Matheus Agio Nerone
Victor Mocelin

Sistema de recomendação de matrículas baseado em técnicas de aprendizado de máquina

Trabalho apresentado como requisito parcial à conclusão do Curso de Bacharelado em Ciência da Computação, setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Ciência da Computação*.

Orientador: André L. Vignatti.

Curitiba PR
2016

Resumo

Neste trabalho apresentamos um sistema de recomendação para disciplinas dos cursos de Ciência da Computação e Informática Biomédica. A ideia surgiu após termos percebido ao longo de nossa graduação, um grande número de reprovações e cancelamento de matrículas por parte dos nossos colegas. A maior finalidade desse trabalho é dar um primeiro passo em direção a um sistema de acompanhamento que possa ajudar alunos a terem uma experiência melhor dentro da faculdade. São utilizados dados dos últimos quinze anos dos alunos do Departamento de Informática. Nossa proposta para o sistema de recomendação é baseado nas notas finais dos alunos, ou seja, o sistema tenta prever qual será a nota final para disciplinas que o aluno ainda não realizou, as ordena de forma decrescente e recomenda as primeiras. Para realizar a predição de notas utilizamos métodos de Aprendizado de Máquina, como *KNN*, *Random Forest* e *SVM*. Escolhemos como características: nota final média do aluno, frequência média do aluno, nota final média da disciplina, número de reprovações, média do estudante no último ano, frequência do estudante no último ano. Mesmo que as características tenham relação com a nota final, elas não possuíram poder de discriminação suficiente para fazer com que os resultados fossem satisfatórios. Entretanto, conseguimos uma taxa de acerto razoável, cerca de 70%, com relação a se um aluno iria reprovar ou não se escolhesse fazer aquela disciplina.

Palavras-chave: Sistema de recomendação, Aprendizado de máquina.

Abstract

In this paper we present a recommender system about Computer Science and Biomedical Informatic courses. The idea was formed after seeing a lot of our colleagues cancel or fail the courses along the graduation years. The biggest goal of this project is to take a first step in the direction of a monitoring system to help the students have a better experience in college. The data used is from the last 15 years of the students from the Computing Department. Our proposal for the recommendation system is based on the students final grade, i.e., the system tries to predict which will be the final grade for courses that the student has not already taken, then these grades are sorted in descending order and the best recommendations are the first ones. To do the final grade predictions we used Machine Learning methods, like KNN, Random Forest and SVM. The features we had chosen were: students average final grade, students average frequency, course average final grade, students number of flunks, student last year average grade, student last year average frequency. Even though the features had a relation with the final grade, they did not had enough discriminative power to have good results. However, we managed to get a reasonable accuracy, around 70%, when tring to predict if a student would fail or not in a course.

Keywords: Recomendation System, Machine Learning.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Acrônimos

UFPR	Universidade Federal do Paraná
SIE	Sistema de Informações para o Ensino
TI	Tecnologia da Informação
KNN	K Nearest Neighbors
PET	Programa de Educação Tutorial
SVM	Support Vector Machine
SGBD	Sistema de Gerenciamento de Banco de Dados
RMSE	Root Mean Square Error

Lista de Símbolos

ϵ	variável de folga
Ψ	conjunto de treino
ρ	um registro completo
Ω	algoritmo de aprendizado de máquina abstrato
Ω_Ψ	o resultado do treinamento de Ω com o conjunto de treino Ψ
α	um aluno
ξ	o conjunto de registros incompletos de α

Capítulo 1

Introdução

Como estudantes de uma graduação situada no Departamento de Informática da UFPR, tivemos contato com alunos de Ciência da Computação e Informática Biomédica ao longo da vida acadêmica. Em tais cursos é muito comum que alunos que ingressaram em diferentes anos na universidade cursem uma mesma disciplina, então, temos uma visão relativamente ampla do desempenho dos alunos. Acreditamos que muitos casos de reprovações, abandonos ou trancamentos de uma disciplina são consequência da escolha de disciplinas para aquele semestre.

Ao fazer uma análise básica do desempenho acadêmico desses alunos, foi possível calcular que a média da nota final é de 51.29 para os dois cursos. Outro dado relevante que encontramos é que cerca de 37% das matrículas resultam em reprovações por frequência ou por nota. Portanto, sabendo que a média de aprovação direta é 70, e aprovação com exame é 50, o desempenho médio dos alunos não é o ideal e existe um desafio para fazer com que o desempenho geral dos alunos melhore.

Investigar esse problema é relevante pois pode fazer com que alunos obtenham um rendimento acadêmico melhor, gerando mais benefícios dentro da universidade, por exemplo, com maior oportunidade de bolsas e intercâmbio. Além disso, do ponto de vista das coordenações dos cursos, uma escolha por parte dos alunos que minimiza o desempenho ruim nas disciplinas, possibilita uma alocação de professores e turmas mais precisa, diminuindo problemas de turmas vazias ou lotadas. Por fim, um desempenho melhor aumenta o reconhecimento dos cursos e da universidade como um todo, proporcionando uma visão melhor da instituição.

Neste sentido, a nossa proposta neste trabalho é melhorar o desempenho dos alunos auxiliando a escolha de disciplinas durante o período de solicitação de matricula, que acontece no início de cada semestre letivo. Este auxílio será feito através de um sistema de recomendação de disciplinas baseado no histórico particular do aluno além do histórico geral dos alunos. A recomendação considera fatores que possivelmente possuem uma relação com a nota final, ou seja, podem influenciar a nota tanto de maneira positiva quanto negativa.

A concretização da nossa proposta pode contribuir para a criação ou ampliação de um sistema de acompanhamento dos alunos. Dentro de tal sistema hipotético, entre várias funcionalidades possíveis, uma delas poderia ser a recomendação de disciplinas aliada com

uma análise estatística do desempenho dos alunos. Desta forma os órgãos administrativos da universidade podem acompanhar de forma mais clara os dados atualizados obtidos no sistema.

Este trabalho está dividido em cinco capítulos: introdução, conceitos preliminares, proposta, experimentação e validação e conclusão e trabalhos futuros. Neste primeiro capítulo, apresentamos o desafio, motivação, proposta e contribuição do nosso trabalho. No segundo, serão explicados todos os conceitos necessários para se entender o que será discutido no decorrer do trabalho. No capítulo três, será explicada a nossa proposta e como foi realizada sua implementação, com seus detalhes específicos. O quarto capítulo tem como objetivo justificar as escolhas realizadas na implementação, apontando as relações entre os dados utilizados. Além disso, serão apresentados os resultados dos experimentos realizados. No último capítulo, vamos apresentar a conclusão que obtivemos do trabalho, baseada nos resultados alcançados, além de listar trabalhos futuros que podem ser desenvolvidos a partir desse.

Capítulo 2

Conceitos Preliminares

Neste capítulo vamos apresentar o conceito de um sistema de recomendação, conceitos de aprendizado de máquina, e por último, os algoritmos de aprendizado de máquina, que utilizamos para desenvolver nosso sistema de recomendação.

2.1 Sistema de Recomendação

A função de um sistema de recomendação é assistir os usuários a escolherem itens de seu interesse, baseado no seu histórico de consumo ou interesse desses itens, como livros, filmes, vídeos, músicas, entre outros, que ainda não foram vistos e/ou avaliados por aquela pessoa. Sistemas como esse já vem sendo utilizados em plataformas de comércio eletrônico e na indústria de TI, e.g., *Amazon*, *Netflix* e *Google News*, e são um dos responsáveis pelo sucesso dessas plataformas [Wen, 2008].

Um sistema de recomendação funciona da seguinte maneira [Wen, 2008]:

1. Constrói-se um perfil do usuário baseado em suas ações passadas.
2. Para cada item que o usuário ainda não avaliou é realizada uma comparação entre as características do item e o perfil do usuário.
3. Essa comparação gera uma "nota", que seria a avaliação que o usuário faria sobre aquele item.

Por exemplo, em uma plataforma de filmes e séries, como a Netflix, primeiramente se constrói o perfil do usuário baseado nos filmes que ele já assistiu e nas avaliações que ele fez. Em seguida, para cada filme que ainda não foi avaliado pelo usuário é realizada uma comparação entre as características do filme e o perfil do usuário, gerando a predição de uma nota. Por fim, basta ordenar de forma decrescente estas predições de notas, assim as primeiras serão as melhores recomendações a serem feitas.

De acordo com [Wen, 2008], um sistema de recomendação pode possuir uma abordagem baseada em conteúdo ou em filtro colaborativo ou ambos, com base na escolha das características dos usuários e dos itens[1].

Filtro baseado em conteúdo

Filtro que utiliza as características de itens que o usuário gostou no passado para gerar novas recomendações. O site de comércio eletrônico *amazon.com* utiliza esse filtro para recomendar itens similares aos que o usuário já comprou [Felfernig et al., 2013].

Filtro colaborativo

A ideia é de que recomendações são feitas baseadas na opinião de pessoas cujo gosto é parecido com o seu, o que seria chamado popularmente de "boca-a-boca". Por exemplo, se um usuário João comprou filmes parecidos com os que Marta comprou, então João e Marta são usuários com gostos parecidos, o que faz com que as recomendações para João sejam os filmes que Marta comprou e gostou mas que João não possui [Felfernig et al., 2013].

2.2 Conceitos de Aprendizado de Máquina

No contexto de recomendação, a literatura nos indica a utilização de algoritmos de aprendizado de máquina como uma solução. Nesta seção, serão apresentados os principais conceitos deste tema, assim como alguns exemplos, que serão importantes para entender os algoritmos na Seção 2.3.

Podemos nos basear em [Mohri et al., 2012] para apresentar os principais conceitos em aprendizado de máquina:

- Exemplos: Itens ou instâncias de dados usados para aprendizado ou avaliação.
- Características: O conjunto de atributos, comumente representados como um vetor, associado a um exemplo.
- Classes: Valores ou categorias atribuídos para exemplos. Em problemas de classificação, é atribuída uma categoria específica para cada exemplo.
- Conjunto de Treino: Exemplos utilizados para treinar um algoritmo de aprendizagem. Este conjunto de exemplos varia de acordo com diferentes cenários.
- Conjunto de Teste: Exemplos usados para avaliar a performance de um algoritmo de aprendizagem. O conjunto de teste é separado do conjunto de treino, e não fica disponível

[1]Características de itens são por exemplo, palavras-chaves, titulo, autor. Já características de usuário são, por exemplo, avaliações que foram feitas sobre um item, itens visualizados, etc...

no estágio de treino. Neste conjunto, os exemplos já estão classificados corretamente e são comparados com o que foi obtido pela predição do algoritmo, medindo assim, a sua performance.

