DANIELA DE SOUZA GOMES

SISTEMA DE APRENDIZADO DE MÁQUINA PARA APOIAR DECISÕES EM PLANOS DE ESTUDO

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para obtenção do título de *Magister Scientiae*.

Orientador: Giovanni Ventorim Comarela

VIÇOSA - MINAS GERAIS
2021

Ficha catalográfica elaborada pela Biblioteca Central da Universidade Federal de Viçosa - Campus Viçosa

T

G633s
2021

Gomes, Daniela de Souza, 1995-
Sistema de aprendizado de máquina para apoiar decisões em planos de estudo / Daniela de Souza Gomes. - Viçosa, MG, 2021.
1 dissertação eletrônica (101 f.): il. (algumas color.).

Inclui apêndice.
Orientador: Giovanni Ventorim Comarela.
Dissertação (mestrado) - Universidade Federal de Viçosa, Departamento de Informática, 2021.
Referências bibliográficas: f. 93-96.
DOI: https://doi.org/10.47328/ufvbbt.2021.049
Modo de acesso: World Wide Web.

1. Mineração de dados (Computação). 2. Redes neurais (Computação). 3. Aprendizado do computador. 4. Sistema de reconhecimento de padrões. 5. Inteligência artificial - Aplicações educacionais. I. Comarela, Giovanni Ventorim. II. Universidade Federal de Viçosa. Departamento de Informática. Programa de Pós-Graduação em Ciência da Computação. III. Título.

CDD 22. ed. 006.312

Bibliotecário(a) responsável: Renata de Fátima Alves CRB6/2578

DANIELA DE SOUZA GOMES

SISTEMA DE APRENDIZADO DE MÁQUINA PARA APOIAR DECISÕES EM PLANOS DE ESTUDO

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para obtenção do título de *Magister Scientiae*.

APROVADA: 7 de outubro de 2021.

Assentimento:

Daniela de Souza Gomes

Daniela de Souza Gomes
Autora

Giovanni Ventorim Comarela

Giovanni Ventorim Comarela
Orientador

Dedico este trabalho a todos que acreditaram em mim e me deram forças para chegar até aqui.

Agradecimentos

A Deus por ter me concedido a oportunidade de conduzir meus estudos em uma universidade como a UFV.

À minha família por todo o apoio e em especial, a minha mãe Eneida e minha tia Elisângela que contribuíram com revisões do texto.

Ao meu irmão Leonardo que disponibilizou um ambiente para facilitar o desenvolvimento de parte do trabalho.

À minha irmã Isabela pela confiança e múltiplas contribuições, com ideias, discussões e revisões do trabalho. Além de compartilhar comigo materiais que me ajudaram a aprimorar minha competência em escrita.

Ao meu namorado Carlos por também contribuir com ideias, revisões e, muitas vezes, me motivar a seguir em frente.

Aos meus amigos por toda a força e suporte que me deram.

À UFV, por proporcionar tantas experiências boas, conhecimento e desenvolvimento pessoal.

Aos professores do DPI e a todos os membros do programa de pós-graduação pelos ensinamentos, dicas e orientações.

Ao meu orientador e coorientador por terem me guiado neste trabalho e por estarem sempre dispostos a ajudar.

Ao Gabriel pela grande colaboração. Suas ideias e contribuições foram de grande valia para este trabalho.

À CAPES pelo investimento. O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001.

"É um erro capital teorizar antes de se ter dados. Insensivelmente, começa-se a distorcer os fatos para se adequar às teorias, ao invés de conjecturar teorias para se adequar aos fatos."

(Sherlock Holmes - Sir Arthur Conan Doyle)

Resumo

GOMES, Daniela de Souza, M.Sc., Universidade Federal de Viçosa, outubro de 2021. **Sistema de aprendizado de máquina para apoiar decisões em planos de estudo**. Orientador: Giovanni Ventorim Comarela.

O elevado índice de reprovações é um problema recorrente enfrentado por diversas instituições de ensino superior. Nas universidades públicas, essa questão impacta, principalmente, em termos de logística e alocação de recursos físicos e financeiros, visto que muitos estudantes atrasam suas colações de grau, gerando uma lotação além da prevista de pessoas nos cursos. Conhecendo os riscos e motivações de uma reprovação, é possível traçar um plano de ação para mitigar que tal evento ocorra. Uma forma de avaliar isso é por meio de dados históricos escolares, os quais dão indícios se o estudante terá sucesso ou não em uma disciplina ou revelam padrões que se repetem ao longo da trajetória acadêmica do indivíduo. Neste trabalho, buscou-se explorar dados históricos da Universidade Federal de Viçosa, utilizando mineração de dados, para construir um sistema inteligente capaz de fornecer insumos que auxilia na tomada de decisões de estudantes e professores, no momento de elaboração do plano de estudos para um dado semestre letivo de graduação, a fim de evitar reprovações. Provendo informações que caracterizam os possíveis cenários que o estudante pode encontrar, o sistema fornece um apoio na escolha por caminhos que visam minimizar as chances de insucesso nas disciplinas cursadas. Amparando cada estudante individualmente, com informações direcionadas ao perfil deste, o resultado que se espera é uma queda nos índices de reprovação da universidade. Por fim, vale destacar que a metodologia proposta aqui é um processo replicável para outras instituições de ensino, considerando as semelhanças adotadas nos processos avaliativos do ensino superior.

Palavras-chave: Ciência de Dados. Mineração de Dados Educacionais. Mineração de Padrões Frequentes. Regras de Associação. Rede Neural. Caminho Crítico.

Abstract

GOMES, Daniela de Souza, M.Sc., Universidade Federal de Viçosa, October, 2021. **Machine Learning system to support decisions in study plans**. Advisor: Giovanni Ventorim Comarela.

The high failure rate is a recurrent problem faced by several higher education institutions. In public universities, this issue has an impact, mainly in logistics and allocation of physical and financial resources, as many students delay their graduation, generating a capacity beyond the expected number of people in the courses. Once known the risks and motivations of a failure, it is possible to draw up an action plan to mitigate such an event from occurring. One way to assess this is through historical scholar data. It provides clues as to whether the student will be successful or not in a subject or reveal repeated patterns throughout the individual's academic trajectory. In this work, we sought to explore historical data from the Federal University of Viçosa, using data mining, to build an intelligent system capable of providing inputs that help students and teachers make decisions when preparing the study plan for a given undergraduate term to avoid failures. Providing information that characterizes the possible scenarios that the student may encounter, the system provides support in choosing paths that aim to minimize the chances of failure in the subjects taken. By supporting each student individually, the expected result is a drop in the university's failure rates. Finally, it is worth noting that the methodology proposed here is a replicable process for other educational institutions, considering the similarities adopted in the evaluation processes of higher education.

Keywords: Data Science. Educational Data Mining. Frequent Pattern Mining. Association Rules. Neural Network. Critical Path.

Lista de Figuras

Sumário

Capítulo 1

Introdução

O alto índice de reprovação dentre estudantes preocupa as instituições de ensino superior, que têm como propósito formar profissionais e garantir a qualidade do ensino no país. Porém, com elevadas taxas de evasão e reprovação, surgem alguns questionamentos acerca de sua efetividade: será que o ensino perdeu a qualidade ou são os alunos que não estão interessados ou chegando despreparados na universidade? Ou ainda, será que a tradicional forma de ensinar é eficaz atualmente? São muitas perguntas e hipóteses que podem ser levantadas sobre o tema. Uma forma de responder essas e outras questões é através do levantamento de perfil acadêmico dos estudantes em universidades.

Dados são gerados a todo momento pelos sistemas de gerenciamento dessas instituições, o que pode ser muito relevante para extrair informação útil no contexto acadêmico. Dado que as instituições conhecem melhor o perfil de seus estudantes, elas conseguem adaptar as políticas de ensino no intuito de tornar o ambiente de aprendizagem cada vez mais agradável e efetivo, além de formar mais profissionais capacitados para o mercado de trabalho.

O problema das reprovações não é algo novo enfrentado pelas instituições de ensino. Muitos estudos buscam encontrar os motivos por trás desse número elevado, de modo a alterar essa realidade, através de mudança nas políticas das escolas ou mesmo com mais investimento na educação. Neste trabalho, buscamos investigar esse problema em uma universidade pública, usando aprendizado de máquina. Dados advindos de históricos escolares podem dizer muito sobre o perfil dos estudantes e da própria instituição.

De acordo com Athias (2019), o IBGE realizou uma análise da gestão de municípios do Brasil avaliando sua qualidade através de um índice. O estudo evidenciou que houve um aumento no índice de gestão municipal entre os anos de 2001 e 2014 devido a maior escolarização da população. Mostrando assim, que o investimento na educação traz retorno em diversos âmbitos para um país.

Dois indicadores que tentam mensurar o desempenho no sistema educacional são os índices de evasão e a reprovação nas escolas, números esses que podem estar intrinsecamente relacionados. O Brasil ocupava a terceira posição entre os países com maiores taxas de abandono escolar dentre os 100 com maior IDH em 2017 [Filho and Araújo (2017)]. Os autores apontam, como um dos motivos, o número sucessivo de reprovações, o que pode acarretar um sentimento de frustração ou incapacidade no estudante.

O número de concluintes em universidades públicas caiu 3,1% de 2018 a 2019, se-

gundo notas estatísticas do Censo de Educação Superior de 2019 do Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (INEP, INEP (2019)). E, conforme o relatório do Programa Internacional de Avaliação de Estudantes (PISA), em 2018, 1 a cada 10 alunos advindos de escolas menos favorecidas não esperam concluir o ensino superior (OCDE (2019)).

O estudo de Souza et al. (2017) sobre evasão no ensino superior no Brasil, descreveu fatores que aparecem frequentemente como possíveis causas do abandono nas Instituições de Ensino Superior (IES). Dentre eles estão dificuldades financeiras, influência familiar e, novamente, a reprovação em disciplinas. Diogo et al. (2016) aponta como determinantes da reprovação fatores externos aos cursos. Já Fornari (2010) argumenta que o sistema capitalista possui grande influência, mesmo que indireta, nesse cenário.

No presente trabalho, investigamos a situação em uma universidade brasileira, a Universidade Federal de Viçosa (UFV), um ambiente com a possibilidade de adquirir, facilmente, dados para análise desse problema. A metodologia proposta aqui pode ser naturalmente replicada para outra universidade brasileira já que essas possuem uma estrutura semelhante no que tange a divisão da grade curricular em períodos letivos com a obrigatoriedade de cumprimento de uma certa carga horária, que pode ser contabilizada tanto em horas quanto em créditos. Da mesma forma, o sistema educacional brasileiro, como um todo, possui diretrizes em comum incorporadas por várias escolas e o método aqui proposto é adaptável e expansível para esses cenários também.

As áreas de ensino e pesquisa na UFV são divididas em quatro Centros de Ciências: Humanas, Agrárias, Biológicas e Exatas. Cada centro é composto por um conjunto de departamentos e esses são os responsáveis pelo fornecimento dos cursos de graduação e programas de pós-graduação.

Geralmente, a duração de um curso de graduação é de 4 ou 5 anos, aproximadamente. Segundo o panorama do IBGE sobre indicadores sociais (Athias (2019)), uma métrica que monitora a adequação da idade com o nível de ensino do estudante é a taxa ajustada de frequência escolar líquida, o que pode dizer muito sobre o atraso escolar. Nesse relatório foi constatado que, em 2017, dentre os jovens com idade entre 18 a 24 anos, apenas 23,2% frequentam o ensino superior no tempo estimado. Isso é reflexo direto do atraso escolar nos cursos de graduação.

A grade curricular possui uma carga horária a ser cumprida, a qual pode ser medida em horas, bem como em créditos, como ocorre na UFV. Por exemplo, 4 créditos correspondem a 60 horas/aula. Além disso, em alguns cursos existem pré-requisitos adicionais para colar grau, tais como estágio. Em grande parte dos cursos, ao final, o aluno deve elaborar uma monografia ou trabalho de conclusão de curso (TCC), aplicando conhecimentos adquiridos ao longo de sua jornada.

O ano letivo na universidade é dividida em períodos, ou seja, os alunos têm em média de 4 a 5 meses para cumprirem as disciplinas nas quais se matricularam. Em todo fim de período na UFV, ocorre o chamado plano de estudos, em que o aluno, com acompanhamento ou não de um professor, escolhe as disciplinas que deseja cursar no próximo período. Para tomar essa decisão, são considerados alguns aspectos, tais como a grade curricular sugerida pela comissão coordenadora do curso, as disciplinas obrigatórias, tempo de curso, tempo para se formar e disciplinas que devem ser cursadas novamente.

O aluno pode fazer seu plano de estudos sozinho se concluir todas as disciplinas

programadas pela comissão até o 3º período. Caso contrário, precisa da supervisão do professor. Muitas vezes, mesmo o estudante que cumpriu os requisitos, sente a necessidade da orientação do professor, mais experiente por já ter presenciado várias situações de alunos distintas.

Uma preocupação recorrente durante o planejamento é o risco de reprovação. Cada reprovação que o aluno obtém representa uma vaga a menos no conjunto de disciplinas para o próximo período, o que implica, consequentemente, no atraso da formatura. O estudante que reprova muito também se sente desmotivado e exausto psicologicamente. Como dito, um grande número de reprovações pode ocasionar evasões.

Para a universidade, também existem impactos relevantes, como a falta de infraestrutura para atender a todos os estudantes, problemas de logística para alocação de recursos, ambientes superlotados (como sala de aula e restaurante universitário), entre outros. Como o número de vagas anuais não vem sofrendo alteração e mais pessoas reprovam ao longo dos anos, tais problemas se intensificam e se tornam mais aparente.

Foram fornecidos dados do sistema administrativo de históricos escolares e, por meio de métodos de aprendizado de máquina, queremos ter clareza sobre essas questões e entender os motivos das reprovações. Estudar padrões que levam a reprovação é importante para tentar mitigar esse problema da dúvida. Com mais informação disponível no momento do plano, o aluno e o orientador conseguem tomar melhores decisões, visto que possuem mais informações para discussão e reflexão.

Sendo assim, neste trabalho visamos construir um protótipo de um sistema de informação, com alertas e dados relacionados à situação acadêmica do estudante ao fim de um período. A incorporação desse protótipo no sistema administrativo, pode levar estudantes e orientadores, em um curto ou médio prazo, a tomarem decisões mais cautelosas e apropriadas o que, o que a longo prazo, reduziria o índice de reprovações na universidade.

O protótipo é formado por módulos que incorporam diversas técnicas da Ciência da Computação e o objetivo principal deste trabalho é sugerir uma metodologia de avaliação de modelos de aprendizado de máquina. Como objetivos secundários, visamos propor um *pipeline* para a construção de sistemas que englobam técnicas de vários campos da Computação, viabilizando a inclusão de todas elas em um produto só. Além disso, iremos propor duas novas métricas de avaliação de modelos.

O Capítulo 2 traz alguns trabalhos relacionados ao tema, os pontos distintos ou como cada um deles se assemelha a este trabalho. Já no Capítulo 3 serão definidos os conceitos primordiais para o entendimento da metodologia empregada, que será abordada no Capítulo 4. Os resultados alcançados serão mostrados e discutidos no Capítulo 5. No Capítulo 6, mostramos como tudo se conecta de modo a obtermos um produto final: um protótipo funcional. Por fim, o Capítulo 7 traz as principais conclusões levantadas pelos autores.

Capítulo 2

Trabalhos Relacionados

Como explicitado no Capítulo 1, existe um grande interesse mundial na melhoria contínua da educação. Porém, para saber como melhorar, é necessário primeiramente investigar os problemas e o que já funciona bem, em um dado contexto. Assim, o campo educacional é um objeto de estudo muito relevante, onde podem ser aplicados métodos de *knowlegde discovery* ou mineração de dados (Zaki and Jr (2014)). Por ser um tema de grande interesse global, uma subárea de pesquisa surge nessa circunstância, a mineração de dados educacionais (Romero and Ventura (2007)). Estudos da área visam principalmente analisar metodologias de ensino, taxas de aprendizagem e desempenho escolar.

No primeiro eixo de aplicação, temos o trabalho de Zingaro and Oztok (2012) que avalia uma metodologia bastante incorporada no ensino remoto: os fóruns de discussão. Os autores ressaltam que para a efetividade de um sistema de Ensino a Distância (EAD) é crucial a interação entre professor e alunos, como também entre os próprios estudantes. Estimular discussões entre os alunos é um interessante artifício formador de opinião. Os autores utilizam modelos preditivos, um deles com regressão logística, em um sistema de aprendizado *online* para analisar a interação de usuários através de comentários. O objetivo é prever se os comentários são relevantes para a discussão do tópico e se receberão respostas ou não. Entende-se que um comentário é relevante se esse possui interações. Foram validadas algumas hipóteses intuitivas acerca do tema, porém também foram descobertos aspectos contraditórios. Isso mostra a importância da utilização de mineração de dados sobre um determinado assunto. Mesmo que pareça óbvio, é válido investigar, pois, os dados podem revelar o contrário. Assim como Zingaro and Oztok (2012), trabalharemos com modelos preditivos, mas com foco na predição de reprovações e não na interação em sistemas de aprendizagem. O contexto trabalhado pelos autores também é um pouco distinto do nosso, já que eles trataram de um tema EAD e o disposto aqui se adapta também ao ensino presencial.

Outra categoria de trabalho que segue essa linha, é o de Cui et al. (2019) que rastreia, a partir de *logs*, o conhecimento adquirido pelo usuário em um sistema de aprendizado gameficado. São aplicados métodos Bayesianos de modo a prover análises sobre o aprendizado do aluno à medida que ele vai utilizando o sistema. O autor concluiu que esses métodos obtiveram acurácia relevante na identificação do conhecimento adquirido. A intenção principal era identificar áreas de estudo nas quais é necessário depositar um esforço extra, ou seja, aquelas que precisam de mais atenção e os alunos possuem maiores dificuldades. No presente trabalho, também identificamos pontos de cautela, ou seja, disciplinas críticas, aquelas que estão obtendo altos

índices de reprovação na universidade.

Em se tratando de desempenho escolar, Baradwaj and Pal (2011) investigaram a reprovação de estudantes através de árvores de decisão. Com seu preditor, foi possível agrupar estudantes de acordo com seus resultados acadêmicos. De forma similar a este trabalho, a partir de dados acadêmicos extraídos do sistema de gerenciamento da universidade, os autores conseguiram prever o desempenho de alunos ao fim do semestre. Isso auxilia na identificação antecipada de possíveis evasões ou ainda fornece informação útil para que os professores deem atenção mais personalizada àqueles que precisam. Aqui também queremos prever a possibilidade de reprovação para um dado período, mas trazendo para o contexto de uma universidade brasileira.

Um estudo semelhante ao anteriormente citado é o de Pelaez et al. (2019) que tenta prever estudantes de risco com relação à reprovação ou evasão em três níveis: baixo, médio e alto risco. Para cumprir tal objetivo, os autores utilizaram técnicas de mineração de dados, também incorporadas aqui, em etapas exploratórias. São elas: análise de correlação, visualização de dados e métodos de agrupamento. Eles trabalharam, porém, com três semestres de dados acadêmicos de uma disciplina de psicologia e nosso intuito aqui é trabalhar com dados de qualquer disciplina da universidade.

Outra abordagem para tratar a reprovação foi trazida por Raji et al. (2017) que tratou o problema de forma visual. Foi construído um sistema de visualização de dados para análise sobre um conjunto de dados coletados por 16 anos que descrevem o progresso dos estudantes. O sistema permitiu questionar algumas políticas adotadas pela escola. Informações trazidas de forma visual se tornam mais intuitiva para os usuários finais. Dessa forma, almejamos representar visualmente a situação acadêmica de um estudante, no intuito de facilitar a mensagem a ser transmitida. Diferentemente de Raji et al. (2017) que construiu a visualização para a administração da instituição, queremos aqui mostrar essas informações de forma clara para o aluno e o professor no momento do plano de estudos.

Nesse sentido, Du et al. (2016) propôs um sistema de recomendação acadêmica para estudantes e coordenadores de curso, também com um foco mais visual e interativo. Aqui, não iremos trabalhar com sistema de recomendações, mas iremos incorporar, de forma clara e visual, alertas que representam situações que o estudante deve evitar e sugestões de disciplinas para o próximo semestre.

Os estudos citados anteriormente não se referem à literatura nacional, foram encontrados poucos trabalhos nesse âmbito sobre reprovação ao nível de graduação. O trabalho de Costa et al. (2019) é um *survey* que mostra alguns métodos utilizados por outros autores para prever alunos com risco de reprovação em cursos de Computação. Ele mapeia os métodos de identificação de estudantes que correm o risco de reprovarem em algumas disciplinas ou até evadirem seus cursos de graduação. Os autores ressaltam que durante o mapeamento, notou-se que é possível prever com precisão o risco de reprovação. Sabendo desse risco, consegue-se evitar reprovações, diminuindo também o risco de evasão. Neste trabalho, também queremos prever uma possível reprovação, porém a um nível mais geral, considerando qualquer curso de graduação oferecido pela universidade. A maioria dos estudos nacionais aborda este outro ponto de interesse para instituições de ensino: as taxas de evasão. Tendo essa informação, a escola pode direcionar melhor os investimentos, ou ainda, tentar resgatar o aluno antes mesmo que ele opte pela desistência. A previsão de abandonos foi estudada por Carvalho et al. (2019) e Saraiva et al. (2019).

Carvalho et al. (2019) visa detectar previamente a taxa de evasão em cursos pre-

senciais de graduação em Computação com o uso de Regressão logística, Florestas aleatórias e Máquinas de vetor de suporte (do inglês *Suporte Vector Machine* - SVM). Os três classificadores obtiveram resultados semelhantes conseguindo prever melhor a classe de formandos do que a de desistentes.

Já Saraiva et al. (2019) prevê a evasão em cursos técnicos de informática, sendo as principais estratégias *Naive Bayes*, *K-Nearest Neighbors* (KNN), Árvore de decisão, Florestas aleatórias, Redes neurais e SVM. Foi detectado que a evasão média por período superou 50% e o algoritmo SVM obteve os melhores resultados com uma taxa de acerto de 97,97%.

Avaliando os trabalhos, é possível notar a recorrência dos principais métodos preditivos. Dessa forma, estes foram caminhos explorados também no presente trabalho, porém com a finalidade de prever a reprovação em cursos de graduação, diferentemente do primeiro que aborda evasão e do segundo que trabalha no contexto de cursos técnicos. Para prover um melhor entendimento dos métodos utilizados, os principais conceitos serão explorados no Capítulo 3.

Capítulo 3

Referencial Teórico

Neste Capítulo serão abordados todos os conceitos técnicos que se fazem necessário para entender a metodologia utilizada. Na Seção 3.1, falaremos da Extração de Padrões Frequentes e Regras de Associação, um método muito conhecido e utilizado em mineração de dados para detecção de correlações entre itens que ocorrem com frequência. O resultado desse processamento pode ser usado para recomendação de itens ou mesmo o entendimento de padrões corriqueiros em um conjunto de dados. Uma vez tendo conhecimento desses padrões, é possível também chamar a atenção para aqueles que não são bons e devem ser evitados, como é o caso deste trabalho.

A Seção 3.2 traz os conceitos básicos de Redes Neurais, um modelo preditivo muito poderoso e com diversas aplicações. Sistemas de Redes Neurais podem ser utilizados para distinção de objetos em imagens e vídeos, como o reconhecimento facial ou detecção de placas de veículos, por exemplo. As redes mais simples são geralmente usadas para prever uma variável de um conjunto de dados, como é a aplicação empregada neste trabalho.

Por fim, na Seção 3.3, será abordado o conceito de Caminho Crítico que visa a otimização na escolha de um caminho dentre vários possíveis. Trata-se de um método muito empregado no gerenciamento de projetos visando principalmente reduzir o tempo de execução. No presente trabalho, utilizaremos o Caminho Crítico para determinar o menor tempo para um aluno se graduar dentre várias possibilidades.

3.1 Padrões frequentes e regras de associação

A mineração de padrões frequentes, como o próprio nome já diz, busca encontrar padrões que ocorrem com frequência em um conjunto de dados. Han et al. (2012) usa a Análise da cesta de compras (em inglês, *Market Basket Analysis* (MBA)) para introduzir os conceitos com um exemplo comum e bem corriqueiro. A MBA procura descobrir quais itens os clientes geralmente compram juntos. Sabendo dessas tendências de compras, os supermercados e lojas podem posicionar tais produtos de forma mais estratégica nas prateleiras de modo a incentivar a compra, aumentando assim os lucros da empresa. Dessa forma, a pergunta a ser respondida por esse problema é: quais produtos os clientes geralmente compram juntos?

Suponha que I seja o conjunto de produtos de um supermercado, por exemplo: arroz, pão, batata, feijão e entre outros. Assim, I é um conjunto finito de itens tal como: $I = \{I_1, I_2, ..., I_m\}$. A partir desses itens, é possível construir subconjuntos de tamanho k e agrupá-los no que chamamos *k-itemset* (Zaki and Jr (2014)), ou seja, é um

conjunto que contém todas as combinações de um número k dos itens pertencentes a I, como ilustra a Figura 3.1 [1].

K	K-itemsets
1	{A, B, C, D, E, F}
2	{AB, AC, AF, BC, BD, BE, CE, CF, DE, DF, EF}
3	{ABC, ABD, ACE, BDF, DEF}
4	{ABDE, BCDF, CDEF}
5	{ABCDE, BCDEF}
6	{ABCDEF}

Figura 3.1: Exemplo de *k-itemset* em uma *Market Basket Analysis*. Na prateleira, existem 6 tipos de produtos que podem ser combinados em conjuntos de 1 a 6 itens, como mostra a tabela.

Como pode-se notar, *1-itemsets* é o conjunto formado por itens que possuem apenas um elemento. *2-itemsets* são aqueles que possuem dois itens e assim por diante. Como a prateleira da Figura 3.1 possui seis categorias de produtos diferentes, o valor máximo que k pode atingir é 6, com o *6-itemset* ABCDEF, ou seja, o conjunto formado por todos os produtos da prateleira.

Seja T um conjunto de transações, sendo que uma transação representa um carrinho de compras, não vazio, descrito por um identificador único. Os itens que compõem essas transações estão contidos em I, já que um cliente somente pode colocar em seu carrinho, os produtos que estão na prateleira do supermercado.

Dessa forma, T é, então, um conjunto de tuplas da seguinte forma $T = (tid_1, t_1), (tid_2, t_2), ..., (tid_n, t_n)$, onde tid_i é o *id* correspondente à transação i, formada pelo conjunto de itens t_i. (Zaki and Jr (2014)) A Figura 3.2 mostra graficamente um exemplo desses conceitos.

[1] As Figuras 3.1 e 3.2 são de autoria própria, desenhadas com o auxílio da ferramenta Flaticon (https://www.flaticon.com/).

Identificador (tid)	Itens da transação (t)
1	ABDE
2	BCE
3	ABDE
4	ABCE
5	ABCDE
6	BCD

Figura 3.2: Exemplo de transações em uma *Market basket analysis*. Uma transação corresponde a um carrinho de compras e os produtos colocados nele.

A Figura 3.2 mostra seis carrinhos de compras com diferentes produtos da prateleira, o que correspondem às transações. Assim, o conjunto T, neste contexto, seria formado pelos seis carrinhos, cada um com seu identificador e conjunto de itens, como ilustrado na tabela. Por exemplo, a primeira transação possui o identificador 1 e os itens A, B, D e E.

Uma regra de associação é uma implicação da forma $X \implies Y$, em que X é um *k-itemset* chamado antecedente e Y é um *k-itemset* chamado consequente da regra. Os conjuntos X e Y podem ser de tamanhos diferentes.

Trazendo para o contexto da MBA, suponha a seguinte regra de associação: { café, pão } $\implies$ { manteiga }. Isso seria o equivalente a dizer que os clientes que geralmente compram café e pão, também levam manteiga, na mesma compra.

Para que $X \implies Y$ seja uma regra válida, X e Y são conjuntos não vazios contidos em I e sem interseção entre eles, ou seja, $X \subset I$, $Y \subset I$, $X \neq \emptyset$, $Y \neq \emptyset$ e $X \cap Y = \emptyset$. (Han et al. (2012))

A porcentagem de transações em T que contém os conjuntos X e Y, chamamos de suporte, ou seja, é a probabilidade de uma transação conter a união dos conjuntos X e Y (Han et al. (2012)), conforme mostra a Equação 3.1.

$$suporte(X \implies Y) = P(X \cup Y) \tag{3.1}$$

Um conjunto de itens é considerado frequente se esse supera um valor mínimo arbitrário de suporte definido a critério do problema a ser solucionado (Zaki and Jr (2014)). Por exemplo, se usarmos um suporte mínimo de 5 na Figura 3.2 queremos selecionar os itens que aparecem pelo menos 80% das vezes, ou seja, em cinco transações, no mínimo. Os itens frequentes nesse caso seriam apenas B, E e BE, visto que são os únicos que ocorrem pelo menos cinco vezes (cerca de 80%) dentre as transações. O valor de suporte pode ser tratado tanto como relativo ou absoluto, como no exemplo mostrado. Em algumas literaturas faz a distinção das duas métricas pelo uso do termo *suporte relativo*.

A confiança de uma regra representa a probabilidade condicional P de uma transação conter Y dado que ela contém X (Zaki and Jr (2014)), isto é, a fração de transações de T que contém X e também Y (Han et al. (2012)), como mostra a Equação 3.2.

$$conf(X \implies Y) = \frac{suporte(X \cup Y)}{suporte(X)} \tag{3.2}$$

Por exemplo, a $conf(B \implies E) = suporte(BE)/suporte(B) = 5/6 = 0,83$, isto é, em 83% das vezes em que B aparece, E também aparece. Os dois itens aparecem juntos em 83% das vezes, o que pode indicar uma correlação. Uma regra é dita forte se esta ultrapassa um valor de confiança mínimo estabelecido a rigor do usuário. (Zaki and Jr (2014))

O objetivo na mineração de padrões frequentes é encontrar os conjuntos de itens que mais aparecem, ou seja, aqueles que superam um valor mínimo de suporte definido e a partir deles, encontrar regras fortes as quais superam um valor mínimo de confiança estabelecido. O primeiro passo pode ser feito por meio da força bruta, enumerando-se cada subconjunto de itens combinados a partir de I, calculando os respectivos suportes e depois selecionando aqueles que ultrapassam o valor desejado.

Porém, existe um algoritmo proposto por Agrawal et al. (1993), chamado Apriori que busca encontrar de forma eficiente os conjuntos de itens frequentes. Ele trabalha com duas premissas para otimizar a geração dos itens no espaço de busca. Suponha um conjunto de itens $X \subseteq Y$, assim o suporte de X é maior ou igual ao de Y, pois Y pode conter elementos a mais que X que o fazem aparecer com menos frequência. Sendo assim, como descreve Zaki and Jr (2014), pode-se observar que:

1. Se X é frequente, qualquer subconjunto de Y contido em X também o é;

2. Se X não é frequente, qualquer conjunto que o contenha também não pode ser.

O algoritmo monta uma árvore de combinações de itens, onde, no final de sua execução, cada nível da árvore k armazenará os *k-itemsets* frequentes. Seu pseudocódigo é mostrado nos Algoritmos 1 e 2.

O primeiro passo do Apriori é gerar os conjuntos de 1 elemento só, ou seja, os *1-itemsets* e contar as ocorrências de cada conjunto nas transações, isto é, calcular os seus respectivos suportes (Agrawal and Srikant (1994)). Esses formarão o nível 1 da árvore. Nesse ponto, os suportes que forem superiores ao mínimo estabelecido se mantém no conjunto e os outros são podados de C_1.

Em cada passo subsequente, um novo nível da árvore será preenchido. Isso ocorre em duas etapas: a geração de candidatos e a poda. A primeira é mostrada no Algoritmo 2. A função recebe o conjunto de itens do nível anterior combinados para gerar os candidatos para o próximo nível, sem duplicatas, presentes nas transações. Um item X_{ab} no novo nível será formado pela união de dois nós irmãos X_a e X_b do nível anterior, com algumas ressalvas. Nesse ponto a premissa 1 é utilizada para reduzir a geração de nós não frequentes. Faz-se uma verificação para garantir que todos os subconjuntos de X_{ab} são frequentes, ou seja, estão contidos no nível anterior C_k. Caso algum não esteja, ele não é frequente e consequentemente X_{ab} também não por isso, ele não é gerado. Caso a folha X_a em questão não tenha irmãos, não é possível gerar mais combinações de filhos para X_a. Então, caso esse não possua filhos, ele é removido do conjunto bem como seus ancestrais sem filhos. Dessa forma, é obtido o conjunto de itens C_{k+1} que preencherá o novo nível da árvore.

Algoritmo 1: Apriori

Entrada: conjunto de transações D, suporte_minimo
Saída: conjunto itens frequentes F
C_1 = Conjuntos de 1 elemento
k = 1
enquanto $C_k \neq \emptyset$ **faça**
    Calcula suportes de C_k
    para cada *folha x de* C_k **faça**
        se $suporte(x) \geq suporte\_minimo$ **então**
            Adiciona x ao conjunto de itens frequentes (F).
        senão
            Remove x de C_k
        fim se
    fim para cada
    $C_{k+1} = Estende\_arvore(C_k)$
    $k = k + 1$
fim enqto
Retorna F

Algoritmo 2: Estende_arvore

Entrada: C_k
Saída: C_k
para cada *folha* X_a *de* C_k **faça**
    para cada *folha* X_b *irmã de* X_a **faça**
        $X_{ab} = X_a \cup X_b$
        se *Todo subconjunto j de* $X_{ab} \in C_k$ **então**
            Adiciona X_{ab} como filho de X_a
        fim se
    fim para cada
    se X_a *não tem filhos* **então**
        Remove X_a e seus ancestrais sem filhos de C_k
    fim se
fim para cada
/* Retorna C_k modificado par preencher o próximo nível. */
Retorna C_k

Assim, para cada passo k, são calculados os suportes dos itens de C_k. Aqueles que possuem suporte superior ao mínimo, são adicionados ao conjunto de itens frequentes e aqueles que não, são removidos de C_k. Esse passo é chamado poda e ele impede que a árvore cresça com a criação de itens inferiores ao suporte mínimo. É a aplicação direta da premissa 2.

Visualizando na árvore da Figura 3.3, se a combinação é filho de um nó podado na etapa anterior, este também será podado, como é o caso de CDE, uma combinação gerada a partir de um item não frequente CD. Para o restante, o suporte é calculado normalmente. Após ter o suporte calculado no nível k, podam-se aqueles nós os quais não superaram o valor mínimo e repete todo o processo para a construção do nível $k+1$.

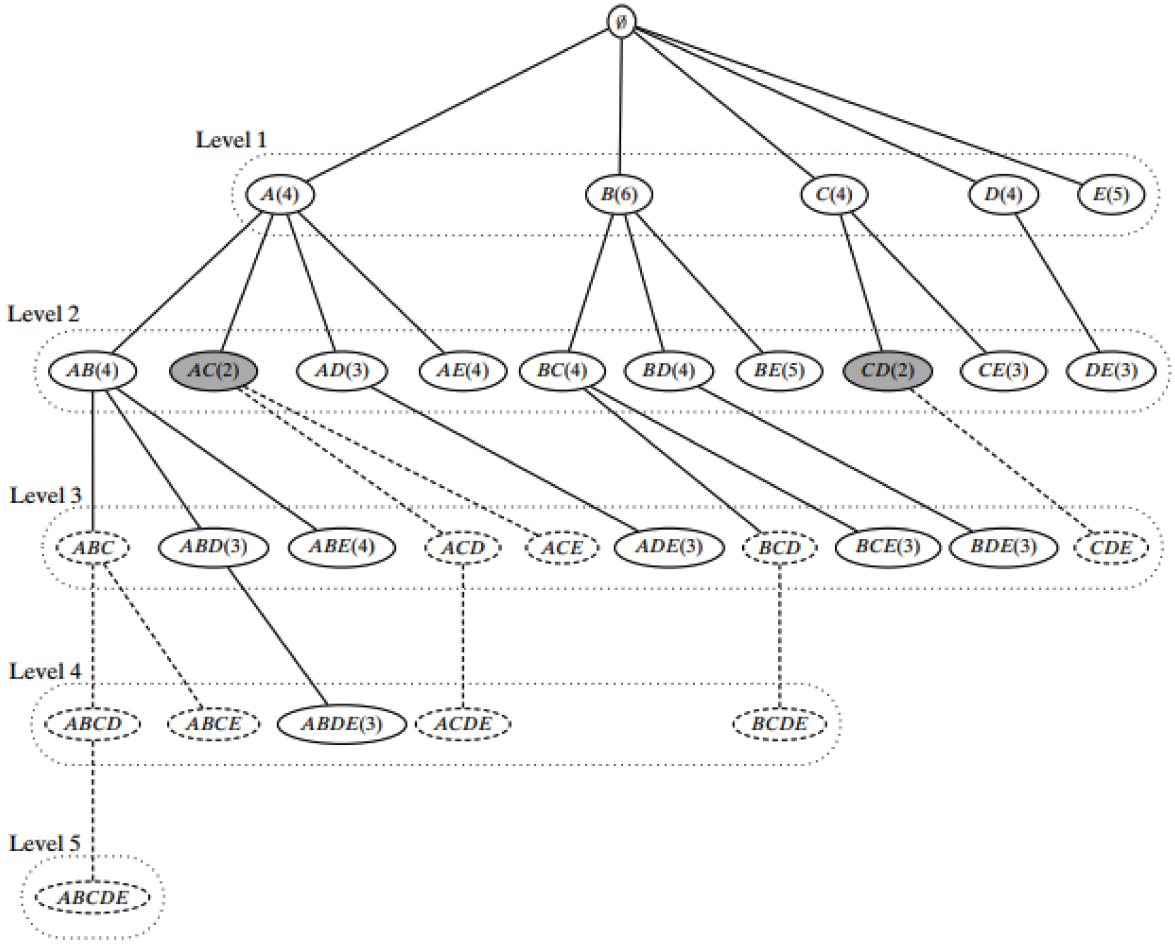

Figura 3.3: Árvore do algoritmo Apriori para o exemplo da *Market basket analysis*, retirado de Zaki and Jr (2014).

