André Rodrigues Barbosa

Análise de Desempenho de Diferentes Algoritmos de Aprendizado de Máquina para Análise de Risco

Projeto de Graduação em Computação

Santo André - SP

2019

André Rodrigues Barbosa

Análise de Desempenho de Diferentes Algoritmos de Aprendizado de Máquina para Análise de Risco

Projeto de Graduação apresentado ao Curso de Bacharelado em Ciências da Computação da Universidade Federal ABC, como requisito parcial para obtenção do título de Bacharel em Ciências da Computação.

Universidade Federal do ABC – UFABC

Centro de Matemática, Computação e Cognição

Curso de Bacharelado em Ciências da Computação

Orientador: Prof. Denis Gustavo Fantinato

Santo André - SP

2019

Sumário

1 INTRODUÇÃO 5

2 JUSTIFICATIVA 6

3 OBJETIVOS 7
3.1 Objetivos Específicos 7

4 FUNDAMENTAÇÃO TEÓRICA 8
4.1 Análise de Risco 8
4.2 Modelagem de Dados 8
4.2.1 Análise Preditiva 8
4.3 Estruturas para Regressão 9
4.3.1 Árvore de Decisão 9
4.3.1.1 Critérios de Divisão 11
4.3.1.2 Regra de Divisão dos Atributos 12
4.3.1.3 Definição dos Nós Terminais 12
4.3.1.4 Utilização da árvore para previsões 12
4.3.2 Modelos Lineares Generalizados 13
4.3.2.1 Modelos Lineares 13
4.3.2.2 Generalização 13
4.3.3 Redes Neurais de Função de Base Radial 14
4.3.3.1 Kernels 16
4.3.3.2 Escolha dos centros e dispersões: K-Médias 16
4.3.3.3 Ajuste dos Pesos Lineares 17
4.3.3.3.1 Mínimos Quadrados 17
4.3.3.3.2 Solução pela Equação Normal 17
4.4 Métodos de Avaliação do Modelo 18
4.4.1 Avaliação do Modelo 18
4.4.2 Avaliação de Classificadores 19
4.4.2.1 Acurácia 19
4.4.2.2 Taxa de Erro 19
4.4.2.3 Sensibilidade ou Revocação 20
4.4.2.4 Especificidade 20
4.4.2.5 Precisão 20
4.4.2.6 Acurácia Balanceada 20
4.4.2.7 Medida F 20
4.4.3 Avaliação de Regressores 21

4.4.4 Outras formas de avaliação . . . 21
4.4.5 Separação dos Dados para Treinamento . . . 21
4.4.5.1 Hold-Out . . . 21
4.4.5.2 Validação Cruzada ou K-Fold . . . 22

5 METODOLOGIA . . . 23
5.1 Base de Dados . . . 23
5.1.1 Dados Utilizados . . . 23
5.1.2 Descrição dos Dados . . . 23
5.2 Ferramentas Utilizadas . . . 25
5.2.1 Modelos . . . 25
5.3 Preparação e Modelagem dos Dados . . . 26
5.3.1 Análise Preliminar dos Dados . . . 26
5.3.2 Preparação dos dados . . . 27
5.3.2.1 Remoção de Colunas . . . 27
5.3.2.2 Conversão de Variáveis Monetárias para Números . . . 27
5.3.2.3 Substituição de Valores Nulos . . . 28
5.3.2.4 Homogeneização de Marcadores(*Flags*) . . . 28
5.3.2.5 Separação de variáveis categóricas em diversas variáveis binárias . . . 28
5.3.3 Conjuntos de Teste E Treinamento . . . 28
5.3.4 Otimização . . . 29
5.3.4.1 Análise Preliminar - Validação Cruzada . . . 29
5.3.4.2 Busca de Parâmetros . . . 29
5.3.4.3 Transformação dos Atributos . . . 29
5.3.5 Avaliação dos Resultados . . . 30
5.3.5.1 Tarefa de Classificação - CLAIM_FLAG . . . 30
5.3.5.2 Tarefa de Regressão - CLM_AMT . . . 30
5.3.5.3 Tarefa de Regressão - CLM_FREQ . . . 30

6 CRONOGRAMA . . . 31

REFERÊNCIAS . . . 33

1 Introdução

A análise de risco é um tópico importante para diversos setores do mercado, desde pequenas seguradoras até grandes entidades financeiras podem se beneficiar de análises acuradas. Esse tipo de instituição depende de estimativas de risco confiáveis para definir suas estratégias, assegurar vantagens competitivas e se manterem sustentáveis num mercado cada vez mais saturado.

Visto que técnicas de aprendizagem de máquina ocupam uma posição notória para análises envolvendo grande volume de dados, esse projeto procura comparar outros modelos com aquele atualmente utilizado no mercado de seguros, os Modelos Lineares Generalizados, também referenciados pela sigla GLM, do inglês *Generalized Linear Model*. Serão analisadas as técnicas de Árvores de Decisão e também as Redes Neurais de Base Radial (doravente denotadas por RBFNN, do inglês *Radial Basis Function Neural Network*) para verificar se são alternativas viáveis ao modelo utilizado atualmente

Serão utilizados dados disponíveis publicamente na plataforma *Keagle*. Os mesmos atributos alimentarão cada um dos modelos escolhidos independentemente e os parâmetros de cada modelo serão ajustados, a fim de se obter uma comparação justa entre eles. No caso das RBFNN, será utilizado o kernel mais comum, o Gaussiano.

2 Justificativa

Na contratação de um seguro automotivo, uma análise de risco é realizada a fim de determinar o preço que será cobrado pela cobertura. Essa análise é vital: se o risco estimado for muito baixo, a seguradora irá ter prejuízo no longo prazo, pagando mais sinistros do que arrecada, já se o risco estimado for muito alto, a seguradora pode perder clientes por estar cobrando valores muito altos.

Hoje as seguradoras ainda tem acesso limitado a fontes de dados externas, baseando suas predições principalmente em informações disponíveis publicamente e no questionário respondido pelo segurado no momento da contratação. Na ausência de mais dados, uma das possíveis maneiras de melhorar a acurácia das predições de risco é a utilização de outras técnicas ou algoritmos.

Portanto, esse trabalho irá comparar dois outros algoritmos com o GLM para verificar se eles podem ser, pelo menos, tão eficientes quanto ele, ou se realmente os GLM são modelos melhores para esse tipo de análise.

3 Objetivos

O principal objetivo desse projeto é verificar se as Árvores de Decisão ou se a RBFNN podem ser uma alternativa viável aos modelos de GLM utilizados atualmente ao analisar uma base com um número limitado de atributos. O desempenho será comparado pela capacidade de predizer corretamente a ocorrência ou não de eventos considerados de risco e também predizer corretamente os prejuízos monetários. Ou seja, será realizada tanto a análise desses algoritmos com classificação quanto com regressões.

3.1 Objetivos Específicos

Para atingir o objetivo principal será necessário:

1. Analisar e explorar a base de dados escolhida;
2. Pré-processar os dados para que eles possam ser utilizados com os algoritmos escolhidos;
3. Avaliar o desempenho dos algoritmos após receber os dados pré-processados:
   a) Modelos Lineares Generalizados
   b) Árvore de Decisão
   c) Redes Neurais Artificiais de Funções de Base Radial
4. Encontrar uma boa combinação de transformação dos atributos para cada modelo, a fim de fazer uma comparação justa entre modelos.
5. Descobrir a relação entre o desempenho do GLM e dos demais modelos e verificar se houve ganho de eficiência, e de quanto foi esse ganho. A eficiência será mensurada com o erro quadrático médio, para as tarefas de regressão, e comparando a acurácia, taxa de falsos positivos, taxa de falsos negativos e F1 score para os problemas de classificação.

