o juiz online e que codificam em uma frequência mais baixa, têm uma tendência para evasão.

Frisa-se que saber antecipadamente se o aluno irá evadir pode ser útil por diversos fatores, como por exemplo: a) o aluno pode ser monitorado para não evadir; b) proporcionar medidas pedagógicas que reduza essa ocorrência; c) ajudar o aluno a descobrir sua vocação e outros.

1.1 Objetivo geral

Propor e validar um método para predição precoce da evasão de alunos da turmas de IPC que usam Juízes *Online*, utilizando algoritmos baseados em árvores de decisão.

1.2 Objetivos Específicos

1. Caracterizar e selecionar estratégias de predição de evasão presentes na literatura;
2. Pré-processar os registros do processo de resolução de exercícios dos alunos, a fim de gerar valores numéricos que podem ser usados como atributos de algoritmos de aprendizagem de máquina;
3. Implementar métodos baseados em árvores de decisão para a predição precoce da evasão de alunos de IPC;
4. Comparar os algoritmos com o intuito de determinar o mais acurado para resolver o problema;
5. Validar os resultados dos modelos preditivos utilizando métricas de avaliação.

1.3 Organização do trabalho

O trabalho está organizado da seguinte forma: na seção 2 apresenta a fundamentação teórica do trabalho, descrevendo sobre o ensino híbrido, análise de aprendizagem, técnicas de pré-processamento, etapa sobre os atributos, aprendizagem de máquina supervisionada, algoritmo árvore de decisão, algoritmo floresta aleatória, métricas de avaliação do modelo preditivo, privacidade dos dados, ética e planejamento dos experimentos; a seção 3 apresenta os

trabalhos relacionados. A seção 4 apresenta o método proposto bem como a arquitetura proposta, a descrição da base de dados, a construção do modelo preditivo e a tecnologia utilizada. A Seção 5 apresenta os resultados experimentais e detalhe da implementação; por fim, a Seção 6 apresenta as considerações finais, as ameaças à validade e trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo serão abordadas as bases para o desenvolvimento do Modelo Preditivo de Evasão com objetivo de trazer clareza dos principais pontos, abordando de forma introdutória e objetiva em cada etapa.

2.1 Metodologia de aprendizagem híbrida

A metodologia de aprendizagem híbrida é um modelo educacional no qual o aluno aprende em parte por meio *online* com algum elemento de controle do discente sobre o tempo, lugar, modo e/ou ritmo de estudo, e em parte numa localidade física supervisionada fora de sua residência (BACICH et al., 2015).

Segundo Galvão et al. (2016), as atividades baseadas em Juízes *Online* são apoiadas no ensino híbrido, possibilitando ao aluno praticar mais exercícios de programação, com *feedback* imediato e proporcionando um aumento da taxa de aprovação no curso. Os Juízes *Online* são sistemas muitas vezes utilizados em disciplinas de IPC, uma vez que automatiza o processo de correção de atividades de programação dos alunos.

Em geral, os Juízes *Online* são equipados com funcionalidade de compilação, execução e teste de código-fonte com base de dados internos para arbitrariamente informar se está correto ou não, dando uma resposta com base na comparação com os dados submetidos pelo aluno (CHAVES et al., 2013).

Na internet podemos encontrar alguns Juízes *Online* (CHAVES et al., 2013), geralmente direcionado a competição de programação, embora sua utilização seja administrada por docentes em diversas universidades. Abaixo citamos alguns Juízes *Online* disponíveis na internet:

- Timus Online Judge: Com dados de exercícios Russos, a maioria dos problemas são recolhidos de concursos realizados na Ural Federal University, Ural Championships, Ural ACM Concursos ICPC Subregional, e Petrozavodsk Training Camps. Sua home page está em <http://acm.timus.ru/>;
- SPOJ Brasil: com objetivo de disponibilizar problemas regionais, de seletivas e de olimpíadas de programação em português. Suas atividades suportam várias linguagens, dentre elas C++, ADA95 e JAVA. Podendo ser encontrada na URL: <http://br.spoj.com/embed/info>;
- URI Online Judge: Disponível em português, inglês e espanhol seu principal objetivo é promover a prática de programação e o compartilhamento de programação. As ati-

vidades são submetidas na linguagem C, C++, Java ou Python. Sua URL é <https://www.urionlinejudge.com.br/judge/pt/login>;

- CodeBench é um Juiz *Online* desenvolvido pelo instituto de computação da Universidade Federal do Amazonas-UFAM, as atividades podem ser codificadas na linguagem Python, C e java. Seu acesso está disponível com a URL <http://codebench.icomp.ufam.edu.br/index.php?r=site%2Flogin>.

2.2 Learning Analytics

De acordo com Blikstein (2011), a análise de aprendizagem, em inglês *LEARNING ANALYTICS (LA)*, é uma técnica automatizada de examinar, avaliar e visualizar como um aluno desenvolve a aprendizagem em ambientes educacionais, nos quais o método de ensino tradicional[1] não pode compreender o progresso dos alunos automaticamente.

Conforme Chatti et al. (2012), LA busca encontrar padrões com possíveis soluções e respostas através de pesquisa em dados educacionais afim de apoiar o ensino educacional. Sua área é multi-disciplinar e envolve aprendizagem de máquina, inteligência artificial, recuperação de informação, estatísticas e visualização. As três principais etapas para realização do ciclo de LA são: coleta de dados e pré-processamento; análise para descobrir padrões ocultos e ação de análise de previsão, e; pós-processamento.

O uso da tecnologia com Juízes *Online* na educação possibilitou um aumento considerado de dados armazenado. Essa quantidade de dados é de grande utilidade na busca de respostas da Educação, podendo proporcionar benefícios na aprendizagem, um melhor acompanhamento do aluno, previsão de evasão e aperfeiçoando de práticas pedagógicas (RIGO et al., 2014).

Assim, na presente pesquisa foram realizadas as três etapas supracitadas para fazer a predição da evasão de alunos de IPC utilizando os dados que os alunos vão deixando ao resolver exercícios em Juízes *Online*.

2.3 Técnicas de Pré-processamento

A qualidade dos dados de entrada em aprendizagem de máquina é umas das principais preocupações, pois influenciam em diversos aspectos no desempenho do sistema de aprendizado na indução de conhecimento feitos pelos algoritmos. A presença de valores desconhecidos e a distribuição entre classes desbalanceadas são os aspectos que podem estar presentes na base de dados e que precisam ser tratados no pré-processamento. Saber quais providências devem ser tomadas ao localizar informações incompletas ou quando as fontes de informações estão indisponíveis é de fundamental importância (BATISTA et al., 2003).

[1] Ensino tradicional em sala de aula.

Já no tratamento, quando o conjunto de dados apresenta classes desbalanceadas, uma solução é balancear artificialmente o conjunto de dados, inserindo ou removendo classes menos confiáveis com uso de heurísticas. Para ilustrar, é possível criar instâncias utilizando uma heurística para balancear a base ou ainda remover aleatoriamente instâncias da classe majoritária, utilizando técnicas de subamostragem (BATISTA et al., 2003).

García et al. (2007) lista uma série de tarefas amplamente utilizadas no pré-processamento de dados:

- Discretização de dados - Esse parâmetro é de fundamental importância em aprendizado supervisionado, geralmente aplicado em algoritmo de classificação, consiste na transformação dos dados contínuos em forma de atributos categorizados.
- Seleção de atributo - somente um subconjunto de atributos essenciais são escolhidos, com intuito de reduzir a dimensão do banco de dados, reduzindo a complexidade e assim o tempo de processamento.
- Derivação de novos atributos - Com essa técnica, novos atributos são criados a partir dos existentes, relacionando-os entre si, podendo reduzir o conjunto de dados ou agregar com os atributos originais, com objetivo de facilitar a extração de informações mais eficaz.
- Transformar o formato de dados - transformando para o formato exigido pelos algoritmos ou estruturas de mineração de dados usados. Uma transformação por exemplo, é a normalização dos dados.

2.3.1 Atributos

Os atributos são de grande relevância no desempenho do sistema de classificação. A identificação dos atributos mais relevantes, realizado com a seleção, aumenta o êxito da classificação, pois nem sempre uma grande quantidade de dados é a melhor solução para sistema de aprendizado (BATISTA et al., 2003).

2.3.2 Técnicas de Seleção de Atributos

Técnicas de seleção de atributos são utilizadas para melhorar a precisão dos classificadores e aumentar seu desempenho. Seu objetivo é separar os atributos mais relevantes avaliando a importância das características, geralmente antes da aprendizagem real (pré-processamento), com o foco de evitar a imprecisão dos classificadores nas deduções de aprendizado supervisionado. Vários algoritmos têm sido apresentados na literatura (PEDREGOSA et al., 2011).

2.3.2.1 Removendo atributos com baixa variância

Uma técnica simples, mas bastante utilizada dentro da seleção de atributos é a remoção de atributo com baixa variância. Por exemplo, numa amostra em que os atributos booleanos têm 80% de repetição, ou seja, a maioria dos atributos possui o mesmo valor em toda as amostras, esse atributo será removido, reduzindo a dimensionalidade da amostra. No site scikit-learn.org, o algoritmo utilizado é o *feature_selection* importando a função *VarianceThreshold* para estipular a variância da distribuição de *Bernoulli* dada fórmula $Var[X] = p(1-p)$ (PEDREGOSA et al., 2011).

2.3.2.2 Algoritmo Ganho de Informação

Outra técnica comum de seleção de atributos é com a aplicação do o algoritmo Ganho de Informação que é utilizada para saber o custo da informação que cada atributo tem para contribuir na tarefa da classificação, usando para isso, a redução da entropia (ARTUSO, 2012). A entropia é uma métrica de incerteza sobre uma variável aleatória, sendo que o ganho de informação acontece na redução da entropia (GAMA, 2002).

Para uma melhor compreensão, imagine uma moeda com ambos os lados sendo coroa, ao ser jogada para cima o resultado sempre dará coroa, logo sua variável aleatória seria um único valor, e como não há incerteza, a entropia seria zero. Já uma moeda com cara e coroa, 0 ou 1, possui entropia igual a 1, uma vez que as chances de ser cara ou coroa são as mesmas e logo há um ambiente de absoluta incerteza e nenhum ganho de informação (NORVIG; RUSSELL, 2014).

Agora, uma moeda viciada em que 99% das vezes dá cara, se apostássemos em cara, somente em 1 por cento estaríamos errado, conforme podemos verificar na equação abaixo (NORVIG; RUSSELL, 2014). Em geral, a entropia de uma variável aleatória *V* com valores v_k cada um com probabilidade *P*(v_k), é calculada conforme a fórmula na Figura 1:

Figura 1 – Fórmula da Entropia

$$\textit{Entropia: } H(V) = \sum_k P(v_k) \log_2 \frac{1}{P(v_k)} = -\sum_k P(v_k) \log_2 P(v_k)$$

Fonte: (NORVIG; RUSSELL, 2014)

Para calcular a entropia do lançamento de uma moeda com cara e coroa (honesta) seria do tipo (NORVIG; RUSSELL, 2014):

$$E(honesta) = \breve{}(0,5.log_2 0,5 + 0,5.log_2 0,5) = 1 \tag{2.1}$$

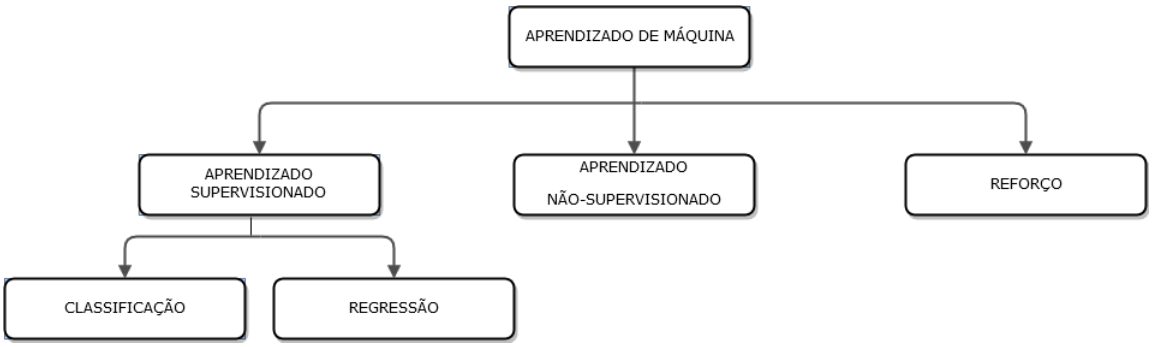
Com a moeda adulterada em que 99% dá cara, ficaria:

$$E(adulterada) = \breve{}(0,99log_20,99 + 0,01log_20,01) \approx 0.08 \tag{2.2}$$

2.4 Aprendizagem de máquina Supervisionada

Aprendizagem de máquina (AM) é subdivida em três áreas: aprendizado supervisionado, aprendizado não supervisionado e aprendizado por reforço conforme a Figura 2. Dependendo do problema a ser resolvido e do tipo dos dados, cada área possui suas particularidades. Neste trabalho, iremos abordar apenas o aprendizado supervisionado, especificamente a classificação (MONARD; BARANAUSKAS, 2003).

