UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

ANDRÉ RODRIGUES OLIVERA

Comparação de algoritmos de aprendizagem de máquina para construção de modelos preditivos de diabetes não diagnosticado

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Valter Roesler
Co-orientador: Prof. Dr. Cirano Iochpe

Porto Alegre
2016

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Olivera, André Rodrigues

Comparação de algoritmos de aprendizagem de máquina para construção de modelos preditivos de diabetes não diagnosticado [manuscrito] / André Rodrigues Olivera. – 2016.

15 f.:il.

Orientador: Valter Roesler; Co-orientador: Cirano Iochpe.

Dissertação (Mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação. Porto Alegre, BR – RS, 2014.

1.Aprendizagem de Máquina. 2.Modelagem Preditiva. 3.Mineração de Dados. I. Roesler, Valter. II. Iochpe, Cirano. III. Título.

RESUMO

O objetivo deste trabalho foi desenvolver e comparar modelos preditivos para detecção de diabetes não diagnosticado utilizando diferentes algoritmos de aprendizagem de máquina. Os dados utilizados foram do Estudo Longitudinal de Saúde do Adulto (ELSA-Brasil), um conjunto bastante completo com aproximadamente 15 mil participantes. As variáveis preditoras foram selecionadas de forma que sejam informações simples dos participantes, sem necessidade de exames laboratoriais. Os testes foram realizados em quatro etapas: ajuste dos parâmetros através de validação cruzada, seleção automática de variáveis, validação cruzada para estimativa de erros e teste de generalização em um conjunto independente dos dados. Os resultados demonstram a viabilidade de utilizar informações simples para detectar casos diabetes não diagnosticado na população. Além disso, os resultados comparam algoritmos de aprendizagem de máquina e mostram a possibilidade de utilizar outros algoritmos, alternativamente à Regressão Logística, para a construção de modelos preditivos.

Palavras-chave: Aprendizagem de Máquina. Modelagem Preditiva. Mineração de Dados.

Comparison of machine learning algorithms to build predictive models of undiagnosed diabetes

ABSTRACT

The aim of this work was to develop and to compare predictive models to detect undiagnosed diabetes using different machine learning algorithms and data from the Longitudinal Study of Adult Health (ELSA-Brasil), which collected an extensive dataset from around 15 thousand participants. The predictor variables were selected from literature research. The tests were performed in four steps: parameter tuning with cross validation, automatic feature selection, cross validation to error evaluation and generalization test in an independent dataset. The results show the feasibility of extracting useful information from ELSA-Brasil as well as the potential to use other algorithms, in addition to logistic regression, to build predictive models from ELSA-Brasil dataset.

Keywords: Machine Learning. Predictive Modeling. Data Mining.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AB	Acurácia Balanceada
ANN	Redes Neurais Artificiais (*Artificial Neural Network*)
AUC	Área sobre a curva ROC (*Area Under Curve*)
ELSA-Brasil	Estudo Longitudinal de Saúde do Adulto
DCNT	Doenças Crônicas não Transmissíveis
DM	Diabetes Mellitus
DP	Desvio Padrão
FN	Falso Negativo (*False Negative)*
FP	Falso Positivo (*False Positive)*
IMC	Índice de Massa Corporal
K-NN	K Vizinhos mais próximos (*K-Nearest Neighbours*)
NA	Dado Faltante (*Not Available*)
OMS	Organização Mundial da Saúde
TN	Verdadeiro Negativo (*True Negative)*
TP	Verdadeiro Positivo (*True Positive*)

SUMÁRIO

1 INTRODUÇÃO

Diabetes do tipo 2 é uma doença crônica caracterizada pela incapacidade do organismo de metabolizar eficientemente glicose, aumentando o seu nível no sangue e levando à hiperglicemia. Tal condição está associada com uma ampla gama de complicações graves de saúde que afetam o sistema renal, neurológico, cardíaco e vascular, causando grande impacto na saúde global e nos custos com saúde (Glauber e Karnieli 2013).

Estudos recentes estimam que cerca de 415 milhões de pessoas tenham diabetes, podendo chegar em 642 milhões de casos em 2040. Além disso, aproximadamente metade desses casos não está consciente de sua condição, podendo intensificar ainda mais as consequências negativas da doença. Em 2015, o diabetes mellitus foi a causa da morte de quase 5 milhões de pessoas, e é estimado que em 2030 se torne a sétima principal causa de morte no mundo (Beagley et al 2014, Guariguata et al 2014, IDF 2015).

Assim como outras doenças crônicas não transmissíveis, acredita-se que o diabetes é causado principalmente por fatores comportamentais, como uma dieta inadequada e a inatividade física. Intervenções precoces com modificações no estilo de vida ou terapias farmacológicas têm se mostrado eficientes para retardar ou prevenir o diabetes do tipo 2 e suas complicações, fazendo com que muitos países invistam em programas nacionais de prevenção a essa doença. Contudo, para redução de custos, intervenções em nível individual normalmente são direcionadas a sujeitos com maior risco de desenvolver diabetes (Buijsse et al 2011).

Para auxiliar profissionais de saúde a selecionar os indivíduos de mais alto risco para essa intervenção preventiva, podem-se utilizar modelos preditivos criados a partir de dados provenientes de diferentes fontes, tais como dados demográficos, clínicos e testes laboratoriais, entre outros (Murphy et al 2011).

Diversos algoritmos de aprendizagem de máquina podem ser utilizados para a criação de modelos preditivos, tais como regressão logística, árvores de decisão, redes neurais, etc. Vários fatores podem influenciar na escolha do algoritmo mais apropriado para uma determinada aplicação, como, por exemplo, a complexidade computacional (ou espacial), capacidade de manipular dados quantitativos ou categóricos, compreensão do modelo gerado, etc. A capacidade preditiva do modelo gerado é um dos principais fatores e é fortemente relacionada com o conjunto de dados utilizados para criar o modelo. Assim, um algoritmo que gere modelos preditivos de maior acurácia utilizando um determinado conjunto de dados pode não ter o mesmo desempenho quando o conjunto de dados for diferente.

Dessa forma, além de conhecer e entender o funcionamento e outras características inerentes aos algoritmos para poder escolher aquele mais adequado para a aplicação de interesse é importante avaliar a capacidade preditiva dos modelos gerados pelos diferentes algoritmos no conjunto de dados que será utilizado. Além do algoritmo utilizado para criar o modelo preditivo, outros aspectos são relevantes no processo de construção de modelos preditivos, como a parametrização dos algoritmos, categorização, seleção ou transformações de variáveis, entre outros.

O Estudo Longitudinal de Saúde do Adulto (ELSA-Brasil) foi criado com a finalidade principal de investigar múltiplos fatores relacionados à saúde da população, principalmente diabetes e doenças cardiovasculares. O estudo acompanha 15.105 participantes, com idade entre 35 e 74 anos, funcionários de seis instituições públicas de ensino e pesquisa de diferentes regiões do Brasil, arrolados entre 2008 e 2010. Na linha de base, foram obtidos dados através de entrevistas, exames clínicos e laboratoriais, coletando as mais diversas características dos participantes, desde informações sobre demografia e estilo de vida até medidas antropométricas e exames de saúde mais sofisticados. (Aquino et al 2012, Schmidt et al 2012).

A presente Dissertação de Mestrado apresenta o desenvolvimento e comparação de modelos preditivos criados a partir de diferentes técnicas de aprendizagem de máquina para detectar diabetes tipo 2 não diagnosticado em indivíduos na linha de base, utilizando dados não laboratoriais do estudo ELSA-Brasil. A principal pergunta que esta dissertação busca responder é: “É possível criar modelos preditivos simples e fáceis de aplicar para predizer prevalência de diabetes não diagnosticado a partir de dados do ELSA-Brasil? Caso positivo, qual o melhor algoritmo de aprendizagem de máquina a ser utilizado e qual a probabilidade de acerto de cada um deles?”.

Para responder essa pergunta, foram selecionados 34 fatores de risco de diabetes, descritos na literatura, não envolvendo exames laboratoriais ou variáveis com alto custo de obtenção para gerar modelos simples. Tais fatores foram validados com especialistas em epidemiologia. O conjunto dos dados foi aleatoriamente dividido em duas partes numa fração de 70:30, sendo que a primeira parte foi utilizada para afinação dos parâmetros e pontos de corte de cada algoritmo, seleção de variáveis e estimativa de erro através de validação cruzada, e a segunda para teste de generalização. Os modelos criados foram avaliados em termos de AUC (*Area Under Curve*), sensibilidade, especificidade e acurácia balanceada (média entre sensibilidade e especificidade). Foram testadas as seguintes famílias de algoritmos de aprendizagem de máquina: Redes Neurais (MLP/*Backpropagation*), Redes

Bayesianas (*Naïve Bayes*), Árvores de Decisão (Quinlan's C5), Regras de Decisão (*Ripper* e Quinlan's C5), *Ensemble* (*Random Forest*) e *Instance-Based Learning* (K-NN).

Os resultados alcançados indicam que os dados do estudo ELSA-Brasil possuem informações ricas para a criação de modelos preditivos de diabetes de qualidade e, para essa tarefa, os algoritmos de Redes Neurais ou Regressão Logística são os mais apropriados dentre os testados. Com isso foi selecionado um modelo, gerado por Redes Neurais, e, com ele, foi construída uma ferramenta WEB capaz de detectar o risco de diabetes em indivíduos através de perguntas simples.

O resultado dessa dissertação, se aplicado na Atenção Primária à Saúde ou em redes hospitalares, pode trazer benefícios significativos a muitas pessoas que sofrem de diabetes e não sabem disso ainda.

O restante da dissertação é dividido da seguinte forma. O Capítulo 2 descreve as principais técnicas envolvidas em todo processo de criação e avaliação dos modelos preditivos. O Capítulo 3 apresenta alguns trabalhos relacionados à criação de modelos preditivos para detectar diabetes. O Capítulo 4 mostra a metodologia deste trabalho, descrevendo com detalhes as etapas, testes e técnicas utilizadas para criação dos modelos e comparação dos algoritmos. No capítulo 5 são descritas os detalhes de implementação e ferramentas utilizadas para a realização do trabalho. O capítulo 6 mostra os resultados alcançados em cada etapa do trabalho. O capítulo 7 conclui este trabalho apresentando reflexões sobre os resultados, limitações do trabalho e possíveis trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

O atual Capítulo apresenta os principais conceitos relacionados à modelagem preditiva, técnicas para aumentar a qualidade dos dados, técnicas de aprendizagem de máquina e de avaliação de modelos. Os principais conceitos e técnicas sobre aprendizagem de máquina e mineração de dados, apresentados neste Capítulo, estão descritos nos livros (Alpaydin 2010) e (Witten et al 2011), que são referências bibliográficas relevantes no ensino dessa disciplina. Os artigos (Bellazi et al 2006, Brown 2008, Harrison 2008, Koh et al 2005, Lavrac 1999, Obenshain 2004, Yoo et al 2012) são artigos introdutórios ao tema de modelagem preditiva na área da saúde. Tais artigos, além de apresentarem resumidamente os principais conceitos e técnicas utilizadas para essa tarefa, eles apresentam os principais desafios e dificuldades de analisar dados nesse domínio.

O presente Capítulo está dividido da seguinte forma. A Seção 2.1 apresenta os principais conceitos e desafios sobre criação de modelos preditivos utilizando dados médicos. A Seção 2.2 apresenta técnicas utilizadas na preparação/pré-processamento dos dados com o objetivo de aumentar a qualidade destes e produzir modelos de maior capacidade preditiva. A Seção 2.3 descreve os algoritmos de aprendizagem de máquina mais populares utilizados para criação de modelos preditivos. Por fim, a Seção 2.4, apresenta os métodos e métricas utilizadas para avaliar a qualidade de modelos preditivos.

2.1 Modelagem Preditiva de Dados Médicos

Modelos preditivos podem ser vistos como funções matemáticas que mapeiam um conjunto de valores de entrada (variáveis preditoras) para um valor de saída (variável-alvo). Um dos objetivos da modelagem preditiva em medicina é criar modelos que utilizam informações de indivíduos para prever um desfecho de interesse ou risco de um desfecho e, dessa forma, auxiliar na tomada de decisão clínica (Bellazi 2006).

Os modelos são gerados a partir de um conjunto de dados de treinamento que contém observações compostas por um conjunto de variáveis preditivas (fatores de risco) e o evento ou desfecho que se deseja prever (variável-alvo). Esse processo de treinamento é chamado de aprendizagem supervisionada. Quando a saída do modelo é contínua, por exemplo, previsão de custos de um hospital, a tarefa é de regressão ou aproximação de funções. Por outro lado, quando a saída é categórica, por exemplo, prever se um determinado indivíduo irá contrair uma doença, a tarefa é de classificação. Assim, na classificação, o objetivo do modelo é atribuir uma classe a um novo caso apresentado ao modelo, ou ainda, estimar a probabilidade

desse caso pertencer a uma determinada classe. A detecção de diabetes, descrita nessa dissertação de mestrado é realizada através de algoritmos de classificação.

Uma das características mais importantes de um modelo preditivo é o seu poder preditivo, ou seja, a capacidade de prever corretamente novos casos (independentes dos casos utilizados para treinar o modelo) que são apresentados ao modelo. Tal característica depende do algoritmo utilizado para gerar o modelo e, principalmente, dos dados utilizados para treinamento do modelo, ou seja, se os dados conseguem explicar claramente o fenômeno que está sendo observado. De modo geral, não existe um algoritmo específico que construa o melhor modelo preditivo sempre. Dessa forma, para um determinado conjunto de dados, um algoritmo (com certa configuração de parâmetros) pode gerar o melhor modelo enquanto em um conjunto diferente o melhor modelo pode ser gerado por um algoritmo diferente.

Outros fatores que podem influenciar no poder de predição do modelo são intrínsecos aos dados. Conjuntos de dados podem conter instâncias ruidosas ou valores anômalos dados faltantes, variáveis supérfluas ou redundantes, tipos de dados heterogêneos, entre outras características que influenciam na qualidade do modelo gerado. Assim, é importante conhecer tais características antes de construir o modelo. Uma das etapas do processo de mineração de dados consiste em realizar a limpeza e preparação dos dados para o algoritmo que irá ser utilizado de forma que o algoritmo consiga extrair o máximo de informações contidas nos dados.

Também é importante levar em consideração características relevantes à aplicação que se quer construir. Por exemplo, um aspecto especialmente importante na área da medicina é a compreensão do modelo gerado. Ou seja, é importante saber como o modelo chegou a uma determinada resposta. Outra característica desejável em aplicações na área da saúde é ter como resposta uma estimativa de probabilidade de um determinado caso pertencer a uma classe em oposição à classificação tradicional de um caso em uma classe.

Outras características importantes, relacionadas ao algoritmo, incluem: a complexidade temporal e espacial do treinamento e da inferência, a capacidade do algoritmo de manipular dados heterogêneos e dados faltantes, a sensibilidade/robustez com relação à valores anômalos, a eficácia do algoritmo para evitar sobreajuste aos dados de treinamento, a dificuldade de configuração do algoritmo, se o algoritmo consegue aprender relações não lineares ou se o algoritmo permite adicionar conhecimento especialista ao modelo gerado.

2.2 Preparação dos Dados

A construção de modelos preditivos vai além da aplicação de um algoritmo de aprendizagem de máquina sobre um conjunto de dados. Em aplicações reais, muitas vezes existem dados mal preenchidos, mal distribuídos, instâncias com dados faltantes ou instâncias que não representam o comportamento que se deseja modelar (exceções), entre outros problemas. É preciso conhecer bem os dados disponíveis para decidir um tratamento adequado para tais situações. A seguir são discutidas técnicas comuns utilizadas para resolver os principais problemas enfrentados durante a preparação dos dados para a criação de modelos preditivos.

2.2.1 Redução de dimensionalidade e seleção de variáveis

Um número muito grande de variáveis (dimensões) disponíveis para análise pode, algumas vezes, atrapalhar o processo de criação de modelos preditivos. Apesar de alguns algoritmos naturalmente conseguirem trabalhar bem com conjuntos de dados de alta dimensionalidade, normalmente é conveniente reduzir a dimensionalidade dos dados sendo analisados com o objetivo de simplificar o processo de treinamento e/ou de classificação e melhorar os resultados obtidos pelo modelo.

A redução de dimensionalidade pode ser alcançada automaticamente através de técnicas de análise de componentes principais ou técnicas de agrupamento de dados que visam, de modo geral, a construção de novos atributos a partir dos existentes contendo os aspectos principais dos dados que estão sendo analisados. Contudo, os atributos gerados (extraídos) a partir de tais abordagens muitas vezes não possuem um significado relevante ao domínio e, para muitas aplicações médicas onde a interpretabilidade do modelo gerado, tais técnicas podem não ser adequadas.

Outro método de extração de variáveis é, manualmente, através de conhecimento do domínio dos dados, derivar novas variáveis, a partir de uma ou mais variáveis, que possuem maior relevância que as variáveis originais para prever um evento. Tal abordagem, contudo, exige um nível alto de conhecimento especialista.

Além disso, sabe-se que a introdução de variáveis redundantes ou irrelevantes (para prever um determinado desfecho) pode interferir negativamente na qualidade dos modelos gerados a partir desses dados. Técnicas automáticas de seleção de variáveis tentam amenizar esse problema. O objetivo dessas técnicas é selecionar o melhor subconjunto de variáveis para

um determinado problema, ou seja, o subconjunto capaz de produzir o modelo de melhor capacidade preditiva (Liu et al 2005).

Análises estatísticas (e.g. Correlação de *Pearson* e Informação Mútua) podem ser utilizadas para estimar individualmente a relevância de uma variável preditora na previsão de uma variável-alvo, assim é possível enfileirar as variáveis de acordo com sua relevância. Tais informações também podem ser obtidas através da análise de modelos gerados por alguns algoritmos de aprendizagem de máquina, tais como regressão logística e árvores de decisão.

Informações sobre relevância de variáveis, por si só, podem ser úteis para a aplicação independentemente da construção do modelo, principalmente no domínio da medicina, podendo ser utilizada para identificar os fatores de mais alto risco para um evento indesejado, por exemplo. Contudo, muitas dessas conclusões não levam em consideração a análise de múltiplas variáveis na previsão de um desfecho. E, em determinadas situações uma variável pode ser irrelevante individualmente mas relevante quando em conjunto com outras variáveis.

Segundo Liu *et al* (2005), um processo típico de seleção de variáveis consiste de quatro etapas básicas: geração de subconjuntos, avaliação de subconjuntos, critério de parada e avaliação dos resultados. A geração dos subconjuntos é um procedimento que produz os subconjuntos de variáveis candidatas baseado em uma determinada estratégia de busca. Cada subconjunto candidato é avaliado e comparado com o melhor subconjunto até o momento através de um critério de avaliação. Se um novo subconjunto for melhor, ele substitui o melhor já encontrado. O processo é repetido até algum critério de parada ser satisfeito. O resultado deve então ser avaliado com conhecimento a priori ou diferentes testes em conjuntos de dados.

Técnicas automáticas de seleção de variáveis podem ser divididas em três categorias principais: *filter*, *wrapper* e *embedded* (Ghyon, I. and Elisseeff, 2003). Técnicas da categoria "*embedded*" são realizadas pelo próprio algoritmo utilizado para criar o modelo preditivo. Geralmente algoritmos que geram árvores de decisão utilizam esse tipo de técnica.

As técnicas *filter* e *wrapper* se diferenciam em como é feita a avaliação de um subconjunto de variáveis. No primeiro caso, a avaliação é baseada em características gerais dos dados e a técnica funciona como um filtro antes do início do processo de aprendizagem, portanto é independente do algoritmo de aprendizagem utilizado. Nesse caso são necessárias métricas como correlação de *Pearson* ou estimativas das informações mútuas entre variáveis preditoras candidatas e a variável-alvo. No segundo caso, a avaliação é realizada através da capacidade preditiva de modelos gerados a partir do subconjunto de variáveis sendo avaliada. Para tanto, é preciso definir um esquema de aprendizagem (preparação de dados + algoritmo

+ parâmetros do algoritmo) para ser utilizado para avaliar o subconjunto. Nesse caso, as variáveis são avaliadas em conjunto com relação à variável alvo.

Independente do método de avaliação utilizado, a forma como é feita a busca pela melhor solução é muito importante para obter resultados satisfatórios nessa tarefa. Idealmente, poderia se testar todos os subconjuntos de variáveis e, a partir de alguma métrica de qualidade de modelo, escolher qual o subconjunto que constrói o melhor modelo. Porém, tal abordagem é intratável computacionalmente quando o número de variáveis é maior do que algumas dezenas devido ao espaço de busca muito grande. Dessa forma, é preciso adotar heurísticas para buscar no espaço de soluções.

Devido à alta complexidade computacional dessas técnicas e do tamanho do espaço de busca, normalmente se utiliza uma estratégia gulosa para procurar pelo melhor subconjunto de variáveis. As principais heurísticas de busca, baseadas nessa estratégia, utilizadas para seleção de variáveis são *forward selection* e *backward elimination* (Ghyon, I. and Elisseeff, 2003). A primeira heurística inicia a busca com um conjunto vazio e, a cada iteração, uma nova variável é adicionada ao subconjunto, o subconjunto que produz o melhor resultado prossegue para a próxima iteração, os outros são descartados. A seleção termina quando não há melhoria no modelo resultante ao adicionar novas variáveis ou o resultado piora. O segundo método é análogo, contudo ele começa com o conjunto de todas as variáveis e, a cada etapa, retira uma das variáveis do subconjunto. As duas abordagens podem ser combinadas para tentar obter um melhor resultado.

Segundo Guyon e Elisseeff (2003), a técnica de *forward selection* é mais eficiente computacionalmente e tende a produzir subconjuntos menores de variáveis (o que melhora o entendimento do modelo gerado), por outro lado, essa técnica pode encontrar subconjuntos mais fracos (ótimos locais), pois ela pode deixar de considerar certas variáveis em conjunto que poderiam melhorar os resultados.

Além dessas buscas sequenciais, buscas randômicas (*random-start hill-climbing*, *simulated annealing*, algoritmos genéticos e o algoritmo Las Vegas), e buscas completas (*branch-and-bound*, e *beam search*) podem ser utilizados na construção dos subconjuntos candidatos. (Liu *et al* 2005)

2.2.2 Tratamento de Variáveis Quantitativas e Categóricas

Muitos algoritmos trabalham somente com um determinado tipo de variável, ou dados categóricos (algoritmos de árvores de decisão) ou dados quantitativos (algoritmos de redes

neurais). Em alguns casos, a adequação (transformação de um tipo de dado para outro) é realizada pela própria implementação do algoritmo, em outros ela precisa ser realizada manualmente. É importante ressaltar também que informação que a variável carrega é deteriorada quando se aplica tais transformações, logo é fundamental conhecer o processo de forma a minimizar essa perda e obter bons resultados.

O processo de transformação de uma variável com valores quantitativos para valores categóricos, conhecido como categorização, funciona através da seleção de um ou mais pontos de corte de tal forma que os valores entre eles representarão uma determinada categoria. A seleção desses limiares influencia diretamente na qualidade do modelo preditivo gerado. Quando não há definição clínica para categorizar a variável, idealmente, a melhor seleção de pontos de corte poderia ser encontrada através de uma busca exaustiva onde se testariam todas as opções disponíveis e se selecionaria a que cria o melhor modelo preditivo. Contudo, o espaço de busca é muito grande, principalmente quando se precisam analisar múltiplas variáveis com múltiplos pontos de corte. Sendo assim, é preciso adotar heurísticas para selecionar pontos de cortes bons.

Um método simples é a divisão da faixa de valores (valor mínimo até o máximo) da variável em N pedaços de mesmo tamanho (onde N seria especificado pelo usuário), porém tal método não leva em consideração o resultado da classificação. Para tarefas de aprendizagem supervisionada onde existe informação da classe de cada instância no conjunto de treinamento, é possível utilizar essa informação para realizar a categorização. Por exemplo, é possível ordenar as instâncias de acordo com os valores da variável em questão e colocar pontos de corte quando há mudança da classe majoritária observando as classes as quais as instâncias pertencem.