4 Fundamentação Teórica

4.1 Análise de Risco

A análise de riscos individuais não relacionados à vida normalmente busca prever a frequência ou severidade de um evento (KAAS et al., 2008). Essas previsões são usadas junto de outros fatores como despesas, carregamento e margem de lucro, para determinar o prêmio comercial que será cobrado do segurado (FERREIRA, 2005).

Yeo et al. (2001) afirmam que existem dois requisitos principais para se obter boas previsões: o primeiro é uma grande massa de dados e o segundo é que essa massa apresente características de homogeneidade. Em outras palavras, são necessários muitos registros para que dados observados na base de treinamento se aproximem da realidade, e aqueles registros num mesmo grupo de risco devem ser suficientemente semelhantes para que as previsões sobre eles sejam acuradas.

4.2 Modelagem de Dados

Existem diferentes formas de se modelar os dados, extraindo informações úteis dos mesmos. Os modelos mais comuns são gerados através da abordagem estatística e/ou de técnicas de aprendizado de máquina (RAJARAMAN; ULLMAN, 2011). Esses modelos podem ser descritivos ou preditivos: enquanto nos modelos descritivos busca-se caracterizar as informações presentes na base de dados, nos modelos preditivos a informação da base de dados é utilizada como base para realizar inferências sobre o futuro. Neste trabalho, focaremos nos modelos preditivos, conforme detalhado a seguir (HAN; PEI; KAMBER, 2011).

4.2.1 Análise Preditiva

A análise preditiva busca prever o valor futuro de uma variável dependente, baseado nas variáveis independentes. A variável dependente também é chamada de atributo alvo, e as variáveis independentes são chamadas de atributos explicativos. Os tipos principais de predição são a classificação e a regressão. As análises preditivas classificatórias normalmente são usadas para prever atributos alvo categóricos/discretos, e as análises preditivas regressivas normalmente tentam prever atributos do tipo numérico contínuo (TAN, 2018). Note que as palavras atributos e variáveis serão tratadas como sinônimos nesse trabalho.

Antes de realizar a análise preditiva, é comum avaliar a relevância dos atributos explicativos com relação à sua capacidade de predição do atributo alvo, a fim de determinar quais variáveis explicativas são relevantes para prever determinado fenômeno e eliminar aquelas que forem irrelevantes (HAN; PEI; KAMBER, 2011). Além disso, também são tomados para se evitar o sobre-ajuste do modelo, também conhecido pelo termo em inglês, *overfitting*, que ocorre quando o modelo ajustado se especializa demais aos dados de treinamento. Nesse caso, o modelo funciona muito bem para prever os dados que foram usados no treinamento, mas tem resultado insatisfatório ao lidar com novas observações que ele venha a receber, ou seja, o modelo não faz uma boa generalização (MATLOFF, 2017).

4.3 Estruturas para Regressão

4.3.1 Árvore de Decisão

O primeiro modelo baseado em árvore foi o *Automatic Interaction Detection* proposto por Morgan e Sonquist (1963). Desde então, diversos outros já foram propostos (LOH, 2014). Esses modelos são relativamente simples, fáceis de explicar e amplamente utilizados. Note que, uma vez que a árvore tenha sido criada, o processo de classificação de uma nova observação pode ser comparado à tomada de decisões sequenciais baseada nas variáveis independentes, ou em regras do tipo se-então (BISHOP, 2006).

Esse tipo de estrutura pode ser utilizada tanto para predizer classes discretas (TAN, 2018) quanto para efetuar regressão de valores contínuos (HASTIE et al., 2005), mas é mais eficiente ao modelar classes discretas que não possuam relação direta de ordem ou hierarquia (TAN, 2018).

Para esse projeto optou-se por utilizar a árvore conhecida como *CART*, a *Classification And Regression Trees*. A árvore *CART* considera todo o espaço de características formado pelas variáveis independentes e particiona esse espaço com linhas paralelas aos eixos coordenados, associando cada região formada durante a execução do algoritmo a uma classe ou valor constante (HASTIE et al., 2005). A Figura 1 mostra uma possível partição de um espaço composto por duas variáveis:

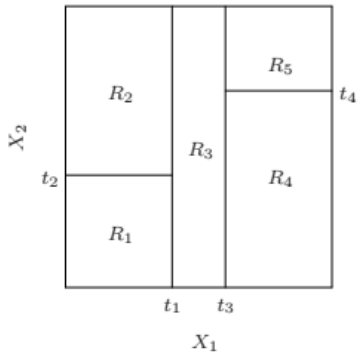

Figura 1 – Exemplo de espaço de características dividido pela árvore de decisão. Autor: (HASTIE et al., 2005)

Em alto nível, o algoritmo CART realiza as seguintes etapas (BREIMAN et al., 1984)

```
1  Inicia com um único nó contendo todas as observações;
2  if Critério de Parada Atingido then
3  |   Determina que o nó é terminal;
4  |   Define uma classe para todas as observações naquele nó;
5  else
6  |   Baseado no nó, encontra a melhor divisão para cada preditor;
7  |   Encontra a melhor divisão para aquele nó, dentre as melhores divisões;
8  |   Divide as observações em dois nós conforme a melhor divisão;
9  |   Aplica o algoritmo da árvore a cada um dos nós;
10 end
11 Retorna a árvore completa;
```

Algorithm 1: Algorítimo da Árvore de Decisão

O critério de parada pode ser definido de acordo com a estratégia de "poda". Por exemplo, na ausência de poda, o critério de parada pode ser definido exclusivamente como "nó puro", ou então pode ser quando cada observação estiver sozinha em um nó. Nesse último caso, a árvore construída pode obter 100% de acurácia nos dados de treinamento, mas provavelmente sofrerá de sobre-ajuste (AGGARWAL, 2015). Por isso, normalmente é definido algum outro critério, como altura máxima da árvore, ou só realizar uma divisão se a variação de algum critério, como por exemplo a soma dos quadrados, for melhor do que algum limite estabelecido. Note que existem outras estratégias de "poda", como aquelas realizadas depois da construção da árvore completa, mas essas estratégias não serão abordadas (HASTIE et al., 2005).

Enquanto o critério de parada não é atingido, o algoritmo age de forma recursiva e busca a próxima melhor divisão possível para a árvore. Ele faz isso comparando as

melhores divisões de cada um dos possíveis preditores, e escolhendo o melhor dos preditores. O critério para definir qual divisão é a melhor difere para as árvores de classificação e de regressão. No caso de classificadores, utiliza-se uma função de impureza que retorna um valor que é máximo quando as observações nos nós estão igualmente distribuídas entre diversas classes e mínima quando cada nó só contém elementos de uma única classe. Assim, a árvore de classificação tenta obter os nós com a menor impureza possível. Já as arvores de regressão tentam reduzir ao máximo o erro quadrático a cada nova divisão (BREIMAN et al., 1984).

