Universidade de Brasília
Departamento de Estatística

Interpretação de redes neurais

Davi Guerra Alves

Projeto apresentado para o Departamento de Estatística da Universidade de Brasília como parte dos requisitos necessários para obtenção do grau de Bacharel em Estatística.

Brasília
2023

Davi Guerra Alves

Interpretação de redes neurais

Orientador(a): Thais Carvalho Valadares Rodrigues

Projeto apresentado para o Departamento de Estatística da Universidade de Brasília como parte dos requisitos necessários para obtenção do grau de Bacharel em Estatística.

Brasília
2022

Lista de Tabelas

1 Dedicatória

2 Agradecimentos

3 Resumo

Sumário

4 Lista de figuras

5 Lista de tabelas

6 Lista de quadros

7 Introdução

As redes neurais são modelos matemáticos que, unidos às técnicas computacionais, visam tentar reproduzir o funcionamento da estrutura neural presente no ser humano, buscando, assim, realizar tarefas complexas, como o reconhecimento de padrões, identificação de imagens, processamento de linguagem natural, etc. No entanto, apesar de sua eficácia em muitas aplicações, as redes neurais podem ficar muito complexas conforme sua arquitetura cresce, sendo consideradas como "caixas pretas", devido à sua complexidade e falta de transparência.

Por isso, a interpretação de redes neurais é uma área cada vez mais essencial, pois busca entender como esses modelos tomam decisões e quais fatores influenciam suas saídas. Entender o porquê uma rede neural tomou tal decisão é importante em diversas áreas, como a área da saúde, em diagnósticos médicos, e na área bancária, analisando um risco de crédito.

Uma das técnicas mais promissoras para a interpretação de redes neurais é o SHAP (Shapley Additive Explanations), que foi introduzido em 2017 (**?**). O SHAP é uma técnica de interpretação que fornece explicações locais e globais para as saídas da rede neural. Ele é baseado no conceito matemático de valor de Shapley (**?**), que atribui uma contribuição de importância para cada recurso de entrada na saída da rede neural.

Portanto, esse trabalho tem como objetivo explorar a técnica SHAP para a interpretação de redes neurais e sua aplicação em diversas áreas, pois, ao compreender como as redes neurais funcionam, e quais são os recursos mais importantes para suas decisões, será possível tornar o método mais confiável e transparentes para os usuários.

8 Referencial teórico

8.1 Conjunto de dados

O conjunto de dados utilizado envolverá informações do mercado financeiro, tendo como análise principal o banco de dados *Loan Data for Dummy Bank*, que contém informações de um tipo de empréstimo chamado *peer-to-peer lending*, onde pessoas que possuem um montante financeiro para aplicações, encontram empresas que precisam de dinheiro. Nesse conjunto de dados existem algumas variáveis que podem ser alvo de predições, como é o caso das variáveis *loan condition cat*, que categoriza se o empréstimo é vantajoso ou não para o banco e também a variável *interest rate*, que indica uma taxa de risco que banco define para um determinado empréstimo. Para explicar essas variáveis existem algumas variáveis explicativas que podem ser úteis como: *installment*, que é o valor da parcela mensal do empréstimo, *recoveries*, valor total de recuperações feitas pelo banco após um empréstimo ter sido inadimplente, *region*, região geográfica do país onde o tomador do empréstimo reside e dentre outras.

Há a tentativa de utilizar dados do Banco do Brasil, mas o mesmo está em processo de liberação.

8.2 Regressão linear

O modelo de regressão linear é um modelo estatístico que relaciona uma variável dependente (ou variável resposta) com uma ou mais variáveis independentes (ou variáveis explicativas). Quando essa relação envolve apenas duas variáveis, variável resposta e variável explicativa, o modelo é chamado de modelo de regressão simples, e quando envolve a relação entre uma variável dependente e múltiplas variáveis independentes, é chamado de modelo de regressão múltipla **?**.