A Figura 3.3 mostra a árvore de itens frequentes gerada após a execução do algoritmo Apriori para um valor 3 de suporte. O primeiro nível é construído a partir dos *1-itemsets* e como todos eles aparecem mais de três vezes nas transações, nenhum deles é podado. Para montar o segundo nível, são geradas as combinações entre nós irmãos, assim A e B geram AB, e assim por diante. Uma vez geradas as combinações, calculam-se os suportes e poda-se os nós que não atingiram o valor mínimo, como AC e CD.

O algoritmo segue sua execução, estendendo a árvore e podando-a até que se

atinge o último nível, não vazio, aquele em que ainda é possível gerar combinações frequentes. O processo termina quando não há mais candidatos formados por uma combinação de k itens presentes nas transações.

Dada a lista de conjuntos frequentes, para gerar as regras de associação, basta combinar os itens de cada um no formato de regra já mencionado, uma implicação do tipo $X \implies Y$. Por fim, calculam-se as confianças de cada regra criada e seleciona-se apenas aquelas que ultrapassarem o limite indicado de confiança de modo a selecionar apenas as mais fortes.

As regras mais fortes são um indicativo de alta correlação e é uma forma de representar os padrões frequentes encontrados em um conjunto de dados.

Nesta Seção foram abordados os principais termos relacionados a extração de padrões frequentes e regras de associação. A Seção 3.2 introduz os conceitos principais referentes a redes neurais, outro método importante empregado neste trabalho.

3.2 Redes neurais

Redes neurais são modelos de aprendizado de máquina (do inglês *Machine Learning* - ML) que evoluíram a partir dos estudos em neurociência (Russell and Norvig (2009)). Uma Rede Neural Artificial (RNA) é um modelo preditivo composto por um conjunto de neurônios agrupados em camadas que se conectam. Elas são amplamente utilizadas para aproximação de uma função não linear (Sutton and Barto (2018)). A Figura 3.4 esquematiza um modelo de RNA, a qual chamamos de *Multilayer Perceptrons*, por ser formada por várias camadas de neurônios (*perceptrons*).

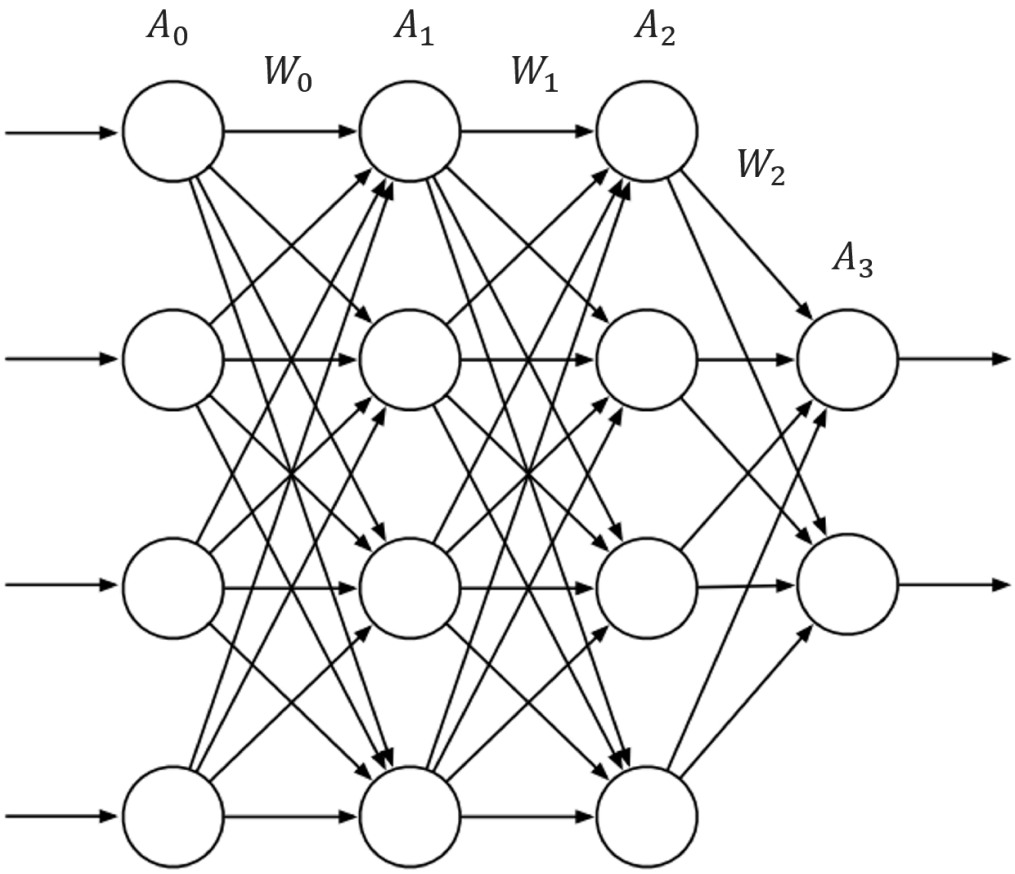

Figura 3.4: Esquema de rede neural, adaptado do livro de Sutton and Barto (2018)

No Esquema 3.4, A_i é o conjunto de neurônios de uma camada i da rede, enquanto W_i é o conjunto de pesos que ligam i a uma camada $i+1$. A primeira camada da rede (A_0) é a de entrada, por onde chegam os dados e a última é a de saída (A_3), que retorna o resultado da predição.

O restante é chamado de camadas internas ou camadas escondidas em tradução literal do inglês *hidden layers*. Elas têm esse nome porque os valores dos atributos ou *features* que passam por ali são criados no processo de treinamento da rede, ou seja, são desconhecidos para o usuário, ao contrário das camadas inicial e final (Goodfellow et al. (2016). Na camada inicial sabemos quais atributos compõem os dados de entrada e na final será obtido um valor numérico, caso o problema seja de regressão ou um valor binário, caso seja de classificação.

A quantidade de nós de cada camada define a largura do modelo de *machine learning* e o número de camadas internas define sua profundidade (Goodfellow et al. (2016)). Quanto mais camadas houver, mais profunda será a rede, o que caracteriza o conceito de redes neurais profundas ou *deep learning*.

De acordo com Russell and Norvig (2009), as propriedades de uma rede neural são determinadas por sua topologia e pelas características de seus neurônios. Os neurônios, na Figura 3.4 são representados pelos círculos, que formam cada unidade da rede. A Figura 3.5 mostra com mais detalhes o esquema de um neurônio.

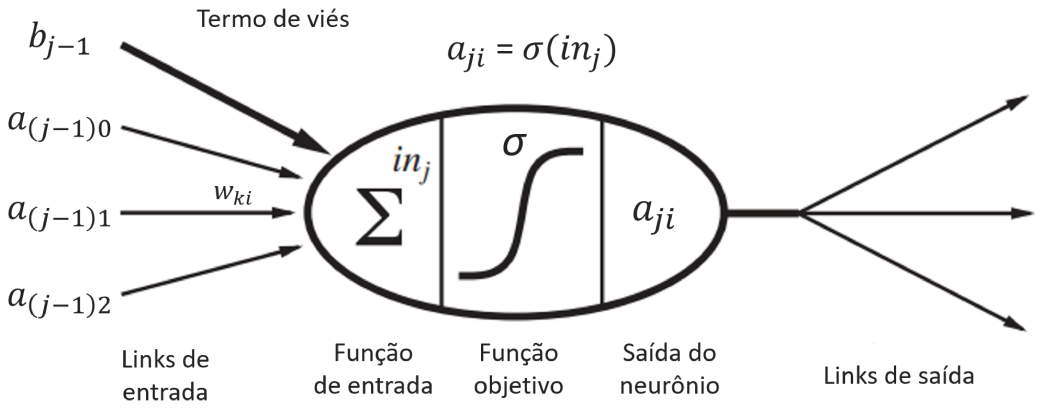

Figura 3.5: Esquema de neurônio, adaptado do livro de Russell and Norvig (2009)

Cada unidade realiza uma soma ponderada das entradas que se ligam a ela. Segundo Zou et al. (2008), tais pesos são uma analogia à força das sinapses no sistema nervoso. O termo de viés ou *bias* é uma constante que pode ser adicionada para ajustar a saída do neurônio, fazendo com que o modelo se adéque melhor aos dados. O resultado dessa operação é submetido a uma função não linear, chamada de função de ativação (Sutton and Barto (2018)). E assim, é obtido o valor de saída da unidade, como mostra a Equação 3.3 (Zou et al. (2008)).

$$a_{ji} = \sigma((\sum_{k=0}^{n} w_{ki} \times a_{(j-1)i}) + b_{j-1}) \tag{3.3}$$

Onde a_{ji} é a saída produzida pelo i-ésimo neurônio de j, σ é a função de ativação, $a_{(j-1)i}$ é a entrada que vem de cada neurônio i da camada anterior $j-1$, b_{j-1} é o termo de viés da camada anterior $j-1$ e w_{ki} são os respectivos pesos de cada conexão k com i, ou seja, cada ligação de um neurônio da camada anterior com a camada atual. Para simplificar, k e i são índices que iteram na largura da rede, ou nas linhas, se fizermos uma analogia a matrizes e j é o índice que itera sobre as camadas, ou colunas. Assim, cada camada j, é formada pelas somas ponderadas de cada neurônio da camada anterior. w_{ki} é então, o valor das arestas entre uma camada e outra.

Os neurônios de uma rede podem ser conectados de diversas formas. Neste trabalho iremos utilizar uma rede *feed-forward* que possui conexões apenas em uma direção, o que, no fim, forma um grafo acíclico Russell and Norvig (2009). A Figura 3.4 é um exemplo disso.

Para simplificar o entendimento, vamos pensar na rede como uma implementação de cálculos vetoriais, ao invés de escalares. Assim, a saída de uma camada j seria um vetor A_j, como mostrado na Equação 3.4.

$$A_j = \sigma((W_{j-1} \times A_{j-1}) + b_{j-1}) \tag{3.4}$$

Na Equação 3.4, W_j representa o vetor de pesos que conecta a camada $j-1$ à camada j e $(W_j \times A_{j-1})$ é um produto vetorial que resulta no vetor da próxima camada e b_{j-1} continua sendo o *bias* da camada anterior. Assim, a última camada z, tem como resultado o vetor de saída A_z, como na Equação 3.5.

$$A_z = \sigma((W_j \times A_j) + b_j) \tag{3.5}$$

Suponha uma rede de 4 camadas, como a da Figura 3.4. A saída dessa RNA seria dada por:

$$A_3 = W2 \times \sigma(W1 \times \sigma(W0 \times A0 + b0) + b1) + b2) \tag{3.6}$$

Como mostrado nas equações anteriores, as somas dos pesos multiplicados pelas entradas passam por uma função de ativação σ, o que garante a possibilidade da descoberta de aproximações não lineares.

Analisando a Equação 3.6, $W0 \times A0 + b0$ é uma combinação linear. Se $A1$ fosse somente igual ao resultado da camada anterior: $W0 \times A0 + b0$, teríamos na próxima camada, outra combinação linear $W1 \times A1 + b1$. E assim ocorreria sucessivamente ao longo das camadas, já que uma combinação de combinações lineares, é outra combinação linear. Assim, para que essa propriedade de linearidade seja descartada ao longo das camadas e possam ser descobertas aproximações não lineares, usa-se a função de ativação. Sendo assim, o termo $A1$, na verdade, corresponde ao resultado da camada anterior passado pela função de ativação $\sigma(W0 \times A0 + b0)$ e assim, por diante.

De acordo com Sutton and Barto (2018), diversas funções podem ser empregadas nessa aplicação, porém uma das mais utilizadas é a função logística, também chamada de sigmoide ou *Sigmoid*, da Equação 3.7. O gráfico da função pode ser visualizado na imagem 3.6 [2].

$$Sigmoid : f(x) = \frac{1}{1 + e^{-x}} \tag{3.7}$$

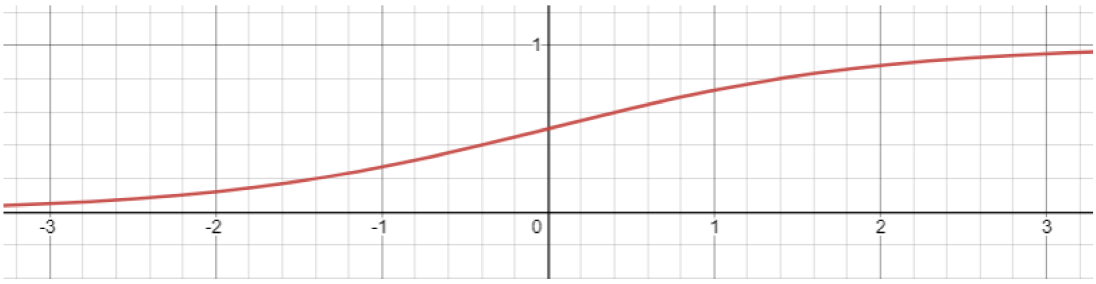

Figura 3.6: Gráfico da função *Sigmoid*.

A função *Sigmoid* gera como resultado um valor entre 0 e 1, normalizando os valores de entrada e também retira a linearidade (Goodfellow et al. (2016)). Outra função comumente usada é a função de ativação linear retificada (ReLU) (Sutton and Barto (2018)), mostrada na Equação 3.8 e no gráfico da Figura 3.7.

$$ReLU : f(x) = max(0, x) \tag{3.8}$$

[2]Os gráficos mostrados nesta Seção foram desenhados usando a ferramenta Desmos, que pode ser acessada em https://www.desmos.com/.

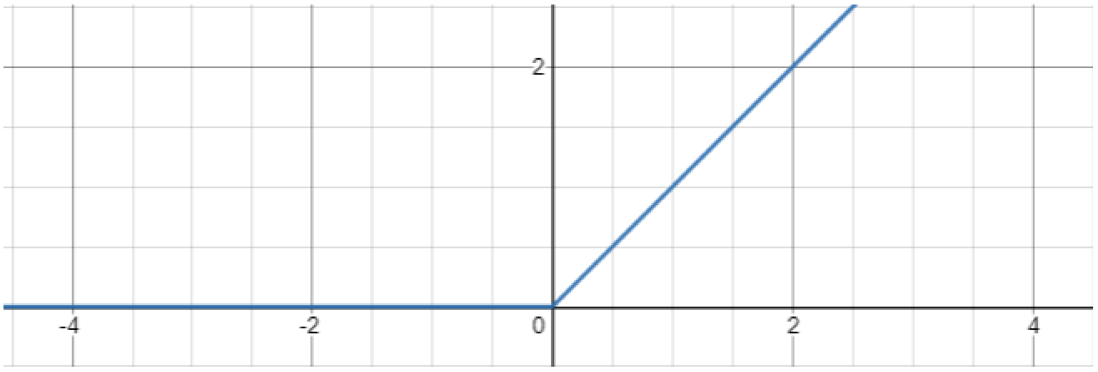

Figura 3.7: Gráfico da função ReLU.

Como pode-se notar pelo gráfico, ela retém apenas os valores positivos e leva todos os negativos para 0 (Zhang et al. (2017)). Assim, ela desativa alguns neurônios, o que pode até tornar a computação menos custosa (Academy (2021)). Entretanto, em algumas situações, essa desativação pode ser um problema. Um gradiente que flui através de um neurônio desativado terá o valor de zero a partir daquele ponto em diante. Se o gradiente é zero, então os pesos não serão mais atualizados e poderá haver neurônios na rede que nunca produzirão valor durante todo o conjunto de treinamento (CS (2021)). O conceito de gradiente será melhor explorado mais a frente, nesta Seção.

Variações da ReLU tentam minimizar o problema da desativação de neurônios, como a *Leaky ReLU* que inclina a reta para valores de $x < 0$ com a adição de um componente linear α (Academy (2021)). A função *Leaky ReLU* é apresentada na Equação 3.9, onde α pode variar entre 0 e 1 (Zhang et al. (2017)). Um esboço do gráfico para $\alpha = 0,1$ é ilustrado na Figura 3.8.

$$LeakyReLU : f(x) = max(\alpha x, x) \tag{3.9}$$

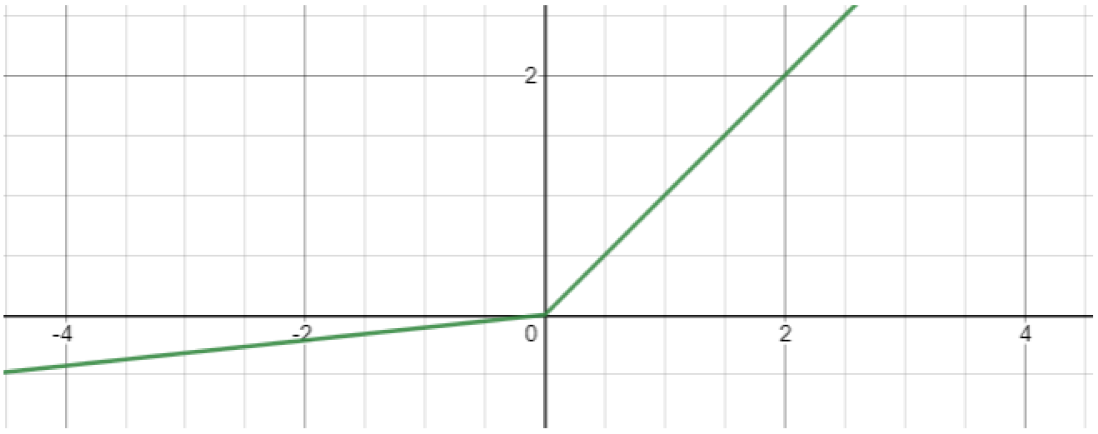

Figura 3.8: Gráfico da função *Leaky ReLU*.

Uma vez definidos os hiperparâmetros da rede, tal como a função de ativação e sua topologia, inicia-se o processo de treinamento, onde são realizados os cálculos para cada neurônio, mostrado na Equação 3.3, até que se alcance a última camada, onde teremos um resultado de saída da rede para uma instância de treinamento. Essa sequência de cálculos é chamada de propagação e, após a passagem de todas as

instâncias do conjunto de treinamento por esse processo, os pesos da rede, bem como seus termos de viés, podem ser ajustados conforme o algoritmo de retropropagação ou *back-propagation*.

Depois que todo o conjunto de treinamento passa pela rede uma vez, tanto no processo de propagação quanto de retropropagação, finaliza-se o que chamamos de uma época de treinamento (Baeldung (2020)). A época é a medida que sinaliza o fechamento de um ciclo de passagem pela rede.

De acordo com Sutton and Barto (2018), o método de *back-propagation* é promissor em redes neurais rasas, com poucas camadas, como é o nosso caso. Vale destacar que este não é o único algoritmo existente para a atualização de pesos. Outra estratégia empregada nessa tarefa é a aprendizagem por reforço que pode ser utilizada, principalmente, para treinar redes profundas. Contudo, este é um tema que está fora do escopo deste trabalho. Aqui, utilizaremos o método de retropropagação.

O objetivo da retropropagação é computar os gradientes em uma rede neural (Goodfellow et al. (2016)). Um algoritmo muito usado para cumprir essa tarefa é o Gradiente Descendente (GD) ou Descida do Gradiente que visa otimizar uma função qualquer, encontrando seu valor mínimo (Academy (2021)). Desse modo, queremos atualizar os pesos e vieses de uma RNA com base nos erros de predição obtidos, de forma a minimizá-los. Para isso, dado um vetor de predição A_z e sendo Y, o vetor de valores real, ou *target*, calcula-se o erro de predição, com uma função de custo, do inglês *Loss function*. Um exemplo de função de custo é mostrado na Equação 3.10.

$$E = |A_z - Y|^2 \tag{3.10}$$

Como já ressaltado, a finalidade aqui é minimizá-la, porque assim, estaremos dizendo que os erros de predição são tão baixos de modo a garantir que estamos nos aproximando do valor real. Um modelo será considerado bom, quanto mais próximo os valores preditos por ele, ficarem do *target*.

Como o resultado da função de custo da Equação 3.10 é sempre positivo e queremos minimizá-la, o menor valor possível que podemos obter é zero, o que significa que o valor predito alcançou exatamente o real (Ribeiro et al. (2020)).

Na retropropagação, o processo ocorre na ordem inversa, partindo da última camada em direção à primeira, corrigindo, a cada passo, os pesos e vieses com base no erro calculado (Cerqueira et al. (2001)). Essa propagação usa a regra da cadeia para ajustar os termos passados, já que, ao longo da rede, eles foram descobertos com base na camada anterior por isso, são variáveis dependentes dela.

Sem entrar no mérito da prova matemática, como vimos, os neurônios de uma camada A_j são calculados sobre valores de W_{j-1} e b_{j-1} da camada anterior. E como observado, a função de custo é calculada com base na camada de saída A_z, que depende de uma camada anterior A_j. Dessa forma, derivando parcialmente a função de custo em relação aos seus coeficientes (pesos e vieses), obtêm-se os valores que serão utilizados para a correção de coeficientes da camada anterior A_{j-1}. Duas ou mais derivadas da mesma função, são expressas por um gradiente (∇). Para simplificar, suponha que queremos descobrir uma reta que se ajusta aos três pontos da Figura 3.9.

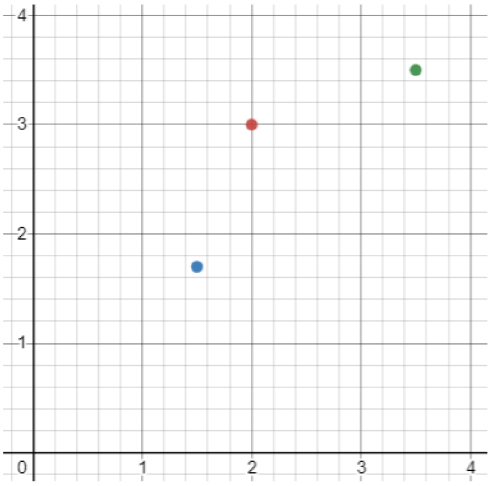

Figura 3.9: Três pontos no plano

Queremos encontrar então, uma função de 1º grau $Y = a \times X + b$, que possui apenas dois coeficientes a e b. Os erros aqui seriam calculados pela diferença dos valores de Y fornecidos pela reta com as reais coordenadas y de cada ponto. O propósito é descobrir os valores de a e b que minimizem esses erros, ou seja, ajuste bem a reta aos dados. Fazemos isso através da otimização da função de erro. Para o conjunto de pontos da Figura 3.9 podemos traçar diferentes tipos de reta, tais como as da Figura 3.10.

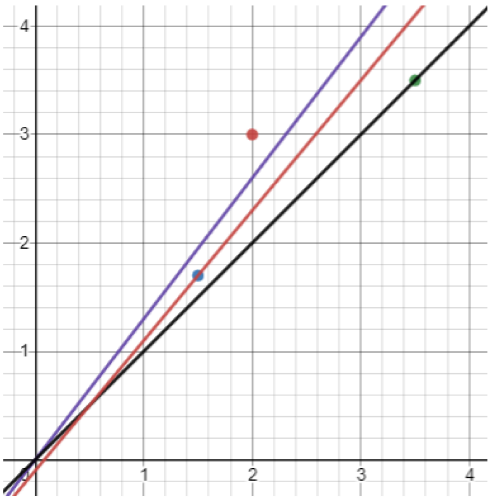

Figura 3.10: Três pontos no plano e possíveis retas que tentam se ajustar a eles.

Dentre as múltiplas opções de reta que se pode traçar, porém, qual será aquela que se encaixa melhor? Suponha a imagem 3.11, com o gráfico formado pelos possíveis valores de coeficientes (a e b) para cada reta, onde cada eixo corresponde a um coeficiente. A reta $Y = X + 1$, por exemplo, terá no gráfico um ponto marcado nas coordenadas (1, 1).

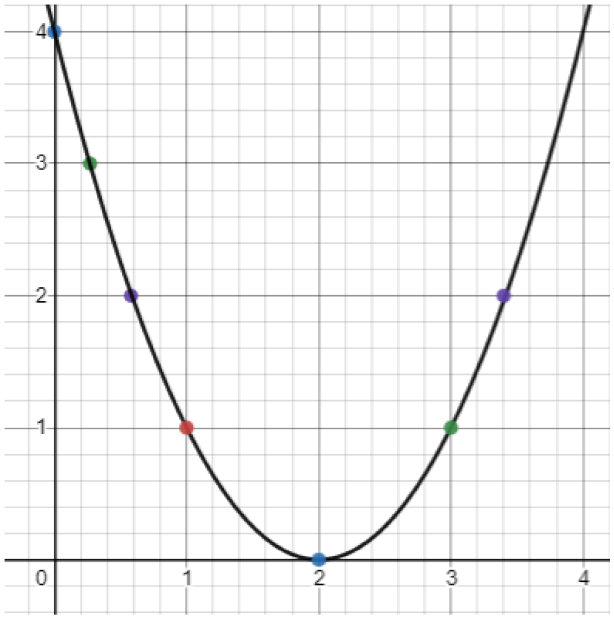

Figura 3.11: Gráfico dos coeficientes de possíveis retas que se ajustam aos dados. Ao ligar esses pontos, forma-se uma curva com um ponto de mínimo. A intenção do algoritmo de Gradiente Descendente é encontrar esse ponto mínimo da curva. Os coeficientes desse ponto descreverão a melhor reta que se ajusta aos dados.

O ponto de mínimo dessa curva é exatamente onde os erros de previsão são menores e a reta melhor se ajusta aos dados. O algoritmo do Gradiente Descendente tenta encontrar esse mínimo (sendo ele global ou local), seguindo os seguintes passos.

1. Deriva a função de custo parcialmente em relação a cada coeficiente, formando o chamado gradiente.
2. Define valores aleatórios iniciais para os coeficientes.
3. Substitui esses valores no gradiente calculado.
4. Calcula o tamanho do passo que será dado, multiplicando uma taxa de aprendizagem α pelo valor obtido em 3.
5. Atualiza os coeficientes sendo que o novo valor do coeficiente é dado pelo antigo subtraído do tamanho do passo, encontrado em 4.
6. Repete o processo a partir do passo 3 até se encontrar um valor muito próximo de 0, o que significa o mínimo foi atingido.

Trazendo novamente para o nosso contexto de RNA, o gradiente com os valores de coeficientes substituídos (∇_j) serão, por fim, multiplicados pela taxa de aprendizagem α que irá dizer o quão rápido uma rede irá convergir, ou, em termos práticos, aprender. Os coeficientes W e b então, são atualizados segundo as equações 3.11 e 3.12.

$$W_{j-1} = W_j + (\alpha \times \nabla_j) \tag{3.11}$$

$$b_{j-1} = b_j + (\alpha \times \nabla_j) \tag{3.12}$$

A atualização de pesos e vieses pode ser custosa, se feita ao final de cada época, uma vez que terá de processar todas as instâncias, expandindo o cálculo do gradiente. Portanto, vários otimizadores já foram propostos, nesse sentido, a fim de garantir uma convergência mais rápida para o mínimo local, na otimização da função de custo e reduzindo o número de cálculos.

O gradiente descendente estocástico (do inglês *Stochastic Gradient Descent* - SGD) propõe que se atualize os coeficientes para cada instância de treinamento, ao invés de esperar para atualizar somente depois que todo o lote de instâncias passou pela rede (Academy (2021)). Isso reduz a computação já que é mais barato recalcular os coeficientes para o gradiente de apenas uma instância ($(W + b) \times 1 \times num\_epocas$) do que para todo o conjunto de treinamento ($(W + b) \times num\_instancias \times num\_epocas$). Utilizando essa abordagem, a cada nova instância a entrar na rede, os cálculos serão realizados já com os pesos e vieses ajustados.

A taxa de aprendizagem α pode ser definida com um valor fixo, como ocorre no SGD, o que pode, em alguns casos, ser um problema, já que é uma das determinantes do tamanho do passo. Se o passo for muito grande quando se está próximo do objetivo, o algoritmo pode demorar a convergir. O ideal seria que o passo fosse grande quando se está distante de 0 e curto quando se está próximo, o que indica que o mínimo da função está sendo alcançado.

Assim, para que o tamanho do passo não seja sempre constante, α pode ser definida dinamicamente por um *scheduler*, isto é, o valor dela é dado por alguma função de atualização que ajusta a taxa a cada época do treinamento.

Outra forma de atualizá-la foi proposta utilizando o conceito de momento ou *momentum* de Polyak (1964). Trata-se de um vetor de velocidades que guarda a informação do caminho para o qual o gradiente está seguindo a cada iteração. Se na iteração corrente o gradiente estiver caminhando na mesma direção do passo anterior, o momento garante que ele se mova ainda mais rápido, acelerando-o. Caso esteja indo na direção contrária, ele freia o movimento. O momento é utilizado para se escolher o melhor valor de α em cada iteração para garantir que o algoritmo convirja mais rápido (Sa (2021)).

Um otimizador que se baseia nesse conceito é o Adam. Segundo Kingma and Ba (2014), o nome Adam vem de estimativa adaptativa de momento, do inglês *adaptive moment estimation*. Ele computa uma nova taxa de aprendizagem a partir de estimativas do primeiro e do segundo momento do gradiente (Kingma and Ba (2014)), ou seja, ele calcula uma média móvel exponencial do gradiente e do gradiente ao quadrado. Segundo Brownlee (2017), "os parâmetros β_1 e β_2 controlam as taxas de decaimento dessas médias móveis".

Segundo Ribeiro et al. (2020), a tarefa mais difícil na construção de uma rede neural é escolher corretamente os parâmetros que melhor a adéqua ao problema e aos dados. Sendo necessário assim, testar vários modelos, com parâmetros distintos. Nesse trabalho realizamos este exercício, alterando a topologia da rede e variando hiperparâmetros como otimizadores, funções de ativação e *scheduler*, também chamados de funções de atualização da taxa de aprendizagem, como será mostrado na Seção 6.2.

A rede neural é um tema muito explorado na área de inteligência artificial, com diversas possibilidades de aplicação. Outro método já bastante conhecido e estudado, porém, na área de pesquisa operacional é o algoritmo do caminho crítico que será abordado na Seção 3.3.

3.3 Caminho crítico

O caminho crítico é um problema amplamente estudado em pesquisa operacional que visa encontrar o caminho mais longo em uma rede, respeitando restrições (Alexander et al. (1994)). Ele é muito usado para identificar a sequência de execução mais longa e sem atrasos em um projeto composto por várias atividades ou tarefas (Schulz (2005)). Um exemplo de restrição, nesse contexto, é a dependência direta entre uma tarefa e outra, isto é, existem tarefas que só podem ser iniciadas depois que outras estejam concluídas.

Além da dependência direta, podem existir outras restrições no projeto tais como escassez de recursos ou de mão de obra que também devem ser consideradas na solução do caminho crítico. O algoritmo é composto por duas etapas: em um passo progressivo pela rede, ele calcula o início e fim antecipados para cada atividade e em um passo regressivo, computa o início e fim tardios para cada uma delas (Alexander et al. (1994)). Esses termos serão melhor explorados mais à frente nesta Seção.

Suponha um conjunto de 6 tarefas ou atividades que precisam ser executadas para a conclusão de um determinado projeto. Tais tarefas dependem de que outras delas estejam finalizadas para serem iniciadas. Cada tarefa possui também uma duração definida, ou seja, um tempo estimado para ser completada, a partir do início de sua execução. A Figura 3.12 mostra essas tarefas e as suas respectivas durações e relações de dependências. Todas as figuras desta Seção são de autoria própria.

Tarefa	Duração	Pré-requisitos
A	2	-
B	4	A
C	1	A
D	1	B
E	3	B
F	3	CDE

Figura 3.12: Exemplo de um conjunto de tarefas a ser realizado em um projeto, com suas respectivas durações de execução e pré-requisitos.

A duração é uma medida de tempo para se executar uma tarefa, podem ser dias, meses, minutos, entre outros. No exemplo da Figura 3.12, a tarefa A tem uma duração de 2 e não depende de nenhuma outra tarefa, isto é, ela pode ser a primeira a ser executada, pois, não possui nenhuma dependência. Já as atividades B e C dependem de A, então elas só podem ser iniciadas assim que A for concluída. O mesmo ocorre com D e E, que dependem de B por isso, só podem ser iniciadas assim que ela terminar. Por fim, a tarefa F depende de C, D e E para começar, ou seja, ela não pode ser iniciada antes que seus pré-requisitos sejam cumpridos. A tabela pode ser modelada em um grafo, como mostra a Figura 3.13, a fim de facilitar a execução do algoritmo.

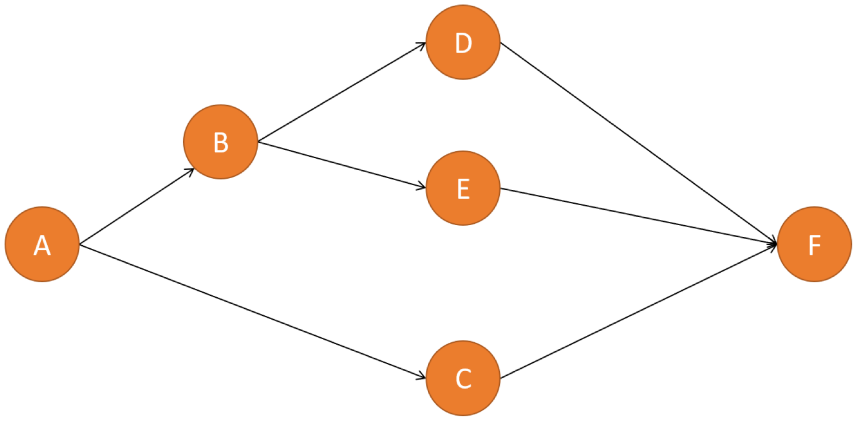

Figura 3.13: Grafo que descreve a dependência entre atividades de um projeto. A tarefa A é pré-requisito de B e C, assim como D e E não podem ser executadas antes de B e a tarefa F depende de C, D e E.

Cada atividade se transforma em um nó do grafo e elas se conectam por suas relações de dependência, ou seja, se B depende de A, deverá existir uma aresta que parte de A e vai para B. Temos assim, um grafo direcionado acíclico (do inglês *Directed Acyclic Graph* - DAG)(Rahman (2017)).

Os nós desse grafo, porém, possuem uma peculiaridade, ao invés de armazenarem apenas os nomes das tarefas, como representado na Figura 3.13, eles já podem ser modelados como uma estrutura que irá facilitar a execução do algoritmo. Dito isso, um nó no grafo é uma matriz 2x3, cuja coluna do meio armazena o nome da atividade e sua respectiva duração. Os demais campos serão preenchidos durante a execução do algoritmo, como mostra a Figura 3.14.