Outras técnicas mais sofisticadas calculam o ganho de informação ao utilizar certo limiar e tentam encontrar o conjunto de limiares que proporcionam o melhor ganho de informação. Ou ainda dividir o ganho de informação pela informação de partição (uma métrica que mede a entropia de um conjunto de instâncias com relação às partições formadas por um conjunto limiares) calculando, assim, a razão do ganho de informação com o objetivo de penalizar variáveis que possuem uma faixa de valores muito extensa.

Alguns algoritmos probabilísticos, como o *Naïve Bayes*, criados originalmente para dados categóricos, podem utilizar funções de densidade probabilísticas em dados quantitativos, baseadas na distribuição da variável (normalmente é suposto uma distribuição normal) para os cálculos das probabilidades a *posteriori* utilizadas na classificação.

Outros métodos, como por exemplo, algoritmos de descoberta de regras associativas, necessitam de variáveis binárias, assim, nesses casos, é necessário realizar o procedimento de dicotomização. Para isso, cada categoria de uma variável categórica é transformada em uma nova variável cujo valor reflete se a categoria é o valor da variável original ou não (por exemplo, se existe ou não algum item em uma cesta de compras).

A transformação de variáveis categóricas em variáveis quantitativas é um processo mais simples e também pode ser obtido através da dicotomização, porém atribuindo valores 1 e 0 (ou -1) indicando se a instância pertence ou não a uma determinada categoria. É preciso levar em consideração que o processo de dicotomização pode aumentar consideravelmente a dimensionalidade do conjunto de dados, o que, como visto anteriormente, pode prejudicar o desempenho de alguns algoritmos, além de poder criar variáveis mal distribuídas.

2.2.3 Tratamento de Valores Faltantes

Uma das maiores dificuldades na análise de dados é lidar com dados incompletos. Essa situação pode ocorrer por diversos motivos, tais como a falta de uma resposta em um questionário (que por sua vez pode ocorrer por diferentes motivos), um erro na captura dos dados ou por outros motivos. Alguns algoritmos de aprendizagem de máquina, como o *Naïve Bayes* e C4.5 podem simplesmente ignorar os valores faltantes nos cálculos necessários, probabilidades condicionais e de ganho de informação, respectivamente. Contudo, a maioria das técnicas de aprendizagem exige que os dados estejam totalmente completos para realizar a análise, impondo que o analista decida o que fazer para lidar com a situação.

O tratamento mais simples para esse problema, e que é geralmente oferecido por ferramentas de análise de dados é não considerar na análise casos que contenham valores faltantes. Claramente tal abordagem é indesejada, principalmente quando a quantidade de dados é limitada, pois tende a eliminar muitos casos quando o conjunto de dados possui muitas dimensões (basta uma das variáveis não possuir valor que o caso inteiro é desconsiderado). Além disso, como a quantidade de dados disponíveis geralmente é pequena, não é interessante excluir casos que possam ter um valor real para o modelo que se deseja construir.

Uma abordagem mais interessante, chamada na área de estatística de *Imputação* (Schafer 1999), é a substituição dos valores faltantes por valores válidos tornando o conjunto de dados apto para análise com o menor viés possível. Para valores quantitativos, uma técnica simples é substituir os valores faltantes pela média ou pela mediana dos valores da variável,

ou ainda, selecionar um valor de forma que a variância da variável não seja alterada. Para valores categóricos, podem-se substituir os valores faltantes pelos valores mais frequentes ou criar novas categorias para tais valores. Técnicas mais complexas baseiam-se nas relações entre as variáveis preditoras para estimar o valor de dados faltantes.

Outra estratégia, complementar às já discutidas, é a eliminação de variáveis com dados faltantes. Além de facilitar a aplicação das técnicas de imputação, essa estratégia ajuda a reduzir a dimensionalidade do conjunto de dados, ajudando no processo de aprendizagem. Contudo, existe o risco de eliminar uma variável que poderia ser importante para a construção do modelo preditivo. Assim, podem ser feitas análises de relevância da variável com ou sem valores imputados bem como utilizar conhecimento de domínio para ajudar na decisão de quais variáveis podem ser eliminadas da análise.

Existem diversos estudos e abordagens para estimar os valores de dados faltantes, tais como técnicas baseadas em simulação como *Multiple Imputation* (Harel and Zhou 2007) ou técnicas baseadas em algoritmos de aprendizagem de máquina (Jerez et al. 2010).

Concluindo, além de conhecer as principais estratégias que podem ser utilizadas para tratar dados faltantes, é importante entender a natureza dos dados faltante para poder escolher quais das estratégias devem ser adotadas no processo de modelagem.

2.2.4 Ruídos, Erros e Variáveis Assimétricas

Outros fatores que podem atrapalhar potencialmente a aprendizagem de modelos preditivos são ruídos nos dados e dados assimétricos. Ruídos nos dados podem ser acidentais, como por exemplo, erros de digitação, ou erro na coleta dos dados ou ainda uma observação considerada uma exceção do fenômeno que se deseja modelar. Em alguns casos é possível identificar dados errôneos ou inconsistentes e corrigi-los manualmente ou até identificar casos indesejáveis e excluí-los do conjunto de treinamento. Alguns algoritmos de aprendizagem são mais sensíveis a esse tipo de problemas que outros. Geralmente, problemas de ruídos nos dados são tratados utilizando métodos para evitar o sobreajuste (*overfitting*) aos dados de treinamento, como por exemplo, poda da árvore de decisão.

A distribuição das variáveis também influencia na qualidade dos modelos criados. Alguns algoritmos assumem que os dados possuam uma distribuição normal, enquanto outros funcionam melhor quando os dados não são assimétricos. Contudo, em bancos de dados reais, normalmente nos deparamos com conjuntos de dados esparsos que podem atrapalhar o processo de aprendizagem. Nesses casos, é possível aplicar transformações numéricas para

tentar deixar os dados num formato mais amigável para o algoritmo de aprendizagem de máquina. Um método bastante utilizado para transformar variáveis com muitos valores baixos e alguns poucos valores altos é a transformação logarítmica. Contudo, antes de realizar a transformação, é necessário entender porque os valores estão assimétricos, às vezes pode existir alguma variável com dados incorretos que pode estar enviesando a análise.

A distribuição das variáveis pode ser analisada através de estatísticas básicas (média, mediana, moda, quartis, desvio padrão) ou diretamente através de gráficos de densidades. Para auxiliar na análise, é possível utilizar a métrica que mede a assimetria da distribuição de probabilidade sobre a média de uma variável.

Para analisar variáveis categóricas, calcula-se a frequência de cada categoria sobre os dados disponibilizados. Variáveis com muitas categorias ou categorias com muito poucas instâncias podem atrapalhar o algoritmo de aprendizagem. Assim, pode ser interessante agrupar múltiplas categorias em uma. Porém tal processo tem que ser analisado caso a caso para saber se esse agrupamento faz sentido e não vai acabar prejudicando os resultados do modelo. Dessa forma, analisar a distribuição das variáveis individualmente ajuda a identificar possíveis problemas com os dados além de permitir preparar melhor os dados através de transformações para um algoritmo de aprendizagem.

2.2.5 Padronização de variáveis

Variáveis com diferentes escalas de valores podem atrapalhar o processo de aprendizagem. Alguns algoritmos, como redes neurais artificiais, praticamente exigem que a entrada dos valores esteja na faixa entre -1 e 1 ou 0 e 1. Assim, é muito importante fazer a normalização ou uma padronização dos valores antes de realizar o aprendizado do modelo.

A normalização "*mini-max*" pode ser realizada quando valor máximo e mínimo que uma variável possa ter é conhecido. Ela é obtida subtraindo o valor original do valor mínimo e dividindo pela diferença entre o valor máximo e mínimo, fornecendo um valor entre 0 e 1. Quando os valores máximo e mínimo possíveis de serem alcançados são desconhecidos, podem-se utilizar valores dos dados de treinamento, porém quando for apresentado ao modelo novos casos com valores fora dessa faixa o resultado da normalização não ficará entre 0 e 1.

Na padronização (escore-z), o valor é subtraído pela média da população e, então, divido pelo desvio padrão. Como normalmente a média e desvio padrão da população são desconhecidos, utilizam-se os valores obtidos da amostra dos dados de treinamento.

2.2.6 Considerações finais sobre Preparação dos Dados

A preparação dos dados é uma etapa fundamental na criação de modelos preditivos. Sempre que possível, deve-se analisar as variáveis manualmente procurando por possíveis problemas ou características que possam atrapalhar o processo de aprendizagem e tomar as medidas possíveis para amenizar os resultados negativos. Para isso é preciso conhecer as principais estratégias para os tratamentos desses problemas e escolher a que melhor se adequa a cada situação.

Também é importante lembrar que qualquer processo de transformação ou padronização aplicado aos dados de treinamento também deve ser aplicado para realizar a predição e, para tanto, devem ser utilizados os mesmos parâmetros utilizados durante a aprendizagem e esses parâmetros devem ser obtidos a partir dos dados utilizados para treinamento do modelo.

2.3 Principais Algoritmos de Classificação

Abaixo segue uma descrição dos principais algoritmos de aprendizagem supervisionada para tarefa de classificação.

2.3.1 Regressão Logística

Regressão logística é uma técnica bem estabelecida da área de estatística e muito utilizada para estudos na área médica, geralmente obtendo bons resultados. Por isso, ele é utilizado como técnica de referência quando comparado com outros algoritmos para análise de dados médicos.

O princípio da regressão logística tem base na técnica de regressão linear onde uma variável quantitativa y é definida como uma relação linear das variáveis preditoras X de acordo com a seguinte fórmula: $h_x = w_0 + w_1x_1 + \cdots + w_kx_k$. O conjunto de pesos W é encontrado através da minimização de uma função de custo baseada nos erros quadráticos dos dados de treinamento. Na regressão logística, utiliza-se uma função sigmoide (logística) que define a equação linear fornecendo respostas entre 0 e 1. A saída dessa função significa a probabilidade condicional de uma instância pertencer a uma determinada classe dado um conjunto de variáveis quantitativas preditoras. A função logística tem a forma $Pr(Y = 1 \mid X) = \frac{1}{1+ e^{-g(X)}}$, onde $g(X) = h_x$, conforme definido na regressão linear. A função de custo, nesse caso tem um formato diferente da função de custo da regressão linear.

O aprendizado do modelo consiste em encontrar, através da máxima verossimilhança, o conjunto de pesos W (parâmetros $w_0, w_1, \dots, w_k$) que melhor se ajusta aos dados de treinamento, ou seja, que minimiza a função de custo. Para a função logística, é mais conveniente tentar encontrar os parâmetros que encontram a máxima verossimilhança logarítmica (*log-likelihood*) da função. Existem muitos métodos de otimização numérica que podem ser utilizados para encontrar a máxima verossimilhança, um dos mais antigos e importantes é o método de *Newton-Raphson*. Contudo tal método, devido à grande complexidade, é inviável para problemas com mais de uma dimensão. Assim, normalmente são utilizadas heurísticas para encontrar a máxima verossimilhança, tais como, o método *IRLS* (*Iterative Least Reweighted Least Squares*) ou métodos *quasi-Newton*, como o *L-BFGS* (*Limited-memory Brouden-Fletcher-Goldfarb-Shanno)*.

Apesar de a definição descrita funcionar somente para classes binárias, a técnica é facilmente estendida para problemas multi-classe através do método de classificação em pares (*pairwise classification*), onde o classificador é construído para cada par de classes, utilizando somente as instâncias dessas duas classes.

O modelo gerado pela técnica é facilmente interpretado, permitindo ao analista inferir quais variáveis possuem mais ou menos relevância para o evento que está sendo analisado. Porém, o modelo não possui mecanismos para lidar com dados faltantes, assim é preciso lidar manualmente com eles. Valores anômalos e variáveis irrelevantes também costumam atrapalhar o desempenho do modelo. Além disso, o modelo exige uma alta taxa de casos de treinamento com relação ao número de variáveis preditoras, sendo necessária uma grande quantidade de dados quando estes possuem grande dimensionalidade.

Por fim, a técnica como foi descrita funciona como um classificador linear, ou seja, não trabalha bem para problemas não linearmente separáveis. Contudo, é possível alterar a função da fórmula ($g(X)$) do algoritmo para torná-lo um classificador não linear. É preciso tomar cuidado com o sobreajuste aos dados de treinamento quando se utiliza tal abordagem, além de se perder a interpretabilidade dos parâmetros do modelo. O formato dessa função bem como a função de custo deve ser definido pelo usuário. Também, pode-se incluir termos de regularização para tentar minimizar o sobreajuste aos dados de treinamento.

A definição de Regressão Logística descrita acima é baseada principalmente no trabalho de Witten (2011).

2.3.2 Redes Neurais Artificiais

A aprendizagem por redes neurais artificiais (Haykin, 2008) é um dos principais métodos de aprendizagem de máquina e muito utilizado, inclusive na área de saúde. Sua principal desvantagem é que o modelo construído funciona como uma caixa preta, ou seja, é muito difícil compreender/interpretar o modelo.

O modelo de rede neural artificial é composto de alguns elementos básicos: o neurônio é uma unidade elementar (nó na rede) que dado um conjunto de conexões de entrada com os respectivos pesos sinápticos, ele calcula uma saída através de uma função de ativação, normalmente uma função sigmoide; as conexões determinam um caminho para o fluxo de informação entre dois nós; e a topologia da rede é um grafo direcionado que determina a conectividade dos neurônios. Uma rede neural possui uma camada de entrada (com 1 neurônio para cada variável de entrada do modelo), uma camada de saída (com 1 neurônio para cada classe) e 0 ou mais camadas ocultas. As camadas ocultas são utilizadas para resolver problemas não lineares através da composição de funções lineares. O número de camadas ocultas e o número de neurônio em cada camada oculta são definidos pelo analista e geralmente são descobertos de forma empírica, através de validação cruzada. A resposta da rede é conhecida através da propagação da saída de cada neurônio seguindo todas as conexões das redes.

O *perceptron* elementar é o exemplo mais básico de rede neural, ele é composto por 1 neurônio de saída que realiza o somatório das entradas ponderadas pelos seus respectivos pesos sinápticos (e nenhuma camada oculta) mais um termo de viés. O treinamento supervisionado do neurônio descobre os pesos sinápticos necessários para resolver o problema. O treinamento visa encontrar o melhor hiperplano que separa as classes sendo analisadas (problemas linearmente separáveis somente) através do algoritmo LMS (*Least Mean Square*). Este algoritmo minimiza uma função de custo que é proporcional ao erro quadrático instantâneo (a inicialização dos pesos sinápticos é aleatória). Quando se utiliza a função logística como função de ativação, essa arquitetura é equivalente à regressão logística.

O esquema mais importante de redes neurais, contudo, utiliza uma arquitetura de *perceptrons* de múltiplas camadas (MLP, *Multilayer Perceptron)* e o algoritmo de retropropagação (*backpropagation)* para aprender os pesos sinápticos dos neurônios baseado na influência de cada neurônio nas respostas dadas pela rede. O algoritmo de retropropagação pode ser visto como uma generalização do algoritmo LMS para múltiplas camadas.

Um dos métodos mais comuns para a minimização da função de custo é a descida de gradiente, mas, assim como regressão logística, é possível utilizar métodos *quasi-Newton* para encontrar os melhores pesos sinápticos para os neurônios. A ideia é a mesma que regressão logística, onde é definida uma função de custo, baseada nos erros quadráticos, e essa função deve ser minimizada. O treinamento é realizado por muitas épocas, onde em cada época todos os exemplos de treinamento são utilizados. O treinamento termina quando se atinge determinado erro ou limitando a quantidade de épocas de treinamento. A convergência do algoritmo depende principalmente da inicialização dos pesos, do tamanho do passo de aprendizagem, do algoritmo para a minimização de custos e dos dados disponíveis para treinamento.

Utilizando pelo menos uma camada oculta, pode-se considerar a rede neural um classificador universal, capaz de resolver problemas não linearmente separáveis. Porém, assim como outros métodos não paramétricos, o aprendizado de redes neurais é suscetível ao sobreajuste do modelo final aos dados de treinamento.

Além do MLP, existem outras arquiteturas de redes neurais artificiais. Redes neurais RBF (*Radial Basis Function*), por exemplo, são equivalentes ao MLP, contudo são baseados em diferentes princípios matemáticos. Nela, cada neurônio da camada oculta possui uma função de base radial (função gaussiana, por exemplo) como função de ativação e a saída da rede é uma combinação linear dessas funções.

Outras arquiteturas de redes neurais que podem ser utilizadas para classificação incluem: KBANN (*Knowledge-based Artificial Neural Network*), ANFIS (*Adaptive Neuro Fuzzy Inference System*), PNN (*Probabilistic Neural Network*), entre outras.

2.3.3 Árvores de Decisão

Árvores de decisão são modelos em estrutura de árvore onde cada nodo interno representa um teste de uma determinada variável que define um caminho para chegar da raiz à folha da árvore (nó terminal). A folha da árvore representa a decisão a ser tomada, ou seja, o resultado da classificação. Uma das principais vantagens da utilização desse tipo de estrutura gráfica é a fácil visualização do modelo gerado. Além disso, uma árvore de decisão pode facilmente ser transformada em um conjunto de regras (o caminho da raiz até cada folha é único e pode ser transformado em regra) que é outra forma conveniente de modelar uma determinada situação.

O procedimento padrão para criar uma árvore de decisão utiliza uma estratégia de "divisão e conquista" onde, a cada passo, uma determinada variável é selecionada (a que separa melhor os dados com relação à variável alvo) e os dados são particionados de acordo com os valores da variável (cada valor define um ramo, ou seja, caminho a ser seguido na árvore). O processo é repetido recursivamente para os ramos inferiores utilizando somente o subconjunto de dados de cada ramo e as variáveis que sobraram. O processo continua até não haver mais como dividir os subconjuntos de dados ou acabarem os dados. O modelo trabalha exclusivamente com dados categóricos, assim, a categorização geralmente faz parte do algoritmo e acontece localmente, ou seja, somente com a partição dos dados que está sendo analisada no momento.

Dessa forma, uma das decisões mais importantes para a criação do modelo é, dado um conjunto de dados e de variáveis, escolher a variável mais relevante para ser o nodo da árvore. Tal escolha normalmente é realizada através do cálculo do ganho de informação, que representa o quanto se ganha em média em pureza ao se dividir um conjunto segundo a variável em questão, isso é equivalente a calcular a entropia da distribuição da classe. A ideia é escolher a variável que produz o maior ganho de informação. Outras métricas para seleção do atributo incluem o índice GINI e o erro de classificação.

Contudo, a árvore criada dessa forma pode ficar muito complexa e não generalizar para instâncias diferentes dos dados de treinamento, ou seja, se sobreajusta aos dados de treinamento. Para evitar isso, é necessário podar a árvore, diminuindo sua complexidade. Duas estratégias podem ser utilizadas para esse fim: pré-poda envolve decidir se expande ou não um determinado ramo durante o processo de construção da árvore; enquanto a pós-poda constrói a árvore primeiro e depois faz a poda. Apesar de a primeira opção ser viável, pois evita o trabalho de construir a árvore, a pós-poda é mais utilizada devido a outras vantagens como considerar atributos em conjunto para avaliar a poda. Existem duas operações principais de poda (pós-poda), a primeira, chamada *subtree replacement*, substitui uma sub-árvore inteira por uma folha e a segunda, chamada *subtree raising*, envolve substituir uma sub-árvore por outra sub-árvore descendente da primeira. A segunda operação, apesar de mais complexa é utilizada no principal algoritmo de construção de árvores de decisão, o C4.5.

Para classificar instâncias com dados faltantes, uma estratégia que pode ser utilizada é percorrer todos os ramos da variável à qual não existe informação e de alguma forma ponderar as respostas encontradas. Já para criar a arvore utilizando dados com valores faltantes, é necessário tomar as medidas apresentadas na Seção 2.2.3.

Um dos principais algoritmos para criação de árvores de decisão e que funciona como descrito anteriormente é o Quinlan C4.5 (bem como seu antecessor ID3 e seu sucessor C5.0) (Quilan 1993). Outro algoritmo importante é o CART (*Classification And Regression Trees*) (Breiman *et* al 1984). Ao contrário do C4.5 que constrói árvores univariadas (que considera somente um atributo por nodo da árvore) o CART tem uma abordagem multivariada em que a árvore consiste em uma hierarquia de modelos lineares permitindo que mais de uma variável seja testada a cada nodo. Porém CART é um algoritmo mais complexo computacionalmente e seu modelo não é tão simples de interpretar. CHAID (*CHi-squared Automatic Interaction Detection*) (Kass 1980), baseado no teste de *Holm-Bonferroni*, é outra opção para criar árvores de decisão.

2.3.4 Regras de decisão

Regras são compostas de duas partes: a parte "antecedente" representa um conjunto de testes (sobre os valores de variáveis preditoras), geralmente agregados com a conjunção "e"; e a parte "Consequente" que é a classe ou distribuição de probabilidades que ocorre quando as condições da parte anterior são satisfeitas. Árvores de classificação podem ser facilmente convertidas em regras de classificação, onde cada caminho da raiz até todos os nodos formam uma regra, contudo as regras geradas por tal abordagem geralmente são muito complexas.

Existem algoritmos específicos para a criação de regras de classificação a partir dos dados de treinamento. A ideia fundamental é gerar um conjunto de regras que cubra todos os exemplos de uma determinada classe e excluir exemplos fora da classe. Tal estratégia é chamada de "abordagem por cobertura", pois a cada passo uma regra que cobre certo número de exemplos é gerada. Tais exemplos são então excluídos e o processo é repetido para os exemplos restantes. Os testes das regras são escolhidos de forma a maximizar a precisão da regra, e cada teste adicionado diminui a cobertura da regra. A precisão é medida dada pela razão entre a quantidade de exemplos positivos cobertos pela regra e o total de exemplos cobertos pela regra (quando houver empate, escolhe o com maior cobertura).

O algoritmo básico descrito acima se chama PRISM (Cendrowska 1987) e emprega uma estratégia de "separar e conquistar" onde inicialmente se identifica uma regra, todos os exemplos cobertos pela regra são então separados e o processo se repete para os exemplos restantes. O principal problema dessa abordagem é que não lida bem com dados ruidosos. O algoritmo CN2 (Clark & Niblett, 1989) tenta lidar com esse problema. A ideia é que ao invés de selecionar somente a melhor regra para o procedimento, são selecionados k candidatos e

todos são analisados, tal método é conhecido como busca em feixe. Outras extensões foram criadas a partir desses algoritmos.

Um dos principais algoritmos para extração de regras de classificação é o *RIPPER* (*Repeated Incremental Pruning to Produce Error Reduction*) (Cohen 1995). Ele foi o primeiro algoritmo de indução de regras a lidar efetivamente com o problema de sobreajuste. O algoritmo utiliza uma métrica de tamanho de descrição (DL, *Descrição Lenght)* para decidir quando interromper a geração de regras. O algoritmo adiciona uma fase de poda para evitar sobreajuste.

2.3.5 Aprendizagem Baseada em Casos

A aprendizagem baseado em casos possui um abordagem diferente das outras e pertence a uma classe de métodos que realizam aprendizagem preguiçosa (*lazy learning*), pois toda a computação necessária é realizada durante a classificação. Em outras palavras, não existe a criação de um modelo propriamente dito, as instâncias rotuladas do conjunto de treinamento são armazenadas em memória, e quando é apresentado um novo caso para classificação, o algoritmo busca, através de uma função de similaridade (ou distância), quais os casos mais parecidos com o caso de estudo e classifica o novo caso baseado nessa informação. Por outro lado, é necessário que os casos estejam disponíveis para consulta durante a classificação, o que pode exigir uma grande quantidade de memória. A função de similaridade pode ser projetada como um conceito de vizinhança onde, quanto mais perto de uma instância, mais alta é a influência dessa instância para realizar a classificação.