Figura 2 – Tipos de Aprendizagem de Máquina



Fonte: (MONARD; BARANAUSKAS, 2003)

Aprendizado de Máquina supervisionada é uma sub área da AM, na qual são desenvolvidos sistemas capazes de tomar decisões, baseando seus resultados nos exemplos de treinamento ao qual foi submetido (BATISTA; MONARD, 2004). Para exemplificar, em aprendizagem supervisionado é dado um conjunto de exemplos de treinamento, em que o rótulo da classe associada é conhecido, ou seja, é fornecida uma classe a qual cada exemplo no treinamento pertence, daí o nome supervisionado. O algoritmo deve aprender em uma fração da base de dados, chamada de treino, e ser avaliado em outra parte da base de dados chamada de teste (MONARD; BARANAUSKAS, 2003).

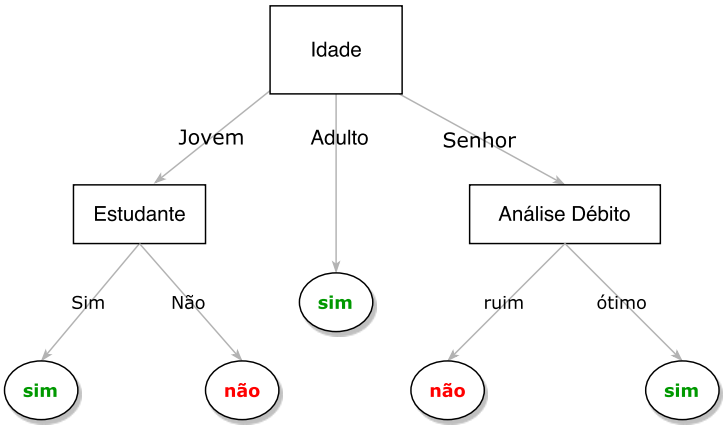
Apesar da AM ser uma ferramenta de grande influência para adquirir conhecimento automático, não existe um algoritmo exclusivo, capaz de resolver todos os problema com maior desempenho sobre outros. Cada algoritmo possui limites e firmeza nos mais variados problemas a ele submetido (MONARD; BARANAUSKAS, 2003). Assim, nas próximas subseções serão apresentados os algoritmos que serão utilizados nos experimentos, sendo o primeiro árvore de decisão e o segundo floresta aleatória, que é um conjunto de árvores de decisões. Geralmente, esses algoritmos apresentam bons resultados, conforme será apresentado no capítulo dos trabalhos relacionados.

2.4.1 Árvore de Decisão

A Árvore de Decisão (AD) é um método tradicional de aprendizagem supervisionado usado para problemas de classificação e regressão [2]. A AD tem a função de tomar um vetor de valores de atributos como entrada e retornar uma decisão, isto é,um único valor de saída. Como o foco deste trabalho é a classificação, vamos nos concentrar em problemas que dada uma entrada, a saída têm exatamente dois valores possíveis 1 (positivo) ou 0 (negativo), representando os alunos evadiram e os que não evadiram, respectivamente (NORVIG; RUSSELL, 2014).

Para ilustrar de forma simples, a Figura 3 apresenta uma hipótese [3] de verificar a liberação de empréstimo por uma pequena agência. Cada nó interno na árvore corresponde a um teste do valor de um dos atributos de entrada, e as ramificações dos nós são os possíveis valores do atributo. Cada nó folha na árvore classifica o valor a ser retornado pela função como resposta, se vai haver a liberação do empréstimo ou não (NORVIG; RUSSELL, 2014).

Figura 3 – Árvore de Decisão



Fonte: Própria

Existem várias versões do algoritmo de AD, sendo as principais: C4.5, C5.0, J48 e o CART (DAYCHOUM, 2013). Tais algoritmos utilizam uma estratégia gulosa de dividir para conquistar, o primeiro passo para construção da árvore é testando dentre todos os atributos o com maior importância para a classificação correta, a fim de defini-lo como nodo raiz da árvore. Uma forma de realizar o teste é calcular em cada atributo a entropia, isto é, mensurar a incerteza e impureza de cada atributo (DWAN et al., 2017), conforme detalhado na subseção 2.3.2.2 deste trabalho.

Esse teste divide o problema em sub-problemas menores que podem então ser resolvidas de forma recursiva, isto é, inúmeras vezes com a mesma regra. Onde todos os caminhos na árvore

[2] Onde Classificação trabalha com valores discretos e Regressão com valores contínuos.

[3] Em AM um modelo preditivo é uma função chamada de hipótese já que o modelo aprende com os exemplos e gera uma aproximação de um possível resultado para um exemplo ainda não classificado.

serão curtos e a árvore como um todo será pouco profunda. Em geral, depois que o primeiro teste de atributo separar os exemplos, cada resultado será um novo problema de aprendizagem de AD em si, com menos exemplos e um atributo a menos. Existem quatro casos a considerar para esses problemas recursivos (NORVIG; RUSSELL, 2014):

1. Se todos os exemplos restantes forem positivos (ou todos negativos), então terminamos, podemos responder Sim ou Não.

2. Se existem alguns exemplos positivos e alguns negativos, escolha o melhor atributo para dividi-los.

3. Se não resta nenhum exemplo, isso significa que nenhum exemplo desse tipo foi observado, e retornamos um valor-padrão calculado a partir da classificação de maioria no pai do nó.

4. Se não resta nenhum atributo, mas há exemplos positivos e negativos, esses exemplos têm exatamente a mesma descrição, mas classificações diferentes. Isso acontece quando alguns dados estão incorretos (dizemos nesse caso que existe ruído nos dados) e também quando o domínio é não determinístico, ou ainda porque não podemos observar um atributo que distinguiria os exemplos. O melhor que podemos fazer é voltar à classificação da maioria dos exemplos remanescentes.

A Figura 4 mostra o algoritmo de aprendizagem em AD. Apesar do conjunto de exemplos serem de fundamental importância para a construção da árvore, os exemplos não aparecem em nenhum lugar na árvore. A árvore é constituída com testes em atributos no interior dos nós, valores de atributos nas ramificações e valores de saída nos nós folha. Sua estrutura é composta de uma raiz, onde é feito o questionamento principal, ao depender da sua resposta, leva-se a um nó subsequente, que por si, poderá levar a um valor de saída ou a outro questionamento, temos assim, uma estrutura de dados com uma AD.

Vale informar que este trabalho selecionou o algoritmo Árvore de Decisão para o desenvolvimento do modelo preditivo, porque este algoritmo proporciona uma alta precisão e uma possibilidade de interpretação das relações existentes nos dados. Isto é, é possível realizar a visualização da árvore e analisar quais fatores estão levando os alunos a evasão ou a persistência na disciplina. Perceba que outros métodos de aprendizagem de máquina podem apresentar, bem como, alta precisão, mas muitos deles funcionam como uma caixa preta.

Figura 4 – Algoritmo de aprendizagem de Árvore de Decisão

função APRENDIZAGEM-EM-ÁRVORE-DE-DECISÃO(*exemplos, atributos, exemplos-pais)*
retorna uma árvore de decisão
  se exemplos é vazio **então retornar** VALOR-DA-MAIORIA (*exemplos_pais*)
  senão se todos os *exemplos* têm a mesma classificação **então retornar** a classificação
  senão se *atributos* é vazio **então retornar** VALOR-DA-MAIORIA(*exemplos*)
  senão
    A ← argmax $_{a \in atributes}$IMPORTÂNCIA (*a, exemplos*)
    árvore ← uma nova árvore de decisão com teste de raiz *A*
    para cada valor v_k de *A* **faça**
      exs ← $\{e : e \in exemplos$ **e** $e.A = v_k\}$
      subárvore ← APRENDIZAGEM-EM-ÁRVORE-DE-DECISÃO (*exs, atributos* — *A, exemplos*)
      adicionar uma ramificação à árvore com rótulo ($A = v_k$) e subárvore *subárvore*
    retornar *árvore*

Fonte: (NORVIG; RUSSELL, 2014)

2.4.2 Floresta Aleatória

Floresta Aleatória é um algoritmo de AM utilizada para a classificação, composto por um conjunto de árvores de decisão como estimadores base. Essa é uma técnica de ensemble em que cada AD é construída a partir de um conjunto de vetores aleatórios, gerados de maneira independente. O conjunto de vetores pode ser obtido através de seleção aleatória de atributos, seleção aleatória de amostras ou variações aleatória de parâmetro de árvores (BREIMAN, 2001).

Segundo Breiman (2001), as variantes mais utilizadas são a seleção aleatória de amostras de treinamento e seleção de características aleatórias para cada nó da árvore (BREIMAN, 2001).

A floresta aleatória utiliza a técnica de votação entre os estimadores base, isto é, a partir da classificação das várias árvores de decisão que compõem a floresta é decidido por votação qual a melhor estimativa. Essa junção de várias árvores de decisão fazendo esse processo de Ranking das classificações controla o *over-fitting*[4]. Conforme Pedregosa et al. (2011), a floresta aleatória se adapta a uma série de preditores das árvores de decisão em várias sub-amostras, esse algoritmo disponibiliza alguns hiperparâmetros que podem ser alterados para melhor ajuste aos dados, os principais são (PEDREGOSA et al., 2011):

- n_estimador: Definição da quantidade de árvores na floresta;
- max_atributo: É o número de características a serem consideradas ao procurar a melhor divisão;
- max_depth: essa variável define a profundidade máxima das árvores;

[4] um super ajuste no aprendizado do algoritmo que dificulta na classificação.

- min_samples_split: nesse parâmetro é definido a quantidade de amostra que será dividido um nó interno.

2.4.3 Técnicas de Avaliação do Modelo Preditivo (acurácia, recall, precision)

Para um problema de classificação, são desenvolvidos vários tipos de modelos preditivos usando uma base de dados. Uma maneira simples de avaliar o modelo é dividindo a base em duas partes, uma para treinamento e outra para teste. Assim, a avaliação do classificador é feita sobre a parte da base que foi utilizada para teste, a qual o algoritmo pode fazer a estimativa de forma correta ou errada (OLSON; DELEN, 2008).

A Tabela 1 apresenta as possíveis estimativas que podem acontecer quando o classificador está sendo testado, isto é, o classificador pode estimar um item como positivo, mas o item ser um negativo. Nesse caso, aconteceu um falso positivo. Do contrário, o estimador pode classificar o item como negativo, mas o *label* do item ser positivo. Nesse caso, aconteceu um falso negativo. Essas duas estimativas aparecem na diagonal secundária da Tabela 1. Já na diagonal principal, aparecem os verdadeiros positivos, quando a classificação é positiva e o item é positivo, e o verdadeiro negativo, quando a classificação é negativa e o item é negativo. Sumarizando, a diagonal principal representa as decisões corretas do classificador, já a diagonal secundária representa os erros (OLSON; DELEN, 2008).

Tabela 1 – Matriz de Coincidência

		Classe Verdadeira	
		Positivo	Negativo
Classe Prevista	Positivo	Contagem Verdadeiro Positivo (VP) 1 1	Contagem Falso Positivo (FP) 0 1
	Negativo	Contagem Falso Negativo (FN) 1 0	Contagem Verdadeiro Negativo (VN) 0 0

Fonte: (OLSON; DELEN, 2008)

Com base na Tabela 1, a seguir serão apresentadas as três principais métricas para realizar a avaliação de desempenho do preditor em problemas de classificação. Destaca-se que a Tabela 1 apresenta a fonte das medidas do desempenho de onde são retiradas as equações para solução dos problemas de classificação com duas classes (OLSON; DELEN, 2008).

Acurácia é a precisão geral do classificador, calculado conforme a fórmula abaixo

(OLSON; DELEN, 2008):

$$Acurácia = \frac{VP + VN}{VP + VN + FP + FN} \tag{2.3}$$

A medida de desempenho Recall, ou sensibilidade, tem como objetivo, localizar todos os elementos reais positivos (POWERS, 2011)

$$Recall = \frac{VP}{VP + FN} \tag{2.4}$$

Outra medida de desempenho é o Precision, proporção dos elementos previstos positivos dos elementos corretamente Positivos reais (POWERS, 2011)

$$Precision = \frac{VP}{VP + FP} \tag{2.5}$$

2.4.4 Ética e privacidade dos dados

Poucas pesquisas procuram mencionar questões de éticas e privacidade. Entretanto, uma vez que este trabalho reúne dados dos alunos de programação para uso de classificação, é de grande importância apurar essas questões. (IHANTOLA et al., 2015).

2.4.4.1 Privacidade

O contexto da privacidade está ligado a capacidade de reconhecer a identidade dos alunos na base de dados. Apesar da prática em ocultar dados pessoais como nome, número da matrícula e email por exemplo, o aluno ainda pode ser identificado, por outros dados, como exemplo, o endereço IP usado, idade, gênero, conteúdos que escrevem e etc. Todos esses dados precisam ser analisados antes da divulgação. Apesar de ser uma preocupação óbvia com respeito a coleta de dados do processo de programação, poucas pesquisas abordam problemas relacionados à privacidade (IHANTOLA et al., 2015).