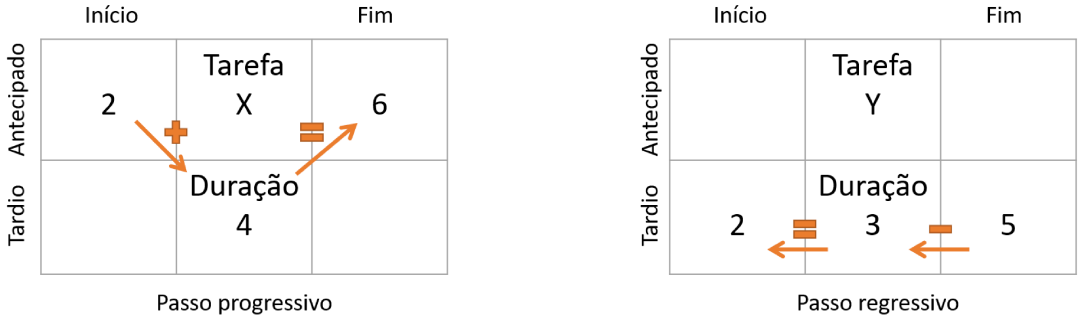

Figura 3.14: Estrutura de um nó do grafo para o algoritmo de caminho crítico.

A primeira coluna de cada nó representa os tempos iniciais de cada tarefa, ou seja, antes de ser iniciada, qual era o tempo marcado, em duas situações: na primeira linha coloca-se o tempo caso ela seja iniciada antecipadamente e na segunda, caso seja iniciada de forma tardia. Do mesmo modo, a última coluna marca os tempos (antecipado e tardio) para o fim da tarefa.

Iremos caminhar no grafo em dois sentidos, preenchendo os valores calculados para cada nó. Chamaremos de passo progressivo o caminho no grafo que vai da tarefa inicial A até a final F e de regressivo, o caminho no sentido contrário, isto é, partindo de F e chegando a A. No primeiro passo, preenche-se a primeira linha

das matrizes ou tempos antecipados e no segundo, os tempos tardios que estão na segunda linha da matriz.

No primeiro passo, soma-se o início com a duração da tarefa para se obter o tempo final, caso ela seja iniciada antecipadamente. No exemplo da Figura 3.14, a tarefa X, que se iniciou no tempo 2 e tem uma duração de 4, termina sua execução no tempo 6.

No segundo passo, computam-se os tempos, caso ela tenha sido iniciada tardiamente. O início é calculado a partir da subtração do tempo final menos a duração da tarefa. No exemplo da Figura 3.14, uma tarefa Y que terminou atrasada no tempo 5, com uma duração de 3, teria se iniciado no tempo 2.

Esse é o procedimento de preenchimento dos nós em cada um dos dois passos do algoritmo. Uma vez tendo conhecimento do final antecipado de uma tarefa, é possível saber quando a próxima atividade que depende dela pode ser iniciada. Basta que essa tarefa propague seu tempo final para todos os seus dependentes no grafo. A Figura 3.15 mostra o grafo preenchido após o passo progressivo do algoritmo.

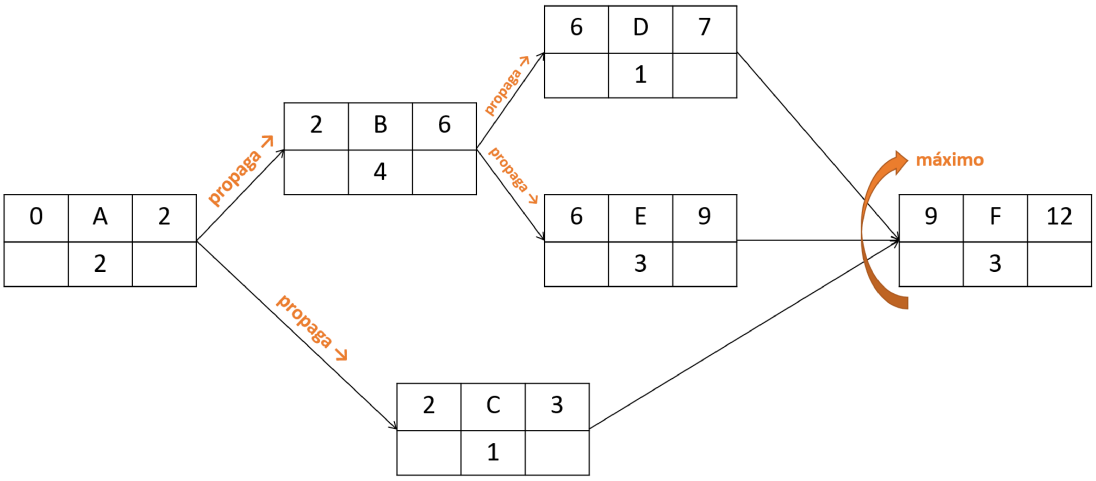

Figura 3.15: Passo progressivo do algoritmo de caminho crítico.

Nesse passo, o algoritmo tem como ponto de partida a primeira atividade que precisa ser executada, ou seja, aquela que não depende de outras, como a tarefa A. Como nenhuma tarefa foi feita ainda, iniciamos no tempo 0. Esse é o valor que irá preencher a posição (0, 0) do nó A. Na posição (0, 2) irá entrar o valor correspondente ao tempo em que A é finalizada, ou seja, $0 + 2 = 2$. Como B e C dependem de A, elas não podem ser iniciadas antes do tempo final de A, sendo 2. Esse valor é então propagado de A para B e C, sendo então o início antecipado dessas duas tarefas. Novamente, é executado o procedimento de preenchimento do nó, como descrito anteriormente, na Figura 3.14. Calculados os tempos finais antecipados de B e C, propagam-se os valores de B para seus dependentes e repete o processo. Como a tarefa F depende de 3 outras tarefas, ela só poderá iniciar sua execução assim que todas elas estiverem finalizadas. Dessa forma, ela começará ao final daquela que gastou mais tempo, ou seja, basta escolher dentre elas, aquela que obtiver o maior valor de final antecipado. No exemplo da Figura 3.15, F depende de C, D e E por isso não pode iniciar antes do tempo 9 que é o valor máximo entre os três tempos finais. Assim que se alcança a última tarefa a ser concluída, se dá início o segundo passo do algoritmo, o passo regressivo, mostrado na Figura 3.16.

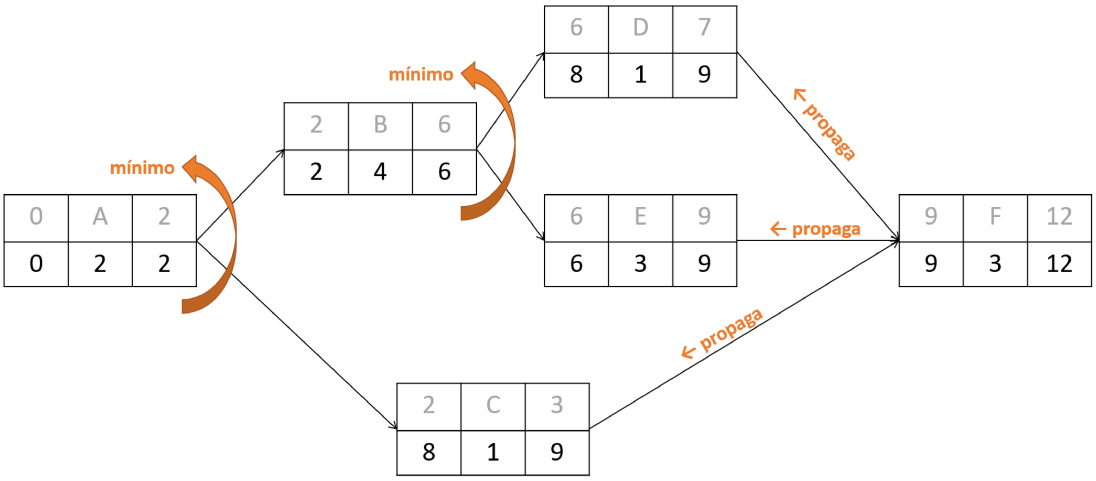

Figura 3.16: Passo regressivo do algoritmo de caminho crítico.

O passo regressivo começa no preenchimento do nó correspondente à última tarefa. Replica-se o valor final antecipado, obtido no primeiro passo, para o valor final tardio e iniciam-se os cálculos de subtração como mostrado na Figura 3.14. Sabendo o tempo de início da tarefa, ele é propagado no grafo para seus pré-requisitos como tempos de final tardio. Novamente, calculam-se os tempos de início tardio de cada tarefa que recebeu o valor propagado. Se uma atividade é pré-requisito de outras duas, ela só pode ter sido finalizada no menor dentre os tempos iniciais dessas duas tarefas. Por exemplo, a atividade B trava a execução de D e E, que se iniciaram nos tempos 8 e 6, respectivamente. Se B tivesse terminado no tempo 8, não seria possível que E tivesse sido iniciada no tempo 6, já que ela depende da finalização da execução de B. Assim, B, recebe como tempo final tardio, o valor mínimo entre os tempos iniciais tardios de seus dependentes, que, nesse caso, é 6. Mais uma vez, calcula-se o tempo de início tardio de B e deve-se escolher entre o mínimo valor de início entre B e C para ser preenchido em A, como tempo final tardio. O segundo passo do algoritmo chega ao fim, com o cálculo do início tardio de A, a primeira tarefa.

Com tudo calculado e o grafo todo preenchido, deve-se buscar por aquele caminho no qual todos os nós dele obtiveram valores iguais de inícios e fins, ou seja, faz-se uma busca para encontrar os nós que possuem valores iguais tanto na primeira coluna quanto na terceira. No exemplo aqui mostrado, os nós que satisfazem essa condição são A, B, E e F. Logo, o caminho crítico encontrado é $A \rightarrow B \rightarrow E \rightarrow F$ e possui uma duração total de 12. Assim, o projeto iniciado no tempo 0, não pode ser finalizado antes do tempo 12. Nada impede que ele gaste mais tempo que isso, pois atrasos podem ocorrer, mas em menos de 12 unidades de tempo, ele não pode ser concluído.

Aqui mostramos um exemplo de projeto cujas atividades possuíam durações diferentes. No problema tratado neste trabalho, porém, o projeto que queremos executar é a conclusão de um curso de graduação e as atividades que o compõem são as disciplinas de uma grade curricular a serem cursadas. Nesse contexto, as atividades ou tarefas, possuem a mesma duração, ou seja, um período letivo. Fazendo então um paralelo entre o exemplo mostrado e o problema que queremos solucionar, a execução do algoritmo de caminho crítico sobre um conjunto de tarefas de mesma duração é mostrada na Figura 3.17.

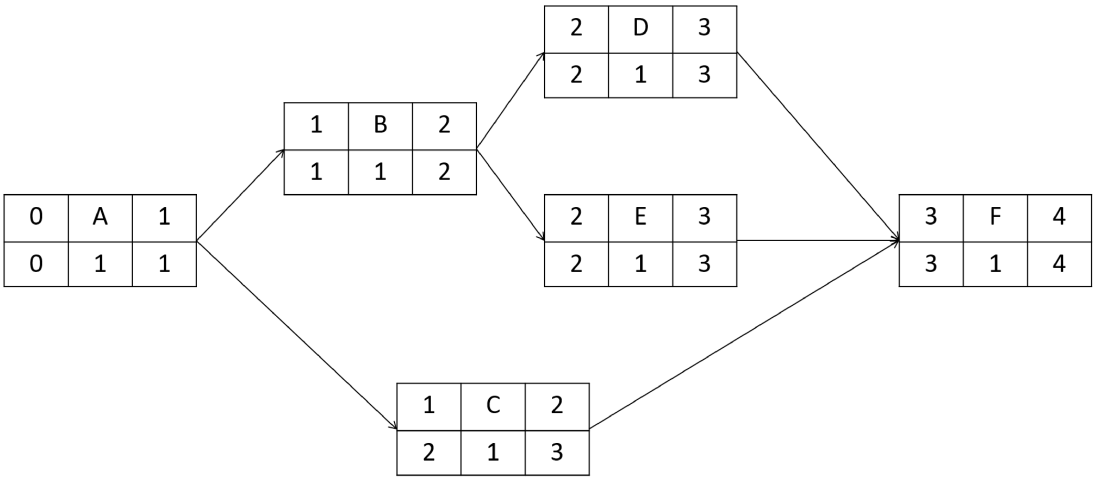

Figura 3.17: Algoritmo de caminho crítico executado para um conjunto de tarefas de mesma duração.

Percebe-se que a medida de tempo aqui é o período e cada tarefa ou disciplina duram exatamente um período. Dessa forma, o problema fica mais simplificado. A duração total do caminho corresponderá a quantidade de nós percorridos pelo caminho. No exemplo da Figura 3.17, o caminho crítico continua sendo $A \rightarrow B \rightarrow E \rightarrow F$, porém, com uma duração total de 4 períodos. Apesar de surgir um novo caminho $A \rightarrow B \rightarrow D \rightarrow F$ que também pode ser considerado crítico, com os mesmos 4 períodos de duração. Assim, podemos concluir que o aluno que tiver de cursar essas disciplinas não consegue se formar em menos de 4 períodos e finalizar todas as disciplinas para se graduar.

Neste Capítulo vimos os conceitos necessários para a compreensão dos métodos empregados na implementação do protótipo proposto por este trabalho, que serão abordados no Capítulo 4.

Capítulo 4

Material e métodos

Este Capítulo irá mostrar cada passo da metodologia utilizada. Iremos mostrar como cada método citado no Capítulo 3 foi usado para cumprir uma finalidade específica. A Seção 4.1 descreve como foram recebidos e tratados os dados e a Seção 4.2 mostra as investigações realizadas sobre eles de modo a entendê-los melhor.

As próximas seções constituem os módulos do protótipo final. Enquanto a Seção 4.3 expõe as etapas seguidas na identificação dos padrões frequentes relacionados a reprovações, a Seção 4.4 aborda os passos para a criação de um modelo capaz de predizer a chance de reprovação. Por fim, a Seção 4.5 traz um ponto diferente da reprovação, mas que também contribui para a tomada de decisão do estudante. Ela introduz a finalidade do caminho crítico em nosso contexto e como surgiu a proposta de uma heurística para preenchimento automático do plano, o que pode garantir mais agilidade nos planejamentos.

4.1 Pré-processamento

A Pró-Reitoria de Ensino (PRE) da UFV de Viçosa forneceu uma base de dados extraída do sistema de gerenciamento de históricos acadêmicos, o Sapiens. Essa base contém registros desde 2003 até 2017, contando com 23.636 estudantes, 2.493 disciplinas, 1.832 professores e 819.326 conjuntos de históricos. Um dado histórico representa uma disciplina cursada por um estudante em um determinado ano e semestre, além de mostrar a nota ou conceito alcançados pelo estudante e em quais turmas teóricas e/ou práticas cursou a disciplina naquele período. Sendo assim, um registro de histórico é identificado pelas chaves: estudante, disciplina, ano e semestre. Neste Capítulo, serão apresentados diagramas e figuras, de autoria própria com o intuito de facilitar a compreensão. O esquema da Figura 4.1 mostra um exemplo de como é representado um registro de histórico.

Matrícula	Disciplina	Ano	Semestre	Nota	Conceito	Turma teórica	Turma prática
12345	ARQ 106	2014	1		L	1	1
12345	INF 100	2014	2	66		1	2

Figura 4.1: Exemplos de registros históricos.

Um mesmo estudante terá mais de um registro na tabela, pois a chave primária

é uma chave composta. Os registros possuem notas ou conceitos, dependendo da característica avaliativa da disciplina ou do comportamento do estudante. O conceito L, por exemplo, indica que o aluno excedeu o limite de faltas permitido. Nesse caso, a coluna de nota pode vir preenchida juntamente com a coluna de conceito, porém quando aparece o conceito L, entende-se que houve uma reprovação por frequência e a nota é desconsiderada. Se a disciplina for apenas de caráter prático, a coluna de turma teórica aparecerá zerada e se for apenas de caráter teórico, o mesmo acontece na coluna de turma prática. Para disciplinas teóricas que possuem parte prática, as duas colunas vêm preenchidas. Os dados referentes a identificação de estudantes e professores, como matrícula e código, respectivamente, foram devidamente anonimizados para impedir o reconhecimento. Com os dados fornecidos em vários arquivos de formato CSV, foi construído um banco de dados, com *SQLite*, para facilitar as consultas e extração. O diagrama do banco, com suas tabelas, atributos e relações é mostrado na Figura 4.2.

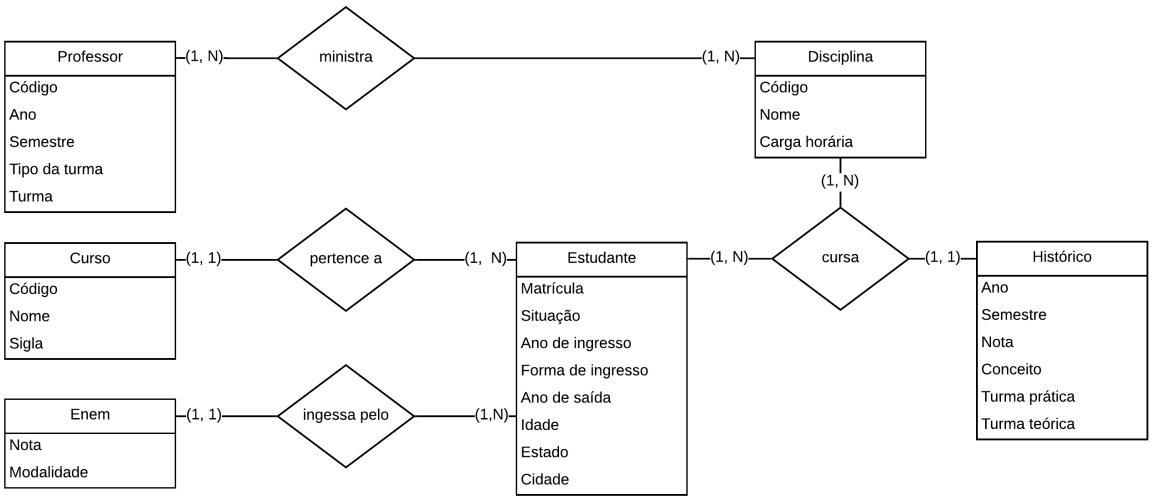

Figura 4.2: Diagrama entidade-relacionamento do banco de dados criado a partir das planilhas eletrônicas fornecidas.

Na tabela Estudante, foram colocados os dados pessoais de cada aluno que passou pela UFV desde 2003. Estudantes que trocaram de cursos recebem um código diferente, ou seja, se trocou de curso de uma vez, haverá duas instâncias no banco para o mesmo estudante, uma para cada curso, com código de identificação distintos. Cada estudante possui uma forma de ingresso na universidade, se foi pelo Exame Nacional do Ensino Médio (ENEM), será atrelado à tabela ENEM que contém a nota alcançada no exame e a modalidade de cota escolhida no SISU. A tabela Curso abrange todos os cursos ministrados na UFV, Campus Viçosa, com seus códigos, nomes e siglas. Como um curso possui vários estudantes, essas duas tabelas possuem uma relação de 1 para N. Disciplina é a tabela que traz todos os códigos de disciplinas já ofertadas na UFV, Campus Viçosa, desde 2003. Os outros dois atributos dessa tabela, foram incluídos posteriormente, com um procedimento de *Web-scrapping* que será melhor descrito adiante, neste mesmo Capítulo. A tabela Professor contém os registros de qual professor ministrou cada disciplina em um dado ano e semestre e por quais turmas foi responsável. Dessa forma, um mesmo professor pode estar associado a

vários registros dessa tabela. E como um professor pode oferecer várias disciplinas, da mesma maneira que uma disciplina pode ser ministrada por mais de um professor, a relação entre essas tabelas é da forma N para N. Por fim, como definido anteriormente, a tabela Histórico é constituída de registros de chave composta, ou seja, um estudante pode ter vários registros nessa tabela, que compõem seu histórico acadêmico. Do mesmo modo, uma disciplina aparecerá em vários registros de histórico, mantendo uma relação de 1 para N. Os quatro primeiros atributos (matrícula, código da disciplina, ano e semestre) integram a chave primária de um registro de histórico.

Como a intenção principal é trabalhar com reprovações, existem algumas disciplinas que são mais preocupantes que outras, pois possuem números bem elevados. Nesse sentido, a base passou por um filtro, com critérios definidos pela PRE a fim de manter apenas os registros das disciplinas consideradas alarmantes. Foram filtradas aquelas que continham pelo menos 25 matriculados em média ao longo dos períodos em que foram oferecidas e um percentual de reprovação superior 40% em, pelo menos, uma oferta. Para os dois casos, foram desconsiderados os *outliers*, aqueles cuja média extrapolaram 2 desvios padrão. As disciplinas cuja média geral dos percentuais de reprovação em cada oferta ultrapassou novamente 40%, foram consideradas disciplinas críticas, já que, frequentemente, elas possuem um número elevado de reprovações. Também foram removidas disciplinas que possuíam menos de 5 ofertas, ou seja, não possuíam dados suficientes para a análise. Dentre elas, podem estar disciplinas antigas que não existem mais ou disciplinas recém-criadas que ainda não foram ofertadas pelo menos 5 vezes.

Além disso, estávamos interessados na carga horária de cada disciplina e em seus nomes, não apenas nos códigos, dados estes que não estavam presentes na base fornecida inicialmente. Porém, por ser um dado público, o extraímos com *Web-scrapping* do site institucional Catálogo UFV [1]. O catálogo possui dados desde 2006, portanto, há disciplinas para as quais não foram encontradas informações. Ficaram faltando dados de carga horária e nome para 12 disciplinas, o que não compromete o estudo, em um universo de mais de 2.000 disciplinas. Por fim, também foi aplicado um filtro de data: foram considerados apenas aquelas disciplinas que possuem registro de oferta a partir de 2013, o ano em que a UFV começou a adotar o ENEM como forma de ingresso.

4.2 Análise exploratória

Uma vez tendo a base já pré-processada, iniciou-se uma etapa de análise exploratória. No intuito de investigar se a reprovação era afetada por características do estudante, foram elaboradas algumas visualizações de dados confrontando atributos do estudante com a reprovação. Além disso, foi produzido um modelo preditivo preliminar de árvores aleatórias, com 1.000 estimadores, de modo a identificar se seria possível construir, posteriormente, um modelo robusto de predição em torno de uma variável que explicasse a reprovação.

Neste modelo inicial, os atributos considerados foram a nota que o estudante atingiu no ENEM e usou para ingressar na faculdade, o curso e a unidade federativa do aluno, sua forma de ingresso na universidade e a modalidade de cota concorrida. O alvo do modelo foi criado a partir do histórico do estudante, indicando se houve ou

[1] http://www.catalogo.ufv.br/

não reprovação em seu histórico. Foram considerados apenas os alunos que cursaram pelo menos 4 períodos. A partir disso, foi verificado, para cada disciplina cursada, se o estudante obteve a nota mínima de 60 pontos para ser aprovado ou não. Os conceitos que indicam reprovação, independentemente das notas obtidas também foram apurados, são eles: L para reprovação por infrequência, F por fraude e N por um desempenho não satisfatório.

Uma vez, sabendo se houve reprovação em cada um dos registros de histórico, calculou-se, o número de reprovações totais de cada estudante e o número de períodos em que houve reprovação. Para entrada do modelo, foram considerados reprovados aqueles estudantes que falharam em no mínimo 4 disciplinas ao longo de seu histórico acadêmico. Alunos que reprovaram menos que isso, não foram vistos como um problema. Em média, os alunos cursam por volta de 50 disciplinas ao longo de seus cursos de graduação logo, consideramos aceitável 3 reprovações ou menos. O conjunto de dados foi dividido em treino e teste, sendo 30% separado para o último. A matriz de confusão para esse modelo preliminar já mostrou um indício de que um novo modelo com mais atributos seria promissor por isso, seria um ponto interessante a ser explorado em uma etapa posterior.

Contudo, o próximo passo foi procurar por padrões frequentes na base completa, ou seja, situações que corriqueiramente ocorrem juntas e, principalmente, àquelas que acontecem em conjunto com reprovações.

4.3 Mineração de padrões frequentes

Para realizar a mineração de padrões frequentes, construiu-se uma base de transações a partir dos dados iniciais e utilizamos o algoritmo *Apriori* sobre ela para extrair os itens frequentes que haviam ali, com um suporte mínimo de 40%. Uma vez, tendo os itens que ocorrem com mais frequência, extraímos as regras de associação com uma confiança mínima de 70%. Uma transação na nova base é composta por dados do estudante, como nota do ENEM usada para ingressar na universidade, sua modalidade de cota utilizada e seu curso. Além disso, também compõem a transação, o conjunto de disciplinas cursadas em um período (ano/semestre), o resultado obtido em cada uma delas (aprovado ou reprovado), os professores que as ministraram, identificados pelo tipo de turma (teórica ou prática), a carga horária total, dada pela soma de créditos de cada disciplina cursada, o número total de disciplinas e o período, sem identificar o estudante. Um exemplo de transação é mostrado no esquema 4.3.

FIS 225_A, QUI 320_A, QUI 151_A, QUI 251_R, FIS 203_R,
T:7ef61751, P:257c52ac, T:b98acc46, T:e39884e6, T:f82c6c98,
18 créditos, 5 disciplinas, 2013-2, curso: 125.0,
modalidade enem: 4, nota enem: 600

Figura 4.3: Exemplo de registro da base de transações.

O exemplo da Figura 4.3 mostra uma transação da nova base, composta por 5 disciplinas, cursadas no segundo semestre de 2013, que somam a carga horária de 18 créditos. Na primeira linha, são mostradas as disciplinas com os sufixos A e R que

indicam aprovação e reprovação, respectivamente. Na segunda linha são mostrados os códigos anonimizados dos professores que ministraram as disciplinas com os prefixos T e P que indicam turma teórica e prática, respectivamente. A sequência longa de caracteres se refere a anonimização do nome do professor. O código do curso do estudante, mostrado na terceira linha, também compõe a transação. Por fim, na última linha, estão os dados relativos ao ENEM daquele estudante: a nota obtida e a modalidade de cota escolhida.

Devido ao grande número de disciplinas e a variedade entre os centros de ensino da UFV, não foram encontradas tantas regras relevantes. Dessa forma, acreditamos que poderíamos encontrar regras mais fortes em grupos de disciplinas semelhantes que descrevem cenários mais específicos e similares. Isso provavelmente reforçaria as regras gerais encontradas na base toda.

Portanto, foi utilizado o algoritmo *k-means* com um k = 4, escolhido através da análise de silhueta e do método do cotovelo, para clusterizar a base em 4 grupos de disciplinas semelhantes entre si e diversos em relação aos demais. Tendo os grupos formados, foram replicados os procedimentos realizados na base toda, porém para cada *cluster*. Mais detalhes serão abordados na Seção 6.1.

O conjunto dos principais resultados relacionados aos padrões e perfis de disciplinas encontrados integram um relatório destinado à PRE, a fim de que possam analisar e tomar medidas institucionais cabíveis para melhorar o ensino cada vez mais.

Uma vez identificados os padrões que ocorrem com mais frequência, a próxima etapa explorada foi a construção de um modelo preditivo mais robusto que o modelo preliminar com árvores aleatórias.

4.4 Modelo preditivo

A primeira alteração proposta para o novo modelo foi a mudança da variável alvo na construção das novas tuplas. Entendemos que representar uma ocorrência de um semestre para cada aluno seria uma melhor abordagem que considerar apenas características mais constantes dos estudantes, como estado de origem ou curso. Dessa forma, o alvo passa a ser prever qual a chance de um aluno reprovar em alguma das disciplinas que escolheu cursar no próximo semestre. Assim, uma instância para o novo modelo preditivo deve refletir uma ocorrência no tempo. Para isso, as tuplas de entrada foram criadas com o processo de *Feature Engineering*.

Considerando esse aspecto temporal, uma instância é composta por três tipos diferentes de atributo construídos: independente de histórico, referente à transição entre o período atual e o próximo e referente ao histórico a partir do período atual. Cada registro na base de dados inicial é marcado por um ano e um semestre. Logo, cada registro na nova base, simula uma situação que ocorreu naquele espaço de tempo, marcado por tal ano e período. Todos os registros para um mesmo estudante que possuem o ano menor que o simulado como atual, a cada iteração, é considerado como histórico daquele aluno até então. Um exemplo é mostrado no esquema 4.4.

Ano	Semestre	Estudante	Número de disciplinas
2017	1	Maria	6
2017	2	Maria	4
2018	1	Maria	5

Semestre letivo	Estudante	Número médio de disciplinas por período
2017 / 1	Maria	-
2017 / 2	Maria	6
2018 / 1	Maria	5 (6 + 4) / 2
2018 / 2	Maria	5 (6 + 4 + 5) / 3

Figura 4.4: Transcrição da base de dados original para a base de transações.

No exemplo hipotético da Figura 4.4, Maria é uma estudante que possui três registros na base de dados original. Ela ingressou em 2017/1 e, em 2018/1, ela está cursando seu terceiro período na universidade. Para construir instâncias temporais, levamos em conta cada período cursado por Maria. Considerando 2017/1 o período atual, Maria ainda não possui nenhum histórico, consequentemente, os atributos dependentes de histórico serão vazios para esta instância. Já no período 2017/2, Maria possui o histórico do período anterior: 2017/1. Portanto, os atributos relacionados a histórico compreendem os dados do período anterior. Quando tratamos 2018/1 como período atual, o histórico de Maria contém os dois períodos anteriores. Dessa forma, os atributos dependentes de histórico serão correspondentes ao resultado cumulativo dos períodos anteriores. Se considerarmos, por exemplo, o número médio de disciplinas cursadas por período, em 2017/1, como Maria ainda não possui histórico, não é possível calcular. Já em 2017/2, o número médio será o número de disciplinas cursadas no período anterior dividido por 1 que resultará em 6, ou seja, o próprio número de disciplinas cursadas em 2017/1. Em 2018/1, o valor desse atributo é dado pela soma do número de disciplinas cursadas nos dois períodos anteriores divididos por 2 para se obter o número médio e, assim por diante. Esse raciocínio vale para o cálculo de todos os atributos relativos a histórico. A imagem 4.5 traz todos os atributos construídos através do método de *Feature Engineering*, indicando também o tipo de cada um. *Feature Engineering* é um processo no qual cálculos são realizados sobre os atributos iniciais de modo a produzir novos atributos ou *features* (Nargesian et al. (2017)).

Independente de histórico
Matrícula *
Curso
Dados coletados na transição entre o período atual e o próximo
Carga horária que o aluno deseja cursar
Número de disciplinas matriculadas no plano de estudos
Número de reprovações *
Coeficiente de rendimento do período que está finalizando
Coeficiente de rendimento sobre o número de disciplinas (CRD)
Média dos alunos nas turmas matriculadas
Próximo semestre (1 ou 2)
Idade naquele período
Histórico a partir do período atual
Carga horária média por semestre
Número médio de disciplinas por semestre
Número médio de reprovações por semestre
Número total de reprovações
Coeficiente de rendimento acumulado
Coeficiente de rendimento médio
Nota média nas disciplinas
Número de disciplinas com notas maiores que 80
Número de disciplinas com notas entre 60 e 80
Número de disciplinas com notas menores que 10
Número de reprovações por frequência
Tempo total de UFV até o período atual

Figura 4.5: Atributos selecionados e construídos para as instâncias de entrada dos modelos preditivos.

Os atributos marcados com * na Figura 4.5 foram removidos após a criação das tuplas. O primeiro é a identificação do aluno e somente foi necessário no momento da construção dos atributos. O segundo é o alvo que desejamos prever com o modelo: o número de reprovações que um estudante provavelmente terá no próximo período cursado. Para representar o curso do estudante, foram escolhidos os códigos definidos pela UFV e identificados por números inteiros. Os atributos da segunda seção do diagrama são aqueles calculados apenas para o período atual. O Coeficiente de Rendimento por Disciplinas (CRD) foi um atributo criado pelos autores deste traba-

lho no intuito de dar um indício de sobrecarga ou desempenho. Para isso, divide-se o coeficiente alcançado no fim do período pelo número de disciplinas cursadas. Se esse número for baixo, existem duas opções: ou o número de disciplinas foi muito alto o que caracteriza a sobrecarga ou o coeficiente é que foi baixo, caracterizando um mau desempenho no período. Por fim, a terceira seção do diagrama traz os atributos calculados para todo o histórico do estudante, a partir do período atual, como exemplificado na Figura 4.4.

O novo conjunto de dados criado contou com 153.132 registros que tiveram de passar pelo processo de *One hot encoding* (Srinidhi (2018)), devido à presença de um atributo categórico: o curso do estudante, que apesar de ser um número inteiro, é uma identificação, apenas, e não um quantitativo. Muitos modelos de *Machine learning* não lidam bem com dados categóricos e precisam que atributos desse tipo sejam transformados em numéricos. Porém, apenas a simples transcrição de categoria para um número que a represente, como funciona o processo de *Label encoding* (Srinidhi (2018)), pode indicar para o modelo que a categoria 4 é maior que a 3, por exemplo. Uma comparação que não é válida para esse caso, já que não é coerente dizer que o curso de um estudante é maior que o curso de outro estudante. Sendo assim, o mais adequado é extrair os possíveis valores de um atributo categórico e replicá-los em colunas, criando artificialmente, uma quantidade de atributos novos, correspondentes ao número de valores que o atributo categórico inicial pode assumir. Cada um desses novos atributos pode assumir dois valores lógicos, 0 ou 1 (falso ou verdadeiro, respectivamente). O esquema da Figura 4.6 mostra como é feita essa conversão.

Matrícula	Curso
12345	ARQ
56789	ENQ
90123	MAT

Matrícula	ARQ	ENQ	MAT
12345	1	0	0
56789	0	1	0
90123	0	0	1

Figura 4.6: Processo de *One hot encoding* sobre o atributo curso.

No esquema 4.6, foram utilizadas as siglas dos cursos para ficar mais intuitivo, porém vale lembrar que os cursos neste novo conjunto de dados são números inteiros que representam cada uma das siglas. Com o processo de *One hot encoding*, foram acrescidas à base 50 novas colunas de valores 0 ou 1, correspondentes aos cursos da UFV - Campus Viçosa, registrados na base inicial.

Posteriormente, os dados foram normalizados com o *StandardScaler* da biblioteca *Scikit-learn* (Pedregosa et al. (2011)) para deixar todas as variáveis numéricas na mesma ordem de grandeza. O *StandardScaler* implementa um processo de padronização, no qual o novo valor do atributo é reduzido da média do conjunto de todas as instâncias e o resultado dividido pelo desvio padrão, segundo a fórmula do *z-score* (equação 4.1).

$$z = \frac{(x - m)}{d} \tag{4.1}$$

Sendo x, o valor inicial do atributo a ser padronizado para cada instância, m, a média e d, o desvio padrão de todos os valores assumidos. Todas as instâncias

passam por esse cálculo e o conjunto de valores normalizado fica entre 0 e 1. No fim, a média vai para 0 e o desvio padrão para 1. Com o conjunto de dados tratados, a matriz de correlação entre os 19 atributos numéricos (com exceção do curso) e o *target* foi obtida com o propósito de investigar aqueles que estão correlacionados.

Em seguida, o conjunto foi subdivido em dois: um conjunto de treinamento e um de teste, sendo esse representado por 15% (determinados empiricamente) dos registros totais. Quatro modelos foram executados sobre as bases de treino e teste: *Random Forest Regressor* (RFR), *Multilayer Perceptron* (MLP), *Support Vector Machine* (SVM) e uma rede neural. Usando os modelos implementados na *Scikit-learn*, o MLP mostrou um desempenho melhor que o *Random Forest* e o *SVM*, o que nos levou a optar pela construção de um modelo de rede neural.

Para alcançar um bom modelo final, que fosse capaz de prever, com confiança, o número de reprovações, foram feitos vários experimentos. A primeira etapa testou isoladamente a performance do modelo para cada parâmetro possível de ser incluído na rede. A partir dos resultados dos testes, a segunda etapa foi feita com modelos de redes neurais, combinando de diferentes formas os parâmetros que demostraram melhor desempenho. Algumas métricas foram estabelecidas no início dos experimentos e confrontadas, logo após os testes, a fim de escolher um modelo final de rede neural artificial. Esse processo será visto com detalhes na Seção 6.2.

4.5 Caminho crítico e heurística

Do ponto de vista do estudante, ter conhecimento dos padrões que ele deve evitar repetir e o número provável de reprovações que ele terá em um dado semestre, o deixa cada vez mais próximo de sua graduação. Porém, outro ponto interessante a ser considerado durante o plano de estudos é o tempo mínimo necessário que esse aluno precisa para se formar. Diante disso, foi implementado um algoritmo de caminho crítico para determinar o número mínimo de períodos que o estudante, obrigatoriamente, ainda precisa cursar, ou seja, dado sua situação acadêmica, ele não consegue formar em menos tempo do que o identificado pelo caminho crítico. O funcionamento do algoritmo aplicado a esse contexto será melhor descrito na Seção 6.3.