- Classificação: Processo de classificar cada item. Por exemplo, notícias podem ser classificadas em esportivas, políticas, culturais, etc.
- Regressão: Processo de predição de um valor real para cada item. Exemplos de regressão incluem predição de variáveis de economia, predição de valor de um imóvel, etc.

2.3 Algoritmos

Agora vamos apresentar os algoritmos que utilizamos em nossa pesquisa. Começamos apresentando o algoritmo Floresta Aleatória, em seguida mostramos o SVM (Support Vector Machine), e por último o KNN (K-Nearest Neighbors).

2.3.1 Floresta Aleatória

Conforme [Breiman, 2001], floresta aleatória é um classificador que consiste em uma coleção de árvores de decisão onde, dado uma entrada X, a predição é a classe mais votada dentre as predições feitas por suas árvores. Dado essa definição, antes de prosseguirmos com a explicação de floresta aleatória, se faz necessário esclarecer o que são e como funcionam árvores de decisão.

De acordo com [Mohri et al., 2012], árvore de decisão é um tipo de classificador utilizado em classificação multi-classe[2], e embora normalmente sua taxa de acerto não seja tão boa, ele pode ser utilizado em conjunto com outros métodos de aprendizado de máquina para gerar um classificador mais efetivo.

Para predizer a classe de uma entrada válida qualquer, começamos na raiz da árvore e a percorremos até encontrarmos uma folha. Então associamos a entrada com a classe encontrada [Mohri et al., 2012]. Por exemplo, na figura 2.1 a instância

$$(Clima = Ensolarado, Temperatura = Quente, Umidade = Alta, Vento = Forte)$$

percorreria o ramo mais a esquerda da árvore, sendo classificada como Não, i.e, a predição é de que não se deve jogar Tênis naquele dia.

Dado que uma floresta aleatória é um conjunto de árvores de decisão, então para se realizar o processo de classificação de um exemplo, basta executá-lo em cada uma das árvores da floresta. Cada árvore retorna uma classificação, assim, dizemos que a árvore "vota" por aquela

[2]Classificação multi-classe ocorre quando existem mais de duas possíveis classes que um exemplo pode assumir, e.g., a entrada X pode ser classificada em *Vermelho, Verde, Amarelo ou Azul*.

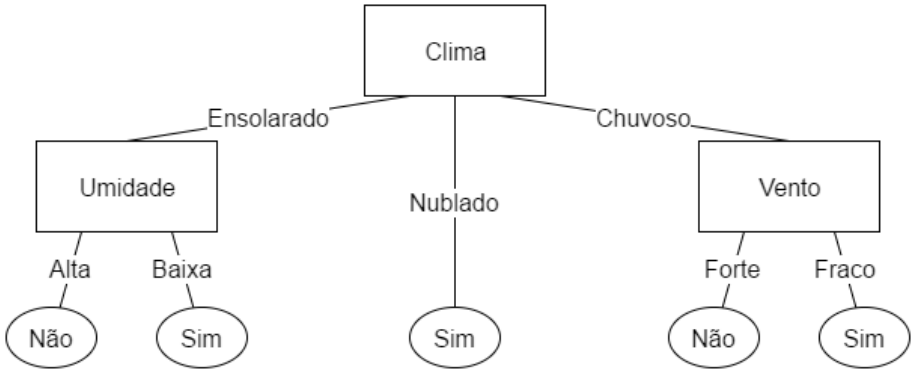

Figura 2.1: Uma árvore de decisão para o conceito de jogar tênis. Um exemplo é classificado percorrendo a árvore até chegar em uma folha, então se retorna a classe associada a essa folha (Neste caso, Sim ou Não). (Adaptado de [Mitchell, 1997])

classe. Finalmente, a floresta escolhe a classe com o maior número de "votos". O algoritmo é chamado de floresta aleatória pelo fato de que, durante o treinamento cada árvore de decisão seleciona de forma aleatória algumas das características.

2.3.2 SVM

Suport Vector Machines (SVMs) é um dos algoritmos modernos de classificação mais efetivos no aprendizado de máquina [Mohri et al., 2012]. Vamos apresentar por primeiro, de maneira simplificada, o problema da classificação linear, seguido pela explicação do algoritmo para o caso em que os dados são linearmente separáveis e o caso em que os dados não são linearmente separáveis.

Classificação linear

O problema da classificação linear consiste em receber um conjunto de treino, que representa vários pontos em um espaço e encontrar uma função que seja um classificador binário dos pontos. Tal função divide o espaço entre duas classes, classificando assim, cada ponto em uma das duas classes possíveis.

Existem infinitas funções possíveis como solução para este problema, o que levanta questionamentos como: qual é a melhor função para separar os dados, como comparar uma função com outra para saber se é melhor.

No contexto do SVM, os dados podem estar contidos em um espaço de várias dimensões. Dessa forma, a função de separação passa a ser um hiperplano que separa os dados. Os questionamentos mencionados se mantém e o aspecto relevante, no caso do SVM, é a margem entre os dados de cada classe, de forma que o objetivo é maximizar margem.

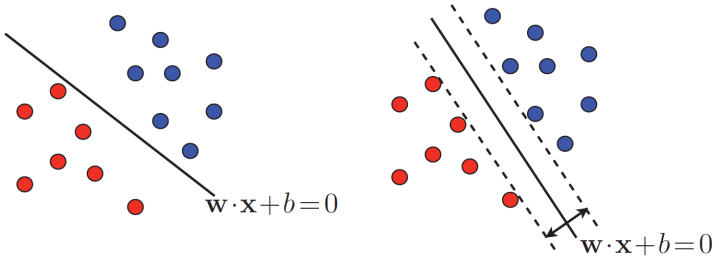

Figura 2.2: Exemplo de duas possíveis funções para separar os dados. A separação da direita maximiza a margem entre os dados de cada classe. Fonte: [Mohri et al., 2012]

Caso linearmente separável

Neste caso, assumimos que o conjunto de treino pode ser linearmente separável, ou seja, existe ao menos um hiperplano que separa os dados em duas possíveis classes, como ilustrado na Figura 2.2. Existem infinitos hiperplanos válidos, então, a solução retornada pelo SVM é aquela com a maior margem, ou seja, o hiperplano está posicionado de forma que a sua distância para os pontos mais próximos de cada uma das classes é a maior possível. Na figura 2.2, podemos ver esta solução do lado direito.

Existe um grande embasamento teórico e científico acerca da maximização da margem do hiperplano até os dados. Por isso, é possível afirmar que a solução do SVM é a escolha "mais segura" para separar os dados. Um exemplo de teste é classificado corretamente por um hiperplano que separa o espaço com margem p mesmo quando o ponto está dentro da distância p.

Por exemplo, na Figura 2.2, a distância p é a distância da fronteira dos dados (linha tracejada) até o hiperplano de separação (linha contínua). Dado um exemplo que pertence à classe azul, mas está posicionado entre a fronteira dos dados e o hiperplano, então ele vai ser corretamente classificado como azul, pois o SVM prevê a maior margem para os dados de cada classe.

Para calcular o melhor hiperplano, são selecionados alguns exemplos do conjunto de treino, onde tais exemplos passam a ser chamados de vetores de suporte. A função do vetor de suporte é auxiliar o cálculo da distância entre as classes, com a finalidade de gerar a margem dos dados.

Caso não linearmente separável

Na prática a maior parte dos conjuntos de dados não são linearmente separáveis, ou seja, para qualquer hiperplano gerado, existe um exemplo no conjunto de treino que estará no lugar errado (*outlier*).

Para encontrar um plano neste caso, deve-se criar uma regra mais flexível em relação ao caso linearmente separável. Isso é feito ao se calcular uma nova propriedade, que pode ser

chamada de variável de folga (*slack variables*), ela mede a distância pela qual um vetor viola a margem dos dados. Na Figura 2.3, vemos esta variável representada por ξ_i.

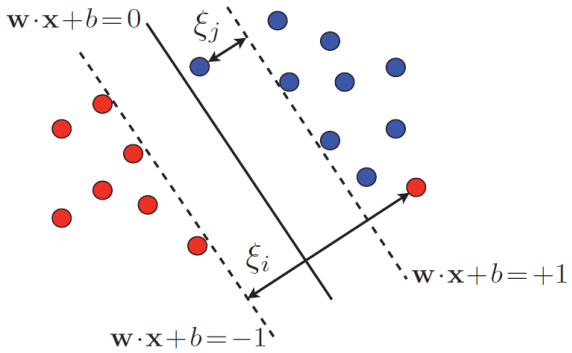

Figura 2.3: Exemplo de um conjunto de dados que não são linearmente separáveis. Fonte: [Mohri et al., 2012]

Para o cálculo do melhor hiperplano neste caso, pode-se definir diferentes abordagens. Um exemplo é ignorar os *outliers* do conjunto de treino, e assim, os dados passam a ser linearmente separáveis. Outra abordagem, é calcular o plano que minimiza o erro empírico, ou seja, considerar os *outliers* como parte relevante para calcular o hiperplano.

O problema neste caso é que dois fatores devem ser considerados. De um lado, deseja-se diminuir a soma dos erros gerados pelos *outliers*, calculado pela soma de todos os ξ_i. Por outro lado, buscamos pelo hiperplano que forneça a maior margem para os dados de cada classe, o que pode aumentar o número de *outliers*.

2.3.3 KNN (K Nearest Neighbors)

De acordo com [Mitchell, 1997], o algoritmo mais básico baseado em instância[3] é o KNN. Ele não possui fase de treinamento por construção e o processo de classificação de novos exemplares é bastante simples. A seguir vamos explicar o conceito do algoritmo, como a distância entre dois exemplos é medida e como é o processo de classificação.