O algoritmo mais conhecido de aprendizagem baseada em casos é o dos *k* vizinhos mais próximos (k-NN, *k Nearest Neighborhood*). O funcionamento é bem simples, dado uma instância que se deseja classificar, ele identifica as *k* instâncias mais próximas e, através de uma votação, a instância nova é rotulada com a classe mais frequente. Ainda é possível estimar a densidade da probabilidade dos dados através do volume (da esfera, aproximado a partir da distância entre a instância e cada vizinho) ocupado pelos *k* vizinhos mais próximos.

2.3.6 Classificadores Bayesianos

Redes bayesianas (Langley *et* al 1992, Friedman *et al 1997*) são modelos gráficos probabilísticos representados por grafos direcionados acíclicos (*DAG*) onde cada nodo representa uma variável aleatória (estocástica) e as conexões representam dependências condicionais entre as variáveis.

Uma rede bayesiana é definida através de sua estrutura (estrutura do grafo, nodos e conexões) e das probabilidades *a priori* de cada variável com relações ao nodo antecedente (pai), tais probabilidades são descritas em uma Tabela de probabilidade condicional (*CPT*). Possuindo essas informações, através de inferência bayesiana (utilizando o teorema de *Bayes*), é possível calcular a probabilidade *a posteriori* de uma determinada variável dado um conjunto de observações sem necessidade de conhecer as probabilidades conjuntas. Dessa forma pode-se calcular a probabilidade de uma variável-alvo pertencer a uma determinada classe sabendo-se o valor das outras variáveis (preditoras).

Uma vantagem dessa abordagem é que ela lida bem com dados faltantes. Além disso, é possível adicionar conhecimento de domínio na forma da estrutura do grafo ou das probabilidades *a priori*. Uma desvantagem é que a inferência pode ser muito custosa computacionalmente para redes complexas. Ainda, apesar de a probabilidade *a priori* ser facilmente inferida dos dados, através de contagem de frequência, é muito difícil estimar a melhor estrutura de dependências quando não se tem esse conhecimento disponível.

O classificador "*Naïve Bayes"* é uma rede bayesiana com uma estrutura fixa onde cada variável preditora é independente das outras variáveis dada a variável alvo. Em outras palavras, no grafo representando a estrutura, as únicas conexões que existem saem da variável-alvo em direção a todas as variáveis preditoras. Apesar dessa forte e ingênua (*naïve*) suposição de independência, o classificador costuma funcionar bem geralmente. Além disso, o classificador trabalha bem com dados de alta dimensionalidade, não precisando de muita quantidade de exemplos para a criação do modelo. Naturalmente a adição de muitos atributos redundantes tende a dar problemas. A principal vantagem do classificador é sua simplicidade computacional, pois não é necessário utilizar um mecanismo de busca para encontrar as melhores hipóteses.

Uma extensão do *Naïve Bayes* que visa relaxar as suposições de independência entre as variáveis mantendo a simplicidade computacional é o classificador "*Tree Augmented Naïve Bayes*". Tal estrutura permite que uma variável tenha como ancestral outra variável além da variável-alvo.

2.3.7 Combinações de múltiplos modelos (*Ensembles*)

A ideia por trás de classificadores *ensembles* (Kotsiantis 2006) é de utilizar diversos modelos simples (*weak learners*) para fazer a classificação, ao invés de confiar somente em um modelo, com o objetivo de aumentar a acurácia da previsão.

O algoritmo *Bagging* (*Bootstrap Aggregating*) utiliza a técnica de amostragem *bootstrap* que cria vários subconjuntos de dados de treinamento (do mesmo tamanho do conjunto original) a partir do conjunto de treinamento original através de uma amostragem uniforme (cada exemplo tem a mesma chance de ser escolhido) e com repetição (é possível escolher o mesmo exemplo mais de uma vez). Com tal abordagem, é esperado que cada subconjunto amostrado tenha aproximadamente 63% de exemplos únicos e o resto de duplicatas. Assim, é possível gerar um modelo de classificação diferente (pois utilizam diferentes dados de treinamento) a partir de cada subconjunto de treinamento amostrado. A classificação é feita por todos os modelos e é feita uma combinação para dizer o resultado final. A criação dos modelos pode ser feita utilizando um determinado algoritmo de aprendizagem de máquina, sendo muito comum o uso de árvores de decisão.

O algoritmo *AdaBoost* (*Adaptive Boosting*) se diferencia do *Bagging* pois os modelos não são construídos individualmente. No *AdaBoost* os modelos são construídos iterativamente de modo que a construção de um modelo influencia a construção dos modelos que serão criados nas iterações posteriores. A ideia é utilizar instâncias ponderadas para construção de modelos e a cada iteração ponderar novamente as instâncias de forma que as instâncias que foram classificadas erroneamente tenham um valor mais alto e as que foram corretamente tenham um valor menor; assim, os próximos modelos são construídos favorecendo as instâncias classificadas previamente de forma incorreta, iterativamente. Dessa forma um modelo complementa o outro. A classificação de uma nova instância é realizada de forma ponderada, levando em consideração o erro que cada modelo teve para classificar os dados de treinamento.

Outro algoritmo muito importante que utiliza técnicas de *ensemble* é o *Random Forest*. A ideia é construir uma floresta de *Random Decision Trees* e utilizar todas as árvores para classificar uma nova instância. Cada árvore é gerada utilizando um subconjunto das variáveis disponíveis escolhidas de forma aleatória, além de um diferente conjunto de dados, gerado pela técnica de amostragem *bootstrap*. O algoritmo ainda pode ser utilizado para classificação de relevância de variáveis.

A principal desvantagem da utilização desses métodos é a falta de interpretabilidade do modelo gerado por tal abordagem.

2.3.8 Outros métodos

Além das estratégias apresentadas, existem muitos outros algoritmos ou variações que podem ser utilizados para criação de modelos preditivos. Um algoritmo capaz de construir modelos de grande poder preditivo utilizando dados de alta dimensão é chamado de máquinas de vetor suporte (SVM, *Support Vector Machine*) (Cortes, 1995), cuja ideia principal é buscar em espaços de alta dimensão pelo hiperplano ótimo cuja margem entre dois objetos de classes diferentes seja máxima. Para tanto o SVM utiliza vetores suporte e a margem é determinada utilizando tais vetores. A principal vantagem dessa abordagem é a acurácia do modelo gerado, além da capacidade de trabalhar com dados de muitas dimensões. Contudo, assim como redes neurais, também são modelos de difícil interpretação e de difícil configuração. SVM permite ao usuário escolher o *kernel* que irá ser utilizado para o aprendizado, o *kernel* pode ser tanto linear como não linear. Além disso, o treinamento do modelo é muito lento, quando comparado a técnicas menos sofisticadas.

É possível também realizar classificação através de técnicas de agrupamento (aprendizagem não supervisionada), utilizando informação da classe mais comum das instâncias dentro dos grupos. Também é possível utilizar algoritmos de descoberta de regras associativas, como o *Apriori* (Agrawal 1994), para fazer classificação.

Devido a grande diversidade de ferramentas disponíveis para criação de modelos preditivos, geralmente é difícil escolher qual a melhor opção para um determinado problema. Assim, é importante conhecer os principais tipos de algoritmos e suas principais características para ver algum que se adeque aos tipos de dados, tipo de resposta que se quer e requisitos da aplicação. O que se faz, na prática, é selecionar alguns algoritmos e tentar aplicá-los e comparar empiricamente os resultados.

2.4 Avaliação e comparação de modelos preditivos

Modelos preditivos nem sempre acertam suas previsões, quanto mais complexo o problema, mais difícil realizar as previsões corretamente. Assim, é de fundamental importância poder estimar corretamente a capacidade preditiva de um modelo gerado para saber, por exemplo, a confiabilidade dele em aplicações reais, especialmente em aplicações médicas. Além disso, a avaliação de modelos preditivos também é essencial para escolher a configuração de parâmetros de um determinado algoritmo (como por exemplo, o número de neurônios ocultos nas camadas de uma rede neural artificial, ou ainda, o número de vizinhos

em K-NN) que gera o modelo de maior capacidade preditiva ou ainda tentar escolher modelos preditivos gerados por diferentes algoritmos.

O objetivo da avaliação de modelos preditivos é estimar a probabilidade de o modelo realizar previsões corretas quando um novo dado é apresentado. As subseções a seguir apresentam aspectos relevantes para realizar avaliação de modelos.

2.4.1 Métricas de Avaliação

A capacidade preditiva de um modelo em um conjunto de dados de teste (rotulados) pode ser avaliada através de diversas métricas. Quando o objetivo da aplicação é prever uma determinada classe, como por exemplo, estimar se um indivíduo de uma população terá diabetes, pode-se considerar tal classe como a classe positiva, enquanto a ausência de diabetes seria a classe negativa (classificador binário). Nesses casos, existem quatro medidas elementares que servem como base para as métricas de avaliação de modelos: verdadeiros positivos (TP) são observações classificadas como positiva e que possuem o rótulo de classe positiva; verdadeiros negativos são observações negativas (TN) classificadas corretamente; falsos positivos (FP, ou falso alarme) são exemplos classificados erroneamente como positivos; e, falsos negativos (FN) são exemplos classificados erroneamente como negativos. Tais medidas elementares podem ser dispostas visualmente em uma Tabela de contingência, também chamada de matriz de confusão.

A partir desses valores é possível derivar uma série de outras métricas que podem servir para avaliar a qualidade de um modelo. Em epidemiologia, as métricas mais utilizadas são **sensibilidade** e **especificidade**. Sensibilidade (outros nomes incluem: taxa de verdadeiros positivos, *recall*, taxa de acertos ou abrangência) é a razão entre os verdadeiros positivos e todos os positivos, ou seja, quantos dos casos positivos o modelo realmente acertou. Especificidade (ou taxa de verdadeiros negativos), por outro lado, é a razão entre os casos classificados corretamente como negativos e todos os casos negativos. Muitas vezes, através de alterações de parâmetros do algoritmo, é possível variar entre essas duas métricas, ou seja, aumentar a sensibilidade ao preço da especificidade. Por isso, é importante conhecer as diferenças entre elas uma vez que em certas aplicações é mais significativa uma métrica com relação à outra. Em alguns casos, pode ser mais interessante para o modelo ter uma alta taxa de acertos positivos, mesmo que para isso o modelo tenha uma alta taxa de falsos alarmes, enquanto em outros tal abordagem pode trazer um custo muito alto. No presente trabalho, foi

utilizada a média aritmética entre sensibilidade e especificidade como um dos critérios de seleção de modelo.

Outra medida bastante utilizada em aprendizagem de máquina é a **acurácia** do modelo que é calculada como a taxa de acertos (TP + TN) com relação a todos os casos ou a taxa de **erros** do modelo (FP + FN) com relação a todos os casos. Contudo tais métricas são inadequadas quando existe uma distribuição desbalanceada das classes nos dados, como no trabalho atual. Por exemplo, se em um determinado conjunto de testes, somente 10% da população possui câncer e um modelo construído para detectar câncer prever sempre que um paciente não tem câncer, o modelo possuirá 90% de acurácia, apesar de o modelo não ter utilidade nenhuma. É muito comum ter classes desbalanceadas em aplicações médicas.

Outras métricas incluem: precisão ou Valor Preditivo Positivo (taxa dos verdadeiros positivos com relação a todos classificados como positivos), taxa de falsos alarmes (complemento da precisão), valor preditivo negativo (taxa dos verdadeiros negativos com relação aos classificados como negativos) e *fall-out* (ou taxa de falsos positivos, que é o complemento da especificidade). A medida F (*F-score)* é a média harmônica entre precisão e sensibilidade. A contagem *Kappa* (*Cohen's Kappa statistic*), que mede o grau de concordância entre dois avaliadores, a contagem *Brier* (*Brier Score*), bastante adequada para medir a calibração de previsões probabilísticas, entre outras métricas (ou variações das mesmas) também podem ser utilizadas para avaliar modelos preditivos.

Quando o classificador apresenta uma resposta contínua ou uma estimativa de probabilidade, é possível variar um limiar de decisão onde instâncias em lados opostos ao limiar apresentam classes diferentes. Através da variação desse limiar é possível desenhar uma curva plotando as taxas de verdadeiros positivos (sensibilidade) e de falsos positivos (inverso da especificidade) em cada limiar (ou de algum parâmetro do classificador). Tal curva é chamada de curva ROC (*Receiver Operating Characteristic*), pois ela apresenta uma comparação de duas características receptoras (sensibilidade e especificidade) conforme um critério muda, retratando os *trade-offs* entre os verdadeiros positivos (benefícios) e os falsos positivos (custos). A área sob a curva ROC (**AUC**, *Area Under the Curve*, também conhecida por *c-statistic*) também é utilizada para avaliação e comparação de modelos preditivos e é bastante utilizada em modelagem preditiva em saúde.

2.4.2 Métodos de avaliação

A regra mais fundamental para avaliação de modelos é que o cálculo das métricas de avaliação nunca deve ser feito utilizando dados que foram empregados para o treinamento do modelo. Uma das principais dificuldades da geração de modelos preditivos é a tendência que muitos algoritmos (principalmente os não paramétricos) têm de se sobreajustar aos dados de treinamento. Isso significa que, quando eles são testados utilizando esses dados, eles apresentam muito bom desempenho. Contudo, isso não garante que esse desempenho seja o mesmo para dados ainda não vistos, em outras palavras, o algoritmo pode não generalizar para dados novos.

A ideia básica, portanto, é utilizar uma parte dos dados disponíveis para treinamento/aprendizado do modelo (conjunto de treinamento) e separar uma parte para ser utilizada para teste (conjunto de teste). Além disso, a maioria dos algoritmos precisa definir uma configuração de parâmetros, sendo necessário um terceiro conjunto de dados (conjunto de validação) para testar se uma determinada configuração é melhor que outra em um mesmo esquema de aprendizagem. Tal estratégia de validação é conhecida como *holdout*. Outra preocupação que deve ser levada em consideração para estimativas confiáveis é que a distribuição das classes no conjunto de treinamento seja a mesma do conjunto de teste, tal característica é obtida através de um processo de estratificação quando for feita a amostragem dos dados para os diferentes conjuntos.

Para fazer a validação, o método mais utilizado em aprendizagem de máquina é a validação cruzada. O método consiste em dividir o conjunto de dados em um determinado número *k* (normalmente 10) estratos independentes de dados (*folds*), utilizar um dos estratos como conjunto de teste e os restantes para treinamento. O processo é repetido variando o estrato de teste de forma que no fim do processo todos os estratos são utilizados como teste exatamente uma vez. Esse processo é conhecido como ***k-fold cross-validation*** e resulta em k modelos construídos com diferentes conjuntos de dados e testados com conjuntos de dados independentes dos de treinamento. Os resultados da avaliação dos modelos gerados são então combinados. Para uma estimativa mais confiável, o processo pode ser repetido por várias vezes (normalmente 10), particionando os dados de forma diferente a cada vez.

Um caso específico desse método acontece quando são utilizados *n*-1 instâncias para treinar o modelo (sendo *n* a quantidade total de instâncias) e 1 instância para teste do modelo a cada iteração. Tal processo é conhecido pelo nome de ***leave-one-out cross-validation***.

Devido à quantidade de iterações, o processo tem alto custo computacional, contudo ele dá uma boa estimativa do erro do esquema de aprendizagem.

Sendo assim, a avaliação de modelos tem dois objetivos principais: (i) Escolher dentre diversas configurações de um determinado esquema de aprendizagem qual a que mais é adequada aos dados e, (ii) estimar o desempenho que um determinado modelo teria caso fosse aplicado a novos dados (generalização) com um determinado grau de certeza.

3 TRABALHOS RELACIONADOS

O presente Capítulo é dividido em duas seções. A Seção 3.1 apresenta uma relação de revisões sistemáticas sobre desenvolvimento de modelos de risco de diabetes. A Seção 3.2 apresenta trabalhos que compararam algoritmos de aprendizagem de máquina para a criação de modelos preditivos de diabetes.

3.1 Revisões sistemáticas

A construção de modelos preditivos de diabetes pode ser encontrada em diversos estudos recentes. De forma geral, tais estudos podem possuir características distintas com relação principalmente ao conjunto de dados, técnicas e métodos utilizados para a criação e validação dos modelos. Mais características que podem variar são: a definição da variável alvo, os objetivos do modelo, tipos de variáveis utilizadas, entre outras. Contudo, na grande maioria dos trabalhos, a construção dos modelos é realizada com técnicas convencionais, como Regressão Logística e Regressão Cox.

Buijsse et al (2011) fez uma revisão sistemática de escores de risco de diabetes, evidenciando suas validações em populações externas e explorando questões metodológicas sobre desenvolvimento, validação e comparação de escores de risco. Foram analisados ao todo 56 artigos. A grande maioria dos modelos foi desenvolvida utilizando Regressão Logística, enquanto alguns foram criados através de Regressão Cox. Os autores concluem que, em geral, os escores de risco mostraram uma boa habilidade discriminatória nas populações para os quais eles foram criados, contudo o desempenho é mais fraco em populações externas, sugerindo que os escores de risco precisam ser validados dentro da população que eles foram concebidos.

Thoopputra et al (2012) revisou artigos relatando desenvolvimento de ferramentas de risco de diabetes buscando informações sobre a efetividade e limitações dessas ferramentas. Foram coletadas 41 ferramentas, desenvolvidas em 22 países, sendo que a maioria (26) foi desenvolvida na América do Norte e Europa. Todas as ferramentas possuem formato de pequenos questionários contendo de 2 a 16 perguntas básicas do tipo: idade, gênero, circunferência da cintura, IMC, histórico familiar de hipertensão e diabetes e uso de medicamentos anti-hipertensivos. Três das ferramentas foram criadas através de árvores de decisão, as demais foram criadas através de Regressão Logística. O desempenho geral das ferramentas demonstrou de moderada a alta acurácia alcançando valores entre 40-97%, 24-86% e 62-87% de sensibilidade, especificidade e AUC, respectivamente. Contudo, os autores

afirmam que são necessárias comparações futuras mais detalhadas para avaliar se tais ferramentas podem ser utilizadas adequadamente na prática.

O trabalho de (Abbasi et al 2012) teve por objetivo identificar os mais relevantes modelos existentes para prever risco futuro (incidência) de diabetes tipo 2 e validou-os em um grande conjunto de dados independente. Foram incluídos na revisão 16 artigos contendo 25 modelos. Desses modelos, 12 foram considerados básicos, ou seja, criados somente com variáveis não invasivas enquanto os outros 13 foram considerados estendidos porque utilizavam informações mais complexas como biomarcadores. Todos os modelos identificados foram criados com uma das seguintes técnicas: Regressão Logística, *Cox* e *Weibull*. A validação externa foi realizada utilizando dados da coorte holandesa do estudo "Investigação prospectiva europeia de câncer e nutrição" (EPIC-NL). Após exclusões necessárias, foram utilizados 38.379 casos para avaliar os modelos básicos e 2.506 para os modelos estendidos (pois somente esses continham as variáveis necessárias para esse tipo de modelo). A discriminação dos modelos (a habilidade de o modelo distinguir indivíduos com alto risco dos com baixo risco) foi medida em termos de "*c-statistic*" (equivalente à AUC) enquanto a calibração dos modelos (habilidade do modelo estimar corretamente os riscos absolutos) foi avaliada através de plotagens de calibração e da métrica "*Hosmer-Lemeshov goodness of fit*". Foram utilizadas diferentes janelas de tempo para as avaliações uma vez que cada modelo foi criado para uma janela de tempo distinta. Os modelos básicos atingiram valores de discriminação entre 0,74 e 0,84 de *c-statistic* para risco em 7,5 anos. Nos modelos estendidos o valor ficou entre 0,81 e 0,93. Ou seja, o uso de biomarcadores aumentou a qualidade dos modelos gerados significantemente. Segundo os autores, os modelos existentes apesar de terem um bom desempenho para identificar casos com alto risco de diabetes eles não quantificam satisfatoriamente o risco atual de diabetes no futuro.

Collins et al (2011) apresenta estudos que relatam métodos utilizados para desenvolver modelos para prever risco de ter diabetes não diagnosticado (prevalência) ou risco de ter diabetes no futuro (incidência). Foram extraídas informações essenciais que descrevem aspectos do desenvolvimento do modelo tais como: projeto, tamanho da amostra e número de evento, definição da variável alvo, seleção e codificação das variáveis preditoras, tratamento de dados faltantes, estratégias de construção e aspectos de desempenho. Foram incluídos no estudo 43 modelos descritos em 32 estudos. Desses, 17 relataram o desenvolvimento de modelos de incidência de diabetes e 15, de prevalência de diabetes. Os autores identificaram uma série de pontos críticos que, segundo os autores, podem prejudicar o desempenho do modelo. Em 9 estudos, o número de eventos (casos positivos) por variável foi menor que 10.

Oito estudos utilizaram triagem univariada para seleção de variáveis, tal método, segundo os autores, não é aconselhável para tal tarefa porque não leva em consideração relações entre as variáveis preditoras. Além disso, 14 estudos não relataram como foi realizada seleção de variáveis enquanto 10 estudos não deixaram claro qual método foi utilizado. Em 21 modelos todas as variáveis quantitativas foram categorizadas. Isso, segundo os autores, também pode diminuir a qualidade do modelo gerado. Em 16 estudos não foi relatado o tratamento dos dados faltantes. No geral, foram identificados muitos artigos mal relatados, onde faltam detalhes fundamentais para julgar a utilidade dos modelos. Os algoritmos utilizados para o desenvolvimento dos modelos estudados foram: Regressão Logística em 29 artigos; "*Cox proportional hazards*" em sete artigos, "particionamento recursivo" em dois artigos e *Weibull* em um artigo.

Em (Noble et al 2011), o objetivo da revisão sistemática foi avaliar modelos de risco de diabetes tipo 2 e informar suas implementações na prática. Foram incluídos 43 artigos descrevendo desenvolvimento ou validação de 95 modelos. Alguns dos modelos apresentaram propriedades estatísticas robustas (por exemplo, boa discriminação e calibração) e foram validados externamente em diferentes populações. Segundo os autores, em geral, a adição de marcadores genéticos não impactou relevantemente no desempenho dos modelos com relação aos modelos criados somente com fatores sociodemográficos e clínicos. A maioria dos autores descrevem seus modelos como simples ou facilmente implementados, contudo poucos foram específicos sobre o público-alvo e as circunstâncias de utilização do modelo. Por fim, os autores concluem que, apesar de existirem muitos modelos disponíveis, a maior parte deles é raramente utilizada porque requerem testes ou exames que não estão rotineiramente disponíveis ou porque foram desenvolvidos sem um usuário específico ou claro em mente. Este trabalho não comentou sobre quais técnicas foram utilizadas para construção dos modelos analisados.

Barber et al (2014) pesquisou por artigos que relataram o desenvolvimento de ferramentas de risco de pré-diabetes. Foram identificadas 18 ferramentas das quais 11 foram criadas utilizando Regressão Logística, 6 utilizando árvores de decisão e 1 utilizando SVM. As variáveis mais utilizadas foram: idade, IMC e histórico familiar de diabetes e hipertensão. O tamanho dos conjuntos de dados e o número de variáveis por evento foram considerados aceitáveis em todas as ferramentas. Oito modelos não discutiram tratamento de dados faltantes e 10 modelos, criados com regressão logística, categorizaram todas as variáveis quantitativas, o que os autores consideraram pontos críticos que podem influenciar negativamente a qualidade dos modelos criados. Sete das ferramentas relataram validação em

um conjunto de dados externo. Por fim os autores concluem que a avaliação da calibração e validade das ferramentas dentro da população de interesse ainda é um quesito que precisa ser melhorado.

Ao contrário do trabalho realizado na presente dissertação de mestrado, a grande maioria dos modelos descritos nas revisões sistemáticas apresentadas utilizaram técnicas convencionais de estatística para a construção dos modelos preditivos de diabetes. A utilização de algoritmos de aprendizagem de máquina ainda é recente nessa área e um campo a ser explorado.