Sabendo o número mínimo de períodos que um estudante deve ficar na universidade e tendo em vista que o desejo é formar o quanto antes, é possível automatizar o planejamento de escolhas de disciplinas até o final de seu curso. De modo a fornecer uma solução que possa ser empregada no sistema acadêmico de planos de estudos da universidade, facilitando o trabalho de orientadores e otimizando suas reuniões com estudantes em fins de período, foi criada uma heurística para preencher automaticamente todos os períodos restantes de um estudante na universidade. Isso não exclui o papel do orientador, pois ele conseguirá validar ou não, com olhar crítico, o resultado fornecido pelo algoritmo e fazer ajustes, se necessário. O funcionamento da heurística será melhor abordado na Seção 6.4.

Neste Capítulo foi mostrado cada passo da metodologia empregada. O Capítulo 5 abordará os resultados alcançados através dessa metodologia.

Capítulo 5

Resultados e discussão

As ferramentas e análises descritas no Capítulo anterior nos permitiram alcançar resultados interessantes, que serão melhor detalhados neste Capítulo. A Seção 5.1 descreve os resultados encontrados a partir da exploração dos dados. Já na Seção 5.2 serão mostrados os padrões identificados na base da UFV, após algumas maneiras de extração. Na Seção 5.3 serão discutidas cada etapa de avaliação do modelo e mostraremos como escolhemos um preditor final. Por fim, a Seção 5.4 conclui o Capítulo fazendo algumas considerações acerca do algoritmo de caminho crítico e da heurística gulosa implementadas.

5.1 Análise exploratória

Posteriormente ao método de pré-processamento, descrito no Capítulo 4, incluindo os filtros definidos pela PRE, foram selecionadas 114 disciplinas, sendo 31 delas, consideradas críticas. O corte temporal que seleciona, apenas, registros a partir de 2013, resultou em 557.039 registros de histórico, um número ainda bem significativo e com dados mais atuais, que refletem melhor a realidade da UFV, no momento.

Os resultados da análise exploratória sobre os dados pré-processados, confrontando atributos dos estudantes com a porcentagem de reprovados, podem ser vistos nos próximos gráficos. A Figura 5.1 mostra o percentual de reprovação dentre os cursos da UFV.

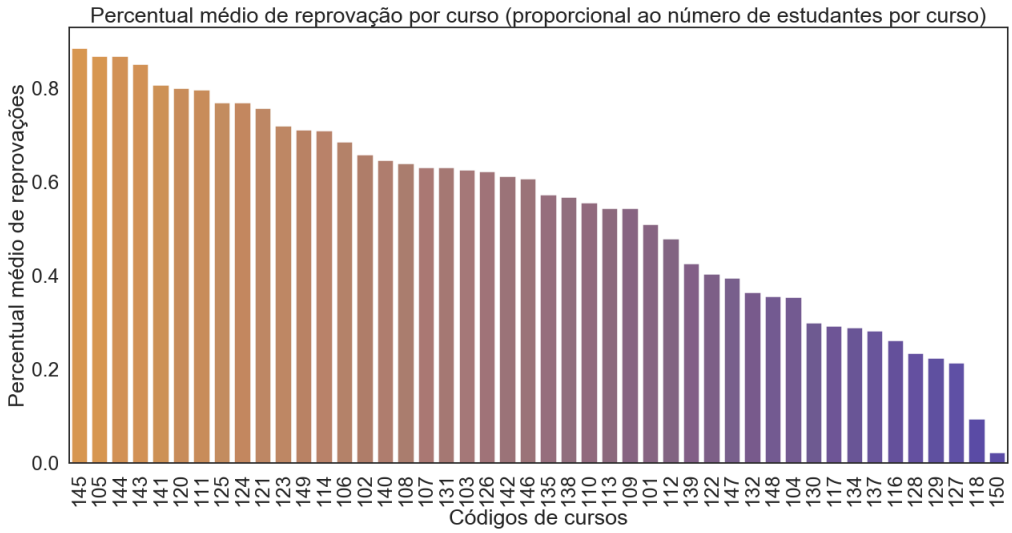

Figura 5.1: Percentual de reprovação por curso.

Percebe-se que, em alguns cursos, o percentual médio de reprovação é muito alto e ele varia consideravelmente de curso para curso. Dentre os cursos que menos reprovam, estão Medicina, Direito e Pedagogia e os cinco cursos que mais reprovam são Licenciatura em Química, Ciências Econômicas, Licenciatura em Matemática, Licenciatura em Física e Ciência da Computação. Essa informação é útil para ser levada e discutida nas coordenações desses cursos ou até mesmo dentro da Pró-reitoria de Ensino.

A Figura 5.2 traz essa mesma relação de reprovados agora confrontados com atributos relativos à entrada do estudante como a modalidade de ingresso.

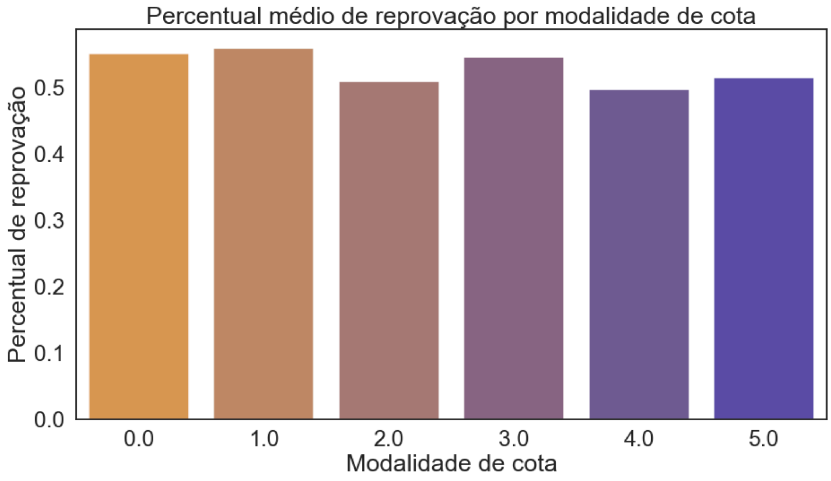

Figura 5.2: Percentual de reprovação por modalidade de cota.

Em relação à modalidade de cota, o percentual médio não flutua muito, são pequenas as variações de uma para outra. Apesar de todos se manterem próximos aos 50%, ou seja, pelo menos metade dos estudantes que ingressam na universidade em cada grupo de cota reprovam alguma vez durante sua jornada acadêmica, segundo o crité-

rio estabelecido. Não é a modalidade de cota que vai ditar o desempenho acadêmico do estudante, mas ela pode auxiliar na predição.

Com o aumento do alcance demográfico que o ENEM proporcionou para o ingresso na universidade, aumentou-se também a entrada de pessoas que não tinham interesse primário pelo curso para o qual foram aprovadas. Algumas consequências desse comportamento são prejudiciais tanto para o estudante quanto para a própria Universidade. O aluno pode ficar frustrado, desanimado e não se esforçar muito, o que o leva a ter notas baixas, reprovações ou até mesmo evadir o curso e desvalorizá-lo. Como é recorrente em projetos de ciência de dados, essa reflexão cria oportunidade para outras questões, que podem ser estudadas em trabalhos futuros. Nesse âmbito, seria possível usar mineração de dados para melhor direcionar estudantes rumo a uma carreira profissional que os agrade?

Outro aspecto importante a ser analisado é o estado de origem do estudante, mostrado na Figura 5.3.

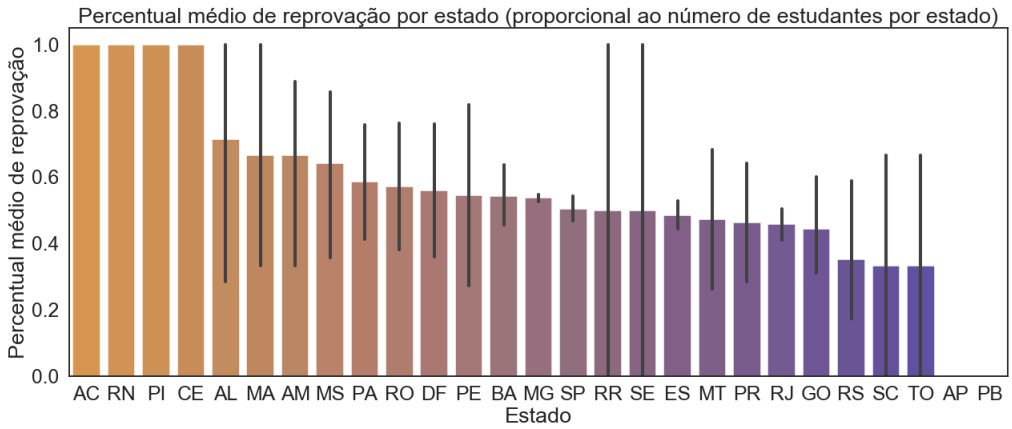

Figura 5.3: Percentual de reprovação por estado de origem.

É perceptível que o percentual também varia de estado para estado, sendo Acre, Rio Grande do Norte, Piauí e Ceará, os mais críticos em relação à reprovação. A alta porcentagem indica que a maioria dos estudantes que vieram dessas UFs reprovaram alguma vez na UFV. Entretanto, a ausência da linha preta sobre a barra indica que não há variabilidade sobre os dados desses estados (Waskom (2021)). O desvio padrão é nulo, assim todos os alunos que vieram dali reprovaram. O que também ocorre com Amapá e Paraíba, porém de forma contrária: todos os alunos desses dois estados nunca foram reprovados segundo os critérios de corte. Com isso, temos o indício de que essas amostras podem ser muito pequenas e dessa forma, é válido olhar para a quantidade de estudantes matriculados por estado ou região, como mostra a Figura 5.4.

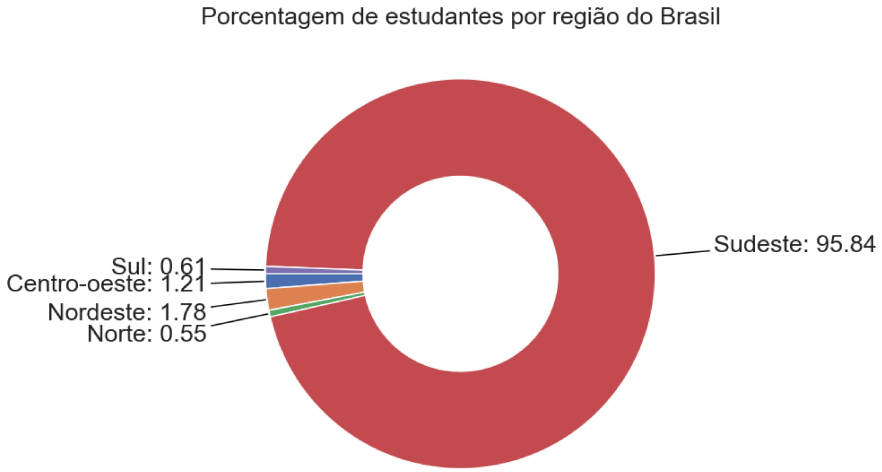

Figura 5.4: Percentual de alunos por região.

Na figura, percebemos que a maioria dos estudantes da UFV vêm da região Sudeste, cerca de 95% dos registros do banco de dados são de estudantes advindos dos estados que compõem essa região. Dessa forma, foi feito um recorte para se analisar melhor esse aspecto, já que os outros estados não demonstraram tanta relevância estatística. Podemos notar que o percentual de reprovação médio não varia muito entre os estados da região Sudeste, mas existe uma pequena variação, sendo MG o estado com maior índice de reprovação e RJ o menor, como podemos visualizar na Figura 5.5.

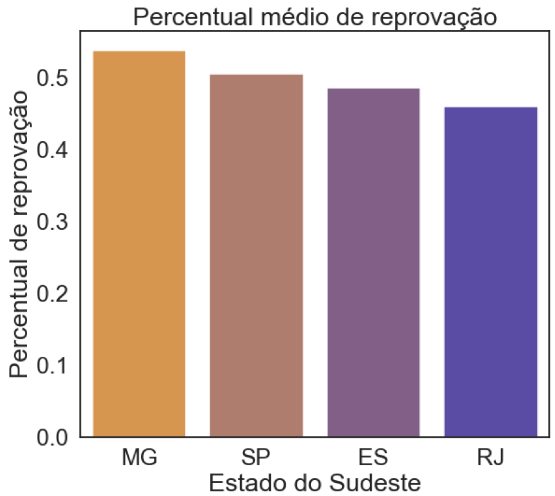

Figura 5.5: Percentual de reprovação por estado de origem localizado na região Sudeste.

Portanto, sobre a UF de origem, é possível notar que apesar de alguns estados da região Norte e Nordeste terem os maiores percentuais de reprovações na UFV, poucos

alunos vêm desses lugares. Assim, filtrando somente a região Sudeste, percebe-se que aqueles alunos que já são de MG, ou seja, a maioria dos estudantes, reprova mais. Apesar de não variar consideravelmente, todos esses estados obtiveram taxas superiores a 40% e considerando o gráfico geral, todos eles superam 30%, exceto AP e PB que não possuem nenhum registro na base desde 2013. Com isso, percebe-se alguma influência da localidade na previsão de reprovação e ela pode ser incorporada como uma *feature* do modelo.

Por fim, ao olhar para a distribuição (Figura 5.6) de notas alcançadas no ENEM e o percentual de reprovação, nota-se uma distribuição normal, ou seja, poucos estudantes da UFV possuem notas próximas a 500 no ENEM e dentre os que possuem, a maioria já reprovou. Por outro lado, poucos estudantes também conseguem notas altíssimas, próximas a 800, mas dentre esses, a maioria não reprova. A maior parte dos estudantes se encontre entre o intervalo de notas de 600 a 700, cuja a curva de reprovação supera em 1% a curva de aprovados (aqueles que durante sua vida acadêmica obtiveram menos de 4 reprovações no total).

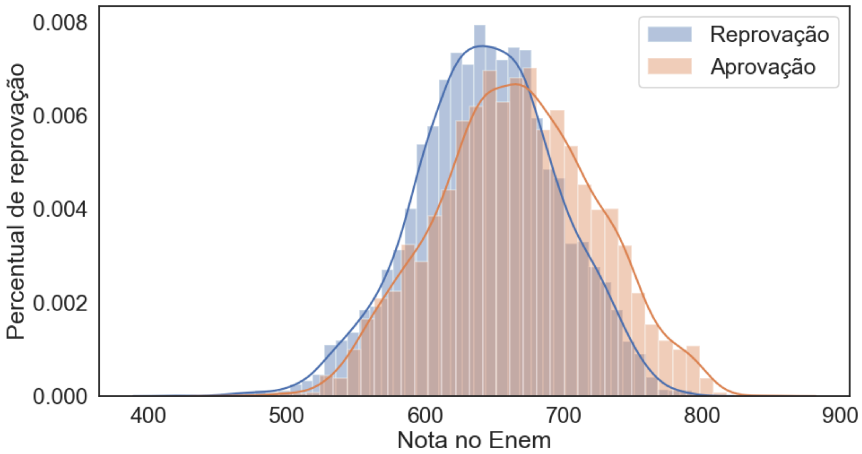

Figura 5.6: Percentual de reprovação por distribuição de notas no ENEM

No gráfico de distribuição das notas alcançadas no ENEM pela porcentagem de reprovação, percebermos uma distribuição normal tanto na curva de aprovados quanto de reprovados. Entretanto, é notável que a curva de aprovação está mais à direita, ou seja, mais perto das maiores notas. Poucos alunos, em geral, conseguem mais de 750 pontos nessa prova, mas poucos também ingressam na faculdade com menos de 550 pontos. Aqueles que ingressam com notas menores consequentemente estão reprovando mais.

Outro fator relevante para a análise de reprovação, é o caráter das disciplinas. Após obter essa visão geral da reprovação na UFV, foi construído também um módulo de análise individual por disciplina, que pode ser incorporado em um painel para monitoramento da PRE ou quaisquer outros órgãos da Universidade.

Algumas disciplinas possuem um nível de dificuldade maior que outras e exigem mais dos alunos. É importante então, poder analisar o contexto específico para cada disciplina, principalmente as críticas. Uma vez sabendo quais são as disciplinas críticas, pode-se investigar melhor o que ocorre em cada uma delas. Por isso, através de um módulo exploratório que tem como entrada o código da disciplina, é possível

observar suas taxas de reprovação ao longo do tempo. Utilizamos como exemplo, a disciplina de Introdução à Programação, INF 100. Os gráficos são mostrados na Figura 5.7.

Pode-se notar que o número de matriculados nessa disciplina segue um padrão de variação, o que pode ser explicado pelo número de vagas ofertadas por semestre do ano. Trata-se de uma disciplina muito demandada e que, na maioria das vezes, tem seu número total de vagas preenchidos. Em semestres ímpares são ofertadas mais vagas já que é o período no qual a UFV faz seu processo de admissão de novos estudantes e essa disciplina é ofertada, principalmente para calouros. O número de reprovações, porém, não oscila como o de matriculados, ele segue uma tendência mais constante. O que é preocupante, principalmente naqueles períodos em que a oferta é menor.

Ao segmentar por tipo de reprovação, nota-se que a taxa de abandono nessa disciplina é altíssima, superando a própria reprovação em alguns períodos, ou seja, muitos alunos desistem de cursar INF100, outro dado preocupante. Porém, outro ponto perceptível pelos gráficos é que aqueles que não desistem, ou estão entre os aprovados ou reprovaram com uma nota mais alta. O índice de reprovações com notas baixas é muito pequeno. Esse mesmo tipo de análise pode ser feito para qualquer disciplina da base filtrada, segundo os critérios da PRE, para se ter uma melhor visão do comportamento da reprovação em uma disciplina específica.

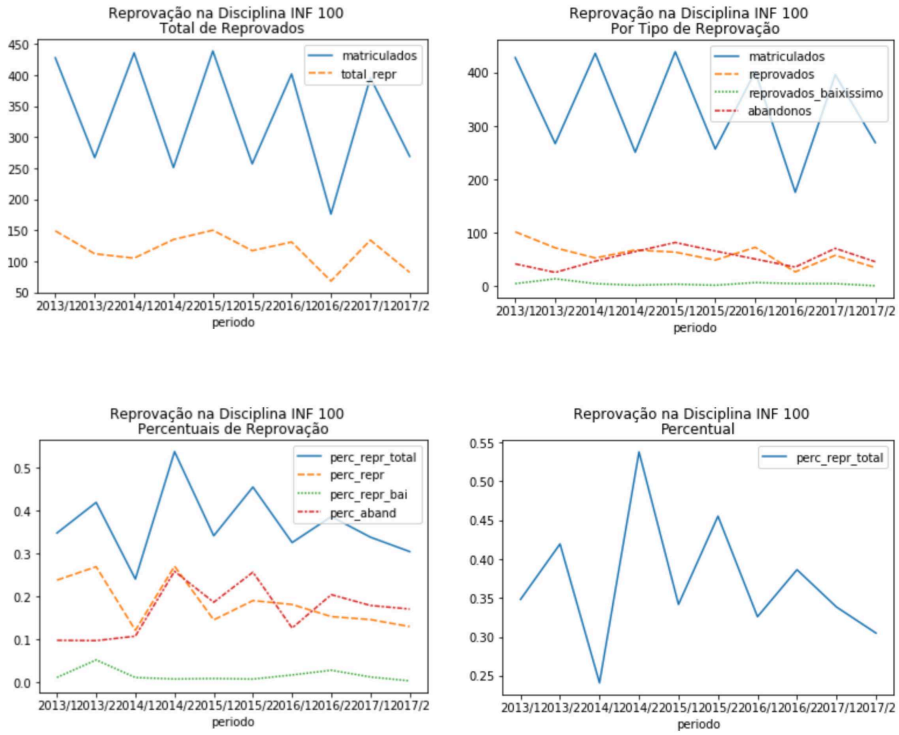

Figura 5.7: Análise detalhada da disciplina INF100. Os gráficos mostram a reprovação por vários aspectos diferentes de modo a garantir um entendimento melhor sobre os índices na disciplina.

No primeiro gráfico comparativo, é possível notar a sazonalidade do número de matriculados: é um número que varia de acordo com o semestre do ano, visto que no primeiro semestre entram mais calouros e a oferta precisa ser maior já que eles são o

público predominante da disciplina. Em períodos anteriores a 2016, porém, percebe-se uma tendência de estabilidade no número de reprovações e não uma tendência que acompanha o número de matriculados, o que se vê apenas nos períodos mais recentes.

Ao segmentar os tipos de reprovação, é notável que a maioria dos reprovados se enquadra no perfil de abandono de disciplina, pois a curva acompanha, quase em todo o tempo, as reprovações totais. Poucos alunos que não desistem ficam com notas muito baixas. Esse é um resultado importante que pode ser mostrado para os alunos matriculados no começo do período de forma a incentivá-los a não desistir da matéria. Por ser um conteúdo totalmente novo, que muitos não tiveram contato no ensino médio, já que no Brasil, programação não é ensinada na educação básica, como currículo obrigatório, há naturalmente uma adversidade ao novo.

Ao levarmos em conta o percentual de reprovação, percebemos que ele sobe muito para aqueles períodos nos quais o número de vagas é menor, mas, os tamanhos desses picos vêm diminuindo ao longo do tempo. Ao segmentar novamente por tipo de reprovação, reafirmamos que a desistência é muito presente dentre os alunos de INF 100. Menos de 5% reprovam com nota baixíssima, ou seja, menos que 10 pontos. Por volta de 20% reprova com nota até 40 e os outros 20% abandonam.

Essa Seção mostrou os principais resultados obtidos com uma análise exploratória dos dados. A Seção seguinte irá descrever os resultados alcançados com a mineração de padrões frequentes.

5.2 Padrões frequentes e regras de associação

Na tentativa de identificar o que acontece em conjunto com as reprovações, usamos o algoritmo de extração de padrões frequentes e encontramos 47 regras associadas às reprovações. As principais regras são mostradas na Figura 5.8. Consideramos conveniente mostrar as regras no formato de grafo, onde cada nó representa um elemento da transação, descrita na Figura 4.3. Esse elemento pode ser uma disciplina cursada quando houve reprovação ou aprovação, o professor que ministrou a turma prática ou teórica de uma disciplina, o número de créditos cursados no período, entre outros.

Os nós em vermelho representam disciplinas nas quais o aluno reprovou ao fim do semestre e os demais nós são coloridos de rosa. Como nosso foco é a reprovação, a ideia foi destacá-los usando uma cor vibrante. As arestas definem uma regra e são direcionadas: partem de um ou mais antecedentes e terminam em um único consequente. Como são mostradas várias regras juntas no grafo, a cor das arestas diferencia cada conjunto de regras.

Além disso, um certo nó pode receber várias arestas, o que significa que várias regras levam àquele consequente, como acontece com a reprovação em MAT-146 na Figura 5.8. Percebe-se que quatro regras resultam na reprovação em MAT-146, já que as arestas que chegam a esse nó possuem quatro cores distintas. Duas dessas regras são compostas por dois antecedentes, como é o caso da regra representada pela cor rosa: BIO 131 em conjunto com uma reprovação em QUI 100, geralmente levam à uma reprovação em MAT 146.

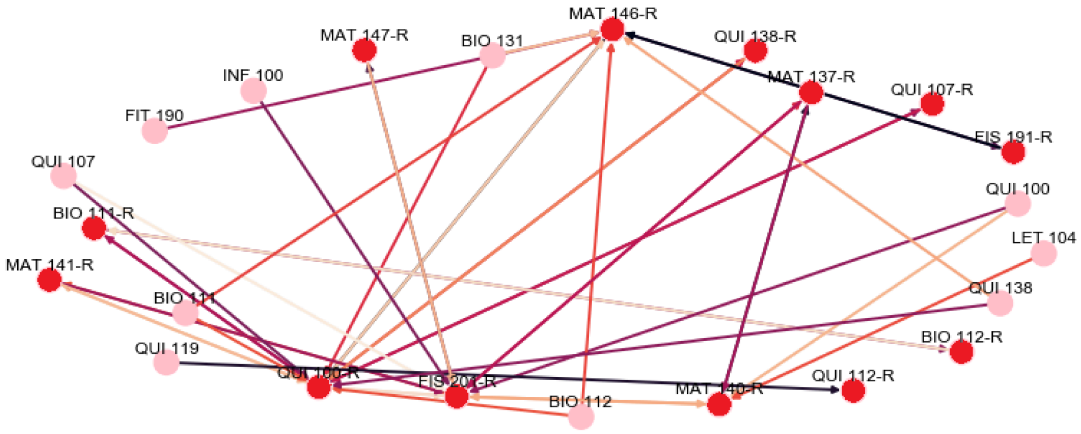

Figura 5.8: Principais regras encontradas na base toda sem agrupar representada por um grafo direcionado, no qual as arestas partem dos antecedentes e vão em direção aos consequentes.

Além de serem regras fortes, elas confirmaram tendências já observadas, como, por exemplo, alunos que cursam mais disciplinas por semestre possuem uma maior probabilidade de reprovação naquele semestre.

Nota-se que poucas regras foram encontradas e dentre elas, a maioria das disciplinas presentes pertencem aos departamentos do Centro de Ciências Exatas (CCE). Muitos estudantes possuem deficiência nesse tipo de conteúdo, que é a base para muitos cursos superiores. É possível perceber que FIS 201, QUI 100 e MAT 146 lideram o ranking de ocorrências, já que muitos antecedentes podem levar a esses consequentes, ou seja, uma reprovação nessas disciplinas. Essas são disciplinas cursadas por alunos de diferentes cursos e se mostrou como um padrão geral da UFV, logo, são situações críticas que precisam de atenção.

Algumas regras são compostas por aprovação em uma disciplina no antecedente e reprovação no consequente. Para esses casos, isso pode indicar que os estudantes, quando cursam duas disciplinas juntas, optam por focar em uma delas e abandonam a outra. Já no caso em que tanto o antecedente da regra quanto o consequente são reprovações, é um indício de que as duas disciplinas cursadas juntas é uma combinação de risco. Entre os vários fatores que podem estar relacionados a isso, existe a possibilidade de o conjunto de disciplinas ser de conteúdo maçante, o que exige muito esforço do aluno. Ou ainda, podem se tratar de conteúdos similares no qual o estudante possui dificuldade.

A fim de verificar se as regras se tornavam mais específicas em grupos de disciplinas semelhantes, a próxima análise foi então, clusterizar essas disciplinas e novamente executar a extração de padrões frequentes para cada grupo.

Como mencionado no Capítulo 4, foi utilizado como algoritmo de clusterização K-means. E como já tratado no Capítulo 3, existem duas formas de se encontrar o melhor valor de k, que corresponde ao número de grupos, para esse algoritmo. Uma delas é o método do cotovelo que a partir de várias execuções do K-means, uma para cada valor de k dentro de um certo intervalo predefinido, calcula-se a soma dos quadrados intraclusters (*Within-Clusters Sum-of-Squares* - WCSC). A ideia é que esse valor seja o menor possível para garantir que criamos grupos homogêneos, mas distintos uns dos outros.

A partir do gráfico de execuções gerado (Figura 5.9), faz-se uma análise que pode

ser tanto visual quanto matemática a fim de encontrar o k que representa o ponto da curva que destaca o fim da queda brusca e começo de uma queda mais suave nos valores de WCSC, ou seja, o cotovelo do gráfico.

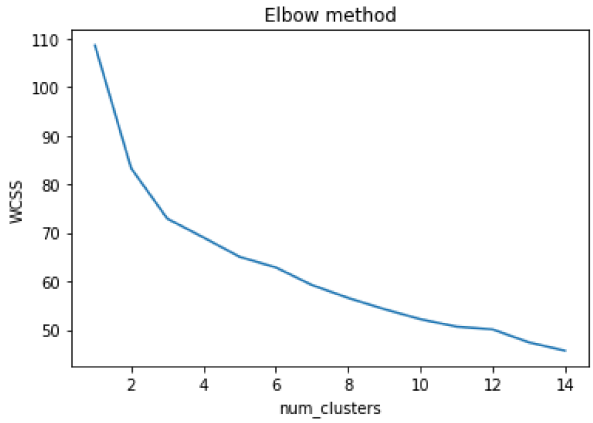

Figura 5.9: Resultado da aplicação do método do cotovelo.

A curva gerada pelo método do cotovelo nesse caso, foi bem sutil. Não fica tão expressivo o ponto que define o melhor k, mas é possível perceber que ele está em torno de 3 a 5. Assim, executamos também a análise de silhueta para garantir uma melhor escolha.

A segunda forma de se encontrar o melhor k é chamada de coeficiente de silhueta, na qual novamente várias execuções do K-means são executadas para múltiplos valores de k. Nesse caso, são calculadas para cada instância, a distância média intra-cluster (a qual chamaremos de a) e a distância média para o cluster mais próximo (tratada por b)(Pedregosa et al. (2011)). O coeficiente é calculado como $(b-a)/max(a,b)$.

Depois de calculado o coeficiente para cada instância é possível exibir seus valores em uma treliça de barras finas, segmentadas por *cluster* com cores distintas. Isso cria um perfil ou uma silhueta dos grupos montados a cada iteração, o que dá nome ao método. A ideia então é selecionar um valor de k tal que o coeficiente médio seja o maior possível assim como a distribuição dos itens por grupo, evitando ao máximo coeficientes negativos.

Os valores de coeficientes podem variar entre -1 e 1. Quanto mais próximo ele estiver de 1, significa que o item está mais distante dos seus vizinhos de grupo. O valor 0 indica que a instância está próxima da fronteira de decisão entre dois grupos. Enquanto que valores negativos são indicativos de que o item foi alocado no grupo errado. (Pedregosa et al. (2011)) A Figura 5.10 mostra os coeficientes de silhueta gerados para seis execuções do K-means com o k variando de 2 a 6.

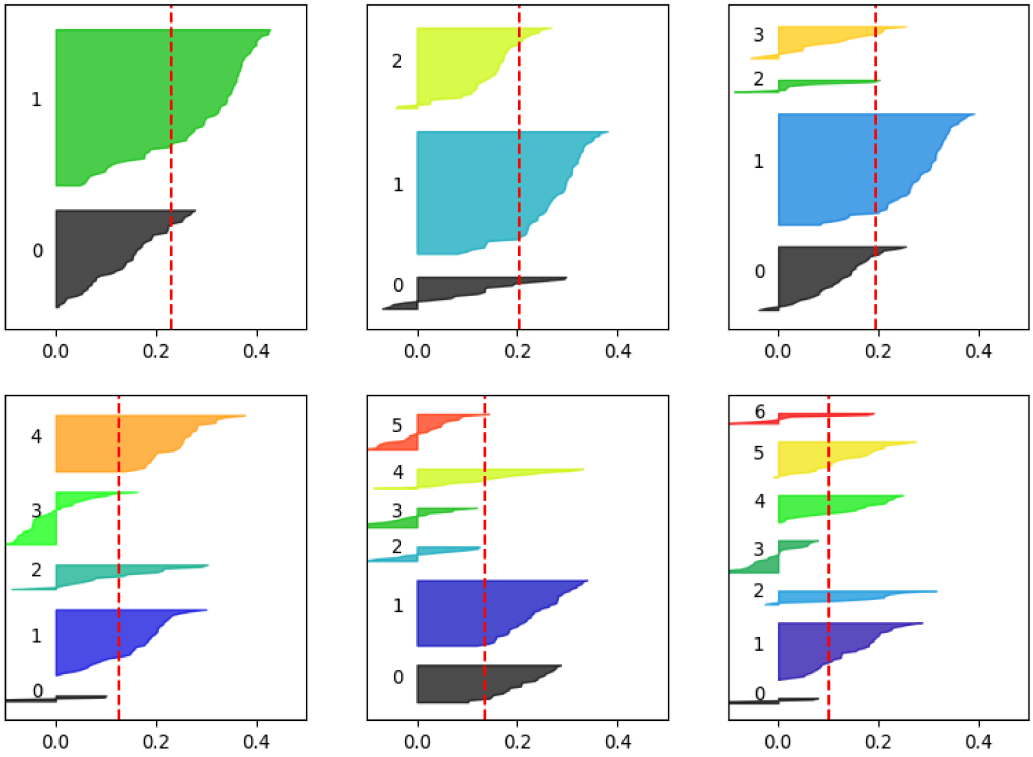

Figura 5.10: Resultado da aplicação da análise de silhueta.

A reta vermelha tracejada representa o coeficiente de silhueta médio entre todas as instâncias. Segundo de Souza (2007), o melhor número de *clusters* é dado pelo conjunto que possui o maior coeficiente médio. Nota-se então que, a partir de 5 grupos, o coeficiente fica abaixo de 0,2. As outras separações alcançaram um valor de coeficiente maior. Dessa forma, optamos pela divisão em 4 grupos, cujo coeficiente atingiu 0,2 e conseguiu diversificar melhor os grupos, diminuindo a largura do grupo predominante e redistribuindo-a entre os outros.

Uma vez escolhido o valor de k, temos o resultado dos grupos retornado pelo algoritmo K-means. Um grupo é formado por várias instâncias, mas pode ser descrito por um elemento central para que tenhamos uma visão descritiva aproximada de cada grupo. Esses elementos são chamados de centroides e podem ser calculados a partir da média ou da mediana dos atributos das instâncias pertencentes a um grupo. A Figura 5.11 mostra os centroides de cada um dos grupos criados.

Os centroides foram representados por gráficos de radar, onde cada atributo do grupo é definido como um eixo do gráfico. O centroide é o ponto central do grupo e seus valores são demarcados em cada eixo sendo, quanto mais próximo do centro menor o valor para aquele atributo. Depois de demarcados todos os pontos, eles são conectados de forma a construir um perfil que representa aquele grupo. Os valores são normalizados para que os eixos possam ser divididos igualmente e forme um círculo. Essa característica circular que permite a visualização de múltiplos eixos radiais de uma só vez é o que dá nome ao gráfico.

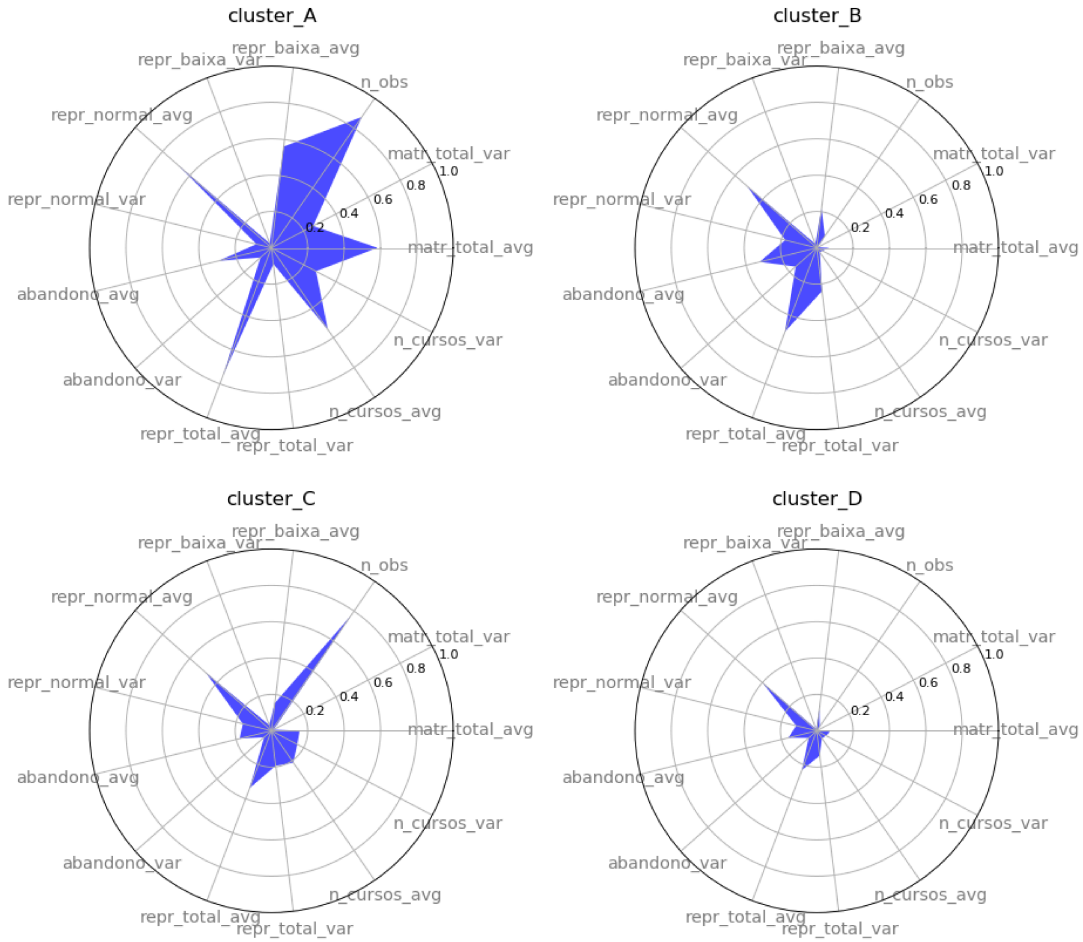

Figura 5.11: Forma dos centroides dos clusters encontrados.