4.3.1.1 Critérios de Divisão

No caso das árvores de classificação, é possível usar o ganho de informações como a função de impureza que determina o melhor corte em cada atributo. O ganho de informação é um critério baseado na variação da entropia causada por uma divisão. Seja $E(S) = \sum_{j=1}^{k} p_j * log_2(p_j)$ a entropia de um nó de uma árvore com k classes, onde p_j representa a fração de objetos da classe j naquele nó, o ganho de informação é igual a $E(S) - Entropia - Divisao(S \rightarrow S_1, \ldots, S_r)$, onde $Entropia - Divisao(S \rightarrow S_1, \ldots, S_r)$ é a média ponderada das entropias de S_1 a S_r (AGGARWAL, 2015). Ou seja:

$$G.I = E(S) - \sum_{j=1}^{k} \frac{N_j}{N} E(S_j) \tag{4.1}$$

Onde $E(S_j)$ é a entropia do nó j criado pela divisão, N_j é o número de elementos nesse nó, e N é o número de elementos no nó original, S (TAN, 2018).

Já no caso das árvores de regressão, o critério utilizado é o da redução do erro quadrático. Nele, compara-se o valor de cada observação de um nó, com o valor médio das observações daquele nó, e eleva-se o valor ao quadrado. Esse é o erro quadrático daquele nó, que será representado por $R(t)$. Após realizar uma divisão, calcula-se os erro quadráticos de cada divisão, e subtraí-se esses erros de $R(t)$. Essa será a redução do erro quadrático causada em t pela divisão s, $\Delta R(s,t)$, que deverá ser maximizada (BREIMAN et al., 1984).

Seja $R(t)$ o erro quadrático de um nó t com N_t observações, x uma observação daquele nó, e $y(x)$ o valor daquela observação, $R(t)$ é calculado da seguinte forma (BREIMAN et al., 1984):

$$\bar{y}(t) = \frac{\sum_{x \in t} y(x)}{N_t} \tag{4.2}$$

$$R(t) = \sum_{x \in t} (y(x) - \bar{y}(t))^2 \tag{4.3}$$

O cálculo da variação do erro quadrático, $\delta R(s,t)$, para qualquer divisão s de t em t_R e t_L (BREIMAN et al., 1984):

$$\Delta R(s,t) = R(t) - R(t_R) - R(t_L) \tag{4.4}$$

4.3.1.2 Regra de Divisão dos Atributos

Note que a busca pela melhor divisão em um atributo deve ser realizada de maneira diferente para atributos categóricos e contínuos. Os atributos categóricos podem ser divididos de forma binária, em que um nó tem todos os elementos de um conjunto de categorias, e o outro fica com todos os elementos que não pertencem àquele conjunto, ou então a árvore pode criar um nó novo para cada categoria. O mesmo acontece com atributos contínuos, que podem se separados binariamente entre valores superiores e inferiores a uma referência, ou em múltiplas categorias de valores entre diversos cortes (TAN, 2018).

O algoritmo *CART* sempre efetua divisões binárias, tanto para as variáveis contínuas quanto categóricas. A fim de encontrar a melhor divisão das variáveis categóricas, ele avalia todas as $2^c - 1$ possibilidades, onde c é o número de categorias daquele atributo. Já para as variáveis contínuas, ele busca o melhor valor para separar a variável, entre seu mínimo e máximo (BREIMAN et al., 1984). Como existe um número finito de valores a considerar, é possível determinar o melhor corte testando todos os valores, mas existem abordagens que podem ser mais rápidas, como baseadas em intervalos equidistantes (AGGARWAL, 2015). Outra maneira é ordenar as ocorrências conforme o atributo avaliado e considerar como candidatos a melhor divisão os valores que estão entre dois registros de classes diferentes(TAN, 2018).

4.3.1.3 Definição dos Nós Terminais

Quando o critério de parada for atingido para todos os nós, o crescimento da árvore termina, mas ela ainda pode sofrer um processo de pós poda para melhorar sua capacidade de generalização (TAN, 2018). Para cada nó terminal é atribuído um valor, no algoritmo *CART*: para árvores classificadoras o valor daquele nó é o da classe que possui o maior números de instancias naquele objeto, e para árvores regressoras esse valor é igual à média dos valores de todas as instâncias naquele nó (BREIMAN et al., 1984).

4.3.1.4 Utilização da árvore para previsões

É trivial utilizar uma árvore concluída: para cada registro sob análise, basta percorrê-la desde o primeiro nó aplicando os critérios de decisão para encontrar o próximo nó, até atingir um nó terminal. O valor retornado para o objeto será o valor atribuído a esse nó terminal (AGGARWAL, 2015).

4.3.2 Modelos Lineares Generalizados

4.3.2.1 Modelos Lineares

Os modelos lineares são extremamente simples, e continuam sendo muito utilizados para tarefas de regressão e classificação. Para regressão, ou seja, para a predição de variáveis contínuas, a regressão linear é um dos modelos mais simples e mais utilizados. Já para classificação, ou seja, para a predição de valores discretos, pode ser utilizada a técnica chamada de regressão logística (JAMES et al., 2013).

A regressão linear tenta encontrar uma relação linear entre uma variável alvo e uma ou mais variáveis independentes. Essa relação possui a forma

$$y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_j X_j \tag{4.5}$$

Onde y é a variável alvo, X_j são as variáveis independentes e $\beta_j são o coeficientes que devem ser estimados pel$

Já a regressão logística, em sua forma mais simples, é usada para lidar com problemas de classificação com apenas duas classes (AGGARWAL, 2015), e encontra um valor relacionado à probabilidade de uma observação pertencer à determinada classe, através da função logística (JAMES et al., 2013):

$$p(X) = \frac{^{\beta_0+\beta_1 X_1+\beta_2 X_2+\cdots+\beta_j X_j}}{1+^{\beta_0+\beta_1 X_1+\beta_2 X_2+\cdots+\beta_j X_j}} \tag{4.6}$$

Nessa equação, X_j são as variáveis independentes e β_j são coeficientes que devem ser estimados pelo modelo.

A Equação 4.6 pode ser manipulada para apresentar a seguinte forma (JAMES et al., 2013):

$$log(\frac{p(X)}{1-p(X)}) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_j X_j \tag{4.7}$$

Note que o lado esquerdo da equação é chamado de *logit* (JAMES et al., 2013).

4.3.2.2 Generalização

Os Modelos Lineares Generalizados foram propostos por Nelder e Wedderburn (1972) e são uma forma de expansão dos modelos mais simples, como a regressão linear e a regressão logística. Enquanto os modelos lineares comuns possuem uma determinada distribuição de valores fixa, essa distribuição pode ser inadequada para alguns fenômenos. Por exemplo, é possível supor que o número de sinistros de determinado condutor seguirá uma distribuição de Poisson, e o valor desses sinistros seguirá uma distribuição Gamma (KAAS et al., 2008). Embora a regressão linear simples não seja capaz de lidar com essas distribuições, ambos os fenômenos podem ser modelados com o uso de Modelos Lineares Generalizados. Note, no entanto, que é necessário possuir conhecimento a priori

sobre a distribuição dos dados, o que nem sempre é possível. (NELDER; WEDDERBURN, 1972).

Ainda assim, os modelos lineares generalizados são uma maneira elegante de lidar com diferentes fenômenos, principalmente quando comparado com os modelos lineares comuns. Esses modelos também permitem aplicar transformações no atributo alvo, ampliando a capacidade de separação do mesmo, que pode tratar variáveis que sejam linearmente separáveis em outras escalas, como a logarítmica (KAAS et al., 2008). As funções que transformam o atributo alvo de uma escala para outra são chamadas de funções de ligação. Note que o modelo de uma *GLM* pode ser representado da seguinte forma (GOLDBURD; KHARE; TEVET, 2016):

$$g(x_i) = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} \tag{4.8}$$

Onde β_p é o peso do atributo x_{ip}. Exemplos de funções de ligação, $g(\mu)$, são a função identidade, $(\mu) = \mu$, a função log-linear, $g(\mu) = log(\mu)$ (GOLDBURD; KHARE; TEVET, 2016), e a função logit $g(\mu) = log(\frac{\mu}{1-\mu})$ (AGGARWAL, 2015).