Para o modelo de regressão simples, temos a seguinte fórmula:

$$y_i = \beta_0 + x_i\beta_1 + \varepsilon_i, \tag{8.2.1}$$

onde β_0 é chamado de intercepto, β_1 é o coeficiente linear, x_i é a entrada da variável explicativa e ϵ_i é o erro para cada estimação de y_i.

Já para o modelo de regressão múltipla, temos o seguinte resultado:

$$y_i = \beta_0 + x_{i1}\beta_1 + x_{i2}\beta_2 + ... + x_{ik}\beta_k + \varepsilon_i, \tag{8.2.2}$$

onde para cada variável explicativa existe um parâmetro β fazendo sua ponderação no resultado de y_i.

A forma matricial do cálculo de todos os y_{is}, sendo $i = 1, ...n$, dada da seguinte maneira:

$$Y = X\beta + \varepsilon, \tag{8.2.3}$$

onde Y é um vetor coluna com os n valores de y_i, X é uma matriz com as k variáveis explicativas, e ε é um vetor de dimensão n dos erro, ou seja,

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1k} \\ 1 & x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{nk} \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix} + \begin{bmatrix} \varepsilon_0 \\ \varepsilon_1 \\ \vdots \\ \varepsilon_n \end{bmatrix}. \tag{8.2.4}$$

Para fazer a estimativa dos parâmetros, é utilizado o método dos mínimos quadrados ordinários(MQO), que busca minimizar as distâncias quadradas entre a variável dependente observada e a variável prevista pelo modelo. Essa estimação é representada por $S(\beta)$ e pode ser descrita da seguinte maneira:

$$S(\beta) = \sum_{i=1}^{N} (y_i - x'\beta)^2 = (Y - X\beta)'(Y - X\beta). \tag{8.2.5}$$

Minimizando a Equação **??**, temos que o estimador para β é calculado por:

$$\hat{\beta} = (X'X)^{-1} XY. \tag{8.2.6}$$

Dado o estimador $\hat{\beta}$, podemos estimar $E(y_i)$ como $\hat{y}_i = x_i\hat{\beta}$ consequentemente a estimativa de ϵ_i é dada por $\hat{\epsilon}_i = \hat{y}_i - x_i\hat{\beta}$, este é chamado de resíduo.

Para se utilizar uma regressão linear é preciso observar algumas suposições:

1. Os erros ε_i seguem uma distribuição Normal, com variância constante σ^2, sendo representada por: $\varepsilon_i \sim N(0, \sigma^2)$.

2. As variáveis aleatórias $\varepsilon_1, \varepsilon_2, ..., \varepsilon_n$ são independentes;

3. As variáveis explicativas $X_1, X_2, ..., X_n$ são não correlacionadas – hipótese de ausência de multicolinearidade entre as variáveis explicativas;

4. A relação entre a média da variável resposta e as covariáveis é linear.

O modelo respeitando as suposições apresenta as seguintes propriedades.

- O estimador MQO de β é não viesado:

$$E[\hat{\beta}|X] = \beta. \tag{8.2.7}$$

- O estimador MQO é (multivariado) normalmente distribuído:

$$\hat{\beta}|X \sim N(\beta, V[\hat{\beta}|X]), \tag{8.2.8}$$

com variância $V[\hat{\beta}|X] = \sigma^2(X'X)^{-1}$, sob a hipótese de homocedasticidade e $V[\hat{\beta}|X] = \sigma^2(X'X)^{-1}\ X'\Omega X(X'X)^{-1}$, considerando heterocedasticidade conhecida, onde Ω é a matriz de variância-covariância dos erros. Sob homocedasticidade, a variância V pode ser estimada não viesadamente como

