Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Algoritmos de Mineração de Dados para Análise de Evasão na Graduação da Universidade de Brasília

Wladimir Ganzelevitch Mesquita Gramacho

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Orientador
Prof. Dr. Guilherme Novaes Ramos

Brasília
2019

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Algoritmos de Mineração de Dados para Análise de Evasão na Graduação da Universidade de Brasília

Wladimir Ganzelevitch Mesquita Gramacho

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Prof. Dr. Guilherme Novaes Ramos (Orientador)
CIC/UnB

Prof. Dr. Sergio Antônio Andrade de Freitas
FGA/UnB, DEG/UnB

Prof. Dr. Donald Matthew Pianto
EST/UnB

Prof. Dr. Edison Ishikawa
Coordenador do Bacharelado em Ciência da Computação

Brasília, 4 de dezembro de 2019

Dedicatória

Dedico este trabalho à minha família, que desde sempre me ajudou e me apoiou para que eu pudesse chegar até aqui. Meus pais, amor incondicional, guerreiros incansáveis, mentores da minha vida, essenciais para o meu ser. Minhas irmãs, que me ajudaram a crescer como pessoa desde a infância até os dias presentes. E os outros familiares, que com carinho e amor fizeram a jornada ser incrível. Dedico também a minha companheira, Beatriz, com a qual tenho o prazer de caminhar junto e que, com paciência e empatia, me incentivou neste trabalho.

Amo vocês.

Agradecimentos

Agradeço, primeiramente, à orientação atenciosa e comprometida do Professor Guilherme. Agradeço ao Professor André Drummond pelos conselhos e conversas descontraídas. Agradeço também a todos os professores que fizeram parte da minha trajetória até hoje.

Aproveito também para agradecer a todas as pessoas que divulgaram conteúdo de qualidade publicamente na Internet. Para cada artigo acadêmico, tutorial ou vídeo-aula, existe uma mente jovem, ávida por conhecimento.

Por fim, agradeço a todos que contribuíram, direta ou indiretamente, para a realização deste trabalho.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES), por meio do Acesso ao Portal de Periódicos.

Resumo

A evasão de estudantes nas universidades tem sido motivo de muita discussão e análise. Nas universidades públicas, as perdas de estudantes que iniciam mas não terminam seus cursos são desperdícios de recursos públicos: sociais, acadêmicos e econômicos. Identificar estudantes em risco de evasão pode auxiliar na prevenção desse problema. Este trabalho tem o intuito de avaliar a aplicação de métodos de Mineração de Dados e Aprendizagem de Máquina para predição de risco de evasão na Universidade de Brasília. Seguindo a metodologia *Knowledge Discovery Process*, foram obtidos classificadores com sensibilidade de 82% e acurácia de 81%. Além disso, foram geradas regras de associação de alta confiança (chegando até 100% de confiança) que podem ser usadas para a tomada de decisão por gestores dos cursos de graduação.

Palavras-chave: mineração de dados, aprendizagem de máquina, evasão estudantil

Abstract

Student dropout at universities has been the subject of much discussion and analysis. In public universities, the loss of students who start but do not finish their courses is a waste of public resources: social, academic and economic. Identifying students at risk of dropout can assist in preventing this problem. This work aims to evaluate the application of Data Mining and Machine Learning methods to predict dropout risk at the University of Brasília. Following the Knowledge Discovery Process methodology, classifiers with a sensitivity of 82% and an accuracy of 81% were obtained. In addition, high confidence association rules (up to 100% in confidence) were generated that can be used for decision making by undergraduate managers.

Keywords: data mining, machine learning, student evasion, classification

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

CART Classification and Regression Trees.

ENEM Exame Nacional do Ensino Médio.

IES Instituição de Ensino Superior.

Inep Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira.

KDP *Knowledge Discovery Process*.

MD Mineração de Dados.

UnB Universidade de Brasília.

Capítulo 1

Introdução

O Ensino Superior, não só no Brasil como no mundo todo, possui uma importância muito grande pois pode ser considerado como um dos fatores fundamentais para o desenvolvimento econômico, social e cultural nas nações [1, 2, 3]. Dessa forma, é mister que a sociedade se preocupe e atue na qualidade e eficiência do ensino. Uma Instituição de Ensino Superior eficiente é uma instituição com baixas taxas de evasão, conseguindo assim formar profissionais capacitados para o mercado de trabalho e academia. Isso posto, este trabalho direciona-se à diminuição da evasão no Ensino Superior por meio da identificação de estudantes em risco de evasão e geração de conhecimento através de técnicas de Mineração de Dados.

1.1 Contextualização

A evasão de estudantes nas universidades tem sido motivo de muita discussão e análise entre educadores, gestores e outros agentes da Educação [4, 5]. Nas universidades públicas, as perdas de estudantes que iniciam mas não terminam seus cursos são desperdícios de recursos públicos: sociais, acadêmicos e econômicos [6]. Esses recursos são desperdiçados porque o investimento feito na formação dos ingressantes não é convertido em profissionais formados se tais estudantes evadem ao longo do curso.

A última Pesquisa de Retenção e Evasão da Universidade de Brasília [7], feita em 2016, analisou os ingressantes entre 2002 e 2008. Avaliando todos os cursos, a universidade obteve uma taxa de evasão de 23.9%. Mas ao avaliar cursos de Ciências Exatas, essas taxas aumentaram consideravelmente. Por exemplo, o curso de Ciência da Computação (Diurno) obteve uma taxa de evasão de 53.3%. No curso de Matemática (Diurno), a taxa foi de 57.3%.

Conseguir prever se um estudante evadirá ou não é importante para poder auxiliar gestores no apoio à diminuição da evasão na Universidade. Descobrir manualmente os

elementos que mais influenciam nessa evasão é uma tarefa muito difícil e onerosa. Por outro lado, aplicar técnicas de mineração de dados para a classificação do sucesso dos alunos pode ser uma solução. Este trabalho tem o intuito de mostrar a construção e os resultados da aplicação de métodos de aprendizagem de máquina à evasão de estudantes nas universidades. Parte-se da hipótese de que o resultado de um aluno num curso da universidade (formatura ou evasão) pode ser previsto a partir do histórico dele nas matérias cursadas.

1.2 Objetivos

Este trabalho ambiciona fazer uma análise da trajetória acadêmica do aluno da UnB. A partir de dados dos históricos de estudantes da Universidade de Brasília, o objetivo principal é a identificação dos estudantes em risco de evasão. Ademais, outro objetivo é a geração de conhecimento que possa ser aplicado por gestores de Educação em medidas para mitigar a evasão universitária. Pretende-se, ainda, criar uma ferramenta pública para construção dos classificadores de alunos.

Seguindo a metodologia *Knowledge Discovery Process* (KDP), os dados serão extraídos, processados e transformados para que se possa aplicar métodos de aprendizagem de máquina. Considerando os trabalhos correlatos e seus resultados, serão utilizados os algoritmos de Árvores de Decisão, Florestas Aleatórias, Redes Neurais e Regressão Logística como métodos de classificação. Além disso, o algoritmo Apriori é utilizado para a construção de regras de associação, a fim de obter conhecimento a partir dos dados. Juntamente com o esclarecimento de cada etapa do KDP, são apresentados os processos e os resultados desenvolvidos ao longo deste trabalho. Por fim, são expostos os resultados obtidos e discutidos os próximos passos desta pesquisa.

O Capítulo 2 realiza uma revisão da literatura tanto dos trabalhos correlatos a este como de conceitos utilizados. O Capítulo 3 apresenta a metodologia seguida no decorrer do trabalho. O Capítulo 4 apresenta os resultados obtidos com os experimentos, além de mostrar os conhecimentos adquiridos e consolidados. Por fim, o Capítulo 5 conclui o trabalho, discutindo caminhos para avanços na pesquisa.

Capítulo 2

Revisão de Literatura

Este capítulo revisa a literatura acerca de trabalhos correlatos a este, que classificam estudantes como em risco de evasão em universidades. Também são discutidos estudos feitos em relação à evasão no ensino superior e suas definições. Para o embasamento teórico, é feita uma introdução à mineração de dados e aprendizagem de máquina, além da explicação sucinta do funcionamento dos algoritmos de classificação utilizados.

2.1 Trabalhos Correlatos

Ao observar que 30% dos estudantes de instituições de bacharelado dos Estados Unidos evadem após o primeiro ano de curso, Aulck *et al.* mostraram que a evasão de estudantes pode ser classificada por algoritmos de aprendizagem de máquina [8]. Com uma base de dados de mais de 66 mil estudantes, analisando os históricos de todos os estudantes da Universidade de Washington até 2017 e suas características demográficas e de desempenho escolar, os autores utilizaram Regressão Logística, Florestas Aleatórias, Máquina de Vetores de Suporte (SVM), Árvores com gradiente de reforço (*gradient boosting trees*, em inglês) e *k-Nearest Neighbors* para classificar estudantes entre evadidos e formados. A definição de evasão que foi utilizada é a de que o estudante haverá evadido caso não conclua um curso de graduação após 6 anos da data de ingresso na universidade, e o classificador com melhor acurácia foi o de Regressão Logística, com 83.2% de acurácia.

Em [9], Delen obteve dados de mais de 25 mil estudantes matriculados em uma universidade pública nos Estados Unidos que continham informações acadêmicas, financeiras e demográficas dos estudantes. Deles, 19 mil voltaram à universidade após o primeiro ano de curso, observando-se uma evasão de aproximadamente 21% após 12 meses de curso. Dessa maneira, combinou os históricos de Ensino Médio e Universidade dos alunos no primeiro ano para treinar classificadores e obteve acurácias de 81%, 78% e 74% para Redes Neurais, Árvores de Decisão e Regressão Logística, respectivamente. Delen também

analisou a estrutura da Árvore de Decisão e descobriu que os principais fatores para a classificação da evasão são relacionados ao desempenho acadêmico no passado e presente do aluno.

Dekker *et al.* [10] aplicaram mineração de dados ao problema da evasão especificamente no curso de Engenharia Elétrica da Universidade de Tecnologia de Eindhoven, na Holanda. A taxa de evasão após um ano de curso era de aproximadamente 40%. Foi definido como evadido o estudante que não houver completado com sucesso as matérias do primeiro ano do curso após 3 anos da sua matrícula. Os autores analisaram tanto dados acadêmicos prévios à universidade quanto dados de desempenho durante a universidade. Foram treinados utilizados diversos algoritmos: *OneR* (algoritmo que utiliza somente uma regra de decisão), Classification and Regression Trees (CART), C4.5, um classificador Bayesiano, Regressão Logística, um algoritmo baseado em aprendizagem de regras e Florestas Aleatórias. Os melhores resultados foram obtidos pelos algoritmos que utilizam Árvores de Decisão (CART, C4.5 e Florestas Aleatórias) e o algoritmo de Regressão Logística, com acurácias de aproximadamente 80%.

No Brasil, Assis [11] estudou os fatores que levam à evasão no Ensino Superior em relação a cursos, áreas de estudo e Instituições de Ensino Superior, além da possibilidade de se classificar os estudantes a partir de algoritmos de aprendizagem de máquina. Com dados do Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (Inep) e do Exame Nacional do Ensino Médio (ENEM), foram testados cinco algoritmos: *Naive Bayes*, Redes Neurais, Regressão Logística, e dois algoritmos de Árvores de Decisão (CART e C5.0). A nível de curso, o melhor resultado foi obtido pelo CART, que se sobressaiu quando comparado aos outros classificadores, ao atingir a sensibilidade de 83.84%.

Tendo em vista os resultados alcançados por estes trabalhos correlatos, serão utilizados neste trabalho os algoritmos de Árvores de Decisão, Florestas Aleatórias, Regressão Logística e Redes Neurais pois apresentam os melhores resultados e podem trazer conhecimentos importantes para o usuário final. Para cada algoritmo, será avaliada a sensibilidade e a acurácia, onde qualifica-se, respectivamente, a habilidade dos classificadores de encontrar o máximo de alunos que de fato evadem e de classificar corretamente os alunos em evadidos e formados. Ademais, visto que estudos mostram que mais estudantes evadem logo após o primeiro ano do que o resto do curso [12, 13], este trabalho lida somente com os dados dos históricos acadêmicos dos estudantes nos seus dois primeiros semestres letivos, desconsiderando informações demográficas e sócio-econômicas dos estudantes e seus contextos familiares. Além dos algoritmos de classificação, será utilizado também um algoritmo de regras de associação (Apriori) com o objetivo de trazer mais conhecimento para os gestores e coordenadores de graduação dos cursos da universidade.

