Machine Learning como ferramenta de auxílio na predição de rendimento de alunos do curso de Sistemas para Internet

Trabalho de Conclusão do Curso Superior de Tecnologia em Sistemas Para Internet

Ariel Corrêa Pezzoli
Orientador(a): Karen Selbach Borges

Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul (IFRS)
Campus Porto Alegre
Av Cel Vicente, 281, Porto Alegre – RS – Brasil

ariel_pezzoli@hotmail.com,karen.borges@poa.ifrs.edu.br

Abstract. *In IT courses, there is a lot of evasion and retention of students. A hypothesis is that the incoming students isnt in the final stage of cognition level, which is identified by Jean Piaget as Formal Operative. There are several papers that try to explain the reasons of students difficulties. However, most of such studies try to relate social aspects and academic performance. This work focus on the Programming Logic course at IFRS and on Operational tests of François Longeot, to identify relationships among cognition level of the students and its academic performance. This paper will apply machine learning techniques under the data obtained through the performance of IFRS Programming Logic students, and in the results of the Operational Tests of François Longeot. With this it hopes to identify those with a tendency to evade and disapprove, and, with that, contribute to improve the quality of the future IT professionals*.

Resumo. Na educação superior, especialmente em cursos de TI, existe muita evasão e retenção de alunos. Como hipótese, acredita-se que um dos motivos é que os alunos ingressantes nos cursos ainda não se encontram no último estágio de desenvolvimento cognitivo, o qual foi identificado por Jean Piaget como Operatório Formal. Existem diversos trabalhos que estudam os motivos pelos quais os alunos têm dificuldades, porém, em sua maioria, esses estudos levam em consideração aspectos sociais e desempenho acadêmico. Este trabalho irá aplicar técnicas de machine learning sobre os dados obtidos a partir do desempenho dos alunos na disciplina de Lógica de Programação do IFRS e nos resultados dos Testes Operatórios de François Longeot, os quais irão auxiliar na identificação do nível cognitivo dos alunos. Espera-se, assim, identificar aqueles com propensão à evasão ou retenção e, com isso, contribuir para melhorar a qualidade da formação dos futuros profissionais de TI.

1. Introdução

Uma preocupação institucional relacionada ao curso de Tecnologia em Sistemas para Internet do IFRS é o grande índice de evasão, reprovação e retenção por parte dos alunos. Entretanto, diversas instituições de ensino superior compartilham dessa realidade. Um levantamento realizado em 2012 pelo Sindicato das Entidades Mantenedoras de Estabelecimentos de Ensino Superior no Estado de São Paulo mostrou que "a cada três alunos que entram no curso de sistemas de informação, apenas um recebe o diploma. Em Ciência da Computação, a cada quatro alunos que entram no curso, apenas um termina" (HOED, 2012 p. 1).

Temos como hipótese que essa situação possa estar ocorrendo porque alguns alunos, no momento do ingresso no curso superior, ainda não alcançaram o último estágio de desenvolvimento cognitivo, o qual foi denominado por Piaget (1973) como operatório formal. Szymanski (2011), mostrou em sua pesquisa que a maioria dos alunos do curso de Atendente em Enfermagem, embora adultos, ainda se encontravam em nível operatório concreto. O mesmo pode estar acontecendo com a área da Computação, que demanda dos seus estudantes um bom raciocínio lógico e uma boa capacidade de reflexão para conseguir se desenvolver no curso de Sistemas para Internet, aqueles que ainda não atingiram esse patamar podem reprovar ou se sentir desestimulados, tornando-se propensos à evasão.

Este é um dos assuntos pesquisados pelo LECC (Laboratório de Estudos Cognitivos Apoiados por Computação) do IFRS. Com o auxílio dos Testes Operatórios de François Longeot, estão sendo estudadas relações entre o nível de desenvolvimento cognitivo e o desempenho na disciplina de Lógica de Programação. Esses testes permitem identificar em que nível cada aluno se encontra: nível operatório concreto ou formal, de acordo com a teoria de Jean Piaget. Ao se identificar o nível em que cada aluno recém-ingressado no curso se encontra, torna-se possível que a instituição ofereça formas para ajudar àqueles que ainda não se encontram no nível formal, seja com aulas extras, acompanhamento psicológico, mudanças nas metodologias de aprendizagem ou outras técnicas que a instituição julgue pertinente.

Ainda não há um sistema computacional capaz de prever, com um bom grau de acerto, quais alunos terão propensão à evasão e retenção. No caso deste trabalho, está sendo desenvolvido um sistema utilizando o conceitos de aprendizado de máquina, onde o software, depois de treinado com o uso de uma base de dados históricos, poderá identificar o nível de cada novo aluno do curso, reconhecendo assim, quem terá mais dificuldades e as possíveis evasões. O diferencial desse trabalho reside no fato de que a base de dados está sendo construída a partir dos conceitos finais na disciplina de Lógica de programação e dos resultados dos alunos nos Testes Operatórios. Dependendo do resultado de cada teste operatório de Longeot, o aluno tenderá a se encaixar em algum grupo, seja ele para os que provavelmente não terão dificuldades (aprovados), seja para

aqueles com mais dificuldades (reprovados), ou então para aqueles com tendência à deixarem o curso (evadidos).