4.3.3 Redes Neurais de Função de Base Radial

As redes neurais artificiais são estruturas projetadas de forma a imitar o funcionamento de um cérebro biológico. Nelas, diversas unidades de processamento, chamadas de neurônios, são conectadas. O ajuste dos pesos sinápticos entre essas conexões configuram o aprendizado da rede. Uma das grandes vantagens desses modelos é sua capacidade de obter respostas não lineares, sem fazer suposições sobre a distribuição final dos resultados do modelo. (HAYKIN, 2007).

As redes de função de base radial são um tipo de rede neural artificial capaz de projetar os dados de entrada em um espaço de alta dimensionalidade a fim de que a tarefa de classificação seja realizada de forma mais simples. Sua estrutura é composta de 3 camadas (HAYKIN, 2007).

Através dessa estrutura, para cada um dos M centróides da camada oculta, uma função kernel, $h_m(\chi)$ é aplicada sobre os dados de entrada χ, onde $m = 1, 2, \ldots, M$. Em seguida, esses dados que foram transformados da dimensão original, N, para a maior, M, são combinados linearmente a fim de realizar a regressão ou classificação (HASTIE et al., 2005):

$$f_\theta(\chi) = \sum_{m=1}^{M} \theta_m h_m(\chi) \tag{4.9}$$

onde $f_\theta(\chi)$ é o valor retornado pela combinação linear, M é o número de neurônios na camada oculta, $k_m(\chi)$ é o valor que representa a entrada x com relação ao kernel

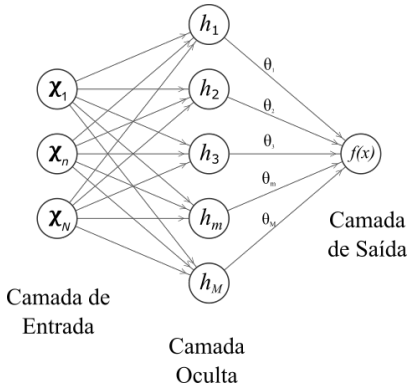

Figura 2 – Arquitetura de uma rede RBF clássica. Autor: Próprio

escolhido quando aplicado ao m-ézimo neurônio, e θ_m é o peso associado a esse neurônio na combinação linear final. (HASTIE et al., 2005)

Em resumo, a arquitetura da rede neural de função radial tradicionalmente é composta por três camadas, como mostrado na Figura 2, e suas responsabilidades são, de acordo com Haykin (2007):

- Camada de Entrada: é a camada que recebe os dados que serão processados pela rede. Possui N nós, um para cada atributo x_n da entrada.
- Camada Oculta: essa única camada oculta aplica uma transformação não linear, dada pelo Kernel centralizado em cada centróide, ampliando assim o espaço da entrada para o espaço oculto, na maioria dos casos de alta dimensionalidade, e possui um total de M nós, um para cada centróide h_m.
- Camada de Saída: Essa camada realiza a combinação linear das saídas da Camada Oculta, usando os pesos θ_m. Para problemas de regressão ou classificação binária, há um único nó de saída.

A escolha de uma camada de saída linear pode ser justificada pois a camada oculta transformou os dados para um espaço de alta dimensionalidade e, ainda conforme Haykin (2007, p. 287), "um problema complexo, de classificação de padrões dispostos não linearmente em um espaço de alta dimensionalidade tem uma probabilidade maior de ser linearmente separável que em um espaço de baixa dimensão".

O processo de criação de uma rede neural de função radial passa por determinar algumas das características que fazem parte de sua estrutura. É necessário definir o número de entradas, χ, que normalmente é o número de atributos originais; e o número de neurônios, M, que existirão na camada oculta. Dada a família da função kernel $h(\chi)$ escolhida para fazer a transformação da camada oculta, alguns parâmetros também devem ser definidos (IDRI; ZAKRANI; ZAHI, 2010):

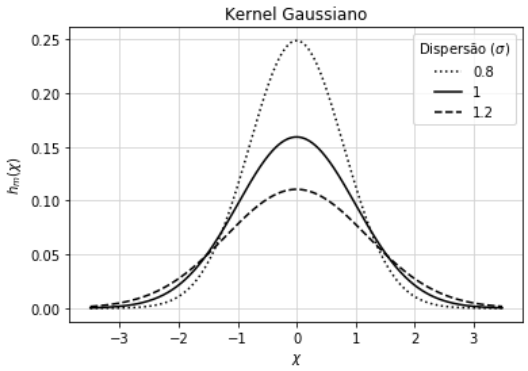

Figura 3 – Efeitos da dispersão no Kernel Gaussiano. Origem: Autor

1. Quais serão os centróides c_m, $m \in 1, 2, \ldots, M$ que servirão de base para cada uma das funções kernel h_m.
2. Para o kernel gaussiano, também é necessário obter o parâmetro de dispersão σ_m.
3. Os pesos θ_m que serão aplicados na combinação linear da camada de saída.

4.3.3.1 Kernels

Dentre os diversos tipos de kernels que podem ser adotados nas redes neurais de função de base radial, o mais utilizado é o Kernel Gaussiano, dado pela fórmula a seguir (HAYKIN, 2007):

$$h_m(\chi) = \frac{1}{(2\pi)} exp(-\frac{||\chi - c_m||^2}{2\sigma_m^2}) \tag{4.10}$$

O valor de saída para um kernel $h_m(\chi)$ depende do centro c_m e da dispersão σ utilizada, na figura 3 podemos observar o efeito da dispersão σ_m sobre $h_m(\chi)$:

Embora esse trabalho não aborde outros kernels, existem diversos outros que poderiam ser utilizados como, por exemplo, o kernel de Epanechnikov (SCOTT, 2015).

Os parâmetros dos *kernels*, nesse caso os centros c_m e as dispersões σ_m podem ser obtidos através de uma etapa de agrupamento através do algoritmo K-Médias.

4.3.3.2 Escolha dos centros e dispersões: K-Médias

A escolha dos pontos que serão utilizados como centro das funções de base radial pode ser realizada de diferentes formas. (HASTIE et al., 2005). Ning et al. (2018) usa o

algoritmo de agrupamento K-Médias, que tem essa forma:

1 Inicialize um vetor c com k objetos aleatórios da amostra;
2 **do**
3 | Compute a distância entre cada ponto x_i da base e cada centro c_p, $p = 1, 2, 3...k$;
4 | Associe o ponto ao cluster ϑ_p, $p = 1, 2, 3, ...k$ associado ao centro c_p.;
5 | Ajuste o novo centro c_p como a média dos valores de todos os pontos associados ao cluster ϑ_p, $p = 1, 2, 3, ...k$. Use o vetor c como centro das funções de base radial.
6 **while** *algum* c_p *mudar*;
7

Algorithm 2: Algoritmo K-Médias para determinação do centro das funções de base radial

Note que, além dos centros, também devemos definir as dispersões usadas para cada função de kernel. Se esse valor for muito pequeno, os valores dos kernels convergirão para zero muito rapidamente, e se forem muito grandes, pode haver muita sobreposição. Existem diversas estratégias que podem ser utilizadas para determinar esse valor (IDRI; ZAKRANI; ZAHI, 2010). Nesse trabalho, usaremos as dispersões dentro de cada um dos agrupamentos criados pelo K-Médias como valor para a dispersão, uma das estratégias citadas por Benoudjit e Verleysen (2003, p. 5).