2.2 Evasão no ensino superior

Existem diversos trabalhos que se propõem a definir evasão [14, 15, 16], sendo o desenvolvido por Tinto [17] a principal referência nos trabalhos correlatos a este [8, 11, 18] pois ele coleta um *survey* dos trabalhos acerca do tema analisando diversos aspectos do estudante. Tinto defende que o processo de evasão de um aluno é determinado por questões individuais, familiares, sociais, acadêmicas e estruturais (da instituição). Segundo ele, é necessário distinguir o motivo do não término de um curso, uma vez que um aluno pode ser expulso do curso ou sair do curso voluntariamente. Este trabalho não faz distinção do motivo da evasão, uma vez que visa apenas encontrar os alunos que não concluíram o curso. Além disso, vale observar que pode-se analisar evasão em diferentes níveis: em relação à Instituição de Ensino Superior (IES), em relação à Área de Conhecimento e em relação ao curso, entre outros [11]. Isto posto, este trabalho só leva em consideração a evasão a nível de curso e avalia a finalização do curso com sucesso ou não, considerando como evadido o estudante que não concluir o curso como formado.

Segundo os trabalhos correlatos, é possível aplicar técnicas de mineração de dados ao problema da evasão nas universidades. Uma vez feita a definição de evasão, faz-se necessário explicar os conceitos de mineração de dados.

2.3 Mineração de Dados

Em 1992, Frawley *et al.* conceituaram *Knowledge Discovery* como a extração não trivial de informações implícitas, previamente desconhecidas e potencialmente úteis a partir de dados [19]. A partir desse conceito, surgiu a Mineração de Dados (MD). Mineração de Dados é definido como o processo de descobrir padrões (automaticamente ou semi automaticamente) a partir de dados [20]. Com os padrões encontrados, são feitas previsões dos dados de forma a obter conhecimento. Tais previsões podem ser feitas por ferramentas estatísticas como SAS Enterprise Miner ou SPSS [21]. Por outro lado, o processo de mineração de dados pode usufruir de algoritmos de aprendizagem de máquina, de forma a obter esses padrões de forma automatizada a partir de grandes bases de dados [22].

2.4 Aprendizagem de Máquina

Aprendizagem de Máquina (*Machine Learning*, em inglês) é a área de estudo que pesquisa como computadores conseguem aprender (ou melhorar seu desempenho) baseados em dados, fazendo com que algoritmos reconheçam automaticamente padrões complexos a partir de dados e tomem decisões em relação a esses dados [23]. Além dos dados, é

preciso levar em consideração o tipo de problema que está posto e os objetivos a serem alcançados, levando em conta as três categorias de Aprendizagem de Máquina [24, 25]:

- aprendizagem supervisionada: quando cada instância observada possui classes conhecidas, ou seja, as saídas corretas correspondentes.
- aprendizagem não supervisionada: não há classes conhecidas para as instâncias, de forma que os algoritmos devem identificar as classes a partir de padrões.
- aprendizagem por reforço: o modelo aprende com base em sinais de reforço fornecidos pelo ambiente onde está inserido. À medida que o modelo executa ações, ele recebe sinais que o recompensam ou não de forma a treiná-lo a exercer melhor sua função.

O problema proposto neste trabalho se encaixa em aprendizagem supervisionada, visto que sabe-se o resultado da classificação. Essa categoria trabalha com algoritmos que encontram padrões a partir de instâncias fornecidas externamente para produzir hipóteses gerais, que então fazem previsões sobre instâncias futuras [25]. Dessa forma, esta pesquisa objetiva a utilização de aprendizagem de máquina supervisionada para identificar quais alunos têm risco de evasão a partir de seus desempenhos acadêmicos. Em outras palavras, pretende-se classificar os estudantes em *formados* e *evadidos*.

2.5 Classificação

Classificação de dados é o processo de prever a classe de uma série de dados a partir de um conjunto de classes e consiste em duas etapas: uma etapa de treinamento (em que o classificador é construído) e uma etapa de teste (em que o modelo é usado para prever os rótulos das classes para os dados fornecidos) [23]. Na fase de treinamento, o algoritmo utiliza um conjunto de exemplos composto de uma série de tuplas de uma base de dados e suas respectivas classes. A j-ésima tupla $\mathbf{x}_j$ é representada por um vetor de n dimensões, onde

$$\mathbf{x}_j = (x_1, x_2, \ldots, x_n) \tag{2.1}$$

de forma que cada x_i representa o valor do i-ésimo atributo da j-ésima tupla. O conjunto de todas as tuplas $\mathbf{x}$ das bases de dados é chamado de X. Assume-se que cada tupla $\mathbf{x}$ pertence a uma classe predefinida, determinada pelo atributo de rótulo de classe. O conjunto das classes é chamado de Y. Quando $Y = \{0, 1\}$ tem-se que a classificação é binária, que é o caso deste trabalho, no qual se pretende classificar os alunos em evadidos e formados.

Segundo Witten [20], em problemas de classificação, a organização dos dados é apresentada como um conjunto de exemplos classificados a partir dos quais se espera aprender uma maneira de classificar exemplos não vistos. Um exemplo é uma instância a ser categorizada. Neste trabalho, uma instância corresponde a um aluno rotulado como evadido ou formado. Como apresentado na Seção 2.1, as técnicas utilizadas para a classificação dos alunos são Árvores de Decisão, Florestas Aleatórias, Regressão Logística e Redes Neurais.

2.6 Árvores de Decisão

Uma árvore de decisão é um preditor, $h : X \to Y$, que prevê a classe associada a uma instância x caminhando do nó raiz da árvore para uma folha [26]. Simplificando, uma árvore de decisão é um modelo hierárquico de decisões e suas consequências [27], de forma que se construa um fluxograma em estrutura de árvore, na qual cada nó interno (nó não-folha) indica um teste em um único atributo, cada ramificação representa um resultado do teste e cada nó terminal possui um rótulo de classe [28].

Uma árvore é construída dividindo o conjunto de instâncias original, inicialmente a raiz da árvore, em subconjuntos que constituem os conjuntos sucessores [26]. Esse processo é repetido em cada subconjunto de forma recursiva. A recursão termina quando todas as instâncias de um subconjunto tem a mesma classe ou quando a divisão não traz valor para a classificação. Esse processo *top-down* é um exemplo de algoritmo guloso, estratégia comum para a construção de árvores de decisão a partir de dados [29, 30].

Para determinar qual teste deve ser executado a cada nó, o algoritmo utiliza um critério para encontrar a melhor divisão [22, 30]. Um dos critérios é o ganho de informação, utilizado nos algoritmos de geração de árvores ID3, C4.5 e C5.0. Ganho de informação é baseado no conceito de entropia e conteúdo da informação, apresentado no trabalho de Claude Shannon sobre teoria da informação [31]. Seja um nó T na árvore, sua entropia $H(T)$ é definida como

$$H(T) = -\sum_{i=1}^{J} p_i \log_2 (p_i), \tag{2.2}$$

onde as p_i frações somam 1.0 e representam a porcentagem de cada classe das instâncias presentes no nó gerado da divisão de T e J é a quantidade de classes distintas do conjunto dos dados [20]. O ganho de informação da divisão de um nó T a partir do atributo a é

$$IG(T, a) = H(T) - H(T|a), \tag{2.3}$$

onde $H(T)$ é a entropia do nó T e $H(T|a)$ é a soma ponderada das entropias dos nós gerados da divisão do nó T pelo atributo a, definida como

$$H(T|a) = \sum_{j=1}^{v} \frac{|T_j|}{|T|} \times H(T_j), \tag{2.4}$$

onde v é a quantidade de partições geradas da divisão, $|T_j|$ é a quantidade de instâncias no j-ésimo nó após a divisão por a, $|T|$ é a quantidade de instâncias no nó T e $H(T_j)$ é a entropia do j-ésimo nó gerado da divisão. O atributo com o maior ganho de informação é escolhido para a divisão do nó. Logo, minimiza-se a informação necessária para classificar as instâncias nas partições subsequentes, minimizando também a quantidade de testes necessários para classificar uma instância e garantindo que uma árvore simples (não necessariamente a mais simples) seja construída [23].

Outra abordagem para o critério da divisão é a do índice Gini. Índice Gini foi desenvolvido por Corrado Gini e mede a desigualdade entre os valores de uma distribuição de frequências das classes de uma partição D [26]. Em outras palavras, é medida a impureza do conjunto de instâncias D tal que

$$Gini(D) = 1 - \sum_{i=1}^{m} (p_i)^2, \tag{2.5}$$

onde p_i é a probabilidade de que uma instância x em D tenha a classe C_i e m é a quantidade de classes. A probabilidade p_i é estimada por

$$p_i = \frac{|C_{i,D}|}{|D|}, \tag{2.6}$$

onde $C_{i,D}$ é a quantidade de instâncias da classe C_i na partição D [23]. O índice Gini da divisão do conjunto de instâncias D em D_1 e D_2 a partir do atributo A é dado por

$$Gini_A(D) = \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2). \tag{2.7}$$

A partir dessa fórmula, o atributo que minimizar o índice Gini da partição será escolhido como o novo nó na árvore de decisão [23].

Esses dois critérios serão utilizados nos experimentos usando a implementação da biblioteca Scikit Learn do algoritmo CART. Classification and Regression Trees (CART) é um algoritmo que gera árvores binárias de forma gulosa, onde a árvore é construída de cima para baixo levando em conta a melhor divisão local de cada nó [23]. O CART pode gerar tanto classificações (quando a saída do algoritmo é uma classe) quanto regressões (quando a saída é um número real).

2.7 Florestas Aleatórias

Inicialmente proposto por Tin Kam Ho [32] e depois aperfeiçoado por Leo Breiman [33], o algoritmo de Florestas Aleatórias é um método *ensemble* de aprendizagem. Aprendizagem por conjunto, traduzindo do inglês, é um paradigma de aprendizagem de máquina em que vários classificadores são treinados associadamente para resolver o mesmo problema [34]. Em contraste com métodos comuns de aprendizagem que tentam encontrar uma hipótese a partir dos dados, métodos de aprendizagem por conjunto se propõe a encontrar um conjunto de hipóteses e combiná-las para obter melhores resultados.

Uma floresta aleatória é um classificador que consiste em uma coleção de árvores de decisão, onde a k-ésima árvore é construída a partir de um conjunto de instâncias D associado a um vetor aleatório, $\mathbf{\Theta}_k$, que é uma amostra escolhida aleatoriamente das características da instância. Cada $\mathbf{\Theta}_k$ é independente e identicamente distribuído [26, 33]. A classificação de uma instância será a previsão feita pela maioria das árvores individuais [26, 32, 33].

Em 1995, Ho mostrou que Florestas Aleatórias podem aumentar em acurácia à medida que crescem sem sofrer sobre-ajuste [32]. A demonstração da resistência de Florestas Aleatórias ao sobre-ajuste se baseia na teoria de discriminação estocástica de Kleinberg [35], a qual discorre sobre a síntese de classificadores precisos a partir de um grande número de modelos "fracos"imprecisos, utilizando de processos estocásticos discretos.

2.8 Regressão Logística

A regressão é um dos métodos estatísticos mais usados na prática [36]. Regressão é a tarefa que se propõe a encontrar padrões simples nos dados, um relacionamento funcional entre os conjuntos X e Y dos dados [26]. Dado um conjunto de dados de treinamento com n instâncias tal que $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, juntamente com seus respectivos valores-alvo $Y = \{y_1, y_2, \ldots, y_n\}$, o objetivo é prever o valor-alvo de um $\mathbf{x}$ desconhecido. Na abordagem mais simples, isso pode ser feito pela da construção direta de uma função

$$h(\mathbf{x}) = y \tag{2.8}$$

cujos valores para novas entradas $\mathbf{x}$ constituem as previsões para os valores correspondentes de y [37].

O primeiro tipo estudado foi a regressão linear por Legendre em [38]. Esta pode ser generalizada na fórmula

$$y = \beta_0 + \beta_1\chi_1 + \beta_2\chi_2 + \ldots + \beta_m\chi_m, \tag{2.9}$$

onde y é a variável independente, $\beta_0, \beta_1, \ldots, \beta_m$ são coeficientes da regressão e $\chi_1, \chi_2, \ldots, \chi_m$ são variáveis independentes do modelo (características da instância x) [37]. A equação pode ser simplificada para

$$y = \boldsymbol{\beta}^T \boldsymbol{\chi}, \tag{2.10}$$

onde $\boldsymbol{\beta} = (\beta_0, \beta_1, \beta_2, \ldots, \beta_m)$ e $\boldsymbol{\chi} = (1, \chi_1, \chi_2, \ldots, \chi_m)$.