Este trabalho foi dividido em seis seções. Na seção 2 são abordados trabalhos desenvolvidos com propostas similares a este. Na seção 3 são apresentados os fundamentos teóricos que sustentam a presente pesquisa. Na seção 4 e 5 são apresentadas, respectivamente, a metodologia utilizada e a descrição do sistema. Na seção 6 estão descrito os resultados e na seção 7 as conclusões do projeto.

2. Trabalhos Relacionados

O uso de tecnologias para ter um melhor entendimento do porquê alunos evadem ou retém no ensino superior tem sido amplamente pesquisado. Diversos trabalhos relatam que a mineração de dados permite uma visão mais clara do ambiente educacional onde é aplicada, proporcionando que estratégias mais eficazes sejam elaboradas para lidar com os alunos.

Cerveira (2008), fez uso de mineração de dados, com auxílio da ferramenta *RapidMiner,* para identificar perfis de alunos com risco de evasão ou reprovação, gerando assim, subsídios para a criação de alertas que indicassem quais dos alunos poderiam necessitar de ajuda. A pesquisadora concluiu que esses alertas automatizados contribuem no acompanhamento dos estudantes e auxiliam na elaboração de melhores técnicas de orientação.

Campello e Lins (2008), tiveram como objetivo apresentar uma metodologia para criar estratégias de resolução do problema de evasão e retenção em cursos de Engenharia de Produção da UFPE (Universidade Federal de Pernambuco). Utilizaram-se, para tanto, do artifício da clusterização, que consiste em gerar agrupamentos de alunos para identificar perfis. Os pesquisadores foram capazes de identificar as principais causas de evasão e retenção e propor ações para minimizar o problema. Como conclusão do trabalho, destacaram que cada curso têm suas peculiaridades, não cabendo assim a generalização desses estudos para outras instituições.

Natek e Zwilling (2014), buscaram responder questões do tipo: É possível prever a taxa de sucesso dos alunos matriculados? Existem características específicas que podem ser associadas ao sucesso dos alunos? Existem dados disponíveis que possam ajudar a prever o sucesso do aluno? Para responder a tais perguntas, utilizaram a mineração de dados, com ferramentas como MS Excel Table tools e Weka. Ao final do estudo, os pesquisadores conseguiram responder suas questões iniciais e concluíram que a mineração tem potencial para se tornar parte importante de sistemas de gestão de conhecimento em instituições de ensino superior.

3. Fundamentação Teórica

Considerando que este trabalho une áreas distintas como a ciência de dados e a epistemologia genética, se faz necessário esclarecer os elementos associados a cada uma dessas áreas e relacionados à presente pesquisa.

3.1 Testes Operatórios Coletivos

Os testes de François Longeot aparecem como instrumento de avaliação do nível de desenvolvimento cognitivo em diversas pesquisas, tais como as desenvolvidas por Cantelli, Borges e Assis (2005); Camargo (1990); Souza e Macedo (1986); Lemos e Queiroz, (2015), entre outros. Longeot (1979) em sua pesquisa sobre o desenvolvimento de uma Escala do Desenvolvimento do Pensamento Lógico (*L'Échelle du Développement de la Pensée Logique* - EDPL) elaborou, além de um conjunto de provas operatórias, testes de papel e lápis de 'operações formais'. Esses testes, chamados de Testes Operatórios Coletivos (*Tests Opératoires Collectifs* - TOC), são complementares e cada um deles possui várias questões que possibilitam raciocínios próprios aos diferentes estágios. Depois de avaliados pelo experimentador, os testes situam o sujeito em um dos níveis da EDPL, as quais são: concreto A, concreto B, pré-formal, formal A e formal B. (BORGES, 2018 p. 54-57) Os TOCs são três, conforme mostra a Tabela 1

Tabela 1 - Descrição dos Testes

Nome do Teste	Conteúdo do Teste	Resultados
TOFLP Teste das operações formais em relação a lógica proposicional	6 problemas de nível operatório concreto 7 problemas de nível operatório formal	até 5 pontos, operatório concreto; de 6 a 8 pontos, operatório formal A; acima de 9 pontos, operatório formal B.
TOFC Teste das operações formais combinatórias	3 problemas de nível operatório concreto 5 problemas de nível operatório formal	até 3 pontos, operatório concreto; de 4 a 5 pontos, operatório formal A; acima de 6 pontos, operatório formal B.
TOFP Teste das operações formais em relação a lógica das	4 problemas de nível operatório concreto, 2 problemas de nível pré-formal	até 3 pontos, operatório concreto; de 4 a 5 pontos, pré-formal;

probabilidades ou proporções	5 problemas de nível operatório formal.	acima de 6 pontos, operatório formal.