Conceito e medida da distância

Segundo [Mitchell, 1997], o algoritmo trabalha com a suposição de que todos os exemplos correspondem a pontos em um espaço n dimensional $\mathbb{R}^n$. Dessa forma os vizinhos próximos de uma instância são definidos em termos de suas distâncias euclidianas. Por exemplo, seja x um exemplo arbitrário descrito pelo vetor de características $\langle a_1(x), a_2(x), a_3(x), ..., a_n(x)\rangle$

[3]Algoritmos baseados em instância são aqueles que simplesmente guardam o conjunto de treino sem realizar nenhuma ação sobre ele, ou seja, sem realizar algum treinamento.

onde a_r(x) representa a característica r da instância x. Definimos então a distância entre dois exemplos, x_i e x_j como

$$d(x_i, x_j) = \sqrt{\sum_{r=1}^{n} (a_r(x_i) - a_r(x_j))^2} \tag{2.1}$$

É válido notar que a distância euclidiana é somente um das maneiras de se calcular a distância, existem outras formas como diferença de cossenos, distância de Hamming, distância de Manhattan, entre outros. Cada qual funciona melhor para um conjunto de dados.

Classificação

A seguir vamos descrever como funciona o processo de classificação de um exemplo novo. E finalmente através da figura 2.4 podemos entender como a mudança do K pode afetar esse processo.

- Processo de classificação

  1. São selecionados os K vizinhos mais próximos do exemplo.
  2. Cada vizinho "vota" em sua respectiva classe.
  3. A classe com mais votos é a escolhida para ser associada ao exemplo.

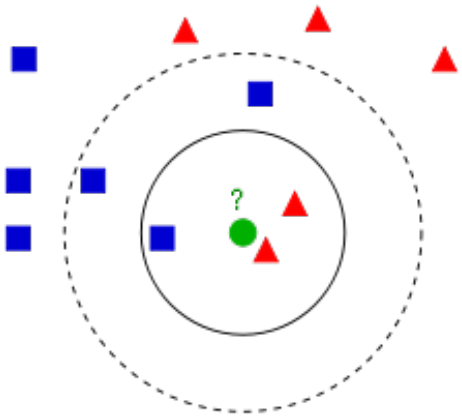

Figura 2.4: Exemplo de como a mudança do K afeta a classificação. Se o K for igual a 3, a bola verde será classificada como Triângulo Vermelho, porém, se aumentarmos o K para 5 a sua classificação será Quadrado Azul. Fonte: `https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm`

2.4 Conclusão

Neste capítulo apresentamos todos os conceitos necessário para que se possa entender o restante do trabalho. Primeiramente, expomos os conceitos de sistema de recomendação e de aprendizagem de máquina que são a base de nosso trabalho. Em seguida, apresentamos os

algoritmos que utilizamos para classificação em sistema. No próximo capítulo vamos detalhar como montamos o banco de dados, nosso modelo entidade relacionamento, a proposta de nosso trabalho e quais características escolhemos para fazermos a recomendação.

Capítulo 3

Proposta

Neste capítulo vamos descrever nossa proposta de sistema, figura 3.1, iniciamos relatando como montamos nosso banco de dados, apontando quem nos forneceu o dataset, o formato dos arquivos recebidos, o processo de importação e o modelo entidade relacionamento. Então, apresentaremos detalhadamente nossa proposta de um sistema de recomendação descrevendo as características escolhidas, o processo de classificação, o funcionamento do sistema, e por fim apresentamos possíveis casos de uso.

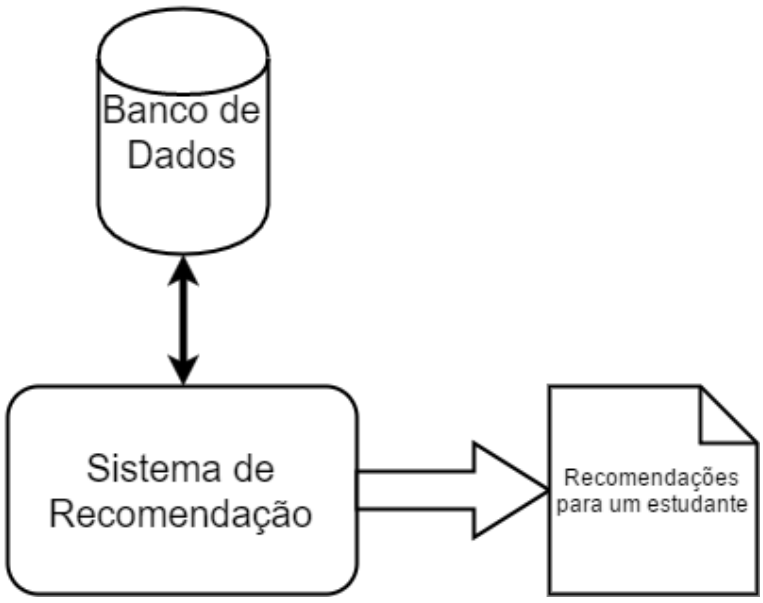

Figura 3.1: Visão geral do sistema.

3.1 Banco de dados

Esta seção é responsável por discutir como o banco de dados foi construído, mostrando o processo de obtenção dos dados, o formato dos arquivos que foram recebidos, o processo de importação e, por fim, nosso modelo entidade relacionamento.

3.1.1 Obtenção

Obter os dados para nossa pesquisa é importante para testar o funcionamento de nosso sistema baseado em fatos reais, mesmo que os dados pudessem ser criados de forma aleatória, eles em sua grande parte não refletiriam o comportamento único e temporal que o grupo discente possui.

Para conseguirmos os dados foi requisitado ao grupo de pesquisa PET Computação, com a autorização do Coordenador do Curso de Ciência da Computação Professor Doutor André L. Vignatti, a versão que o grupo possuía e que lhes foi fornecida pelo SIE[1].

Feito isso nos foi enviado um conjunto de arquivos .csv que representavam os dados dos alunos do Departamento de Informática dos últimos 15 anos.

3.1.2 Formato dos arquivos

O formato CSV (Comma Separated Values), definido no RFC4180, é usado para a troca de dados entre programas que utilizam planilhas, e.g., Microsoft Excel, como descrito na documentação do RFC4180 o formato CSV não é definido formalmente, então pode haver muitas interpretações de como esse formato é estruturado.

Os arquivos com os conjuntos de dados que foram enviados para nós continham um cabeçalho com o nome de cada campo, e as linhas subsequentes era os dados, conforme exemplificado na figura 3.2.

```
Nome, Sobrenome, Cidade
João, Silva, Curitiba
Maria, Silvana, São Paulo
```

Figura 3.2: A primeira linha contem o nome de cada campo, i.e., Nome, Sobrenome e Cidade, e as linhas a seguir são os dados propriamente ditos, i.e., o primeiro dado é Nome: João, Sobrenome: Silva e Cidade: Curitiba.

3.1.3 Importação

Com os arquivos *.csv* em mãos optamos por realizar uma importação dos dados para um SGBD qualquer, visto que, dessa forma o manuseio e as operações seriam mais fáceis de serem feitos, o que por sua conta aumentaria a produtividade dos desenvolvedores e também possibilitaria que se pudesse focar exclusivamente no problema principal que é o sistema de recomendação.

[1]Sistema de Informações para o Ensino (SIE) é o sistema utilizado pela Universidade Federal do Paraná, desde o ano de 2003, para tratar todas as atividades relacionadas à tramitação de processos, controle de almoxarifados, controle de acesso ao sistema e gestão acadêmica [SIE, 2016].

Para realizar a importação primeiramente selecionamos os dados que consideramos importantes de dentro do CSV e então os mapeamos para uma nomenclatura própria, a tabela 3.1 explica esse processo.

Nome no CSV	Nome Escolhido	Descrição do dado
matr_aluno	GRR	GRR do aluno
ano	year	Ano em que realizou a matéria
media_final	final_grade	Média final
período	period	Período do ano que realizou a matéria
frequência	frequency	Frequência do aluno
cod_ativ_curric	name	Nome da matéria

Tabela 3.1: A primeira coluna mostra o nome que existia no cabeçalho do CSV para um determinado campo, a segunda coluna mostra o mapeamento feito e a terceira é uma descrição do que aquele dado representa.

Com esses dados é possível separar em um banco de dados relacional a entidade Aluno e Disciplina porque os atributos GRR e name são únicos para cada entidade, respectivamente. Além disso é possível criar o relacionamento entre essas entidades, chamado de Registro, contendo o ano, período, média final e frequência. Entretanto, com o dataset que nos foi fornecido não é possível dizer qual foi o professor que ministrou aquele registro.

Alguns problemas relacionados aos dados surgiram, por exemplo, matérias que o aluno realizou trancamento, cancelamento, obteve equivalência, obteve dispensa ou até mesmo as matérias em que estava matriculado no momento de exportação dos dados do SGBD do SIE, possuíam como nota final o valor 9999.0, o que não é uma nota válida. Optamos por não realizar a importação desse tipo dado.

3.1.4 Modelo Entidade-Relacionamento

Optamos por utilizar os nomes em inglês por ser um padrão de codificação mais comum. As entidades Student e Course representam os estudantes e as matérias, respectivamente, e ambas possuem relação de 1 para N com a entidade Registry, que representa a matrícula do estudante em uma disciplina, como mostra a figura 3.3

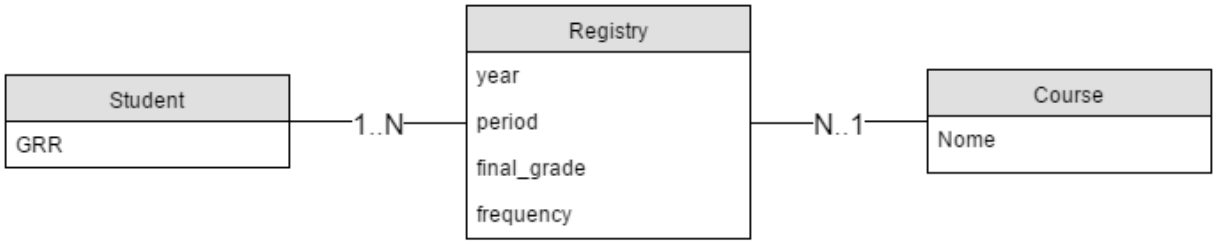

Figura 3.3: Modelo Entidade-Relacionamento utilizado.