Regressão Logística é uma abordagem para prever o resultado de uma variável dependente categorial baseada em uma ou mais variáveis observadas [39]. Esse algoritmo destina-se a prever a probabilidade de que um evento ocorra. Portanto, regressão logística tem a forma

$$P(y = 1) = \frac{1}{1 + e^{-(\boldsymbol{\beta}^T \boldsymbol{\chi})}} \tag{2.11}$$

onde $P(y = 1)$ é a probabilidade da variável dependente y ser 1 [40, 41]. A função logística p calcula as probabilidades que descrevem as possíveis classificações de uma entrada, e é uma função sigmoide, como apresentada na Figura 2.1. Em um problema com duas classes (evadido e formado), probabilidades maiores que 50% significam que o caso é atribuído à classe designada como "1" ou "0", caso contrário [9]. A partir do ponto onde a entrada é traçada na função logística, a classe é definida. Após treinar o modelo, ele pode ser usado para prever o sucesso (no caso, evasão) de novos indivíduos [42].

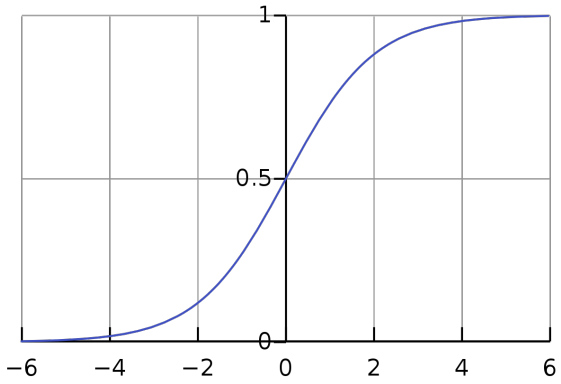

Figura 2.1: Função sigmoide (Ilustrado e disponibilizado publicamente por Geoff Richards).

2.9 Redes Neurais

Inspiradas na Biologia, redes neurais são redes com técnicas analíticas altamente sofisticadas e capazes de modelar funções não-lineares extremamente complexas [9]. Formalmente, uma rede neural é um processador distribuído e massivamente paralelo feito de

unidades de processamento simples (*perceptrons*) que são propensos a armazenar conhecimento e disponibilizá-lo para uso [43]. O *perceptron* (ou neurônio), corresponde, segundo Bishop [37], a um modelo de classes binárias no qual o vetor de entrada $\mathbf{x}$ é transformado usando uma transformação não-linear fixa para fornecer um vetor de características $\boldsymbol{\phi}(\mathbf{x})$, que é usado para construir um modelo linear generalizado da forma

$$y(\mathbf{x}) = f(\boldsymbol{\beta}^T \boldsymbol{\phi}(\mathbf{x})) \tag{2.12}$$

onde $f(a)$ é uma função de ativação não linear. A função de ativação pode ter várias formas [43] como função logística(Equação 2.13), função ReLU [44] (Equação 2.14) ou função tanh (Equação 2.15).

$$f(a) = \frac{1}{1 + e^{-a}} \tag{2.13}$$

$$f(a) = max(0, a) \tag{2.14}$$

$$f(a) = \frac{(e^a - e^{-a})}{(e^a + e^{-a})} \tag{2.15}$$

Cada neurônio recebe como entrada uma soma ponderada das saídas dos neurônios conectados às suas entradas, gerando uma saída a partir da função de ativação. Ao ajustar os pesos das arestas associados ao *perceptron*, ocorre o aprendizado [26].

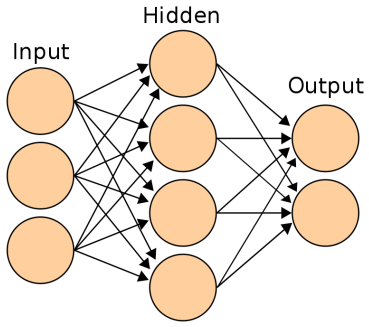

Figura 2.2: Rede neural com uma camada escondida (Ilustrado e disponibilizado publicamente por Colin M.L. Burnett).

Este trabalho utiliza um *perceptron* multicamadas (*multi-layer perceptron*, em inglês), um tipo de rede neural *feedforward*. Um MLP consiste em pelo menos três camadas de nós: uma camada de entrada, uma camada oculta e uma camada de saída. Na camada de entrada, são inseridos os valores das características da instância. Nas n camadas ocultas ocorrem os ajustes dos valores, na qual se utiliza a Equação 2.12 para calcular o valor

de cada neurônio. Por fim, o maior valor na camada de saída representa a classe da instância [26, 37].

2.10 Apriori

Além dos classificadores utilizados para a tarefa de prever estudantes em risco de evasão, investiga-se o uso de um algoritmo de regras de associação ao problema buscando a descoberta de conhecimento para os gestores dos cursos. A aquisição de conhecimento por regras de associação é um método de aprendizagem de máquina baseado em regras para descobrir relações interessantes entre variáveis de grandes massas de dados [45]. Agrawal *et al.* [46] introduziram o conceito de regras de associação por meio do algoritmo Apriori, com o intuito de identificar como as compras de produtos em um mercado se relacionavam entre si. Observando a frequência de cada característica nas instâncias, o algoritmo associa uma ou mais características (denominado LHS, do inglês *Left Hand Side*) com outra característica (RHS, *Right Hand Side*) produzindo uma instrução "Se LHS, então RHS será observada" [47].

Há dois conceitos importantes para a análise do método Apriori: suporte e confiança. Dado um conjunto X, suporte desse conjunto é a proporção de transações da base de dados que o contém. Confiança, por sua vez, é uma indicação de quantas vezes a regra foi considerada verdadeira, ou seja, a probabilidade de encontrar o RHS a partir do LHS [46, 48].

Exemplificando, supondo que 10% de todos os alunos obtiveram MI em Cálculo 1 no primeiro semestre e que 98% deles evadiram, tem-se que a seguinte regra de associação seria criada:

> Se obteve MI em Cálculo 1 no primeiro semestre, irá evadir com 98% de confiança e suporte de 10%.

Capítulo 3

Metodologia

Por não haver pesquisas científicas consolidadas que abordam as causas e a previsão da evasão em um curso da Universidade de Brasília, esta pesquisa tem cunho exploratório. A ideia foi aplicar Mineração de Dados para prever evasão especificamente para curso de Engenharia Mecatrônica, ou seja, um estudo de caso.

Mineração de Dados é o processo de descobrir padrões a partir de dados [20]. Ao longo do tempo, surgiram diversas metodologias para alcançar tal objetivo. Por exemplo, o CRISP-DM é uma metodologia que foi proposta pela indústria e segue 6 etapas: Entendimento de Negócio, Entendimento dos Dados, Preparação dos Dados, Modelagem, Avaliação e Implantação [49]. Por outro lado, a primeira metodologia para Mineração de Dados criada foi o *Knowledge Discovery Process* (KDP). Este trabalho utiliza o KDP e a explicação de cada etapa dessa metodologia será apresentada em conjunto com os experimentos feitos.

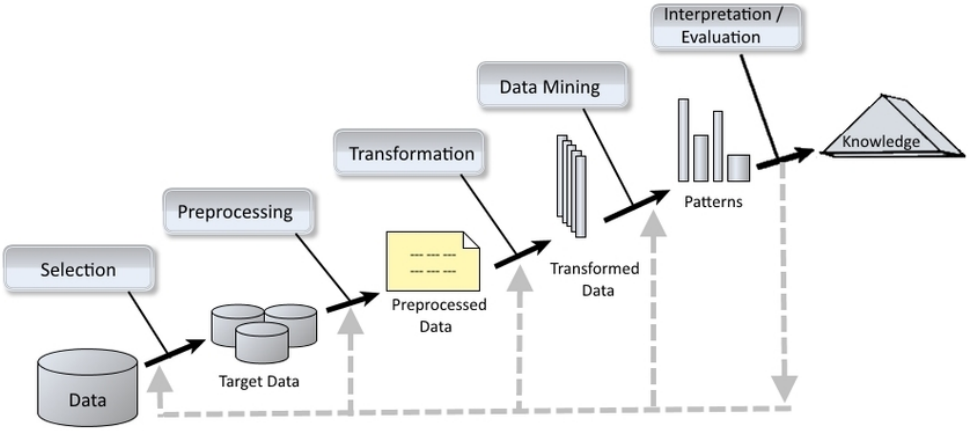

Figura 3.1: Diagrama do *Knowledge Discovery Process* (Ilustrado e disponibilizado publicamente por Heloisa Candello).

3.1 *Knowledge Discovery Process*

O *Knowledge Discovery Process* (KDP), ou *Knowledge Discovery in Databases*, é um processo que foi criado a partir da academia [50]. Criado por Fayyad em 1996, foi definido como o processo não trivial de identificar padrões válidos, novos, potencialmente úteis e, em última análise, compreensíveis nos dados, por meio de etapas interativas e iterativas [51]. Interativo porque o autor do trabalho interage constantemente com os dados e os usuários do problema. Iterativo porque o pesquisador pode voltar nas etapas anteriores para obter melhores resultados. Parecido com o CRISP-DM, o KDP tem 9 etapas, sendo elas [51]:

1. **Entendimento do Negócio:** identificar os problemas dos usuários e compreender os conhecimentos necessários para a construção de soluções no domínio da aplicação. Esse entendimento e os objetivos do processo de obtenção do conhecimento foram introduzidos no Capítulo 1.

2. **Seleção e Extração dos Dados:** selecionar um conjunto de dados ou focar em um subconjunto de variáveis ou amostras de dados, no qual a descoberta será realizada.

3. **Pré-processamento dos Dados:** limpar e remover ruído nos dados, determinando estratégias para o tratamento de campos com dados ausentes.

4. **Transformação dos Dados:** redução e projeção dos dados, de forma a encontrar recursos úteis para representar as instâncias de estudo em relação ao objetivo da tarefa. O número efetivo de variáveis em consideração pode ser reduzido ou projetado em representações invariantes.

5. **Identificação do Tipo de Problema:** combinar os objetivos do processo (etapa 1) a um determinado método de mineração de dados, por exemplo, classificação, regressão ou clusterização. A definição do tipo de problema (classificação) é apresentada na Seção 2.5.

6. **Escolha dos Algoritmos:** determinar os algoritmos que serão utilizados para solucionar o problema a partir da sua identificação. Os algoritmos usados neste trabalho são apresentados no Capítulo 2.

7. **Aplicação dos Algoritmos:** executar os experimentos de forma a obter padrões de interesse a partir da representação dos dados feita após as etapas anteriores. São apresentadas também algumas técnicas utilizadas para generalização e melhoria dos algoritmos.

8. **Interpretação dos Resultados:** são apresentados os resultados dos experimentos, avaliando-os com as métricas propostas para o problema. Também são analisados os padrões encontrados pelas regras de associação e a estrutura das árvores de decisão.

9. **Consolidação do Conhecimento:** por fim, discute-se o conhecimento obtido a partir do processo como um todo e a disponibilização da ferramenta criada para o público, de forma a auxiliar gestores na prevenção da evasão estudantil.

3.2 Seleção e Extração dos Dados

A Universidade de Brasília possui um sistema integrado a um armazém de dados (do inglês, *data warehouse*) chamado Saiku, que contém diversas informações sobre os docentes e discentes da universidade. Dentre essas informações, estão os históricos acadêmicos dos estudantes de graduação. Os coordenadores de curso de graduação têm acesso a esse sistema e às informações dos alunos de seus respectivos cursos, com objetivos auxiliá-los no acompanhamento dos alunos e na tomada de decisões.