3.2 Mineração de Dados

Mineração de dados é uma das etapas do processo conhecido como KDD (*Knowledge Discovery in Databases* – Descoberta de Conhecimento em Bases de Dados). Segundo FAYYAD (apud BRITO, 2012), "KDD é um processo de várias etapas, não trivial, interativo e iterativo, para identificação de padrões compreensíveis, válidos, novos e potencialmente úteis a partir de grandes conjuntos de dados". Esta etapa consiste em extrair informações relevantes de dentro de um grande conjunto de dados, a fim de agregar valor a elas. Seu processo necessita que os dados sejam organizados e submetidos a técnicas capazes de identificar aspectos que provavelmente o ser humano não conseguiria, seja pela enorme quantidade de dados ou por suas complexas relações. A sua organização proporciona que padrões e associações possam ser descobertos e que mudanças e anomalias nos dados minerados sejam identificados. As informações, depois de organizadas e tratadas, servirão de alicerce para gerar estratégias mais eficazes àquilo que for interessante aos analistas.

Antes de começar a minerar, existem algumas etapas que precisam ser cumpridas. Primeiramente precisa-se definir qual o problema a ser resolvido e que conjunto de dados será utilizado para tal. Com o problema definido, parte-se para a preparação dos dados, onde inconsistências como entradas ausentes ou incorretas deverão ser corrigidas. Dados repetidos e desnecessários para a análise deverão ser eliminados. Também é preciso identificar quais dados são mais relevantes dentro do seu contexto para a resolução do problema. Definido o problema e qual a fonte que será usada para a mineração, é preciso definir a técnica mais apropriada para realizar esta tarefa. Ressaltando que cada técnica possui suas próprias características e é necessário ter o conhecimento de seu funcionamento e objetivo, para assim conseguir interpretar os resultados de maneira correta.

Algumas das técnicas são: Associação, Regressão Linear, Classificação, Previsão e Clusterização. Dentre essas técnicas o foco estará na Clusterização, e para efeitos de comparação está sendo utilizada a Classificação. A Classificação consiste em examinar uma certa característica nos dados e atribuir uma classe previamente definida. Por exemplo, é possível classificar carros em diferentes tipos (sedan, 4x4, conversível) identificando atributos diferentes (número de lugares, formato do carro, rodas motrizes). Dado um novo carro, você pode aplicá-lo a uma classe específica comparando os atributos com as definições já conhecidas. Já a Clusterização é uma técnica capaz de criar agrupamentos de dados a partir de um determinado conjunto de informações, utilizando aprendizado de máquina não supervisionada. Digamos que o objetivo é classificar flores, e para isso utilizamos como base, dados do tipo: tamanho da pétala e

tamanho da semente. Com essas características, grupos que contêm pétalas de tamanho pequeno e semente pequena, podem ser formados, bem como grupos com sementes grandes e pétalas pequenas. Ou seja, objetos com alta similaridade devem se encontrar no mesmo agrupamento, e objetos com pouca similaridade devem se distanciar.

Após selecionar a técnica desejada e mais apropriada, e com os padrões desejados identificados, cabe aos analistas interpretarem as informações obtidas, seja para resolver os problemas iniciais ou pela descoberta de informações que até então não estavam visíveis. Vale lembrar que sempre que os objetivos da mineração ou o banco de dados utilizado forem modificados, deve-se refazer todos os processos para readequar às necessidades. Por fim, os conhecimentos obtidos são armazenados e ficam a disposição para serem usados da forma que os detentores desse conhecimento acharem mais apropriado.

4. Metodologia

Os dados que irão alimentar o algoritmo são colhidos de planilhas Excel, geradas a partir de formulários do Google Docs utilizados na aplicação dos testes. Estes formulários podem ser acessados a partir dos seguintes endereços eletrônicos:

- TOFP - https://goo.gl/forms/OQ5Xm6vZp69eWJU73
- TOFLP - https://goo.gl/forms/Ygp9peBoPv4nIWb53
- TOFC - https://goo.gl/forms/VpYrTkYmD5wkjDK13

Os testes, depois de resolvidos pelos alunos, são avaliados e os resultados são armazenados em um banco de dados relacional. Está sendo desenvolvido, por outro pesquisador do LECC, um sistema Web que será responsável pela aplicação e avaliação dos testes.

Atualmente conta-se com os resultados das turmas de Lógica de Programação dos semestres 2016/1, 2016/2 e 2018/2. A seguir, são detalhados os procedimentos aplicados sobre essa base de dados históricos, referente a metodologia de descoberta de conhecimento em base de dados - KDD.

4.1 Definição dos Dados

Para esse estudo, definiu-se que os dados relevantes para mineração seriam os dados obtidos a partir dos testes operatórios e da disciplina de lógica de programação, justificando o foco na busca pela relação entre o nível cognitivo dos alunos e seu desempenho na disciplina.