O reconhecimento formalmente da privacidade não só contribui no rigor da apresentação da pesquisa, como também proporciona informações fundamentais acerca dos principais desafios no recolhimento automáticos e avaliação de dados. Caso a privacidade não for formalmente abordada como parte do estudo de pesquisa, a ética poderá vir a ser questionada, bem como, ser comprometida a capacidade de se estender a pesquisa que está sendo realizada. Exemplo prático seria o comprometimento em dar continuidade num trabalho de pesquisa, não podendo ser compartilhado ou usado por outros pesquisadores à base de dados processado, gerando uma incapacidade em compartilhar detalhes do processamento dos dados, para que a pesquisa seja replicada ou reproduzida (IHANTOLA et al., 2015).

2.4.4.2 Ética

As questões da ética estão relacionadas com às da privacidade. A coleta de dados, a transparência para os participantes, e o uso dos dados coletados são métodos que estão

relacionados com a ética. As diretrizes e os regulamentos existem com o objetivo de proteger a privacidade e a segurança do aluno (IHANTOLA et al., 2015).

Na coletar os dados referentes aos alunos é importante estar ciente desses regulamentos. A divulgação desses dados podem acarretar sérios danos ao aluno, desde sua carreira até mesmo na vida social. Em uma coleta de dados, o pesquisador deverá ter permissão com antecedência para reunir e usar esses dados. O motivo pelo qual são usados os dados coletados sem dúvida é de grande importância na consideração da ética. A elaboração de uma pesquisa nos dados, com o propósito de entender as caraterísticas que influenciam na aprendizagem do aluno, não apresenta preocupação a respeito da ètica, todavia, o uso desses dados para adaptar tutoria inteligente pode começar a gerar algumas preocupações éticas, pois isso pode induzir que um sistema possa orientar no comportamento do aluno (IHANTOLA et al., 2015).

2.5 Planejamento e Projeto dos Experimentos

Na metodologia de avaliação, o algoritmo *CfsSubsetEval* seleciona os atributos mais importantes na base, medindo sua capacidade de predizer por antecipação e o grau de redundância. Já no método de busca, o algoritmo *BestFirst* como o próprio nome diz, o primeiro melhor, inicia a avaliação com nenhum atributo e inclui um atributo por vez na lista de *Ranking* (FRANK et al., 2004).

Realizadas as técnicas de seleção de atributos, outras técnicas como balanceamento de dados e normalização foram aplicadas na base como estratégia para obtenção de um melhor resultado na predição. Uma base de dados desbalanceada é quando existe quantidade de casos desiguais dentro de uma classe, por exemplo, a quantidade de caso de evasão diferente da quantidade dos que permaneceram na disciplina (CHAWLA et al., 2004) dentro da classe.

Ainda, em alguns experimentos foram realizados o Cross-validation, que é um método de validação cruzada em que a base de dados é dividida em k(nesse caso 10) partes no qual é usado k-1 para treino e 1 para teste (DWAN et al., 2017).

3 TRABALHOS RELACIONADOS

Pesquisas sobre como melhorar o processo de ensino e aprendizagem utilizando dados da interação de alunos com ambientes *online* têm sido um dos grandes focos na área de LA. Em geral, são utilizados algoritmos de mineração de dados e AM aplicados à base de dados educacionais, a fim de compreender os fatores implícitos que podem ser melhorados no contexto educacional. Nesse sentido, serão apresentados alguns trabalhos da literatura que são relevante e que podem contribuir com o problema do alto índice de evasão em turmas de introdução à programação, o qual é o foco deste trabalho.

3.1 Predição de Zona de Aprendizagem de Alunos de Introdução à Programação em Ambientes de Correção Automática de Código

No trabalho de Dwan et al. (2017) os autores propuseram um sistema de predição binária, para inferir o desempenho dos alunos nas avaliações da disciplina de Programação, por meio da aplicação de 5 algoritmos diferentes (Support Vector Machine (SVM), Random Forest (RF), AdaBoosting (AB), Árvore de Decisão (AD) e K-Nearest Neighbours (KNN)). Os autores trabalharam com uma amostra de 486 alunos e utilizaram atributos fornecidos pela interação do estudante com o juiz *Online*, como por exemplo: número de tentativas de solução; número de linhas do código; velocidade de implementação etc. A pesquisa alcançou uma acurácia de 78,3% com a base de dados desbalanceada e uma precisão de 72,8% com a base balanceada.

Ressalta-se que este trabalho se baseia na pesquisa de Dwan et al. (2017), utilizando sua base de dados de treinamento e teste, e realizando uma análise e otimização de seus atributos. Porém, destaca-se que esse estudo possui uma finalidade diferente da pesquisa supracitada, uma vez que objetiva detectar alunos com tendências a evasão nas duas primeiras semanas de aula da disciplina, possibilitando ao professor tomar medidas pedagógicas corretivas.

Tabela 2 – Apresentação da acurácia, recall dos positivos, precision dos positivos, recall dos negativos, precision dos negativos dos algoritmos de aprendizagem de máquina com melhores resultados (em negrito)

	Sessão 1				Sessão 2				Sessão 3				Sessão 4			
	SVM	RF	AD	AB	SVM	RF	AD	AB	SVM	RF	AD	AB	SVM	RF	AD	AB
Acc.(%)	71.5	74.1	**78.3**	73.1	84.9	**86.0**	84.0	84.3	75.3	**76.7**	72.2	75.6	83.8	**84.9**	78.6	86.5
Rec. Pos.(%)	**94.0**	90.3	85.8	84.3	**98.5**	96.2	95.8	96.2	**96.2**	92.5	85.0	85.9	95.7	95.2	89.6	**99.6**
Prec. Pos.(%)	73.1	76.9	**93.1**	78.9	85.6	**90.2**	85.6	86.5	76.2	78.8	78.7	**81.7**	87.0	**88.3**	86.2	86.7
Rec. Neg.(%)	18.5	35.9	43.5	**46.7**	17.3	**36.4**	25.0	25.0	19.0	32.9	38.0	**48.1**	8.3	**19.4**	8.3	3
Prec. Neg.(%)	56.7	61.1	**72.7**	55.8	**69.2**	61.6	54.2	56.5	**65.2**	61.9	48.4	55.9	23.1	**38.9**	11.1	50

Fonte: (DWAN et al., 2017)

Tabela 3 – Avaliação dos Modelos Preditivos mais Precisos

Sessão	Acc.	*Recall* Pos.	Prec. Pos.	*Recall* Neg.	Prec. Neg.
1	72.8%	66.3%	76.3%	79.3%	70.2%
2	71.2%	76.9%	68.9%	63.4%	73.9%
3	74.7%	71.8%	75.7%	77.5%	73.8%
4	72.2%	83.3%	68.2%	61.1%	78.6%

Fonte: (DWAN et al., 2017)

3.2 Previsão de Estudantes com Risco de Evasão Utilizando Técnicas de Mineração de Dados

Manhães et al. (2011) realizaram três experimentos para fazer a predição se os estudantes corriam risco de evasão. Para tanto, foi utilizado dez algoritmos de classificação disponível na ferramenta WEKA, que são: BayesNet (BN); DecisionTable (DT); J48 (J48); JRip (JR); MultilayerPerceptron (MP); NaiveBayes (NB); OneR (OR); RandomForest (RF); SimpleCart (SC); SimpleLogistic (SL).

O número de estudantes envolvidos no experimento foi de 887 calouros do curso de Engenharia Civil, sendo 543 os quais concluíram o curso e 344 alunos que não concluíram. Os autores obtiveram acurácia média variando entre 75 a 80%.

Para chegar a esses resultados, foram coletados dados do sistema acadêmico da UFRJ no período de 1994 a 2005, preservando a identificação dos alunos, com intuito de levantar estudos não só para entender mas de propor possíveis soluções para o problema. A base adquerida foi composta pelos atributos:

1. Coeficiente de rendimento: valor do coeficiente de rendimento acumulado no período, que é calculado através do produto entre a nota do aluno e o valor do crédito da disciplina;
2. Notas nas disciplinas: de Cálculo Diferencial e Integral I, Engenharia de Meio Ambiente, Programação de Computadores, Química e Introdução à Engenharia Civil;
3. Situação na disciplina de Cálculo I: Se o aluno foi Aprovado, Reprovado por Nota ou Reprovado por falta;
4. Situação das disciplinas: Introdução à Engenharia Civil, Química, Programação de Computadores, Engenharia de Meio Ambiente, e;
5. o atributo identificador da classe do aluno.

No decorrer de sua pesquisa foram apresentados três tabelas para comparar o desempenho de cada algoritmo aplicado no domínio do problema. No primeiro experimento a base de dados foi dividida em 10 conjuntos utilizando o método de validação cruzada conforme a Tabela 4. No segundo experimento a base de dados foi dividida de outra forma, utilizando um processo randômico de selecionar o conjunto para treinamento e teste, sendo 66 e 34% respectivamente, resultando na Tabela 5. Já o terceiro e último experimento foi escolhido a opção *Supplied test set* para seleção do conjunto de treinamento e teste da base de dados, separando 2/3 e 1/3 respectivamente. A Tabela 6 mostra a acurácia dos classificadores para o conjunto de teste.

Tabela 4 – Resultado do experimento 1

Classificadores	**OR**	**JR**	**DT**	**SC**	**J48**	**RF**	**SL**	**MP**	**NB**	**BN**
Acurácia	78,39	77,88	78,07	78,92	77,86	76,74	78,32	76,36	78,85	78,78

Tabela 5 – Resultado do experimento 2

Classificadores	**OR**	**JR**	**DT**	**SC**	**J48**	**RF**	**SL**	**MP**	**NB**	**BN**
Acurácia	78,50	78,44	79,63	79,36	77,87	77,28	79,76	76,35	80,12	79,66

Tabela 6 – Resultado do experimento 3

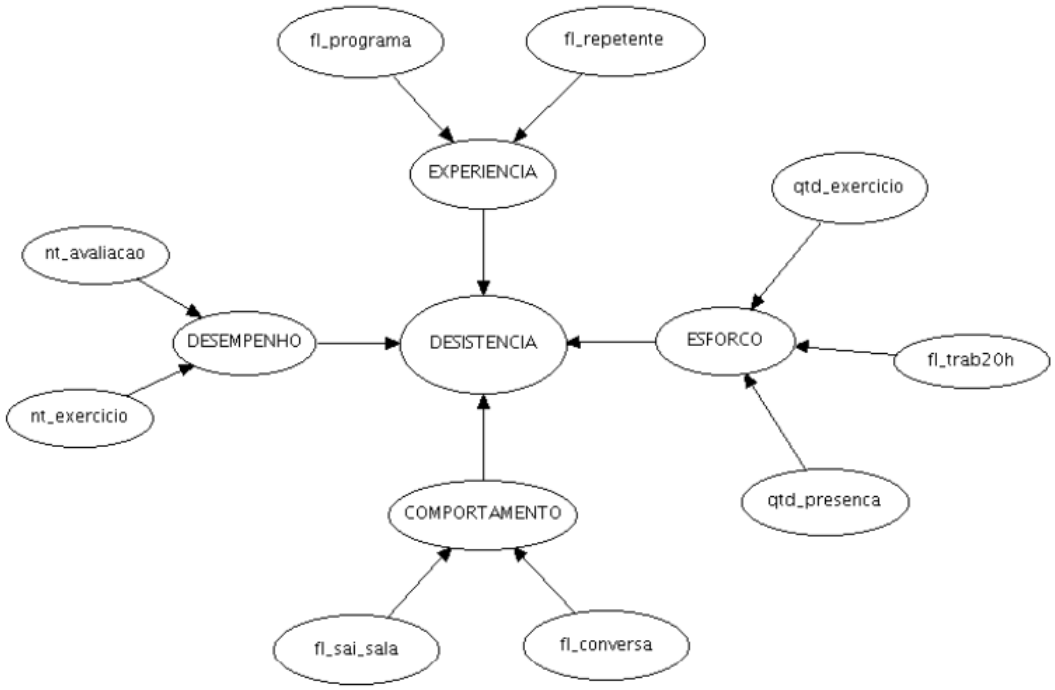
Classificador	**OR**	**JR**	**DT**	**SC**	**J48**	**RF**	**SL**	**MP**	**NB**	**BN**
Acurácia	81,94	76,04	72,92	76,39	80,21	80,21	82,29	74,31	81,25	80,56
matriz de confusão	162 16 36 74	142 36 33 77	146 32 46 64	145 33 35 75	161 17 40 70	153 25 32 78	167 11 40 70	134 44 30 80	162 16 38 72	161 17 39 71
Verdadeiro Positivo	0,91	0,80	0,82	0,81	0,90	0,86	0,94	0,75	0,91	0,90
Falso Negativo	0,09	0,20	0,18	0,19	0,10	0,14	0,06	0,25	0,09	0,10
Verdadeiro Negativo	0,67	0,70	0,58	0,68	0,64	0,71	0,64	0,73	0,65	0,65
Falso Positivo	0,33	0,30	0,42	0,32	0,36	0,29	0,36	0,27	0,35	0,35

Fonte: (MANHÃES et al., 2011)

3.3 Um Assistente de Predição de Evasão aplicado a uma disciplina Introdutória do curso de Ciência da Computação

Em seu trabalho, Martins et al. (2012) buscaram através de dois métodos alcançar uma previsão de possíveis alunos com tendência à evasão do curso da ciência da computação, precisamente na disciplina de algoritmo e programação. O primeiro experimento utilizou redes bayesianas, em que as variáveis foram agrupadas e classificadas por características conforme Figura 5. Ao classificador, foi dado o nome de Assistente de Previsão da Evasão (APE), o qual foi associado a um Sistema Tutor Inteligente (STI) por nome Alice. Foi utilizado nesse experimento duas turmas, ambas contendo 30 alunos.