Nesse sentido, o artigo (Shankaracharya et al 2010) procurou fazer um resumo de desenvolvimentos recentes e potenciais em algoritmos de aprendizagem de máquina como ferramenta de diagnose de diabetes. Foram apresentados 30 trabalhos contendo modelos criados com uma ampla variedade de algoritmos de aprendizagem de máquina. Foram apresentados, para cada modelo, os conjuntos de dados utilizados para construção e a acurácia obtida. Poucos modelos relataram as métricas sensibilidade e especificidade. A grande maioria dos modelos utilizou o conjunto de dados PID (*Pima Indian Dataset*), disponível no repositório de aprendizagem de máquina de Irvine, na Universidade da Califórnia (UCI). Tal conjunto de dados, apesar de ser bastante utilizados para comparação de algoritmos de aprendizagem de máquina, não possui a confiabilidade necessária para a construção de modelos para serem utilizados na prática, principalmente quando o público alvo não pertencer ao grupo étnico em questão.

Diferentemente da maioria dos modelos descritos nos resumos sistemáticos apresentados, os modelos criados no trabalho apresentado nesta dissertação, além de utilizar um conjunto de dados com bastante potencial para extração de conhecimento relevante que pode ser utilizado na prática, procurou testar diferentes algoritmos de aprendizagem de máquina e abordagens de pré-processamento de dados através de uma metodologia robusta de validação com o intuito de selecionar as melhores técnicas disponíveis para construção de modelos de risco de diabetes.

3.2 Comparações de algoritmos de aprendizagem de máquina

A maioria dos trabalhos que realizaram comparações de técnicas de aprendizagem de máquina para construção de modelos preditivos de diabetes compararam os resultados obtidos com o uso de uma ou duas técnicas específicas com os resultados provenientes de Regressão

Logística, tida como referência nesse tipo de estudo. Uma das principais diferenças desses estudos é o conjunto de dados utilizado para a construção dos modelos.

O estudo (Choi et al 2014) apresenta a criação de modelos de risco de pré-diabetes utilizando técnicas de Redes Neurais Artificiais e SVM (*Support Vector Machines*) em dados de 4.685 participantes do estudo KNHANES (*Korean National Health and Nutrition Examination Survey*) coletados entre 2010 e 2011. Os resultados foram comparados com o anterior (Lee et al 2012) que utilizou Regressão Logística no mesmo conjunto de dados. Lee utilizou dados de 2001 e 2005 do KNHANES para desenvolver o modelo utilizando regressão logística e validou os modelos utilizando dados do mesmo estudo em outro período (2007-2008) obtendo 80% de sensibilidade, 53% de especificidade (Acurácia Balanceada seria de 66,5%) e AUC de 0,73. Na validação de Choi, utilizando diferente parte do conjunto de dados, o modelo de Lee obteve 0,712 de AUC enquanto os novos modelos, criados com SVM e Redes Neurais obtiveram 0,731 e 0,729, respectivamente, se mostrando mais eficazes que o modelo anterior para triagem de pré-diabetes.

Os trabalhos (Wang et al 2013) e (Mansour et al 2013) também compararam técnicas de Redes Neurais Artificiais com Regressão Logística para criação de modelos preditivos de diabetes. O primeiro utilizou dados de entrevistas de 8.640 moradores (760 casos positivos) de uma zona rural na China. Nesse trabalho os modelos criados com Redes Neurais apresentaram melhores resultados (AUC de 0,891 ± 0,015) comparados com os criados com Regressão Logística (0,744 ± 0,021). O segundo comparou Redes Neurais Artificiais do tipo RBF (*Radial Basis Function*), Regressão Logística e *Discriminant Analysis* para criar modelos com objetivo de distinguir entre indivíduos com pré-diabetes e diabetes utilizando dados de 200 pessoas (100 casos de diabetes e 100 com pré-diabetes) de 17 centros de saúde rurais da cidade de Kermanshah no Iran. Os AUCs obtidos para RBF, Regressão Logística e *Discriminant Analysis* foram de 0,864, 0,884 e 0,8, respectivamente. Os altos valores de AUC, comparados a outros modelos apresentados ou com o presente trabalho, devem-se ao fato que foram utilizados resultados de exames para criação dos modelos, tais como pressão sanguínea, lipídios no sangue (Triglicerídeos), tolerância à glicose e glicemia em jejum.

Outro estudo (Lee et al 2014) comparou a criação de modelos de risco de diabetes utilizando as técnicas de Regressão Logística e *Naïve Bayes* utilizando dados de 4.870 pessoas (2.955 mulheres e 1915 homens) do estudo KHGES (*Korean Health and Genome Epidemiology Study*) utilizando somente medidas antropométricas. Foram criados modelos distintos para homens e para mulheres. Os resultados de AUC dos modelos criados com

Regressão Logística e *Naïve Bayes* foram de 0,741 e 0,739, respectivamente, nos modelos para mulheres e 0,687 e 0,686, respectivamente, nos modelos para homens.

De forma geral, percebe-se que a utilização de técnicas de aprendizagem de máquina pode se mostrar uma alternativa viável para a construção de modelos preditivos de diabetes, apresentando bons resultados ao lado de técnicas convencionais estatísticas. Porém, foram encontrados poucos trabalhos comparando diferentes algoritmos para criar modelos preditivos utilizando dados reais. Além disso, nesses trabalhos, são comparados dois ou, no máximo, três algoritmos diferentes. Também não foram encontrados estudos relatando desenvolvimento de modelos preditivos de diabetes ou comparando algoritmos de aprendizagem de máquina utilizando dados do ELSA-Brasil, como foi feito no presente trabalho.

O presente trabalho evidenciou um processo de desenvolvimento de modelos preditivos, cuidando especialmente da validação dos modelos para obter estimativas mais confiáveis sem utilizar dados externos. Os resultados obtidos neste trabalho se mostraram tão bons quanto, e algumas vezes superiores, os resultados obtidos nos trabalhos apresentados neste Capítulo. Além disso, nenhum dos trabalhos encontrados que comparavam diferentes algoritmos de aprendizagem de máquinas utilizando um conjunto de dados real comparou tantos algoritmos de aprendizagem de máquina como foram comparados neste trabalho.

4 METODOLOGIA

Este Capítulo descreve a metodologia utilizada para a criação e comparação dos modelos preditivos para detecção de diabetes utilizando dados do ELSA-Brasil e diferentes algoritmos de aprendizagem de máquina. Ele é dividido da seguinte forma: a Seção 4.1 apresenta detalhes sobre o conjunto de dados utilizado para criação dos modelos preditivos e sobre a pré-seleção e descrição das variáveis candidatas; a Seção 4.2 apresenta os algoritmos utilizados no trabalho; a Seção 4.3 detalha as técnicas de pré-processamento para preparar os dados para a criação dos modelos; a Seção 4.4, descreve o processo geral utilizado para o desenvolvimento e avaliação dos modelos preditivos; por fim, a Seção 4.5 mostra como foi feito o desenvolvimento do protótipo da ferramenta web para detecção de diabetes não diagnosticado.

4.1 Conjunto de Dados e Pré-seleção de Variáveis

Os dados utilizados neste trabalho para a criação dos modelos preditivos são provenientes da linha de base do ELSA-Brasil (Estudo Longitudinal de Saúde do Adulto) que aconteceu entre 2008 e 2010. Esse estudo foi criado com a finalidade principal de investigar fatores de risco de doenças crônicas, principalmente diabetes e doenças cardiovasculares (Aquino et al 2012, Schmidt et al 2014).

No ELSA-Brasil, os participantes são avaliados através de entrevistas, exames clínicos e laboratoriais. As entrevistas abordam, entre outros aspectos, os seguintes temas: história educacional e ocupacional; características e composição do domicílio e da família; história reprodutiva (mulheres); hábitos alimentares; atividade física no trabalho e no lazer; autopercepção de saúde; internações, acidentes de trabalho e de trânsito; uso de serviços de saúde e práticas preventivas e de detecção precoce de doenças. Os exames abordam medidas antropométricas, exames de sangue e de urina, eletrocardiograma, entre outros. O estudo é bastante completo, disponibilizando para esta análise aproximadamente 1.600 variáveis para cada participante.

Dentre essas variáveis, 266 são derivadas. Variáveis derivadas são variáveis criadas (derivadas) a partir da combinação de uma ou mais variáveis básicas ou outras derivadas, considerando definições clínicas e epidemiológicas relevantes. Por exemplo, a variável “a_imc1” (Índice de Massa Corporal) é uma das variáveis derivadas presentes no ELSA-Brasil e pré-selecionada na análise deste trabalho. Essa variável é formada pela variável básica “anta3”, que traz o valor da altura em pé (em cm) do participante, e pela variável

derivada “a_peso”, que é o peso corrigido. A variável de peso corrigido, por sua vez, é formada pela variável básica “anta5”, que é o peso calculado na avaliação física do participante descontando o peso do uniforme do participante por centro, variável derivada “a_punif”. Essa variável derivada é formada a partir da variável básica “rcpa13” que diz o tamanho do uniforme do participante (PP, P, M, G, X) e da variável básica “centroa” que diz qual estado do país o participante foi avaliado. Ou seja, todas essas informações influenciam no valor da variável quantitativa contendo o Índice de Massa Corporal.

A variável derivada, categórica, “a_imc2”, que representa o Índice de Massa Corporal categorizada com quatro categorias (magreza, eutrofia, sobrepeso e obesidade), foi criada a partir da variável “a_imc1” utilizando os seguintes pontos de corte: 18,5; 25 e 30. Ou seja, os participantes que tiverem um a_imc1 menor que 18,5 são incluídos na primeira categoria da variável “a_imc2”, que significa “Magreza”; os participantes com a_imc1 igual ou maior que 18,5 e menor que 25 são enquadrados na segunda categoria, que significa “Eutrofia”, e assim por diante. A definição dos pontos de corte foi realizada pela equipe do ELSA-Brasil, baseando-se em conhecimento da área da saúde e é independente deste trabalho.

Todas as informações sobre a criação das variáveis derivadas foram fornecidas junto com o conjunto de dados pela equipe do ELSA-Brasil. No total, o estudo acompanha 15.105 funcionários de seis instituições públicas de ensino e pesquisa de diferentes regiões do Brasil. Os participantes são adultos entre 35 e 74 anos de idade, sendo que 45,6% são homens e 54,4%, mulheres. Outras características sobre os participantes do estudo podem ser encontradas em (Schmidt et al 2014).

Foram excluídos deste trabalho 1.473 indivíduos com diabetes autorreferida, ou seja, que sabiam que tinham diabetes e, por isso, provavelmente estariam realizando algum tipo de tratamento que poderia distorcer o treinamento dos modelos. Também foram excluídos três casos que não tinham a variável alvo definida. Dos 13.632 participantes restantes, 1.497 (aprox. 11%) tinham diabetes não diagnosticado. Diabetes não diagnosticado foi considerado presente quando, nos exames laboratoriais, o participante tinha glicemia de jejum >= 126 mg/dL, glicemia 2h após sobrecarga >= 200 mg/dL ou hemoglobina glicada (HbA1C) >= 6.5%.

No trabalho atual foi utilizado apenas um subconjunto de referência das variáveis disponíveis. Para encontrar esse subconjunto foi realizada uma pesquisa sobre fatores de risco de diabetes e variáveis utilizadas na construção de escores de risco de diabetes na literatura. As fontes para essa pesquisa foram os trabalhos relacionados apresentados no Capítulo 3, principalmente as revisões sistemáticas apresentadas na Seção 3.1. Nessa busca foram

excluídos todos os fatores que tivessem algum custo de obtenção como, por exemplo, resultados de exames clínicos e laboratoriais. Dessa forma, o modelo resultante do trabalho pode ser aplicado em uma população a partir de um questionário simples sobre estilo de vida, histórico de saúde e familiar e medidas antropométricas simples.

A partir desses fatores, foi realizada uma busca no conjunto de variáveis do ELSA-Brasil para encontrar quais variáveis correspondiam aos fatores pesquisados. O conjunto de variáveis encontrado foi então validado por especialistas e chegou-se ao conjunto de 34 variáveis preditoras descritas nas Tabelas 4.1 (que mostra as variáveis categóricas) e 4.2 (que mostra as variáveis quantitativas). A Tabela 4.1 também contém a variável-alvo do estudo, que é a variável "a_dm", indicando se a pessoa tem ou não diabetes.

As colunas da Tabela 4.1 têm os seguintes significados. A coluna "*Nome da Variável*" mostra o nome de cada variável disponibilizada pelo conjunto de dados e serve como um identificador da variável. A coluna "# Níveis" apresenta a quantidade de categorias (níveis) da variável categórica, ou seja, os valores distintos que ela possui no conjunto de dados. A coluna "*Frequência por nível*" apresenta a quantidade de casos existentes em cada nível no conjunto de dados e a quantidade de casos com valores faltantes (NA). A coluna "*Descrição*" mostra uma descrição sucinta do significado da variável enquanto a coluna "*Valores Possíveis*" mostra o que o valor da variável significa dentro do domínio.

Tabela 4.1 – Variáveis Categóricas

Nome da Variável	# Níveis	Frequência por nível	Descrição	Valores Possíveis
a_ativfisica	3	1: 10318; 2: 1854; 3: 1253; NA's: 204;	Atividade física no lazer	1 = Fraca; 2 = Moderada; 3 = Forte
a_bebexcessivo	2	0: 12578; 1: 1028; NA's: 23;	Bebedor excessivo	0 = Não; 1 = Sim
a_binge	2	0: 11771; 1: 1835; NA's: 23;	Bebedor excessivo esporádico	0 = Não; 1 = Sim
a_chdhard	2	0: 11561; 1: 510; NA's: 1558;	Doença Coronariana grave autorreferida	0 = Não; 1 = Sim
a_chdlight	2	0: 11331; 1: 756; NA's: 1542;	Doença Coronariana leve autorreferida	0 = Não; 1 = Sim
a_consdiafrutas	2	0: 5929; 1: 7671; NA's: 29;	Consumo diário de frutas	0 = Não; 1 = Sim
a_consdiaverduras	2	0: 6538; 1: 7060; NA's: 31;	Consumo diário de verduras e legumes	0 = Não; 1 = Sim
a_dm	**2**	**1: 12132; 0: 1497;**	Diabetes mellitus	**0 = Sim; 1 = Não***
a_escolar	4	1: 713; 2: 872; 3: 4673; 4: 7371;	Escolaridade	1 = Fundamental Incompleto; 2 = Ensino Médio incompleto; 3 = Ensino Médio Completo; 4 = Superior Completo
a_fumante	3	0: 7863; 1: 3968; 2: 1798;	Fumante	0 = Nunca Fumou; 1 = Ex fumante; 2 = Fumante
a_gidade	4	1: 3241; 2: 5499; 3: 3629; 4: 1260;	Grupo de idade do participante	1 = 35 a 44 anos; 2 = 45 a 54 anos; 3 = 55 a 64 anos; 4 = 65 a 74 anos
a_imc2	4	1: 139; 2: 5155; 3: 5462; 4: 2867; NA's: 6;	Índice de Massa Corporal com 4 categorias	1 = Magreza; 2 = Eutrofia; 3 = Sobrepeso; 4 = Obesidade
a_medanthipert	2	0: 10130; 1: 3485; NA's: 14;	Uso de medicamento anti-hipertensivo	0 = Não; 1 = Sim
a_medoutahip	2	0: 13547; 1: 82;	Uso de outros anti-hipertensivos	0 = Não; 1 = Sim
a_medredlip	4	0: 12191; 1: 1209; 2: 107; 3: 122;	Uso de hipolipemiantes	0=Não usa; 1= Uso de Estatina; 2 = Uso de outro; 3 = Uso de mais de um tipo
a_prvdcc	2	0: 11719; 1: 312; NA's: 1598;	Prevalência de doença coronariana	0 = Não; 1 = Sim
a_sfhfprem	2	0: 13541; 1: 85; NA's: 3;	Insuficiência Cardíaca (<50 anos) autorreferida	0 = Não; 1 = Sim
a_sfmiprem	2	0: 13561; 1: 65; NA's: 3;	Infarto do Miocárdio (<50 anos) autorreferido	0 = Não; 1 = Sim
a_sfrevprem	2	0: 13579; 1: 47; NA's: 3;	Revascularização (<50 anos) autorreferida	0 = Não; 1 = Sim
a_sfstkprem	2	0: 13551; 1: 77; NA's: 1;	Derrame (<50 anos) autorreferido	0 = Não; 1 = Sim
a_sintsono	2	0: 9081; 1: 4536; NA's: 12;	Qualidade do sono	0 = Não; 1 = Sim
a_sitconj	5	1: 9012; 2: 2229; 3: 1407; 4: 527; 5: 454;	Situação conjugal	1 = Casado; 2 = Divorciado; 3 = Solteiro; 4 = Viúvo; 5 = Outro;
claa2	2	0: 9130; 1: 4463; NA's: 36;	Dor/desconforto pernas quando anda (Q2)	0 = Não; 1 = Sim
diea133	3	0: 1118; 1: 11889; 2: 287; NA's: 335;	Consome café (Q133)	0 = Não; 1 = Sim, com cafeína; 2 = Sim, descafeinado
hfda07	2	0: 3508; 1: 9846; NA's: 275;	Hipertensão na família (Q7)	0 = Não; 1 = Sim
hfda11	2	0: 8504; 1: 4940; NA's: 185;	Diabetes na família (Q11)	0 = Não; 1 = Sim
hmpa08	2	0: 8989; 1: 4614; NA's: 26;	Colesterol alto (Q8)	0 = Não; 1 = Sim
rcta8	2	1: 6105; 2: 7524;	Sexo	1 = Masculino; 2 = Feminino

Fonte: Autoria Própria.

Além das colunas "*Nome da Variável*" e "*Descrição*", que já foram apresentadas na descrição da Tabela 4.1, a Tabela 4.2 apresenta as estatísticas básicas dos valores cada variável quantitativa no conjunto de dados: Mínimo, Mediana, Média, Máximo, Desvio Padrão (coluna "D.P."). Também apresenta na coluna "NA's" contendo a quantidade de valores faltantes de cada variável.

Tabela 4.2 – Variáveis Quantitativas

Nome da Variável	Mínimo	Mediana	Média	Máximo	D.P.	NA's	Descrição
a_cint	56,40	89,50	90,36	200,00	12,53	2	Circunferência da Cintura (cm)
a_cons_est_nacl	0,14	9,67	11,65	3533,00	42,57	402	Consumo diário de sal estimado
a_imc1	14,43	26,14	26,76	58,29	4,63	6	Índice de massa corporal (kg/m²)
a_rcq	0,40	0,89	0,89	1,27	0,09	7	Relação cintura-quadril
a_rendapercapita	27,63	1410,90	1756,34	7884,50	1436,67	61	Renda familiar per capita
a_volalc	0,00	0,00	68,68	3906,00	136,67	23	Quantidade de álcool ingerida por semana (ml)
afia7	0,00	0,00	0,12	7,00	0,68	199	Uso de bicicleta para transporte (dias/semana)

Fonte: Autoria Própria.

4.2 Técnicas de Aprendizagem de Máquina

Os modelos preditivos de diabetes não diagnosticado foram criados através dos seguintes algoritmos de aprendizagem de máquina: Regressão Logística, *Multi-Layer Perceptron/Backpropagation* (Redes Neurais Artificiais), *Naïve Bayes (*Redes Bayesianas), *k-Nearest Neighbors (Instance Based Learning), Quinlan's C5 (*Árvores de Decisão), *RIPPER (Regras de Decisão) e Random Forest (Ensemble).*

4.3 Preparação dos Dados

O tratamento dos dados faltantes ocorreu da seguinte forma. De acordo com a Tabela 4.1, dentre as variáveis pré-selecionadas, três delas (Doença coronariana autorreferida leve, grave e prevalência de doença coronariana) tinham uma quantidade significativa de dados faltantes (>10% dos casos). Esses valores não foram preenchidos pois os entrevistados não sabiam responder a pergunta em questão e devido ao número muito baixo de respostas positivas para essa questão entre os participantes a chance de a resposta ser negativa nesses casos é alta. Assim eles foram substituídos por respostas negativas, modificando a semântica da resposta de "não" para "não ou não sabe" e permitindo a utilização dos casos que continham somente alguma dessas variáveis sem valor. Isso pode atrapalhar o treinamento dos modelos mas o ganho de informação ao incluir os casos que possuíam somente alguma das três variáveis nas análises provavelmente é maior. Após isso, os demais casos que continham pelo menos uma das demais variáveis com valores faltantes foram eliminados (total de 1.118 exclusões).

Para o tratamento das diferentes escalas, a cada construção de um modelo preditivo, todas as variáveis quantitativas foram padronizadas conforme descrito na Seção 2.2.5, subtraindo o valor original pela média dos valores da variável nos dados de treinamento e dividindo pelo desvio padrão. Para a validação/teste do modelo, as variáveis do conjunto de teste também foram padronizadas utilizando para tanto a mesma média e desvio padrão utilizados para padronizar as variáveis do conjunto de treinamento.

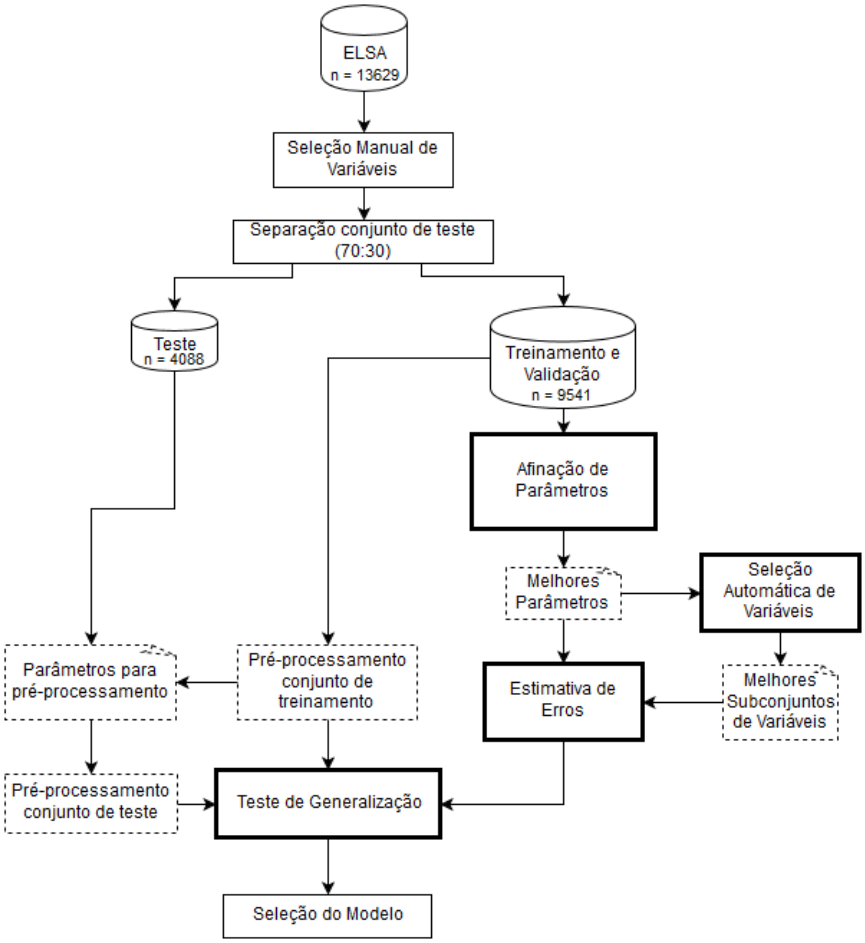
Quanto ao tratamento dos diferentes tipos de dados, foram realizadas as seguintes transformações: a) todas variáveis quantitativas para categóricas através de categorização de dados; b) todas variáveis categóricas para quantitativas através de dicotomização; c) sem realizar transformações e deixando que a ferramenta ou o próprio algoritmo faça o tratamento adequado para os diferentes tipos de variáveis.