O coordenador do curso de Engenharia Mecatrônica da UnB extraiu os históricos dos estudantes do curso, de forma que não fosse possível identificar os alunos a partir dos dados, e estes são os dados utilizados neste trabalho. A extração foi realizada com a seguinte *query*:

```
WITH
SET [~ROWS_ALUNO_ALUNO.ALUNO_RA] AS
    {[ALUNO.ALUNO_RA].[ALUNO_RA].Members}
SET [~ROWS_ALUNO_ALUNO.ALUNO_PERIODO_INGRESSO] AS
    {[ALUNO.ALUNO_PERIODO_INGRESSO].[ALUNO_PERIODO_INGRESSO].Members}
SET [~ROWS_ALUNO_ALUNO.ALUNO_FORMA_INGRESSO] AS
    {[ALUNO.ALUNO_FORMA_INGRESSO].[ALUNO_FORMA_INGRESSO].Members}
SET [~ROWS_ALUNO_ALUNO.ALUNO_PERIODO_SAIDA] AS
    {[ALUNO.ALUNO_PERIODO_SAIDA].[ALUNO_PERIODO_SAIDA].Members}
SET [~ROWS_ALUNO_ALUNO.ALUNO_FORMA_SAIDA] AS
    {[ALUNO.ALUNO_FORMA_SAIDA].[ALUNO_FORMA_SAIDA].Members}
SET [~ROWS_PERIODO_REFERENCIA_PERIODO_REFERENCIA.SEMESTRE_ANO_REFERENCIA] AS
    {[PERIODO_REFERENCIA.SEMESTRE_ANO_REFERENCIA].[SEMESTRE_ANO_REFERENCIA].Members}
SET [~ROWS_DISCIPLINA_DISCIPLINA.DISC_CODIGO] AS
    {[DISCIPLINA.DISC_CODIGO].[DISC_CODIGO].Members}
SET [~ROWS_DISCIPLINA_DISCIPLINA.DISC_MENCAO] AS
    {[DISCIPLINA.DISC_MENCAO].[DISC_MENCAO].Members}
SET [~ROWS_DISCIPLINA_DISCIPLINA.DISC_TURMA] AS
    {[DISCIPLINA.DISC_TURMA].[DISC_TURMA].Members}
SET [~ROWS_DISCIPLINA_DISCIPLINA.DISC_NOME] AS
    {[DISCIPLINA.DISC_NOME].[DISC_NOME].Members}
```

Tabela 3.1: Descrição das variáveis dos dados.

Variável	**Descrição**
`IdAluno`	Número que identifica o aluno anonimizado.
`SemestreIngresso`	Número de 5 dígitos que representa o semestre de ingresso. Os 4 primeiros dígitos são o ano de ingresso e o último é o semestre.
`SemestreFinal`	Número de 5 dígitos que representa o semestre que o estudante saiu do curso (formado ou evadido). Tem o mesmo formato da variável `SemestreIngresso`.
`SemestreMateria`	Número de 5 dígitos que representa o semestre que o estudante cursou a matéria. Tem o mesmo formato da variável `SemestreIngresso`.
`CodigoMateria`	Código da matéria que o aluno cursou em um dado semestre.
`Conceito`	Menção do aluno ao cursar aquela matéria naquele semestre.
`StatusFinal`	Situação final do aluno no curso.

```
SELECT
NON EMPTY {[Measures].[QUANTIDADE_ALUNOS]} ON COLUMNS,
NON EMPTY NonEmptyCrossJoin([~ROWS_ALUNO_ALUNO.ALUNO_RA],
NonEmptyCrossJoin([~ROWS_ALUNO_ALUNO.ALUNO_PERIODO_INGRESSO],
NonEmptyCrossJoin([~ROWS_ALUNO_ALUNO.ALUNO_FORMA_INGRESSO],
NonEmptyCrossJoin([~ROWS_ALUNO_ALUNO.ALUNO_PERIODO_SAIDA],
NonEmptyCrossJoin([~ROWS_ALUNO_ALUNO.ALUNO_FORMA_SAIDA],
NonEmptyCrossJoin([~ROWS_PERIODO_REFERENCIA_PERIODO_REFERENCIA.SEMESTRE_ANO_REFERENCIA],
NonEmptyCrossJoin([~ROWS_DISCIPLINA_DISCIPLINA.DISC_CODIGO],
NonEmptyCrossJoin([~ROWS_DISCIPLINA_DISCIPLINA.DISC_MENCAO],
NonEmptyCrossJoin([~ROWS_DISCIPLINA_DISCIPLINA.DISC_TURMA],
[~ROWS_DISCIPLINA_DISCIPLINA.DISC_NOME])))))))))))) ON ROWS
FROM [Dados Academicos Mecatronica]
```

Os dados coletados são descritos na Tabela 3.1. Foram considerados os dados somente dos alunos que não estavam ativos, ou seja, que não estão cursando atualmente Engenharia Mecatrônica. Cada linha do conjunto de dados representa uma matéria que um estudante cursou em um dado semestre e o desempenho obtido. Cada linha também contém a informação da situação do aluno quando saiu do curso. A coluna `SemestreFinal` somente é utilizada na análise dos dados, na Seção 3.4.1, logo é descartada do processo de modelagem dos dados para uso nos classificadores. Nos dados, há vários valores para a coluna `StatusFinal` (como “Formatura”, “Desligamento - Abandono”, “Desligamento Voluntário”, entre outros), sendo que para classificá-los em evadidos e formados só pode haver dois valores possíveis para essa coluna. Isto posto, faz-se necessário pré-processar os dados.

3.3 Pré-processamento dos Dados

O primeiro pré-processamento necessário se dá ao analisar a coluna `StatusFinal`. A partir dos dados, o aluno será considerado como evadido caso não seja listado como formado. Os estudantes que vieram a falecer durante o curso são ignorados pois não concluíram o curso mas também não evadiram. Portanto, os valores da coluna `StatusFinal` são alterados da seguinte forma:

- "FORMADO", se `StatusFinal` for "Formatura";
- "EVADIDO", caso contrário.

Como dito na Seção 2.1, será avaliado somente o histórico do aluno em seu primeiro ano letivo. Logo, é preciso filtrar os dados. É criada uma nova coluna chamada `Semestre` que representa o período no qual o estudante cursou a matéria a partir do seu ingresso no curso. Os valores dessa nova coluna são calculados a partir das colunas `SemestreIngresso` e `SemestreMateria`. Para filtrar os dados pelos históricos dos dois primeiros semestres de cada aluno, são removidas as linhas nas quais o `Semestre` for maior que 2. Além disso, há também a filtragem pelos códigos das matérias obrigatórias de Engenharia Mecatrônica, segundo a definição do fluxo do curso[1], sendo elas:

1º Semestre

- Química Geral Teórica (114626)
- Química Geral Experimental (114634)
- Física 1 (118001)
- Física 1 Experimental (118010)
- Algoritmos e Programação de Computadores (113476)
- Cálculo 1 (113034)
- Introdução a Engenharia Mecatrônica (168891)

2º Semestre

- Cálculo 2 (113042)
- Introdução a Álgebra Linear (113093)
- Física 2 (118028)
- Física 2 Experimental (118036)

[1] `https://matriculaweb.unb.br/graduacao/fluxo.aspx?cod=6912`

- Probabilidade e Estatística (115045)
- Desenho Mecânico Assistido por Computador 1 (168874)

Após fazer esse recorte, o conjunto de dados possui pouco mais de mil alunos diferentes, que ingressaram entre o 2º semestre de 1997 e o 1º semestre de 2018, sendo que as classes formado e evadido podem ser consideradas balanceadas (54% formados, 46% evadidos).

3.4 Transformação dos Dados

Com dados limpos e normalizados, pode-se avançar na transformação dos dados. O objetivo desta etapa é estruturar o conjunto dos dados para criar os modelos de entrada. Um modelo de entrada é uma forma de organizar os dados para que sirvam de insumo para os classificadores. No problema apresentado, uma instância de um modelo de entrada deve representar o histórico acadêmico do primeiro ano letivo de um aluno e seu `StatusFinal` (formado ou evadido).

Neste trabalho, são propostos três modelos de entrada para os classificadores. Utilizar diversos modelos parte da ideia de melhorar o desempenho da solução a partir de engenharia de características. Engenharia de características, do inglês *feature engineering*, é o processo de utilizar conhecimento do domínio do problema para criar características que melhorem o desempenho dos algoritmos de aprendizagem de máquina. Os modelos irão avaliar diferentes níveis de especificidade nos dados.

Como não há estudo sobre fatores que levam à evasão no curso de Engenharia Mecatrônica na UnB, esses modelos serão construídos de forma arbitrária de modo a ter uma base comparativa para futuras análises de outros modelos. A coluna `Conceito` tem os seguintes possíveis valores e seus respectivos equivalentes em notas de 0 a 10: SS (9,0 a 10), MS (7,0 a 8,9), MM (5,0 a 6,9), CC (Crédito Concedido, sem nota vinculada), MI (3,0 a 4,9), II (0,1 a 2,9), SR (zero), TR (trancamento parcial de matéria, não conta como reprovação nem aprovação), TJ (trancamento justificado da matéria, similar ao TR). No primeiro modelo, será avaliada somente a informação de se o aluno reprovou reprovou cada matéria. Dessa forma, são considerados SR, II e MI como reprovação e todos os outros como não reprovação.

O segundo modelo é semelhante ao Modelo 1 exceto que é usada a informação exata do conceito do aluno naquela disciplina, ou seja, a menção que o estudante obteve na disciplina. Por fim, o terceiro modelo tem as mesmas informações do segundo modelo adicionadas a quantidade de créditos reprovados a cada semestre do curso e a quantidade total de créditos reprovados ao final do primeiro ano de curso. A quantidade de créditos de cada matéria foi extraída diretamente do sistema MatriculaWeb e corresponde a uma medida da carga horária de cada disciplina.

Para a criação dos modelos apresentados, é necessário fazer transformações para que uma linha no conjunto de dados corresponda ao histórico de um aluno. Com o objetivo de associar a informação da disciplina com o semestre no qual ela foi cursada, são concatenadas as colunas `Semestre` e `CodigoMateria`, gerando a coluna `Semestre_Materia`. Os valores dessa nova coluna são os valores das duas colunas unidos pelo caractere '_'. Dessa maneira, a concatenação dos valores 1 (`Semestre`) e 113034 (`CodigoMateria`) será `1_113034`.

Com isso, são removidas as colunas `Semestre`, `CodigoMateria`, `SemestreIngresso` e `SemestreMateria`. Logo, as colunas presentes no conjunto de dados após essa transformação são:

IdAluno	Conceito	Semestre_Materia	StatusFinal

Tabela 3.2: Colunas dos dados após 1ª transformação

Por fim, é necessário transpor o conjunto de dados, agrupando-os pelas colunas `IdAluno` e `StatusFinal`, de forma que para cada valor em `Semestre_Materia` crie-se uma nova coluna que terá como valor o `Conceito` que o aluno obteve naquela matéria. As tabelas 3.3 e 3.4 exemplificam a transposição.

IdAluno	Conceito	Semestre_Materia	StatusFinal
1234	MS	1_113034	FORMADO

Tabela 3.3: Exemplo de linha nos dados antes da transposição

IdAluno	1_113034	StatusFinal
1234	MS	FORMADO

Tabela 3.4: Exemplo de linha nos dados depois da transposição

Uma vez que cada linha no conjunto dos dados representa o histórico de um aluno nas matérias obrigatórias do primeiro ano letivo do curso, são feitas as transformações inerentes a cada modelo proposto. No primeiro modelo, as menções SR, II e MI são traduzidas para o valor inteiro 1 e todas as outras menções para 0.

No segundo modelo, é feito um mapeamento entre cada menção possível para cada matéria de forma a criar novas colunas. Exemplificando, a partir da coluna `1_113034` serão criadas as colunas `1_113034_MM`, `1_113034_MS`, `1_113034_SS`, etc. O aluno terá o valor inteiro 1 na coluna `1_113034_MS` se tiver tirado MS ao cursar a disciplina 113034 no 1º semestre, ou o valor 0 caso não tiver obtido tal menção.

Por fim, o terceiro modelo é construído a partir do segundo modelo, adicionando as colunas `CreditosReprovados1`, `CreditosReprovados2` e `CreditosReprovadosTotal`,

que guardam a quantidade de créditos reprovados no 1º semestre, 2º semestre e no final do primeiro ano, respectivamente.

Além dos três modelos usados para os classificadores, foi criado um quarto modelo que será utilizado pelo Apriori para identificação das regras de associação. A ideia desse modelo é fazer com que o histórico do aluno se transforme em uma série de transações. Dessa forma, cada menção em um matéria em um determinado semestre é uma transação no histórico do aluno. Cada linha no modelo 4 terá todo o histórico do aluno, levando em consideração a matéria, o semestre e a menção. Essas transações serão usadas para identificação das transações que mais aparecem quando o aluno evade. Por exemplo, uma linha de um aluno poderia ser:

IdAluno	0	1	2	...	StatusFinal
321	1_113034_MS	1_118001_MI	1_118010_MM	...	EVADIDO

Tabela 3.5: Exemplo de linha no modelo 4, com as transações do histórico do aluno

3.4.1 Análise dos Dados

Após fazer as transformações, passa-se à análise estatística dos dados com mais facilidade, visto que cada instância nos modelos de entrada representa um único estudante. Esta análise é feita a fim de ter um entendimento qualitativo dos dados e possibilitar o cruzamento dessa análise com os resultados dos classificadores.