Figura 5 – Rede Bayesiana de desistência proposta por especialista



Fonte: (MARTINS et al., 2012)

As variáveis apresentadas na Figura 5, que foram utilizadas pelo APE para realizar a classificação, são descritas abaixo:

- nt_exerc: Média das notas dos exercícios realizados em sala de aula pelos alunos;
- nt_avaliação: Média das notas do aluno em avaliações realizadas na sala de aula;
- fl_programa: *Flag* que discrimina se o aluno tem experiência em programação (0 – Não, 1 – Sim, null – Não informou);
- fl_repetente: *Flag* indica se o aluno é repetente (0 – Não, 1 – Sim, null – Não informou);
- qtd_exercício: Quantidade de exercícios realizados em sala de aula por um determinado aluno;
- fl_trab20h: *Flag* indica se o aluno trabalha mais de 20 horas semanais (0 – Não, 1 – Sim, null – Não informou);
- qtd_presença: Quantidade de presença no diário de aula da disciplina do aluno;
- fl_conversa: *Flag* indica se o aluno conversa durante as explicações em sala (0 – Não, 1 – Sim, null – Não informou); e

- fl_sai_sala: *Flag* indica se o aluno costuma sair da sala durante a aula (0 – Não, 1 – Sim, null – Não informou).

Entretanto o APE obteve resultados modestos, conforme Tabela 7.

Tabela 7 – Resultados dos experimentos da Rede Bayesiana

	Turma A		**Turma B**	
Total de alunos	30		30	
Desistentes	7		10	
Reprovados	1		3 (identificou-se 2)	
Índice de Desistência	**0,4**	**0,45**	**0,4**	**0,45**
Verdadeiro Positivo	4	6	7	7
Verdadeiro Negativo	0	3	9	13
Falso Positivo	3	1	3	3

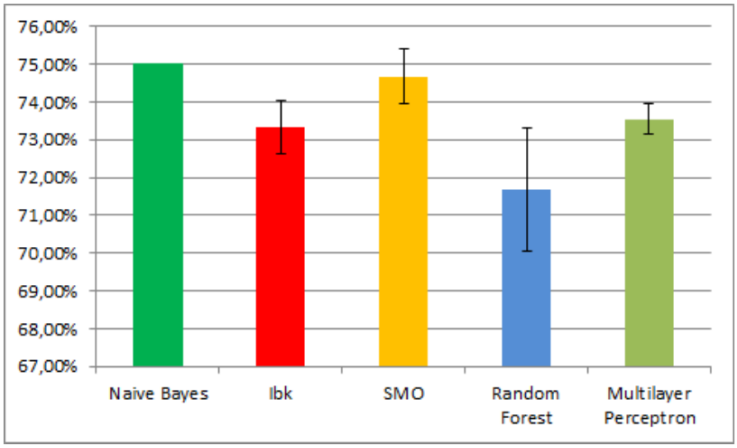
Fonte: (MARTINS et al., 2012)

Em um segundo experimento, os autores obtiveram uma maior precisão utilizando processo KDD (*Knowledge Discovery in Database* - Descoberta de Conhecimento na Base de Dados) com o algoritmo NNge.

3.4 Predição de desempenho de alunos do primeiro período baseado nas notas de ingresso utilizando métodos de aprendizagem de máquina

Brito et al. (2014) propõem predizer o desempenho dos alunos, isto é, se o aluno irá passar na disciplina ou não. Para tanto, foram utilizadas as notas de ingresso no vestibular, isto é, um conjunto de três atributos de entrada para a predição: Média Geral, Média de Matemática e a Média de Física. Os autores submeteram à base de dados a cinco algoritmos de aprendizagem de máquina: NaiveBayes, Ibk, SMO, Random Forest e Multilayer Perceptron. A precisão dos algoritmos podem ser vistas na Figura 6.

Figura 6 – Gráfico com a média da precisão dos algoritmos



Fonte: (MARTINS et al., 2012)

O número de alunos envolvidos no experimento foram 300 estudantes, do primeiro período do curso de Ciência da Computação, sendo 138 aprovados e 162 reprovados. Os autores utilizaram a validação cruzada para a divisão do conjunto de treinamento e teste, dividindo o conjunto de dados em 10 partes iguais, sendo 9 para treinamento e 1 utilizado para o teste.

3.5 Modelagem e Predição de Reprovação de Acadêmicos de Cursos de Educação a Distância a partir da Contagem de Interações

O trabalho de Detoni et al. (2015) buscou inferir quais alunos de ensino a distancia tinham risco de reprovação, através da análise da contagem de interações dos estudantes dentro de ambientes virtuais de aprendizagem (AVA). Em um primeiro experimento, os autores usaram como atributos dos modelos preditivos o número de interações acumulado até um dado momento, enquanto que em um segundo experimento essas interações eram acumuladas semanalmente.

O número de alunos envolvidos foi de 604, no nível de graduado da Universidade Federal de Pelotas (UFPEL) do 1º e 2º Semestres, nos cursos de Licenciatura em Educação do Campo e Licenciatura em Pedagogia. Os autores utilizaram para predição cinco modelos: Rede Bayesiana, Redes Neural, Árvores de Decisão (C45) e Floresta Aleatória, todas disponíveis na ferramenta WEKA. A maior precisão foi obtida com Redes Bayesianas, com um número suficientemente grande de semanas (maior ou igual a 3), e para as primeiras semanas o algoritmo de Floresta Aleatória atingiu a maior acurácia.

No primeiro experimento os autores realizaram dois testes conforme a Tabela 8. No caso "entre semestre", para treinar os modelos foram utilizados os semestres das turmas (A e B), os

quais foram avaliados nos semestres seguintes nas mesmas turmas e vice-versa. Já no caso "entre turmas"a divisão é feita dentro de um mesmo semestre do mesmo curso, utilizando uma turma "A"para treinar o modelo e a outra turma "B"foi utilizada para avaliar o modelo (e vice-versa).

Tabela 8 – Precisão para classificação de reprovado - sem discriminar o tipo de atributos

Experimento	Caso	Modelo	S1	S2	S3	S4	S5	S6	S7
1	Entre Semestres	Rede Bayesiana	0,00	0,22	0,44	0,51	0,60	0,63	0,64
		C4.5	0,00	0,21	0,37	0,40	0,52	0,56	0,56
		Floresta aleatória	0,10	0,33	0,36	0,40	0,48	0,51	0,54
	Entre Turmas	Rede Bayesiana	0,00	0,24	0,45	0,51	0,58	0,62	0,66
		C4.5	0,00	0,20	0,39	0,42	0,51	0,59	0,60
		Floresta aleatória	0,10	0,28	0,33	0,41	0,50	0,56	0,61
2	Entre Semestres	Rede Bayesiana	0,00	0,53	0,60	0,66	0,70	0,71	0,72
		C4.5	0,00	0,50	0,58	0,58	0,57	0,58	0,57
		Floresta aleatória	0,01	0,49	0,54	0,57	0,57	0,58	0,59
	Entre Turmas	Rede Bayesiana	0,00	0,46	0,60	0,67	0,71	0,73	0,73
		C4.5	0,00	0,45	0,55	0,56	0,59	0,59	0,58
		Floresta aleatória	0,02	0,48	0,54	0,55	0,55	0,57	0,59

Fonte: (DETONI et al., 2015)

Os atributos utilizados nos modelos preditivos são apresentados abaixo:

- Situação Acadêmica Descrição: Situação final do aluno na disciplina (classe);
- Número de Interações com o AVA;
- Média: média do total de interações dividida pelo número de semanas;
- Mediana: mediana do conjunto de interações por semana;
- Semanas Zeradas: número de semanas com zero interações;
- Média da Diferença: média da diferença entre a semana i e a semana $i + 1$;
- Razão com Professores: razão entre o total de interações do aluno e dos professores;
- Razão com Tutores: razão entre o total de interações do aluno e dos tutores;
- Fator de Empenho: razão entre as interações da semana do aluno e a média de interações da turma naquela semana.

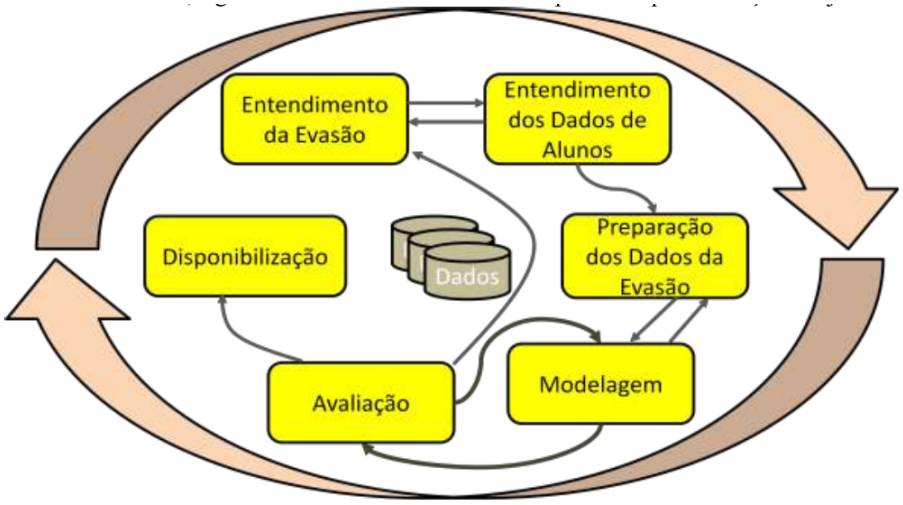
Finalmente, os autores detectaram que os modelos tornam-se mais precisos à medida que os atributos são rotulados com um timestamp da contagem de interações por semana, ao invés de apenas informar quantas interações foram realizadas até um dado momento.

3.6 UMA VISÃO DO FUTURO: PREVISÃO DE EVASÃO EM CURSOS DE GRADUAÇÃO PRESENCIAIS DE UNIVERSIDADES PÚBLICAS: O CASO DO CURSO DE ZOOTECNIA

KANTORSKI et al. (2015) apresentaram três simulações para previsão da evasão em cursos de graduação presenciais, aumentando o número de atributos em cada simulação. A pesquisa mostra uma visão da combinação de vários modelos de mineração de dados e aprendizagem de máquina no intuito de otimizar o resultado do processo. Foi utilizado para a base de dados 302 alunos do curso de Zootecnia do primeiro semestre de 2015. Como resultados, os autores obtiveram uma taxa de 74% de acurácia.

Os autores utilizaram um processo cíclico para maximizar a precisão do modelo preditivo, conforme Figura 7. Esse processo foi baseado em uma metodologia chamada de CRISP-DM (*CRoss Industry Standard Process for Data Mining*), na qual foram analisados os fatores que encadeavam a evasão, através da identificação e descoberta de conhecimento nas bases de dados.

Figura 7 – Metodologia CRISP-DM adaptada para Evasão



Fonte: (KANTORSKI et al., 2015)

3.7 Considerações dos Trabalhos Relacionados

Como visto no referido capítulo, a disponibilidade de dados armazenados em ambientes educacionais tem despertado o interesse de pesquisadores no estudo detalhado sobre o aprendizado dos alunos, muitos com o intuito de resolver o problema de evasão e reprovação de estudantes.

O que essas pesquisas tinham em comum era o uso de técnicas de AM e mineração de dados para o melhor entendimento dos padrões de aprendizagem do aluno, usando para tanto os dados que esses estudantes vão deixando à medida que eles vão interagindo com ambientes *online*. Destaca-se que à medida que se identifica, por exemplo, quais as características de um aluno que reprova, ou qual as características de um aluno que vai evadir, é possível generalizar essa aprendizagem em outros contextos a fim de que se tome medidas proativas para evitar que ocorra esse problema.

Frisa-se ainda que esse problema é recorrente em disciplinas de IPC, mesmo usando juízes *online*. Assim o presente trabalho irá usar as descobertas dos trabalhos analisados juntamente com novas propostas para fazer a predição da evasão de alunos de turmas de IPC que usam Juízes *Online* em tempo hábil.