4.4 Processo Geral

A criação, avaliação e comparação dos modelos foram realizadas em quatro etapas sequenciais: (i) afinação dos parâmetros configuráveis de cada algoritmo, (ii) seleção automática de variáveis, (iii) estimativa de erro e (iv) teste de generalização em conjunto de dados independente.

O processo geral utilizado pode ser visualizado na Figura 4.1. Primeiramente, foi realizada uma pré-seleção manual de variáveis como explicado na Seção 3.1 (bloco "*Seleção Manual de Variáveis*" na Figura 4.1). Após, uma parte do conjunto de dados (conjunto "*Teste*" na Figura 4.1), contendo 3.755 casos completos (aprox. 30% dos dados), foi separado para o teste de generalização (etapa (iv)), enquanto o restante (conjunto "Treinamento e Validação" na Figura 4.1), contendo 8.682 casos completos, foi utilizado para as etapas (i), (ii) e (iii).

Figura 4.1 – Processo utilizado no trabalho



Fonte: Autoria Própria.

A primeira etapa (bloco "*Afinação de Parâmetros*" na Figura 4.1) foi realizada através de validação cruzada avaliando os algoritmos com diferentes configurações. Os resultados dessa etapa (item "*Melhores Parâmetros*" na Figura 4.1) compreendendo os melhores parâmetros, pontos de corte e pré-processamento foram utilizados nas próximas etapas. A segunda etapa (bloco "*Seleção Automática de Variáveis*" na Figura 4.1) gerou cinco diferentes subconjuntos de variáveis utilizando diferentes algoritmos e validação cruzada (usando somente as melhores configurações encontradas na etapa anterior) através do método *wrapper* para seleção automática de variáveis. Os melhores subconjuntos descobertos nessa etapa (item "*Melhores Subconjuntos de Variáveis*" na Figura 4.1) foram utilizados nas próximas etapas (iii) e (iv). A terceira etapa (bloco "*Estimativa de Erros*" na Figura 4.1) utilizou validação para obter estimativas mais confiáveis de desempenho dos esquemas de aprendizagem utilizando as melhores configurações e os diferentes subconjuntos obtidos nas etapas anteriores. Por fim, a última etapa (bloco "*Teste de Generalização*" na Figura 4.1) utilizou o esquema de aprendizagem que obteve melhor desempenho de cada algoritmo na etapa anterior para construir modelos utilizando todo o conjunto de "Treinamento e Validação" como conjunto de treinamento e o conjunto de "Teste" para avaliar os modelos gerados.

A seguir, cada etapa é descrita com mais detalhes.

4.4.1 Afinação dos Parâmetros

A primeira etapa avaliou cada algoritmo de aprendizagem de máquina utilizando diferentes configurações de parâmetros para descobrir qual configuração produz o melhor resultado em termos das métricas de avaliação utilizadas para cada algoritmo e para cada tipo de transformação utilizada. Os parâmetros testados de cada algoritmo estão listados e descritos na Tabela 4.3.

Tabela 4.3 – Parâmetros analisados de cada algoritmo

Algoritmo	Parâmetro	Descrição
Redes Neurais	size	Número de neurônios na camada oculta.
	decay	Decaimento dos pesos.
	skip	Indicativo de ligação direta entre camada de entrada e de saída.
Regressão Logística	maxit	Número máximo de iterações do método dos mínimos quadrados ponderado.
	epsilon	Valor de tolerância de convergência.
Naïve Bayes	laplace	Número real para controlar a suavização por Laplace.
K-NN	minVotes	Número mínimo de votos para definir uma decisão.
	k	Número de vizinhos considerados.
Random Forest	ntree	Número de Árvores para criar.
C5	costs	Custo de classificação errada de um caso positivo.
	minCases	Número indicando o número mínimo de amostras que devem existir para dividir o conjunto de dados.
	fuzzyThreshold	Indicativo para avaliar possíveis divisões avançadas dos dados.
	earlyStopping	Indicativo de uso de um método interno para terminar o algoritmo.
	winnow	Indicativo se deve ser utilizado o método "*predictor winnowing*" de seleção de variáveis.
	noGlobalPrunning	Indicativo se não deve ser utilizado uma poda da árvore para simplificar o modelo final.

Fonte: Autoria Própria.

Devido à grande quantidade de valores possíveis dos parâmetros, foi necessário adotar uma estratégia de busca. Inicialmente selecionou-se manualmente um conjunto limitado de valores para cada parâmetro avaliado e, através de validação cruzada de 10 *folds* repetida por três vezes, cada combinação de valores dos parâmetros foi testada. Cada teste foi realizado utilizando um tipo de transformação (categorização, dicotomização) ou nenhuma transformação. Os resultados dos 30 modelos gerados em cada etapa da validação cruzada foram ponderados gerando uma média das métricas de avaliação para cada algoritmo, transformação e combinação de valores de parâmetros. Além disso, no caso de o modelo gerar uma estimativa de probabilidade como saída, foram extraídas métricas de classificação utilizando diferentes pontos de corte (selecionados junto com os valores dos parâmetros) para decidir qual produz o melhor resultado.

O critério utilizado para seleção, no caso do algoritmo gerar um classificador binário, foi a média da *Acurácia Balanceada*. Do contrário, no caso do algoritmo gerar uma estimativa de probabilidade como saída, utilizou-se a média de AUC como critério. Nesse caso, foi escolhido como melhor ponto de corte aquele que gerou a melhor média de *Acurácia Balanceada* (AB) para os modelos gerados pelos parâmetros selecionados.

Após isso, os resultados obtidos foram analisados e, quando necessário, novos pontos de corte e/ou valores de parâmetros (na volta dos melhores obtidos até então) foram selecionados manualmente para mais testes. Esse processo foi repetido até não haver melhora na variação dos parâmetros.

4.4.2 Seleção Automática de Variáveis

A etapa de seleção automática de variáveis teve por objetivo encontrar subconjuntos de variáveis, a partir do conjunto original contendo as 34 variáveis pré-selecionadas, que eventualmente poderiam gerar modelos com um desempenho superior aos modelos gerados com todas as variáveis pré-selecionadas. A geração dos melhores subconjuntos foi realizada através da técnica *Wrapper*, que utiliza um algoritmo de aprendizagem de máquina em conjunto com uma técnica de avaliação de modelos e uma estratégia de busca para decidir qual subconjunto de variáveis tem melhor desempenho.

Para tanto foram utilizados quatro algoritmos de aprendizagem de máquina: Regressão Logística, Redes Neurais, K-NN e *Naïve Bayes*. A estratégia de busca heurística utilizada foi a "*Forward Selection"*, já que esta tende a encontrar subconjuntos de variáveis menores. A mesma técnica de validação cruzada utilizada na etapa anterior (de 10 *folds*, repetida por três vezes) foi utilizada para avaliar os subconjuntos de variáveis, e a média dos valores de AUC foi utilizada como critério para decidir qual o melhor subconjunto de variáveis. Foram utilizados somente os melhores parâmetros obtidos na etapa anterior para configurar cada algoritmo.

4.4.3 Estimativa de Erros

Após a descoberta desses subconjuntos de dados, uma nova etapa de validação cruzada de 10 *folds*, porém repetida 10 vezes para aumentar a confiabilidade das estimativas de erro e utilizando somente os parâmetros encontrados na primeira etapa, foi realizada para cada algoritmo, tipo de transformação utilizada e subconjunto de variáveis (incluindo o subconjunto original). Nesse teste foram considerados os resultados utilizando somente a combinação de parâmetros e ponto de corte que obteve melhor resultado na própria etapa para cada algoritmo. Os resultados desse teste servem como indicativos para escolher o algoritmo e esquema de aprendizagem (combinação de valores de parâmetros, transformação utilizada no pré-processamento e subconjunto de variáveis utilizado) a ser utilizado na construção de modelos preditivos na prática.

4.4.4 Teste de Generalização

Por fim, foi realizado o teste de generalização com o objetivo de avaliar o comportamento dos modelos preditivos quando apresentados a dados ainda não vistos em nenhuma das etapas anteriores. Para isso, todos os dados de treinamento/validação utilizados

nas etapas anteriores foram utilizados para a criação dos modelos preditivos e esses modelos foram testados em um conjunto de dados separado antes da primeira etapa dos testes. Foi realizado um teste para cada algoritmo utilizando o esquema de aprendizagem que obteve melhores resultados na validação cruzada anterior. Os resultados obtidos nessa etapa servem como métrica de qualidade para os modelos que poderão ser utilizados na prática ou para futuros testes.

4.5 Ferramenta WEB para Detecção de Diabetes

Ao final do processo, o esquema de aprendizagem que obteve melhor resultado na etapa 3 (validação cruzada para estimativas de erro) e também obteve resultado satisfatório no teste de generalização, incluindo: algoritmo, parâmetros de configuração do algoritmo, pré-processamento de dados e subconjunto de variáveis utilizado; foi selecionado para servir de base para a construção de um protótipo de ferramenta WEB, no formato de questionário, para detecção de diabetes não diagnosticado.

5 DESENVOLVIMENTO

O presente Capítulo apresenta os detalhes das análises realizadas no trabalho. A Seção 5.1 apresenta detalhes sobre a pré-seleção das variáveis preditoras e sobre a importação dos dados na ferramenta de análise. A Seção 5.2 apresenta os detalhes e bibliotecas utilizados para pré-processamento dos dados. A Seção 5.3 mostra detalhes sobre o particionamento dos dados utilizados nas validações cruzadas. A Seção 5.4 descreve detalhes de implementação sobre a construção dos modelos preditivos para cada algoritmo de aprendizagem de máquina. A Seção 5.5 mostra detalhes sobre a avaliação dos modelos. A Seção 5.6 apresenta detalhes sobre a seleção automática de variáveis. Por fim, a Seção 5.7 descreve como foi feita a implementação da ferramenta WEB utilizando os resultados das análises realizados.

As análises foram realizadas utilizando a linguagem de programação e ferramenta de análise estatística R[1] versão 3.2.3. Para tarefas mais complexas como a criação e utilização de modelos preditivos foram utilizados pacotes (bibliotecas) implementados por terceiros, conforme descrito no decorrer do Capítulo. A escolha dos pacotes se deu por popularidade e/ou por familiaridade com a utilização do mesmo por parte do analista. Todos os pacotes estão disponíveis no repositório CRAN[2]. Os códigos-fonte utilizados para as análises encontram-se disponíveis publicamente para consulta no repositório do GitHub e podem ser acessados através do endereço: https://github.com/andrebrujah/ElsaPredictiveModeling.

5.1 Importação dos dados

Os dados foram disponibilizados para análise no formato binário "*Stata*[3]" (extensão *dta*). Inicialmente, foi utilizado o pacote o pacote R *foreign*[4] (versão 0.8-66) para realizar a importação desse tipo de dado diretamente na ferramenta de análise. Porém ao realizar essa importação, as informações de códigos de dados faltantes, utilizadas para uma análise preliminar do conjunto de dados, que não foi descrita nesse trabalho; foram perdidas. Por esse motivo, os dados disponibilizados foram primeiramente importados na ferramenta de análise SPSS v.18, disponibilizado pela instituição de ensino, e foram exportados para um arquivo

[1] http://www.r-project.org/
[2] https://cran.r-project.org/
[3] http://www.stata.com/

com extensão CSV ("*Comma Separated Values*"). No entanto, para exportar com sucesso, foi necessário remover algumas variáveis que continham valores de texto livre pois alguns caracteres especiais dificultavam a exportação correta do arquivo. Por fim, os dados, em formato CSV, foram importados através do método "*read.table*", do pacote "*utils*", nativo do R. Esse método permite especificar os códigos que devem ser interpretados como dados faltantes (*NA's*) entre outras definições.

No total foi disponibilizado um conjunto de dados contendo 13.632 observações, representando os participantes do estudo, descritas por 1.661 variáveis. Após a importação dos dados na ferramenta, as seguintes ações foram realizadas:

(i) Selecionaram-se, do conjunto inicial, somente as 34 variáveis preditoras em adição à variável alvo, descritas nas Tabelas 4.1 e 4.2;

(ii) Foram excluídos da análise três participantes que não tinham informações quanto à variável alvo (os participantes que tinham conhecimento da condição de diabetes, excluídos do estudo, já não faziam parte do conjunto original entregue pela equipe do ELSA-Brasil);

(iii) O código dos valores da variável alvo "a_dm" foi substituído de modo que o valor "0" representasse o caso positivo e o valor "1" representasse o caso negativo porque essa é a interpretação utilizada pela maioria das ferramentas utilizadas;

(iv) As variáveis categóricas e quantitativas foram definidas explicitamente a partir de arquivos de texto externos criados manualmente pelo analista, isso foi necessário pois a ferramenta utilizada para análise pode interpretar erroneamente os tipos das variáveis.

5.2 Preparação dos dados

As variáveis quantitativas foram padronizadas subtraindo o valor da variável pela média e dividindo pelo desvio padrão dos valores de treinamento. A padronização foi realizada em todas as etapas e em todas as variáveis quantitativas. Para tanto, foi utilizado o

[4] http://cran.r-project.org/web/packages/foreign/index.html

método "*preProcess*" do pacote "*caret*[5]" (versão 6.0-62) com o argumento "*method = c("center", "scale")*". Tanto as conversões de tipo quanto a padronização das escalas das variáveis foram realizadas utilizando exclusivamente informações obtidas a partir do conjunto de treinamento.

Após a padronização, os dados foram dicotomizados ou categorizados ou ainda não sofreram nenhum tipo de conversão de dados. A dicotomização das variáveis foi realizada com o auxílio do pacote "*dummies*[6]" (versão 1.5.6), através do método "*dummy.data.frame*". A execução padrão desse método, e que foi utilizada neste trabalho, cria uma nova variável para cada categoria da variável que está sendo dicotomizada. Por outro lado, nos casos em que não foi realizada nenhuma transformação como pré-processamento, algumas implementações utilizadas realizaram automaticamente a dicotomização, como é o caso de Regressão Logística e Redes Neurais, contudo, nesse caso, não foram criadas novas variáveis para as primeiras categorias de cada variável categóricas. Tais categorias foram representadas quando o valor de todas as outras variáveis era negativo (valor "0").

A categorização das variáveis quantitativas foi realizada com o auxílio do pacote "*discretization*[7]" (versão 1.0-1) através do método "*disc.Topdown*" que implementa, entre outros, o algoritmo *Ameva* (Gonzalez-Abril et al 2009). Esse algoritmo foi utilizado nesse trabalho para encontrar os melhores pontos de corte levando em consideração informações da variável alvo. Este algoritmo utiliza a métrica chi-quadrado para avaliar a relação entre as variáveis categorizadas e a variável alvo tentando minimizar o número de categorias geradas durante a categorização.

Tanto a padronização, quanto as conversões de tipos de dados foram feitas separadamente no conjunto de treinamento e no conjunto de testes. No caso da padronização, foram utilizados a média e o desvio padrão das variáveis do conjunto de treinamento tanto para fazer a padronização das variáveis desse conjunto como para fazer a padronização das variáveis no conjunto de teste. No caso da categorização, os pontos de corte utilizados para

[5] https://cran.r-project.org/web/packages/caret/index.html

[6] https://cran.r-project.org/web/packages/dummies/index.html

[7] https://cran.r-project.org/web/packages/discretization/index.html

categorizar as variáveis do conjunto de treinamento, foram os mesmos utilizados para categorizar as variáveis do conjunto de teste.

No caso da dicotomização quando uma determinada variável possuía mais categorias no conjunto de teste que no conjunto de treinamento, foi preciso eliminar as variáveis adicionais criadas pelo algoritmo de dicotomização para que ao final da dicotomização os dois conjuntos (treinamento e teste) tenham criado as mesmas variáveis. Quando possuía menos, foi preciso criar novas variáveis (com valor zerado), dessa forma durante a dicotomização do conjunto de teste foi preciso ter essas informações do conjunto de treinamento.

Além disso, ao realizar as conversões de tipos de dados, a variável representando o Índice de Massa Corporal (IMC) do tipo que está sendo transformado é eliminada da análise. Isso acontece pois existem duas variáveis entre as selecionadas, uma quantitativa e outra categórica. A categórica foi criada manualmente pela equipe ELSA-Brasil utilizando conhecimento de domínio, então se decidiu que ao realizar transformações, caso a variável "transformada" já faça parte do conjunto, não seria necessário transformá-la mais uma vez.

5.3 Particionamento dos dados

A divisão aleatória e estratificada do conjunto de dados foi realizada através do método "*createDataPartition*" do pacote "*caret*" com o parâmetro "p = 0,7" (percentual de dados que vai para o treinamento), ou seja, 70% foram utilizados para treino/validação e 30% dos dados foram guardados para serem utilizados no teste de generalização.

Após, o conjunto de treinamento/validação foi particionado de duas formas, uma para cada tipo de validação cruzada de 10 *folds* (repetida por 3 vezes e repetida por 10 vezes). Assim foram criadas 30 partes de dados (3 repetições x 10 *folds*) para a validação cruzada repetida por 3 vezes e 100 partes (10 repetições de 10 *folds*) para a validação repetida por 10 vezes. As partes de dados de cada repetição foram geradas aleatoriamente através do método "*createFolds*" (pacote "*caret*") que foi repetido gerando partes diferentes para cada repetição.

Além disso, para cada uma dessas partes, foram geradas mais duas partes contendo as mesmas observações que a parte original, uma com as variáveis dicotomizadas e outra com as variáveis categorizadas. Por fim, todas as partes geradas foram armazenadas em disco para serem utilizadas durante a criação dos modelos evitando que todo o processo de particionamento e pré-processamento dos dados fosse repetido.

5.4 Construção e Predição dos Modelos

Os detalhes de implementação da construção dos modelos e da utilização desses modelos nos dados de teste são independentes para cada algoritmo e é descrita a seguir.

5.4.1 Regressão Logística

A construção dos modelos de Regressão Logística foi feita através do método "*glm*" (de *Generalized Linear Models*) do pacote "*stats*". *Esse pacote é* um dos pacotes básicos do R. Essa função pode ser utilizada para gerar diferentes tipos de modelos, a regressão logística é produzida definindo o parâmetro "*family*" para "*binomial(link="logit")*". Esse parâmetro permite estabelecer uma família de distribuição de erro (*binomial*) e a função de ligação para ser utilizada no modelo (*logit*). Através do parâmetro "*control*" é possível definir os parâmetros "*epsilon*" e "*maxit*". Os parâmetros representam, respectivamente, a tolerância de convergência e o número máximo de iterações da função de minimização de custos (IWLS - *Iterative Weighted Least Squares*), e são utilizados basicamente como critérios de convergência do algoritmo. Somente esses dois parâmetros foram variados para a regressão logística. Os valores padrões deles são 1e-8 e 25, respectivamente.

Para realizar a predição, foi utilizado o método "*predict.glm*" passando como parâmetro "*type = 'response'*" indicando que a resposta desejada é uma estimativa de probabilidade.

5.4.2 Redes Neurais Artificiais

Para a construção dos modelos de redes neurais, foi utilizado o pacote "*nnet*[8]" (versão 7.3-11) com o método de mesmo nome para a criação dos modelos. Essa implementação permite a criação de redes neurais artificiais MLP com uma camada oculta somente, portanto, para os experimentos, não foram variadas a quantidade de camadas ocultas. Os seguintes parâmetros foram testados. O parâmetro "*size*" define a quantidade de neurônios na camada oculta. Lembrando que a quantidade de neurônios na camada de entrada e de saída depende somente da quantidade de variáveis preditoras e da quantidade de classes da variável-alvo,

[8] https://cran.r-project.org/web/packages/nnet/index.html

respectivamente. O parâmetro "*skip*" define se serão criadas ou não ligações do tipo *skip* na rede criada. A ligação *skip* é uma ligação direta dos neurônios de entrada para os neurônios de saída (sem ligação com a camada oculta). Dessa forma é obrigatório o uso desse tipo de ligação quando não existe neurônio na camada oculta. Por fim, o parâmetro *"decay"* define o termo *weight decay* que controla a importância relativa entre os termos da função de custo do algoritmo de retropropagação, utilizado para treinamento da rede. Não existe valor padrão para o campo size (pois é um dos principais parâmetros para variar e o analista deve estar a par disso). Os valores padrões para os campos *decay* e *skip* são "0" e "FALSO", ou seja, sem *weight decay* e sem camada *skip*. Nos testes foram definidos os valores de *MaxNWts* (número máximo de pesos permitido) e *maxit* (número máximo de iterações) para 70.000 e 500, respectivamente (Os valores padrões para esses parâmetros são 1.000 e 100). A função de ativação dos neurônios utilizada foi a função logística, definida por padrão. Essa implementação de redes neurais artificiais utiliza o método BFGS (*Broyden–Fletcher–Goldfarb–Shanno)* como otimização para minimização de custo. Para a predição, foi utilizado o método "*predict.nnet*", do mesmo pacote, passando o parâmetro "*type = raw*" indicando que a resposta do modelo será um número real.

5.4.3 *Random Forest*

Para esse algoritmo foi utilizado o pacote "*randomForest*[9]" (versão 4.6-12) e o método de mesmo nome. Segundo a documentação do pacote, o método implementa o algoritmo *Random Forest* de Breiman baseado no código original em Fortran de Breiman e Cutler. Foram utilizados os parâmetros padrões para a criação do modelo (com exceção do número de árvores que variou na etapa de afinação de parâmetros conforme descrito na Seção 4.4.1). Foi variado somente o número de árvores da floresta (parâmetro *ntree*, com valor padrão de "500").

A predição foi realizada através do método "*predict.randomForest*" do mesmo pacote utilizando como argumento "*type='prob'*" para indicar que a resposta deve ser dada como uma matriz de probabilidades.

---

[9] https://cran.r-project.org/web/packages/randomForest/index.html

5.4.4 *K-Nearest Neighbors*

Para a predição utilizando o algoritmo K-NN foi utilizado o método "*knn*" do pacote "*class*[10]" (versão 7.3-14). Devido à natureza desse algoritmo, não existe uma etapa de criação de modelo, o algoritmo classifica um caso baseado na distância dos casos contidos no conjunto de treinamento. Essa implementação utiliza a distância euclidiana como métrica para calcular o afastamento entre as instâncias. A classificação é realizada por voto majoritário. Os parâmetros variados foram *"k"*, que é o número de vizinhos candidatos (1 por padrão) e *"l", q*ue é o número de votos mínimos necessários para tomar uma decisão (por padrão é 0).

Foi utilizado o parâmetro "*prob = TRUE*" indicando que a resposta da predição deve conter a proporção dos votos da classe vencedora. Dessa forma, foi necessário processar a saída do método para obter a proporção da classe positiva, independente se for vencedora ou não. Ou seja, quando a classe vencedora foi a negativa, a resposta foi 1 menos a probabilidade produzida como saída pelo método.

Outro detalhe de implementação desse algoritmo foi a necessidade de utilizar captura de exceção (*try-catch*) devido ao fato de a implementação não funcionar para diversas combinações de parâmetros testadas, principalmente quando os dados foram categorizados antes de serem apresentados ao algoritmo. Como os testes com cada combinação de parâmetro aconteciam dentro de um laço, ao lançar exceção sem capturar, o laço era interrompido e os testes do algoritmo eram perdidos.

No início das análises também foram realizados testes desse algoritmo utilizando somente a classe vencedora (sem contar a proporção dos votos), porém os resultados foram muito insatisfatórios (menos de 60% de AUC) e essa abordagem foi desconsiderada para o restante das análises.

5.4.5 Naïve Bayes

Para a criação dos modelos classificadores bayesianos foi utilizado o método "*naiveBayes*" do pacote "e1071[11]" (versão 1.6-7). A predição do modelo foi realizada através

[10] https://cran.r-project.org/web/packages/class/index.html

[11] https://cran.r-project.org/web/packages/e1071/index.html

do método "*predict.naiveBayes*" do mesmo pacote. Essa função computa as probabilidades condicionais posteriores de uma variável-alvo dado um conjunto de variáveis preditoras independentes usando da regra de Bayes. A versão utilizada permite variar o parâmetro *laplace* utilizado para realizar uma regularização através de suavização de Laplace. Por padrão esse valor é 0.