Como dito na Seção 2.1, o maior índice de evasão geralmente acontece logo após o primeiro ano letivo. Filtrando pelos alunos evadidos e analisando a coluna `SemestreFinal`, foi construído o gráfico na Figura 3.2, onde pode-se observar tal comportamento. É constatado que a maior quantidade de evasões ocorre no terceiro semestre, corroborando com a decisão de avaliar somente o primeiro ano letivo, de forma que coordenadores de curso possam intervir antes das evasões acontecerem.

Analisando o desempenho nas matérias, os maiores índices de reprovação estão nas matérias Física 1 e Cálculo 1. Em relação à Física 1, 20.21% de todos os estudantes reprovam no 1º semestre. Por outro lado, considerando somente os alunos evadidos, 37.01% reprovam a matéria. Ao fazer a mesma análise para somente os alunos formados, o índice de reprovação é de 6.15%. Em relação a Cálculo 1 , apenas 17.56% do total de estudantes reprovam a matéria no primeiro semestre. Considerando somente os evasores, essa proporção cresce para 34.30% e somente os formados, o índice é de 3.56%. Essas duas matérias têm os maiores índices de reprovação no 1º semestre entre os evadidos. A Figura Figura 3.3 simplifica a visualização desses índices de reprovação.

Em média, os estudantes reprovam 4 créditos no 1º semestre, com um desvio-padrão de 7 créditos. No segundo semestre, são reprovados 2 créditos em média, com desvio-

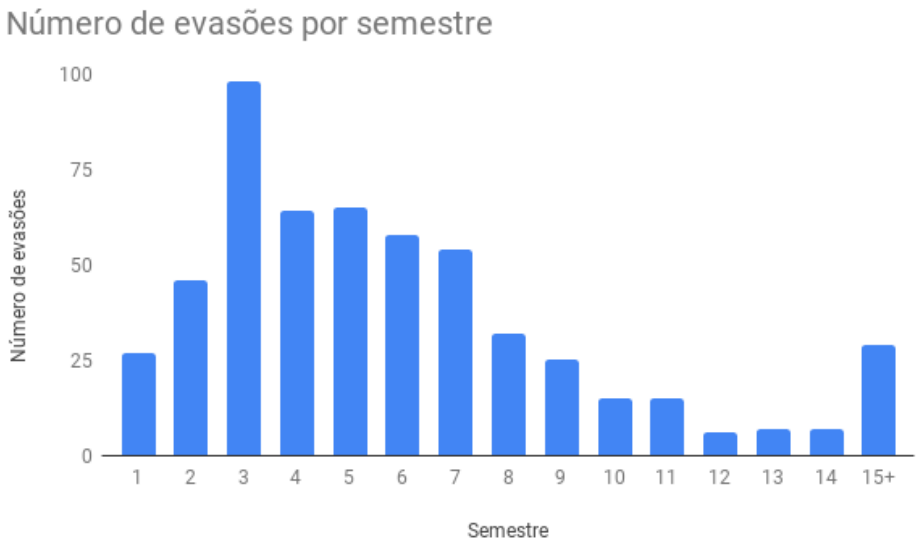

Figura 3.2: Número de evasões por semestre na Engenharia Mecatrônica.

padrão de 5 créditos. No total, os alunos reprovam 6 créditos no primeiro ano letivo, com um desvio-padrão de 10 créditos.

A menção mais comum é MS, com 36.22% do total de aparições nas menções obtidas, seguida de MM com 30.30%. Para os alunos que cursaram Desenho Mecânico Assistido por Computador 1 no 2º semestre, 60.36% deles tiraram MS. Ao observar essa matéria em relação a menções aprovantes, temos que a proporção de aprovados ao cursar essa matéria nesse período é de 96.55%. Se fizermos essa mesma análise para a matéria Introdução à Álgebra Linear (código 113093), temos que 82.10% dos alunos que cursaram tal matéria conseguiram a aprovação nela no segundo semestre. Se formos mais específicos e analisarmos essa matéria em relação aos estudantes que evadiram, 58.41% conseguiram uma nota de aprovação ao cursar a matéria no segundo semestre.

3.5 Aplicação dos Algoritmos

Na 7ª etapa do KDP, são usadas técnicas para implementação, generalização e aperfeiçoamento dos algoritmos. Para execução dos experimentos, foi utilizada a linguagem de programação Python[2], auxiliada da biblioteca Scikit-learn[3], que implementa vários algoritmos de aprendizagem de máquina. Para o algoritmo Apriori foi usada a biblioteca

[2] `https://www.python.org/`
[3] `https://scikit-learn.org/`

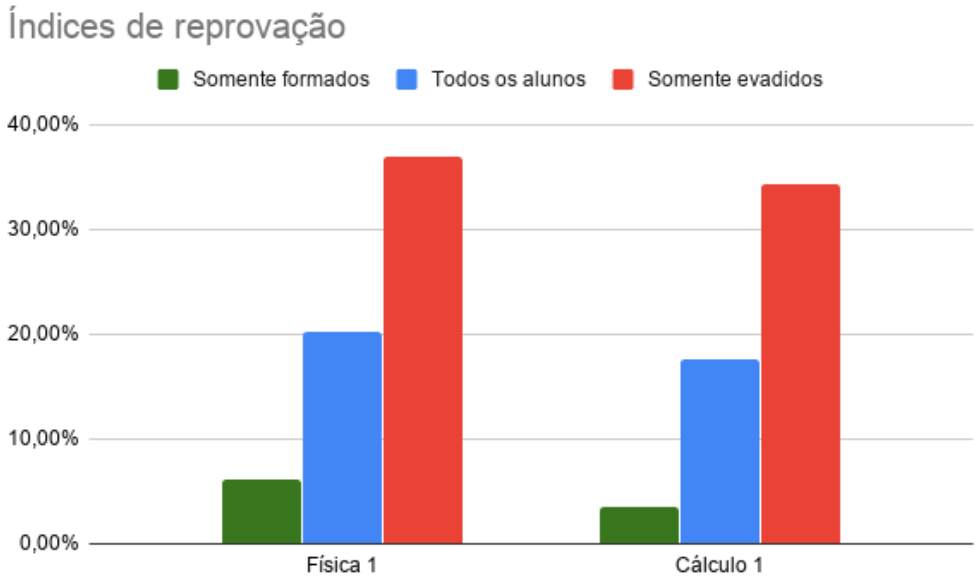

Figura 3.3: Índices de reprovação em Física 1 e Cálculo 1 considerando todos os alunos e somente os evadidos.

Apyori[4]. A escolha por essas ferramentas se deu pela facilidade na implementação, pela comunidade de desenvolvedores e pela familiaridade do pesquisador em relação a elas.

Como dito na Seção 2.5, a classificação de dados ocorre em duas etapas: treinamento e teste. Isto posto, o conjunto de dados é separado de forma estratificada em dados para treinamento (70%) e dados de teste (30%). Estratificação é o processo de dividir membros de uma população em subgrupos homogêneos [52], ou seja, que tenham a mesma proporção das classes.

Antes do treinamento dos classificadores e sua avaliação por meio dos dados de teste, são necessárias algumas etapas prévias à execução que generalizem os classificadores e permitam melhores resultados. Foi utilizada validação cruzada para generalização e ajustes nos hiperparâmetros para a obtenção de melhores resultados.

3.5.1 Sobre-ajuste e Validação Cruzada

Ao treinar o classificador, é possível que este tenha um ótimo desempenho nos dados de treinamento, mas o desempenho nos dados de teste seja muito aquém do esperado. Esse fenômeno se chama sobre-ajuste (*overfitting*, em inglês) e deve ser evitado, de modo a construir classificadores genéricos [26]. Sobre-ajuste é o uso de modelos ou procedimentos que incluem mais termos do que os necessários ou usam abordagens mais complicadas do

[4] `https://pypi.org/project/apyori/`

que as necessárias [53]. Logo, sobre-ajuste ocorre quando treinamos em demasia o nosso classificador e este se torna um classificador perfeito para os dados de treinamento, mas não são generalizados o suficiente para outros dados de entrada [24]. Uma das formas de evitar sobre-ajuste é validação cruzada.

Validação cruzada é uma técnica para avaliar a capacidade de generalização de um modelo, a partir de um conjunto de dados [54], onde o conjunto de treinamento é dividido em subconjuntos mutuamente exclusivos e de tamanho igual e, para cada subconjunto, o classificador é treinado na união de todos os outros subconjuntos [25]. Um dos métodos mais utilizadas para se fazer validação cruzada [55] é k-fold, no qual os dados iniciais são particionados aleatoriamente em k subconjuntos mutualmente exclusivos, D_1, D_2, ..., D_k, cada um com tamanho aproximadamente igual [23]. As etapas de treinamento e teste são realizadas k vezes. Na iteração i, a partição D_i é separada como o conjunto de testes e as outras partições são agrupadas e utilizadas para treinar o modelo. A partir dos resultados de Kohavi [54] e das práticas de validação cruzada nos trabalhos correlatos [8, 11, 9], este trabalho utiliza validação cruzada estratificada (subdivisões com classes balanceadas [20]) 10-fold, ou seja, 10 divisões diferentes dos dados.

3.5.2 Ajuste de hiperparâmetros

A implementação dos algoritmos pelo Scikit-learn possibilita o ajuste de hiperparâmetros visando o aprimoramento dos resultados. Um hiperparâmetro é um parâmetro do algoritmo cujo valor deve ser definido antes do processo de treinamento [56]. Os outros parâmetros de um algoritmo de aprendizagem de máquina são definidos durante o treinamento.

Para cada tipo de classificador, será feita *grid search* em alguns de seus hiperparâmetros. *Grid search* é a definição de alguns valores para hiperparâmetros de um classificador e a avaliação de todas as combinações possíveis desses hiperparâmetros a fim de obter a melhor configuração avaliada de cada classificador, dada uma métrica [57]. Na Tabela 3.6 estão os hiperparâmetros ajustados nos classificadores e suas descrições.

Feito o *grid search* e a generalização dos classificadores por meio da validação cruzada, são executados os experimentos e analisados os resultados produzidos. Vale ressaltar que o Apriori não utiliza validação cruzada, nem *grid search* e nem tem o conjunto de dados separado em treinamento e teste, uma vez que as regras de associação devem utilizar todos os dados para identificar as observações mais relevantes dada uma saída desejada.

Algoritmo	**Nome**	**Descrição**	**Valores**
Árvores de Decisão	*criterion*	Critério para avaliar a qualidade da divisão no nó	gini, entropia
	splitter	Estratégia para a escolha da divisão do nó	melhor divisão, divisão aleatória
	min_samples_split	Quantidade mínima de instâncias para fazer a divisão do nó	{10%, 20%, 30%, ..., 100%}
Florestas Aleatórias	*criterion*	Critério para avaliar a qualidade da divisão no nó	gini, entropia
	n_estimators	Número de árvores de decisão na floresta	{10, 30, 50, 70, ..., 190}
Regressão Logística	*solver*	Algoritmo usado no problema de otimização	liblinear, lbfgs
	C	Controle da generalização do classificador	{0.001, 0.01, 0.1, 1, ..., 100000}
Redes Neurais	*alpha*	Multiplicador do erro associado	{0.00001, 0.0001, 0.001, 0.01, 0.1}

Tabela 3.6: Hiperparâmetros ajustados na aplicação dos classificadores

Capítulo 4

Resultados

Este capítulo faz a interpretação dos resultados obtidos, etapa 8 do KDP. Após o treinamento dos algoritmos e sua avaliação com os dados de teste, são apresentados e analisados quantitativamente e qualitativamente os resultados dos melhores classificadores de cada algoritmo para cada modelo de entrada. Mas antes de explorar os produtos dos experimentos, é necessário definir as métricas a serem observadas.

4.1 Métricas

Para o melhor entendimento desses dois conceitos é apresentada a matriz de confusão. Apesar do nome enganoso, a matriz de confusão é uma tabela (apresentada na Tabela 4.1) que auxilia a assimilação de como os classificadores estão avaliando uma determinada entrada [23]. Na matriz de confusão, cada predição feita pelo classificador será separada em uma das quatro categorias apresentadas, dado que a classe é binária (no problema da evasão, evadido ou não evadido). As métricas que surgem a partir da matriz de confusão exploram maximizar uma ou mais categorias sobre as outras.

		Valor Verdadeiro	
		Positivo	Negativo
Valor Previsto	Positivo	Verdadeiros Positivos	Falsos Positivos
	Negativo	Falsos Negativos	Verdadeiros Negativos

Tabela 4.1: Matriz de Confusão

Um dos objetivos deste trabalho é servir de insumo para que coordenadores de cursos de graduação na UnB possam identificar quais são os alunos que têm risco de evadir e poder auxiliá-los e reverter essa situação. Portanto, é interessante que, dentre as pessoas que irão evadir, seja possível identificar o maior número delas. Logo, prioriza-se maximizar a sensibilidade (*recall* em inglês) do modelo.