Para tanto, foi importante analisar quais algoritmos de AM, quais atributos, quais ferramentas são utilizadas na literatura e quais obtiveram os melhores resultados. Além disso, observou-se o tamanho da amostra utilizada, já que os dados são os combustíveis para os algoritmos de AM e Mineração de Dados. Finalmente, observou-se também qual foi o ambiente *online* utilizado nos experimentos. Tais informações coletadas durante a revisão da literatura são apresentadas na Tabela 9.

Tabela 9 – Resumo dos trabalhos relacionados

	Manhães 2011	**Martins 2012**	**Brito 2014**	**Detoni 2015**	**Kantorski 2015**	**Proposta**
Tamanho Amostra	887	60	300	604	302	482
Algoritmos de Predição	BayesNet (BN); DecisionTable (DT); J48 (J48); JRip (JR); Multilayer Perceptron (MP); Naive Bayes (NB); OneR (OR); Random Forest (RF); Simple Cart (SC); Simple Logistic (SL)	Nnge (NG); OneR (OR) e Rede Bayesiana (RB).	Ibk (IB); Multilayer Perceptron (MP); NaiveBayes (NB); Random Forest (RF); SMO (SM).	C4.5 (C4); Random Florest (RF); Rede Bayesiana (RB); Rede Neural (RN).	C4.5 (C4); CART (CT) ; Multilayer Perceptron (MP); Naïve Bayes (NB); Nearest neighbor (NN) e; OneR (OR).	Árvode de Decisão (AD) e Random Forest (RF)
Atributos	Numéricos e categóricos	Binário, categóricos e numéricos.	Desempenho no primeiro período do curso e numérico.	Interação, categórico e numérico	pessoais, acadêmicas, sociais, econômicas e numéricos	atributos de intereção, análise estática de código e numérico
Ferramenta	Weka	Weka	Weka	Weka	-	Weka e scikit-learn
Ambiente Online	-	-	Moodle	Moodle	-	Juiz Online
Oferta	Predição com Risco de Evasão	Predição de Evasão	Desempenho de alunos	Predição de Reprovação	Predição de Evasão	

Fonte: Própria

Percebeu-se que, em sua maioria, os trabalhos desenvolvidos propõem a previsão da evasão dos alunos do curso de graduação. E mesmos as pesquisas que utilizaram o modelo preditivo para verificar o desempenho do aluno e a condição se aprovado ou reprovado, os atributos de ambos são relevantes para serem utilizados no modelo preditivo de evasão.

Finalmente, os trabalhos relacionados nessa seção fornecem métodos e técnicas que auxiliarão no desenvolvimento deste trabalho. No próximo capítulo serão especificadas as etapas do desenvolvimento desta pesquisa, o tratamento dos dados, gerenciamento da base de dados e seleção dos métodos que serão utilizados para previsão de alunos com risco de abandono do curso de Programação.

4 MÉTODO PROPOSTO

Foram pesquisados na literatura diferentes métodos para inferir a evasão de alunos, conforme explicado no capítulo 3. Percebeu-se que em geral são construídos modelos preditivos utilizando atributos baseados na interação dos alunos com ambientes *online*.

Nesta pesquisa, foram utilizados atributos orientados aos dados da interação do aluno com um Juiz *Online*, os quais foram pré-processados e filtrados através de algoritmos de seleção de atributos para depois serem utilizados em algoritmos de AM baseados em AD. Assim, com os atributos mais relevantes foi formado uma coleção de evidências capaz de inferir se o aluno irá evadir ou não na disciplina.

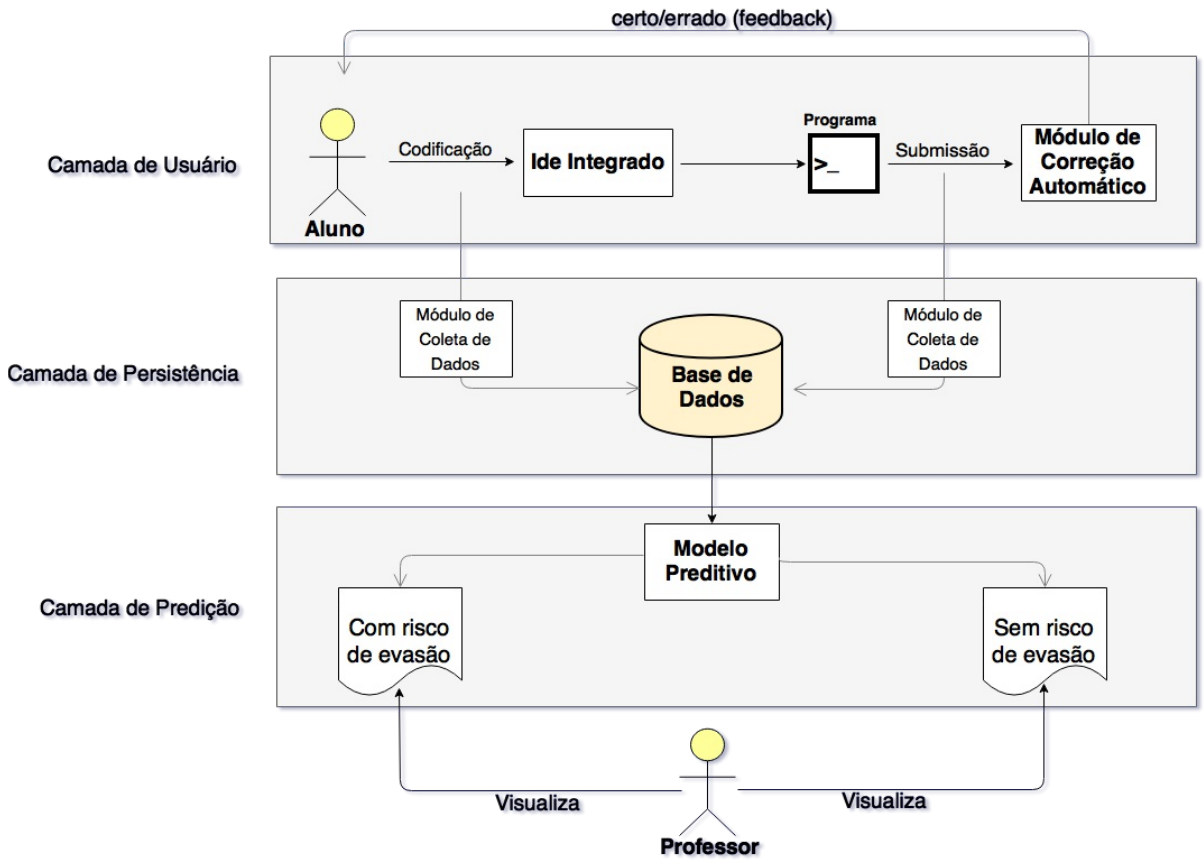
Neste capítulo foram explanados a arquitetura proposta, a descrição da base de dados, o método utilizado para a predição de evasão e as tecnologias utilizadas.

4.1 Arquitetura

À medida que os discente resolvem exercícios em uma IDE integrada ao Juiz *Online*, o processo de codificação é registrado, assim todo o itinerário de aprendizagem do aluno é monitorado por um módulo do próprio Juiz *Online*. Além disso, as submissões também são gravadas na base de dados. Tais dados serão utilizados para fazer a predição da evasão de alunos do curso de IPC.

Uma visão geral do funcionamento do método proposto nesta pesquisa é apresentado na Figura 8, o qual foi organizado em camadas. Na camada de usuário, o aluno resolve problemas de programação em uma IDE integrada ao juiz *online* e recebe *feedback* automático à medida em que faz a submissão. Na camada de persistência, existe um módulo do sistema que registra os *logs* do discente enquanto ele resolve os problemas na IDE. Além disso, o módulo de coleta de dados persiste as submissões dos alunos, independente se eles estiverem certos ou errados. Por fim, existe uma camada de predição, na qual um algoritmo de aprendizagem de máquina usa os dados coletados na camada de persistências para construir um modelo preditivo que infere se o aluno irá evadir ou não.

Figura 8 – Arquitetura do Método Proposto



Fonte: Própria

Destaca-se que a camada de predição, através da estimativa do modelo preditivo, permite que o professor saiba ao término da segunda semana de aula quais alunos possuem uma tendência a evadir da disciplina.

4.2 Descrição da Base de Dados

A base de dados utilizada neste estudo foi a mesma disponibilizada por Dwan et al. (2017). Nela foram coletados dados a partir do sistema *CodeBench*, que é um juiz *Online* desenvolvido pelo Instituto de Computação da UFAM, utilizado como suporte a professores e alunos iniciantes em programação. Através do *CodeBench*, os professores podem elaborar exercícios de programação para seus alunos, que por sua vez podem desenvolver soluções e submetê-las através da interface do sistema.

Os dados foram coletados em 9 turmas de Introdução à Programação de Computadores. A escolha da base de dados utilizada por Dwan et al. (2017) se deu devido a ela ter sido coletada com um alto nível de granularidade (abordagem data-driven), o que possibilita uma análise mais

acurada do perfil de programação do aluno se comparado com uma abordagem que o perfil do aluno é modelado utilizando variáveis estáticas coletadas por questionários e observações empíricas.

Sendo que, neste trabalho foram usados apenas os dados das duas primeiras semanas de aula e destaca-se que nesse período os professores passaram uma lista de atividade com 10 problemas de programação, onde cada questão valia um ponto. As turmas era de cursos diferentes, sendo que nenhum dos cursos tinha computação como atividade fim.

O Semestre durou 14 semanas. Assim, uma sessão tinha dados de 2 semanas, totalizando 7 sessões, isto é, um período de 14 semanas, conforme a Tabela 10. Em que, S1 significa a primeira semana, S2 a segunda semana, S3 a terceira semana, e assim sucessivamente até a décima quarta semana. Cada sessão o aluno faz uma lista com 10 exercícios no *Codebench* e após os exercícios realiza uma avaliação no próprio *Codebench*.

Tabela 10 – Período da disciplina

SEMESTRE													
Sessão 1		Sessão 2		Sessão 3		Sessão 4		Sessão 5		Sessão 6		Sessão 7	
S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14

Fonte: Própria

Cada evento executado dentro do editor de texto do *Codebench* é registrado a data e a ação realizada pelo aluno. Entre os principais eventos registrados estão viewportchange, mousedown, change, keyhandled, contextmenu, blur, focus, testar, console_testar, console_submeter e submeter. Segue abaixo o significado de cada evento (PEREIRA, 2016):

- viewportchange - representa o redimensionamento da caixa de edição, onde tem o cursor do texto. Ex.: O aumento da caixa de edição pode conter o código digitado;
- mousedown - representa o click com o botão esquerdo do mouse;
- change - informa uma mudança no texto, podendo ser uma inserção, delete de caracter ou colar um texto vindo de uma área de transferência, popularmente o CONTROL + V;
- keyhandled - informa o uso de qualquer tecla, especialmente as teclas de navegação como o teclas para cima, para esquerda ou tecla end;
- contextmenu - informa o clique do botão direito do mouse;
- blur - informa qual área da caixa de edição perdeu o foco. Quando outro programa ou outra parte da interface da página foi clicada, por tanto ganhado o foco;

- focus- informa qual área de edição recebeu o foco. O aluno voltou a posicionar a clicar na área de edição;
- testar - indica que pressionou o botão testar o código-fonte dentro do próprio Codebench;
- console_testar - imprimi a saída do interpretador python;
- submeter - quando o aluno submete seu código fonte como resposta ao sistema, porem o sistema recusa caso o código-fonte submetido não produza os resultados esperados;
- console_submeter - devolve uma mensagem de erro ou de sucesso de envio do código fonte.