Foi utilizado o parâmetro "*type = 'raw'*" para obter como saída do modelo a probabilidade de pertencer a cada classe, a qual foi feito um pós-processamento para extrair somente a probabilidade da classe positiva para poder calcular a métrica de AUC e fazer a predição comparando com um ponto de corte.

Também foi utilizado o parâmetro "*type = 'class'*" no método de predição para obter como resposta do modelo a classe que obteve a maior probabilidade *a posteriore*. Os resultados obtidos por essa abordagem foram encarados como resultados de um algoritmo diferente dos resultados obtidos pela predição com saída probabilística.

5.4.6 *Quinlan's C.5*

Para gerar esse tipo de modelo foi utilizado o método "C5.0" do pacote "C50[12]" (versão 0.1.0-24). Tal algoritmo produz modelos de árvores de classificação ou modelos baseados em regras baseado no algoritmo C5 de Quilan. A implementação permite rodar o algoritmo múltiplas vezes, como um método ensemble (*boosting)*, através do parâmetro "*trials*", contudo tal opção não foi utilizada nos experimentos. O parâmetro "*rules*" define se o modelo criado será uma árvore de classificação ou um conjunto de regras. Nos experimentos, os dois tipos de modelos foram comparados distintamente (como se fossem algoritmos diferentes).

O parâmetro "*control*" permite definir parâmetros para controlar o treinamento do modelo, conforme descrito a seguir: o atributo *winnow* (por padrão é falso) especifica se deve ser realizado uma pré-seleção de variáveis antes da criação do modelo; o atributo *noGlobalPruning* (por padrão é falso) define se não deve ser realizada uma poda global ao final do treinamento para simplificar o modelo; o atributo *minCases* (valor padrão é 2) define o menor número de casos que devem ser colocados em pelo menos duas das partições; o

[12] https://cran.r-project.org/web/packages/C50/index.html

atributo *fuzzyThreshold* (por padrão é falso) especifica se devem ser buscadas possíveis partições avançadas dos dados; por fim, o parâmetro *earlyStopping* (verdadeiro por padrão) define se o método interno para parada do *boosting* deve ser utilizado.

Essa implementação permite ainda definir uma matriz de custo para ser utilizada no treinamento, porém tal solução é válida somente quando a saída do classificador for categórica. Dentre os valores variados na afinação dos parâmetros, conforme Tabela 4.3, está o custo de classificação errada de caso positivo, esse valor é utilizado para criar a matriz de custo, o custo de classificar um caso negativo como positivo é fixo em 1 (custo usual) e os outros valores da matriz de custo são desconsiderados pelo método.

A predição do modelo foi realizada utilizando o método "predict.C5.0" do mesmo pacote. O parâmetro "*type*" define o tipo de resultado ("*class*" indica saída binária e "*prob*" saída probabilística). Inicialmente foram feitos testes utilizando saída probabilística, porém os resultados foram muito insatisfatórios. Nesse caso também não é possível utilizar matriz de custos. Assim as análises apresentadas para esse algoritmo utilizam exclusivamente o parâmetro "*type='class'*".

5.4.7 *RIPPER*

Para a construção dos modelos segundo o algoritmo *RIPPER* foi utilizado o método "*JRip*" do pacote "*RWeka*[13]" (versão 0.4-24, com RWekajars versão 3.7.12-1 e rJava versão 0.9-7). Esse método implementa o algoritmo de indução de regras proposicionais *"Repeated Incremental Pruning to Produce Error Reduction"* (*RIPPER*) como proposto por Cohen. Essa implementação permite, através do parâmetro "*control*" definir algumas configurações para a aprendizagem do modelo. Assim, foram variados os seguintes parâmetros: número de *folds* (parâmetro -F, valor padrão é "3"); pesos mínimos das instâncias dentro de uma partição (parâmetro -N, valor padrão é "2.0"); número de execuções de otimizações (parâmetro -O, por padrão é "2"); se é para conferir uma taxa de erros maior que 0,5 como critério de parada (parâmetro -E, valor padrão é verdadeiro); e, por fim, um valor lógico definindo se é para realizar a etapa de poda (parâmetro -P, valor padrão é verdadeiro).

[13] https://cran.r-project.org/web/packages/RWeka/index.html

Os resultados iniciais para esse método, tanto para saída binária como probabilística foram muito insatisfatórios e esse algoritmo foi removido da análise.

5.5 Cálculo das Métricas de Avaliação

A construção da curva ROC e o cálculo de AUC das avaliações dos modelos com saída probabilística foram realizadas utilizando os métodos, respectivamente, "*roc*" e "*auc*" do pacote "*pRoc*[14]" (versão 1.8) passando o resultado previsto pelo modelo junto com os rótulos conhecidos com o resultado esperado. Após isso, foram obtidos os resultados de classificação utilizando pontos de corte definidos pelo analista.

Com os resultados da classificação, sejam obtidos pelo resultado dos modelos com saída probabilística após comparar com um ponto de corte ou obtidos diretamente como saída do modelo, as demais métricas (Acurácia Balanceada, Sensibilidade e Especificidade) foram extraídas através do método "*confusionMatrix*" do pacote "*caret*". Esse método tem como parâmetro os resultados binários previstos pelo modelo junto com os rótulos conhecidos de cada caso e gera uma matriz de confusão disponibilizando as diversas métricas de classificação a partir daí.

Para cada validação cruzada, onde são criados diversos modelos (um para cada *fold*/repetição), os resultados da avaliação de cada modelo foram armazenados em uma lista e ao final da validação, foi extraída a média e desvio padrão do desempenho do esquema de aprendizagem que se estava avaliando. Os resultados desses testes foram armazenados em disco em arquivos CSV para serem investigados pelo analista.

5.6 Seleção Automática de Variáveis

Para a geração dos subconjuntos de variáveis da etapa de seleção automática de variáveis, foi utilizado o método "*forward.search*" do pacote "*FSelector*[15]" (versão 0.20). Esse método implementa uma busca gulosa. Inicialmente, o algoritmo expande o nodo inicial, avalia os filhos e escolhe o melhor nodo que se torna o nodo inicial, iterativamente até não ter mais melhora. Cada nodo é um subconjunto de variáveis e o nodo inicial é vazio, sendo que

[14] https://cran.r-project.org/web/packages/pROC/index.html

[15] https://cran.r-project.org/web/packages/FSelector/index.html

os filhos são todos subconjuntos contendo uma variável. Os demais nodos são subconjuntos contendo a variável do nodo pai e mais uma nova variável.

O método utilizado recebe como parâmetro o conjunto com todas as variáveis e uma função de avaliação que recebe como parâmetro um subconjunto de variáveis e retorna um valor numérico representando o quão bom é o conjunto passado como parâmetro. Para cada algoritmo utilizado nessa etapa foi gerada uma função de avaliação distinta que carregava os dados da memória, criava o modelo com o subconjunto passado como parâmetro, avaliava o modelo através de validação cruzada e retornava a métrica AUC. O pacote propicia o método auxiliar "*as.simple.formula*" que cria um objeto do tipo formula, que é utilizado nos métodos de geração de modelo, a partir de uma lista de *strings* com os nomes das variáveis contidas no subconjunto. Esse método também foi utilizado para criar os modelos utilizando os subconjuntos de variáveis selecionados.

5.7 Ferramenta WEB

A construção do protótipo de ferramenta WEB foi realizada por um aluno de graduação do Instituto de Informática, bolsista, com minha orientação como parte de um projeto relacionado a este mestrado. O desenvolvimento da interface WEB foi realizado utilizando a linguagem de programação Python.

Como o conjunto de variáveis utilizado para a criação da ferramenta continha variáveis derivadas, foi necessário utilizar as variáveis básicas (perguntas de questionários ou aferições de medidas, por exemplo) que formavam essas variáveis derivadas. Para tanto foi requisitado à equipe de pesquisa do ELSA-Brasil documentos com os questionários e instruções de como obter as respostas necessárias para preencher essas variáveis básicas. Assim a criação das variáveis derivadas foi feita pela ferramenta WEB baseado em um documento do estudo ELSA-Brasil que define como cada variável derivada foi criada. E por este motivo o protótipo criado apresenta uma quantidade maior de perguntas do que o número de variáveis utilizado para a construção do modelo.

Também foram feitas orientações no sentido de melhorar a disposição das perguntas para aumentar a facilidade de uso da ferramenta.

Como o modelo utilizado pela ferramenta não foi um modelo muito complexo, a própria ferramenta pôde calcular o resultado final sem precisar ter integração com a ferramenta R, aumentando a eficácia da ferramenta com relação à sua complexidade temporal e espacial.

6 RESULTADOS

Os resultados alcançados em cada etapa do processo descrito na Seção 4.4 são apresentados a seguir.

6.1 Resultados da Afinação dos Parâmetros

A primeira etapa avaliou o comportamento de diferentes valores de parâmetros dos algoritmos utilizando dicotomização, categorização e sem utilizar nenhuma transformação. Nessa etapa os algoritmos C5 (tanto a versão que gera árvores de decisão e regras, ambas com saída probabilística), JRIP (*RIPPER*, todas as versões) e a versão com saída binária do algoritmo K-NN foram descartados das análises por apresentarem desempenho muito insatisfatório. As causas desse desempenho não foram investigadas. Os parâmetros que obtiveram melhores resultados para os demais algoritmos e os desempenhos desses modelos encontram-se na Tabela 6.1.

Tabela 6.1 – Resultados da afinação de parâmetros, melhores por transformação

Algoritmo	Transformação	Param.	Configuração dos Parâmetros e ponto de corte	AUC (média)	AUC (d.p.)	AB (média)	AB (d.p.)
Redes Neurais	**-**	**1**	**size = 0; decay = 2; skip = 1; cutoff = 0,1**	**75,412%**	**1,946%**	**69,205%**	**2,037%**
Redes Neurais	Dicotomização	2	size = 45; decay = 11; skip = 1; cutoff = 0,11	75,353%	2,035%	69,147%	2,169%
Redes Neurais	Categorização	3	size = 70; decay = 3; skip = 1; cutoff = 0,1	74,194%	2,034%	67,921%	2,021%
Regressão Logística	**-**	**1**	**maxit = 5; epsilon = 0,01; cutoff = 0,1**	**75,273%**	**1,981%**	**68,904%**	**2,286%**
Regressão Logística	Dicotomização	1	maxit = 5; epsilon = 0,01; cutoff = 0,11	75,102%	2,033%	68,653%	2,149%
Regressão Logística	Categorização	1	maxit = 5; epsilon = 0,01; cutoff = 0,09	74,031%	2,087%	67,965%	1,985%
K-NN	**-**	**1**	**min.votes = 1; neighbor = 495; cutoff = 0,1**	**74,526%**	**2,066%**	**68,342%**	**2,010%**
K-NN	Dicotomização	2	min.votes = 0; neighbor = 470; cutoff = 0,09	73,971%	2,117%	68,012%	2,107%
K-NN	Categorização	3	min.votes = 0; neighbor = 400; cutoff = 0,09	73,933%	2,084%	67,855%	1,923%
Naïve Bayes	**Categorização**	**1**	**laplace = 1e-05; cutoff = 0,08**	**73,665%**	**2,125%**	**68,156%**	**2,354%**
Naïve Bayes	-	2	laplace = 0; cutoff = 0,18	72,991%	2,505%	68,026%	2,232%
Naïve Bayes	Dicotomização	1	laplace =1e-05; cutoff = 0,01	69,273%	4,856%	63,714%	4,745%
Random Forest	**-**	**1**	**ntree = 1010; cutoff = 0,13**	**73,227%**	**2,228%**	**67,796%**	**2,389%**
Random Forest	Dicotomização	2*	ntree = 2250; cutoff = 0,13	72,967%	2,356%	67,577%	1,776%
Random Forest	Categorização	3	ntree = 870; cutoff = 0,12	71,836%	2,033%	66,633%	2,298%
C5 (Class)	**Dicotomização**	**1**	**costs = 9; minCases = 240; fuzzyThreshold = 1; earlyStopping = 1; winnow = 0; noGlobalPruning = 0**	**-**	**-**	**67,100%**	**2,406%**
C5 (Class)	-	2	costs = 10; minCases = 230; fuzzyThreshold = 1; earlyStopping = 1; winnow = 0; noGlobalPruning = 0	-	-	66,680%	1,960%
C5 (Class)	Categorização	3*	costs = 8; minCases = 230; fuzzyThreshold = 1; earlyStopping = 1; winnow = 0; noGlobalPruning = 1	-	-	65,806%	2,345%
C5-Rules (Class)	**Dicotomização**	**1**	**costs = 11; minCases = 250; fuzzyThreshold = 0; earlyStopping = 1; winnow = 1; noGlobalPruning = 0**	**-**	**-**	**67,087%**	**1,797%**
C5-Rules (Class)	-	2*	costs = 11; minCases = 260; fuzzyThreshold = 0; earlyStopping = 1; winnow = 0; noGlobalPruning = 0	-	-	66,591%	1,837%
C5-Rules (Class)	Categorização	3	costs = 8; minCases = 64; fuzzyThreshold = 0; earlyStopping = 1; winnow = 1; noGlobalPruning = 0	-	-	66,088%	2,702%
Naïve Bayes (Class)	**-**	**1**	**laplace = 0,1**	**-**	**-**	**66,622%**	**2,680%**
Naïve Bayes (Class)	Categorização	2	laplace = 0,01	-	-	61,063%	2,524%
Naïve Bayes (Class)	Dicotomização	2	laplace = 0,01	-	-	59,117%	3,770%

Fonte: Autoria Própria.

Essa Tabela apresenta na primeira e na segunda coluna o nome do algoritmo e a transformação utilizada, respectivamente. A terceira e a quarta coluna identifica o índice e valores de cada parâmetro da melhor configuração obtida para aquele algoritmo/transformação. O asterisco na terceira coluna indica que aquela configuração de

parâmetros específica foi a que obteve melhores resultados na etapa de estimativa de erros, apresentada na Seção 6.3. Quando não há esse indicativo, significa que as melhores configurações de parâmetros coincidiram nas duas etapas, ou seja, a configuração que obteve o melhor resultado na afinação de parâmetros também obteve o melhor resultado na etapa de estimativas de erro. As últimas quatro colunas mostram os resultados da validação cruzada (média e desvio padrão de AUC e Acurácia Balanceada dos 30 modelos gerados pela validação cruzada de 10 folds repetida por 3 vezes) utilizados como critério para escolha dos melhores parâmetros.

A Tabela mostra o mesmo algoritmo de aprendizagem de máquina com diferentes tipos de transformações agrupados e ordenados de forma decrescente em termos do melhor AUC e AB obtido com cada algoritmo. O mesmo ordenamento foi utilizado dentro de cada grupo.

Apesar de o objetivo dessa etapa não ter sido comparar quais dos algoritmos produziram os melhores modelos, foi possível ter uma ideia inicial do poder preditivo de cada algoritmo. Nesse quesito, os melhores resultados foram produzidos utilizando Redes Neurais e Regressão Logística com 75,412% (linha 1) e 75,273% (linha 4) de AUC, respectivamente.

Também foi possível perceber o impacto entre os diferentes tipos de transformações utilizados no mesmo algoritmo de aprendizagem de máquina com relação ao desempenho dos modelos. Por exemplo, a Tabela mostra uma perda relevante de desempenho quando as variáveis são todas categorizadas nos modelos gerados por Redes Neurais, Regressão Logística, *Random Forest* e C5. O algoritmo K-NN apresentou um desempenho muito inferior utilizando qualquer tipo de transformação. Porém, em alguns casos, a transformação ajudou o algoritmo a melhorar o desempenho, como no caso do algoritmo C5 (versão que gera regras de decisão) utilizando dicotomização e *Naïve Bayes* (versão com saída probabilística) utilizando categorização.

De forma geral, o comportamento dos algoritmos e transformações utilizadas com relação ao desempenho dos modelos desenvolvidos permaneceu similar nas demais etapas.

Outro resultado que pôde ser observado em muitos casos é o impacto causado pela transformação utilizada na escolha da melhor configuração de parâmetros. Por exemplo, o melhor desempenho obtido com o algoritmo de Redes Neurais, alcançado sem usar transformação, foi obtido sem neurônios na camada oculta (size = 0, na linha 1, coluna 4). Quando dicotomização e categorização foram utilizadas para a criação dos modelos, os melhores modelos foram obtidos utilizando uma configuração diferente de parâmetros: 45

neurônios na camada oculta (size = 45, na linha 2, coluna 4) utilizando dicotomização e com 70 neurônios (size = 70, na linha 3, coluna 4), utilizando categorização.

A melhor configuração alcançada de cada algoritmo (referente aos parâmetros da primeira linha de cada algoritmo) foi utilizada para configurar os quatro algoritmos utilizados na etapa de seleção automática de variáveis. Após, todas as melhores configurações de parâmetros de cada algoritmo e de cada transformação (todas as configurações de parâmetros mostradas na Tabela) foram utilizadas na etapa de estimativa de erro.

6.2 Resultados da Seleção Automática de Variáveis

A etapa de seleção automática de variáveis gerou quatro subconjuntos distintos de variáveis como mostrado nas primeiras quatro linhas da Tabela 6.2: "*glm-fs"*, criado com Regressão Logística ("*fs"* no nome vem de "*Forward Selection*" e "*glm"* vem de "*Generalized Linear Model*", que é o nome do método utilizado para gerar os modelos de Regressão Logística); "*nnet-fs"*, criado com Redes Neurais; *knn-fs* criado com K-NN; e "*nb-fs",* criado com o algoritmo *Naïve Bayes* (versão com saída probabilística). Também foi criado o conjunto "*union*" contendo todas as variáveis, ou seja, a união de todos os subconjuntos anteriores, mostrado na linha 5 da Tabela. A Tabela também mostra o conjunto "*original"*, que contém todas as 34 variáveis candidatas.

Tabela 6.2 – Subconjuntos gerados na etapa de Seleção Automática de Variáveis

Subconjunto	# var	Diferenças do original somadas	Nomes das variáveis
glm-fs	12	1,819%	a_ativfisica a_bebexcessivo a_cint a_escolar a_gidade a_imc1 a_imc2 a_medanthipert a_rcq diea133 hfda11 rcta8
nnet-fs	15	1,730%	a_ativfisica a_bebexcessivo a_binge a_cint a_escolar a_gidade a_imc1 a_imc2 a_medanthipert a_medredlip a_rcq a_sfhfprem diea133 hfda11 rcta8
knn-fs	11	0,291%	a_bebexcessivo a_escolar a_gidade a_imc1 a_medanthipert a_rcq a_sfhfprem a_sfmiprem diea133 hfda11 rcta8
nb-fs	11	1,436%	a_ativfisica a_bebexcessivo a_binge a_escolar a_gidade a_imc2 a_medanthipert a_rcq afia7 diea133 hfda11
union	17	1,537%	a_ativfisica a_bebexcessivo a_binge a_cint a_escolar a_gidade a_imc1 a_imc2 a_medanthipert a_medredlip a_rcq a_sfhfprem a_sfmiprem afia7 diea133 hfda11 rcta8
original	34	-	a_ativfisica a_bebexcessivo a_binge a_chdhard a_chdlight a_cint a_cons_est_nacl a_consdiafrutas a_consdiaverduras a_escolar a_fumante a_gidade a_imc1 a_imc2 a_medanthipert a_medoutahip a_medredlip a_prvdcc a_rcq a_rendapercapita a_sfhfprem a_sfmiprem a_sfrevprem a_sfstkprem a_sintsono a_sitconj a_volalc afia7 claa2 diea133 hfda07 hfda11 hmpa08 rcta8

Fonte: Autoria Própria.

A primeira coluna da Tabela mostra o nome identificador do subconjunto, a segunda apresenta a quantidade de variáveis presentes em cada subconjunto e a quarta lista as variáveis contidas no subconjunto.

O valor da terceira coluna ("Diferenças do original somadas") foi obtido da seguinte forma:

1. Primeiramente, cada um dos subconjuntos gerados (inclusive o conjunto original) foi avaliado utilizando validação cruzada repetida por 3 vezes (assim

como na geração dos subconjuntos e na afinação de parâmetros) em cada um dos quatro algoritmos utilizados nessa etapa.

2. Para cada algoritmo, a média de AUC obtida para cada subconjunto de variáveis foi subtraída da média de AUC obtida utilizando o subconjunto de variáveis original.
3. As diferenças obtidas para cada subconjunto em todos os quatro algoritmos foram somadas.

Esse valor dá uma ligeira ideia do quão melhor é o desempenho, em termos de AUC, de um determinado subconjunto de variáveis com relação ao subconjunto original entre todos os algoritmos. Assim pode-se observar que os subconjuntos *glm-fs* e *nnet-fs* obtiveram um bom desempenho nessa etapa comparado aos demais, enquanto o *knn-fs* não apresentou resultados muito bons.

Devido à natureza do método de seleção de variáveis automática com *Wrapper*, a tendência é que o algoritmo tenha um melhor desempenho com o subconjunto de variáveis gerado com o próprio algoritmo, contudo, nos testes posteriores, cada algoritmo foi testado em todos os subconjuntos de variáveis.

6.3 Resultados da Estimativa de Erros

O objetivo desta etapa foi obter estimativas mais confiáveis das avaliações dos algoritmos. Para isso, os testes nessa etapa foram realizados utilizando 10 repetições na validação cruzada ao invés de 3 repetições como nas etapas anteriores gerando 100 modelos ao invés de 30 em cada teste.

Os algoritmos de aprendizagem de máquina foram testados, com todos os parâmetros encontrados na primeira etapa e todos os subconjuntos de variáveis gerados na segunda etapa, incluindo o subconjunto de variáveis original. Também foram criados modelos utilizando diferentes tipos de transformações de dados. Os principais resultados dessa etapa são apresentados a seguir.

Primeiramente, para cada algoritmo, foram selecionados os resultados obtidos com a combinação de parâmetros e o ponto de corte de classificação que obtiveram o melhor resultado, independentemente do tipo de transformação ou subconjunto de variáveis utilizado. Os resultados obtidos com outras combinações de parâmetros/pontos de corte foram desconsiderados desta análise.

A Tabela 6.3 apresenta os melhores resultados para cada algoritmo de aprendizagem de máquina e tipo de transformação utilizada independentemente do subconjunto de variáveis utilizado. Respectivamente, as colunas representam: o nome do algoritmo de aprendizagem usado; tipo de transformação de dados; subconjunto de variáveis; média e desvio padrão da AUC alcançada na validação cruzada; e, média e desvio padrão da AB alcançada na validação cruzada. As últimas três colunas mostram as diferenças em termos de AUC ou AB (nos casos dos algoritmos que geram modelos com saída binária) dos resultados obtidos com outros tipos de transformação (sem transformação, dicotomização e categorização, respectivamente) para o mesmo algoritmo de aprendizagem de máquina.