3.2 Sistema de recomendação

Nosso sistema é baseado nas notas finais dos alunos, ou seja, as recomendações são feitas baseadas em uma previsão de qual seriam essas notas em disciplinas ainda não vencidas. A seguir vamos apresentar o sistema criado. Iniciamos descrevendo quais são as características e o processo de criação dos vetores de características, em seguida explicamos a classificação, indicando os tipos que criamos e o processo para classificar uma instância. Por último, descrevemos o algoritmo de recomendação.

3.2.1 Características

A escolha do conjunto de características é fundamental para obter melhores resultados ao utilizar os classificadores. Nesta etapa, as melhores características são aquelas que possuem uma relação com as classes possíveis, pois serão discriminantes quando combinadas em um vetor de características.

Nesta subseção, será apresentada cada característica que foi extraída da base de dados e como é o processo de criação do vetor de características.

Descrição das características

As características foram escolhidas de forma a representar a relação entre o estado do aluno naquele momento e sua nota final em alguma disciplina. Observe que registros completos são capazes de fornecer todos os dados necessários, porque informam o ano, período, nota final e frequência do aluno quando cursou a disciplina. A seguir, apresentaremos quais foram as características escolhidas e explicaremos como cada uma é calculada.

- Média geral do estudante: valor médio das notas finais do aluno, contabilizando as disciplinas que já cursou.
- Média geral da disciplina: valor médio das notas finais dos alunos para a disciplina.
- Média de frequência: valor médio de frequência do aluno em todas as disciplinas que já cursou.
- Número de reprovações: número total de reprovações do aluno.
- Média do estudante no último ano: valor médio das notas finais do aluno, contabilizando apenas as disciplinas cursadas no último ano.
- Média de frequência no último ano: valor médio da frequência do aluno, contabilizando apenas as disciplinas cursadas no último ano.

Processo de criação do vetor de características

O processo de criação do vetor de características se baseia em registros completos[2] do banco de dados. Neles temos as informações de ano e período em que um aluno cursou uma disciplina, qual foi sua nota final e sua frequência. A partir destas informações, podemos calcular todas as características descritas acima.

Note que as características são temporais, i.e., todas elas dependem do ano e período a que se referem. Assim, por exemplo, todos os registros de um aluno para um mesmo semestre e ano compartilham a média geral para seus respectivos vetores de características. O mesmo vale para as médias do último ano e o número de reprovações, pois o último ano foi o mesmo para tais registros.

A ideia é analisar e representar a situação do aluno no momento em que ele cursou cada disciplina e criar uma relação entre as informações extraídas da base (características) e o seu desempenho final (nota final).

Um detalhe para a implementação é que no primeiro semestre de um aluno, ou na primeira aparição de uma disciplina no banco, os cálculos relativos a acontecimentos passados, como a média do aluno e a média da disciplina, não existirão. Neste caso, serão gerados vários vetores de características contendo valores zerados pertencendo a diversas classes. Optamos por descartá-los, pois das seis características válidas, cinco são diretamente ligadas ao desempenho do aluno, que ainda não existe nestes casos.

3.2.2 Classificação

A classificação é uma parte importante do sistema visto que ela interfere diretamente na taxa de acerto do algoritmo de recomendação, como será demonstrado empiricamente no capítulo 4. Optamos por criar vários métodos de classificação para observar qual seria o comportamento obtido. A seguir, vamos explicar quais são esses métodos e depois vamos expor como é o processo de classificação de um registro.

Tipos diferentes de classificação

Nesta seção vamos apresentar os métodos de classificação que foram criados. Todos eles funcionam da mesma maneira: recebem como parâmetro uma nota entre 0 e 100 e retornam a classe a qual aquela nota pertence.

Ponto a ponto A classificação ponto a ponto é a mais simples de todas. Ela retorna o piso da nota final, e.g., para o valor 75.2 ela retorna 72. Então suas classes são os valores de 0 até 100.

[2]Chamamos de registro completo o registro que possui todos os campos preenchidos, i.e., nota final, frequência, ano, período e possui os atributos de relação com algum aluno e uma disciplina.

Cinco em cinco A classificação de cinco em cinco, dado uma nota final x, retorna a qual conjunto x pertence. Por exemplo, a nota 42 pertence ao conjunto 40-45, a nota 89 pertence ao conjunto 85-90. Suas classes são 0-5, 5-10, 10-15, ..., 90-95, 95-100.

Dez em dez Este método retorna a qual dezena a nota final pertence, por exemplo, 55 pertence a dezena 50. Suas classes são as dezenas de 0 até 100.

Reprovado ou não Esta classificação também é simples, dado a nota final ela retorna se o aluno foi reprovado ou não, ou seja, se a nota for estritamente menor que 50, o aluno foi reprovado, caso o contrário, ele foi aprovado.

Estilo americano É uma conversão do estilo de notas brasileiro para o estilo de notas americano, como as relações da tabela 3.2.

Nota americana	Nota brasileira
A+	100,00 à 91,7
A-	91,7 à 83,4
B+	83,4 à 75,1
B-	75,1 à 66,8
C+	66,8 à 58,5
C-	58,5 à 50,2
D+	50,2 à 41,9
D-	41,9 à 33,6
E+	33,6 à 25,3
E-	25,3 à 17,0
F+	17,0 à 08,7
F-	08,7 à 00,0

Tabela 3.2: Conversão entre o estilo de notas americano e brasileiro.

Processo de classificação

O processo de classificação pode ser dividido em duas partes, a fase de treino e a fase de predição.

Durante a fase de treino, no momento em que um registro completo é selecionado, obtemos sua nota final para decidir a qual classe ele pertence, ou seja, passamos a nota final como parâmetro para o método de classificação, que é um dos métodos apresentados acima, e obtemos como resposta a sua classe.

Já durante a fase de predição, a responsabilidade de classificar um registro incompleto é do classificador, baseado no conjunto de treino que lhe foi dado, o algoritmo retorna a classe mais provável para aquele registro.

3.2.3 Sistema

Vamos descrever o comportamento do algoritmo em partes, seguindo uma sequência lógica, explicando sobre o que se trata cada parte e dando as definições necessárias.

O algoritmo 1 descreve o processo de criação do conjunto de treino. Seja *Registries* o conjunto de registros, Ψ o conjunto de treino, que se inicia vazio, e $F(\rho)$ uma função em que dado o registro completo ρ, retorna o vetor de características e a classe que ρ pertence, o cálculo de $F(\rho)$ é explicado no processo de classificação na subseção 3.2.1.

Algoritmo 1 Criação do conjunto de treino

1: $\Psi \leftarrow []$
2: **for all** $\rho \in$ `Registries` **do**
3: $\quad \Psi \leftarrow \Psi + F(\rho)$
4: **end for**

Com o conjunto de treino criado, é necessário treinar o algoritmo de aprendizado de máquina para que seja possível realizar predições. Seja Ω esse algoritmo, e Ω_Ψ o resultado do treinamento.

Algoritmo 2 Treinamento do classificador

1: $\Omega_\Psi \leftarrow$ `treina`(Ω, Ψ)

Na próxima parte o algoritmo 3 descreve o processo de recomendações para um aluno específico. Caso seja necessário realizar recomendações para todos os aluno basta iterar sobre o conjunto de estudantes e executar o algoritmo.

Seja *Students* e *Courses* os conjuntos de alunos e disciplinas, respectivamente, tal que $\alpha \in$ *Students*, ξ o conjunto de registros incompletos[3] que α possui, $P(x)$ a função que dado um registro incompleto x, retorna seu conjunto de características e $\Omega_\Psi(y)$ a função que dado um conjunto de características y, retorna a classe que y pertence baseado no conjunto de treinamento Ψ e no algoritmo Ω.

[3]Registro incompleto é um registro cuja nota final e frequência ainda não estão preenchidos. Ou seja, são as disciplinas que o aluno ainda não conseguiu aprovação.

Algoritmo 3 Processo de recomendação para um aluno

```
1: recomendações ← []
2: for all σ ∈ ξ do
3:    recomendações ← recomendações + Ω_Ψ(P(σ))
4: end for
5: recomendações ← orderna(recomendações)
6: return recomendações
```

Na linha 1 iniciamos um vetor de recomendações vazio. Nas linhas 2 a 4 iteramos por todos os registros incompletos que o aluno α possui, adicionando o resultado da predição de cada registro ao final do vetor de recomendações. Na linha 5 ordenamos esse vetor de recomendações de forma que as melhores recomendações sejam as primeiras, mesmo que essa ordenação não seja tão intuitiva como ordenar números é possível visualizar que os tipos de classe que um registro pode ter possuem uma ordenação. Por exemplo, se utilizarmos o sistema americano é fácil notar que A+ é uma classe "melhor" que A-. Por fim, na linha 6 retornamos as recomendações.

3.3 Possíveis casos de uso

Apresentado o funcionamento técnico do sistema, serão abordados possíveis casos de uso seguindo o ponto de vista de quem irá utilizar o sistema. Podemos apontar duas formas de utilização do sistema para o momento da solicitação de matrícula. O portal do aluno pode incorporar essas funcionalidades e disponibilizá-las aos alunos.

A primeira forma é recomendar matérias para um aluno fazer durante o próximo semestre. Neste caso, primeiramente o estudante especifica quantas disciplinas ele pretende cursar, seja esse número N, então o sistema gera as predições para todas as matérias não vencidas pelo estudante, as ordena da melhor predição para a pior, e apresenta as N primeiras como recomendações.

A segunda forma para se utilizar o sistema é avaliar uma solicitação de matrícula de um aluno. O aluno escolhe as matérias que quer cursar no próximo semestre e o sistema realiza uma avaliação das escolhas para prever o desempenho do aluno. Desta forma, o estudante pode ser avisado de uma possível escolha ruim de disciplinas.

3.4 Conclusão

Neste capítulo apresentamos a proposta de nosso trabalho. Primeiro mostramos como foi o processo de obtenção dos dados, depois discutimos o formato dos arquivos que nos foram enviados, o método de importação com seus problemas e soluções, e por fim demos uma descrição de como ficou organizado o banco de dados. Logo depois descrevemos a ideia principal do trabalho que é o sistema de recomendação, expondo as características, a

classificação, demonstrando como funciona o sistema e depois expondo possíveis casos de uso. No capítulo seguinte apresentaremos nossas justificativas para o porque escolhemos determinadas características e algoritmos, e depois vamos exibir os resultados que conseguimos.