Percebe-se, pelas suas diferentes formas, que existem perfis distintos de disciplinas, então faz sentido procurar por padrões frequentes em cada grupo separadamente. Os atributos considerados na clusterização, foram avaliados tanto em relação às suas médias (sufixo _avg), quanto às suas variâncias (sufixo _var), com exceção do número de observações (n_obs). Portanto, para entender o perfil de cada *cluster*, basta olhar para essas duas medidas estatísticas de cada atributo. Por exemplo, em relação aos outros grupos, o *cluster* A é aquele que possui maior média de matriculados. Além disso, a baixa variância desse atributo indica que, na maioria das vezes, o número de matriculados fica próximo da média, ou seja, é um número alto sempre.

Pela Figura 5.11, é válido tentar entender os agrupamentos gerados a partir da análise do formato dos centroides com o interesse de identificar quais características definem cada grupo de disciplinas. Pode-se notar que o agrupamento A é formado por disciplina cursadas por um grande número de cursos, em média. Entretanto, esse número varia de um semestre para o outro. Novamente, segundo a média, são muitas pessoas matriculadas e uma boa parte delas sempre reprova, sem distinção por tipo de reprovação. Disciplinas com essas características na UFV são as obrigatórias de massa.

Sobre as disciplinas do *cluster* B, pode-se dizer que não são muitos cursos que as possuem na grade e também não são muitos alunos que se matriculam nelas. Porém, o número de reprovações normais e os abandonos são altos. Essas podem

ser consideradas as disciplinas específicas optativas. No *cluster* C, encontram-se as disciplinas que ainda possuem um índice alto de reprovação, mas não sofrem tanto com o abandono e são poucos cursos que fazem. Esses pontos podem descrever as disciplinas obrigatórias específicas. Por fim, no grupo D, estão aquelas disciplinas que não são tão alarmantes em relação à reprovação, pois possuem médias baixas em quase todos os atributos.

Depois de clusterizada em relação ao perfil de disciplinas foram construídas 41.378 transações sobre a base original, no processo de montagem que será descrito na Seção 6.1. Tais transações serviram de entrada para o algoritmo de regras de associação, abordado na Seção 3.1. Com a extração de padrões frequentes executada para cada *cluster* a um suporte mínimo de 3%, foram encontradas, no total, 1.020 regras de associação, com confiança de no mínimo 70%. Os valores de suporte e confiança foram definidos empiricamente. Em alguns *clusters*, o número de regras foi muito pequeno, enquanto em outros o número foi bem maior, como podemos notar pela tabela da Figura 5.12.

Cluster	Regras
A	2
B	966
C	1
D	51

Figura 5.12: Número de regras encontradas em cada *cluster*.

Ao compararmos as principais regras encontradas com uma confiança maior que 70% na base sem clusterizar (Figura 5.8) e na base clusterizada (Figura 5.13), percebe-se que o número de padrões encontrados cresceu razoavelmente ao se especificar melhor o contexto. Começam a aparecer disciplinas relacionadas a outros centros de ciências e não somente do CCE e surgiram muitas regras relacionadas a professores que ministram as disciplinas. Com relação a essas regras relacionadas a professores, cabe à PRE ou outros órgãos analisarem a situação e tomarem as providências cabíveis. Também foram encontradas regras relacionadas a carga horária e curso.

Dos 24 nós vermelhos do grafo da Figura 5.13, ou seja, das 24 reprovações, 6 delas são nomes de disciplinas que ocorrem no antecedente e representam conjuntos de disciplinas arriscados para se cursar em conjunto. Esse tipo de regra, quando aparecer nos alertas para o estudante, deve ser evitada ao máximo. Regras que aparecem nos alertas devem ser evitadas pois representam um risco alto visto que o suporte de corte considerado foi de 0,03, ou seja, de 41378 transações montadas, elas representam 1241, aproximadamente. Trata-se de um universo pequeno e, por isso, a chance de aparecer algum alerta é baixa. Contudo, se a regra aparece, ela realmente deve ser analisada com cautela. É recomendado que o aluno tente evitar, ao máximo, sua ocorrência, como medida de precaução, com o propósito de se livrar da possibilidade de reprovação.

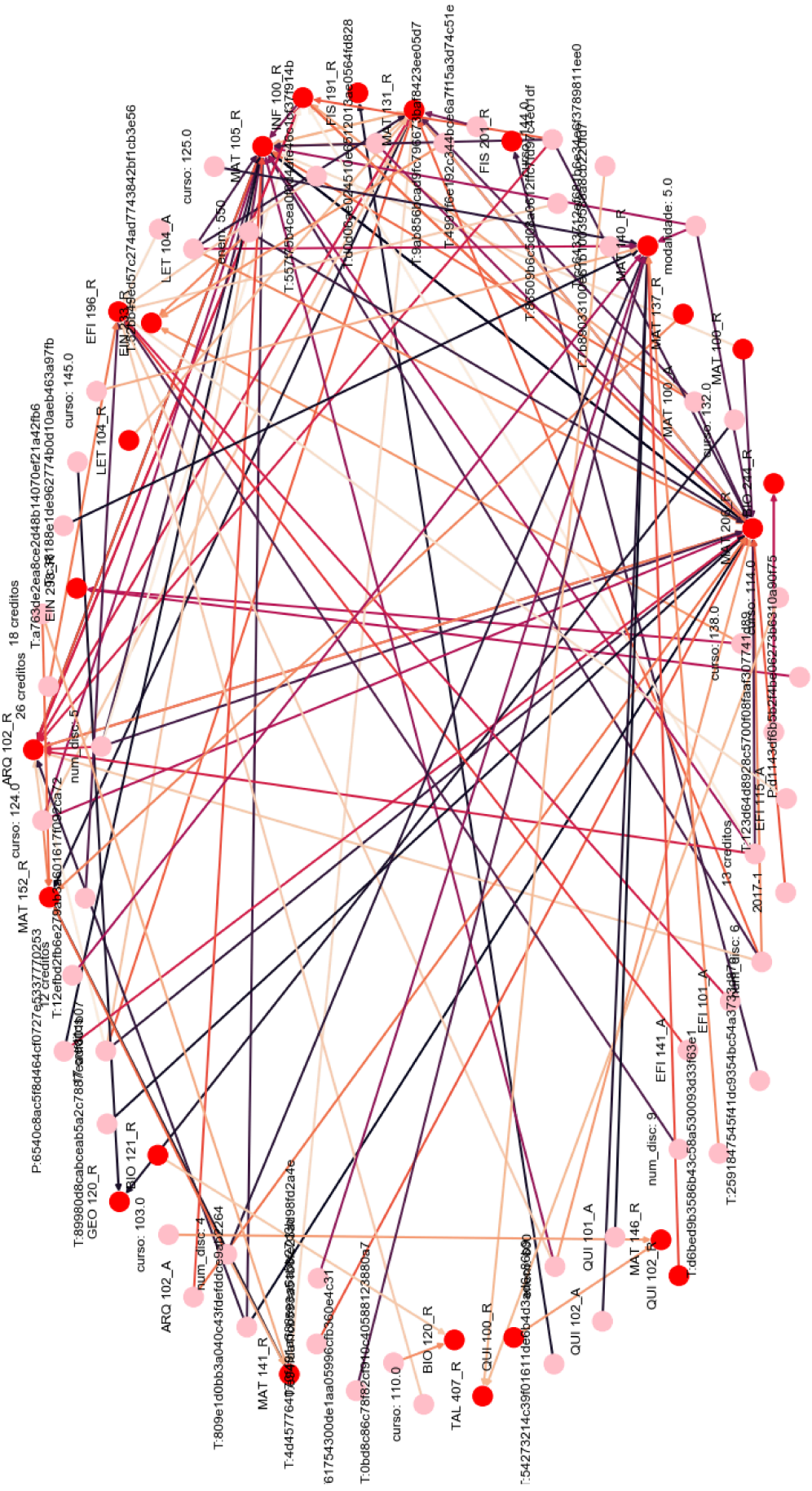

Figura 5.13: Principais regras encontradas na base clusterizada.

Um estudo foi realizado especificando o contexto da computação, apenas com transações montadas sobre os registros de alunos do curso de Ciência da Computação e outro sobre aqueles estudantes que cursam disciplinas no Departamento de Informática. Os resultados desses estudos estão publicados em Gomes et al. (2020). Também foi feita uma análise específica sobre os calouros, ou seja, estudantes que estão em seu primeiro ano de faculdade.

Para as transações de calouros foram encontradas 153 regras enquanto que no conjunto do curso de Ciência da Computação foram encontradas 36 regras, ambas com confiança acima de 98%, que são resultados consideravelmente mais fortes. A maioria das regras do primeiro conjunto está relacionada à reprovação em duas disciplinas específicas: Fundamentos de Matemática Elementar (MAT105) e Física II (FIS202), denotando problemas relacionados aos temas básicos, uma vez que regras envolvendo outros cursos introdutórios também puderam ser encontradas.

Já no segundo, um resultado notável é o comportamento de Introdução à Álgebra (MAT131). Sua porcentagem média geral de reprovação é de cerca de 69%. Quando um aluno está cursando 5 ou mais disciplinas simultanêas, as chances chegam a 89%, e quando um determinado professor leciona a matéria, ela chega a 98% de chance de reprovação. É um resultado alarmante, uma vez que MAT131 é um conteúdo extremamente importante para o currículo básico de Ciência da Computação. Esses resultados podem ser vistos nas Figuras 5.14 e 5.15.

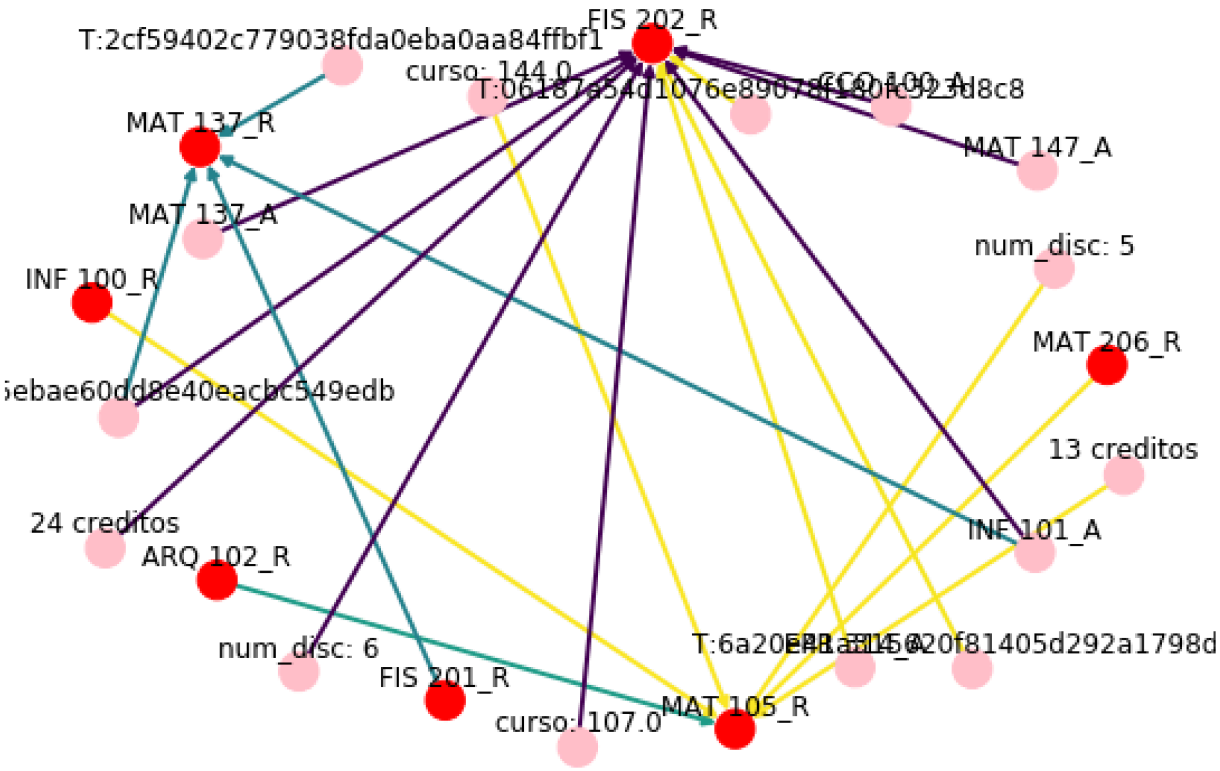

Figura 5.14: Principais regras encontradas para o experimento com disciplinas do Departamento de Informática.

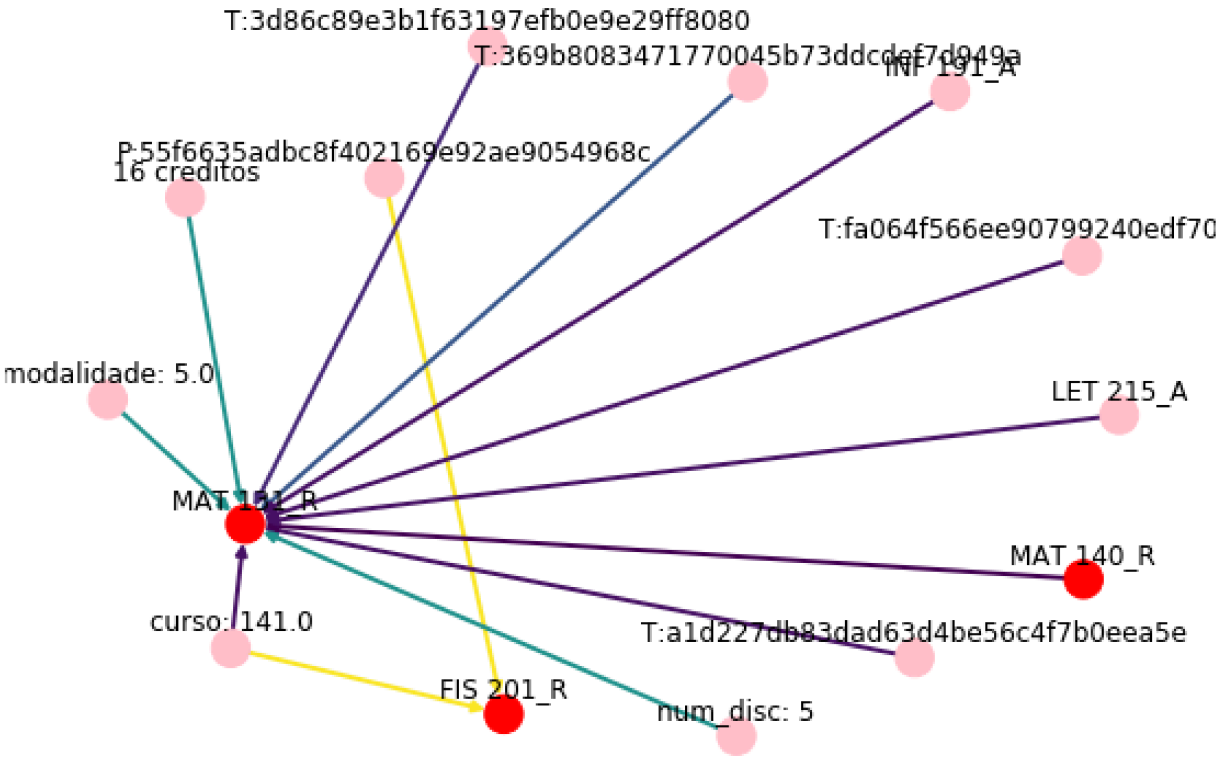

Figura 5.15: Principais regras encontradas para o experimento com alunos da Ciência da Computação.

5.3 Modelo preditivo

Da análise exploratória realizada, foi possível notar que algumas variáveis estão diretamente ligadas à reprovação. Tais variáveis foram incorporadas ao modelo preditivo preliminar, descrito no Capítulo 4 que visava classificar se houve reprovação ou não. A matriz de confusão gerada a partir desse modelo é mostrada na Figura 5.16. O modelo classificatório acertou 67% das instâncias positivas, ou seja, para essas instâncias, ele previu que haveria reprovação, o que realmente aconteceu. Também previu corretamente 60% das instâncias negativas, isto é, onde não houve reprovação. Isso é um bom indício de que um modelo preditivo mais elaborado pode prever com melhor acurácia se haverá reprovação ou não.

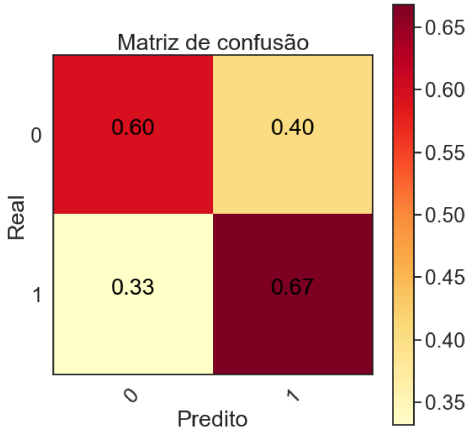

Figura 5.16: Matriz de confusão para o modelo preliminar usando Random Forest.

Na tentativa de construir um modelo preditivo, exploramos as correlações entre os atributos da base que podem ser vistos na Figura 5.17.

As correlações entre *features* são representadas por um mapa de calor no qual em cada eixo são dispostas a mesma lista de atributos. Quanto maior for a correlação entre duas variáveis, mais claro será o quadrado no mapa, nesse caso e quanto menor, mais escuro. Como cada *features* é confrontada com todas as outras e a lista é disposta na mesma ordem nos dois eixos, a diagonal principal representa a relação da variável com ela mesma e, portanto, a correlação é máxima, ou seja, ela vale 1. Os valores de correlação podem variar de -1 a 1. O intuito do mapa de calor é destacar pontos de alta ou baixa correlação a partir das cores extremas que saltam aos olhos.

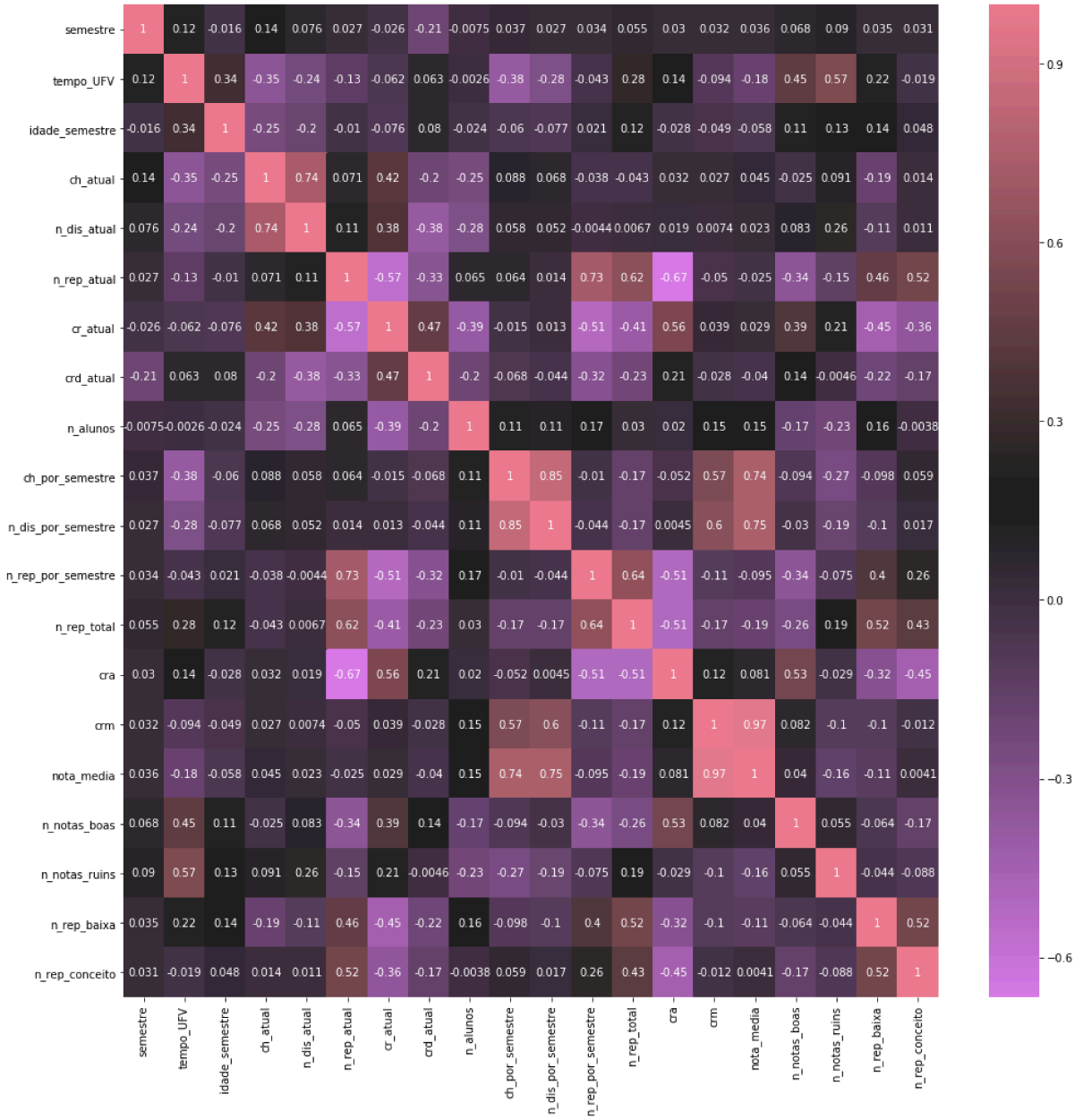

Figura 5.17: Mapa de calor da correlação entre atributos e variável alvo. Quanto mais próximo for da cor preta, menor a correlação. O rosa indica uma correlação positiva e o roxo uma correlação negativa.

Através do gráfico, é possível concluir que poucas variáveis são altamente correlacionadas, a maioria possui baixa correlação entre si. Porém, pode-se perceber algumas correlações que se destacam no mapa tais como número de reprovações com CRA e com CR atual ou carga horária por semestre com número de disciplinas por semestre, ou ainda, nota média em disciplinas com CRM.

Nos dois primeiro a correlação é negativa e faz todo o sentido já que quanto maior o número de reprovações, mais será impactado negativamente o coeficiente de rendimento acumulado do estudante. E se o estudante obteve um CR alto no período que acabou a chance de ele reprovar no próximo semestre é baixa.

As duas últimas claramente são correlações positivas já que número de disciplinas e carga horária estão diretamente relacionadas e a nota média alcançada nas disci-

plinas do período irão impactar diretamente o CRM. Quando um aumenta, o outro também irá aumentar. Apesar de baixas as correlações, ao incluir cada uma dessas *features* no modelo, elas podem, juntas, contribuir para uma boa predição.

Foram executados 2 modelos preliminares com Floresta Aleatória (*Random Forest*) e Perceptron multicamadas (*Multilayer Perceptron*) com um MSE de 0,052 e 0,048, respectivamente. Optamos por abandonar uma terceira abordagem com Máquinas vetor de suporte (*SVM*) devido ao grande tempo gasto na execução do modelo. Ele demorou cerca de 156 vezes mais que os outros dois, enquanto os dois primeiros executaram em aproximadamente 5 minutos cada um, o SVM terminou sua execução depois de 13 horas rodando. Assim, partimos para uma outra alternativa: a construção de uma rede neural.

O processo para a criação de um modelo de rede neural foi dividido em duas etapas: parametrização e modelos combinados. Na primeira etapa, testou-se separadamente alguns hiperparâmetros para identificar seu potencial de contribuição para a rede e na segunda, foram construídos modelos combinando os parâmetros que melhor performaram.

Podemos ver nas figuras 5.18 e 5.19 o KDE dos erros de predição de cada fator isoladamente e da rede inicial, usada como base de comparação. Percebe-se que quanto maior a largura da curva, maiores são os valores que um erro de previsão pode assumir. O erro é definido como o valor real menos o previsto. A altura da curva indica a quantidade de registros que obtiveram aquele valor de erro. Portanto, nesse contexto, é melhor ter uma curva estreita e alta do que larga e baixa, o que quer dizer que a maioria das instâncias possuem erros bem pequenos, próximos de 0.

Como opção de função de ativação, foram testadas *Sigmoid* e *Leaky ReLU*. Comparando as estimativas de densidade kernel (*Kernel Density Estimation* - KDE, Xie and Yan (2008)) gerado no teste, das duas funções anteriores com a rede base, é possível verificar que ambas comprimiram a curva (Figura 5.18), apesar da Leaky ReLU ter sido um pouco mais significativa.

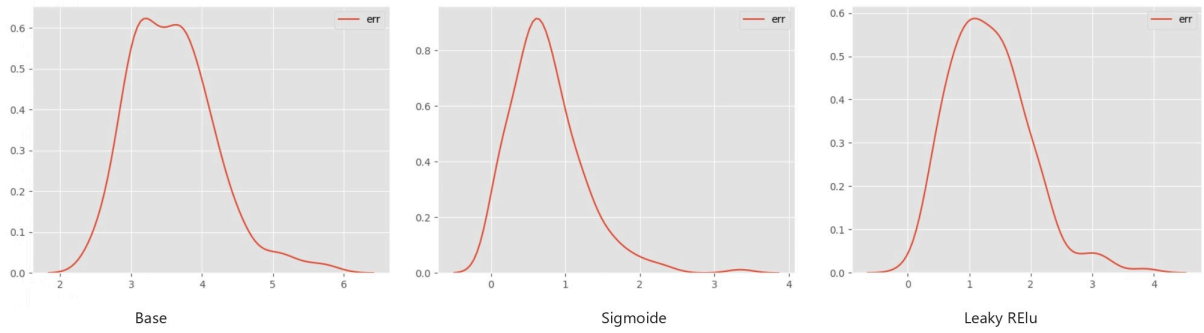

Figura 5.18: KDEs das funções de ativação testadas.

A compressão da curva significa que o modelo está errando para um número menor de instâncias. Outro parâmetro analisado foi o método de adaptação da taxa de aprendizagem. A Figura 5.18 mostra os métodos testados e as KDEs geradas, em contraste com a rede base, que foi o ponto de partida. É notável o comportamento similar da atualização passo a passo e linear.

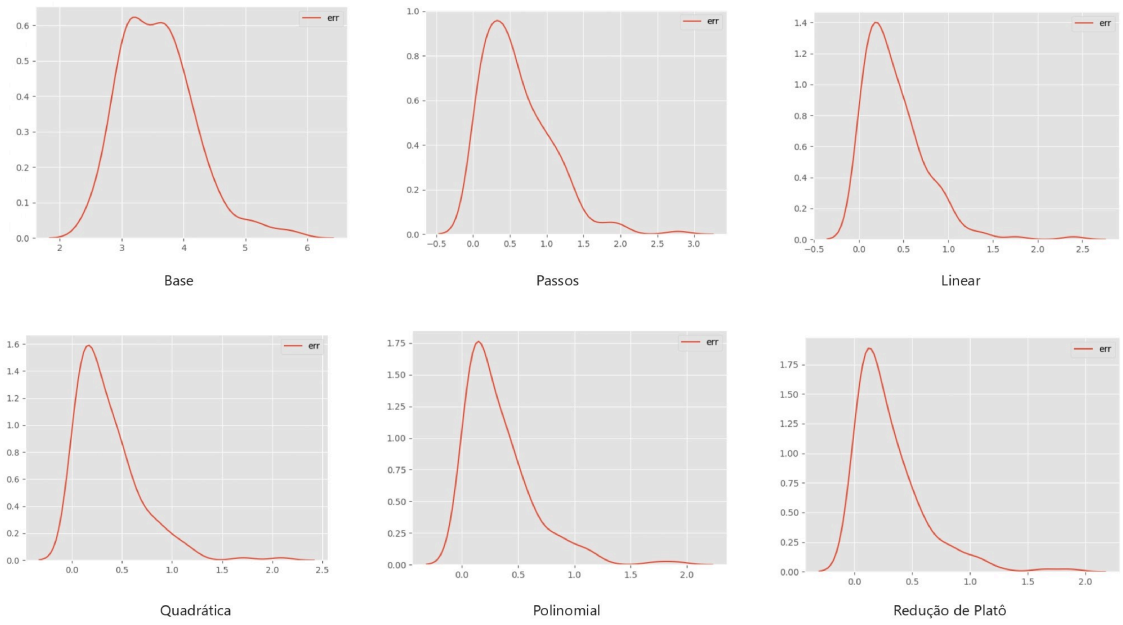

Figura 5.19: KDEs das formas testadas para atualização da taxa de aprendizagem.

Também é possível perceber que as atualizações quadrática, polinomial e redução de platô se comportaram de forma semelhante, porém comprimindo melhor a curva que as duas anteriores, se comparado com a rede inicial. Portanto, foram escolhidas as últimas três opções para a montagem dos modelos combinados.

O modelo base foi feito, utilizando o SGD, mas também foi testado outro otimizador, o Adam. O gráfico das Figuras 5.20 e 5.21 mostram um recorte dos itens preditos (em vermelho) e reais (em azul) e é possível perceber que o Adam possui uma cobertura maior que o SGD, ou seja, as previsões feitas com Adam chegaram bem mais próximas dos valores reais. Durante o treinamento e teste de cada modelo, foram coletadas algumas métricas de avaliação.

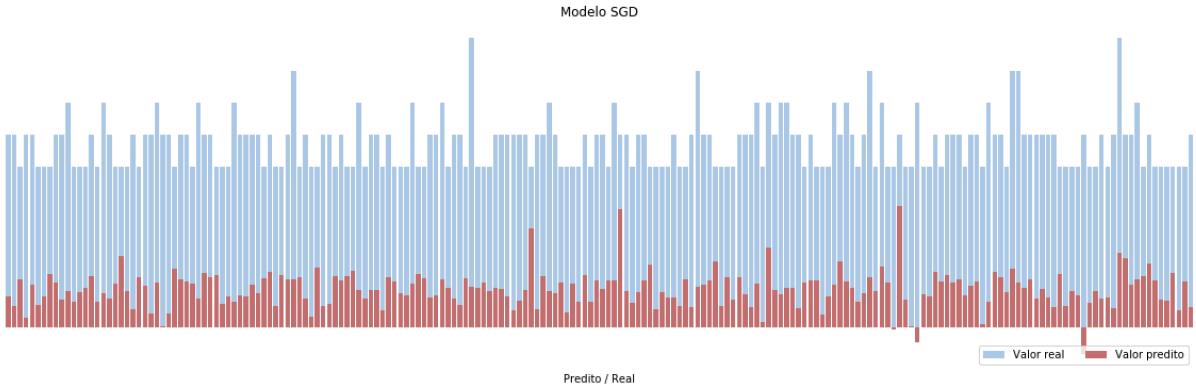

Figura 5.20: Gráfico de cobertura do modelo com SGD. As barras vermelhas representam as predições e as azuis, os valores reais.

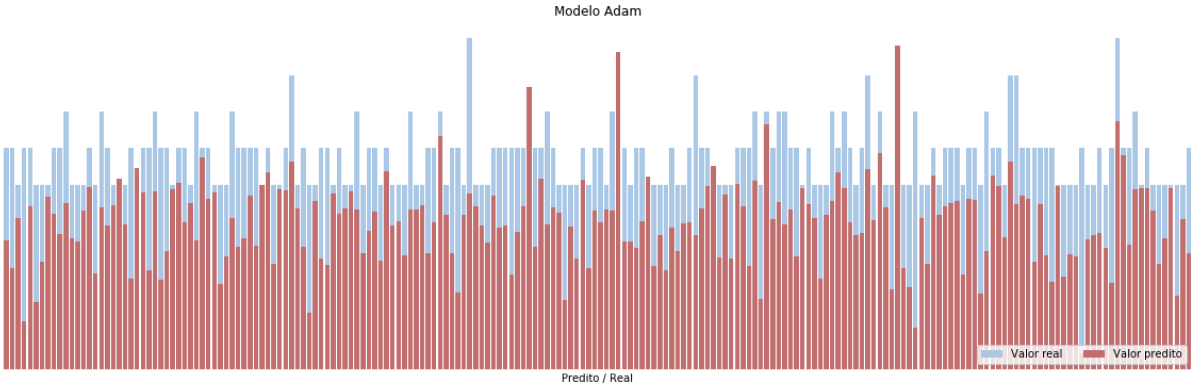

Figura 5.21: Gráfico de cobertura do modelo com o otimizador Adam.

Nas Figuras 5.20 e 5.21, é possível perceber que o Adam possui uma cobertura maior que o SGD, ou seja, as previsões feitas com Adam chegaram bem mais próximas dos valores reais, as alturas das barras vermelhas quase alcançam as azuis.

Para mensurar a performance do modelo, coletamos cinco métricas, três delas já existentes e propomos duas novas. O Erro quadrático médio (*Mean Squared Error* - *MSE*, Wasserman (2004)), o Erro médio absoluto (*Mean Absolute Error* - MAE, Wasserman (2004)) e a Variância explicada (*Explained Variance Score* - EVS, O'Grady (1982)). Para os erros, a meta é alcançar os menores valores possível, enquanto um EVS bom é aquele que se aproxima de 100%. As outras duas foram propostas pelos autores, são métricas auxiliares mais interpretáveis. A primeira é definida como Contagem de erros baixos (*Small Errors Count* - SEC), mostrada na Equação 5.1.

$$SEC = \frac{1}{n}\sum_{i=1}^{n} 1, \forall(y_i - \widehat{y_i}) < 0.1 \tag{5.1}$$

Ela representa a fração de instâncias preditas com um erro menor que um limiar de 0,1 em relação ao valor real. A segunda é definida como Erro médio por item (*Average Error by Item* - AEI). Ela é a soma de todos os erros de previsão dividido pelo número total de itens, como mostra a Equação 5.2. É uma medida que tenta mostrar o sentido do erro, ou seja, se o modelo está, em média, superestimando ou subestimando as instâncias.

$$AEI = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y_i}) \tag{5.2}$$

Comparando as métricas calculadas no teste, obtivemos os gráficos da Figura 5.22, no qual é possível notar que as escolhas para atualização de taxa de aprendizagem foram bem definidas, já que demonstraram menores MAE, MSE e AEI, além de elevados SEC e EVS. A partir da nova rede base, chamada de Combinada, foram criados dois novos modelos (1 e 2). O primeiro explorando a atualização de taxa de aprendizagem de forma quadrática com uma camada mais extensa e o segundo com uma camada mais extensa e redução de platô. Com um treinamento de apenas 20 épocas, o modelo 1 já se mostrou promissor.

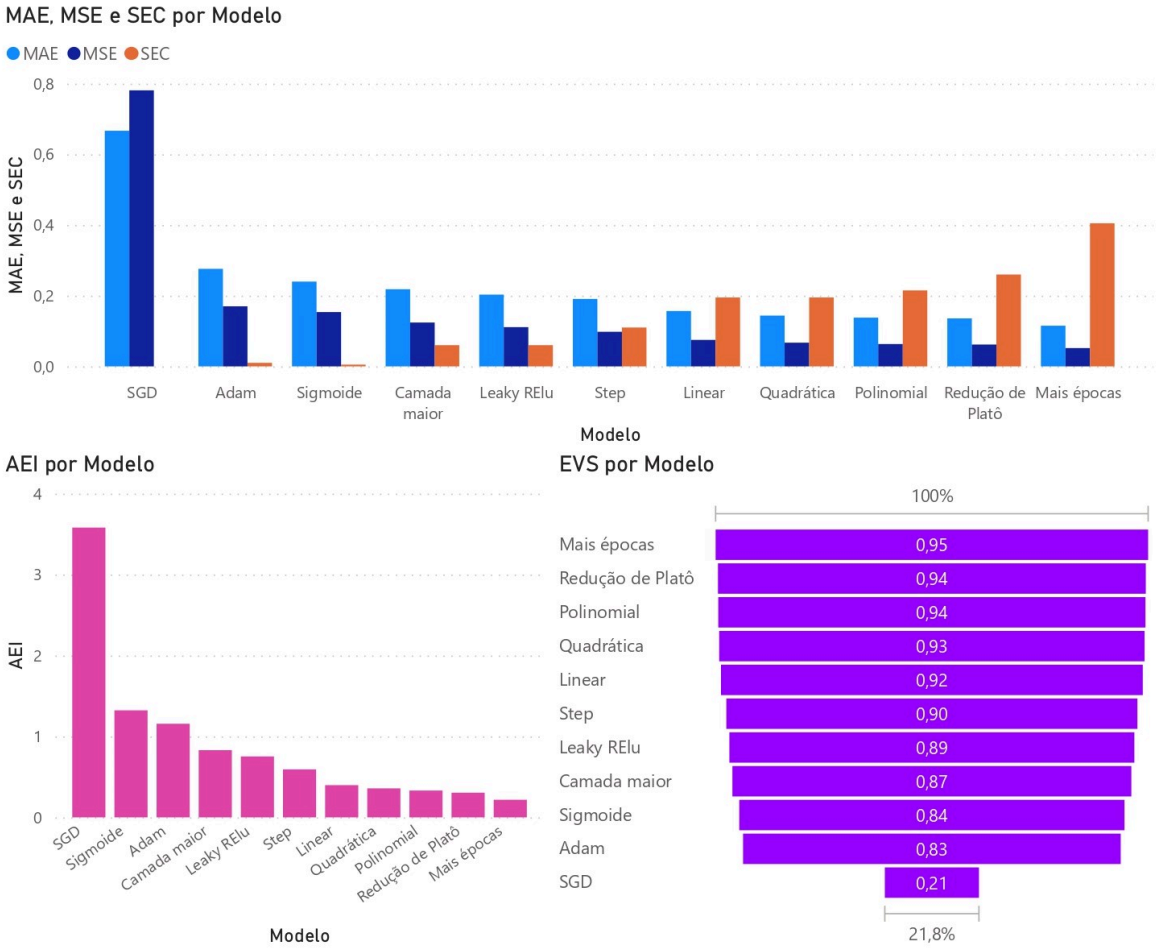

Figura 5.22: Desempenho das métricas calculadas por hiperparâmetro testado.