Tabela 6.3 – Resultados obtidos na etapa de estimativa de erros, melhores por transformação

Algoritmo	Transformação	Subconj. Var.	AUC (média)	AUC (d.p.)	AB (média)	AB (d.p.)	Dif -	Dif Bin	Dif Disc
Redes Neurais	**-**	**nnet-fs**	**75,855%**	**2,131%**	**69,716%**	**2,353%**	-	0,142%	**1,227%**
Redes Neurais	Dicotomização	nnet-fs	75,713%	2,128%	69,320%	2,214%	-	-	**1,086%**
Redes Neurais	Categorização	glm-fs	74,627%	2,202%	68,271%	2,108%	-	-	-
Regressão Logística	**-**	**glm-fs**	**75,854%**	**2,137%**	**69,809%**	**2,232%**	-	0,170%	**1,190%**
Regressão Logística	Dicotomização	glm-fs	75,683%	2,175%	69,567%	2,215%	-	-	**1,020%**
Regressão Logística	Categorização	nb-fs	74,663%	2,134%	68,211%	2,166%	-	-	-
K-NN	**-**	**knn-fs**	**75,623%**	**2,241%**	**69,031%**	**2,210%**	-	0,650%	
K-NN	Dicotomização	knn-fs	74,974%	2,191%	69,006%	2,282%		-	
Random Forest	**-**	**original**	**73,304%**	**2,408%**	**67,928%**	**2,422%**	-	0,259%	**1,497%**
Random Forest	Dicotomização	original	73,045%	2,540%	67,731%	2,399%	-	-	**1,238%**
Random Forest	Categorização	original	71,807%	2,506%	66,449%	2,712%	-	-	-
Naïve Bayes	**-**	**knn-fs**	**75,112%**	**2,299%**	**68,745%**	**2,094%**	-	**2,219%**	0,504%
Naïve Bayes	Categorização	nb-fs	74,608%	2,191%	68,132%	2,044%	-	**1,716%**	-
Naïve Bayes	Dicotomização	glm-fs	72,893%	2,620%	67,097%	2,780%	-	-	-
C5 (class)	**Dicotomização**	**knn-fs**	**-**	**-**	**67,437%**	**2,381%**	0,187%	-	0,867%
C5 (class)	-	nb-fs	-	-	67,251%	2,076%	-	-	0,681%
C5 (class)	Categorização	glm-fs	-	-	66,570%	2,362%	-	-	-
C5-Rules (class)	**-**	**nb-fs**	**-**	**-**	**67,008%**	**2,230%**	-	0,029%	0,941%
C5-Rules (class)	Dicotomização	knn-fs	-	-	66,979%	2,242%	-	-	0,912%
C5-Rules (class)	Categorização	nb-fs	-	-	66,068%	2,310%	-		-
Naïve Bayes (class)	**-**	**original**	**-**	**-**	**66,492%**	**2,665%**	-	**3,115%**	**5,413%**
Naïve Bayes (class)	Dicotomização	glm-fs	-	-	63,377%	2,972%	-	-	**2,298%**
Naïve Bayes (class)	Categorização	original	-	-	61,080%	2,530%	-	-	-

Fonte: Autoria Própria.

Assim como nos resultados da primeira etapa (Afinação de Parâmetros), os algoritmos de Redes Neurais e Regressão Logística obtiveram os melhores resultados. Os dois, sem utilizar transformação como pré-processamento, produziram modelos com 75,855% (linha 1) e 75,854% (linha 4) de AUC, respectivamente. Cada um utilizando o subconjunto de variáveis gerado pelo próprio algoritmo na segunda etapa.

Esta Tabela mostra principalmente o impacto da transformação de dados no desempenho dos modelos criados. Com relação a isso, os resultados também são parecidos com os resultados obtidos na etapa de afinação de parâmetros. Isso pode ser visto, por exemplo, nas linhas 1 e 3, onde o algoritmo de redes neurais alcançou uma média de 75,855

em AUC sem transformação de dados e caiu para 74,627% (perda de 1,227%) quando foi utilizada categorização. Este foi o melhor resultado obtido com categorização para este algoritmo, o resultado utilizando o mesmo subconjunto de variáveis (*nnet-fs)* e categorização pode ter sido pior. Ou ainda o algoritmo *Naïve Bayes* sem transformação alcançou 75,112% em AUC enquanto o mesmo algoritmo teve uma queda de 2,219%, alcançando 72,893% utilizando dicotomização.

Seguindo a mesma intuição da Tabela 6.3, a Tabela 6.4 mostra os melhores resultados obtidos de cada algoritmo para cada subconjunto de variáveis utilizado, independentemente da transformação. As duas colunas adicionais mostram a diferença do resultado da linha para o resultado obtido com o conjunto de variáveis original para o mesmo algoritmo. Valores negativos nessas colunas nessas colunas significam que o resultado foi pior que o resultado obtido com o conjunto original de variáveis.

Tabela 6.4 – Resultados obtidos na etapa de estimativa de erros por subconjunto

Algoritmo	Transformação	Subconj. Var.	AUC (média)	AUC (d.p.)	AB (média)	AB (d.p.)	Dif AUC orig	Dif AB orig
Redes Neurais	**-**	**nnet-fs**	**75,855%**	**2,131%**	**69,716%**	**2,353%**	**0,437%**	**0,545%**
Redes Neurais	-	glm-fs	75,835%	2,160%	69,868%	2,302%	0,417%	0,697%
Redes Neurais	-	union	75,816%	2,129%	69,661%	2,305%	0,397%	0,490%
Redes Neurais	Dicotomização	knn-fs	75,640%	2,128%	69,386%	2,180%	0,222%	0,215%
Redes Neurais	-	nb-fs	75,511%	2,101%	69,396%	2,364%	0,093%	0,225%
Redes Neurais	-	original	75,418%	2,110%	69,171%	2,219%	0,000%	0,000%
Regressão Logística	**-**	**glm-fs**	**75,854%**	**2,137%**	**69,809%**	**2,232%**	**0,589%**	**1,089%**
Regressão Logística	-	nnet-fs	75,816%	2,113%	69,520%	2,302%	0,551%	0,800%
Regressão Logística	-	union	75,764%	2,120%	69,366%	2,223%	0,499%	0,645%
Regressão Logística	-	knn-fs	75,608%	2,137%	69,280%	2,203%	0,343%	0,560%
Regressão Logística	-	nb-fs	75,481%	2,066%	69,132%	2,150%	0,216%	0,412%
Regressão Logística	-	original	75,265%	2,098%	68,720%	2,225%	0,000%	0,000%
K-NN	**-**	**knn-fs**	**75,623%**	**2,241%**	**69,031%**	**2,210%**	**1,072%**	**0,773%**
K-NN	-	glm-fs	74,824%	2,232%	68,712%	2,168%	0,273%	0,453%
K-NN	-	nnet-fs	74,793%	2,246%	68,875%	2,035%	0,242%	0,617%
K-NN	-	nb-fs	74,791%	2,169%	68,718%	2,188%	0,240%	0,460%
K-NN	-	union	74,671%	2,249%	68,694%	2,083%	0,120%	0,436%
K-NN	-	original	74,551%	2,330%	68,258%	2,115%	0,000%	0,000%
Naïve Bayes	**-**	**knn-fs**	**75,112%**	**2,299%**	**68,745%**	**2,094%**	**1,409%**	**0,593%**
Naïve Bayes	Categorização	nb-fs	74,608%	2,191%	68,132%	2,044%	0,905%	-0,020%
Naïve Bayes	-	glm-fs	74,228%	2,334%	68,131%	2,248%	0,525%	-0,020%
Naïve Bayes	-	nnet-fs	74,185%	2,343%	68,247%	2,244%	0,483%	0,095%
Naïve Bayes	Categorização	union	74,180%	2,292%	68,186%	2,159%	0,477%	0,034%
Naïve Bayes	Categorização	original	73,703%	2,376%	68,151%	2,386%	0,000%	0,000%
Random Forest	**-**	**original**	**73,304%**	**2,408%**	**67,928%**	**2,422%**	**0,000%**	**0,000%**
Random Forest	-	knn-fs	72,615%	2,500%	63,624%	2,554%	-0,689%	-0,689%
Random Forest	-	nnet-fs	72,360%	2,511%	66,359%	2,614%	-0,944%	-0,944%
Random Forest	Dicotomização	union	71,978%	2,489%	65,273%	2,729%	-1,326%	-1,326%
Random Forest	Dicotomização	glm-fs	71,967%	2,545%	65,269%	2,560%	-1,337%	-1,337%
Random Forest	-	nb-fs	71,223%	2,561%	61,554%	2,485%	-2,081%	-2,081%
C5 (class)	**Dicotomização**	**knn-fs**	**-**	**-**	**67,437%**	**2,381%**	**-**	**0,627%**
C5 (class)	Dicotomização	glm-fs	-	-	67,313%	2,433%	-	0,503%
C5 (class)	-	nb-fs	-	-	67,251%	2,076%	-	0,441%
C5 (class)	Dicotomização	nnet-fs	-	-	67,154%	2,405%	-	0,344%
C5 (class)	Dicotomização	union	-	-	67,154%	2,405%	-	0,344%
C5 (class)	Dicotomização	original	-	-	66,810%	2,561%	-	0,000%
C5-Rules (class)	**-**	**nb-fs**	**-**	**-**	**67,008%**	**2,230%**	**-**	**0,433%**
C5-Rules (class)	Dicotomização	knn-fs	-	-	66,979%	2,242%	-	0,404%
C5-Rules (class)	Dicotomização	glm-fs	-	-	66,809%	2,168%	-	0,234%
C5-Rules (class)	Dicotomização	nnet-fs	-	-	66,772%	2,232%	-	0,196%
C5-Rules (class)	Dicotomização	union	-	-	66,772%	2,232%	-	0,196%
C5-Rules (class)	Dicotomização	original	-	-	66,575%	2,386%	-	0,000%
Naïve Bayes (class)	**-**	**original**	**-**	**-**	**66,492%**	**2,665%**	**-**	**0,000%**
Naïve Bayes (class)	Dicotomização	glm-fs	-	-	63,377%	2,972%	-	-3,115%
Naïve Bayes (class)	-	union	-	-	63,276%	2,423%	-	-3,216%
Naïve Bayes (class)	-	nnet-fs	-	-	62,722%	2,406%	-	-3,770%
Naïve Bayes (class)	Dicotomização	nb-fs	-	-	60,978%	2,780%	-	-5,514%
Naïve Bayes (class)	Dicotomização	knn-fs	-	-	60,543%	3,430%	-	-5,950%

Fonte: Autoria Própria.

A Tabela mostra o impacto de utilizar um determinado subconjunto de variáveis comparado com os resultados obtidos pelos modelos gerados com o conjunto de variáveis original. As diferenças dos melhores resultados de cada algoritmo para os resultados obtidos com o conjunto original não foram relevantes (ficaram em aproximadamente 0,5%) para os algoritmos de Redes Neurais, Regressão Logística, *Random Forest*, C5 e *Naïve Bayes* (com saída binária). Dentre esses algoritmos, *Random Forest* e *Naïve Bayes* alcançaram os melhores resultados quando utilizaram o conjunto de variáveis original.

Por outro lado, os algoritmos K-NN e *Naïve Bayes* (com saída probatilística) alcançaram uma melhora relevante (1,072% e 1,409%, respectivamente) utilizando o melhor subconjunto de variáveis, gerado na seleção automática de variáveis, comparado com o resultado utilizando o conjunto de variáveis original.

Por fim, a Tabela 6.5 mostra o melhor resultado de cada algoritmo em termos de AUC, Acurácia Balanceada, Sensibilidade e Especificidade. Para cada métrica, é apresentada a média das 100 avaliações realizadas na validação cruzada e, ao lado, o intervalo de confiança de 95% (z = 1,96). O intervalo de confiança foi calculado através da fórmula $Z * \frac{d.p.}{\sqrt{n}}$, onde z = 1,96 (para uma confiança de 95%), d.p. é o desvio padrão da amostra e n = 100 é o número de amostras.

Tabela 6.5 – Melhores resultados por algoritmo com intervalos de confiança de 95%

Algoritmo	AUC		Acurácia Balanceada		Sensibilidade		Especificidade	
	média	I.C.	média	I.C.	média	I.C.	média	I.C.
Redes Neurais	75,855%	(75,437% - 76,272%)	69,716%	(69,255% - 70,178%)	70,815%	(69,914% - 71,717%)	68,617%	(68,307% - 68,928%)
Regressão Logística	75,854%	(75,435% - 76,273%)	69,809%	(69,372% - 70,246%)	71,330%	(70,460% - 72,199%)	68,288%	(67,953% - 68,624%)
K-NN	75,623%	(75,184% - 76,062%)	69,031%	(68,598% - 69,464%)	71,442%	(70,525% - 72,359%)	66,620%	(66,246% - 66,994%)
Naïve Bayes	75,112%	(74,661% - 75,562%)	68,745%	(68,334% - 69,155%)	76,140%	(75,311% - 76,968%)	61,349%	(61,015% - 61,683%)
Random Forest	73,304%	(72,832% - 73,776%)	67,928%	(67,454% - 68,403%)	66,291%	(65,293% - 67,289%)	69,566%	(69,245% - 69,886%)
C5 (class)	-	-	67,437%	(66,971% - 67,904%)	69,970%	(68,925% - 71,014%)	64,905%	(64,328% - 65,482%)
C5-Rules (class)	-	-	67,008%	(66,571% - 67,445%)	75,158%	(74,202% - 76,114%)	58,859%	(58,131% - 59,586%)
Naïve Bayes (class)	-	-	66,492%	(65,970% - 67,015%)	54,269%	(53,016% - 55,521%)	78,716%	(78,186% - 79,247%)

Fonte: Autoria Própria.

Segundo a Tabela, os algoritmos de redes neurais, regressão logística, K-NN e *Naïve Bayes* estariam estatisticamente empatados em termos de AUC. Por outro lado, para a métrica de Acurácia Balanceada, existe uma diferença estatisticamente significante (para pior) entre o algoritmo *Naïve Bayes* (linha 4) e os algoritmos de Redes Neurais e Regressão Logística. Para os cinco primeiros algoritmos, que possuem saída probabilística, foi selecionado o ponto que maximizaria a acurácia balanceada, sem dar preferência para a sensibilidade ou especificidade dos modelos gerados. Essa escolha ajudou a manter um equilíbrio entre essas duas métricas na maioria dos casos.

A partir do intervalo de confiança, utilizando o mesmo método, para os resultados em AUC obtidos com o conjunto de variáveis original para os algoritmos de Redes Neurais (75.005% - 75.832%), Regressão Logística (74.854% - 75.676%), K-NN (74.095% - 75.008%) e Naïve Bayes (73.237% - 74.168%), pode-se observar que as diferenças entre os resultados obtidos utilizando o melhor subconjunto de variáveis comparado com o resultado obtido utilizando o conjunto de variáveis original não é estatisticamente relevante para os algoritmos de Redes Neurais e Regressão Logística, contudo tal diferença torna-se relevante para os algoritmos de K-NN e Naïve Bayes. Mesmo assim, a utilização de um subconjunto menor de variáveis torna o modelo menos complexo, mais simples de aplicar e mais computacionalmente eficiente para treinar ou fazer as previsões.

6.4 Resultados do Teste de Generalização

No teste de generalização, todo o conjunto de dados de treinamento/validação foi utilizado para criar um modelo para cada algoritmo, utilizando esquema de aprendizagem que ofereceu melhor resultado na etapa anterior e o desempenho desses modelos foram testados em um conjunto de testes separado no início dos testes e não utilizado até então.

A Tabela 6.6 mostra os melhores resultados para cada algoritmo obtidos na etapa anterior ao lado dos resultados obtidos no teste de generalização. As duas últimas colunas apresentam a diferença entre os resultados das duas etapas em termos de AUC e AB. A Tabela está ordenada baseada nos resultados obtidos na etapa anterior, de estimativa de erros.

Tabela 6.6 – Resultado do teste de generalização comparado com validação cruzada

Algoritmo	Resultado Estimativa Erro		Resultado teste de generalização				VC - Gen	
	AUC (média)	AB (média)	AUC	AB	Sensibilidade	Especificidade	AUC	AB
Redes Neurais	75,855%	69,716%	74,523%	68,818%	70,223%	67,413%	1,332%	0,898%
Regressão Logística	75,854%	69,809%	74,556%	69,287%	71,464%	67,110%	1,297%	0,522%
K-NN	75,623%	69,031%	73,565%	66,845%	67,246%	66,445%	2,058%	2,186%
Naïve Bayes	75,112%	68,745%	73,759%	68,761%	76,675%	60,847%	1,352%	-0,017%
Random Forest	73,304%	67,928%	71,873%	65,396%	63,772%	67,020%	1,432%	2,533%
C5 (class)	-	67,437%	-	65,819%	66,253%	65,386%	-	1,618%
C5-Rules (class)	-	67,008%	-	66,283%	76,923%	55,643%	-	0,725%
Naïve Bayes (class)	-	66,492%	-	63,818%	49,876%	77,761%	-	2,674%

Fonte: Autoria Própria.

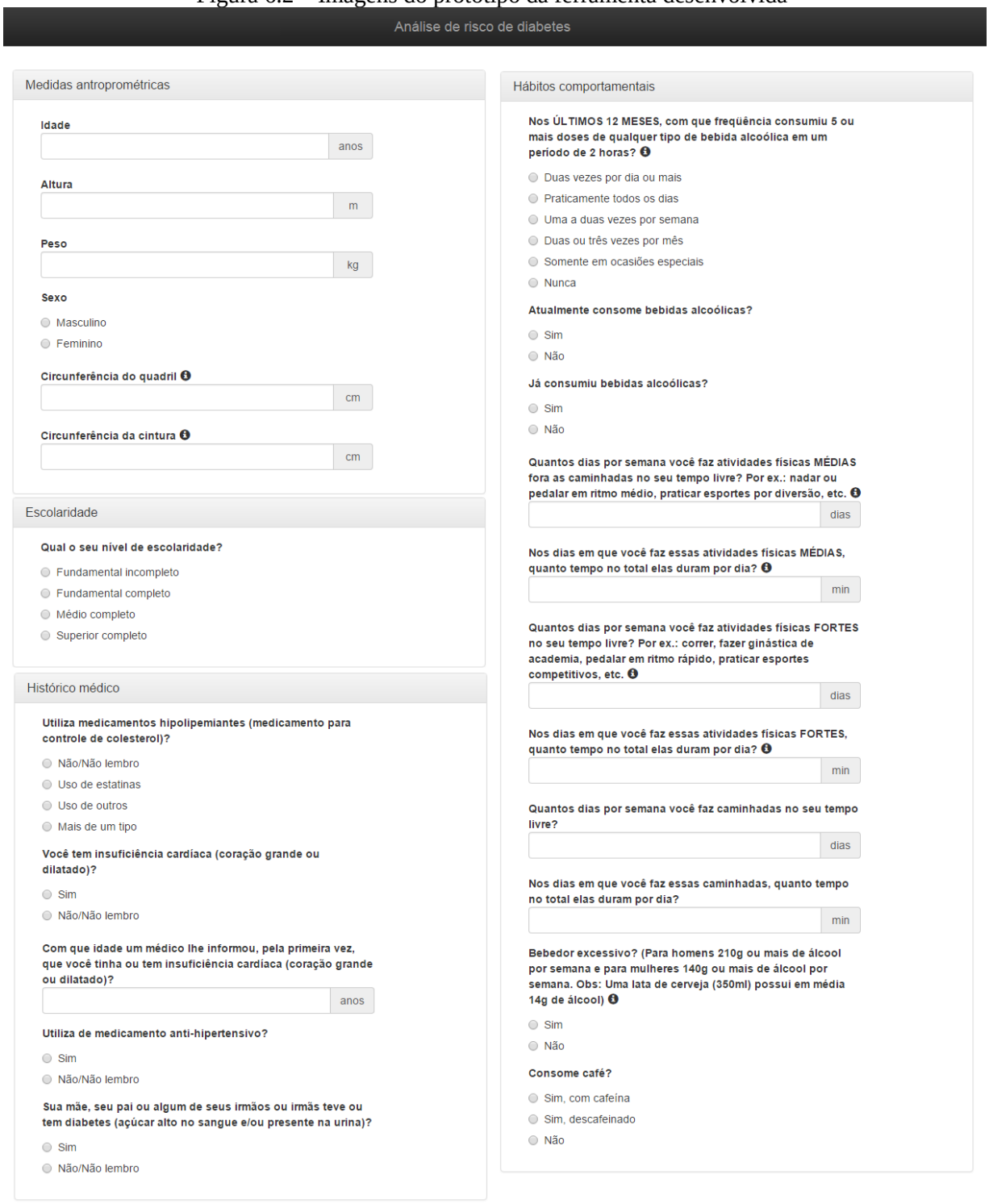
Primeiramente, apesar de os resultados terem ficado fora do intervalo de confiança descrito na Tabela 6.5 demonstrando um leve sobreajuste aos dados de treinamento, a maioria dos algoritmos permaneceu com um bom desempenho no teste de generalização, com diferenças abaixo de 1,5% para os resultados da etapa anterior. O algoritmo K-NN apresentou o pior resultado entre os algoritmos com saída probabilística, tendo um decremento de 2,058% na média de AUC de um teste para outro. Mesmo assim a média de AUC no teste de generalização foi de 73,565%, que indica um bom desempenho.

Nesta etapa, os modelos criados com Regressão Logística tiveram melhores resultados em média dos modelos gerados com Redes Neurais, em oposição aos resultados da etapa de estimativa de erros. Porém a diferença entre eles continua não relevante. O mesmo aconteceu com os algoritmos K-NN e *Naïve Bayes*, onde o último apresentou melhor desempenho neste teste ao contrário do resultado da etapa anterior.

6.5 Ferramenta Web para Detecção de Diabetes não diagnosticado

Por fim, o modelo gerado com Redes Neurais, que obteve melhor resultado na validação cruzada para estimativas de erros (e também apresentou resultados satisfatórios no teste de generalização), foi selecionado para a construção de uma ferramenta web para detecção de diabetes não diagnosticado. Esse modelo, contudo, deve ser comparado com escores de risco já utilizados para detecção de diabetes em um conjunto de dados independente dos dois modelos comparados. Uma imagem adaptada do protótipo da ferramenta pode ser vista na Figura 6.2.

Figura 6.2 – Imagens do protótipo da ferramenta desenvolvida



Fonte: Autoria Própria.

Os pesos sinápticos das ligações da rede neural do modelo selecionado são descritos na Tabela 6.7.

Tabela 6.7 – Pesos sinápticos do modelo selecionado

Variável Dicotomizada	Peso Sináptico aprendido
bias	-1,59441303
rcta8.2	0,09754571
hfda11.1	0,33198989
diea133.1	-0,34972045
diea133.2	-0,14280913
a_imc1	0,33757347
a_imc2.2	-0,80899773
a_imc2.3	-0,8245655
a_imc2.4	-0,64030241
a_cint	-0,16799617
a_rcq	0,57975475
a_escolar.2	-0,21583397
a_escolar.3	-0,39423055
a_escolar.4	-0,70969956
a_gidade.2	0,51739246
a_gidade.3	0,84807029
a_gidade.4	0,98165439
a_medanthipert.1	0,40522408
a_bebexcessivo.1	0,40374753
a_binge.1	0,12092468
a_ativfisica.2	-0,07996297
a_ativfisica.3	-0,34265755
a_medredlip.1	-0,22684034
a_medredlip.2	0,13770741
a_medredlip.3	-0,4219824
a_sfhfprem.1	-0,29823203

Fonte: Autoria Própria.

A classificação de um novo caso utilizando esse modelo pode ser realizada da seguinte maneira:

1. Padronizar os valores das variáveis quantitativas através da subtração do valor pela média e divisão pelo desvio padrão descritos na Tabela 6.8.
2. Dicotomizar as variáveis categóricas conforme descrito nas seções 2.2.2 e 5.2, onde a primeira categoria da variável não possui uma variável "*dummy*" associada e é definida quando todas variáveis criadas relacionadas com a variável categórica em questão possuem valor "0".
3. Fazer a soma ponderada dos valores das variáveis resultante das etapas anteriores utilizando os pesos sinápticos descritos na Tabela 6.7.
4. Somar o valor de "*bias*", descrito na primeira linha da Tabela 6.8.
5. Passar o resultado da soma (β) na função logística (função de ativação da rede neural do modelo em questão): $\frac{1}{1-e^{-\beta}}$
6. Se o resultado for maior que 0,11 (ponto de corte definido no trabalho), classificar o caso como positivo (com diabetes) senão, como negativo (não tem diabetes).

Tabela 6.8 – Parâmetros para padronização das variáveis quantitativas

Variável Quantitativa	Média	Desvio Padrão
a_imc1	26,7663321	4,617303
a_cint	90,3675441	12,55463
a_rcq	0,8889311	0,08615528

Fonte: Autoria Própria.