4.3.3.3 Ajuste dos Pesos Lineares

4.3.3.3.1 Mínimos Quadrados

Para ajustar os pesos lineares, é possível utilizar o método dos mínimos quadrados. Nesse método, buscam-se os melhores parâmetros $\theta = \{\theta_1, \theta_2, \ldots, \theta_m\}$ que minimizam a soma do erro quadrático: (TAN, 2018)

$$C = SSE = \sum_{i=1}^{N} [y_i - f_\theta(\chi^i)]^2 \tag{4.11}$$

A essa equação também damos o nome de função custo. Uma forma de minimizar a função custo é através da Equação Normal (TAN, 2018).

4.3.3.3.2 Solução pela Equação Normal

Note que, para a observação $\chi^i = \{\chi_1^i, \chi_2^i, \ldots, \chi_p^i\}$, o valor previsto, $\hat{y}_i = f_\theta(\chi^i)$, pode ser calculado como (ORR et al., 1996):

$$\hat{y}_i = f_\theta(\chi^i) = \sum_{j=1}^{m} \theta_j * h_j(\chi_j^i) \tag{4.12}$$

Agora, considere a seguinte matriz:

$$H = \begin{bmatrix} h_1(x_1) & h_2(x_1) & \dots & h_m(x_1) \\ h_1(x_2) & h_2(x_2) & \dots & h_m(x_2) \\ \vdots & \vdots & \vdots & \vdots \\ h_1(x_p) & h_2(x_p) & \dots & h_m(x_p) \end{bmatrix} \tag{4.13}$$

Dessa forma, o vetor resposta previsto para todas as observações, $\hat{y}$, pode ser escrito como:

$$\hat{y} = \theta H \tag{4.14}$$

A fim de calcular o valor de θ_j que minimiza o erro da função custo representada pela equação 4.11, podemos igualar sua derivada a zero:

$$\frac{\partial C}{\partial \theta_j} = 2 \sum_{i=1}^{p} (f(x_i) - \hat{y}_i) \frac{\partial f}{\partial \theta_j}(x_i) = 0 \tag{4.15}$$

Com alguma manipulação algébrica, minimizar essas derivadas resulta num conjunto de equações lineares com m pesos distintos, que podem ser representadas pela equação matricial abaixo (ORR et al., 1996).

$$\theta = (H^T H)^{-1} H^T \hat{y} \tag{4.16}$$

Assim, através da resolução dessa equação, podemos encontrar o vetor de pesos $\theta = \theta_1, \theta_2, \dots, \theta_m$ associado m kernels. Esse método de resolução para o problema dos mínimos quadrados é chamado de método normal (ORR et al., 1996).

4.4 Métodos de Avaliação do Modelo

4.4.1 Avaliação do Modelo

Conforme Tan (2018), é possível avaliar e comparar o desempenho de modelos de várias formas. Para classificadores, utilizam-se quatro métricas fundamentais(verdadeiros positivos, verdadeiros negativos, falsos positivos e falsos negativos) e medidas derivadas delas (HAN; PEI; KAMBER, 2011). Já para os regressores, podem ser feitas análises dos resíduos, ou então, para regressões lineares, pode ser calculada qualidade do ajuste, em inglês, *goodness of fit* (TAN, 2018). Ainda, para modelos que assumem alguma distribuição, a medida conhecida como *Log-Likelihood* pode ser utilizada (GOLDBURD; KHARE; TEVET, 2016).

4.4.2 Avaliação de Classificadores

Em se tratando de classificadores, Han, Pei e Kamber (2011) afirmam que existem quatro métricas fundamentais das quais derivam muitas dessas avaliações. Essas métricas se baseiam na comparação entre os resultados previstos pelo modelo e a classificação real da observação:

- **Verdadeiros Positivos**: Corresponde à quantidade de observações em que o modelo prevê que o objeto pertence à classe, e o objeto realmente pertence àquela classe. Em outras palavras, a quantidade de vezes que o modelo previu corretamente que um objeto pertence àquela classe.
- **Verdadeiros Negativos**: Corresponde à quantidade de observações em que o modelo prevê que um objeto não pertence à classe e o objeto realmente não pertence àquela classe. Em outras palavras, o modelo previu corretamente que o objeto não pertence àquela classe.
- **Falsos Positivos**: Corresponde à quantidade de observações em que o modelo prevê que o objeto pertence à classe, mas o objeto não pertence àquela classe. Ou seja, o modelo errou ao prever que aquele objeto pertencia à classe.
- **Falsos Negativos**: Corresponde à quantidade de observações em que o modelo prevê que o objeto não pertence à classe, mas o objeto pertence à classe. Ou seja, o número de vezes em que o modelo previu erroneamente que o objeto não pertence à classe.

A partir desas métricas, podem ser calculadas: Acurácia, Taxa de Erro, Sensibilidade, Especificidade, Precisão, dentre outros. (HAN; PEI; KAMBER, 2011)

4.4.2.1 Acurácia

Também chamada de taxa de reconhecimento, a acurácia calcula a porcentagem de observações corretamente classificadas, e é dada por $\frac{VP+VN}{P+N}$, onde VP é a quantidade de Verdadeiros Positivos, VN é a quantidade de Verdadeiros Negativos, P a quantidade de observações com classe real Positivas, ou seja, que pertencem à classe, e N a quantidade de observações com classe real Negativa, ou seja, que não pertencem à classe. (HAN; PEI; KAMBER, 2011)

4.4.2.2 Taxa de Erro

Calculada por $\frac{FP+FN}{P+N}$, em que FP é a quantidade de Falsos Positivos, FN a quantidade de Falsos Negativos e, assim como na acurácia, P e N são a quantidade de

observações com classe real Positiva e Negativa, respectivamente. A taxa de erro é o complemento da acurácia, o que significa que $TaxaDeErros + Acurácia = 100\%$. (HAN; PEI; KAMBER, 2011)

4.4.2.3 Sensibilidade ou Revocação

A Sensibilidade, também chamada de Revocação, é calculada apenas por $\frac{VP}{P}$, ou seja, a razão entre a quantidade de Verdadeiros Positivos(VP) e o total de Positivos(P). (HAN; PEI; KAMBER, 2011)

4.4.2.4 Especificidade

Analogamente à Sensibilidade, a especificidade é a razão entre a quantidade de Verdadeiros Negativos(VN) sobre o total de objetos Negativos(N), ou seja $\frac{VN}{N}$. (HAN; PEI; KAMBER, 2011)

4.4.2.5 Precisão

A precisão é a razão entre a quantidade de observações corretamente classificadas como verdadeiras(VP) sobre a quantidade de observações classificadas como verdadeiras(o que inclui as observações incorretamente classificadas como Verdadeiras, os Falsos Positivos). Ou seja $\frac{VP}{VP+FP}$. (HAN; PEI; KAMBER, 2011)

4.4.2.6 Acurácia Balanceada

Em problemas desbalanceados, utilizar a acurácia pode mascarar um problema: podemos ter um classificador enviesado muito bom em acertar a classe dominante, mas mal em acertar a classe mais rara, e teríamos uma acurácia alta. Brodersen et al. (2010) defende o uso da acurácia balanceada nesses casos, que é calculada por $\frac{\frac{VP}{P}+\frac{VN}{N}}{2}$. Caso o classificador seja igualmente bom em adivinhar ambas as classes, Verdadeiro Positivo(VP) e Verdadeiro Negativo(VN), então essa acurácia será igual à acurácia convencional. Mas se houver algum viés em errar alguma das classes, então a acurácia balanceada será penalizada. (BRODERSEN et al., 2010)

4.4.2.7 Medida F

A medida F são uma maneira de combinar precisão e revocação numa única medida, e são definidas como a média harmônica entre essas duas medidas (HAN; PEI; KAMBER, 2011):

$$F = \frac{2 \times precisao \times revocação}{precisao + revocação} \tag{4.17}$$

4.4.3 Avaliação de Regressores

Uma maneira muito comum de avaliar um regressor linear é através do uso do coeficiente de determinação, R^2, no entanto, esse coeficiente é apropriado apenas no caso de modelos lineares. Para modelos não lineares, a simples soma do erro quadrático pode ser utilizada como medida de erro (AGGARWAL, 2015).