Capítulo 4

Experimentação e Validação

Neste capítulo vamos apresentar as justificativas para as características e algoritmos que selecionamos, então vamos mostrar os resultados obtidos.

4.1 Justificativa das características

A seguir vamos apresentar gráficos que mostram a relação entre uma característica e o desempenho final do aluno. A construção dos gráficos foi feita da seguinte forma: para cada característica criamos alguns conjuntos, por exemplo, para a média geral do estudante temos os conjuntos 0-10, 10-20, 20-30, ..., 90-100. Alocamos os registros em cada conjunto de acordo com a média geral do estudante ou a característica desejada. Por fim fizemos a média das notas finais em cada conjunto e assim a relação característica por nota final.

4.1.1 Relação entre nota final e média geral do estudante

É fácil visualizar no gráfico 4.1 que quanto maior a média geral das notas em todas as disciplinas que o estudante já cursou mais provável que sua média final em uma disciplina seja maior também.

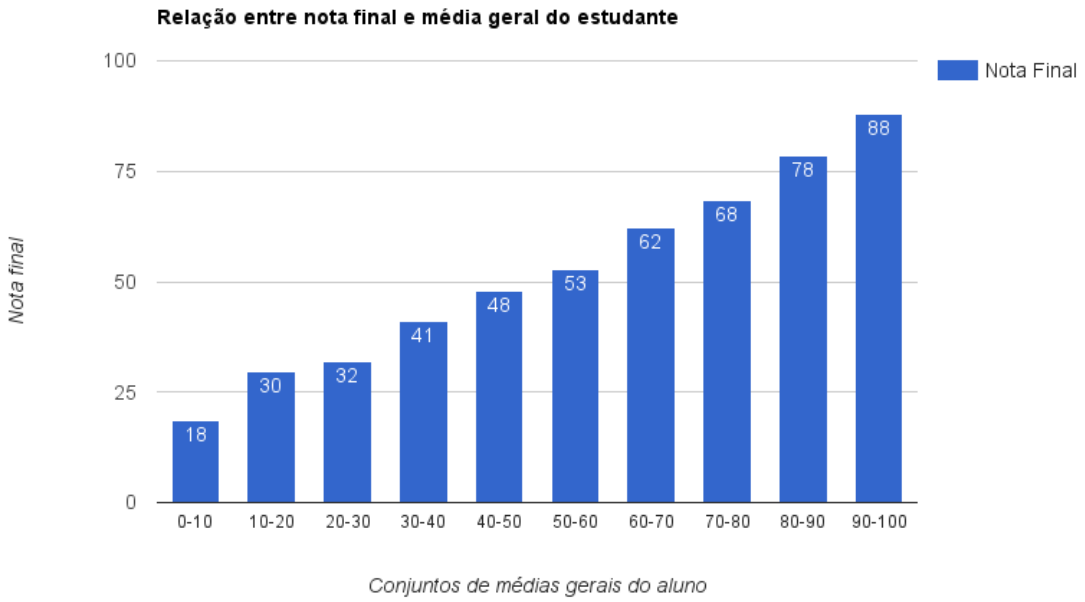

Figura 4.1: Gráfico que demonstra a relação entre a média do estudante e a nota final.

4.1.2 Relação entre nota final e média geral da disciplina

Apesar de na primeira parte do gráfico 4.2 haver algum ruído, também não é difícil de notar que a relação entre média geral da disciplina e nota final é direta.

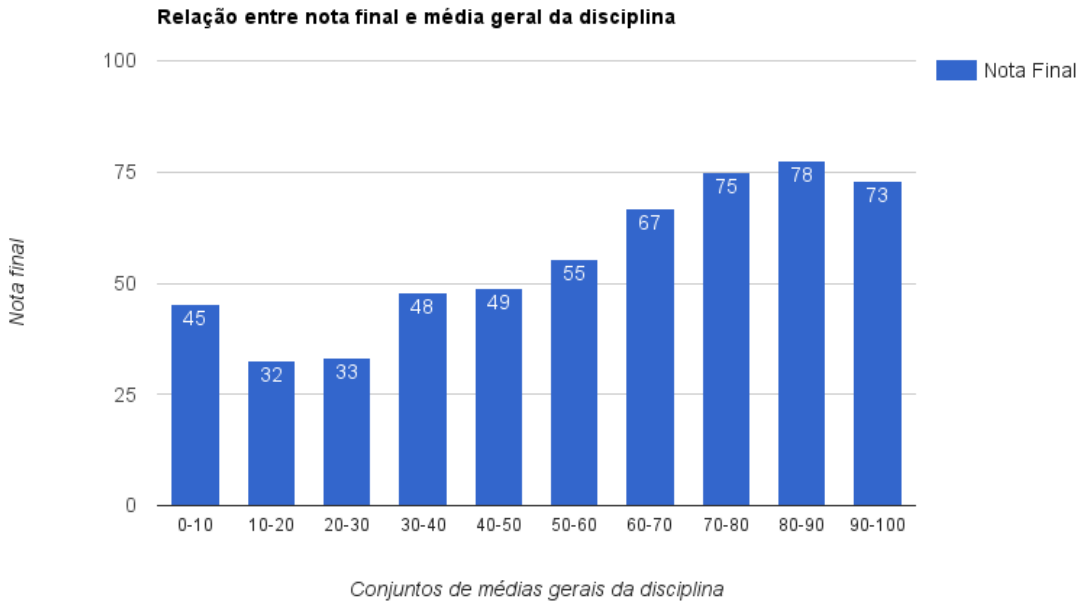

Figura 4.2: Gráfico que demonstra a relação entre a média geral da disciplina e a nota final.

4.1.3 Relação entre nota final e número de reprovações

O gráfico 4.3 demonstra que a nota final tende a cair quanto maior o número de reprovações.

Figura 4.3: Gráfico que demonstra a relação entre o número de reprovações e a nota final.

4.1.4 Relação entre nota final e a frequência média do estudante

O gráfico 4.4 demonstra que a nota final tende a aumentar quanto maior a frequência média do estudante. Nesse gráfico optamos por excluir as classes 0-10, 10-20, 20-30 por possuírem poucos dados.

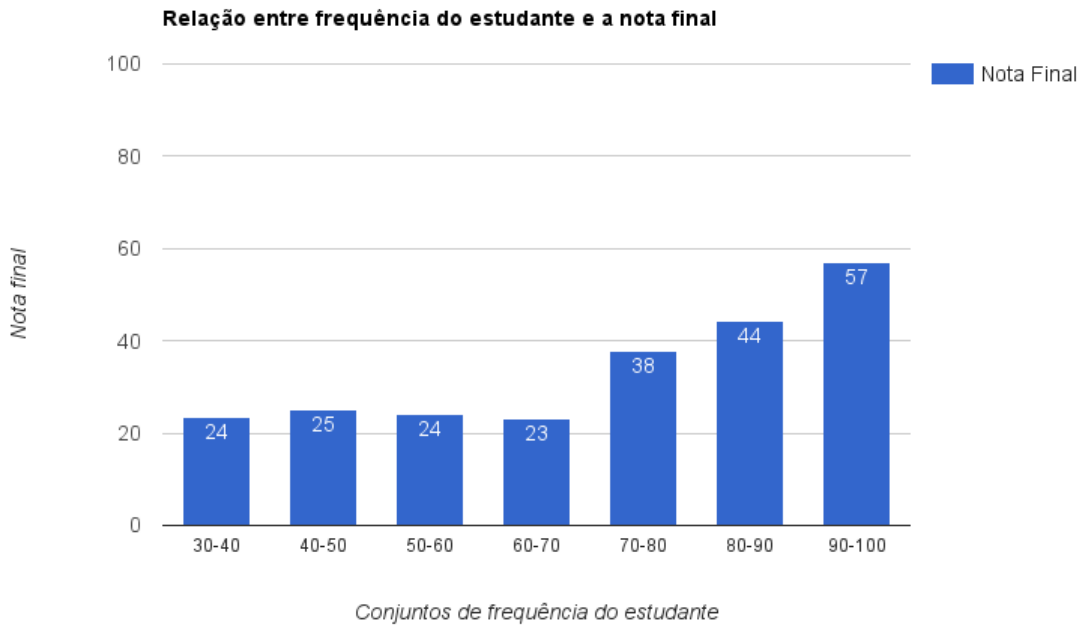

Figura 4.4: Gráfico que demonstra a relação entre a frequência média do estudante e a nota final.

4.1.5 Relação entre nota final e a média geral do último ano

O gráfico 4.5 mostra uma relação direta entre a nota média do último ano e a nota final.

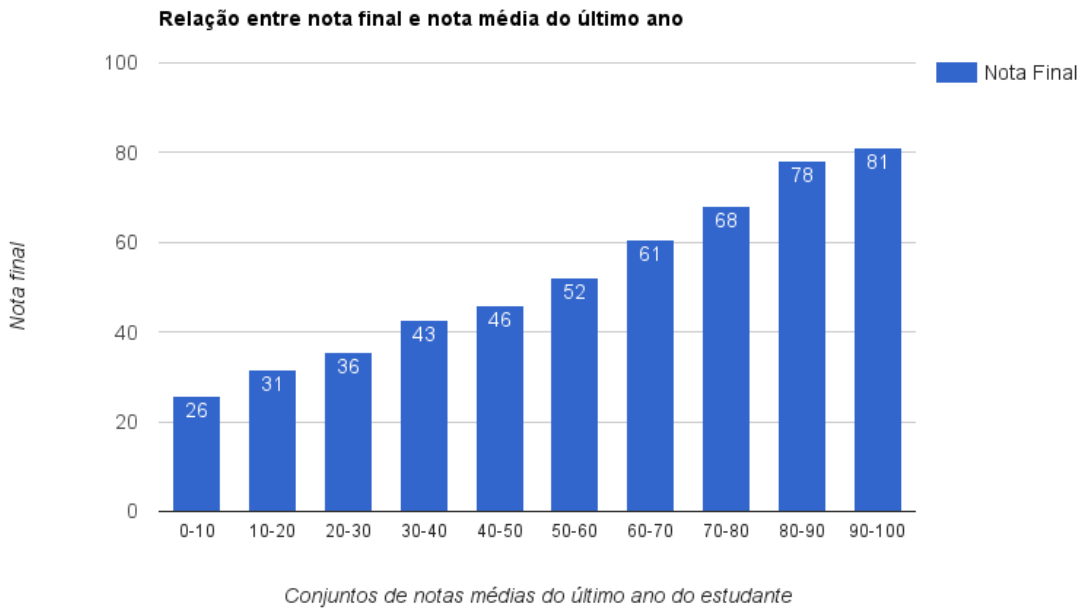

Figura 4.5: Gráfico que demonstra a relação entre a nota final e a média geral do último ano de cada estudante.