Os Atributos extraídos do processo de resolução de questões no juiz *online* e que foram utilizados na elaboração de perfis de programação dos alunos, são os seguintes (DWAN et al., 2017):

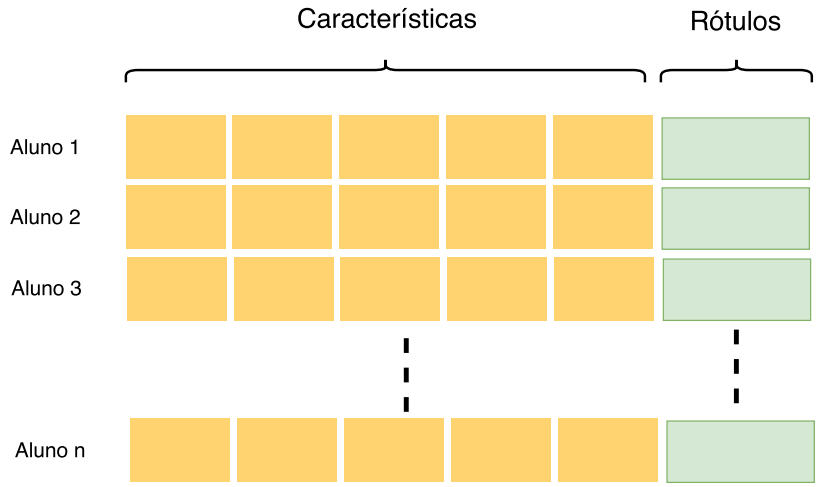
- blank_line: quantidade de linha em branco no código deixado pelo estudante;
- comments: quantidade de comentários no código;
- dificuldade: opção de 0 a 2, onde o aluno seleciona o grau de dificuldade encontrado ao resolver o exercício, em que 0 significa que o aluno não encontrou qualquer grau de dificuldade, 1 razoável e 2 questão difícil;
- errosSintaxe: quantidade de erro no algoritmos submetidos;
- lloc: média de número de linhas lógicas dos códigos, sem contabilizar importação de bibliotecas, comentários e linhas em branco;
- mediaDeletes: média de quantas vezes a tecla Delete foi pressionada;
- mediaNumLinhasLog: média da quantidade de linha no código em uma lista de exercício;
- mediaSubmissaoPorListExerc: média de quantas vezes os exercícios foram submetidos a correção;
- mediaSucesso: Média em que o aluno obteve sucesso, isto é, a razão entre o número de questão avaliado corretamente pelo número de tentativa de submissão do código;
- meidaTeste: razão entre o total de tentativas de solucionar as questões de uma lista e número total de questões da lista;
- nota_avaliacao_codebench: nota na avaliação realizada em cada sessão;
- notaLista: nota da lista de exercício;

- notaListaCehckCola: nota do aluno em cada lista de exercício, considerando apenas questões resolvidas que geram no mínimo 50 linhas no log;
- num_acesso: quantidade de acesso ao sistema de aprendizagem;
- proporcaoCaract: fração entre a quantidade de vezes que foi ativado as teclas "Ctrl + V"e a quantidade de caracteres digitados na IDE;
- RC1: quantidade de tentativas de submissão, independente se o código está certo ou errado;
- single_comments - linhas de comentários no código;
- sloc: média de número de linhas dos códigos submetidos;
- tempoUsoIde: tempo em que o aluno ficou com a IDE online;
- velocidadeDigit: velocidade em que as teclas foram digitadas;

4.3 Construção do Modelo Preditivo

Para compor o modelo preditivo foi utilizado o perfil de programação do aluno, baseados em algoritmos de aprendizagem de máquina supervisionada presentes na literatura. Mais estritamente, foram usados algoritmos de classificação para a predição de evasão. A base de dados foi dividida em duas partes, uma para treinamento do algoritmo e outra para teste, uma vez que as características foram capturadas juntamente com os rótulos na parte do treinamento conforme Figura 9 abaixo, houve uma classificação binária, isto é, alunos que se evadiram e que concluíram o curso, capacitando o algoritmo para indução na parte de teste da divisão. Os modelos foram comparados utilizando a métrica de previsão, recall e acurácia (POWERS, 2011).

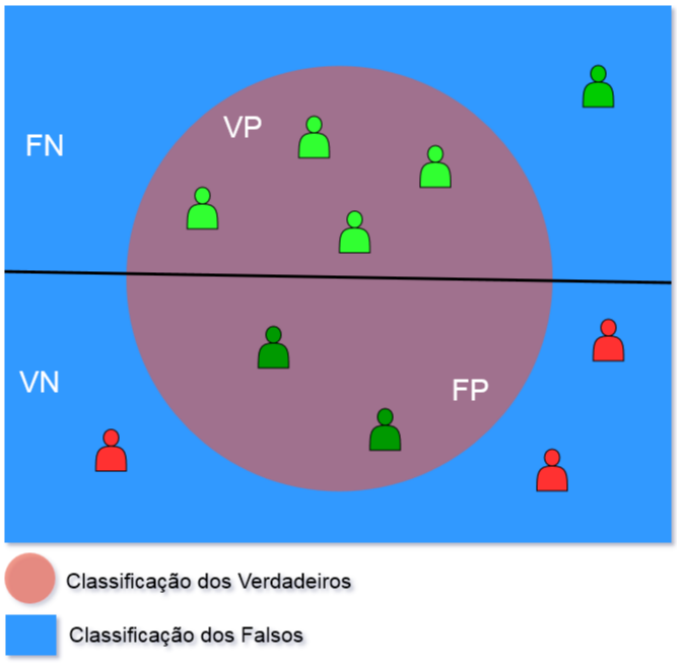
Figura 9 – Classificação de dados supervisionados



Fonte: Própria

Para uma explicação transparente de como as métricas funcionam no modelo preditivo, conforme foram detalhadas na subseção 2.4.3, segue a Figura 10 abaixo com as seguintes informações. O círculo representa a classificação realizada pelo modelo preditivo dos alunos que evadiram, e o retângulo dos que não evadiram.

Figura 10 – Funcionamento das Métricas no Modelo Preditivo



Fonte: Própria

Na demonstração da Figura 10, o modelo preditivo acertou 4 dos que realmente evadiram (VP), classificou 2 como evadiu, quando na realidade não evadiram (FP). Por outro lado, o modelo acertou na classificação como "não evadiu"três vezes (VN) e errou uma vez (FN).

Ressalta-se que para realizar a predição da evasão dos alunos, foram feitas algumas adaptações na classe da base de dados originalmente capturada. A classe dos dados foi ajustada conforme Tabela 11. Como os alunos aprovados e os reprovados por nota perseveraram até o da fim da disciplina, foi atribuído o número 1 para a classes desses alunos, indicando que o aluno não evadiu. Já os estudantes reprovados por frequência ou os que trancaram a disciplina foram considerados como evadidos, sendo atribuído o rótulo 0.

Tabela 11 – Modificações na Classe da Base de Dados

DE:	PARA:	
SITUAÇÃO	SITUAÇÃO	RÓTULO
APROVADO	CONCLUIU	1
REPROVADO POR NOTA		
REPROVADO POR FREQUENCIA	EVADIU	0
TRANCADO		

Fonte: Própria

4.4 Tecnologias Utilizadas

Para a implementação dos modelos preditivos foi utilizada uma biblioteca específica de AM chamada *scikit-learn* na linguagem *python* (PEDREGOSA et al., 2011) e a ferramenta WEKA (FRANK et al., 2004). Um dos algoritmos de classificação mais recomendados para o contexto apresentado foi o *Random Forest*, e por ser uma continuidade do algoritmo de AD, ambos foram selecionados para a construção dos modelos preditivos.

Foram realizados alguns testes com outros algoritmos de classificação disponíveis na literatura com objetivo de comparar qual deles apresenta maior precisão para resolução do problema.

No processo de desenvolvimento do método proposto foram utilizados as seguintes tecnologias:

- MySql: sistema de gerenciamento de banco de dados utilizado para acessar os logs de atividades do processo de programação dos alunos;
- CodeBench: Juiz *Online* que corrige automaticamente códigos-fonte, utilizado para coletar logs dos alunos;
- Scikit-Learn: desenvolvida em Python, é uma biblioteca com algoritmos de AM. Foi utilizada para criar os modelos preditivos;
- Weka: ferramenta desenvolvida pela University of Waikato na Nova Zelandia é composta por diversos algoritmos de AM e mineração de dados. Foi utilizada para a seleção dos atributos mais relevantes;
- Radon: biblioteca utilizada para calcular métricas do código, isto é, dado um código fonte ela faz cálculos do valor SLOC, número de comentários, McCabe's complexity e outros itens da matriz de aprendizagem;
- CodeMirror: embutido no CodeBench, é um editor de código responsável por capturar todos os eventos keystroke do processo de codificação do aluno.

5 RESULTADOS EXPERIMENTAIS

O presente capítulo consiste em apresentar os experimentos e resultados do método proposto, que visa não só fazer predição de evasão, mas sobretudo apresentar subsídios de informações que possam servir de auxílio para um melhor entendimento do processo de aprendizado do aluno na turma de IPC em relação a que fatores podem levar o aluno a evadir na disciplina.

5.1 Execução do Experimento I

Neste primeiro experimento, a base de dados submetida para **treinamento** do algoritmo J48 estava desbalanceada, esta base é a "Sessão1". Ainda, a classe binária encontrava-se dividida da seguinte forma, dentre os 407 alunos matriculados 322, perseveraram na disciplina, que corresponde a 79,11%, enquanto 85 desistiram da disciplina, isto é, 20,88% dos alunos evadiram. Para o **teste** do modelo preditivo foi utilizada a base da "Sessão2" , também não-balanceada, obtendo os resultados apresentados na Tabela 12 com uma acurácia de 79.8469%.

Tabela 12 – Resultados Sessão1 prevendo Sessão2 não-balanceada

Precision	Recall	Class	Situação
0,346	0,127	0	Evadiu
0,831	0,947	1	Persistiu

Fonte: Própria

O modelo preditivo de forma geral apresentou uma acurácia razoável com o algoritmo AD. No entanto, o resultado do *Precision* e *Recall* foram insatisfatórios na classificação dos alunos que evadiram, com apenas 34% e 12% respectivamente. Notou-se que o modelo preditivo não generalizou de forma satisfatória ao treinar com dados de uma sessão e ser testado com os dados de outra sessão. Diante desse resultado, foi utilizado para treinamento e teste a base da "Sessão1" não balanceada com uso do Cross-validation (Folds=10) obtendo os resultados da Tabela 13. Como resultado, o modelo preditivo atinge uma taxa de acerto 88.6305 % para inferir se o aluno irá evadir ou não, utilizando apenas os dados das duas primeiras semanas de aula (sessão1).

Tabela 13 – Resultados Sessão1 não-balanceado

Precision	Recall	Class	Situação
0,804	0,577	0	Evadiu
0,900	0,964	1	Persistiu

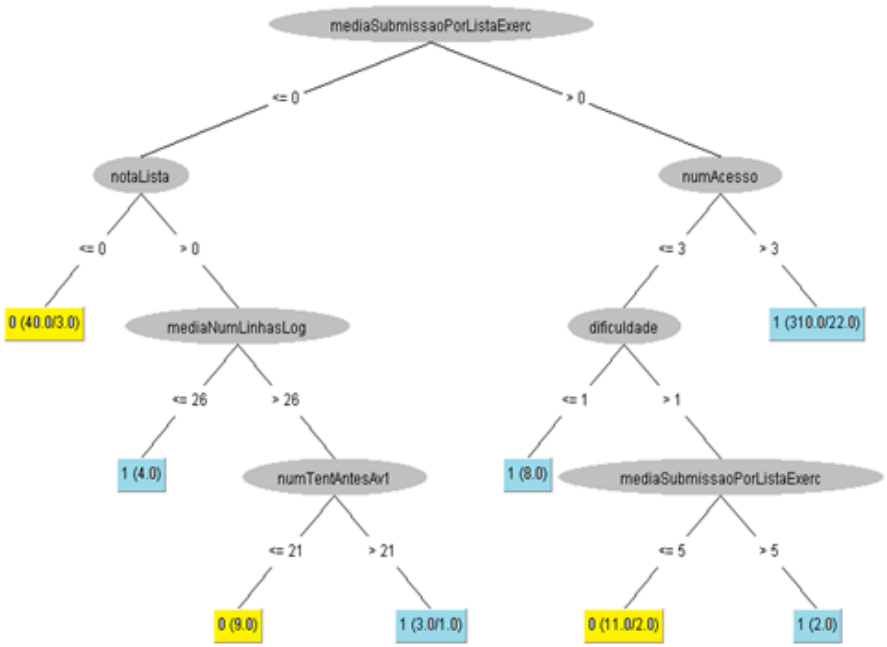
Fonte: Própria

Para que houvesse uma análise das relações existentes nos dados, a AD do resultado da Tabela 13 é impressa na Figura 11. Percebe-se que os estudantes que menos submeteram o código dos exercícios para correção, tiveram maior tendência a evadirem da disciplina. Em outras palavras, os alunos que realizaram os exercícios tiveram maior tendência a persistirem na disciplina.

De acordo com a análise, a quantidade de acesso do aluno é outro forte atributo para que o aluno não evada da disciplina, bem como a média de submissão por lista de exercício, em que mostra que a dedicação nos exercícios é de fundamental importância.

Por fim, apesar da acurácia de 88,6305% atingida ao realizar o treino e o teste na sessão1 transparecer um resultado bastante promissor, ao analisar a matriz de confusão na Tabela 14, observou-se que dentre os 78 alunos que evadiram da disciplina, o modelo preditivo conseguiu acertar 45, isto é 57,69%. E dos que persistiram até o fim da disciplina o modelo acertou 298 dos 309, perfazendo um percentual de 96,44% de acerto na amostragem. Dessa forma, percebeu-se que o modelo atingiu resultados promissores, mas com potencial para ser melhorado. Diante disso, viu-se a necessidade de realizar o balanceamento prévio da base de dados, a fim de que o classificador não ficasse propício a sempre estimar a classe majoritária.

Figura 11 – Árvore Sessão1 desbalanceada com método Cross-validation



Fonte: Própria

Tabela 14 – Resultados Matriz de Coincidência Sessão1 não-balanceado

a	**b**	Predição / Real
45	33	**a** = 0
11	298	**b** = 1

Fonte: Própria

5.2 Execução do Experimento II

Neste segundo experimento, com uso da ferramenta WEKA (HALL et al., 2009), os dados do **treinamento** e **teste** do modelo preditivo foram balanceados e posteriormente submetidos ao modelo em três etapas.