Como visto na equação 4.11, a soma do erro quadrático é igual à soma das diferenças, elevadas ao quadrado, entre os valores previstos e os valores reais de cada observação, elevadas ao quadrado. Seja $\hat{y}_i$ o valor previsto para a observação i, e y_i o valor real daquela observação, a soma do erro quadrático, SSE, é (TAN, 2018):

$$SSE = \sum_{i=1}^{N} [y_i - \hat{y}_i]^2 \tag{4.18}$$

4.4.4 Outras formas de avaliação

Além dessas, existem diversas outras formas de se avaliar um modelo, mas que não são relevantes para esse trabalho, como por exemplo a velocidade, robustez e escalabilidade do modelo. (HAN; PEI; KAMBER, 2011)

4.4.5 Separação dos Dados para Treinamento

É necessário usar observações diferentes no treinamento e na avaliação do modelo, caso contrário os resultados serão muito otimistas e não retratarão o real desempenho do modelo para padrões ainda não observados (AGGARWAL, 2015) .

4.4.5.1 Hold-Out

Nesse método, uma parte dos dados, escolhida aleatoriamente, é separada do restante, formando dois conjuntos distintos: o conjunto de treinamento e o conjunto de testes. Uma das limitações desse método é que os conjuntos não são independentes entre si. Dessa forma se houver uma classe super-representada num dos conjuntos, ela estará sub-representada no outro, e vice-versa. É possível repetir esse método algumas vezes, para tentar mitigar esse problema, mas ainda assim não há controle sobre a quantidade de vezes que uma observação será usada no teste ou no treinamento. (TAN, 2018)

4.4.5.2 Validação Cruzada ou K-Fold

Ao invés de repetir o método de hold-out, outra alternativa é utilizar o método de Validação Cruzada, também conhecido como *Cross Validation*. A validação cruzada garante que cada observação será usada uma única vez para testes, e que todas serão utilizadas um mesmo número de vezes para o treinamento (TAN, 2018). Ela funciona da seguinte forma: os dados são divididos em k subconjuntos de mesmo tamanho. Cada um dos seguimentos será utilizado uma vez como conjunto de teste, e para cada um desses testes o modelo será treinado com os outros conjuntos. (AGGARWAL, 2015)

5 Metodologia

Descrição da metodologia

5.1 Base de Dados

Nesse trabalho, os algoritmos serão analisados conforme seu desempenho numa base de dados sobre sinistros de carros, a *Car Insurance Claim Data*[1] , disponível no Kaggle.

5.1.1 Dados Utilizados

Essa base de dados é composta por 27 colunas e 10302 linhas. No total, a tabela possui 1 coluna de data, 15 colunas numéricas e 11 colunas categóricas. A tabela 1 mostra um resumo das colunas disponíveis, bem como o tipo de dados existentes. Dessas colunas, foram consideradas como variáveis alvo as variáveis número 21, 24 e 26: *CLM_FREQ*, *CLM_AMT* e *CLAIM_FLAG*, respectivamente. As outras variáveis foram consideradas variáveis explicativas.

5.1.2 Descrição dos Dados

Não há um dicionário oficial dos dados, mas foi possível inferir o significado de algumas das variáveis pelo seu nome e pelo seu domínio.

- ID: um número único que identifica cada linha.
- KIDSDRIV: não foi possível inferir exatamente o significado, embora o domínio de 0 a 4 indique que esteja relacionado à quantidade de crianças.
- BIRTH: Data de nascimento.
- AGE: Idade.
- HOMEKIDS: não foi possível inferir exatamente, mas o domínio de 0 a 5 indica que está relacionado à quantidade de crianças.
- YOJ: não foi possível inferir o significado dessa variável, embora seja um número entre 0 a 23.

[1] Disponível no Endereço: https://www.kaggle.com/xiaomengsun/car-insurance-claim-data

Tabela 1 – Colunas presentes na Base de Dados

	Variável	Tipo	Exemplos
01	ID	numérica	63581743
02	KIDSDRIV	numérica	0, 1, 2, 3, 4
03	BIRTH	data	16MAR39
04	AGE	numérica	60.0
05	HOMEKIDS	numérica	0, 1, 2, 3, 4, 5
06	YOJ	numérica	11.0
07	INCOME	numérica	$67,349
08	PARENT1	categórica	'No', 'Yes'
09	HOME_VAL	numérica	$0
10	MSTATUS	categórica	'z_No', 'Yes'
11	GENDER	categórica	'M', 'z_F'
12	EDUCATION	categórica	'PhD', 'z_High School'...
13	OCCUPATION	categórica	'Professional', 'Manager'...
14	TRAVTIME	numérica	14
15	CAR_USE	categórica	'Private', 'Commercial'
16	BLUEBOOK	numérica	$14,230
17	TIF	numérica	11
18	CAR_TYPE	categórica	'Minivan', 'Van'...
19	RED_CAR	categórica	'yes', 'no'
20	OLDCLAIM	numérica	$4,461
21	CLM_FREQ	numérica	0, 1, 2, 3, 4, 5
22	REVOKED	categórica	'No', 'Yes'
23	MVR_PTS	numérica	0, 1, 2, 3...
24	CLM_AMT	numérica	$0
25	CAR_AGE	numérica	18.0
26	CLAIM_FLAG	categórica	0 ou 1
27	URBANICITY	categórica	'Highly Urban/ Urban', 'z_Highly Rural/ Rural'

- INCOME: Renda.
- PARENT1: Uma marcação (sim/não) relacionada a pais.
- HOME_VAL: O valor da residencia.
- MSTATUS: Não soi possível inferir exatamente o significado, mas se trata de uma marcação do tipo sim/não.
- GENDER: O gênero.
- EDUCATION: Grau de escolaridade
- OCCUPATION: O tipo de profissão.
- TRAVTIME: variável relacionada ao tempo de viagem.

- CAR_USE: Indica se o carro é usado para fins comerciais ou não.
- BLUEBOOK: Não foi possível inferir o significado, mas é uma variável monetária.
- TIF: não foi possível inferir o significado, mas é uma variável numérica que varia de 0 a 25.
- CAR_TYPE: o tipo do veículo.
- RED_CAR: pode estar relacionada à cor do carro. É uma marcação do tipo sim/não.
- OLDCLAIM: indica se já ouve algum outro sinistro para aquele motorista.
- REVOKED: não foi possível inferir o significado dessa variável.
- MVR_PTS: não foi possível inferir o significado dessa variável. Seu domínio é de 0 a 13.
- CAR_AGE: A idade do veículo
- URBANICITY: Se o veículo está em região urbana ou rural.
- CLM_FREQ: A quantidade de sinistros.
- CLM_AMT: O valor pago por sinistros.
- CLAIM_FLAG: Uma marcação que indica se houve algum sinistro.