Sensibilidade é a habilidade do classificador de encontrar todas as instâncias relevantes para o problema [58]. Portanto, sensibilidade será a proporção de verdadeiros positivos (VP) sobre todas todas as entradas que de fato são positivas, ou seja verdadeiros positivos e falsos negativos (FN). De acordo com a matriz de confusão, sua fórmula é dada por:

$$sensibilidade = \frac{VP}{VP + FN} \tag{4.1}$$

Será observada também a acurácia dos classificadores. Acurácia é a capacidade do modelo de conseguir classificar corretamente as entradas em relação aos seus valores verdadeiros [58]. Dessa forma, a fórmula da acurácia se dá por:

$$acurácia = \frac{VP + VN}{VP + FP + FN + VN} \tag{4.2}$$

sendo VN os verdadeiros negativos e FP os falsos positivos.

4.2 Resultados e Análise

Após execução dos experimentos, os melhores classificadores de cada algoritmo são analisados em relação às métricas descritas acima. A análise é feita para cada modelo de entrada construído. Para ter uma noção da variância das métricas de cada classificador, os experimentos foram executados 20 vezes e coletados os desvios padrão das métricas. Os resultados são apresentados nas Tabelas 4.2 a 4.3, que mostram o resultado do melhor classificador gerado para cada algoritmo. Vale ressaltar que o resultado apresentado quanto à sensibilidade e quanto à acurácia se tratam do mesmo classificador. Ou seja, o melhor classificador gerado utilizando Regressão Logística tendo o o modelo de entrada 1 obteve uma sensibilidade de 57.42% e acurácia de 75.00%.

	Sensibilidade		
	Modelo 1	Modelo 2	Modelo 3
Árvores de Decisão	65.23 % ± 0.30 %	73.55 % ± 0.00 %	**82.58% ± 0.0%**
Florestas Aleatórias	64.28 % ± 1.01 %	69.71 % ± 1.83 %	72.22 % ± 1.43 %
Redes Neurais	64.48 % ± 1.30 %	73.34 % ± 1.24 %	74.84 % ± 1.79 %
Regressão Logística	57.42 % ± 0.00 %	70.97 % ± 0.00 %	71.61 % ± 0.00 %

Tabela 4.2: Resultados dos classificadores em relação à sensibilidade para cada modelo de entrada

Na análise da sensibilidade, observa-se que Árvores de Decisão obtém os melhores resultados para todos os modelos, chegando a 82.58% ao executar o experimento com o

	acurácia		
	Modelo 1	Modelo 2	Modelo 3
Árvores de Decisão	$75.62\,\% \pm 0.14\,\%$	$78.82\,\% \pm 0.00\,\%$	$78.82\,\% \pm 0.00\,\%$
Florestas Aleatórias	$75.15\,\% \pm 0.66\,\%$	$79.55\,\% \pm 0.90\,\%$	$80.82\,\% \pm 0.82\,\%$
Redes Neurais	$75.15\,\% \pm 0.47\,\%$	$78.00\,\% \pm 0.73\,\%$	$78.16\,\% \pm 0.92\,\%$
Regressão Logística	$75.00\,\% \pm 0.00\,\%$	$80.59\,\% \pm 0.00\,\%$	$\mathbf{81.18\% \pm 0.0\%}$

Tabela 4.3: Resultados dos classificadores em relação à acurácia para cada modelo de entrada

terceiro modelo. Quanto à acurácia, Regressão Logística se destacou ao atingir 81.18%, seguido de Florestas Aleatórias com 80.82%.

A partir da hipótese de que é possível predizer se um aluno irá evadir a partir do seu histórico, foram obtidos classificadores com mais de 82% de sensibilidade e 81% de acurácia, implicando que é possível identificar alunos em risco de evasão com um alto grau de acurácia. Além de não haver ferramenta assim disponível para a identificação de alunos em risco de evasão na UnB, os resultados atingidos são consideráveis quando comparados aos trabalhos correlatos, que obtêm resultados similares mas utilizando uma quantidade de dados maior em uma ordem de grandeza. Como ter mais dados ajuda a ter melhores resultados, os resultados deste trabalho são consideráveis.

Vale ressaltar o impacto que engenharia de características tem sobre os resultados. Engenharia de características (do inglês, *feature engineering*) é o ato de extrair características dos dados e formatá-los para que possam ser usados em um algoritmo de aprendizagem, objetivando melhores resultados [59]. Nos experimentos, essa técnica foi empregada ao criar novas colunas com as quantidades de créditos reprovados por semestre e ao final do ano letivo, obtendo um acréscimo de quase 10% em sensibilidade para Árvores de Decisão. Além disso, ao comparar os resultados do módulo 1 com os modelos 2 e 3, fica evidente que avaliar o desempenho estudantil com um maior grau de especificidade ao analisar a menção e não somente se houve aprovação ou reprovação gera resultados melhores.

4.2.1 Estrutura da Árvore de Decisão

Além de obter os melhores resultados quanto à sensibilidade, Árvores de Decisão também se destacam pela sua simplicidade e transparência [27]. Após o treino e construção da árvore, pode-se analisar cada nó para obter mais conhecimento sobre quais características mais influenciam na classificação das instâncias. No problema deste trabalho, analisar a estrutura da árvore pode trazer entendimento acerca de quais matérias são as mais relevantes na determinação da trajetória do aluno, segundo o algoritmo.

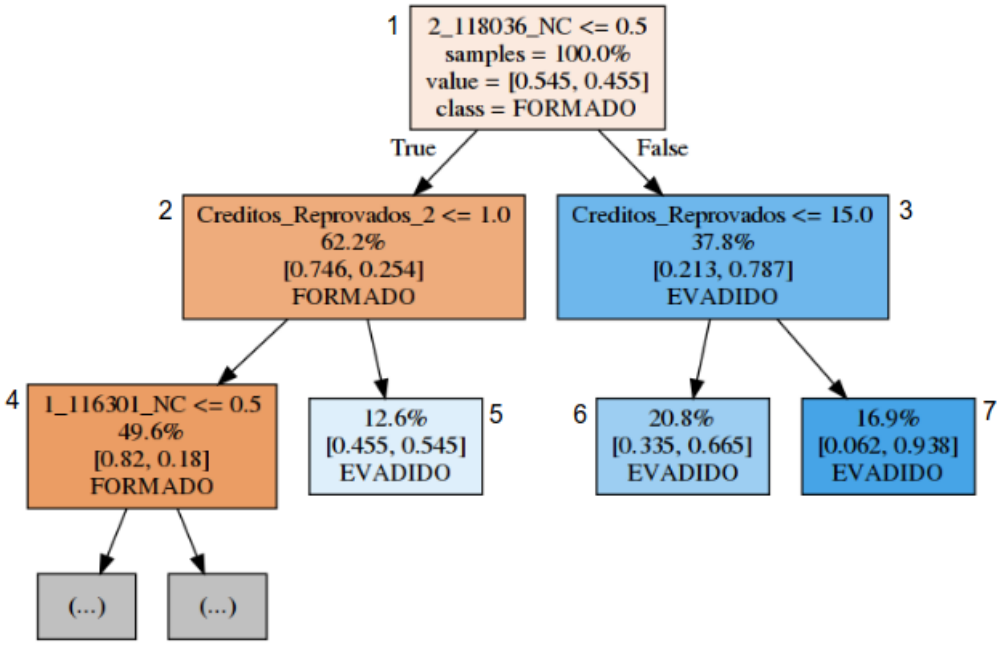

Figura 4.1: Estrutura da Árvore de Decisão com o Modelo 3 como entrada.

A Figura 4.1 apresenta a estrutura da Árvore de Decisão tendo o terceiro modelo como entrada. O número ao lado de cada nó é seu identificador. Cada nó tem 4 linhas, sendo elas:

1. Condição para a tomada de decisão. A partir disso, os dados são separados entre os que têm e os que não têm tal condição.
2. Proporção da quantidade total de instâncias a serem analisados no nó.
3. Proporção das instâncias de cada classe no nó. O primeiro valor é a proporção de formados e a segunda de evadidos.
4. Classe da maioria das instâncias do nó.

Na raiz da árvore, tem-se que a primeira comparação se faz na matéria Física 2 Experimental (código 118036). A condição é a de se o estudante não cursou (NC) a matéria no segundo semestre. Se tiver cursado, o valor que terá na coluna será 0 (pois terá valor 1 sob a coluna da menção que tiver obtido) a condição será verdadeira e a instância descerá para o nó 2, tendo 74.6% de chances de formar no curso. Caso contrário, irá para o nó 3, onde será classificada como evadida, tendo então 78.7% de chances de ser de fato evadida.

Continuando no nó 3, se o aluno tiver menos que 15 créditos reprovados ao final do primeiro ano letivo, suas chances de evadir de fato diminuem para 66.5% (nó 6). Mas caso contrário, terá 93.8% de chances de evasão (nó 7).

Voltado ao nó 2, a condição se dá ao avaliar se o aluno tem menos que 1 crédito reprovado no segundo semestre. Como as matérias têm pelo menos 2 créditos, essa pergunta pode ser lida como a avaliação de se o aluno não reprovou nenhuma matéria no 2º semestre. Se não tiver reprovado, a instância segue para o nó 4, aumentando suas chances de formar para 82% (nó 4). Por outro lado, se tiver reprovação, seguirá para o nó 5, tendo 54.5% de chances de evadir, sendo classificado como evadido pelo algoritmo.

Por fim, o nó 4 avalia se o aluno não cursou Algoritmos e Programação de Computadores, antiga Computação Básica (código 116301) no primeiro semestre. A partir dessa questão, a estrutura da árvore segue sendo construída, com perguntas menos relevantes para a determinação da evasão.

Ao analisar a estrutura da Árvore de Decisão, observa-se que a raiz da árvore avalia se o estudante cursou ou não Física 2 Experimental no 2º semestre. Essa matéria está no segundo semestre do fluxo da Engenharia Mecatrônica e tem como pré-requisitos as matérias Cálculo 1, Física 1 e Física 1 Experimental. Como Cálculo 1 e Física 1 têm altas taxas de reprovação e reprová-las aumenta as chances de evadir, faz sentido que avaliar se o estudante conseguiu cursar a matéria no segundo semestre tenha muita relevância na identificação da evasão.

Além disso, a estrutura da árvore mostra que utilizar a quantidade de créditos reprovados por semestre e ao final do ano letivo é relevante. Foi descoberto que, ao não ter cursado Física 2 Experimental no 2º semestre e ter reprovado mais que 16 créditos ao final do primeiro ano letivo, o aluno tem aproximadamente 94% chances de evadir, por exemplo.

4.2.2 Regras de Associação

Como apresentado na Seção 2.10, a abordagem Apriori gera regras de associação que podem ser usadas para obter conhecimento pelo usuário final. O modelo de entrada 4 foi usado para a construção das associações visando identificar as principais transações (menção em uma matéria em um dado semestre) que implicam na evasão do aluno. A seguir são apresentadas as regras de associação obtidas dos experimentos com mais de 80% de confiança e 4% de suporte e que implicam na evasão.

> Se tirou SR em Física 1 no 1º semestre, então será EVADIDO com 100.0% de confiança e suporte de 4.15%.

> Se tirou II em Cálculo 1 no 1º semestre, então será EVADIDO com 96.15% de confiança e suporte de 4.41%.

> Se tirou SR em Cálculo 1 no 1º semestre, então será EVADIDO com 94.55% de confiança e suporte de 4.59%.

Se tirou SR em Algoritmos e Programação de Computadores no 1º semestre, então será EVADIDO com 94.23% de confiança e suporte de 4.32%.

Se tirou II em Física 1 no 1º semestre, então será EVADIDO com 91.78% de confiança e suporte de 5.91%.

Se tirou MI em Física 1 Experimental no 1º semestre, então será EVADIDO com 86.81% de confiança e suporte de 6.97%.

Se tirou II em Algoritmos e Programação de Computadores no 1º semestre, então será EVADIDO com 82.76% de confiança e suporte de 4.24%.

Se tirou MI em Cálculo 1 no 1º semestre, então será EVADIDO com 81.52% de confiança e suporte de 6.62%.

Vale ressaltar que, como esperado, as matérias com os maiores índices de reprovação (Física 1 e Cálculo 1) estão presentes nas regras com as maiores confianças. Usando a primeira regra como exemplo, tem-se que não há aluno na história do curso de Engenharia Mecatrônica (até o 1º semestre de 2018) que conseguiu se formar, mantendo a matrícula, após obter SR em Física 1.