A Figura 5.22 mostra o resultado das métricas coletadas para cada parâmetro possível de inclusão na rede. Com relação aos erros médios, apenas a troca do otimizador, já reduz o MAE em 0,4, aproximadamente. Cada um dos parâmetros contribui isoladamente para a diminuição dos erros. A adição de mais épocas ao treino faz com que 40% dos erros estejam dentro de um limiar de 0,1, isto é, vale investir na inclusão dessa característica. As opções de atualização de taxa de aprendizagem se comportam de forma bem similar, garantindo um aumento de 73% na EVS da rede original, sendo as melhores abordagens, a RP e polinomial. Nos três gráficos da Figura 5.22, a progressão dos parâmetros aparece em uma ordem semelhante, com exceção do Adam que performa melhor em AEI.

A partir daí, com alguma variação de parâmetros, foram produzidos mais 4 modelos. A Figura 5.23 mostra as métricas calculadas para cada um dos modelos criados a partir da nova rede base. Percebe-se que, com relação aos erros médios (MAE e MSE), os modelos de número par foram piores que a rede base enquanto que os de número ímpar performaram melhor, já que obtiveram erros menores. Analisando o AEI e o SEC, todos eles conseguiram melhorar o modelo inicial de base, pois todos obtiveram um número maior de SEC e menor de AEI. Por fim, analisando o EVS, a Combinada novamente separa os modelos pares dos ímpares, sendo eles os que performaram melhor segundo essa métrica.

Figura 5.23: Desempenho das métricas calculadas por modelos.

Todos os modelos obtiveram uma porcentagem de erros pequenos (SEC) maior que a rede base e todas as médias de erro por item (AEI) foram menores (Figura 5.23), logo, é necessário observar outras métricas, apesar de ter uma indicação de que os modelos 1, 5 e 3 foram os melhores. Ao analisar o EVS, a rede base possui um valor intermediário e separa dois conjuntos de modelos. Os modelos de número ímpar superam seu valor e os de número par conseguiram explicar menos que o próprio modelo base. Dessa forma, eles podem ser descartados. Novamente, ao inspecionar os erros, a Combinada separa os dois conjuntos de modelos mantendo a classificação já observada anteriormente.

Outra abordagem para análise da performance dos modelos é vista na Figura 5.24. Ela mostra o erro das amostras em cada um dos modelos combinados. Aqueles que possuem uma amplitude maior, não performaram tão bem. Por isso, novamente confirma-se a observação de que os modelos ímpares foram melhores que os pares, já que os modelos 1, 3 e 5 apresentam menor amplitude.

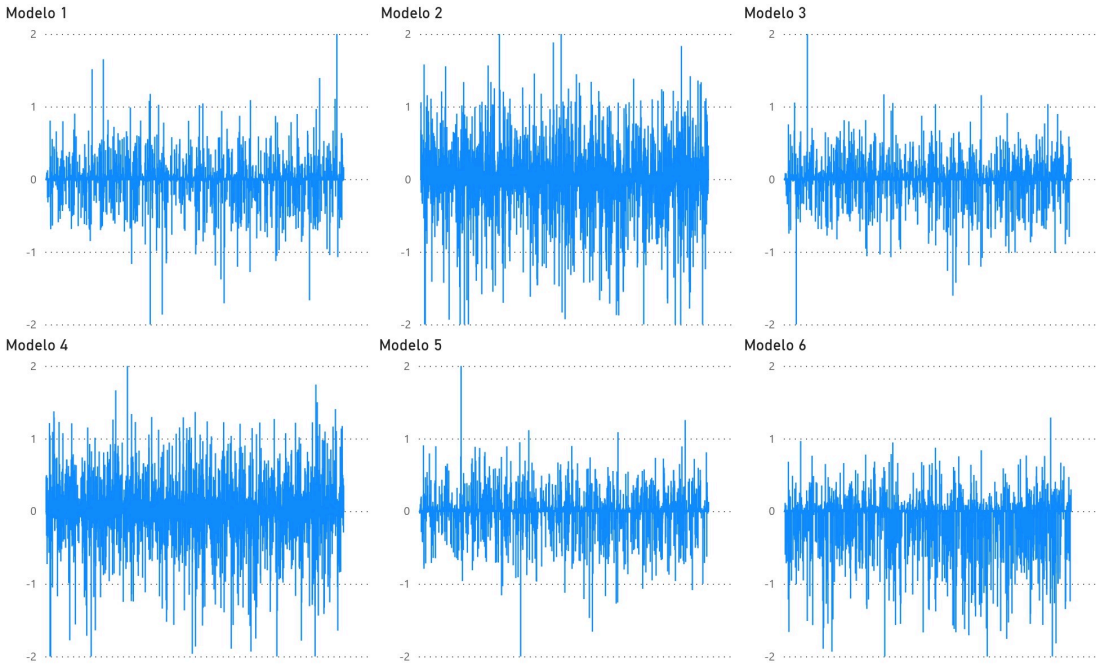

Figura 5.24: Gráficos de amplitude dos erros de predição por modelo. Se a barra está no zero, o erro para aquela instância foi nulo. Caso ele saia do eixo zero, pode representar um erro positivo ou negativo.

Dentre eles, é válido avaliar seus respectivos comportamentos ao longo do treino e validação através de um gráfico de função de perda ou *loss*. Ele revela como a função de custo está sendo atualizada a cada passo do período analisado. A curva começa com uma alta taxa de erro e vai decaindo conforme vai aprendendo até convergir para um valor estável de erro. De acordo com o gráfico de *loss* da Figura 5.25, é possível observar que o MSE do modelo 5 cai muito mais rápido que o dos outros dois, mas logo se estabiliza com eles. Na validação (imagem 5.26), a *loss* dos modelos 5 e 3 oscilam mais que a do 1. Entretanto, no fim, a do modelo 3 atinge um valor ligeiramente mais baixo que a dos outros dois.

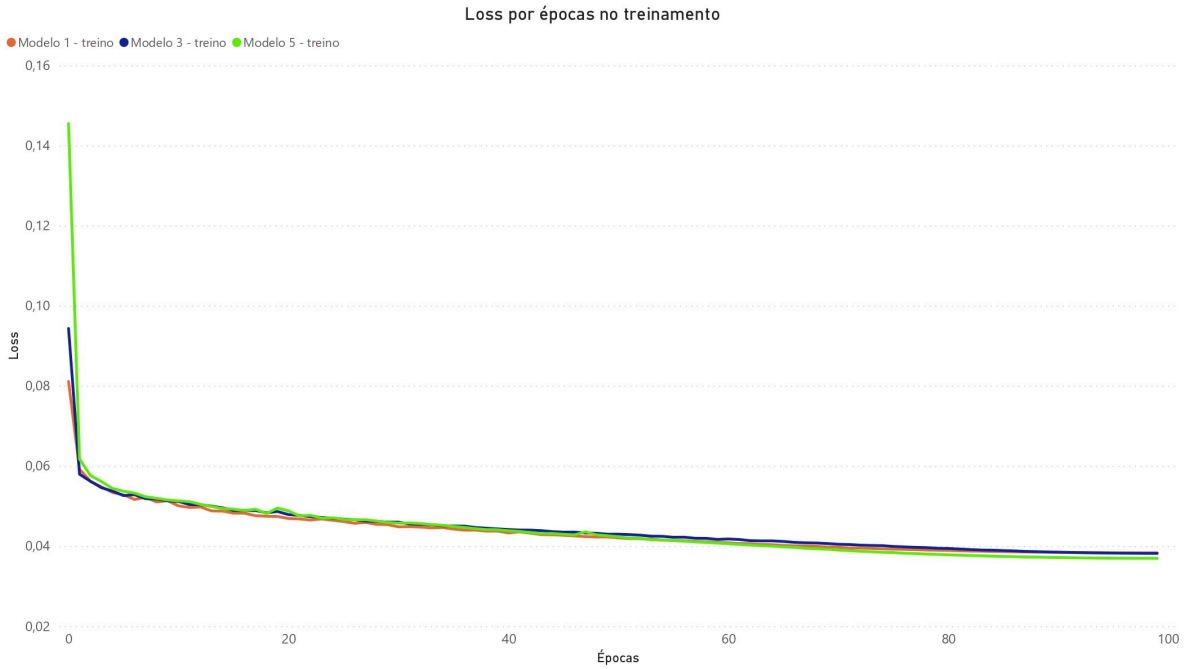

Figura 5.25: *Loss* no treino para os modelos ímpares.

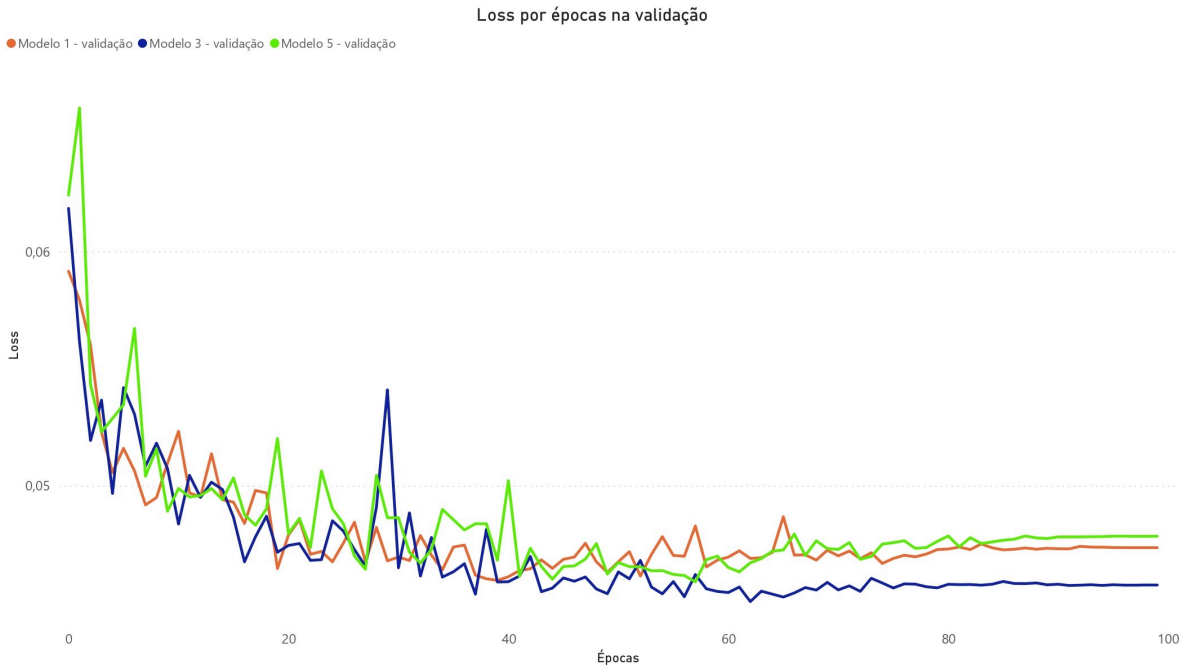

Figura 5.26: *Loss* na validação para os modelos ímpares.

O modelo 3 foi um pouco melhor nesse quesito, mas o modelo 1 se mostrou melhor nas outras métricas. Analisando os gráficos de cobertura das Figuras 5.27, 5.28 e 5.29, percebe-se novamente um comportamento similar, sendo os modelos 1 e 5 um pouco melhores que o 3. Percebemos que a cobertura foi bem significativa, alcançando valores bem próximos aos reais.

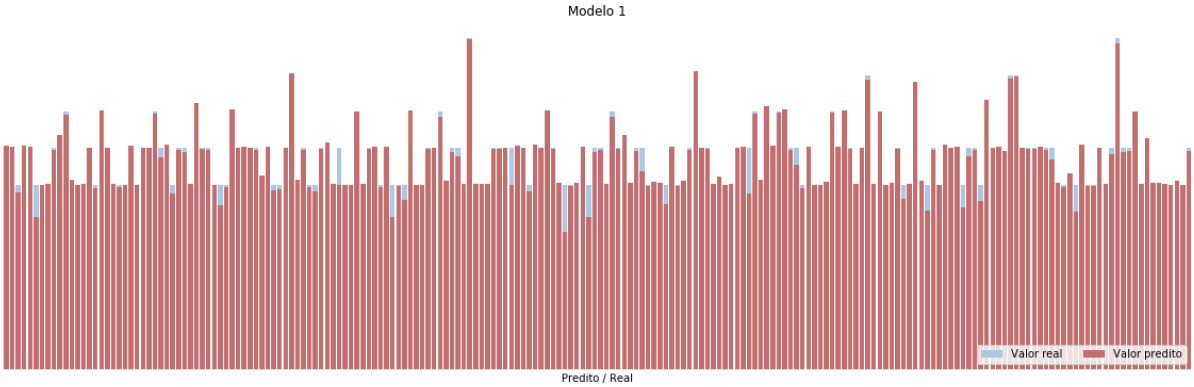

Figura 5.27: Gráfico de cobertura para o modelo 1.

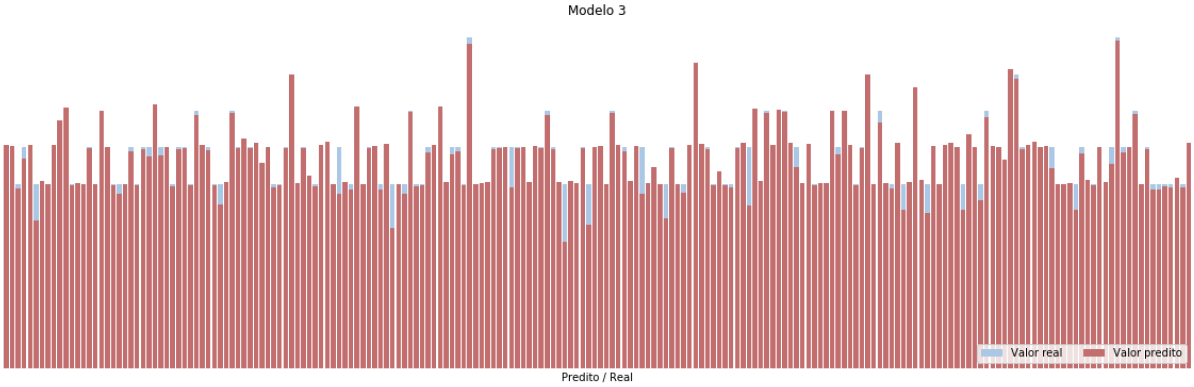

Figura 5.28: Gráfico de cobertura para o modelo 3.

Figura 5.29: Gráfico de cobertura para o modelo 5.

Qualquer um dos três modelos ímpares poderia ter sido escolhido já que a diferença de desempenho entre eles foi mínima, mas optamos pelo modelo 1 como modelo final.

Analisando melhor apenas o modelo 1, percebe-se pela Figura 5.27, que o erro entre o valor predito e o real é realmente irrisório. A cobertura foi bem significativa. Além disso, em 80% dos casos, o erro foi menor que 0,1, como podemos observar pela medida SEC.

Um erro de 0,1 significa dizer que para uma previsão de reprovação em 2 disciplinas, na verdade o valor real pode estar entre 1,9 ou 2,1. Porém como a saída do modelo é um provável número de disciplinas, optou-se por mostrar apenas o número inteiro correspondente, já que não é possível reprovar em uma parte da disciplina e ser aprovado em outra. Logo, após a predição, o resultado é arredondado e, nesse sentido, um erro de 0,1 não compromete o resultado final, visto que o número exibido continuaria sendo 2. Visto todos os aspectos discutidos, o modelo final é, portanto, uma combinação de fatores que melhor contribuem para uma previsão do número de disciplinas nas quais o estudante pode reprovar.

5.4 Caminho crítico e heurística gulosa

Uma vez tendo escolhido um modelo capaz de prever o número de reprovações em um dado semestre, o próximo passo, como mostrado no Capítulo 4, foi a implementação de um algoritmo de caminho crítico que pudesse mostrar ao aluno quais disciplinas ele deve priorizar para não atrasar sua formatura. O algoritmo de caminho crítico implementado mostra, a partir de uma grade curricular, quais são as disciplinas que devem ser cursadas prioritariamente em cada período, ou seja, com ele é possível identificar o número mínimo de períodos que um aluno ainda deve ficar matriculado na universidade para finalizar sua graduação. A Figura 5.30 mostra um exemplo de caminho crítico na nova grade do curso de Ciência da Computação.

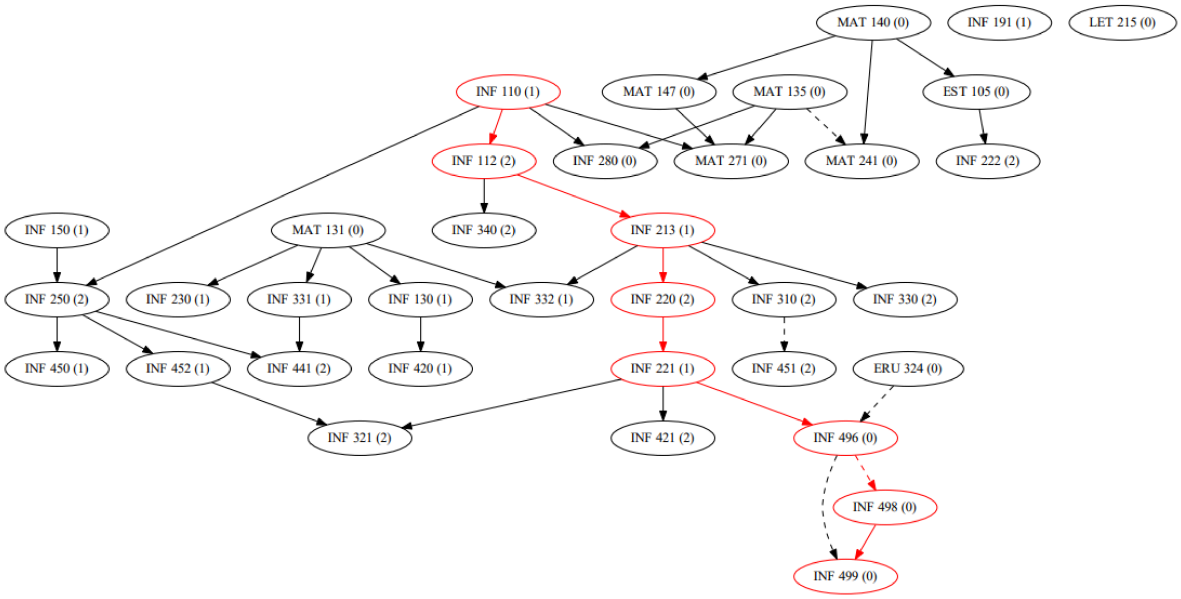

Figura 5.30: Exemplo de caminho crítico para o curso de Ciência da Computação, destacado em vermelho, no grafo que representa a grade do curso.

As disciplinas marcadas em vermelho são aquelas que devem ser cursadas prioritariamente assim que satisfeitos seus pré-requisitos. Cada disciplina mostrada no grafo aponta para outra disciplina que a possui como pré-requisito. As arestas tracejadas indicam relação de correquisito e o número entre parênteses, ao lado do nome da disciplina, indica a oferta daquela disciplina, isto é, 1 representa que a disciplina é

oferecida no primeiro semestre do ano, 2 no segundo e 0 indica que ela é ofertada nos dois. Para o exemplo mostrado, percebe-se que o estudante precisa de, no mínimo, 8 períodos para se formar. Dadas as condições dessa grade, ele não consegue se formar em menos que isso. Porém, se bem planejado, ele consegue não ultrapassar tanto esse tempo.

A escolha por cursar uma disciplina no momento certo pode mudar o resultado do aluno ao fim do semestre. É uma escolha difícil, mas muito importante. Na maioria das vezes, porém é um fator comumente decidido baseado em intuição e repetição, sem uma análise de dados históricos, que podem informar bastante. Muitos alunos tentam ou são orientados a se manterem dentro da grade, mas nem sempre é possível ou o melhor para o aluno, dada sua situação no momento. Com isso, o algoritmo de caminho crítico identifica os gargalos para que, respeitando os pré e correquisitos, o aluno possa adaptar a grade do curso às suas necessidades, sabendo quais disciplinas devem ser priorizadas em cada período. Como o algoritmo percorre o grafo em profundidade, buscando pelo maior caminho, respeitando as restrições, garantidamente, sabemos o número mínimo de períodos que o estudante ainda deve ter vínculo com a universidade, se seguir a rigor a recomendação do caminho crítico. Quanto às demais disciplinas do período, o estudante pode diversificar ou optar por seguir a recomendação feita pela heurística.

A heurística tem por finalidade fornecer um plano de estudos até o fim do curso do estudante de maneira que ele forme o mais cedo possível. Ela aloca as disciplinas respeitando suas relações de pré-requisitos e oferta, priorizando aquelas disciplinas do caminho crítico e o tempo mínimo que ainda resta para o estudante se graduar. Ela é parametrizável pelo número de créditos máximo a serem cursados por período.

Cada resultado encontrado contribuiu para a construção do protótipo que pode ser incorporado ao sistema de gerenciamento de históricos da UFV. Ao fazer o plano de estudos para o próximo semestre, o aluno escolhe um conjunto de disciplinas cujos pré-requisitos estejam cumpridos e, caso a situação simulada se enquadre em algum padrão identificado pelas regras de associação, isso será informado em forma de alerta.

Com essa informação o estudante pode decidir se precisa realmente cursar aquele conjunto de disciplinas naquele momento ou não. Se precisar de mais informações para decidir, ele pode consultar o caminho crítico gerado no canto superior da tela que irá mostrar quais disciplinas devem ser cursadas prioritariamente para que o estudante forme no menor tempo possível. É um *trade-off* sobre tempo e esforço onde as decisões devem ser tomadas.

Além disso, o estudante ainda pode examinar o número provável de reprovações que irá enfrentar no próximo semestre e refletir se o risco vale a pena. Com todos esses dados, o aluno fica mais confiante em suas decisões e inicia o próximo semestre mais preparado, sabendo dos riscos que assumiu correr e o quanto de esforço deve empenhar para evitar reprovações.

Todos os resultados aqui mostrados, compõem um protótipo que pode ser incorporado ao sistema de gerenciamento de notas e plano de estudos. Cada módulo será tratado com detalhes no Capítulo 6 e, no fim, será mostrado como todos eles juntos formam o protótipo funcional.

Capítulo 6

Prova de conceito

Como mencionado no Capítulo 4, o protótipo final é constituído de módulos que, trabalhando juntos, fornecem uma visão ampla das projeções para o próximo período acadêmico do estudante em questão. Esse Capítulo traz de modo mais aprofundado a implementação de cada módulo incorporado no sistema.

A mineração de padrões frequentes e extração de regras de associação é abordada na Seção 6.1. Como visto, ela tem como propósito revelar os padrões relacionados à reprovação na UFV, ou seja, o que com frequência ocorre na universidade que acarreta reprovação ao fim do semestre.

Na Seção 6.2, é mostrado o processo de construção do modelo preditivo de rede neural que retorna um número provável de reprovações ao fim do semestre que está sendo planejado.

Por fim, a Seção 6.3 irá mostrar a aplicação do caminho crítico cuja finalidade é dizer quantos períodos, no mínimo, o estudante ainda precisa estar vinculado à universidade para se formar. Além disso, também será apresentada a heurística gulosa que visa preencher automaticamente o plano de estudos, agilizando assim o trabalho do usuário final, que deve apenas validar com olhar crítico as recomendações.

6.1 Padrões frequentes e regras de associação

Após as análises preliminares, descritas no Capítulo 4 e o processo de agrupamento que dividiu a base em 4 grupos de disciplinas, foi executado o método de extração de padrões frequentes tratado a seguir.

A etapa de processamento de padrões frequentes inicia-se com uma consulta ao banco de dados, onde são extraídos os históricos nos quais houve alguma reprovação. Esses históricos vêm no formato descrito na Figura 4.1, acrescidos da carga horária da disciplina. Sobre cada histórico, é adicionado um atributo calculado: o período que o estudante está cursando naquele momento, dado pela Equação 6.1.

$$periodo = ((ano - ano\_ingresso) * 2) + semestre \tag{6.1}$$

Sendo o ano e o semestre referentes ao histórico e ano_ingresso, o ano que o estudante entrou na universidade. Utilizando a Figura 4.1 como exemplo, suponha que o aluno de matrícula 12345 tenha ingressado em 2013. Logo, ele cursou ARQ 106 em seu 3º período e INF 100 no 4º, segundo os cálculos 6.2 e 6.3, respectivamente.

$$ARQ\ 106 : ((2014 - 2013) * 2) + 1 = 3 \tag{6.2}$$

$$INF\ 100 : ((2014 - 2013) * 2) + 2 = 4 \tag{6.3}$$

Como pode-se notar pela Equação 6.2, que o aluno teria cursado ARQ 106 em seu 3º período porque a diferença entre seu ano de ingresso 2014 e o ano em que cursou a disciplina, 2013, retorna o valor 1. Multiplicado esse resultado por 2 e somado ao semestre 1, referente ao primeiro semestre de 2013, obtemos o valor de 3. De forma análoga, o estudante teria cursado INF 100 em seu 4º período de faculdade, como visto no cálculo 6.3. Esse cálculo é realizado para todas as entradas da base.

Essa extração da base nomearemos como históricos de reprovação. Outra extração importante para os passos seguintes é o histórico estendido, uma visão criada no banco de dados contendo os seguintes atributos: matrícula, código da disciplina, ano, semestre, nota, conceito, turma prática, turma teórica, nota no ENEM, modalidade, código do curso, ano de ingresso, forma de ingresso, UF de origem e carga horária da disciplina. Sobre ela também foi calculado o período seguindo a Equação 6.1. A Figura 6.1 resume os dados que temos até aqui e serão utilizados para a construção de transações.

Histórico de reprovação

Matrícula	Disciplina	Ano	Semestre	Nota	Conceito	Turmas	Créditos	Ano de ingresso	Período
12345	ARQ 106	2014	1		L	T1 P1	2	2013	3
12345	INF 100	2014	2	66		T1 P2	0	2013	4

Histórico estendido

Matrícula	Disciplina	Ano	Semestre	Nota	Conceito	Turmas	Créditos	Ano de ingresso	Período	Nota no ENEM	Modalidade	Código do curso	Forma de ingresso	UF
12345	ARQ 106	2014	1		L	T1 P1	2	2013	3	660	4	125	Y	ES
12345	INF 100	2014	2	66		T1 P2	0	2013	4	660	4	125	Y	ES
98765	ARQ 106	2013	1	87		T2 P1	2	2012	1	704	4	125	L	PE

Figura 6.1: Exemplo de dados das bases extraídas para montar as transações.

Pode-se observar que a base de histórico estendido contém os mesmos dados do histórico de reprovação e ainda conta com atributos a mais, referentes ao estudante, como nota do ENEM, modalidade, código do curso, forma de ingresso e estado. Vale lembrar que o histórico estendido contém tanto registros de reprovações quanto de aprovações, como pode-se notar pela Figura 6.1, onde as notas foram superiores a 60. Já no histórico de reprovações, como o próprio nome diz, estão apenas os registros os quais houve reprovação no semestre.

Os recortes mostrados na Figura 6.1, são usados para criar o conjunto de transações que serve como ponto de entrada para a detecção de padrões frequentes. A partir de cada registro do histórico estendido, conseguimos filtrar no histórico de reprovação se naquele mesmo período, um certo aluno reprovou em alguma disciplina cursada. Se houve alguma reprovação no período, ele será considerado na transação. Caso contrário, será desconsiderado, ou seja, os períodos onde o estudante foi aprovado em todas as disciplinas não compõem uma transação.

O próximo passo para a criação das tuplas de transação foi consultar no banco a lista de professores e mapeá-los em relação às disciplinas que ofereceram ao longo do tempo. Dessa forma, para cada conjunto de disciplina, ano, semestre e turma, havia um professor responsável.

Foram criadas funções para montar transações de uma forma genérica. A função precisa receber como parâmetros a base de históricos completa, o subconjunto dos registros em que houve reprovação e a lista de disciplinas que fazem parte de um dado *cluster*, como especificado no algoritmo 3.

Algoritmo 3: Monta transações

```
Entrada: histórico_estendido, histórico_reprovação, cluster
Saída: conjunto de transações
Cria um conjunto vazio de transações
para cada registro de histórico_estendido faça
    créditos = 0
    se registro.código_disciplina está presente no cluster então
        matrícula = registro.matrícula
        ano = registro.ano
        semestre = registro.semestre
        período = Concatena ano com semestre
        disciplinas_do_semestre = Filtra histórico_reprovação por matrícula,
          ano, semestre
        se tamanho de disciplinas_do_semestre > 0 então
            Insere uma transação vazia no conjunto de transações
            para cada disciplina em disciplinas_do_semestre faça
                código_disciplina = disciplina_R se houve reprovação ou
                  disciplina_A, caso contrário
                professor = Mapeia_professor (disciplina.código, ano, semestre,
                  disciplina.turma)
                créditos += disciplina.créditos
                Adiciona na transação corrente o código_disciplina e o professor
            fim para
            Adiciona na transação corrente os seguintes dados:
              número_de_disciplinas, créditos, nota_enem, código_curso e
              modalidade
        fim se
    fim se
fim para
```

As transações foram criadas iterando sobre a base de histórico estendido. Para cada iteração, inicia-se um contador de créditos com 0. Faz-se a verificação se o código da disciplina pertence ao *cluster* que está sendo tratado e caso positivo, é incluído na transação a matrícula, o ano, o semestre e o período no formato ano-semestre. Em seguida, a base de histórico de reprovação é filtrada pelos dados de matrícula, ano e semestre, ou seja, queremos saber o conjunto de disciplinas nas quais o aluno reprovou no mesmo período em que cursava a disciplina da iteração corrente. Se esse conjunto é vazio, ele não entra nas transações, isto é, naquele período, o estudante

foi aprovado em todas as matérias. Esse tipo de situação não entrará na base de transações já que o ponto de interesse são as reprovações.

Caso haja reprovação, é inserido no dicionário de transações uma instância para aquela matrícula e período, criando assim, uma nova transação vazia. Para cada disciplina cursada em tal período, adiciona-se um sufixo em seu código a fim de indicar em quais delas houve reprovação (_R) e em quais o aluno fora aprovado (_A).

Também é incluído na transação, os respectivos professores que ministraram as disciplinas naquele período. A função Mapeia_professor retorna o código do professor que ofertou uma disciplina em um dado ano e semestre para uma certa turma. Assim, dados o código da disciplina, o ano e semestre em que foi cursada, e as características da turma a qual o aluno estava alocado, como tipo (teórica ou prática) e número a turma, conseguimos saber quem era o professor responsável. É acrescido ao código anonimizado do professor, um prefixo para indicar se a turma a qual ministrou na época especificada era teórica (_T) ou prática (_P).

Após esse passo, a carga horária de cada disciplina do período é somada à contagem de créditos do período. Por fim, também são incluídos o número de disciplinas cursadas naquele período e atributos do estudante, tais como, a nota no ENEM, o código do curso e a modalidade de cota.

No fim, uma transação se parece com o descrito na Figura 4.3. Foram feitos vários experimentos com diferentes conjuntos de transações. Para evitar a segmentação em cluster, basta enviar na lista *cluster* todas as disciplinas.

A partir das transações montadas, extraem-se os conjuntos de itens frequentes abordado na Seção 3.1 com um suporte mínimo de 0,03. Uma vez tendo o conjunto de itens frequentes, aplica-se o algoritmo de regras de associação, também tratado na Seção 3.1. Foram selecionadas apenas aquelas regras que possuíam no mínimo 70% de confiança. Por fim, o conjunto de regras ainda foi filtrado para manter apenas aquelas cujo consequente continha somente um item, sendo esse, uma reprovação. Queremos analisar quais padrões estão relacionados a reprovações, ou seja, não estamos interessados em transações que levam a uma aprovação como consequência. Optamos por restringir o tamanho do consequente a 1 para isolar o efeito de um conjunto de itens em vários resultados ao fim do semestre. Uma regra de associação nesse contexto possui o formato descrito na Figura 6.2.

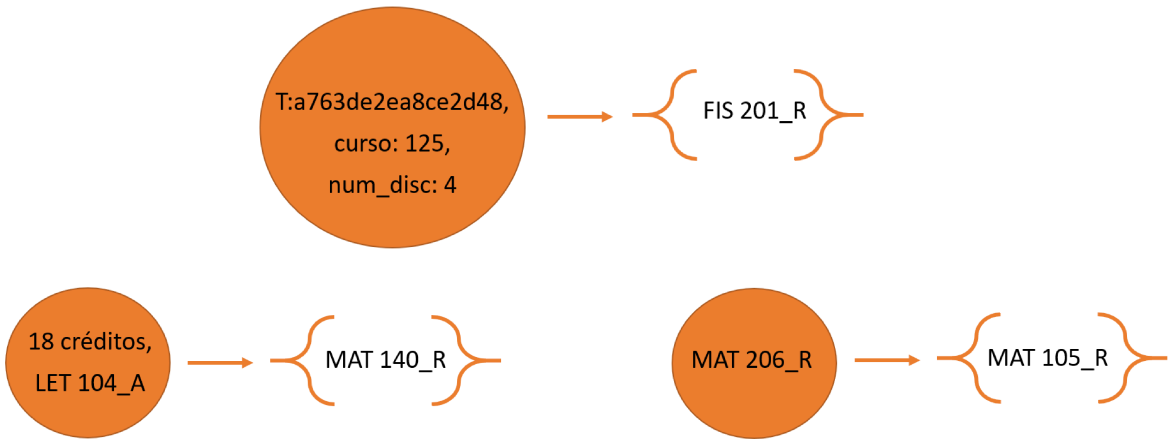

Figura 6.2: Exemplo de regras de associação encontradas.

Os conjuntos representados por círculos são os antecedentes da regra e os con-

juntos unitários representados por chaves são os consequentes. São mostrados três exemplos de regras: no primeiro, um professor de turma teórica (identificado pelo prefixo T), devidamente anonimizado, no código de curso 125 em um semestre onde estão sendo cursadas 4 disciplinas ocorrem em conjunto com uma reprovação em FIS 201. O segundo exemplo mostra que a ocorrência de 18 créditos e uma aprovação em LET 104 está relacionada à reprovação em MAT 140. E o último exemplo retrata que o conjunto de antecedentes pode ser composto apenas por um item. Neste caso, muitas das vezes em que houve reprovação em MAT 206, também ocorreu a reprovação em MAT 105. Vale ressaltar que o conjunto final de regras passou por um filtro que removeu aquelas com consequentes que possuíam mais de um item. Dessa forma, só serão encontrados na base final regras cujos consequentes são conjuntos unitários.

Esta Seção abordou um ponto de vista das reprovações: os padrões relacionados a ocorrência delas. Outro aspecto interessante a ser explorado é a possibilidade de prever uma reprovação, antes mesmo que essa ocorra. Neste trabalho, fizemos isso utilizando redes neurais, como será mostrado na Seção 6.2.

6.2 Rede neural

Outro módulo incorporado no protótipo final foi um modelo preditivo. Como descrito no Capítulo 4, alguns modelos preliminares foram testados, porém, aquele com o melhor resultado é uma rede neural artificial, que será melhor abordada nesta Seção.