5.2 Ferramentas Utilizadas

Esse trabalho utilizou as seguintes ferramentas para realizar a leitura, tratamento e análise da base de dados:

- Python (versão 3.8.1)
- Pandas(versão 1.0.0), Numpy(versão 1.18.1), Scikit-Learn(versão 0.22.1) e StatsModels(versão 0.11.0)

Além disso, o Excel foi utilizado para visualizar os resultados das análises.

5.2.1 Modelos

Os modelos utilizados foram:

- Árvore de Decisão: Implementado pelo scikit-learn, através das classes sklearn.tree.DecisionTreeClass e sklearn.tree.DecisionTreeRegressor.
- GLM: Implementado pelo statsmodels na classe statsmodels.api.GLM.
- RBF: implementação própria, disponível como anexo e no repositório incluir repositório do github onde o código da árvore está

5.3 Preparação e Modelagem dos Dados

Primeiramente, o csv com os dados foi salvo na máquina local, a fim de ser carregado na memória principal através da biblioteca pandas.

Com a leitura dos dados, realizou-se uma análise superficial dos dados. Em seguida, os dados foram tratados a fim de corrigir alguns problemas observados.

Os dados tratados foram separados em duas partes: 30% das observações para validação, e 70% para treino e definição dos modelos. Os dados de treino foram utilizados para treinar e testar os três algoritmos através da metodologia de validação cruzada com 10 pastas. Essa mesma metodologia também foi utilizada para procurar pelos melhores parâmetros no algoritmo, e verificar o efeito de algumas transformações sobre os dados. Por último, a melhor versão de cada conjunto de (transformações nos dados + parâmetros) para cada algoritmo foi utilizada nos 30% dos dados separados para validação.

5.3.1 Análise Preliminar dos Dados

Nessa primeira etapa, carregou-se a base de dados da memória e verificou-se o número de linhas e colunas da base de dados, além de confirmar os nomes das colunas. Também obteve-se informações sobre cada uma das colunas, como tipo, valores máximos, mínimos, nulos e únicos. Os valores únicos foram obtidos apenas para as variáveis categóricas ou as numéricas com menos de 10 valores distintos.

Após essa etapa, constatou-se que:

- As variáveis relacionadas à dinheiro haviam sido carregadas como texto.
- Apenas sete colunas apresentavam valores nulos.
- Não havia uma coerência quanto às variáveis categóricas binárias. Por exemplo, algumas categorias possuiam os valores "yes"e "no", todas minúsculas, enquanto outras possuiam valores 1/0 e outras ainda "Yes"e "No", com a primeira letra maíuscula.

- Algumas categorias apresentavam um prefixo "z_" (no máximo uma categoria por variável categórica)
- Algumas das variáveis com nomes compostos dividem as palavras usando o símbolo "_", como em CLM_AMT, enquanto outras simplesmente unem as duas palavras, como em TRAVTIME.

5.3.2 Preparação dos dados

Enquanto algumas das características observadas na etapa anterior não influenciariam diretamente os algoritmos, alguns dos pontos levantados tiveram que ser tratados. A etapa de correção dos dados se deu da seguinte forma:

1. Remoção de Colunas;
2. Conversão de Variáveis Monetárias para Números;
3. Substituição de Valores Nulos;
4. Homogeneização de Marcadores(*Flags*);

Além disso, a seguinte transformação teve que ser feita pois o algoritmo da árvore de decisão fornecido pelo scikit-learn não consegue tratar dados categóricos:

- Separação de variáveis categóricas em diversas variáveis binárias, utilizando o *one-hot encoding* do scikit-learn.

5.3.2.1 Remoção de Colunas

Uma vez que o processo de seleção de variáveis não é o foco desse trabalho, foram removidas apenas duas variáveis óbvias: *ID* e *BIRTH*. ID foi removida por se tratar de uma identificação, portanto totalmente irrelevante para a análise, e BIRTH foi removida pois já existe a coluna AGE, e o tratamento das datas em BIRTH seria redundante e trabalhoso.

5.3.2.2 Conversão de Variáveis Monetárias para Números

As variáveis *INCOME*, *HOME_VAL*, *BLUEBOOK*, *OLDCLAIM* e *CLM_AMT* contém informação numérica, mas foram lidas como dinheiro pois apresentam o prefixo $. A fim de convertê-las para números, foi necessário apenas remover esse prefixo e fazer uma conversão de tipo para float.

5.3.2.3 Substituição de Valores Nulos

Aproximadamente 1/4 de todos os registros, apresentavam algum valor nulo em uma dessas variáveis: *AGE, YOJ, INCOME, HOME_VAL, OCCUPATION ou CAR_AGE*. A fim de tratar essas variáveis, optou-se por substituí-los pelo o valor da média, no caso das variáveis numéricas, e criar uma nova categoria, "Desconhecido"para a variável *OCCUPATION*.

5.3.2.4 Homogeneização de Marcadores(*Flags*)

As variáveis PARENT1, MSTATUS, RED_CAR, REVOKED e CLAIM_FLAG, eram todas do tipo sim/não, mas cada uma representava essa informação de uma maneira diferente. Para a homogeneização, substituiu-se todos os textos que representavam a informação "Sim" por 1, e os que representavam a informação "Não" por 0.

5.3.2.5 Separação de variáveis categóricas em diversas variáveis binárias

O algoritmo da árvore CART implementado na versão do pacote scikit-learn utilizada nesse projeto não é capaz de lidar com variáveis categóricas, embora o algoritmo original fosse capaz de fazê-lo. Por isso, as variáveis categóricas GENDER, EDUCATION, OCCUPATION, CAR_USE, CAR_TYPE e URBANICITY foram convertidas para variáveis numéricas do tipo 1/0, através do uso da função One-Hot-Encoder do SciKit-Learn.

As variáveis binárias, GENDER, CAR_USE e URBANICITY foram representadas apenas por um de seus significados e o valor de 1/0.

Já as variáveis com várias categorias foram separadas em diversas variáveis, de forma que no máximo uma delas terá o valor 1 e as outras terão o valor 0. Uma das categorias foi removida, de forma que se todos os valores derivados de uma categoria forem zero, então a categoria daquela observação é igual à categoria removida.

A tabela 2 mostra as transformações sofridas pelas colunas. O Valor Base é o valor para o qual não há coluna representada. Ou seja, se o valor original da obervação era o Valor Base, então todas as novas colunas terão valor zero:

5.3.3 Conjuntos de Teste E Treinamento

30% teste, 70% treinamento. Elaborar

Tabela 2 – Conversão de Colunas Categóricas

Coluna Original	Valor Base	Nova Coluna
GENDER	M	x0_z_F
EDUCATION	‘‘<High School’’	x1_Bachelors
EDUCATION	‘‘<High School’’	x1_Masters
EDUCATION	‘‘<High School’’	x1_PhD
EDUCATION	‘‘<High School’’	x1_z_High School
OCCUPATION	Clerical	x2_Desconhecido
OCCUPATION	Clerical	x2_Doctor
OCCUPATION	Clerical	x2_Home Maker
OCCUPATION	Clerical	x2_Lawyer
OCCUPATION	Clerical	x2_Manager
OCCUPATION	Clerical	x2_Professional
OCCUPATION	Clerical	x2_Student
OCCUPATION	Clerical	x2_z_Blue Collar
CAR_USE	Commercial	x3_Private
CAR_TYPE	Minivan	x4_Panel Truck
CAR_TYPE	Minivan	x4_Pickup
CAR_TYPE	Minivan	x4_Sports Car
CAR_TYPE	Minivan	x4_Van
CAR_TYPE	Minivan	x4_z_SUV
URBANICITY	‘‘Highly Urban/ Urban’’	x5_z_Highly Rural/ Rural

5.3.4 Otimização

Elaborar

5.3.4.1 Análise Preliminar - Validação Cruzada

Elaborar - kfolds = 10

5.3.4.2 Busca de Parâmetros

Elaborar

5.3.4.3 Transformação dos Atributos

A normalização de dados foi efetuada numa etapa posterior, a fim de verificar como afetaria o desempenho dos algoritmos. Ela foi realizada usando a função *scale* do *sklearn.preprocessing*.