4.1.6 Relação entre nota final e a frequência média do último ano

É fácil visualizar no gráfico 4.6 a relação direta entre a frequência média do último ano e a nota final. Novamente optamos por retirar as classes 0-10 e 10-20 por possuírem poucos dados.

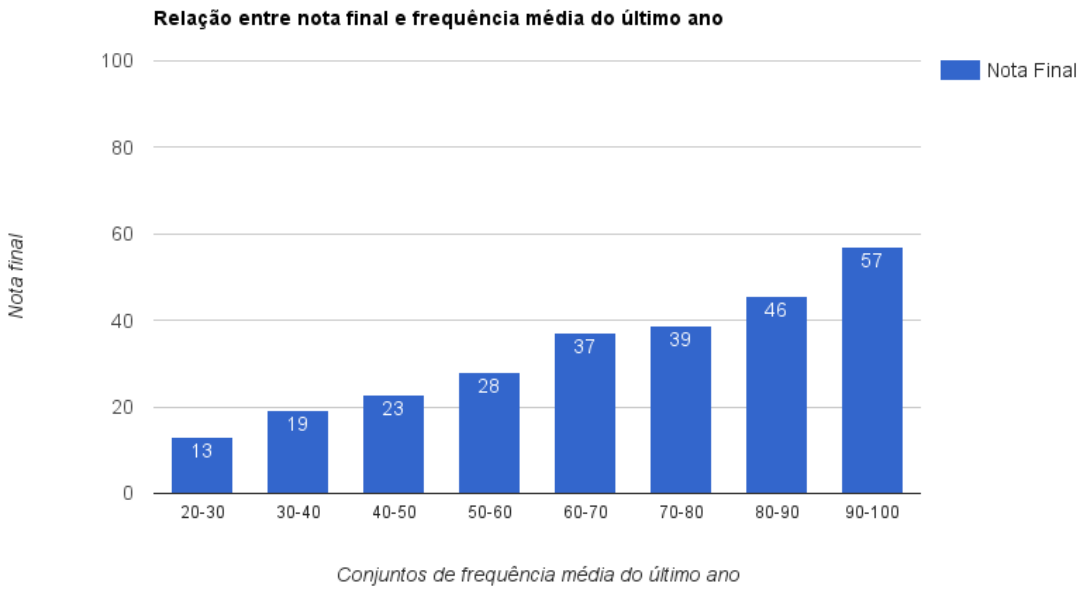

Figura 4.6: Gráfico que demonstra a relação entre a nota final e a frequência geral do último ano de cada estudante.

4.2 Justificativa dos algoritmos

Seguindo o mapa de algoritmos de aprendizado de máquina, figura 4.7, fornecido pela biblioteca open-source Scikit-learn [Pedregosa et al., 2011], chegamos a conclusão que deveríamos usar os algoritmos KNN e SVM como classificadores em nosso trabalho.

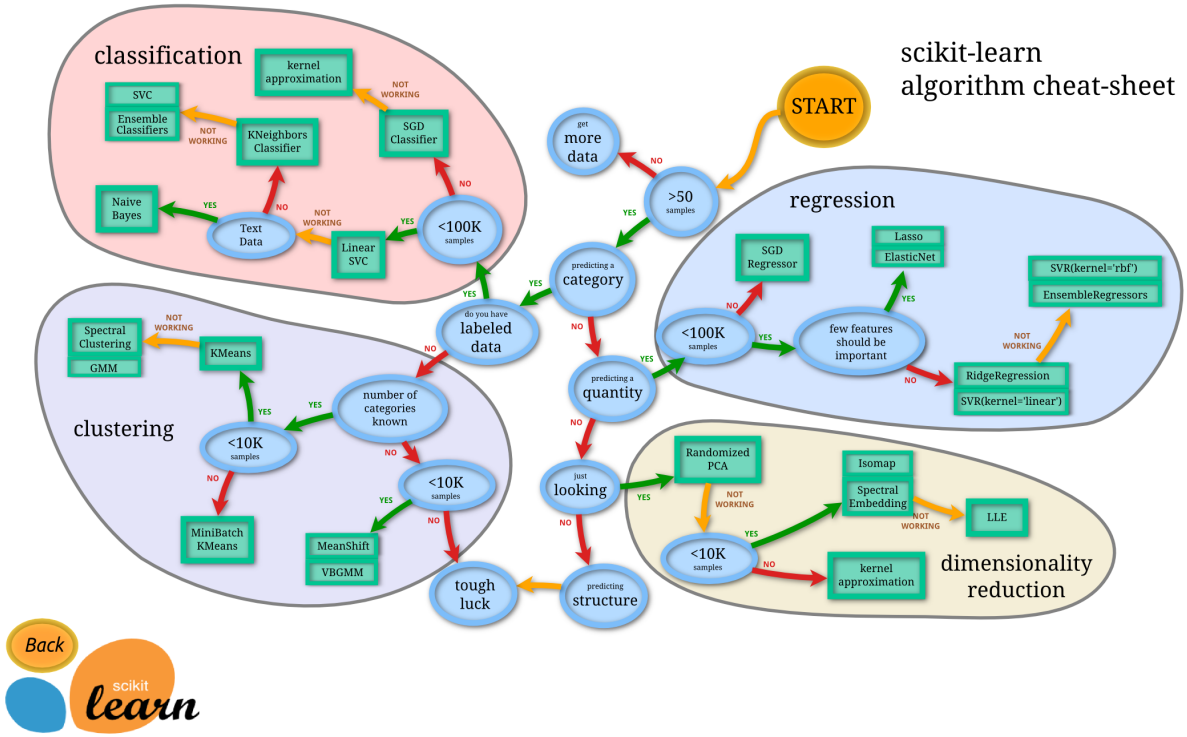

Figura 4.7: Mapa de algoritmos de aprendizado de máquina.

Também optamos por utilizar outro mapa de algoritmos, figura 4.8, fornecido pela empresa Microsoft. Percorrendo este mapa concluímos que o algoritmo floresta aleatória deveria ser utilizado.

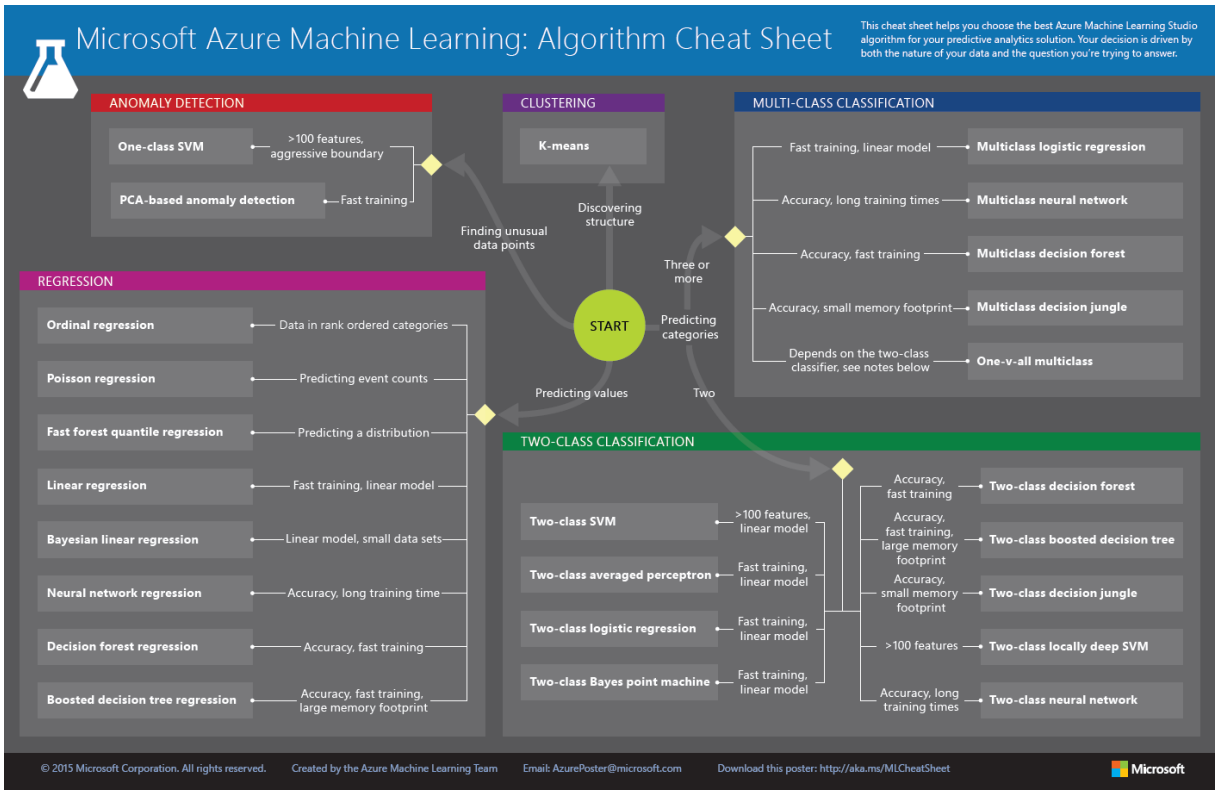

Figura 4.8: Mapa de algoritmos de aprendizado de máquina.

Nossa decisão de escolha foi baseada no fato de que ambas as referências, Scikit-learn e Microsoft, são internacionalmente respeitadas, então optamos por seguir o conselho de quem possui maior experiência na área do que nós.