Concluindo, foram encontradas várias informações que podem ajudar coordenadores de graduação na tomada de decisão e no auxílio de estudantes em risco. Por exemplo, foi identificado que todos os estudantes que obtiveram SR em Física 1 no 1º semestre acabaram evadindo. Ademais, 96% dos estudantes que obtiveram II em Cálculo 1 no 1º semestre evadiram do curso.

4.3 Disponibilizando os experimentos

O Apêndice A apresenta o código desenvolvido para os experimentos. Como dito anteriormente, o código foi escrito na linguagem de programação Python e foi disponibilizado com o objetivo de que possa ser verificado e reproduzido por outros pesquisadores.

No início do código, há duas variáveis que precisam ser inicializadas para a execução do código (`training_data_file_path` e `test_data_file_path`). A primeira é o caminho para o arquivo CSV com os dados de treinamento para os classificadores e a segunda o caminho para os dados dos alunos a serem classificados. O algoritmo irá pegar os dados do arquivo de treinamento, transformá-los e gerar as entradas. Essas entradas são utilizados para treinar os classificadores de forma a obter o melhor classificador para cada algoritmo. Esses classificadores então são usados para classificar os dados dos alunos ativos.

Após isso, os dados de treinamento são lidos, transformados e classificados. Os classificações feitas são escritas no arquivo `students_classifications.csv`, no qual são apresentadas as classificações feitas por cada algoritmo.

Capítulo 5

Conclusão

Evasão nas universidades é um tema de preocupação de diversos atores da Educação Superior e é uma adversidade que acarreta na falta de eficiência de recursos públicos e problemas sociais para os estudantes. Com base nisso e em trabalhos correlatos, partiu-se da hipótese de que é possível identificar acuradamente alunos em risco de evasão utilizando técnicas de mineração de dados.

Após extrair, selecionar e transformar os históricos acadêmicos dos alunos de um curso de graduação da Universidade de Brasília, foram aplicados algoritmos de classificação e de regras de associação. Os resultados obtidos corroboram com a hipótese inicial ao atingir uma sensibilidade de 82% e uma acurácia de 81%. Além disso, as regras identificadas a partir dos dados trazem conhecimento que pode ser empregado diretamente por coordenadores de graduação na prevenção da evasão.

Por fim, o conhecimento obtido por esse trabalho foi consolidado, seguindo a última etapa do KDP, em uma preditor disponível publicamente[1] para que atores da Educação pudessem utilizá-la para tomar decisões. Os classificadores gerados nessa ferramenta podem ser utilizados para identificar alunos que estão ativos e em risco de evasão. Os históricos acadêmicos utilizados para os experimentos não estão disponíveis publicamente. Nessa preditor, é possível obter classificações para estudantes ativos a partir de históricos de estudantes que já formaram ou evadiram no curso, além das regras de associação.

5.1 Trabalhos Futuros

Esta pesquisa foi o início da construção de soluções que possam auxiliar gestores da Educação na prevenção da evasão estudantil. Dessa forma, há vários caminhos para continuar esse trabalho. O primeiro deles é o de propagar o uso essa ferramenta para outros cursos da UnB, através da divulgação do trabalho e suporte aos departamentos

[1] `http://bit.ly/preditor-evasao-unb`

para utilização dos seus históricos e avaliação do impacto obtido. Além disso, espera-se que o conhecimento adquirido sirva de base para mais estudos e ações para prevenção da evasão.

Um caminho também é o de estudar mais os dados. Construir uma análise estatística mais aprofundada dos dados pode possibilitar a identificação de novas características que melhorem os resultados.

Outro caminho é o de aperfeiçoamento dos resultados dos classificadores. Os hiperparâmetros dos classificadores podem ser otimizados utilizando técnicas como o método do gradiente para obter os melhores valores para cada hiperparâmetro. Ademais, outros classificadores podem ser avaliados, como algoritmos de aprendizagem profunda (do inglês, deep learning), a fim de averiguar se eles obtém melhores resultados.

Por fim, também é interessante explorar outras informações que estejam disponíveis nos dados da UnB, como dados demográficos e sociais dos estudantes. Analisar a forma de ingresso e que tipo de escola estudou podem trazer muito conhecimento para os modelos, como visto em trabalhos correlatos [60, 61]. Outra alternativa é analisar o gênero dos estudantes. Por exemplo, no curso de Computação, a retenção das alunas tem diminuído nos últimos anos e é possível fazer uma análise desse perfil a fim de aumentar a retenção [62]. Uma outra ideia seria a de usar o CEP da residência do estudante para calcular a distância de sua casa até a universidade. Ainda, pode-se computar o IRA (Índice de Rendimento Acadêmico) parcial de cada semestre do indivíduo, de forma a ter uma noção de como está o rendimento geral do estudante a cada semestre.

Referências

[1] Ferreira, José Brites, de Lourdes Machado Maria e Magalhães Antônio M: *A "importância"e a "satisfação"no ensino superior: a perspectiva dos estudantes.* Em *X congresso da Sociedade Portuguesa de Ciências da Educação*, 2009. 1

[2] Moraes, Flavio Fava-de: *Universidade, inovação e impacto socioeconômico.* São Paulo em Perspectiva, 14(3):8–11, julho 2000. 1

[3] Psacharopoulos, George: *Returns to investment in education: A global update.* World development, 22(9):1325–1343, 1994. 1

[4] Orçamento, Decanato de Planejamento e: *Relatório de Gestão de 2018.* `http://www.dpo.unb.br/images/phocadownload/documentosdegestao/relatoriogestao/2018/Relatrio_de_Gesto_UnB_2018.pdf`. (Acessado em 09/06/2019). 1

[5] Santos Baggi, Cristiane Aparecida e Lopes, Doraci Alves dos: *Evasão e avaliação institucional no ensino superior: uma discussão bibliográfica.* Avaliação: Revista da Avaliação da Educação Superior, 16(2), 2011. 1

[6] Silva Filho, Roberto Leal Lobo e Motejunas, Paulo Roberto e Hipólito Oscar e Lobo Maria Beatriz Carvalho Melo: *A evasão no ensino superior brasileiro.* Cadernos de pesquisa, 37(132):641–659, 2007. 1

[7] Avaliação UnB, Comissão Própria de: *Relatório final de Autoavaliação Institucional 2019.* Relatório Técnico, Universidade de Brasilia, 2019. (Acessado em 08/07/2019). 1

[8] Aulck, Lovenoor, Dev Nambi, Nishant Velagapudi, Joshua Blumenstock e Jevin West: *Mining university registrar records to predict first-year undergraduate attrition.* Em *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, páginas 9–18, 2019. 3, 5, 23

[9] Delen, Dursun: *Predicting student attrition with data mining methods.* Journal of College Student Retention: Research, Theory & Practice, 13(1):17–35, 2011. 3, 10, 23

[10] Dekker, Gerben W. e Pechenizkiy, Mykola e Vleeshouwers Jan M.: *Predicting students drop out: A case study.* International Working Group on Educational Data Mining, 2009. 4

[11] Assis, Lucas Rocha Soares de: *Perfil de evasão no ensino superior brasileiro: uma abordagem de mineração de dados*. Tese de Mestrado, Universidade de Brasília, 2017. 4, 5, 23

[12] DeBerard, M. Scott, Glen Spielmans e Deana Julka: *Predictors of academic achievement e retention among college freshmen: A longitudinal study*. College student journal, 38(1):66–80, 2004. 4

[13] Hermanowicz, Joseph C.: *College attrition at American research universities: Comparative case studies*. Algora Publishing, 2003. 4

[14] Bean, John P: *Interaction effects based on class level in an explanatory model of college student dropout syndrome*. American Educational Research Journal, 22(1):35–64, 1985. 5

[15] Pascarella, Ernest T: *Student-faculty informal contact e college outcomes*. Review of educational research, 50(4):545–595, 1980. 5

[16] Tinto, Vincent e Pusser, Brian: *Moving from theory to action: Building a model of institutional action for student success*. National Postsecondary Education Cooperative, páginas 1–51, 2006. 5

[17] Tinto, Vincent: *Dropout from higher education: A theoretical synthesis of recent research*. Review of educational research, 45(1):89–125, 1975. 5

[18] Delen, Dursun: *A comparative analysis of machine learning techniques for student retention management*. Decision Support Systems, 49(4):498–506, 2010. 5

[19] Frawley, William J., Gregory Piatetsky-Shapiro e Christopher J. Matheus: *Knowledge discovery in databases: An overview*. AI magazine, 13(3):57–57, 1992. 5

[20] Witten, Ian H., Eibe Frank e Mark A. Hall: *Data Mining: Practical machine learning tools e techniques*. Morgan Kaufmann, 2016. 5, 7, 13, 23

[21] Goldschmidt, Ronaldo e Emmanuel Passos: *Data mining: um guia prático*. Gulf Professional Publishing, 2005. 5

[22] Mitchell, Tom M.: *Machine learning*. Burr Ridge, IL: McGraw Hill, 45(37):870–877, 1997. 5, 7

[23] Han, Jiawei e Pei, Jian e Kamber Micheline: *Data mining: concepts e techniques*. Elsevier, 2011. 5, 6, 8, 23, 25

[24] Marsland, Stephen: *Machine learning: an algorithmic perspective*. Chapman and Hall/CRC, 2011. 6, 23

[25] Kotsiantis, Sotiris B. e Zaharakis, I. e Pintelas P.: *Supervised machine learning: A review of classification techniques*. Emerging artificial intelligence applications in computer engineering, 160:3–24, 2007. 6, 23

[26] Shalev-Shwartz, Shai e Shai Ben-David: *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014. 7, 8, 9, 11, 12, 22

[27] Rokach, Lior e Oded Z Maimon: *Data mining with decision trees: theory and applications*, volume 69. World scientific, 2008. 7, 27

[28] Quinlan, J Ross: *C4. 5: programs for machine learning*. Elsevier, 2014. 7

[29] Quinlan, J. Ross: *Induction of decision trees*. Machine learning, 1(1):81–106, 1986. 7

[30] Rokach, Lior e Oded Maimon: *Top–down induction of decision trees classifiers–a survey*. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, página 487, 2005. 7

[31] Shannon, Claude Elwood: *A mathematical theory of communication*. Bell system technical journal, 27(3):379–423, 1948. 7

[32] Ho, Tin Kam: *Random decision forests*. Em *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, páginas 278–282. IEEE, 1995. 9

[33] Breiman, Leo: *Random forests*. Machine learning, 45(1):5–32, 2001. 9

[34] Zhou, Zhi Hua: *Ensemble learning*. Encyclopedia of biometrics, páginas 411–416, 2015. 9

[35] Kleinberg, EM: *Stochastic discrimination*. Annals of Mathematics and Artificial intelligence, 1(1-4):207–239, 1990. 9

[36] Yan, Xin e Xiaogang Su: *Linear regression analysis: theory and computing*. World Scientific, 2009. 9

[37] Bishop, Christopher M.: *Pattern recognition e machine learning*. springer, 2006. 9, 10, 11, 12

[38] Legendre, Adrien Marie: *Nouvelles méthodes pour la détermination des orbites des comètes*. F. Didot, 1805. 9

[39] Aggarwal, Charu C.: *Data classification: algorithms e applications*. CRC press, 2014. 10

[40] Kleinbaum, David G, K Dietz, M Gail, Mitchel Klein e Mitchell Klein: *Logistic regression*. Springer, 2002. 10

[41] Hosmer Jr, David W, Stanley Lemeshow e Rodney X Sturdivant: *Applied logistic regression*, volume 398. John Wiley & Sons, 2013. 10

[42] Hoffait, Anne-Sophie e Schyns, Michaël: *Early detection of university students with potential difficulties*. Decision Support Systems, 101:1–11, 2017. 10

[43] Haykin, Simon: *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1994. 11

[44] Hahnloser, Richard HR, Rahul Sarpeshkar, Misha A Mahowald, Rodney J Douglas e H Sebastian Seung: *Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit*. Nature, 405(6789):947, 2000. 11

[45] Piatetsky-Shapiro, Gregory: *Discovery, analysis, and presentation of strong rules*. Knowledge discovery in databases, páginas 229–238, 1991. 12

[46] Agrawal, Rakesh, Tomasz Imieliński e Arun Swami: *Mining association rules between sets of items in large databases*. Em *ACM sigmod record*, volume 22, páginas 207–216. ACM, 1993. 12

[47] Romero, Cristobal e Sebastian Ventura: *Educational data mining: A survey from 1995 to 2005*. Expert systems with applications, 33(1):135–146, 2007. 12

[48] Hand, David J: *Principles of data mining*. Encyclopedia of Environmetrics, 2, 2006. 12