- Etapa 1: Primeiramente a base da Sessão-1 foi utilizada para **treinamento** do modelo preditivo. Na sequencia foram submetidas como **teste** as Sessões 2, 3 e 4;
- Etapa 2: No segundo momento a base da Sessão2 foi utilizada como **treinamento** do algoritmo e posteriormente foi utilizado em série as Sessões 3 e 4 como **teste**;

- Etapa 3: Finalmente a base da Sessão3 foi aplicada ao modelo para **treinamento** e realizado o **teste** com a Sessão4.

Optou-se em realizar o treinamento e teste com as quatro primeiras sessões, tendo em vista que após a quarta sessão, o semestre estará próximo do término e a informação se o aluno ira evadir ou não será de pouca relevância.

Essa nova investida para realizar o treinamento e teste em sessões diferentes, mas com a base balanceada, não teve resultados tão promissoras, visto que a taxa de acerto na classificação foi entre 60% e 71%. Com base nas observações realizadas no Experimento I e na Tabela 15, notou-se que o modelo preditivo atinge acurácias mais altas ao ser treinado e testado com os dados de uma mesma sessão.

Uma explicação para este fenômeno é que em sessões diferentes os alunos estão estudando tópicos diferentes. Para ilustrar, na sessão1 os alunos estudam variáveis e estrutura sequencial, enquanto que na sessão2 os alunos estudam estrutura de decisão. Dessa forma, os padrões existentes nos dados podem mudar juntamente com a mudança do conteúdo estudado na disciplina de IPC.

Observe ainda que em aprendizagem de máquina, em geral, mais dados significa maior poder de generalização. Entretanto, neste experimento, os resultados contradizem essa máxima, já que ao treinar e testar em uma mesma seção a divisão de treino e teste é feito com os dados da mesma sessão, diminuindo a quantidade de instâncias para treino, enquanto que ao treinando com os dados de uma sessão e testando com os dados de outra sessão, nos mantemos os dados de uma sessão inteira apenas para treinamento.

Tabela 15 – Resultados Entre Sessões balanceadas

TREINO	TESTE	INSTANCIA Classificado Correto		INSTANCIA Classificado Incorreto		Precision 0	Precision 1	Recall 0	Recall 1
Sessão1	Sessão2	91	64,0845%	51	35,9155%	0,628	0,656	0,690	0,592
Sessão1	Sessão3	81	71,0526%	33	28,9474%	0,662	0,800	0,860	0,561
Sessão1	Sessão4	63	64,2857%	35	35,7143%	0,603	0,733	0,837	0,449
Sessão2	Sessão3	71	62,2807%	43	37,7193%	0,606	0,646	0,702	0,544
Sessão2	Sessão4	57	58,1633%	41	41,8367%	0,580	0,583	0,592	0,571
Sessão3	Sessão4	68	69,3878%	30	30,6122%	0,661	0,744	0,796	0,592

Fonte: Própria

5.3 Execução do Experimento III

Levando em consideração que o modelo preditivo apresentava melhores resultados treinando com os dados de uma mesma sessão e com a base de dados balanceada, no Experimento

III foi utilizada essa abordagem com a intenção de inferir se o aluno iria evadir ou não utilizando os dados apenas da sessão1, isto é, das duas primeiras semanas de aula.

Ainda utilizando a ferramenta WEKA (HALL et al., 2009) foram analisados os resultados de cinco diferentes algoritmos baseados em árvore de decisão. Os resultados podem ser vistos na Tabela 16. O algoritmo que apresentou melhor resultado foi o DecisionStump com 72% de precisão. Observe que no Experimento II (Tabela 15), treino-se com os dados da sessão1 e testou-se com os dados da sessão2 e obteve-se apenas 64,08% de acurácia, o que mostra que, com base nos dados analisados neste estudo, resultados mais promissores são alcançados ao treinar e testar o algoritmo com os dados da mesma sessão.

Tabela 16 – Resultados de algoritmos baseados em árvore de decisão

ALGORITMO	INSTÂNCIA Classificado Correto		INSTÂNCIA Classificado Incorreto		Precision 0	Precision 1	Recall 0	Recall 1
J48	115	67.6471 %	55	32.3529 %	0,714	0,650	0,588	0,765
DecisionStump	123	72.3529 %	47	27.6471 %	0,797	0,679	0,600	0,847
Random Forest	117	68.8235 %	53	31.1765 %	0,705	0,674	0,647	0,729
Rule Decision Table	122	71.7647 %	48	28.2353 %	0,761	0,687	0,635	0,800

Fonte: Própria

5.4 Execução do Experimento IV

Finalmente, no experimento IV, ainda usando os dados da sessão 1 para treinamento e teste de modo balanceado, o algoritmo árvore de decisão foi submetido a um processo de otimização utilizando a técnica de *GridSearch*. A escolha de otimizar á AD se deu pelo fato de além do algoritmo ser eficiente, ele proporciona uma fácil interpretação das relações existentes nos dados.

O GridSearch é uma técnica amplamente utilizada na literatura para ajuste de parâmetros. Nessa técnica é realizada uma busca exaustiva pelo espaço de soluções. Destaca-se que essa técnica é viável para um espaço de busca que não é de ordem exponencial, pois do contrário ela pode demandar muito tempo para a execução. Dessa forma, nesta pesquisa apenas 5 parâmetros foram escolhidos para realizar ajuste, os quais podem ser vistos na Tabela 17. O *min_samples_split* estabelece um valor mínimo de amostras para realizar a ramificação de um nodo, o *min_samples_leaf* estabelece a quantidade mínima de amostras que as folhas devem ter, o *max_features* estabelece o percentual de atributos que devem ser considerados no momento em que é realizada a divisão de um nodo da árvore e o *max_depth* designa a profundidade máxima da árvore.

Tabela 17 – Parâmetros da Árvore de Decisão

Parâmetros da Árvore de Decisão	**Valores Testados**
min_samples_split	[5, 8, 10, 14, 16, 18, 20, 23, 26, 30]
max_features	[0.3, 0.5, 0.6, 0.7, 0.8, 0.9]
min_samples_leaf	[5, 7, 9, 10, 15, 17, 18, 20]
max_depth	[2, 3, 4, 5, 6, 7, 8]

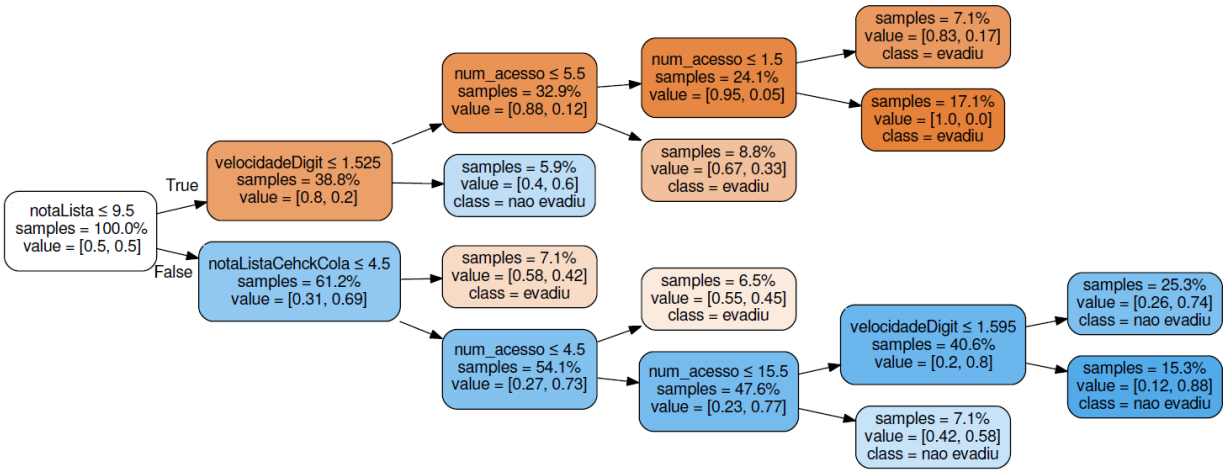
Fonte: Própria

Após o processo de otimização, o modelo preditivo alcançou 80% de acurácia. Ao comparar esse resultado com a acurácia obtida no Experimento 1, onde foi a AD foi treinada e testada bom como com a sessão, houve uma redução da acurácia, entretanto neste experimento as métricas *Recall* e *Precision* se equilibraram. Em outras palavras, com a base balanceada o modelo preditivo foi capaz de identificar os alunos que evadiram e não evadiram com uma taxa de acerto similar.

Diante desse resultado com a base de dados balanceada, optou-se por realizar uma análise da árvore de decisão construída, a fim de que fosse possível interpretar quais condições podem levar os alunos observados neste estudo de caso à evasão e a conclusão da disciplina. Essa árvore pode ser vista na Figura 12, onde os nodos no cor laranja representam a estimativa de evasão da árvore e os nodos azuis representam a estimativa de não evasão. Destaca-se que quanto mais escuro for a cor de um nodo filho, mais pura foi a divisão realizada no nodo pai. Em outras palavras, quanto mais escuro for a cor do nodo, maior é o nível de confiança na predição.

Analisando mais estritamente a Figura 12, que apresenta a árvore de decisão construída, percebe-se que a **nota_lista** foi o atributo mais relevante para o modelo e por isso ele foi colocado na raiz da árvore. Assim, se a **nota_lista** for menor ou igual a 0,95 e o atributo **velocidadeDigit** for maior que 0,525, então o aluno é classificado como "não evadiu". Em uma análise em relação à esse regra, permite-nos entender que alunos que vão bem nas listas de exercício e codificam rapidamente, tem mais chances de não evadir. Observe que o fato do aluno codificar rápido pode indicar que ele já possui uma experiência prévia em programação.

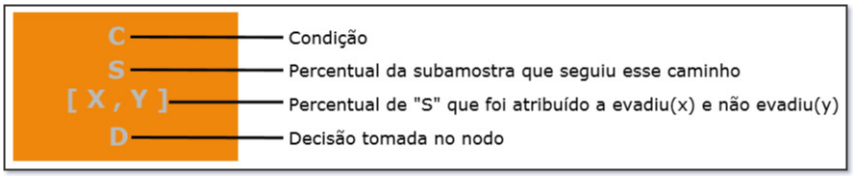
Figura 12 – Árvore de Decisão no scikit-learn Balanceada



Fonte: Própria

Os nodos da árvore podem conter quatro campos, conforme a Figura 13. Na parte superior de cada nodo tem a condição “C” que é utilizada para o estimador tomar decisão que pode ser verdadeira ou falsa. A letra "S"(*samples*) é o percentual da subamostra que foi encaminhado para um nodo filho após a divisão tomada baseada na condição “C”. Logo abaixo, o [x, y] significa quantos porcento foram atribuídos a decisão, onde x contém a porcentagem da subamostra que evadiu e y traz o valor das subamostra que não evadiu. Por fim, o “D” aparece apenas nas folhas da árvore, representando a decisão tomada baseada nas condições anteriores, isto é, se o aluno evadiu ou não. Vale frisar que somente irá acontecer a "Condição"nos nodos internos e a "Decisão"somente nas folhas da árvore.

Figura 13 – Nodo da Árvore de Decisão



Fonte: Própria

No entanto, quando o aluno tira uma **nota_lista** abaixo de 0,95, tem uma **velocidadeDigit** abaixo de 0,529 e tem um número muito baixo de acessos no ACAC (**num_acessos**), então a probabilidade dele evadir é muito alta. Isso pode ser notado nas folhas que ficam na parte superior da árvore (cor laranja), onde o nível de confiança da decisão é de 83% e 100%.

Por outro lado, se a **nota_lista** for maior que 0,95, o aluno tiver acessado muitas vezes o ACAC e a **nota_Lista_Check_Cola** for maior que 0,45, então o aluno é classificado como "não evadiu". Note que essa regra mostra que alunos aplicados, que resolvem as listas de exercícios, que acessam com frequência o ACAC e que tem muitas linhas de *log* na resolução dos exercícios (**nota_Lista_Check_Cola**) nas duas primeiras semanas de aula costumam concluir a disciplina. Entretanto, mesmo que o aluno tenha teoricamente resolvido muitos exercícios, mas tenha gerado poucas linhas de *log* na resolução (perceba que ao gerar poucas linhas de *log* existe um forte indicativo que o aluno pode simplesmente ter copiado e colado o código) e tenha acessado poucas vezes o ACAC, então este aluno é classificado como "evadiu".

Finalmente, se compararmos a AD construída no Experimento 1 (Figura 11) com a AD construída neste Experimento (Figura 12), percebe-se que a última utilizou menos atributos e foi capaz de distinguir melhor os alunos que evadiram dos que não evadiram. Isso mostra que o processo de otimização se mostrou relevante para a acurácia do modelo preditivo.