Foi aplicada normalização nas variáveis KIDSDRIV, AGE, HOMEKIDS, YOJ, INCOME, HOME_VAL, TRAVTIME, BLUEBOOK, TIF, OLDCLAIM, MVR_PTS e

CAR_AGE.

5.3.5 Avaliação dos Resultados

5.3.5.1 Tarefa de Classificação - CLAIM_FLAG

5.3.5.2 Tarefa de Regressão - CLM_AMT

5.3.5.3 Tarefa de Regressão - CLM_FREQ

6 Cronograma

O projeto de graduação em computação foi concebido de forma a compreender as seguintes etapas, que nos serão úteis para melhor acompanhar o andamento do plano de trabalho:

1. Pesquisa bibliográfica e estudo teórico;
    a) Estudo de mineração de dados; pré-processamento;
    b) Estudo dos atributos (base de dados);
    c) Estudo de métodos: RBF (+ uso de kernels diferentes), kNN, árvore;
    d) Métodos de Treinamento: k-fold, hold-out;
2. Implementação;
    a) pré-processamento;
    b) seleção de atributos;
    c) Adequação dos dados, Conjunto de Validação, Conjunto de Teste;
    d) RBF, kNN e árvore;
3. Análise dos Resultados;
4. Redação do Relatório PGC I;
5. Redação do Relatório PGC II;
6. Redação do Relatório PGC III;

A duração estimada de cada etapa é apresentada no cronograma da Tab. 3.

Tabela 3 – Cronograma do PGC.

Etapa	Plano de Trabalho para 3 Quadrimestres											
	1° Quad.				2° Quad.				3° Quad.			
1.1	x	x										
1.2	x	x										
1.3		x	x	x								
1.4				x								
2.1					x							
2.2					x	x						
2.3						x	x					
2.4						x	x	x				
3								x	x	x	x	
4		x	x	x								
5						x	x	x				
6										x	x	x

Referências

AGGARWAL, C. C. *Data mining: the textbook*. [S.l.]: Springer, 2015. Citado 7 vezes nas páginas 10, 11, 12, 13, 14, 21 e 22.

BENOUDJIT, N.; VERLEYSEN, M. On the kernel widths in radial-basis function networks. *Neural Processing Letters*, v. 18, p. 139–154, 10 2003. Citado na página 17.

BISHOP, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006. ISBN 0387310738. Citado na página 9.

BREIMAN, L. et al. *Classification and Regression Trees*. Monterey, CA: Wadsworth and Brooks, 1984. Citado 3 vezes nas páginas 10, 11 e 12.

BRODERSEN, K. H. et al. The balanced accuracy and its posterior distribution. In: IEEE. *2010 20th International Conference on Pattern Recognition*. [S.l.], 2010. p. 3121–3124. Citado na página 20.

FERREIRA, P. P. *Modelos de precificação e ruína para seguros de curto prazo*. [S.l.]: Funenseg, 2005. Citado na página 8.

GOLDBURD, M.; KHARE, A.; TEVET, D. Generalized linear models for insurance rating. *Casualty Actuarial Society, CAS Monographs Series*, n. 5, 2016. Citado 2 vezes nas páginas 14 e 18.

HAN, J.; PEI, J.; KAMBER, M. *Data mining: concepts and techniques*. [S.l.]: Elsevier, 2011. Citado 6 vezes nas páginas 8, 9, 18, 19, 20 e 21.

HASTIE, T. et al. The elements of statistical learning: data mining, inference and prediction. *The Mathematical Intelligencer*, Springer, v. 27, n. 2, p. 83–85, 2005. Citado 5 vezes nas páginas 9, 10, 14, 15 e 16.

HAYKIN, S. *Redes neurais: princípios e prática*. [S.l.]: Bookman Editora, 2007. Citado 3 vezes nas páginas 14, 15 e 16.

IDRI, A.; ZAKRANI, A.; ZAHI, A. Design of radial basis function neural networks for software effort estimation. *IJCSI International Journal of Computer Science Issues*, v. 7, n. 4, 2010. Citado 2 vezes nas páginas 15 e 17.

JAMES, G. et al. *An introduction to statistical learning*. [S.l.]: Springer, 2013. v. 112. Citado na página 13.

KAAS, R. et al. *Modern actuarial risk theory: using R*. [S.l.]: Springer Science & Business Media, 2008. v. 128. Citado 3 vezes nas páginas 8, 13 e 14.

LOH, W.-Y. Fifty years of classification and regression trees. *International Statistical Review*, v. 82, n. 3, p. 329–348, 2014. Disponível em: ⟨https://onlinelibrary.wiley.com/doi/abs/10.1111/insr.12016⟩. Citado na página 9.

MATLOFF, N. *Statistical regression and classification: from linear models to machine learning*. [S.l.]: Chapman and Hall/CRC, 2017. Citado na página 9.

MORGAN, J. N.; SONQUIST, J. A. Problems in the analysis of survey data, and a proposal. *Journal of the American statistical association*, Taylor & Francis Group, v. 58, n. 302, p. 415–434, 1963. Citado na página 9.

NELDER, J. A.; WEDDERBURN, R. W. Generalized linear models. *Journal of the Royal Statistical Society: Series A (General)*, Wiley Online Library, v. 135, n. 3, p. 370–384, 1972. Citado 2 vezes nas páginas 13 e 14.

NING, Y. et al. *An Optimal Radial Basis Function Neural Network Enhanced Adaptive Robust Kalman Filter for GNSS/INS Integrated Systems in Complex Urban Areas*. MDPI AG, 2018. 3091 p. Disponível em: ⟨http://dx.doi.org/10.3390/s18093091⟩. Citado na página 16.

ORR, M. J. et al. *Introduction to radial basis function networks*. [S.l.]: Technical Report, Center for Cognitive Science, University of Edinburgh, 1996. Citado 2 vezes nas páginas 17 e 18.

RAJARAMAN, A.; ULLMAN, J. D. *Mining of massive datasets*. [S.l.]: Cambridge University Press, 2011. Citado na página 8.

SCOTT, D. W. *Multivariate density estimation: theory, practice, and visualization*. [S.l.]: John Wiley & Sons, 2015. Citado na página 16.

TAN, P.-N. *Introduction to data mining*. [S.l.]: Pearson Education India, 2018. Citado 8 vezes nas páginas 8, 9, 11, 12, 17, 18, 21 e 22.

YEO, A. C. et al. Clustering technique for risk classification and prediction of claim costs in the automobile insurance industry. *Intelligent Systems in Accounting, Finance & Management*, Wiley Online Library, v. 10, n. 1, p. 39–50, 2001. Citado na página 8.