Sobre os dados descritos no Capítulo 4, foram calculadas as *features*, mostradas na Figura 4.5, a fim de construir tuplas de entrada para um modelo de preditivo. Como mencionado, o conjunto foi normalizado segundo a Equação 4.1 e a matrícula do aluno foi retirada para remover o elemento de identificação única dos registros, o que poderia enviesar o modelo. Durante as previsões, também se removeu o número de reprovações, a variável alvo. O conjunto de dados foi dividido em treino e teste com as proporções 85% e 15%, respectivamente.

Para construir um modelo de RNA, primeiramente queríamos explorar parâmetros que pudessem garantir um bom resultado. Assim, fizemos um *slot* de testes, com modelos simples para analisar cada parâmetro isoladamente e ter uma ideia de como ele poderia contribuir ou não para a rede. A seguir, serão especificados os parâmetros testados para compor o preditor. Implementamos nossos modelos de redes neurais utilizando o *framework* Keras (Chollet et al. (2015)).

Como explicitado na Seção 3.2, o processo de *backpropagation* em redes neurais pode contar com otimizadores. Neste trabalho, foram construídos modelos com dois tipos de otimizadores: o SGD e o Adam, também abordados na Seção 3.2. No primeiro modelo, usamos SGD numa rede com três camadas internas de 32 neurônios e no segundo, substituímos o otimizador pelo Adam. O segundo experimento foi montado pensando em avaliar o *scheduler* da taxa de aprendizagem. Utilizamos quatro abordagens para tal: uma atualização passo a passo, linear, uma quadrática e outra polinomial de grau 5. O Keras possui um otimizador que reduz a taxa de aprendizado quando uma métrica de avaliação para de melhorar, chamado Redução de platô (*Reduce Learning Rate on Plateau*, Chollet (2017)), incluído também no experimento.

Outros aspectos levantados foram a troca da camada intermediária mais interna por outra com mais neurônios e a inserção de ruídos, tais como *Dropouts* (Srivastava et al. (2014)) ou *Gaussian Noise* (Goodfellow et al. (2016)). A ideia de incluir uma

camada maior no meio é permitir que uma função não linear mais complexa seja descoberta e as duas últimas ideias mencionadas são técnicas para evitar *overfitting* do modelo.

No intuito de investigar também qual função de ativação se adequaria melhor ao nosso propósito, testamos modelos com *Sigmoid* e *Leaky ReLU*.

Por fim, especulamos que aumentar o número de épocas de treinamento de 20 para 100 poderia trazer resultados melhores, já que seria feito um treino mais longo. Dessa forma, este também foi um tema de interesse em nossos experimentos. A tabela da Figura 6.3 mostra o parâmetro avaliado em cada experimento elaborado, relembrando que eles foram executados em cima de uma rede base de três camadas internas de 32 neurônios cada em um treinamento de 20 épocas.

Experimento	Parâmetro de teste
1	Otimizadores: SGD x Adam
2	Atualização de taxa de aprendizagem: passo a passo, linear, quadrática, polinomial e redução de platô
3	Camada interna maior
4	Ruídos (Dropouts e Gaussian Noise)
5	Função de ativação: Leaky ReLU x Sigmoid
6	Mais épocas (100)

Figura 6.3: Experimentos de avaliação de parâmetros para a rede.

No primeiro experimento foram avaliadas duas opções de otimizadores: um modelo contendo SGD e outro com Adam. A segunda bateria de testes visava avaliar a atualização da taxa de aprendizagem. Assim, no segundo experimento foram criados 5 modelos, cada um contendo uma estratégia diferente de atualização, como mostrado na tabela da Figura 6.3. Para o terceiro experimento, construiu-se um modelo que contava com uma topologia diferente da rede inicial. Novos modelos foram construídos aumentando o número de neurônios em uma de suas camadas internas. O experimento 4 avaliava a inclusão de ruído na rede, enquanto o 5 explorava duas opções de função de ativação: *Leaky ReLU* em um dos modelos e *Sigmoid* no outro. Por fim, o último experimento apenas alterava o número de épocas de treinamento para 100, deixando assim, a rede treinar por mais tempo.

Com base nos resultados obtidos para cada parâmetro isolado, criamos alguns modelos utilizando aqueles que obtiveram os melhores desempenhos. Para servir de ponto de partida, foi criada uma rede inicial combinando os melhores parâmetros encontrados, chamada de Combinada. Trata-se de uma rede de três camadas internas de 32 neurônios cada, com a função de ativação *Leaky ReLU* a um alfa de 0,01 e otimizador ADAM. A partir dos resultados dessa rede preliminar, foram construídos outros 6 modelos também por combinação dos melhores parâmetros. Em alguns casos, foi também testado o ajuste fino de argumentos intrínsecos da função de ativação, como o alfa na *Leaky ReLU*. A tabela da Figura 6.4 mostra o que constitui cada modelo criado.

Parâmetros	Modelo 1	Modelo 2	Modelo 3	Modelo 4	Modelo 5	Modelo 6
Adam	✔	✔	✔	✔	✔	✔
Camada interna maior	✔	✔	✔	✔	✔*	✔*
Leaky ReLU (α = 0,1)			✔			
Leaky ReLU (α = 0,01)	✔	✔			✔	✔
Leaky ReLU (α = 0,001)				✔		
100 épocas	✔	✔	✔	✔	✔	✔
Redução de platô		✔		✔		
Atualização Quadrática de LR	✔		✔		✔	✔
Dropout						✔

Figura 6.4: Combinação de parâmetros usada em cada modelo testado.
* Nos modelos de 1 a 4, a camada maior tem 64 neurônios, enquanto nos modelos 5 e 6, elas têm 128.

Na tabela da imagem 6.4, o símbolo de verificado (✓) representa a presença do parâmetro no modelo. Pode-se notar que todo modelo inclui Adam como um otimizador, uma camada maior (de 64 ou 128 neurônios), *Leaky ReLU* (com valores diferentes para alfa) e 100 épocas. O restante dos parâmetros foi combinado de forma variada na construção dos modelos. Mais uma vez, coletamos métricas tanto no treinamento quanto nos testes. Nossos resultados mostram que o modelo 1, dentre os outros, foi o melhor e obteve previsões satisfatórias.

Portanto, o modelo final capaz de prever o número provável de reprovações de um estudante ao fim de um semestre é o modelo 1. Ele é composto por três camadas internas (com 32, 64 e 32 neurônios respectivamente), *Leaky ReLU* como função de ativação e um alfa de 0,01; otimizador Adam e uma atualização quadrática da taxa de aprendizagem, executando por 100 épocas no treinamento, ilustrado na Figura 6.5.

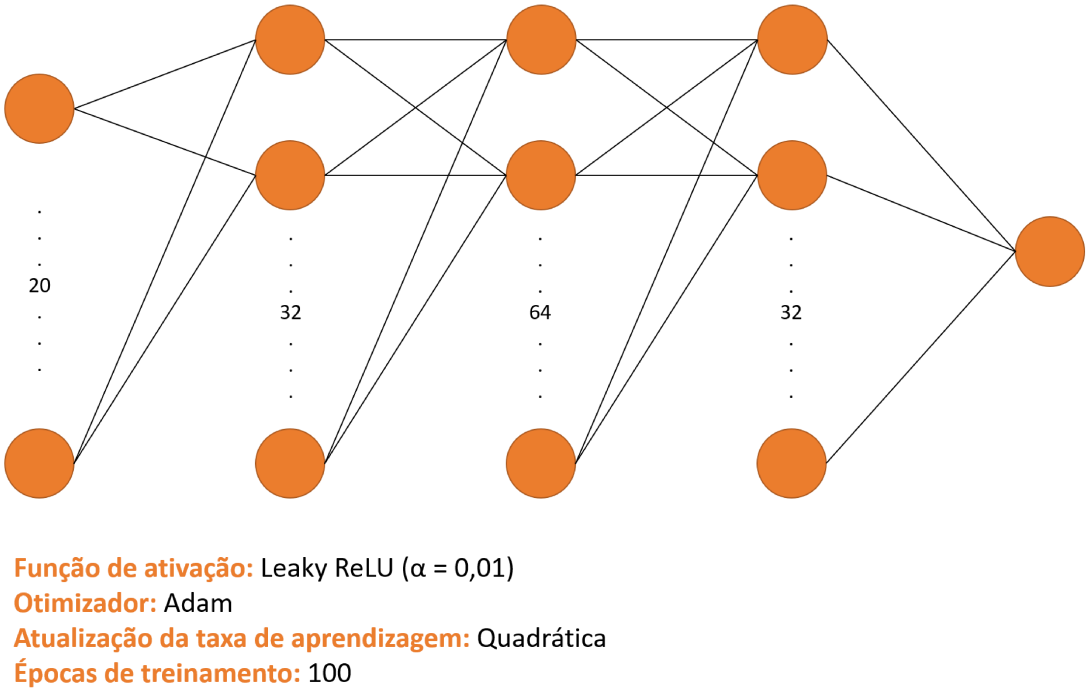

Figura 6.5: Esquema ilustrativo do modelo preditivo selecionado.

Tendo construído um modelo capaz de estimar as reprovações, o usuário consegue antever problemas futuros e tentar evitá-los. Outra informação do futuro que pode agregar é o tempo estimado para a formatura, o que é desenvolvido no último módulo incorporado ao protótipo proposto aqui, tratado na Seção 6.3.

6.3 Caminho crítico e heurística gulosa

No contexto desse trabalho, o caminho crítico é o caminho mais longo de dependências presente no grafo que representa a grade, ou seja, aquele que contém mais nós.

Para conseguir encontrar o caminho crítico sobre a grade de um dado curso, foi necessário, primeiramente, obter tal dado. Dessa forma, foram escolhidos quatro cursos, um de cada Centro de Ciências, e extraímos suas grades curriculares do Catálogo da UFV.

O dado foi obtido manualmente e modelado inicialmente como um dicionário onde as chaves eram os períodos sugeridos e os valores, a lista de disciplinas recomendada pela coordenação do curso, como mostra a Figura 6.6. O problema dessa estrutura foi a impossibilidade de relacionar as disciplinas com seus pré-requisitos e correquisitos. Eram necessárias estruturas de dados adicionais para cumprir essa finalidade.

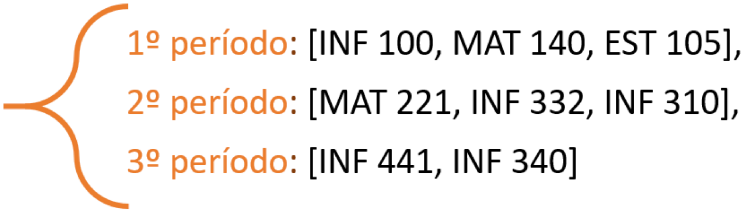

Figura 6.6: Primeira modelagem de grade curricular.

Assim, foi elaborado um novo modelo de representação de grade: um dicionário onde as chaves são disciplinas e os valores seus atributos, sendo eles, uma lista de pré-requisitos, uma lista de correquisitos, um número inteiro que representa em qual semestre a disciplina é oferecida (1, 2 ou 0 se ela é ofertada em ambos) e um inteiro que equivale ao período ideal, segundo a grade sugerida. Desse modo, os relacionamentos entre disciplinas ficam na mesma estrutura, mas, ainda assim, eles continuam separados por duas chaves distintas, ilustrado na Figura 6.7.

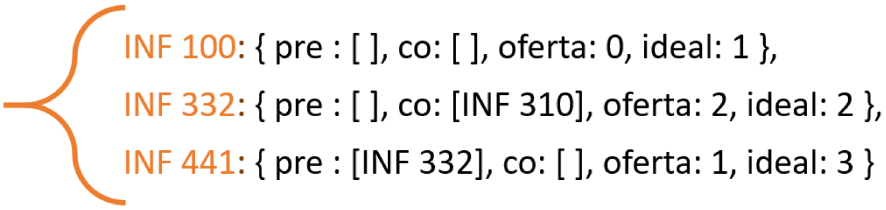

Figura 6.7: Segunda modelagem de grade curricular.

Considerando os entraves dos modelos anteriores, foi definida uma terceira abordagem onde a grade é representada por um dicionário onde as chaves são as disciplinas e o valor é uma tupla com os relacionamentos entre elas, o semestre de oferta, o período ideal e a carga horária da disciplina. Os relacionamentos são descritos por um dicionário onde as chaves são as disciplinas e o valor pode ser 0 para correquisito ou 1 para pré-requisito, como na Figura 6.8.

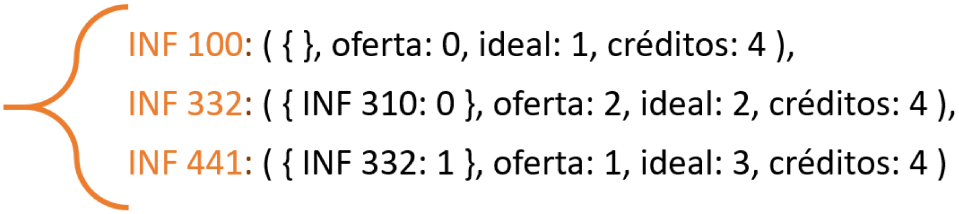

Figura 6.8: Modelagem final para grade curricular.

Como mencionado anteriormente, os dados foram coletados manualmente e para aquelas disciplinas as quais não foram encontradas a carga horária, essa foi preenchida com 4 créditos. Tal escolha foi baseada no número de créditos médio, levando em consideração todas as disciplinas da UFV.

Vale mencionar que a modelagem final não contempla pré-requisitos de horas cursadas nem relacionamentos do tipo "ou". Um exemplo seria o aluno que muda de curso ou de catálogo e concluiu disciplinas semelhantes que podem substituir um pré-requisito na nova grade, ou seja, uma disciplina pode ser cursada caso um conjunto ou outro de pré-requisitos seja satisfeito. Nessas situações, foi inserido na grade apenas o primeiro conjunto.

Uma vez tendo a grade representada pelo dicionário da Figura 6.8, foi necessário converter essa representação para um grafo a fim de encontrar o caminho crítico caminhando por seus nós e arestas.

Um nó é a representação de uma disciplina. Ele contém a lista de disciplinas (nós) as quais se relaciona por pré ou correquisito, o número de caminhos possíveis de serem seguidos a partir dele e o comprimento desses caminhos, o semestre do ano em que é ofertada a disciplina, o período ideal para ser cursada, a sua carga horária em créditos, o código da disciplina, e o caminho mais longo a partir dele, como mostra a Figura 6.9.

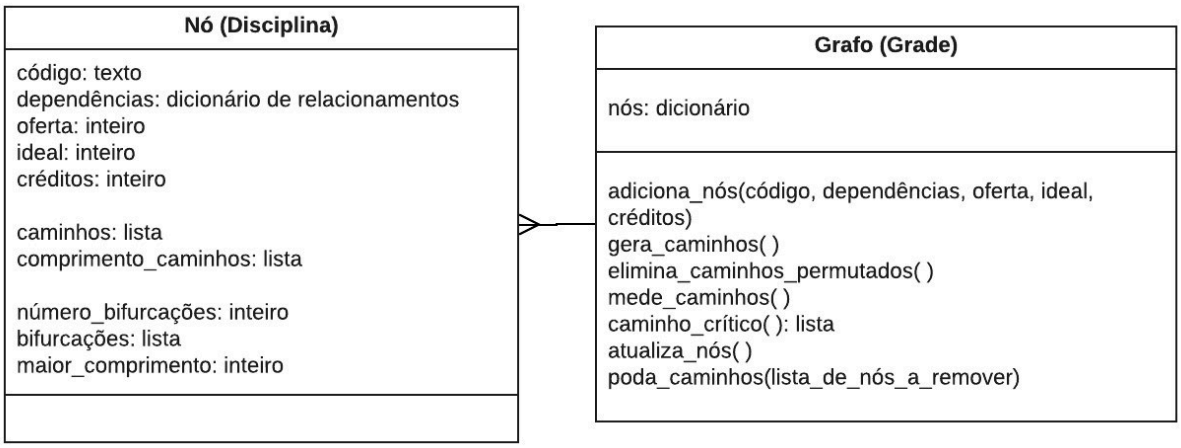

Figura 6.9: Diagrama de classe da modelagem final para grade curricular.

O grafo é um conjunto de nós e possui os métodos ilustrados na Figura 6.9. Ele é a tradução da modelagem 6.8. Como o próprio nome diz, o primeiro método do diagrama 6.9 adiciona nós ao grafo. Já o segundo, parte de cada nó e caminha ao longo de suas arestas, gerando os caminhos a partir de cada um deles. Os nós são ligados uns aos outros pelas dependências de pré ou correquisito. Assim, uma aresta pode ter peso 1 ou 0, respectivamente.

O método que elimina caminhos permutados remove aqueles caminhos que partem de um mesmo nó e chegam a um mesmo lugar por trajetórias distintas. O método seguinte, mede_caminhos( ), apenas percorre os caminhos gerados contando o número de nós. Já a atualização dos nós altera os valores dos atributos, uma vez realizado todos os cálculos. Por fim, o último método listado poda os caminhos, excluindo deles os nós e subcaminhos a partir desses nós que pertencem a uma lista previamente definida. Isso será útil para remover, posteriormente, as disciplinas já cursadas pelos estudantes.

Tendo percorrido o grafo inteiro e gerando todos os caminhos possíveis a partir de cada nó, basta contar o comprimento desses caminhos e eleger o maior deles como o caminho crítico.

Sabendo as disciplinas que compõem o caminho crítico, fica mais fácil alocá-las no plano para concluir o curso no menor tempo possível. Mas ainda restam as outras disciplinas a serem alocadas e para auxiliar nessa tarefa, pensou-se em construir uma heurística capaz de preencher automaticamente o plano de estudos.

Com o intuito de sugerir de forma automática um conjunto de disciplinas a serem cursadas a cada período até a formatura do estudante, criou-se um algoritmo guloso, conforme descrito no Algoritmo 4. Outra finalidade desse algoritmo é agilizar o processo de planejamento. Contudo, o algoritmo deveria obedecer a algumas restrições tais como relacionamentos de pré-requisitos e a priorização pelo caminho crítico, de modo a não atrasar a formatura dos alunos.

Os dados essenciais foram obtidos através de consultas no banco e a leitura do arquivo JSON onde foi salva a grade criada em formato de dicionário seguindo a abordagem citada anteriormente nesta Seção (Figura 6.8). Foram consultadas a lista de disciplinas, seus respectivos números de créditos e o conjunto de disciplinas já concluídas por um estudante em sua jornada acadêmica. A última consulta citada foi parametrizada pela matrícula do aluno.

A partir do dicionário equivalente à grade, montou-se o grafo correspondente (Figura 6.9). O próximo passo foi gerar os caminhos com o método gera_caminhos( ) e depois retirar as disciplinas já cursadas de todos os caminhos gerados, com a função poda_caminhos( ). A seguir, descobrimos o caminho crítico com o método caminho_crítico( ) que retornou o mais longo dentre os caminhos gerados e atualizamos os atributos de todos os nós do grafo com a função atualiza_nós( ).

Feito isso, tem-se todos os insumos primordiais para a execução da heurística. A primeira etapa consiste em alocar o caminho crítico descoberto, devido a sua alta prioridade sobre as demais disciplinas. Para isso, o algoritmo verifica se a primeira disciplina do caminho já pode ser alocada no próximo período a ser planejado. Caso não possa, altera-se o período para o seu subsequente. A seguir, a primeira disciplina é alocada para o período, atualizado conforme o teste.

Para as próximas disciplinas do caminho, verifica-se a relação de pré-requisito: se a última disciplina alocada é pré-requisito da próxima, incrementa o período, senão, adiciona a disciplina no período atual, como é mostrado no algoritmo 4.

Algoritmo 4: Heurística gulosa para preenchimento do plano de estudos

```
Entrada: grade, matrícula, ano_corrente, próximo_semestre, limite_créditos
Saída: plano de estudos sugerido
disciplinas = Consulta disciplinas
concluídas = Consulta pela matrícula as disciplinas já concluídas pelo aluno
grafo.poda_caminhos(concluídas)
grafo.gera_caminhos( )
caminho_crítico = grafo.caminho_crítico( )
grafo.atualiza_nós( )
próximo_período = próximo_semestre
// Primeiro passo:  Alocar o caminho crítico
plano = Cria plano vazio
se primeira disciplina do caminho_crítico não será ofertada no próximo_período então
    próximo_período = próximo_período + 1
fim se
Inclui a primeira disciplina do caminho crítico no plano
para cada disciplina em caminho_critico depois da primeira faça
    se disciplina é pré-requisito da que já foi alocada então
        próximo_período = próximo_período + 1
    senão
        Inclui disciplina no plano
    fim se
fim para cada
próximo_período = próximo_semestre // Redefine para o período inicial
// Segundo passo:  Selecionar disciplinas seguindo os critérios de desempate
disciplinas_grade = Cria estrutura vazia
para cada disciplina em grade faça
    se disciplina não está no caminho_crítico nem em concluídas então
        disciplinas_grade[ disciplina.código ] = (maior caminho, número de
         caminhos, período ideal, oferta)
    fim se
fim para cada
conjunto_disciplinas, carga_horária_total = Filtra disciplinas candidatas
/* Enquanto todas as disciplinas não estiverem sido planejadas                */
enquanto tamanho(plano) < tamanho(disciplinas_grade) faça
    se plano.carga_horária + carga_horária_total <= limite_créditos então
        Inclui disciplinas filtradas no plano
    senão
        enquanto plano.carga_horária + carga da primeira disciplina do
         conjunto_disciplinas <= limite_créditos faça
            Inclui disciplina no plano
            Retira disciplina de conjunto_disciplinas
        fim enqto
    fim se
    próximo_período = próximo_período + 1
    conjunto_disciplinas, carga_horária_total = Filtra disciplinas candidatas
fim enqto
```

Outro parâmetro da heurística é o número limite de créditos que o aluno pode cursar por período. Esse valor será utilizado na segunda etapa do procedimento. Etapa essa que começa filtrando todas as disciplinas da grade que não estão no caminho crítico nem na lista de concluídas.

Cria-se uma estrutura de dados com as disciplinas filtradas e seus atributos em colunas que serão usados para ordenação. Isso é feito para priorizar alguns quesitos, sendo eles: tamanho do caminho mais longo a partir daquele nó, número de caminhos possíveis a partir do nó, carga horária em número de créditos, período ideal segundo a grade e semestre de oferta.

Uma vez ordenado, o conjunto passa por um filtro onde são descartadas as disciplinas já planejadas, as que não possuem seus pré-requisitos cumpridos e aquelas cujas ofertas não correspondem ao período de oferta do período a ser preenchido.

Enquanto houver disciplinas para alocar nos períodos, é verificado se a inclusão dela não ultrapassa o limite de créditos estabelecido. Quando ultrapassar, altera-se o período e filtra de novo o conjunto de disciplinas. Esse procedimento é importante porque ao se alterar o período, muda também a oferta a ser filtrada.

Resumidamente, a cada iteração filtra-se e tenta alocar as disciplinas filtradas no período corrente até que o limite de créditos seja alcançado. As iterações acabam quando todo o conjunto de disciplinas é alocado. Um exemplo de sua execução passo a passo é ilustrado no apêndice A.

A heurística é gulosa na medida em que procura preencher os semestres com as disciplinas mais prioritárias até que atinja o limite de créditos especificado. A ordenação dos dados, antes da escolha, garante essa prioridade, já que o algoritmo irá selecionar as n primeiras disciplinas do conjunto ordenado de acordo com os critérios definidos nesta Seção. A preferência é cursar o quanto antes as disciplinas que tenham seus pré-requisitos cumpridos, e, se der empate, o algoritmo considera as disciplinas que bloqueiam caminhos mais longos. Se der outro empate, mais três critérios são levados em consideração: o número de disciplinas que dependem daquela, o período ideal a ser cursado segundo a grade e a carga horária, da maior para a menor. O algoritmo garante o cumprimento dessas restrições já que o primeiro critério de ordenação é se a disciplina possui ou não seus pré-requisitos cumpridos. Além disso, depois de ordenado, o conjunto de disciplinas ainda passa por um filtro que seleciona apenas as disciplinas em que o valor do atributo seja verdadeiro, isto é, o algoritmo não seleciona disciplinas cujos pré-requisitos estejam pendentes.

O limite de créditos e os pré-requisitos são fatores importantes e devem ser respeitados nessa escolha para que seja um planejamento factível, possível de ser implementado e não ultrapasse as limitações do aluno. Apesar de ser o último critério de desempate, a cada iteração para escolha de disciplina, também é verificado se sua carga horária não ultrapassa o limite de créditos estabelecido. Se isso ocorre, a primeira disciplina da fila é desconsiderada no momento e a próxima da lista é testada.

Após ter conhecimento de todos os módulos mostrados nas Seções anteriores, a Seção 6.4 descreverá como todos eles compõem o protótipo funcional de um sistema inteligente que auxilia a tomada de decisões na elaboração de plano de estudos.

6.4 Protótipo funcional

As técnicas desenvolvidas e descritas nas Seções anteriores fornecem uma visão pontual para o estudante. Cada uma delas aborda um aspecto diferente que é levado em conta no momento do plano de estudos. Dessa maneira, pensou-se em uni-las como módulos de um sistema, incorporando-as no momento do planejamento a fim de auxiliá-lo na tomada de decisões. A ideia é que tais informações aparecessem de forma clara, não ambígua e com um caráter de sugestão para o usuário final. A Figura 6.10 mostra a tela inicial do protótipo funcional.

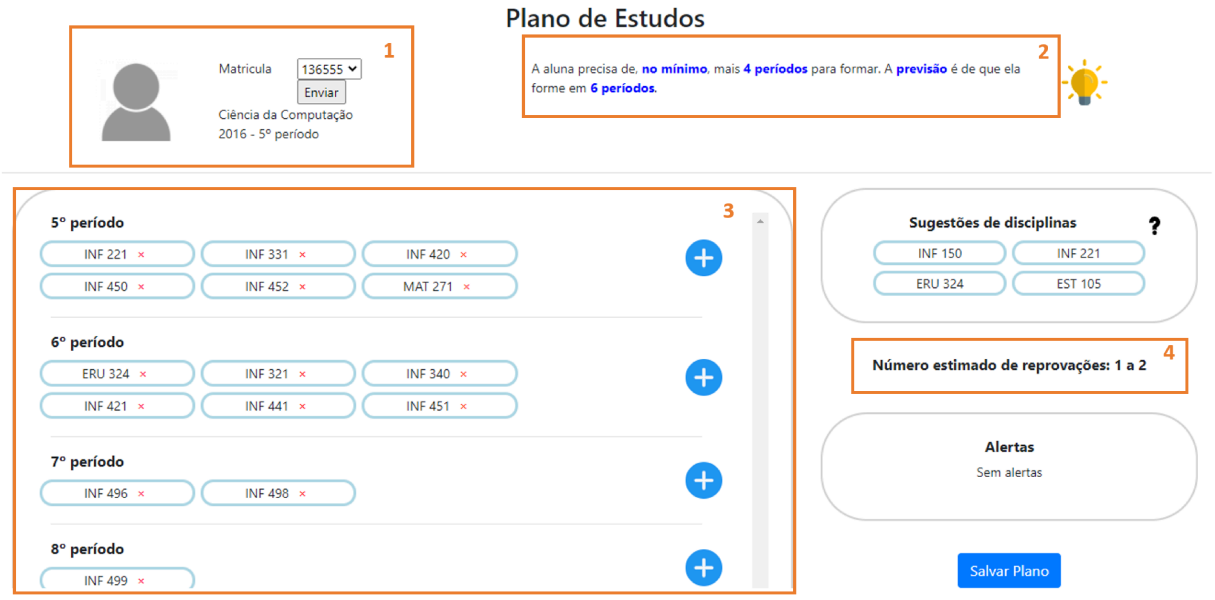

Figura 6.10: Tela inicial do protótipo

O produto do algoritmo de caminho crítico e da heurística, descritos na Seção 6.3 foi traduzido em um texto para melhor compreensão do usuário, localizado na parte superior da tela, ao lado dos dados pessoais do aluno, área identificada pelo número 1.

O usuário deve entrar com a matrícula do estudante e o sistema irá mostrar sua foto, curso, ano de ingresso e período atual para que o orientador confirme se está realmente realizando o plano de estudos do aluno desejado. Na área demarcada com 2, aparecem os resultados do caminho crítico e da heurística em forma de texto para ficarem mais intuitivo. O sistema mostra, quantos períodos, no mínimo, o estudante ainda precisa cursar, o resultado do caminho crítico calculado e qual é o tempo previsto para a formatura, considerando o atual plano e alocação de disciplinas, o resultado da heurística.

Pensando em destacar as recomendações feitas por eles, usou-se uma cor diferente para ressaltar a quantidade de períodos que ainda restam. Por padrão, o plano não viria automaticamente todo preenchido, mas recuperaria da memória o último planejamento salvo. Caso não exista e o plano esteja vazio, o usuário pode acionar a heurística clicando no ícone da lâmpada (remetendo a ter uma ideia para os próximos períodos).

Caso o usuário queira, ele pode replanejar automaticamente todos os períodos seguintes, clicando no ícone da lâmpada, onde se abrirá uma janela perguntando o

número máximo de créditos por período que aquele estudante pode cursar, mostrado na Figura 6.11.

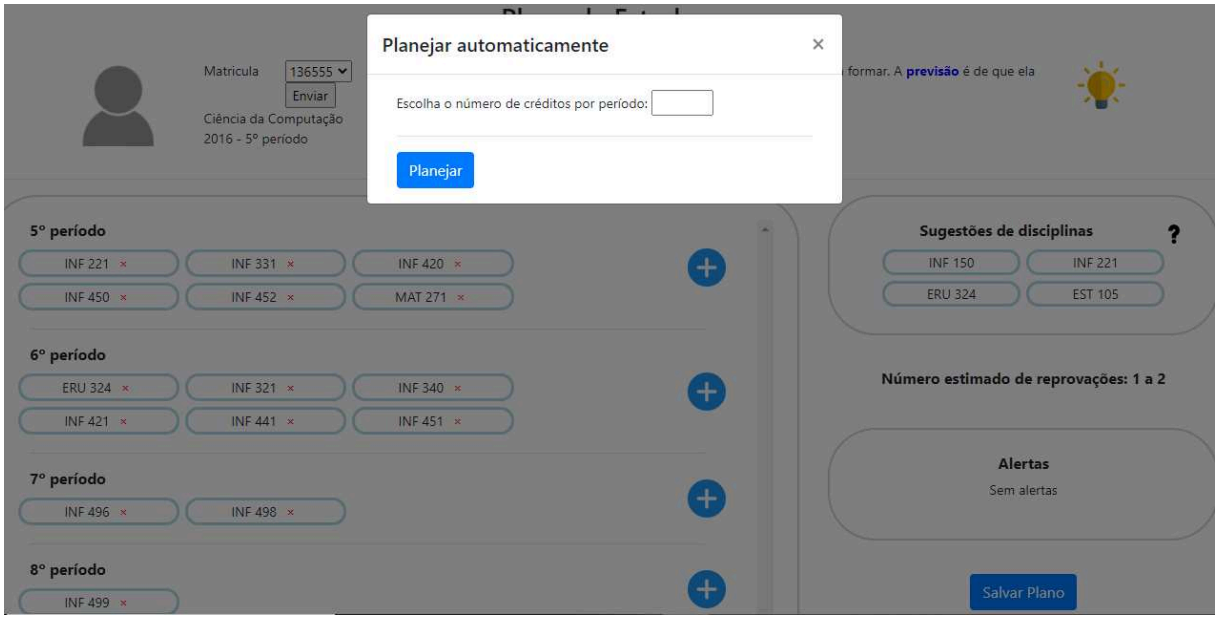

Figura 6.11: Janela para entrada do parâmetro da heurística.

Na área 3, se encontra o plano em si, com todos os períodos restantes a serem cursados e as respectivas disciplinas planejadas para cada um, até então. Acreditamos que seria interessante dar um pouco mais de informação sobre as disciplinas que estão sendo escolhidas no momento do preenchimento. Assim, ao se passar o mouse sobre uma disciplina escolhida, aparece, em um balão informativo a probabilidade geral de reprovação, ou seja, dentre todos os alunos da UFV, quantos já reprovaram nela, em percentual.

A probabilidade geral de reprovação em uma disciplina oferece para o aluno uma contextualização do ambiente, ou seja, ela informa como os estudantes estão se saindo, em geral, em cada disciplina cursada. O próprio estudante, muitas vezes, tem conhecimento de suas capacidades e limitações. Dessa forma, consegue identificar se está na média, se é provável que alcance resultados similares ao da maioria ou não. Sendo assim, dada a possibilidade de conhecer de antemão a média da universidade, ele consegue, comparando seu rendimento aos demais, ter uma ideia se corre o mesmo risco que eles.

A inclusão de outras disciplinas no plano seria feita ao clicar no botão com o símbolo de adição (+), onde se abre uma caixa de seleção de disciplinas, mostrada na Figura 6.12.

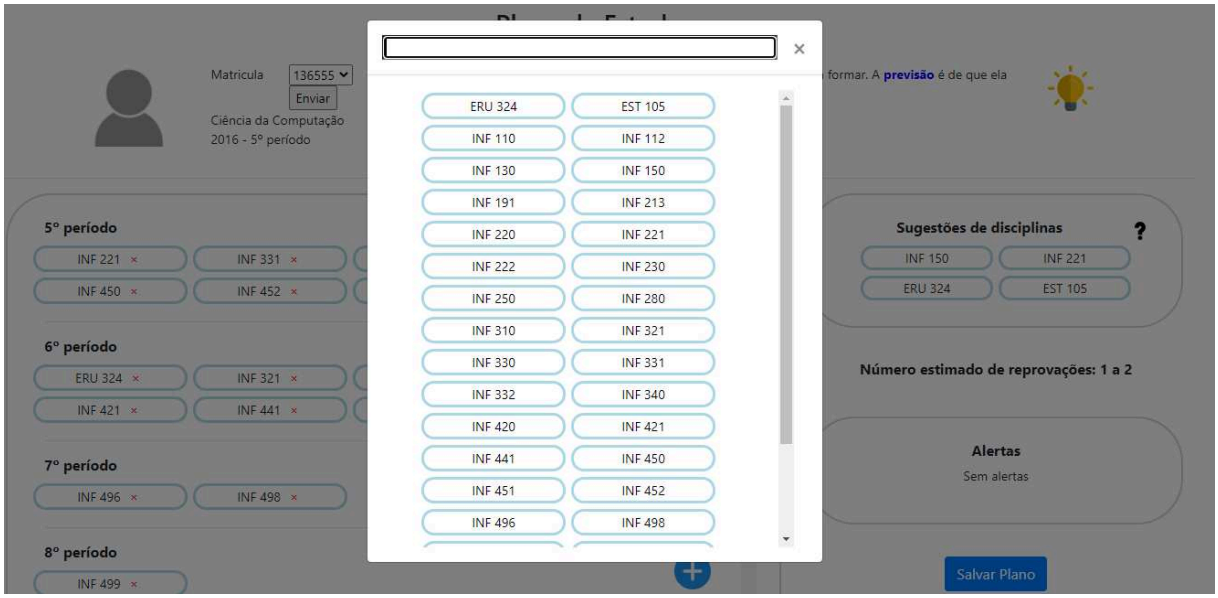

Figura 6.12: Lista de disciplinas que podem ser incluídas no plano de estudos, com uma barra de pesquisa para facilitar a busca.

Para remover uma disciplina de um certo período, basta clicar no x vermelho que se encontra ao lado de seu nome.

Outro resultado que advém da heurística é a caixa de Sugestões de disciplinas, colocada à direita da área 3, como um acesso rápido. As disciplinas que aparecem ali são as 4 primeiras alocadas pela heurística. A caixa de sugestões de disciplinas é um atalho para o usuário que ainda não planejou o próximo período. Nela aparecerão as disciplinas consideradas pela heurística que devem ser cursadas a seguir.

Com um caráter de alerta, aparecem os dois últimos resultados, mostrados também à direita da área 3, logo abaixo da caixa de sugestões. À cada nova inclusão de disciplinas ao período, uma previsão é realizada pela rede neural descrita na Seção 6.2 que retorna a probabilidade de ocorrer reprovações dentre aquele conjunto de disciplinas. Assim, a área 4 mostra a saída do modelo preditivo de rede neural, nela é mostrada o número provável de reprovações que o aluno terá no próximo período caso curse aquele conjunto de disciplinas planejadas.

Também é testado se alguma combinação dessas disciplinas selecionadas para o próximo período se enquadra nas regras de associação encontradas, como mostrado na Seção 6.1. As regras também foram transcritas de um modo mais legível para o usuário final e destacadas em uma caixa de alertas para chamar a atenção. A caixa de alertas, irá mostrar um padrão se esse for encontrado entre as disciplinas planejadas pelo algoritmo de detecção de padrões sobre a base da UFV. Quando isso ocorrer, aparecerá uma mensagem semelhante à mostrada na Figura 6.13, indicando um aumento no risco geral de reprovação em uma determinada disciplina.