5.5 Detalhes de Implementação

Nesta seção são descrita os principais ponto do modelo preditivo utilizado no experimento III, o qual foi desenvolvido na linguagem de programação Python com uso do scikit-learn. Sua estrutura conforme a Figura 14 compõe-se com as seguintes características:

Figura 14 – Estrutura do Algoritmo no Python

```
from sklearn.model_selection import GridSearchCV

from sklearn import tree

def random_search(X, y):
    clf = tree.DecisionTreeClassifier()
    k_range = [5, 7, 9, 10, 15, 17, 18, 20]
    f_range = [8, 10, 14, 16, 20, 25, 26, 30]
    features_range = [.3, .5, .7, .95]

    param_grid = dict(min_samples_split=f_range, min_samples_leaf=k_range, max_features=features_range, max_depth = [2, 3, 4, 5, 6, 7])

    print("loading...")
    grid = GridSearchCV(clf, param_grid, cv=3, scoring='accuracy', n_jobs=-1, verbose=3)
    grid.fit(X, y)

    return grid

path1 = 'Normalizado_Balanceado_1.csv'

import pandas as pd

df1 = pd.read_csv(path1)

df1 = df1[['lloc','num_acesso','notaLista', 'notaListaCehckCola', 'velocidadeDigit', 'DESCR_SITUACAO_1']]

X, y = df1.drop('DESCR_SITUACAO_1', axis=1).values, df1['DESCR_SITUACAO_1'].values

grid = random_search(X, y)

print('\033[1m' + 'Resultado:' + '\033[0m')
print(grid.best_score_)
print(grid.best_params_)
print('desvio padrao: ',grid.cv_results_['std_test_score'][0])
```

Fonte: Própria

Na parte do código ***"from sklearn.model_selection import GridSearchCV"*** (linha 1) o *GridSearchCV* implementa um método de "ajuste" e "pontuação" com uma busca exaustiva sobre valores de parâmetros especificados para o modelo preditivo(PEDREGOSA et al., 2011).

Já na parte **"clf = tree.DecisionTreeClassifier()"** (linha 3) a variável *clf* recebe o classificador da árvore de decisão.

Ainda, no seguimento da Figura 15, o construtor **dict()** da linha 4 forma um dicionário de sequencia de pares de valores-chaves(PEDREGOSA et al., 2011), cujo cada parâmetro entre parênteses estão detalhados na Tabela 17.

Figura 15 – Parâmetros do método *GridSearch*

```
param_grid = dict(min_samples_split=f_range, min_samples_leaf=k_range,
                  max_features=features_range, max_depth = [2, 3, 4, 5, 6, 7])
```

Fonte: Própria

Na Figura 16 a variável *grid* na linha 6 recebe *GridSearchCV*, cujos parâmetros de *GridSearchCV* são as seguintes (PEDREGOSA et al., 2011):

Figura 16 – Método *GridSearchCV*

```
print("loading...")
grid = GridSearchCV(clf, param_grid, cv=3, scoring='accuracy', n_jobs=-1, verbose=3)
```

Fonte: Própria

- clf: é objeto estimador, no caso o classificador árvore de decisão;
- param_grid: Dicionário com nomes de parâmetros como chaves e listas de configurações de parâmetros para testar como valores, ou uma lista de tais dicionários, em cujo caso as grades abrangidas por cada dicionário na lista são exploradas. Isso permite pesquisar sobre qualquer sequência de configurações de parâmetros;
- cv: Determina a estratégia de divisão de validação cruzada;
- scoring: uma chamada para avaliar as previsões no conjunto de teste;
- n_jobs: Número de trabalhos a serem executados em paralelo;
- verbose: Controla o detalhamento: quanto maior, mais mensagens.

Na sequencia o **grid.fit(X, y)** (linha 7) calcula os parâmetros de aprendizagem do modelo, e a variável *df1* de **df1 = pd.read_csv(path)** (linha 11) ler o arquivo CSV (separado por vírgulas) no *DataFrame* (PEDREGOSA et al., 2011).

6 CONSIDERAÇÕES FINAIS

Este trabalho propôs e validou um método para inferir a evasão de aluno de IPC que utilizam juízes online. A intenção principal foi realizar a predição ainda no início da disciplina de modo que a instituição ou mesmo o professor pudesse tomar medida proativa para evitar que o aluno acabasse evadindo. Além disso, foram analisados quais comportamentos de programação podem levar o aluno a evasão.

Dado os diferentes aspectos do problema apresentado na pesquisa, foram utilizadas combinações de técnicas de AM. O algoritmo C45 (árvore de decisão) se mostrou promissor para a tarefa em destaque nesta pesquisa, pois além de possibilitar uma precisão de 80% na predição de evasão de alunos ainda nas duas primeiras semanas de aula, ele possibilitou a interpretação de quais comportamentos de programação tiveram relação com evasão precoce do aluno. Através da análise da árvore de decisão, percebeu-se que alunos menos aplicados nas listas de exercícios, que acessam poucas vezes o juiz online e que codificam em uma frequência mais baixa, têm uma tendência para evasão.

A aplicação e análise dos experimentos poderá ser de grande relevância, pois os professores e as próprias instituições de ensino cientes da possibilidade de evasão do aluno, poderão adotar medidas eficazes para reduzir esses índices. Logo, esta pesquisa poderá contribuir positivamente para alunos e professores, tendo em vista que se propõe a realizar, por meio do modelo preditivo proposto, a mensuração antecipada dos alunos que provavelmente evadirão da disciplina.

6.1 Limitações e ameaças à validade

A limitação inicial para aplicação do método foi relacionada à base de dados utilizada, tendo em vista que os dados foram coletados apenas no semestre 2016.1. Apesar dos dados terem sido coletados a partir de 9 cursos diferentes, todos os cursos são de uma mesma instituição de ensino. Com uma base de dados maior e interinstitucional, acredita-se que o resultado da classificação poderá ser aprimorado, uma vez que haverá maior variação nos dados a amostra terá maior probabilidade de representar a população. Em outras palavras, com uma base de dados multicontextual, o método terá uma maior poder de generalização.

Ademais, os resultados obtidos pelo modelo preditivo poderão ser influenciados ao erro na classificação devido algumas anomalias na base de dados. Para ilustrar, um aluno com capacidade para concluir a disciplina, pode evadir por motivos externos, como por exemplo um problema pessoal. Por outro lado, um aluno pode concluir a disciplina mesmo com capacidade acadêmica abaixo da média, fazendo com que o algoritmo aprenda um perfil de programação

errado e realize uma falsa predição. Tais situações estão em desacordo com os padrões das classes aprendidas pelos classificadores.

Dessa forma, para garantir um resultado eficiente na predição é necessária a replicação do método na base de dados de outras instituições de ensino que utilizam Juiz *online*.

6.2 Trabalhos Futuros

Como trabalhos futuros, objetiva-se fazer um teste com aprendizagem de máquina profunda utilizando os dados de mais turmas de IPC, a fim de melhorar a precisão do modelo preditivo.

Pretende-se ainda realizar um estudo de controle explorando os efeitos de o professor saber antecipadamente a probabilidade do aluno evadir e se essa informação pode contribuir de fato para a redução do índice de evasão das turmas.

REFERÊNCIAS

ALVES, I. M. Competências e habilidades em informática para o trabalho. 2012.

ARTUSO, A. R. Entropias de shannon e rényi aplicadas ao reconhecimento de padrões. *Revista CIATEC-UPF*, v. 3, n. 2, p. 56–72, 2012.

BACICH, L.; NETO, A. T.; TREVISANI, F. de M. *Ensino híbrido: personalização e tecnologia na educação*. [S.l.]: Penso Editora, 2015.

BATISTA, G.; MONARD, M. C. Sniffer: um ambiente computacional para gerenciamento de expe-rimentos de aprendizado de máquina supervisionado. *Proceedings of the I WorkComp Sul*, 2004.

BATISTA, G. E. d. A. P. et al. *Pré-processamento de dados em aprendizado de máquina supervisionado*. Tese (Doutorado) — Universidade de São Paulo, 2003.

BLIKSTEIN, P. Using learning analytics to assess students' behavior in open- ended programming tasks. In: ACM. *Proceedings of the 1st international conference on learning analytics and knowledge*. [S.l.], 2011. p. 110–116.

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

BRITO, D. M. de et al. Predição de desempenho de alunos do primeiro período baseado nas notas de ingresso utilizando métodos de aprendizagem de máquina. In: *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*. [S.l.: s.n.], 2014. v. 25, n. 1, p. 882.

CHATTI, M. A. et al. A reference model for learning analytics. *International Journal of Technology Enhanced Learning*, Inderscience Publishers, v. 4, n. 5-6, p. 318–331, 2012.

CHAVES, J. O. M. et al. Integrando moodle e juízes online no apoio a atividades de programação. In: *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*. [S.l.: s.n.], 2013. v. 24, n. 1, p. 244.

CHAWLA, N. V.; JAPKOWICZ, N.; KOTCZ, A. Special issue on learning from imbalanced data sets. *ACM Sigkdd Explorations Newsletter*, ACM, v. 6, n. 1, p. 1–6, 2004.

DAYCHOUM, M. *40+ 10 ferramentas e técnicas de gerenciamento*. [S.l.]: Brasport, 2013.

DETONI, D.; CECHINEL, C.; ARAÚJO, R. M. Modelagem e predição de reprovação de acadêmicos de cursos de educação a distância a partir da contagem de interações. *Revista Brasileira de Informática na Educação*, v. 23, n. 3, 2015.

DWAN, F.; OLIVEIRA, E.; FERNANDES, D. Predição de zona de aprendizagem de alunos de introdução à programação em ambientes de correção automática de código. In: *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*. [S.l.: s.n.], 2017. v. 28, n. 1, p. 1507.

FRANK, E. et al. Data mining in bioinformatics using weka. *Bioinformatics*, Oxford University Press, v. 20, n. 15, p. 2479–2481, 2004.

GALVÃO, L.; FERNANDES, D.; GADELHA, B. Juiz online como ferramenta de apoio a uma metodologia de ensino híbrido em programação. In: *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*. [S.l.: s.n.], 2016. v. 27, n. 1, p. 140.

GAMA, J. Árvores de decisão. *Palestra ministrada no Núcleo da Ciência de Computação da Universidade do Porto, Porto*, 2002.

GARCÍA, E. et al. Drawbacks and solutions of applying association rule mining in learning management systems. In: *Proceedings of the International Workshop on Applying Data Mining in e-Learning (ADML 2007), Crete, Greece*. [S.l.: s.n.], 2007. p. 13–22.

HALL, M. et al. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, ACM, v. 11, n. 1, p. 10–18, 2009.

IHANTOLA, P. et al. Educational data mining and learning analytics in programming: Literature review and case studies. In: ACM. *Proceedings of the 2015 ITiCSE on Working Group Reports*. [S.l.], 2015. p. 41–63.

KANTORSKI, G. Z. et al. Uma visão do futuro: Previsão de evasão em cursos de graduação presenciais de universidades públicas: O caso do curso de zootecnia. INPEAU/UFSC, 2015.

MANHÃES, L. M. B. et al. Previsão de estudantes com risco de evasão utilizando técnicas de mineração de dados. In: *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*. [S.l.: s.n.], 2011. v. 1, n. 1.

MARTINS, L. C.; LOPES, D. A.; RAABE, A. Um assistente de predição de evasão aplicado a uma disciplina introdutória do curso de ciência da computação. In: *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*. [S.l.: s.n.], 2012. v. 23, n. 1.

MONARD, M. C.; BARANAUSKAS, J. A. Conceitos sobre aprendizado de máquina. *Sistemas Inteligentes-Fundamentos e Aplicações*, v. 1, n. 1, 2003.

NORVIG, P.; RUSSELL, S. *Inteligência Artificial, 3a Edição*. [S.l.]: Elsevier Brasil, 2014.

OLSON, D. L.; DELEN, D. *Advanced data mining techniques*. [S.l.]: Springer Science & Business Media, 2008.

PAES, R. de B. et al. Ferramenta para a avaliação de aprendizado de alunos em programação de computadores. In: *Anais dos Workshops do Congresso Brasileiro de Informática na Educação*. [S.l.: s.n.], 2013. v. 2, n. 1.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011.

PEREIRA, D. F. d. S. *Um estudo de correlação entre o perfil de usuário e o progresso individual dos alunos em disciplinas iniciais de programação*. 2016. 24 p.

POWERS, D. M. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. Bioinfo Publications, 2011.

RIGO, S. J. et al. Aplicações de mineração de dados educacionais e learning analytics com foco na evasão escolar: oportunidades e desafios. *Revista Brasileira de Informática na Educação*, v. 22, n. 01, p. 132, 2014.

ROCHA, P. S. et al. Ensino e aprendizagem de programação: análise da aplicação de proposta metodológica baseada no sistema personalizado de ensino. *RENOTE*, v. 8, n. 3, 2010.

SOUZA, C. T.; SILVA, C. da; GESSINGER, R. M. Um estudo sobre evasao no ensino superior do brasil nos últimos dez anos. In: *Congresos CLABES*. [S.l.: s.n.], 2016.