Em um primeiro momento, foi selecionado uma porção de dados simples do banco de dados, correspondendo aos resultados finais da disciplina de lógica de programação e dos testes operatórios. Após alguns testes, percebeu-se que, com a porção de dados completos dos testes operatórios, isto é, as respostas de cada pergunta de cada teste operatório, mais os resultados finais da disciplina de lógica, seria possível obter melhores resultados.

4.2 Preparação dos Dados

Possivelmente a etapa mais longa de todo o processo KDD, é aqui onde os dados são limpos, onde se ajusta as nomenclaturas dos dados, a estrutura, a transformação dos dados, normalização e balanceamento.

- Limpeza: procura-se por entradas inválidas, dados errados, nulos ou faltantes. Caso sejam encontrados dados nessas condições, eles devem ser ajustados ou removidos.
- Nomenclatura: etapa onde todas os nomes de tabelas, colunas e atributos devem ser ajustados para que haja um padrão bem definido.
- Transformação: etapa onde os dados vindos do banco precisam ser transformados para formato numérico a fim de contemplar os modelos de *machine learning*. Assim, os dados provindos dos testes operatórios, que anteriormente eram representados pelas seguintes *strings*: FORMAL, FORMAL A, FORMAL B, PRÉ-FORMAL, CONCRETO, CONCRETO SUPERIOR; passaram a ter os seguintes formatos: 3, 3, 3, 2, 1, 1 respectivamente. Já os dados provindos da disciplina de lógica de programação tiveram a seguinte transformação: Aprovado(A, B,C), Reprovado(D), Evadido(E, Trancou, Evadido) receberam os valores 3, 2, 1 respectivamente.
- Normalização: etapa onde os valores precisam ser normalizados, isto é, devido a possíveis diferenças de natureza ou escala em que foram medidas, eles precisam ser transformados de forma que nenhum valor de atributo predomine sobre outro. Foi utilizada a função de normalização standardScaler da biblioteca de *machine learning* scikit-learn para realizar essa etapa.
- Balanceamento: etapa onde o conjunto de dados é dividido em porções proporcionais de classes, para que na fase de treinamento o algoritmo não se torne tendencioso.

4.3 Análise Preliminar

Foi feita uma análise com os dados ‘crus’, sem a aplicação dos modelos de predição, com o intuito de entender melhor os dados. Assim quando os modelos fossem aplicados se poderia ter uma noção se os algoritmos estavam levando à direção certa. Foi identificado, analisando as imagens 1 e 2 a seguir, que diversos alunos, apesar de terem nível cognitivo Formal, acabava por vir a reprovar ou evadir. Quando a lógica é de que alunos com melhores notas irão aprovar, os dados acabam por mostrar que nem sempre é o que acontece.