4.3 Construção das bases de treino e de teste

Para construir as bases optamos por simular como seria o comportamento do sistema ao longo dos anos, criando várias bases de treino e de teste, i.e., iniciamos selecionando todos os registros anteriores ao ano 2009-1 e os colocando na base de treino, então selecionamos os registros com ano 2009-1 para a base de teste, assim conseguimos reproduzir o comportamento que o sistema teria para o ano 2009-1. Fizemos este processo incrementando semestre em semestre até chegarmos ao ano 2015-2 que é o último ano que possuímos os dados.

Com as várias bases de testes criadas é possível observar se a quantidade de registros que informamos ao classificador, como treino, interfere na taxa de acerto.

4.4 Resultados obtidos

Nesta seção apresentamos os resultados que foram obtidos, iniciamos mostrando as diferentes taxas de acerto de cada classificador de acordo com os diferentes tipos de classificação (Ponto a ponto, cinco em cinco, . . .), em seguida mostramos a variação na taxa de acerto do

algoritmo caso uma margem de erro seja introduzida, e por fim usando regressão ao invés de classificação.

4.4.1 Mudando as classes

Vamos apresentar a seguir os resultados que demonstram a variação da taxa de acerto nos algoritmos utilizando um tipo diferente de classificação. A métrica que foi utilizada é um cálculo simples da taxa de acerto, como exemplificado na equação a seguir:

$$\text{taxa de acerto} = \frac{\text{número de predições corretas}}{\text{total de predições}} * 100 \tag{4.1}$$

Como já sabemos qual foi a nota final do aluno para aquele registro, para saber se uma predição é correta ou não basta compará-la com essa nota final.

Resultados com a classificação Ponto a Ponto

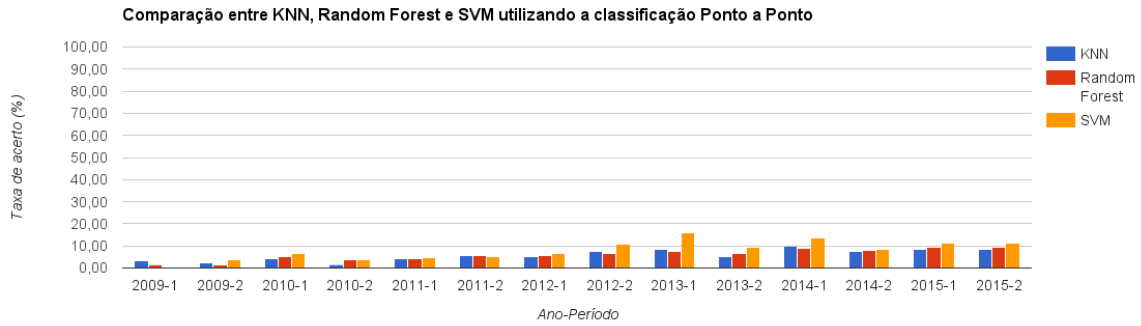

Figura 4.9: Resultados obtidos com a classificação Ponto a Ponto.

Resultados com a classificação Cinco a Cinco

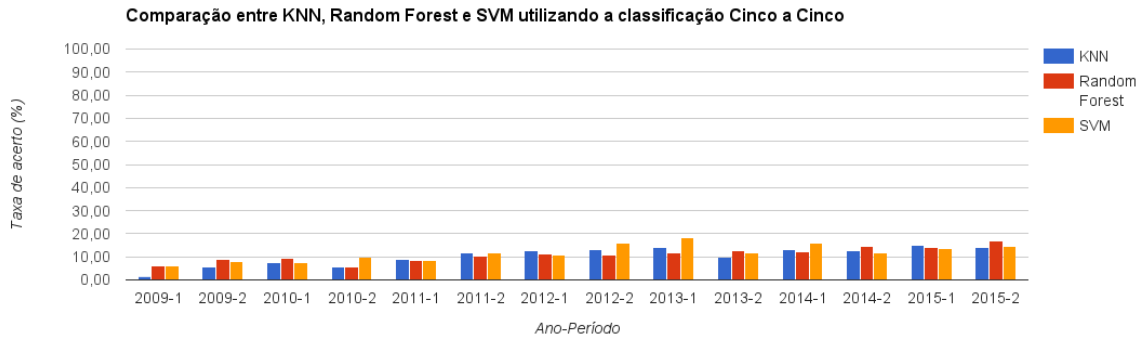

Figura 4.10: Resultados obtidos com a classificação Cinco a Cinco.

Resultados com a classificação Dez a Dez

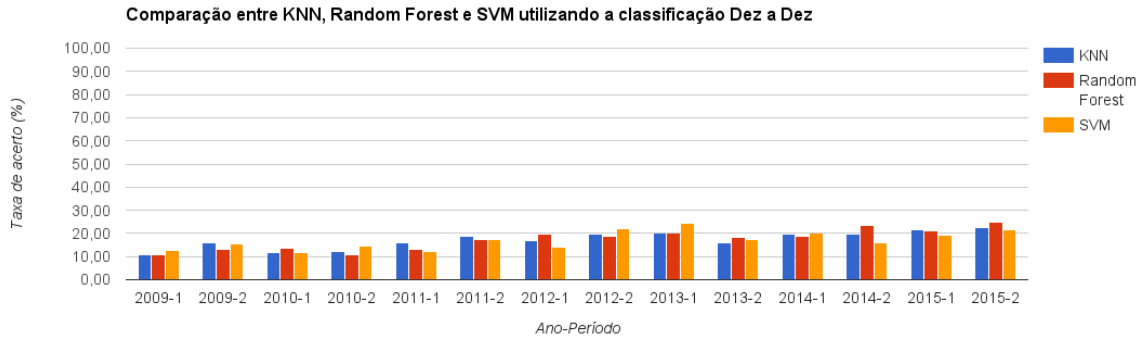

Figura 4.11: Resultados obtidos com a classificação Dez a Dez.

Resultados com a classificação Reprovado ou Não

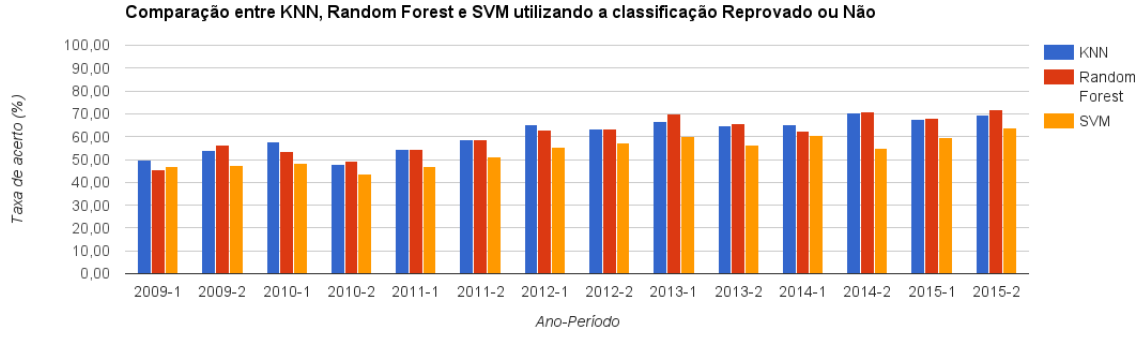

Figura 4.12: Resultados obtidos com a classificação Reprovado ou Não.

Resultados com a classificação Americana

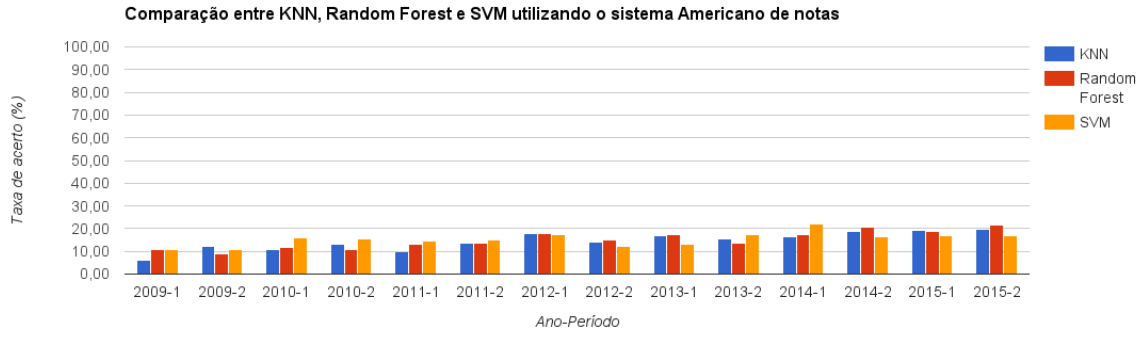

Figura 4.13: Resultados obtidos com a classificação Americana.

4.4.2 Utilizando uma margem de erro na classificação Ponto a Ponto

Nesta subseção vamos mostrar como a introdução de uma margem de erro influencia a taxa de acerto, equação 4.1, dos algoritmos, utilizando a classificação Ponto a Ponto.

Margem de erro de cinco pontos

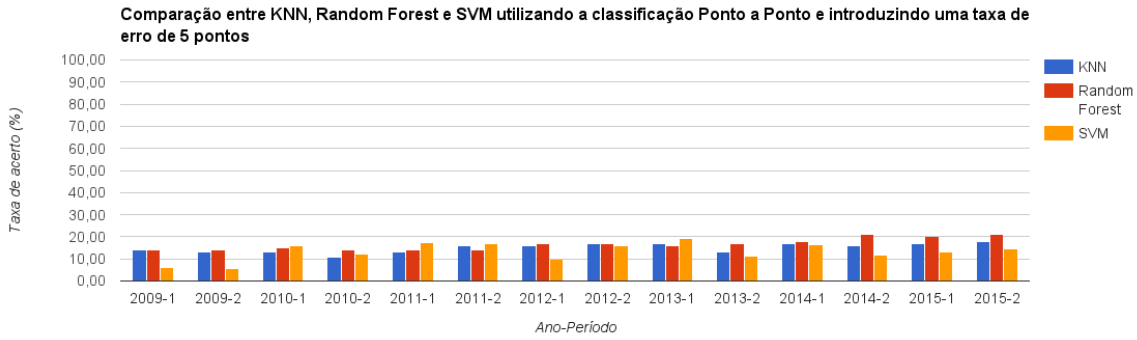

Figura 4.14: Resultados obtidos utilizando uma margem de erro de cinco pontos.