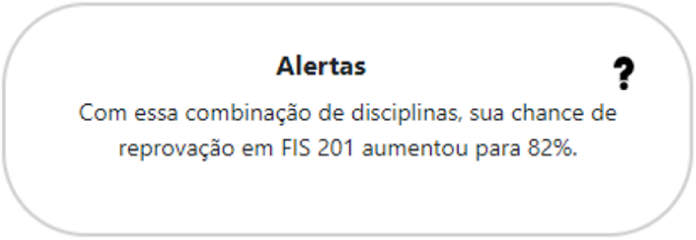

Figura 6.13: Um exemplo de alerta de regras de associação

Caso o usuário queira mais detalhes sobre o padrão mostrado, basta clicar no ícone de ajuda, representado pelo ponto de interrogação, que serão mostrados os antecedentes e consequentes da regra.

Com o intuito de reduzir os elevados índices de reprovação, o protótipo desenvolvido atua no auxílio a tomada de decisão de estudantes e orientadores, no momento do plano de estudos. Ele fornece um panorama da situação acadêmica do usuário para que este tenha mais informações capazes de guiá-lo em suas escolhas. Sabendo a probabilidade geral de reprovação em cada disciplina, quais disciplinas deve cursar prioritariamente para se formar no menor tempo possível e os padrões que intensificam a chance de reprovação em um dado conjunto de disciplinas, o aluno consegue ter uma visão melhor de como será seu semestre e se vale a pena correr os riscos ou não. Ele saberá dosar, o que é melhor para ele em cada momento e tomar melhores decisões em sua jornada acadêmica, visando reduzir suas chances de reprovação.

Este Capítulo expôs como cada módulo contribui para a construção do protótipo final. Este trabalho mostrou que a combinação de várias técnicas de mineração de dados podem fornecer uma visão mais ampla sobre uma situação. Juntas, elas mostram aspectos de um cenário que, se visto apenas por uma perspectiva, podem ser esquecidos alguns detalhes. Para se obter um total entendimento a respeito de uma situação, é necessário observar seus múltiplos aspectos, seus vários pontos de vista e é isso que o protótipo funcional tenta mostrar.

Capítulo 7

Conclusão

Conhecendo os dados de históricos armazenados de 2013 a 2017 pela Universidade Federal de Viçosa, percebemos que algumas disciplinas, com alto índice de reprovação, possuem características similares e talvez uma única proposta possa solucionar o problema para um conjunto de disciplinas assim. Também conseguimos identificar o perfil do estudante da UFV que vem, principalmente, da região Sudeste e apesar de ingressar na universidade com uma nota do ENEM entre 600 a 700, possuem grandes dificuldades com a base matemática.

Analisado os dados da UFV, foi construído um protótipo funcional de um sistema inteligente de apoio na elaboração de plano de estudos que promete auxiliar na tomada de decisão de estudantes e orientadores nas escolhas para o próximo semestre acadêmico. Isso ampara o estudante na medida em que ele é capaz de escolher um conjunto de disciplinas nas quais tem confiança e consiga focar seus esforços de forma mais efetiva para que não se reprove. Além disso, tendo como base os dados da própria universidade, o sistema garante mais segurança para o orientador ao fazer suas recomendações. O protótipo foi construído a partir da união de vários métodos de Ciência da Computação, cada um atuando com um fim específico para atingir, no final, um panorama da situação do estudante.

O sistema possui um modelo de redes neurais, capaz de prever o número provável de reprovações de um estudante em um certo semestre dado o conjunto de disciplinas que opta por cursar. A mineração de padrões frequentes foi incorporada como um módulo de alertas, em que se um estudante se enquadra em uma situação frequente de reprovação dentre os registros da UFV, ele é notificado para tentar evitar essa ocorrência. O protótipo ainda conta com um algoritmo de caminho crítico que informa o número mínimo de períodos que ainda devem ser cursados e uma heurística para planejamento automático de todos os semestres seguintes.

Um próximo passo para a evolução deste trabalho seria a experimentação do protótipo em uma situação real. Podem ser realizados testes com um grupo de alunos usando o protótipo e outro não. Ou ainda, a partir dos dados observados na prática, podem ser feitas simulações dessas situações no protótipo e comparar a previsão de reprovações com o que realmente ocorreu.

Um projeto de ciência de dados é, contudo, uma investigação que abre caminho para muitas outras hipóteses e estudos. Duas análises complementares importantes podem ser feitas, em trabalhos futuros, com os alunos ingressantes, bem como os egressos. Para a primeira, podem ser usados os dados do ENEM para entender quais perfis de alunos estão conseguindo ingressar nas universidades. Na segunda, seria

interessante investigar qual está sendo o destino dos alunos com maiores índices de reprovação, para descobrir qual o impacto do desempenho acadêmico na vida profissional.

Por outro lado, dados adicionais que podem melhorar o desempenho do modelo de previsão, se incorporados à base utilizada, são aqueles relacionados à própria rotina do aluno como a informação se ele utiliza da moradia da UFV ou não, se ele trabalha enquanto estuda, ou até mesmo a forma com que ele interage com os conteúdos em Ambientes Virtuais de Aprendizagem. Eventualmente, outros modelos de aprendizado de máquina podem ser testados, inclusive.

Também podem ser incorporados no protótipo, alertas relacionados aos critérios de desligamento da universidade com o intuito de advertir os estudantes que se enquadram em situações de risco de modo a tomarem decisões mais apropriadas evitando, assim, o desligamento. Na heurística implementada, foi considerado um número máximo de créditos que o aluno consegue cursar, mas seria interessante a possibilidade de inclusão de atividades extracurriculares, o que pode fornecer outros indícios e gerar outros padrões que retratam melhor a rotina dos estudantes.

Outro ponto importante é analisar o rendimento dos estudantes que ingressaram a partir do ENEM contra aqueles que ingressaram antes da implantação do Exame, no intuito de verificar se esse é realmente um bom método de seleção de estudantes, melhor que o vestibular tradicional. O presente trabalho focou em estudantes de graduação, mas o mesmo pode ser adaptado para estudantes da pós-graduação para identificar ou monitorar os problemas relacionados a esse contexto. Alguns outros fatores podem ser considerados na análise, como o impacto do corte de bolsas ou da pandemia de COVID-19 no rendimento desses alunos.

Este trabalho também pode ser expandido com um módulo de recomendação que, com base em trajetórias de sucesso de estudantes similares, consiga fornecer recomendações de planos de estudo. Além de que, para as sugestões já fornecidas, é possível incluir o usuário no processo, que irá classificá-las e os *feedbacks* recolhidos podem ser usados para adaptação das sugestões.

No intuito de fornecer material também para a administração da UFV, podem ser construídos painéis de acompanhamento dos alunos com foco nas reprovações, visualização comparativa entre períodos ou ainda análise de *outliers* e casos críticos para que uma ação possa ser tomada previamente de modo a resgatar o aluno que está passando por dificuldades.

O protótipo desenvolvido, se incorporado aos sistemas acadêmicos de gerenciamento, pode auxiliar cada estudante individualmente e seu orientador a tomarem melhores decisões. O que contribui, em larga escala, para reduzir os índices de reprovação nas instituições de ensino e a entender melhor a deficiência dos alunos. Conhecer melhor os estudantes e perfis das instituições é um passo fundamental para a melhoria contínua do ensino e formação de melhores profissionais.

Este trabalho é multidisciplinar na medida em que aborda métodos de diversas subáreas da Ciência da Computação para construir um pipeline de produto. Sua contribuição é o uso da metodologia para avaliação de modelos que pode ser utilizada em outros trabalhos. Além de propor métricas novas para essa avaliação e mostrar a viabilidade de embarcar técnicas de diversos campos da computação em um único produto.

Referências Bibliográficas

Academy, D. S. (2021). Deep learning book. https://www.deeplearningbook.com.br/.

Agrawal, R., Imielinski, T., and Swami, A. (1993). Mining association rules between sets of items in large databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington D.C.

Agrawal, R. and Srikant, R. (1994). Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, pages 487–499, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Alexander, C., Reese, D., and Harden, J. (1994). Near-critical path analysis of program activity graphs. In *Proceedings of International Workshop on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, pages 308–317.

Athias, L. (2019). *Panorama nacional e internacional da produção de indicadores sociais: estatísticas de governança*. IBGE, Rio de Janeiro.

Baeldung (2020). Baeldung epoch in neural networks. https://www.baeldung.com/cs/epoch-neural-networks#:~:text=An%20epoch%20means%20training%20the,to%20train%20the%20neural%20network. Acessado em 23/05/2021.

Baradwaj, B. K. and Pal, S. (2011). Mining educational data to analyze students performance. *International Journal of Advanced Computer Science and Applications*, 2(6).

Brownlee, J. (2017). Machine Learning Mastery gentle introduction to the adam optimization algorithm for deep learning. https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/. Acessado em 23/05/2021.

Carvalho, L., Santos, A., Nakamura, F., and Oliveira, E. (2019). Detecção precoce de evasão em cursos de graduação presencial em computação: um estudo preliminar. In *Anais do XXVII Workshop sobre Educação em Computação*, pages 233–243, Porto Alegre, RS, Brasil. SBC.

Cerqueira, E. O. d., Andrade, J. C. d., Poppi, R. J., and Mello, C. (2001). Redes neurais e suas aplicações em calibração multivariada. *Química Nova*, 24:864 – 873.

Chollet, F. (2017). *Deep Learning with Python*. Manning Publications Co., USA, 1st edition.

Chollet, F. et al. (2015). Keras. https://keras.io.

Costa, L., Souza, C., Inocêncio, A., and Jr., E. B. (2019). Um mapeamento sistemático sobre métodos de identificação preditiva de alunos com risco de reprovação em educação de computação. In *Anais do XXVII Workshop sobre Educação em Computação*, pages 378–388, Porto Alegre, RS, Brasil. SBC.

CS, S. (2021). Commonly used activation functions. https://cs231n.github.io/neural-networks-1/#actfun. Acessado em 16/06/2021.

Cui, Y., Chu, M.-W., and Chen, F. (2019). Article: Analyzing student process data in game-based assessments with bayesian knowledge tracing and dynamic bayesian networks. *JEDM | Journal of Educational Data Mining*, 11(1):80–100.

de Souza, E. F. (2007). Comparação e escolha de agrupamentos : uma proposta utilizando a entropia – São Paulo , maio de 2007 – Comparação e escolha de agrupamentos : uma proposta utilizando a entropia. *Dissertação (Mestrado) em Ciências do Instituto de Matemática e Estatísctica - USP*, page 76.

Diogo, M. F., Raymundo, L. d. S., Wilhelm, F. A., Andrade, S. P. C. d., Lorenzo, F. M., Rost, F. T., and Bardagi, M. P. (2016). Percepção de coordenadores de curso superior sobre evasão, reprovação e estratégias preventivas. *Avaliação: Revista da Avaliação da Educação Superior (Campinas)*, 21:125 – 151.

Du, F., Plaisant, C., Spring, N., and Shneiderman, B. (2016). Eventaction: Visual analytics for temporal event sequence recommendation. In *2016 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 61–70.

Filho, R. B. S. and Araújo, R. M. D. L. (2017). Evasão e abandono escolar na educação básica no brasil: fatores, causas e possíveis consequências. *Educação Por Escrito*, 8(1):35.

Fornari, L. T. (2010). Re fl exões acerca da reprovação e evasão escolar e os determinantes do capital. , pages 112–124.

Gomes, D., Ribeiro, M. H., Comarela, G., and Pereira, G. (2020). Failure analysis in university and computer science contexts with data mining. In *Anais do XXVIII Workshop sobre Educação em Computação*, pages 71–75, Porto Alegre, RS, Brasil. SBC.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

Han, J., Kamber, M., and Pei, J. (2012). *Data mining concepts and techniques, third edition*. Morgan Kaufmann Publishers, Waltham, Mass.

INEP (2019). Notas Estatísticas. Technical report, Ministerio Da Educação.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Nargesian, F., Samulowitz, H., Khurana, U., Khalil, E., and Turaga, D. (2017). Learning feature engineering for classification. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pages 2529–2535.

OCDE (2019). PISA 2105 - Brazil - Contry Note. *World Population Policies 2015*, pages 9–21.

O'Grady, K. E. (1982). Measures of explained variance: Cautions and limitations. *Psychological Bulletin*, 92(3):766–777.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Pelaez, K., Levine, R., Fan, J., Guarcello, M., and Laumakis, M. (2019). Article: Using a latent class forest to identify at-risk students in higher education. *JEDM | Journal of Educational Data Mining*, 11(1):18–46.

Polyak, B. (1964). Some methods of speeding up the convergence of iteration methods. *Ussr Computational Mathematics and Mathematical Physics*, 4:1–17.

Rahman, M. S. (2017). *Basic Graph Theory*. Springer International Publishing.

Raji, M., Duggan, J., DeCotes, B., Huang, J., and Zanden, B. T. V. (2017). Visual progression analysis of student records data. *2017 IEEE Visualization in Data Science (VDS)*, pages 31–38.

Ribeiro, A. M. et al. (2020). Um estudo comparativo entre cinco métodos de otimização aplicados em uma rnc voltada ao diagnóstico do glaucoma. *Revista de Sistemas e Computação-RSC*, 10(1).

Romero, C. and Ventura, S. (2007). Educational data mining: A survey from 1995 to 2005. *Expert Systems with Applications*, 33:135–146.

Russell, S. and Norvig, P. (2009). *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, USA, 3rd edition.

Sa, C. D. (2021). Acceleration and momentum. http://www.cs.cornell.edu/courses/cs6787/2017fa/Lecture3.pdf. Acessado em 06/06/2021.

Saraiva, D., Pereira, S., Gallindo, E., Braga, R., and Oliveira, C. (2019). Uma proposta para predição de risco de evasão de estudantes em um curso técnico em informática. In *Anais do XXVII Workshop sobre Educação em Computação*, pages 319–333, Porto Alegre, RS, Brasil. SBC.

Schulz, M. (2005). Extracting critical path graphs from mpi applications. In *2005 IEEE International Conference on Cluster Computing*, pages 1–10.

Souza, C., Silva, D., Gessinger, C., and Um, R. (2017). estudo sobre evasao no ensino superior do brasil nos últimos dez anos. *Congresos CLABES*, 9.

Srinidhi, S. (2018). Label encoder vs. one hot encoder in machine learning. https://blog.contactsunny.com/data-science/label-encoder-vs-one-hot-encoder-in-machine-learning. Acessado em 15/08/2021.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Sutton, R. S. and Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. The MIT Press, second edition.

Waskom, M. L. (2021). seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021.

Wasserman, L. (2004). *Models, Statistical Inference and Learning*, pages 87–96. Springer New York, New York, NY.

Xie, Z. and Yan, J. (2008). Kernel density estimation of traffic accidents in a network space. *Computers, Environment and Urban Systems*, 32(5):396–406.

Zaki, M. J. and Jr, W. M. (2014). *Data Mining and Analysis: Fundamental Concepts and Algorithms*. Cambridge University Press, New York, NY, USA.

Zhang, X., Zou, Y., and Shi, W. (2017). Dilated convolution neural network with leakyrelu for environmental sound classification. In *2017 22nd International Conference on Digital Signal Processing (DSP)*, pages 1–5.

Zingaro, D. and Oztok, M. (2012). Interaction in an asynchronous online course: a synthesis of quantitative predictors. *Online Learning*, 16(4).

Zou, J., Han, Y., and So, S.-S. (2008). Overview of artificial neural networks. *Artificial Neural Networks*, pages 14–22.

Apêndice A

Simulação da heurística

A sequência de figuras a seguir simula o funcionamento da heurística, um algoritmo guloso que seleciona um conjunto de disciplinas para serem cursadas a cada período, descrito no Capítulo 6.

Como vimos, o algoritmo sugere as disciplinas prioritárias para cada período, levando em consideração suas dependências de pré-requisitos e correquisitos. Além de dar prioridade para aquelas disciplinas que fazem parte do caminho crítico, visando antecipar a formatura do estudante. Uma disciplina só é sugerida em um dado período se seus pré-requisitos foram cumpridos nos períodos anteriores. A critério de desempate, também são consideradas outras características para a escolha de disciplinas como, por exemplo, a sua carga horária.

Os nós verdes representam as disciplinas já concluídas pelo estudante e os rosas são as disciplinas sugeridas pela heurística. As arestas contínuas representam relacionamentos de pré-requisitos e as linhas tracejadas, correquisitos. O número entre parênteses ao lado do código da disciplina, indica se a disciplina é ofertada em períodos ímpares (1), pares (2) ou ao longo de todo o ano (0).

Suponha um estudante que acabou de cursar seu quarto período de faculdade. Ele possui algumas disciplinas já cursadas durante sua jornada acadêmica como mostra a figura A.1.

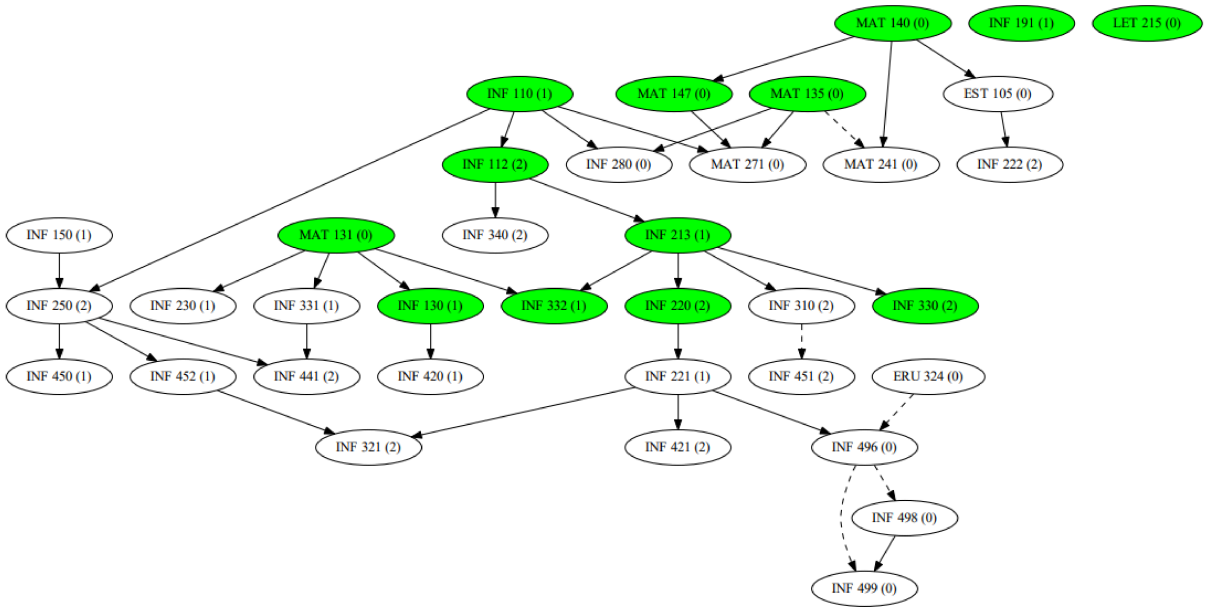

Figura A.1: Situação do aluno ao final do o 4º período.

Suponha que esse estudante não queira cursar mais que 20 créditos, o que corresponde a, aproximadamente, 5 disciplinas, se for considerar que o padrão de uma disciplina na UFV é ter 4 créditos. A sugestão para o próximo período, para esse caso, seria aquela mostrada na figura A.2.

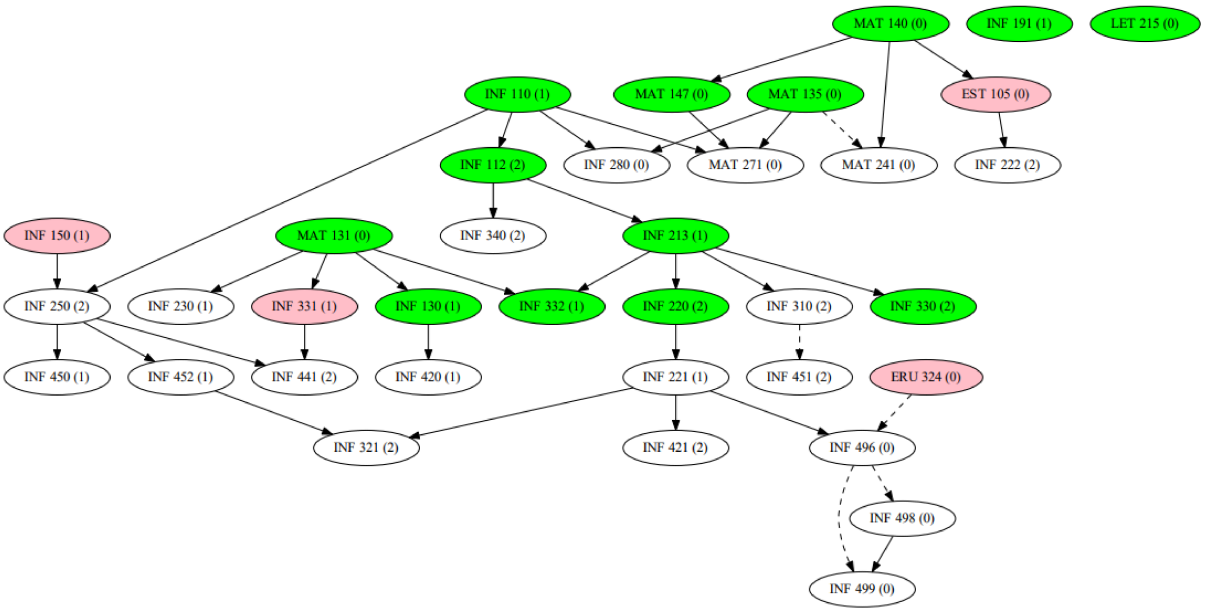

Figura A.2: Disciplinas escolhidas pela heurística para o 5º período.

Nesse caso, a heurística sugere, para o quinto período, que ele curse INF 150 e ERU 324, que não possuem pré-requisitos e EST 105 e INF 331 que possuem seus pré-requisitos já concluídos. Vale ressaltar que essas disciplinas podem ser recomendadas para o quinto período, pois são ofertadas em períodos ímpares (1) ou em qualquer período (0). A figura A.3, mostra as sugestões que a heurística retorna para o sexto período.

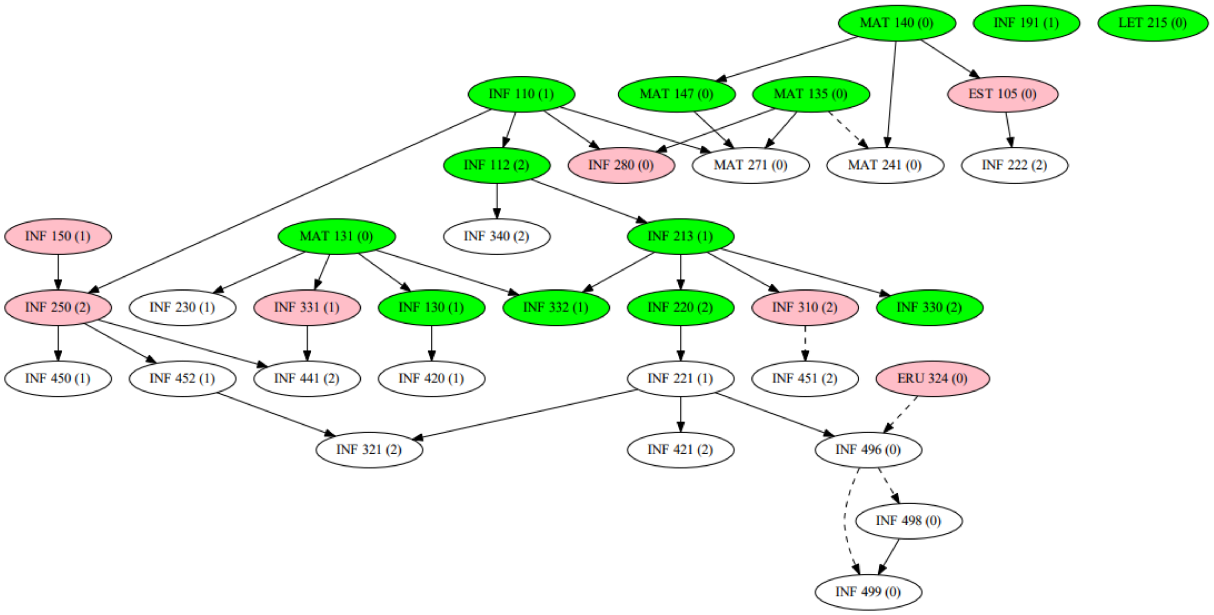

Figura A.3: Disciplinas escolhidas pela heurística para o 6º período.

Nesse ponto são sugeridas as disciplinas INF 250 e INF 310 que são oferecidas em períodos pares e INF 280 que é ofertada em todos os períodos. Novamente, essas disciplinas só podem ser sugeridas porque seus pré-requisitos já foram concluídos pelo estudante. A figura A.4 repete o processo e mostra as sugestões para o sétimo período do estudante.

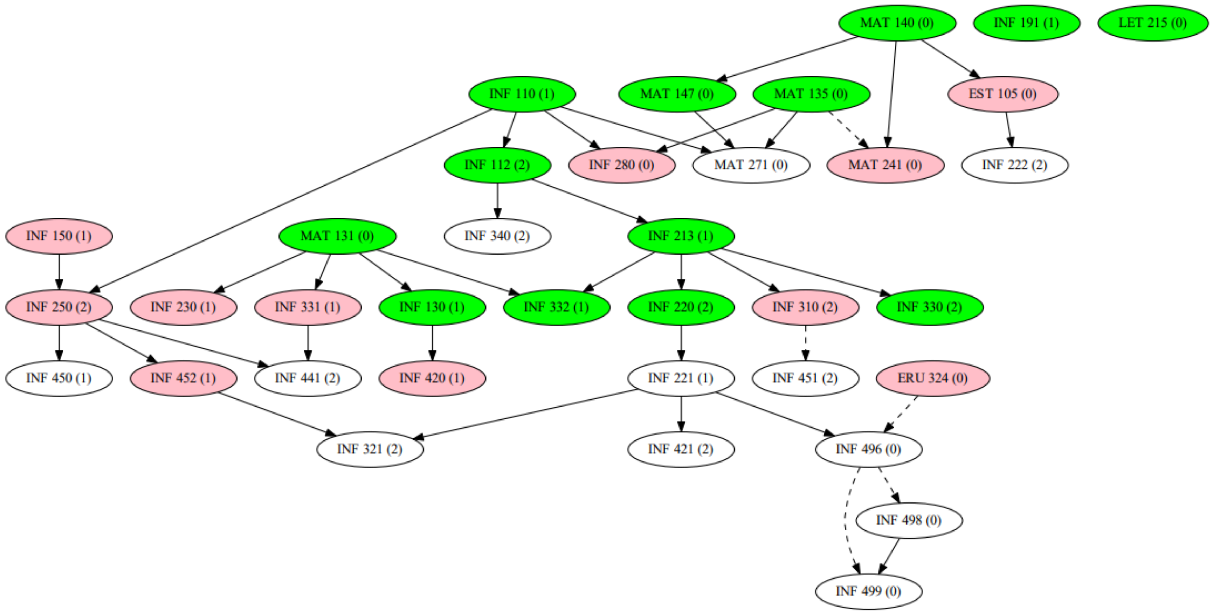

Figura A.4: Disciplinas escolhidas pela heurística para o 7º período.

Para o sétimo período, é sugerida a MAT 241 que possui tanto um pré-requisito cumprido quanto um correquisito também já concluído e é ofertada ao longo de todo o ano. Também fazem parte da sugestão INF 230, INF 452 e INF 420, todas sem dependências de pré-requisitos pendentes e oferecidas em períodos ímpares. A seguir, para o oitavo período é sugerido o cenário mostrado na figura A.5.

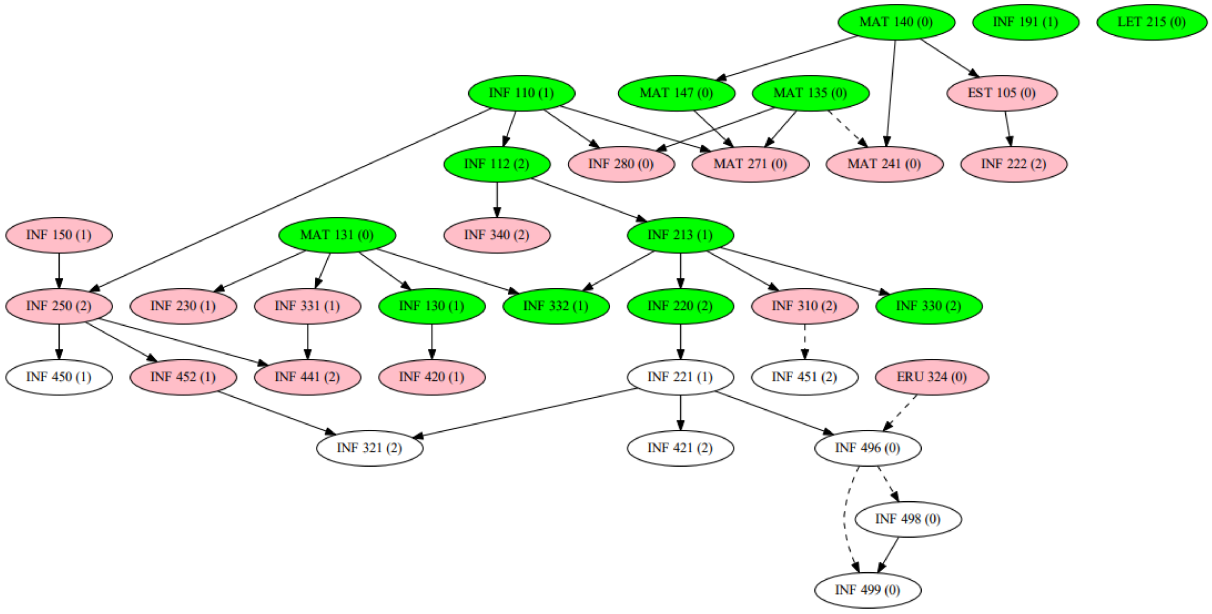

Figura A.5: Disciplinas escolhidas pela heurística para o 8º período.

Pro oitavo período são selecionadas aquelas disciplinas ofertadas em períodos pares (INF 222, INF 340, INF 441) e MAT 271, oferecida o ano todo, que não possuem pendências. Já o cenário simulado para o nono período, é ilustrado na figura A.6.

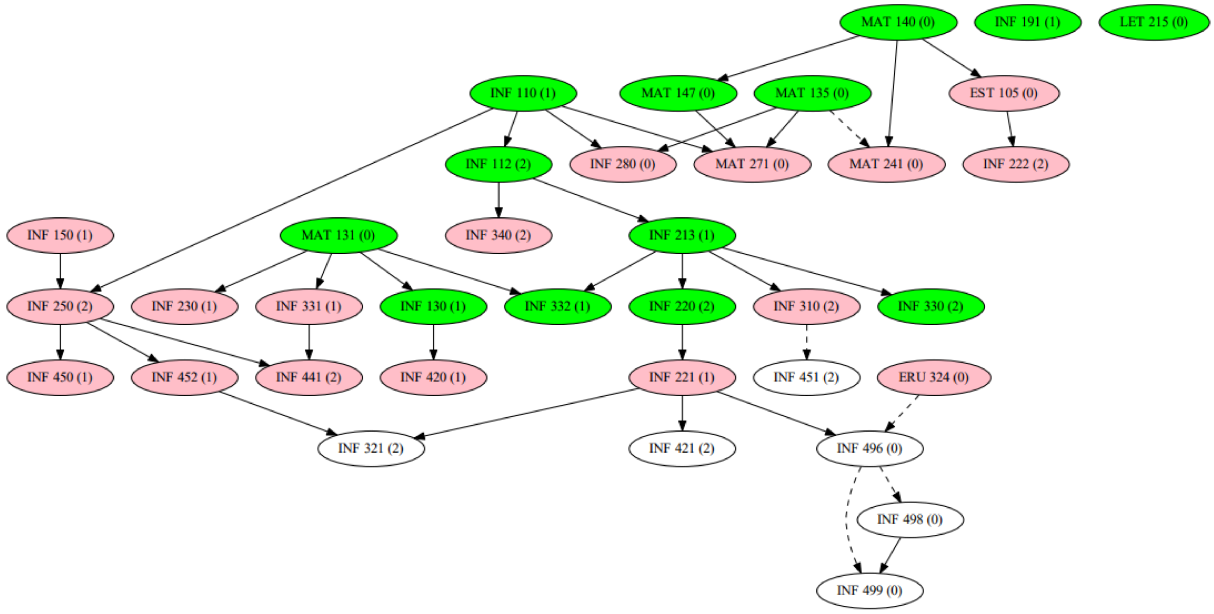

Figura A.6: Disciplinas escolhidas pela heurística para o 9º período.

Nesse ponto apenas duas disciplinas podem ser sugeridas (INF 450 e INF 221), já que são as únicas que são ofertadas em períodos ímpares, sem pendências. INF 469 é ofertada ao longo do ano todo, mas não pode ser selecionada já que ela possui INF 221 como pré-requisito. A figura A.7 mostra a sugestão para o próximo período.

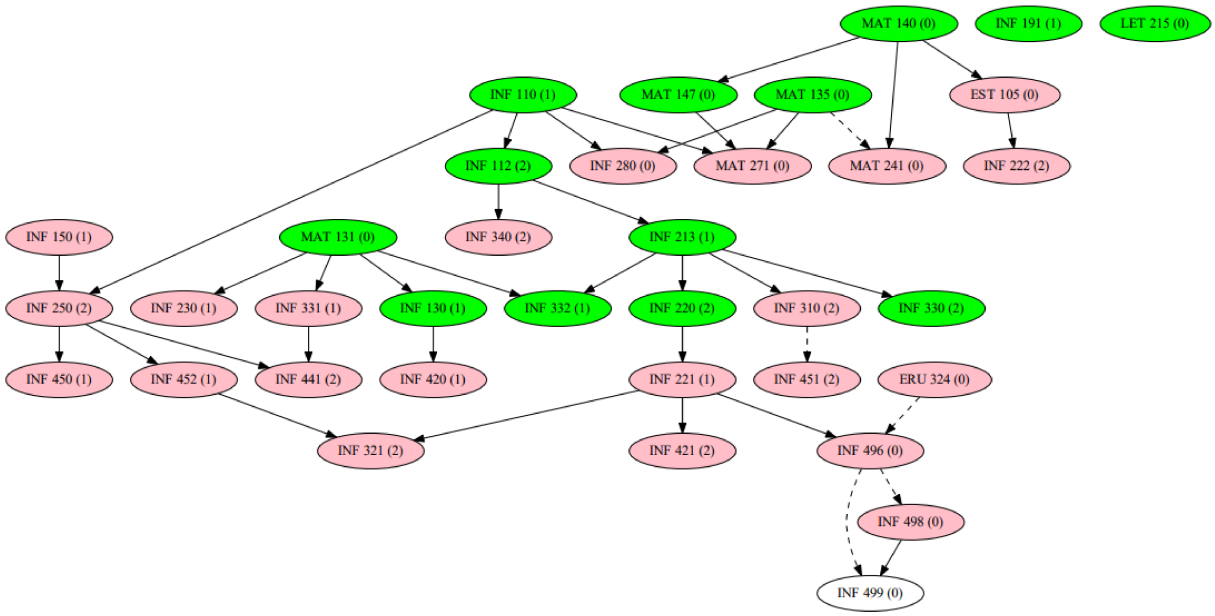

Figura A.7: Disciplinas escolhidas pela heurística para o 10º período.

Aqui INF 496 já pode ser sugerida, pois seu pré-requisito teria sido cumprido no período anterior. INF 498 também pode ser sugerida já que a relação de dependência dela com INF 496 é de correquisito, assim, elas podem ser cursadas juntas. Também são incluídas na sugestão as disciplinas exclusivas de período par (INF 451, INF 321 e INF 421). Para o exemplo mostrado, no último período sobraria apenas uma disciplina obrigatória a ser cursada: INF 499. É uma oportunidade para o estudante completar a carga com as disciplinas optativas que faltam, se necessário. Após a execução do algoritmo, seria constatado que o estudante que se enquadrar na situação exemplificada, poderia concluir seus estudos em 7 períodos. Sabendo disso, o estudante consegue se planejar melhor e focar seus esforços para evitar reprovações, o que atrasaria seu período de conclusão.