7 CONCLUSÃO

O objetivo principal da dissertação apresentada foi comparar modelos preditivos para detectar diabetes não diagnosticado gerados a partir de diferentes técnicas e algoritmos utilizando somente variáveis de baixo custo de obtenção. Para tanto foram utilizados dados da linha de base do Estudo Longitudinal de Saúde do Adulto (ELSA-Brasil).

Foram descritos e discutidos os principais conceitos e algoritmos utilizados para gerar os modelos. Também foram apresentados os principais problemas e desafios existentes para realizar esse tipo de tarefa bem como os meios e orientações para abordá-los de forma eficiente.

O processo de criação e validação dos modelos foi realizado em quatro etapas principais. Uma etapa de afinação dos parâmetros, onde cada algoritmo foi avaliado variando as configurações de parâmetros, como por exemplo, a quantidade de neurônios da camada oculta da rede neural. Na segunda etapa, foi utilizada uma técnica de seleção automática de variáveis para criação de subconjuntos que, quando utilizados para a construção de modelos preditivos, poderiam criar modelos com maior capacidade preditiva e menos complexos que os modelos gerados utilizando o conjunto completo dos dados. A terceira etapa avaliou os esquemas de aprendizagem definidos na etapa anterior através de validação cruzada com o objetivo de estimar os erros de cada esquema de aprendizagem e escolher o com melhor poder preditivo. Por fim, a última etapa foi realizada com os melhores resultados de cada algoritmo da etapa anterior para verificar a capacidade de generalização dos esquemas de aprendizagem em dados nunca visto antes, separado no início dos experimentos.

Como discutido na Seção 2.2.1, a utilização de variáveis redundantes pode atrapalhar a aprendizagem do modelo gerado. Contudo foram selecionados como variáveis candidatas as seguintes variáveis que seriam redundantes ou muito parecidas: (i) a_imc1 e a_imc2; (ii) a_cint e a_rcq; e, (iii) a_bebeexcessivo e a_binge. Uma vez que foi realizada uma etapa de seleção automática de variáveis multivariada, baseada no desempenho dos modelos gerados por diferentes subconjuntos de variáveis selecionados, não houve preocupação em remover tais variáveis da análise. Porém, os subconjuntos, criados nessa etapa: nnet-fs, glm-fs e nb-fs (e, naturalmente, o conjunto união e o original, conforme Tabela 6.2), possuíam uma ou mais dessas redundâncias. Ou seja, os modelos gerados com as redundâncias eventualmente tiveram um desempenho melhor que os modelos gerados sem elas. Mesmo depois, nos demais testes, os resultados utilizando tais conjuntos foram melhores que utilizando o conjunto knn-fs, que não possui as redundâncias. Como trabalho futuro, seria interessante refazer os testes

eliminando tais redundâncias previamente e comparar com os resultados obtidos. O modelo criado com redes neurais que obteve melhor resultado em termos de AUC na terceira etapa e obteve resultados satisfatórios na quarta etapa foi selecionado para a criação de uma ferramenta WEB que pode ser disponibilizada para detecção de diabetes em uma população baseada em um questionário simples.

Os algoritmos RIPPER (*JRip,* todas versões*),* K-NN (versão com saída binária) e C5 (versões com saídas probabilísticas) foram removidos da análise na primeira etapa porque apresentaram resultados insatisfatórios. As causas disso não foram investigadas e fica como sugestão para trabalho futuro. Os demais prosseguiram por todas as etapas do trabalho. Os algoritmos de Regressão Logística e Redes Neurais (sem neurônios na camada oculta) apresentaram os melhores resultados, destacando-se dos demais. Tanto um quanto o outro seriam as melhores escolhas para futuras modelagens utilizando esses dados. Modelos gerados por Redes Neurais Artificiais sem neurônios na camada oculta, como o que obteve melhor resultado e foi selecionado para a construção da ferramenta WEB, são equivalentes a modelos lineares gerados por Regressão Logística, o que indica que existe uma relação linear entre as variáveis preditoras e a variável alvo.

Os resultados obtidos sustentam a utilização da técnica de regressão logística para a criação de modelos preditivos de saúde como os trabalhos apresentados no Capítulo 3.

Em geral, a maioria dos resultados alcançou um valor de AUC acima de 70% com um mínimo de afinação de parâmetros. Isso demonstra um grande potencial para a criação de modelos preditivos de diabetes com esse conjunto de dados para ser utilizado na prática. Outros trabalhos, seguindo o mesmo processo e, inclusive, utilizando as mesmas implementações, também podem ser realizados para verificar potenciais algoritmos e viabilidade do conjunto de dados para detectar outras doenças crônicas. O processo utilizado também pode servir como referência para futuras comparações de algoritmos com objetivo de decidir qual produz os modelos com maior capacidade preditiva.

Analisando os resultados, pode-se calcular que, utilizando o algoritmo de Redes Neurais ou, com perguntas simples a uma população semelhante à analisada pelo banco de dados ELSA-Brasil, das 1.497 pessoas participantes do ELSA-Brasil que têm diabetes e não sabem, o método proposto identificaria aproximadamente 1.051 pessoas (sensibilidade de 70,223%). Se utilizasse o de Regressão Logística, o índice de identificação seria de aproximadamente 1.070 pessoas (sensibilidade de 71,464%). E dentre os 3.078 participantes que não tem diabetes, o modelo proposto classificaria corretamente 2.075 e 2.066 (especificidade de 67,413% e 67,110%), utilizando os modelos de regressão logística e redes

neurais, respectivamente. O impacto desse resultado poderia ser muito grande na população, e teria o potencial, quando integrado com intervenções comprovadamente efetivas, de evitar ou retardar o desenvolvimento de diabetes em grande número de indivíduos.

Outros estudos podem ser realizados adicionando variáveis presentes no ELSA-Brasil que não foram utilizadas neste trabalho, com o objetivo de encontrar alguma característica desconhecida ou ainda não difundida de risco de diabetes. Além disso, futuramente poderá se utilizar dados de outros momentos do ELSA-Brasil (que ainda não estavam disponíveis durante a elaboração do presente trabalho) para realizar análises temporais individuais permitindo, assim, criar modelos para prever o risco de um indivíduo vir a contrair uma doença crônica, dado um determinado tempo.

Uma limitação técnica desse trabalho foi a inexistência de cálculos de significância estatística para algumas das comparações realizadas. Em alguns casos, como na etapa de afinação de parâmetros, a comparação utilizando testes de significância atrapalharia a análise pois mesmo se as diferenças entre os resultados não tivessem significância estatística precisaria de outros critérios mais complexos para decidir qual dentre as configurações de parâmetros seria a ideal. Tais critérios seriam específicos de cada algoritmo.

Testes de significância estatística também são utilizados amplamente durante a seleção automática de variáveis para decidir se um nodo deve ser expandido durante a busca heurística. Porém a implementação utilizada não realizava tal abordagem e teria que alterar a implementação utilizada, utilizar outra implementação ou ainda codificar manualmente a busca para incorporar testes desse tipo.

Outra limitação foi a ausência de comparação dos resultados obtidos com relação à outros trabalhos similares. A comparação da capacidade preditiva de diferentes modelos pode ser realizada quando o conjunto de teste é o mesmo e independente do conjunto de teste utilizado para treinar os modelos que estão sendo testados, caso os modelos tenham sido criados com conjuntos de treinamento diferentes. Além disso, o método de avaliação utilizado deve ser o mesmo.

Dessa forma, como sugestão de trabalho futuro, o modelo gerado nesse trabalho pode ser utilizado em dados provenientes de outras fontes ou ainda o questionário gerado pode ser aplicado diretamente em indivíduos suspeitos de terem diabetes antes de fazerem exames laboratoriais para diagnosticar essa condição. Ao mesmo tempo outros modelos/questionários existentes para esse fim podem ser aplicados e os resultados das duas abordagens comparados. O escore de risco FINDRISK (Lindstrom 2003) possui características parecidas

com o modelo construído neste trabalho utilizando somente variáveis não invasivas para detectar incidência de diabetes e pode ser utilizado como referência para futuras comparações.

REFERÊNCIAS

ABBASI, A.; PEELEN, L. M.; CORPELEIJN, E; VAN DER SCHOUW, Y. T.; STOLK, R. P, SPIJKERMAN, A. M. W.; VAN DER A, D. L.; MOONS, K. G. M.; NAVIS, G.; BAKKER, S. J. L.; BEULENS, J. W. L. Prediction models for risk of developing type 2 diabetes: systematic literature search and independent external validation study. In **BMJ**, 345:e5900, 2012.

AGRAWAL, R.; SRIKANT, R. Fast algorithms for mining association rules in large databases. INTERNATIONAL CONFERENCE ON VERY LARGE DATABASES (VLDB), 20. Santiago de Chile, Chile, 1994. **Proceedings...** Santiago de Chile: Morgan Kaufmann, 1994. p. 487-499.

ALPAYDIN, E. **Introduction to Machine Learning**. MIT Press, 2nd edition. 2010

AQUINO, E. M. L.; BARRETO, S. M.; BENSENOR, I. M.; CARVALHO, M. S.; CHOR, D.; DUNCAN, B. B.; LOTUFO, P. A.; MILL, J. G.; MOLINA, M. C; MOTA, E. L. A.; PASSOS, V. M. A.; SCHMIDT, M. I.; SZKLO, M. Brazilian Longitudinal Study of Adult Health (ELSA-Brasil): Objectives and Design. **American Journal of Epidemiology**, Vol. 175, n. 4, p. 315-324. 2012.

BARBER, S. R.; DAVIES, M. J.; KHUNTI, K.; GRAY, L. J. Risk assessment tools for detecting those with pre-diabetes: A systematic review. **Diabetes Research and Clinical Practice**, Volume 105, Issue 1, Pages 1-13, July 2014.

BEAGLEY, J., GUARIGUATA, L., WEIL, C., MOTALA, A. A., Global estimates of undiagnosed diabetes in adults, **Diabetes Research and Clinical Practice**, Volume 103, Issue 2, Pages 150-160, February 2014.

BELLAZI, R.; ZUPAN, B. Predictive data mining in clinical medicine: current issues and guidelines. **International journal of medical informatics**. v. 77, n. 2, p. 81–97. 2006.

BREIMAN, L.; FRIEDMAN, J.; OLSHEN, R. A; STONE, C. J. **Classification and Regression Trees**. Chapman and Hall/CRC, 1st edition. 1984.

BROWN, D. E. Introduction to Data Mining for Medical Informatics. **Clinics in Laboratory Medicine.** Vol. 28, n. 1, p. 9-35. 2008.

BUIJSSE, B.; SIMMONS, R. K.; GRIFFIN, S. J.; SCHULZE, M. B. Risk Assessment Tools for Identifying Individuals at Risk of Developing Type 2 Diabetes. **Epidemiologic Reviews**, *33*(1), 46–62, 2011.

CASADO L.; VIANNA, L. M.; THULER, L. C. S. Risk Factors for Chronic non Communicable Diseases in Brazil: a Systematic Review. **Revista Brasileira de Cancerologia**. Vol. 55, n. 4, p. 379-388. 2009.

CENDROWSKA, J. PRISM: An algorithm for inducing modular rules. **International Journal of Man-Machine Studies**. Vol. 27, p. 349-370. 1987

CHANG, C.; WANG, C; JIANG, B. C. Using data mining techniques for multi-diseases prediction modeling of hypertension and hyperlipidemia by common risk factors. **Expert Systems with Applications**. Vol. 38, n. 5, p. 5507-5513. 2011.

CHOI, S. B.; KIM W. J.; YOO, T. K.; PARK, J. S.; CHUNG, J. W.; LEE, Y.; KANG, E. S.; KIM, D. W. Screening for Prediabetes Using Machine Learning Models, **Computational and Mathematical Methods in Medicine**, vol. 2014.

CLARK, P.; NIBLETT, T. The CN2 induction algorithm. **Machine Learning**. Vol. 3, n. 4, p. 261-283. 1989.

COHEN, W. Fast effective rule induction. INTERNATIONAL CONFERENCE ON MACHINE LEARNING (ICML), 12. Tahoe City, Estados Unidos, 1995. **Proceedings...** [S.l.:s.n.], 1995. p. 115–123.

COLLINS, G. S.; MALLETT, S.; OMAR, O.; YU, L.M. Developing risk prediction models for type 2 diabetes: a systematic review of methodology and reporting. **BMC Medicine**, Vol. 9. Review. 2011.

CORTES, C.; VAPNIK, V. Support-Vector Networks. **Machine Learning**. Vol. 20, n. 3, p. 273-297. 1995.

COVER, T.; HART, P., Nearest neighbor pattern classification. **IEEE Transactions on Information Theory**, vol.13, no.1, pp.21,27, 1967.

FRIEDMAN, N.; GEIGER, D.; GOLDSZMIDT, M. Bayesian Network Classifiers. **Machine Learning**. Vol. 29, p. 131-163. 1997.

GHYON, I.; ELISSEEFF, A. An Introduction to Variable and Feature Selection. **Journal of Machine Learning Research**. v. 3, p. 1157–1182, 2003.

GLAUBER, H.; KARNIELI, E. Preventing Type 2 Diabetes Mellitus: A Call for Personalized Intervention. **The Permanente Journal**, *17*(3), 74–79, 2013.

GONZALEZ-ABRIL, L.; CUBEROS, F. J.; VELASCO, F.; ORTEGA, J. A. Ameva: An autonomous Discretization algorithm, **Expert Systems with Applications**, 36, 5327–5332, 2009.

GUARIGUATA, L.; WHITING, D.R.; HAMBLETON, I.; BEAGLEY, J.; LINNENKAMP, U.; SHAW, J. E. Global estimates of diabetes prevalence for 2013 and projections for 2035, **Diabetes Research and Clinical Practice**, Volume 103, Issue 2, Pages 137-149, February 2014.

HAREL, O.; ZHOU, X.-H. Multiple imputation: review of theory, implementation and software. **Statistics in Medicine**. 26: 3057–3077. 2007.

HARRISON, J. H. Introduction to the mining of clinical data. **Clinics in laboratory medicine**. v.28, n.1, p.1–7. 2008.

HAYKIN, S. **Neural Networks and Learning Machines**. Prentice Hall, 3rd edition. 2008.

IDF, International Diabetes Federation. *IDF Diabetes Atlas, 7 ed.* Brussels, Belgium: International Diabetes Federation, 2015.

JEREZ, J. M.; MOLINA, I.; LAENCINA, P. J. G-; ALBA, E.; RIBELLES, N.; MARTÍN, M.; FRANCO, L. Missing data imputation using statistical and machine learning methods in a real breast cancer problem. **Artificial Intelligence in Medicine**, Volume 50, Issue 2, Pages 105-115. 2010.

KASS, G. V. An Exploratory Technique for Investigating Large Quantities of Categorical Data. **Applied Statistics**, Vol. 29, No. 2, p. 119-127. 1980.

KHALILIA M.; CHAKRABORTY S.; POPESCU M. Predicting disease risks from highly imbalanced data using random forest. **BMC Medical Informatics and Decision Making**. 2011.

KOBUS, L. S.; ENEMBRECK, F.; SCALABRIN, E. E.; DIAS, J. S.; SILVA, S. H. Automatic Knowledge Discovery and Case Management: an Effective Way to Use Databases to Enhance Health Care Management. **Artificial Intelligence Applications and Innovations**. Vol. 296. 2009.

KOH, H. C.; TAN, G. Data mining applications in healthcare. **Journal of healthcare information management.** v. 19, n. 2, p. 64–72. 2005.

KOTSIANTIS, S. B.; ZAHARAKIS I. D.; PINTELAS, P. E. Machine learning: a review of classification and combining techniques. **Artificial Intelligence Reviews**, Vol. 26, p. 159-190. 2006.

KUROSAKI M.; HIRAMATSU N.; SAKAMOTO M.; SUZUKI Y.; IWASAKI M.; TAMORI A.; MATSUURA K.; KAKINUMA S.; SUGAUCHI F.; SAKAMOTO N.; NAKAGAWA M.; IZUMI N. Data mining model using simple and readily available factors could identify patients at high risk for hepatocellular carcinoma in chronic hepatitis C. **Journal of Hepatology**. Vol. 56, n. 3, p. 602-8. 2012.

LANGLEY, P.; IBA W.; THOMPSON, K. An analysis of Bayesian classifiers. NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE (AAAI), 10. San Jose, Estados Unidos, 1992. **Proceedings...** [S.l.]: AAAI Press, 1992. p 223-228.

LAVRAC, N. Selected techniques for data mining in medicine. **Artificial intelligence in medicine**. v. 16, n. 1, p. 3–23. 1999.

LEE, Y.; BANG, H.; KIM, H. C.; KIM, H. M.; PARK, S. W.; KIM, D. J. A Simple Screening Score for Diabetes for the Korean Population: Development, validation, and comparison with other scores. **Diabetes Care**, 35(8), 2012.

LEE, B. J.; KU, B.; NAM, J.; PHAM, D. D.; KIM, J. Y. Prediction of Fasting Plasma Glucose Status using Anthropometric Measures for Diagnosing Type 2 Diabetes. **IEEE Journal of Biomedical and Health Informatics**, Vol. 18, No. 2, March 2014.

LINDSTROM, J.; TUOMILEHTO, J. The Diabetes Risk Score: A practical tool to predict type 2 diabetes risk. **Diabetes Care,** 26(3), 2003.

LIU, H.; YU, L. Toward Integrating Feature Selection Algorithms for Classification and Clustering. **IEEE Transactions on Knowledge and Data Engineering**, v. 17, n. 4, p. 491–502. 2005.

MANSOUR, R.; EGHBAL, Z.; AMIRHOSSEIN, H. Comparison of Artificial Neural Network, Logistic Regression and Discriminant Analysis Efficiency in Determining Risk Factors of Type 2 Diabetes. **World Applied Sciences Journal**, Vol. 23 Issue 11, p1522, 2013.

MATHIAS J. S.; AGRAWAL A.; FEINGLASS J.; COOPER A. J.; BAKER D. W.; CHOUDHARY A. Development of a 5 year life expectancy index in older adults using predictive mining of electronic health record data. **Journal of the American Medical Informatics Association**. 2013.

MURPHY, S. M. E.; CASTRO, H. K.; SYLVIA, M. Predictive modeling in practice: improving the participant identification process for care management programs using condition-specific cut points. **Population health management**. v.14, n.4, p. 205-210. 2011.

NEUVIRTH, H.; FLATO, M. O-; HU, J.; LASERSON, J.; KOHN, M. S.; EBADOLLAHI, S.; ZVI, M. R-. Toward personalized care management of patients at risk: the diabetes case study. ACM SIGKDD INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING (KDD), 17. San Diego, Estados Unidos, 2011. **Proceedings...** New York: ACM, 2011. p. 395-403.

NOBLE, D.; MATHUR, R.; DENT, T.; MEADS, C.; GREENHALGH, T. Risk models and scores for type 2 diabetes: systematic review. ***BMJ*. Review**. 2011.

OBENSHAIN, M. K. Application of data mining techniques to healthcare data. **Infection Control and Hospital Epidemiology**. vol. 25, n. 8, p. 690-5. 2004.

QUINLAN, J. R. **C4.5: Programs for Machine Learning**. Morgan Kaufmann Publishers. 1993.

RAMEZANKHANI, A.; POURNIK, O.; SHAHRABI, J.; KHALILI, D.; AZIZI, F; HADAEGH, F. Applying decision tree for identification of a low risk population for type 2 diabetes. Tehran Lipid and Glucose Study. **Diabetes Research and Clinical Practice**, Volume 105, Issue 3, Pages 391-398, September 2014.

ROBNIK-SIKONJA, M.; KONONENKO, I. Theoretical and Empirical Analysis of ReliefF and RReliefF. **Machine Learning Journal**, 53:23-69, 2003.

SHANKARACHARYA; ODEBRA, D.; SAMANTA, S.; VIDYARTHI, A. S. Computational Intelligence in Early Diabetes Diagnosis: A Review. **The Review of Diabetic Studies: RDS**, 7(4), 252–262, 2010.

SCHAFER, J. Multiple imputation: a primer. *Statistical Methods in Medical Research*, 8:3-15, 1999.

SCHMIDT, M. I.; DUNCAN, B. B.; MIL, J. G.; LOTUFO, P. A.; CHOR, D.; BARRETO, S. M.; AQUINO, E. M. L.; PASSOS, V. M. A.; MATOS, S. M. A.; MOLINA, M. C. B.; CARVALHO, M. S.; BENSENOR, I. M. Cohort Profile: Longitudinal Study of Adult Health (ELSA-Brasil). **International Journal of Epidemiology**. 2014.

SCHMIDT, M. I.; DUNCAN, B. B.; BANG, H.; PANKOW, J. S.; BALLANTYNE, C. M.; GOLDEN, S. H.; FOLSOM, A. R.; CHAMBLESS, L. E. Identifying Individuals at High Risk for Diabetes: The Atherosclerosis Risk in Communities study. **Diabetes Care**, vol. 28, no. 8. 2005.

SESEN M. B.; NICHOLSON A. E.; ALCANTARA R. B-; KADIR T.; BRADY M. Bayesian networks for clinical decision support in lung cancer care. **PLoS One**, Vol. 8, n. 12. 2013.

THOOPPUTRA, T.; NEWBY, D.; SCHNEIDER, J.; Li, S. C. Survey of diabetes risk assessment tools: concepts, structure and performance. **Diabetes/Metabolism Research and Reviews**, 28: 485–498, 2012.

TRAJMAN, A.; LUIZ, R. R. McNemar $\chi 2$ test revisited: comparing sensitivity and specificity of diagnostic examinations. **Scandinavian Journal of Clinical & Laboratory Investigation** 68:1 , 77-80, 2008.

TRUJILLANO, J.; BADIA, M.; SERVIÁ, L.; MARCH, J.; POZO, A. R-. Stratification of the severity of critically ill patients with classification trees. **BMC medical research methodology.** Vol. 9, n. 83. 2009.

VÖLZKE H.; FUNG G.; ITTERMANN T.; YU S.; BAUMEISTER S. E.; DÖRR M.; LIEB W.; VÖLKER U.; LINNEBERG A.; JØRGENSEN T.; FELIX S. B.; RETTIG R.; RAO B.; KROEMER H. K. A new, accurate predictive model for incident hypertension. **Journal of Hypertension**. Vol. 31, n. 11, p. 2142-50. 2013.

WANG, C.; LI, L.; WANG, L.; PING, Z.; FLORY, M. T.; WANG, G.; XI, Y; LI, W. Evaluating the risk of type 2 diabetes mellitus using artificial neural network: An effective classification approach. **Diabetes Research and Clinical Practice**, Volume 100, Issue 1, Pages 111-118, April 2013.

WEIR, S.; AWEH, G.; CLARK, R. E. Case selection for a Medicaid chronic care management program. **Health care financing review**. v.30, n.1, p. 61–74. 2008.

WHO (World Health Organization) **2008-2013 action plan for the global strategy for the prevention and control of noncommunicable diseases**. Geneva: WHO, 2008.

WITTEN, I. E.; FRANK, E.; HALL, M. A**. DATA MINING: Practical Machine Learning Tools and Techniques**. Morgan Kaufmann, 3rd edition. 2011.

YAN, H.; JIANG, Y.; ZHENG, J.; PENG, C.; LI, Q. A multilayer perceptron-based medical decision support system for heart disease diagnosis. **Expert Systems with Applications**. Vol. 30, n. 2, p. 272-281. 2006.

YEH, D.; CHENG, C.; CHEN, Y. A predictive model for cerebrovascular disease using data mining. **Expert Systems with Applications**. Vol 38, n. 7, p. 8970-8977. 2011.

YOO, I.; ALAFAIREET, P.; MARINOV, M.; Hernandez, K. P-.; GOPIDI, R.; CHANG, J. Data mining in healthcare and biomedicine: a survey of the literature. **Journal of medical systems.** v. 36, n. 4, p. 2431–48. 2012.