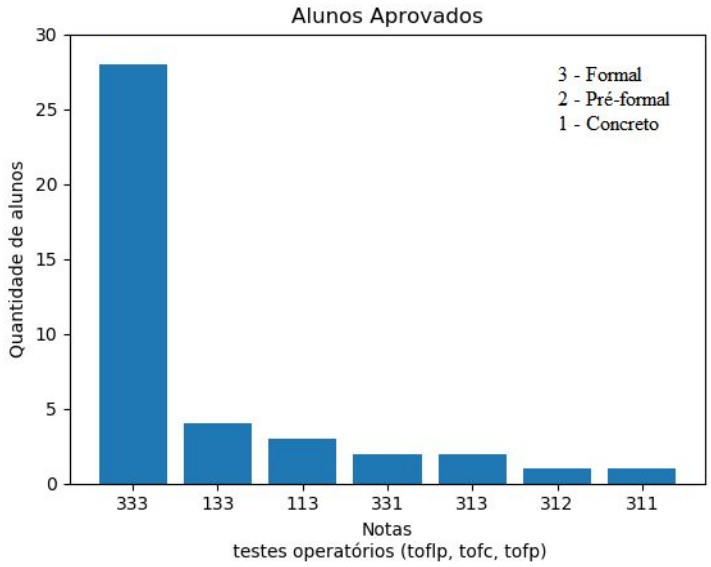

Figura 1 - Alunos aprovados

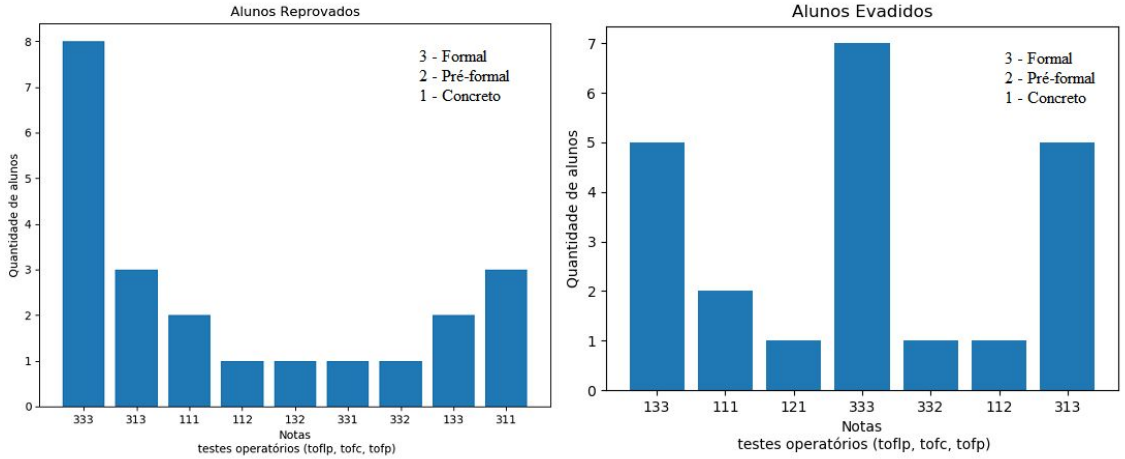

Figura 2 - Alunos Reprovados e Evadidos

4.4 Criação de modelos

Para a criação de modelos, utilizou-se das técnicas de Clustering com o algoritmo de *K-means* e Classificação com os algoritmos *Decision Tree Classifier* e *SVM*. Inicialmente a intenção era apenas encontrar padrões nos dados através do algoritmo de clustering. Porém como os dados históricos já possuem suas classes (se o aluno aprovou ou não) definidas, optou-se também por utilizar o algoritmo de classificação, o qual terá grande parcela de ajuda no momento de classificar novos alunos. Aliou-se então os dois algoritmos, e no fim pode ser efetuadas comparações e melhores análises.

Todas as etapas do KDD, anteriormente descritas, foram aplicadas tanto para a criação dos modelos com dados simples quanto para os modelos com dados completos.

5. Descrição do Sistema

Para fins de implementação, está sendo desenvolvido um software utilizando a linguagem de programação Python, associada à biblioteca de aprendizado de máquina *Scikit-learn*. Dentro desta biblioteca estão disponíveis técnicas de aprendizado de máquina dentre as quais, está sendo usada a de *Clusterização*, com o algoritmo *K-means*, e a técnica de Classificação com algoritmos de Árvores de decisão *e SVM (Support Vector Machine)* para comparar o desempenho entre técnicas.

Na figura 3 estão representadas as etapas do sistema. Inicialmente os resultados dos testes operatórios e as notas finais da disciplina de Lógica de Programação são armazenados no banco de dados. Com os dados já adquiridos, são executadas as etapas do KDD, dentre elas estão a limpeza, transformação e mineração dos dados. Para a mineração é aplicado o algoritmo de aprendizado de máquina *k-means*. Após o algoritmo ter finalizado seu trabalho, a análise dos resultados poderá ser efetuada. O algoritmo de *K-means* funciona da seguinte maneira: Com os dados já definidos e devidamente tratados deve-se escolher a quantidade de *clusters* a serem formados, no caso deste projeto, a quantidade será 3 (K = 3). Após esta definição, é preciso escolher, neste caso, 3 pontos, que servirão de base inicial para iniciar o processo de busca por padrões. Esses pontos são dados colhidos da base de dados e sua escolha pode ser feita de forma aleatória. Em seguida, com os centróides (pontos iniciais) definidos, o algoritmo pode começar as iterações e encontrar resultados. A primeira iteração calcula a distância média entre os pontos atrelados ao seu centro e então ajusta a posição onde o centróide deve ficar. Esse reajuste pode ocasionar a transferência entre pontos de um grupo para outro, por conta da relação entre a distância média até os centróides. O cálculo da média da distância dos pontos até os centróides ocorre em *loop*, até que nenhum ponto mude de agrupamento e até que cada *centróide* não tenha uma mudança significativa dentro de uma tolerância pré estipulada. Quando não houver mais deslocamento dos centros de cada grupo formado, e uma análise inicial for positiva sobre os resultados obtidos na busca por padrões, pode-se fazer o teste na prática, para ver como o algoritmo se sai em predizer novos dados inseridos.