Margem de erro de dez pontos

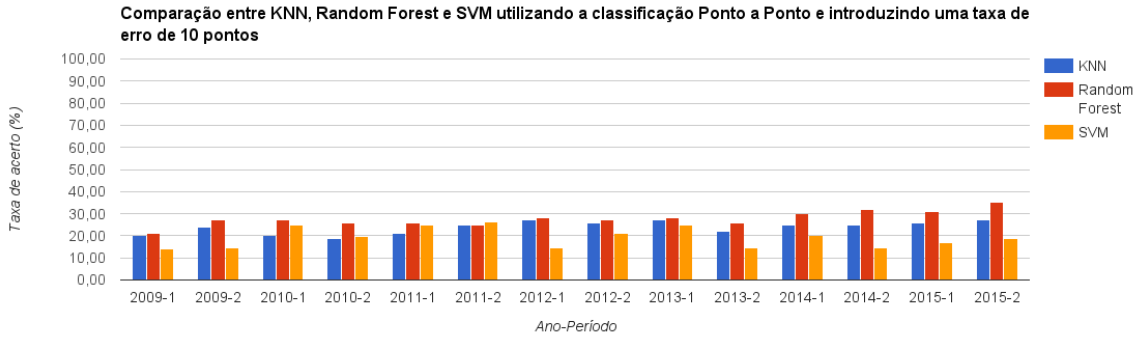

Figura 4.15: Resultados obtidos utilizando uma margem de erro de dez pontos.

Margem de erro de 15 pontos

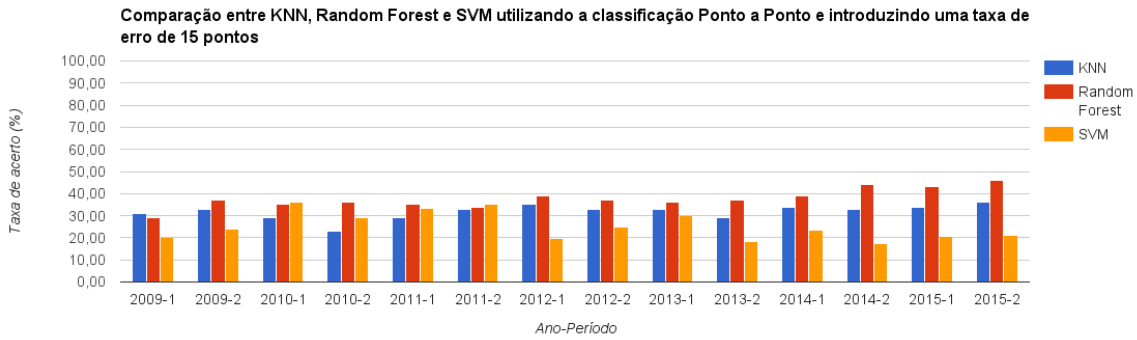

Figura 4.16: Resultados obtidos utilizando uma margem de erro de 15 pontos.

4.4.3 Regressão

Por último testamos os algoritmos utilizando regressão. Utilizamos como métrica o RMSE (Root Mean Square Error), que mede a diferença entre os valores reais e as predições feitas pelos algoritmos.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\texttt{predição}_i - \texttt{real}_i)^2} \tag{4.2}$$

Note que quanto mais próximo o valor da predição é do valor real, menor será o RMSE. E quanto mais distintos eles forem maior será o RMSE.

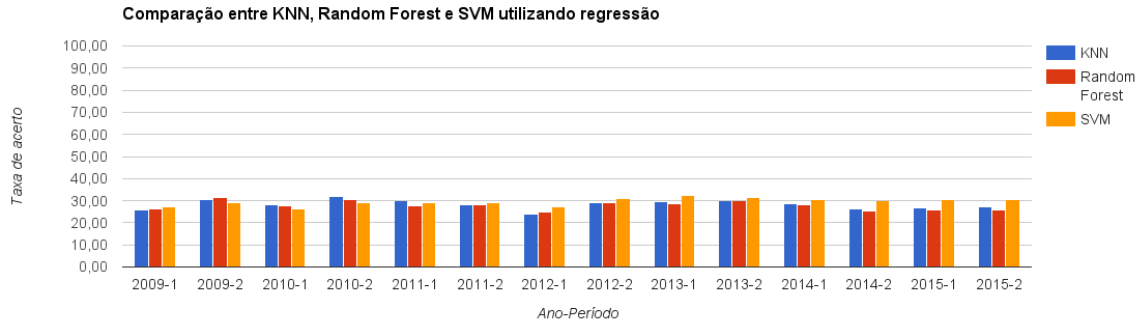

Figura 4.17: Resultados obtidos utilizando regressão.

4.5 Conclusão

Mesmo que os resultados obtidos na classificação não tenham sido satisfatórios, é possível notar que com o passar do tempo existe uma pequena evolução na taxa de acerto, e também que os algoritmos se comportam melhores do que um simples chute. O melhor resultado que obtivemos foi durante o teste com a classificação “Reprovado ou não”, onde conseguimos obter uma taxa de acerto de 70% nos últimos anos.

Com relação aos algoritmos, primeiramente podemos notar que o SVM tem menos poder de acerto quando se tem menos classes, comparado aos outros dois algoritmos. Também vemos que o KNN e o Random Forest tem um comportamento muito parecido em todas as classes.

Quanto aos testes utilizando com uma margem de erro, como esperado com o aumento desse margem melhores foram os resultados. Também é possível constatar que o Random Forest foi o algoritmo que melhor se saiu durante estes testes, por exemplo, com uma margem de 15 pontos ele conseguiu uma vantagem de aproximadamente 10% com relação ao segundo melhor que foi o KNN.

No que diz respeito à regressão, não houve uma grande evolução com o passar do tempo e todos os algoritmos se comportaram da mesma forma.

Neste capítulo iniciamos apresentando nossas justificativas para a seleção das características, demonstrando de forma empírica, que elas influenciam na nota final, então explicamos o por que selecionamos os algoritmos de aprendizado de máquina. Na parte de resultados, mostramos como as bases de treino e teste foram criadas, qual métrica foi usada e então os resultados que obtivemos. No próximo capítulo vamos concluir o projeto e apresentar possíveis trabalhos futuros.

Capítulo 5

Conclusão e Trabalhos Futuros

Observando os resultados obtidos, é possível visualizar que os algoritmos KNN e o floresta aleatória geraram resultados melhores que o SVM. Vale notar que não houve grande variação no resultado final quando os parâmetros dos algoritmos foram modificados, o que nos leva a concluir que apesar dos algoritmos possuírem uma grande interferência no resultado final, as características são quem mais influencia a eficiência do sistema.

Entretanto, com as características que existem agora, há pouquíssima variação com relação à escolha da disciplina em cada semestre, como demonstra a tabela 5.1. Com a base de dados utilizada não foi possível extrair outras características que aumentariam a discriminação entre as classes, por exemplo, acreditamos que com a nota média geral e a nota média do último ano de cada professor, obteríamos uma taxa de acerto melhor. Além disso, outros tipos de característica poderiam ter sido exploradas, como selecionar os alunos que são mais parecidos e utilizar a nota média deles.

Média Geral	Média geral da disciplina	Nº de reprovações	Frequência média do estudante	Média Geral do último ano	Frequência média do último ano
56	72	2	86	75	95
56	55	2	86	75	95
56	66	2	86	75	95
56	82	2	86	75	95

Tabela 5.1: Cada linha da tabela representa um vetor de características gerado a partir de registros de um mesmo semestre pertencentes a um aluno. É fácil observar que a única característica que varia em relação as outras é a média geral da disciplina.

Olhando por outro ponto de vista, ao analisarmos os gráficos, foi possível notar que a quantidade de possíveis classes influencia a eficiência do sistema, visto que quanto mais classes, menor foi a taxa da acerto, enquanto que com duas classes obtivemos a maior taxa. Talvez um sistema que consiga analisar se o aluno irá reprovar ou não na disciplina esteja mais próximo da realidade do que um sistema que consiga prever exatamente sua nota.

Outra análise sobre os gráficos nos mostra que com o passar do tempo as predições ficaram melhores, chegando a ter uma diferença de 20% do primeiro ano-semestre em comparação com o último, no gráfico 4.12. Isso ocorre porque com o passar do tempo, o sistema tem mais dados a seu favor para realizar o treinamento.

Apesar de não termos obtidos resultados satisfatórios, nós, autores, nos consideramos satisfeitos por dar o primeiro passo para a criação um sistema de acompanhamento acadêmico que possa ajudar os alunos a terem um rendimento melhor, e até talvez uma experiência melhor dentro do âmbito acadêmico.

Deixamos para trabalhos futuros uma busca por características que descrevam melhor o comportamento de um aluno, também uma busca por outras maneiras de se fazer a recomendação, visto que não necessariamente a disciplina que o aluno irá obter a melhor nota é a mais recomendada para se fazer. Também é possível observar os detalhes de cada disciplina, ou da grade curricular como um todo, para realizar as recomendações, e por fim a criação do sistema de acompanhamento de que falamos no decorrer do projeto.

Referências Bibliográficas

[Breiman, 2001] Breiman, L. (2001). Random forests. *Mach. Learn.*, 45(1):5–32.

[Felfernig et al., 2013] Felfernig, A., Jeran, M., Ninaus, G., Reinfrank, F. e Reiterer, S. (2013). *Toward the Next Generation of Recommender Systems: Applications and Research Challenges*, páginas 81–98. Springer International Publishing, Heidelberg.

[Mitchell, 1997] Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition.

[Mohri et al., 2012] Mohri, M., Rostamizadeh, A. e Talwalkar, A. (2012). *Foundations of Machine Learning*. The MIT Press.

[Pedregosa et al., 2011] 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. e Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

[SIE, 2016] SIE (2016). Suporte ao sie. `http://cce.ufpr.br/portal/suporte-sie/`. Acessado em 05/11/2016.

[Wen, 2008] Wen, Z. (2008). Filtering.