[49] Shearer, Colin: *The crisp-dm model: the new blueprint for data mining*. Journal of data warehousing, 5(4):13–22, 2000. 13

[50] Cios, Krzysztof J, Roman W Swiniarski, Witold Pedrycz e Lukasz A Kurgan: *The knowledge discovery process*. Em *Data Mining*, páginas 9–24. Springer, 2007. 14

[51] Fayyad, Usama, Gregory Piatetsky-Shapiro e Padhraic Smyth: *From data mining to knowledge discovery in databases*. AI magazine, 17(3):37, 1996. 14

[52] Imbens, Guido W e Tony Lancaster: *Efficient estimation and stratified sampling*. Journal of Econometrics, 74(2):289–318, 1996. 22

[53] Hawkins, Douglas M.: *The problem of overfitting*. Journal of chemical information e computer sciences, 44(1):1–12, 2004. 23

[54] Kohavi, Ron: *A study of cross-validation e bootstrap for accuracy estimation e model selection*. Em *e International Joint Conference on Artificial Intelligence*, volume 14, páginas 1137–1145. Montreal, Canada, 1995. 23

[55] Arlot, Sylvain e Celisse, Alain: *A survey of cross-validation procedures for model selection*. Statistics surveys, 4:40–79, 2010. 23

[56] Claesen, Marc e Bart De Moor: *Hyperparameter search in machine learning*. Em *The XI Metaheuristics International Conference*, fevereiro 2015. 23

[57] Bergstra, James e Yoshua Bengio: *Random search for hyper-parameter optimization*. Journal of Machine Learning Research, 13(Feb):281–305, 2012. 23

[58] Powers, David Martin: *Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation*. Journal of Machine Learning Technologies, 2011. 26

[59] Zheng, Alice e Amanda Casari: *Feature engineering for machine learning: principles and techniques for data scientists*. O'Reilly Media, Inc., 2018. 27

[60] Ameri, Sattar e Fard, Mahtab J. e Chinnam Ratna B. e Reddy Chean K.: *Survival analysis based framework for early prediction of student dropouts*. Em *Proceedings of the 25th ACM International on Conference on Information e Knowledge Management*, páginas 903–912. ACM, 2016. 32

[61] Pal, Saurabh: *Mining educational data using classification to decrease dropout rate ofstudents*. Internation Journal of Multidisciplinary Sciences and Engineering, 3:35–39, maio 2012. 32

[62] Holanda, Maristela, Marília Dantas, Gustavo Couto, Jan Mendonça Correa, Aleteia Patrícia F de Araújo e Maria Emília T Walter: *Perfil das Alunas no Departamento de Computação da Universidade de Brasília*. Em *11º Women in Information Technology (WIT 2017)*, volume 11. SBC, 2017. 32

Apêndice A

Código usado para os experimentos

Listing A.1: Script Python usado para os experimentos

```
# import libraries
import semester_calculator as SC
import pandas as pd
import numpy as np
import ignore_warnings
from sklearn.linear_model import LogisticRegression
from sklearn.neural_network import MLPClassifier
from sklearn import tree
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import accuracy_score, recall_score
from sklearn.preprocessing import StandardScaler
from apyori import apriori

# CONFIGURATIONS
training_data_file_path = '../pibic_private/training_data.csv'
test_data_file_path = '../pibic_private/training_data.csv'

COURSE_CODES_WORKLOAD = {
  '114626': 4,
  '114634': 2,
  '114014': 6,
  '118001': 4,
  '118010': 2,
  '118028': 4,
  '118036': 2,
  '116301': 6,
  '113034': 6,
  '113042': 6,
  '113093': 4,
```

```
  '115045': 4,
  '168874': 6
}

def calculate(entry_semester, course_semester):
  if is_summer_course(course_semester):
    course_semester += 1

  diff = course_semester - entry_semester

  if entry_semester % 2 != 0 :
    if diff >= 10 :
      result = semesters_between_years(diff)
    else:
      result = diff
  else:
    if diff % 10 == 0 :
      result = semesters_between_years(diff)
    else:
      result = diff // 5

  return result + 1

def is_summer_course(semester):
  return semester % 10 == 0

def semesters_between_years(diff):
  return (diff // 5) + (diff % 5)

def remove_old_courses(dataframe):
  if '1_114014' in dataframe: \
  dataframe.loc[dataframe['1_114014'] != 0, '1_114626'] = dataframe['1_114014']
  if '1_114014' in dataframe: \
  dataframe.loc[dataframe['1_114014'] != 0, '1_114634'] = dataframe['1_114014']
  if '2_114014' in dataframe: \
  dataframe.loc[dataframe['2_114014'] != 0, '2_114626'] = dataframe['2_114014']
  if '2_114014' in dataframe: \
  dataframe.loc[dataframe['2_114014'] != 0, '2_114634'] = dataframe['2_114014']
  dataframe.drop(columns=['1_114014', '2_114014'])

  return dataframe

def transform_dataframe(dataframe, aggfunc, fill_value, groupby_columns):
  dataframe['CourseTerm'] = \
  dataframe.Semester.map(str) + '_' + dataframe.CodigoMateria.map(str)
```

```
  dataframe = dataframe.drop(
    columns=['SemestreIngresso', 'SemestreMateria', 'CodigoMateria', 'Semester']
  )

  dataframe = dataframe.pivot_table(
    values='Conceito',
    index=groupby_columns,
    columns='CourseTerm',
    aggfunc=aggfunc,
    fill_value=fill_value
  )
  dataframe.columns.name = None
  dataframe = dataframe.reset_index()
  remove_old_courses(dataframe)

  return dataframe

def failed_workload(row, semester=None):
  failed_workload = 0

  for code, workload in COURSE_CODES_WORKLOAD.items():
    if semester == None:
      failed_workload += row['1_' + code] * workload
      failed_workload += row['2_' + code] * workload
    else:
      failed_workload += row[semester + '_' + code] * workload

  return failed_workload

def calculate_semester(df):
  df['Semester'] = df.apply(
    lambda x: calculate(x['SemestreIngresso'], x['SemestreMateria']),
    axis=1
  )
  return df[(df.Semester > 0) & (df.Semester <= 2)]

def one_hot_encoding(df, groupby_columns):
  columns = df.columns.difference(groupby_columns).tolist()
  for column in columns:
      one_hot = pd.get_dummies(df[column])
      df = df.drop(column,axis = 1)
      one_hot.columns = map(lambda x: column + '_' + x, one_hot.columns)
      df = df.join(one_hot)
  return df
```

```
def calculate_failed_workload(df, aux_df, groupby_columns):
  aux_df['Conceito'] = aux_df['Conceito'].replace(['SR', 'II', 'MI'], 1)
  aux_df['Conceito'] = aux_df['Conceito'].replace(
    ['SS', 'MS', 'MM', 'CC', 'DP', 'TR', 'TJ'], 0
  )
  aux_df = transform_dataframe(aux_df, 'sum', 0, groupby_columns)

  df['Creditos_Reprovados'] = aux_df.apply(
    lambda row: failed_workload(row),
    axis=1
  )
  df['Creditos_Reprovados_1'] = aux_df.apply(
    lambda row: failed_workload(row, '1'),
    axis=1
  )
  df['Creditos_Reprovados_2'] = aux_df.apply(
    lambda row: failed_workload(row, '2'),
    axis=1
  )

  return df

def model_data(df, groupby_columns=['IdAluno', 'StatusFinal']):
  aux_df = df.copy()
  df = transform_dataframe(df, 'last', 'NC', groupby_columns)
  df = one_hot_encoding(df, groupby_columns)
  df = calculate_failed_workload(df, aux_df, groupby_columns)

  return df


print('Classification input creation start')
train_df = pd.read_csv(training_data_file_path)

# calculate semester
train_df = calculate_semester(train_df)
df = model_data(train_df)
print('Classification input creation success')

#
# Fourth model: model for apriori rule association with grades
#

print('Association input creation start')
assoc_df = train_df.copy()
```

```
assoc_df = assoc_df.drop(columns=['SemestreIngresso', 'SemestreMateria'])
assoc_df = assoc_df.applymap(str)


assoc_df['TermCourseGrade'] = \
assoc_df.Semester + '_' + assoc_df.CodigoMateria + '_' + assoc_df.Conceito
assoc_df = assoc_df.drop(columns=['Conceito', 'CodigoMateria', 'Semester'])

grouped = assoc_df.groupby(['IdAluno', 'StatusFinal'])
assoc_df = grouped['TermCourseGrade'].apply(
  lambda x: pd.Series(x.values)
).unstack()
assoc_df = assoc_df.reset_index()

assoc_df = assoc_df.drop(columns=['IdAluno'])
assoc_df.to_pickle('association_rules_grades.pkl')
print('Association input creation success')

scaler = StandardScaler()

logreg_param_grid = {
  'solver': ['liblinear', 'lbfgs'],
  'C': np.logspace(-3, 5, 9),
  # 'penalty': ['l1', 'l2']
}

mlpc_param_grid = {
  'alpha': 10.0 ** -np.arange(1, 6)
}

dtree_param_grid = {
  'criterion': ['gini', 'entropy'],
  'splitter': ['best', 'random'],
  'min_samples_split': np.linspace(0.1, 1.0, 10, endpoint=True)
}

rf_param_grid = {
  'criterion': ['gini', 'entropy'],
  'n_estimators': range(10,200,20)
}

classifiers = [
  ('LogisticRegression', LogisticRegression(max_iter=300), logreg_param_grid),
  ('MLPClassifier', MLPClassifier(max_iter=100), mlpc_param_grid),
  ('DecisionTreeClassifier', tree.DecisionTreeClassifier(), dtree_param_grid),
```

```
  ('RandomForestClassifier', RandomForestClassifier(), rf_param_grid)
]

feature_cols = df.columns.difference(['StatusFinal', 'IdAluno'])
features = df.loc[:, feature_cols]
labels = df.StatusFinal.replace({'EVADIDO': 1, 'FORMADO': 0})
test_size = 0.30
X_train, X_test, y_train, y_test = train_test_split(
    features, labels, test_size=test_size, stratify=labels, random_state=42)

scaler.fit(X_train.values)
X_train = scaler.transform(X_train.values)
X_test = scaler.transform(X_test.values)

print('Dataset - number of columns:', feature_cols.size)
print('Dataset - number of rows:', len(features))
print('Training size:', len(X_train))
print('Test size:', len(X_test))
print('\n')

best_classifiers = []
for classifier in classifiers:
  print(classifier[0])
  grid_search = GridSearchCV(classifier[1], classifier[2], scoring='recall',
                             cv=10, return_train_score=True)
  grid_search.fit(X_train, y_train.values)

  y_pred = grid_search.predict(X_test)
  print('Best params for recall', grid_search.best_params_)
  print("recall = %0.4f" % recall_score(y_test.values, y_pred))
  print("accuracy = %0.4f" % accuracy_score(y_test.values, y_pred))
  best_classifiers.append(grid_search.best_estimator_)

# READ DATA FOR CURRENTLY ENROLLED STUDENTS
test_df = pd.read_csv(test_data_file_path)
test_df = calculate_semester(test_df)
test_df = model_data(test_df, ['IdAluno'])
remaining_columns = df.columns.difference(test_df.columns).tolist()
remaining_columns.remove('StatusFinal')
for column in remaining_columns:
  test_df[column] = 0

feature_cols = test_df.columns.difference(['IdAluno'])
features = test_df.loc[:, feature_cols]
```

```
predictions = []
for classifier in best_classifiers:
    predictions.append(classifier.predict(features))

results_df = pd.DataFrame(
  columns=[
    'IdAluno',
    'Regressao Logistica',
    'Redes Neurais',
    'Arvores de Decisao',
    'Florestas Aleatorias'
  ]
)
index = 0
for student in range(test_df.shape[0]):
  results_df.loc[index] = [test_df['IdAluno'][index]] + \
    ['EVADIDO' if predictions[0][index] else 'FORMADO'] + \
    ['EVADIDO' if predictions[1][index] else 'FORMADO'] + \
    ['EVADIDO' if predictions[2][index] else 'FORMADO'] + \
    ['EVADIDO' if predictions[3][index] else 'FORMADO']
  index += 1

print('CLASSIFICATIONS OF ACTIVE STUDENTS')
print(results_df)
results_df.to_csv('students_classifications.csv')

print('ASSOCIATION RULES: GRADES')
records = assoc_df.T.apply(lambda x: x.dropna().tolist()).tolist()
rules = apriori(records, min_support=0.03, min_confidence=0.9)
for rule in rules:
  if 'EVADIDO' in tuple(rule.items):
    for observation in rule.ordered_statistics:
      if 'EVADIDO' in tuple(observation.items_add):
        print(rule)
```