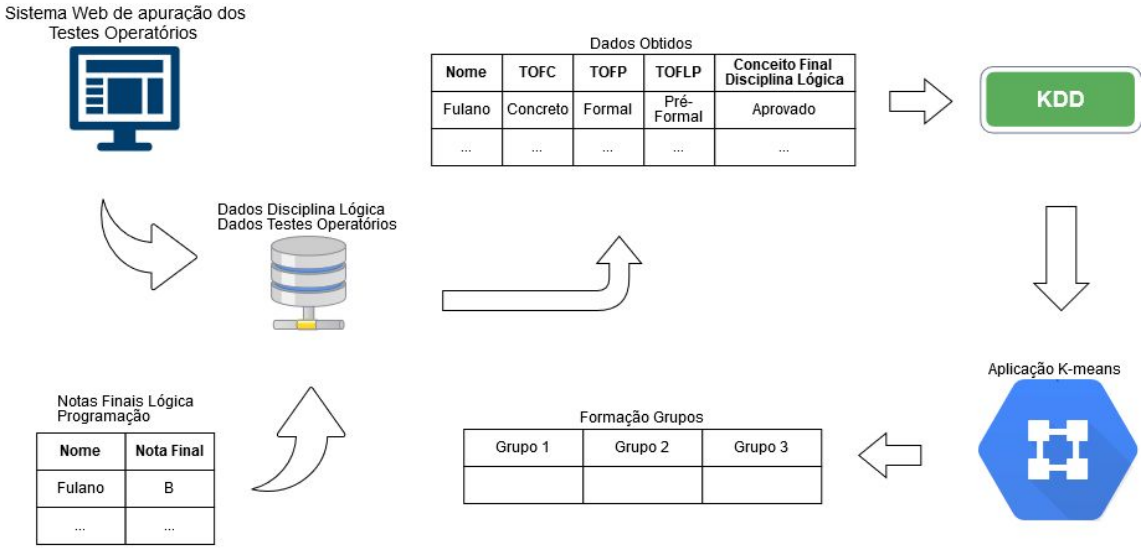

Figura 3 - estrutura do projeto

A seguir estão exemplificados partes do código de clustering e classificação:

- Clustering:

```
if self.tecnica == 'clustering':
  print('Modelo Clustering ML (Dados Completos)')
  y = self.data_frame[target]
  X = self.data_frame[features_completo]
  X_train, X_test, y_train, y_test =
train_test_split(X,y, test_size=0.3, random_state=324)
  model_kmeans = KMeans(n_clusters=3, random_state=324)
  model_kmeans.fit(X_train[['TOFLP_R2','TOFLP_R3',
'TOFLP_R4', 'TOFLP_R5','TOFLP_R6','TOFLP_P2','TOFLP_P3',
'TOFLP_P4','TOFLP_P5','TOFLP_P6','TOFLP_P7', 'TOFP_2',
'TOFP_3', 'TOFP_4', 'TOFP_5', 'TOFP_6', 'TOFP_7',
'TOFP_8', 'TOFP_9','TOFP_10', 'TOFC_2', 'TOFC_3A',
'TOFC_3B', 'TOFC_4A', 'TOFC_4B', 'TOFC_5','TOFC_6']])
  tabela = 'ML_RESULT_COMP_MODEL_CLUST'
  gravar = input('deseja gravar dados no banco?')
if gravar == 'sim':
  self.grava_tabela_resultado_modelo(model_kmeans,
X_test, y_test, tabela)
else:
  previsao =
model_kmeans.predict(X_test[['TOFLP_R2','TOFLP_R3',
'TOFLP_R4','TOFLP_R5','TOFLP_R6','TOFLP_P2','TOFLP_P3',
'TOFLP_P4','TOFLP_P5','TOFLP_P6','TOFLP_P7','TOFP_2',
```

```
    'TOFP_3','TOFP_4','TOFP_5','TOFP_6','TOFP_7','TOFP_8',
    'TOFP_9','TOFP_10','TOFC_2','TOFC_3A','TOFC_3B',
    'TOFC_4A', 'TOFC_4B','TOFC_5','TOFC_6']])
14    print('porcentagem de acerto %.2f' %
    (accuracy_score(y_test, previsao) * 100), '%')
15    print('operacao finalizada')
```

- Classificação:

```
1 if self.tecnica == 'classificacao':
2    print('Modelo Classificação ML (Dados Completos)')
3    y = self.data_frame[target]
4    X = self.data_frame[features_completo]
5    X_train, X_test, y_train, y_test =
    train_test_split(X,y, test_size=0.3, random_state=324)
6    model = svm.SVC(kernel='linear', C=1.0)
7    model.fit(X_train[['TOFLP_R2','TOFLP_R3',
    'TOFLP_R4', 'TOFLP_R5','TOFLP_R6','TOFLP_P2','TOFLP_P3',
    'TOFLP_P4','TOFLP_P5','TOFLP_P6','TOFLP_P7', 'TOFP_2',
    'TOFP_3', 'TOFP_4', 'TOFP_5', 'TOFP_6', 'TOFP_7',
    'TOFP_8', 'TOFP_9','TOFP_10', 'TOFC_2', 'TOFC_3A',
    'TOFC_3B', 'TOFC_4A', 'TOFC_4B', 'TOFC_5','TOFC_6']])
8    tabela = 'ML_RESULT_COMP_MODEL_CLASS'
9    gravar = input('deseja gravar dados no banco?')
10 if gravar == 'sim':
11    self.grava_tabela_resultado_modelo(model, X_test,
    y_test, tabela)
12 else:
13    previsao =
    model.predict(X_test[['TOFLP_R2','TOFLP_R3',
    'TOFLP_R4','TOFLP_R5','TOFLP_R6','TOFLP_P2','TOFLP_P3',
    'TOFLP_P4','TOFLP_P5','TOFLP_P6','TOFLP_P7','TOFP_2',
    'TOFP_3','TOFP_4','TOFP_5','TOFP_6','TOFP_7','TOFP_8',
    'TOFP_9','TOFP_10','TOFC_2','TOFC_3A','TOFC_3B',
    'TOFC_4A', 'TOFC_4B','TOFC_5','TOFC_6']])
14    print('porcentagem de acerto %.2f' %
    (accuracy_score(y_test, previsao) * 100), '%')
15    print('operacao finalizada')
```

Nos dois modelos podemos observar que primeiramente captamos os dados que irão servir como atributos e classes (linhas 3 e 4). Após é utilizada uma função em que os dados são balanceados e divididos em porções para treino e teste (train_test_split,

linha 5). Os algoritmos são iniciados (linha 6), após recebem os dados dos atributos selecionados (linha 7) e iniciam seu aprendizado. Finalizando o processo, o programa pergunta se quer que os resultados sejam salvos ou não (linha 10). A diferença mais notável dos modelos se encontra no momento em que é aplicada a função que executa a técnica (linha 6), seja de clustering ou de classificação. Ambos algoritmos utilizam das mesmas porções de dados, com a ressalva de que o algoritmo de clustering não utiliza as classes para o treinamento, o que dá a característica de aprendizagem não supervisionada. Já o modelo de classificação utiliza tanto as classes quanto os atributos, onde as classes são a porção de dados que dão a característica de aprendizado supervisionado ao modelo.

6. Resultados

Para validação comparou-se as taxas de assertividade entre os modelos, como ilustrado na tabela 2 a seguir:

Tabela 2 - Resultados modelos ML

	Linear svm.svc (% acerto)	**DecisionTreeClassifier (% acerto)**	**K-means (3 clusters)**
Modelo Classificação dados simples	53,85%	53,85%	
Modelo Classificação dados completos	38,46%	34,62%	
Modelo Clustering dados simples			61,54%
Modelos Clustering dados completos			23,08%

Pode-se notar que a aplicação do modelo com dados simples acabou por obter resultados melhores no que se refere a porcentagem de assertividade. Porém classificar o modelo como bom ou ruim apenas pela taxa de assertividade pode ser um erro, visto que dependendo da forma em que o modelo for montado (etapas KDD, parâmetros de algoritmos), essa porcentagem pode ter variações, ocasionando que hora o modelo

simples pode ter melhores resultados e hora o modelo completo pode ter melhores resultados.

7. Conclusões

Para obtenção de resultados dentro da proposta do projeto, foram criados diversos modelos de *Machine Learning*, diferenciando-os com pequenas mudanças de parâmetros e técnicas, para poder analisar o comportamento dos dados e os resultados finais de cada modelo, tentando assim encontrar àquele que fosse mais assertivo possível.

Inicialmente a proposta era utilizar os dados referentes aos resultados finais dos testes operatórios aplicados e da disciplina de lógica de programação, porém os modelos gerados, tanto de classificação quanto de clusterização não obtiveram uma taxa de assertividade relevante o suficiente para extrair conclusões confiáveis.

Após as tentativas iniciais, surgiu a ideia de utilizar não somente os resultados finais dos testes operatórios, mas sim o resultado de cada pergunta feita pelos testes. Assim o modelo pôde ficar mais completo e ter mais artifícios para compor seu aprendizado. Por fim os resultados dos modelos nesse formato não foram tão diferentes dos feitos anteriormente, com dados simples.

As diferenças entre os modelos que se utilizavam de dados simples com os que se utilizavam de dados completos não foram tão significativas, mostrando que o algoritmo não conseguiu encontrar um padrão relevante o suficiente para ser capaz de fazer previsões sobre novos dados. Além disso, com uma análise manual sobre os dados, também não foi possível encontrar padrões que mostrassem o porque de alunos serem aprovados, reprovados ou evadidos. Percebeu-se que notas suficientes para aprovação, não significam necessariamente a aprovação.

Para implementação dos modelos dividiu-se os dados em dois conjuntos, um deles serve para treinar o modelo de predição e o outro serve para testar o modelo. Isso é feito para que o modelo possa testar seu aprendizado em dados que nunca viu antes, e não ser tendencioso. Ao total foram 85 alunos, 70% dos dados foram utilizados para treino e 30% para teste. Apesar de as porcentagens de assertividade no modelo com mais dados terem sido menores, a comparação entre modelos com dados simples e completos pôde ser feita, chegando a conclusão de que independente da forma com que os modelos foram criados, nesse contexto, os algoritmos não conseguiram encontrar padrões relevantes capazes de estabelecer uma relação entre os testes operatórios e as notas finais da disciplina. Por conseguinte, também não foi possível alcançar uma porcentagem relevante de acerto nas previsões de novos alunos. Há diversos casos em que alunos com notas satisfatórias para aprovação acabam por reprovar ou evadir. Há também casos em que o aluno não tem um bom rendimento e acaba por aprovar. A conclusão que se chega é que os algoritmos não foram capazes de encontrar um padrão bem definido se baseando apenas nas notas obtidas dos testes operatórios, não conseguindo identificar uma relação entre o nível cognitivo dos alunos com a aprovação

da disciplina, mostrando que a análise dos resultados obtidos com os algoritmos de aprendizado condiz com a feita na análise preliminar. Por fim, existem diversos outros fatores, alheios aos resultados acadêmicos e ao nível cognitivo, como renda, local onde o aluno reside, situação familiar, situação psicológica, entre outros, que afetam o desenvolvimento dos alunos, não sendo suficiente, portanto, apenas o uso dos dados utilizados por este trabalho. Cabendo assim, em trabalhos futuros, a adição de atributos aos modelos.

Como possíveis trabalhos futuros, pode-se apontar:

- Busca por novos atributos, a fim de construir modelos mais confiáveis e assertivos em suas previsões.
- Criação de *dashboards* na interface web, com gráficos e dados relevantes para facilitar a interpretação dos dados.

Referências

BORGES, K.S. (2018) **Um Estudo Sobre Pensamento Formal No Contexto dos Makerspaces Educacionais.** Tese de Doutorado em Informática na Educação, PGIE, UFRGS. Porto Alegre. Disponível em https://www.lume.ufrgs.br/handle/10183/187572. Acesso em março de 2019.

BRITO, Marcelo. (2012) **Aspectos teóricos da mineração de dados e aplicação das regras de classificação para apoiar o comércio.** Devmedia. Disponível em: <http://www.devmedia.com.br/aspectos-teoricos-da.-mineracao-de-dados-e-aplicacao-das-regras-de-classificacao-para-apoiar-o-comercio/25429>

CAMARGO, D. A. F. de. (1990) **Desempenho operatório e desempenho escolar.** Cadernos de Pesquisa, São Paulo, v. 74, p.47-56, ago. 1990. Disponível em: <http://publicacoes.fcc.org.br/ojs/index.php/cp/article/view/1082/1087>. Acesso em maio de 2018.

CAMPELLO, A. V. C., LINS, L. N (2008). M**etodologia de análise e tratamento de evasão e retenção em cursos de graduação de instituições federais de ensino superior**. XXVIII Encontro Nacional de Engenharia de Produção. Disponível em: <http://secaio.com/personal/TC/enegep2008_TN_STO_078_545_11614.pdf>. Acesso em 4 Set 2018.

CANTELLI, V. C. B.; BORGES, R. R.; ASSIS, O. Z. M. (2005). **Avaliação do Desenvolvimento Intelectual de Alunos da Educação de Jovens e Adultos Brasileiros Numa Perspectiva Piagetiana**.VIII Congresso Galaico Português de PsicoPedagogia. Anais...Centro de Investigação em Educação (CIEd). Instituto Educação e Psicologia. Universidade Minho, 2005. Disponível em: <http://www.educacion.udc.es/grupos/gipdae/documentos/congreso/viiicongreso/pdfs/96.pdf>. Acesso em maio de 2018.

CERVEIRA, A. J. K., REATEGUI,i E. B., LIMA, J. V. (2012), **Mineração de dados educacionais para a construção de alertas em ambientes virtuais de aprendizagem como apoio à prática docente**. Disponível em: <http://hdl.handle.net/10183/22888>. Acesso em 4 Set 2018.

HOED, R. M. (2016), **Análise da evasão em cursos superiores: o caso da evasão em cursos superiores da área de Computação**. Disponível em: <http://repositorio.unb.br/bitstream/10482/22575/1/2016_RaphaelMagalh%C3%A3esHoed.pdf>. Acesso em 22 Agosto de 2018.

LEMOS, M. F. de; QUEIROZ, S. S. de (2015). **Desempenho operatório de adultos e idosos nas provas da Escala de Desenvolvimento do Pensamento Lógico (EDPL)**. Revista Vozes dos Vales: Publicações Acadêmicas, n. 7, maio 2015.

LONGEOT, F. (1979) **L'Échelle de développement de la Pensée Logique**, Issy-les-Moulineaux, EAP, 1979.

NATEK S, ZWILLING M. (2014), **Student data mining solution-knowledge management system related to higher education institutions**. Expert Systems with Applications. Disponível em:<https://www.sciencedirect.com/science/article/pii/S0957417414002462>. Acesso em 8 Out 2018.

PIAGET, J. (1973) **A Epistemologia Genética**. 2. ed. Rio de Janeiro: Vozes, 1973.

SOUZA, M. T. C. C. de; MACEDO, L. (1986) de. **Operações Formais em Universitários de Diferentes Áreas Profissionais: Uma Análise Comparativa**. Psicologia: Teoria e Pesquisa,v. 2, n. 2, p. 165–178, 1986

SZYMANSKI, M. L. S. (2011) **A difícil aprendizagem de tarefas que exigem um raciocínio complexo**. Boletim Técnico do Senac, v. 37, n. 1, p. 24–33, 2011. Disponível em: http://bts.senac.br/index.php/bts/article/view/199. Acesso em 19 abr. 2011.