$$\hat{V}(\hat{\beta}|X) = \hat{\sigma^2}(X'X)^{-1}, \tag{8.2.9}$$

com

$$\hat{\sigma^2} = \frac{\hat{\epsilon}'\hat{\epsilon}}{n-k-1}. \tag{8.2.10}$$

8.3 Redes Neurais Artificiais

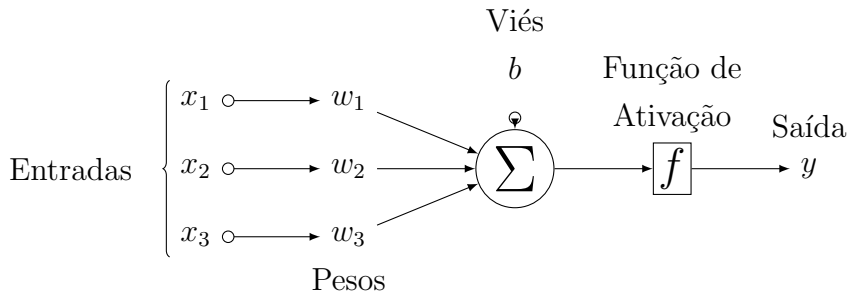
Redes Neurais Artificiais (ou *Deep Learning*) é uma técnica preditiva presente no campo de Inteligência Artificial. As redes neurais tem sido amplamente utilizadas devido ao seu alto poder preditivo e também à flexibilidade de se aplicar esse método em diversos contextos, permitindo ser um modelo com menos restrições que os modelos tradicionais estatísticos.

8.3.1 Neurônio

Uma rede neural tem esse nome devido à tentativa de se reproduzir o comportamento do cérebro humano. Sua arquitetura é composta por um conjunto de unidades denominadas neurônios, e cada neurônio é responsável por receber informações, fazer o tratamento do que foi recebido, e repassar o resultado disso para frente. A Figura **??** ilustra a estrutura de 1 neurônio. Quando as informações x_i entram no neurônio, acontece primeiramente um processo onde é ponderada cada informação que foi recebida, os chamados **pesos**. Logo em seguida ocorre a soma dessa ponderação. Feito isso, é realizado mais um processo de soma, agora adicionando uma informação própria daquele neurônio nesse resultado. Essa informação é chamada de **bias** (ou Viés). Antes desse resultado

ser repassado para outro neurônio, ele passa por uma função que vai definir a natureza daquela informação, chamada de **função de ativação**, retornando assim uma saída y.

Figura 1: Neurônio da Rede Neural

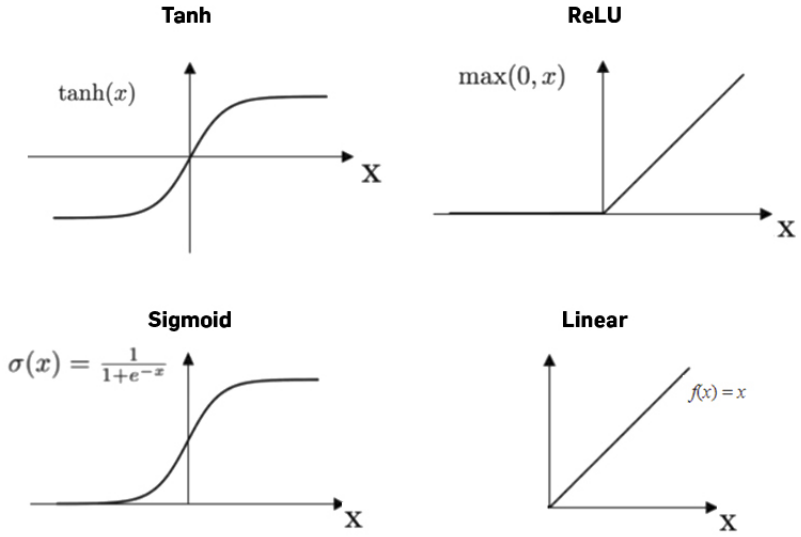


A Figura **??** pode ser representada matematicamente da seguinte maneira:

$$y = f(\beta + \sum_{i=1}^{d_x} w_i x_i) \tag{8.3.1}$$

onde d_x é o número de entradas.

Figura 2: Tipos de Função de Ativação.



Fonte: https://machine-learning.paperspace.com/wiki/activation-function

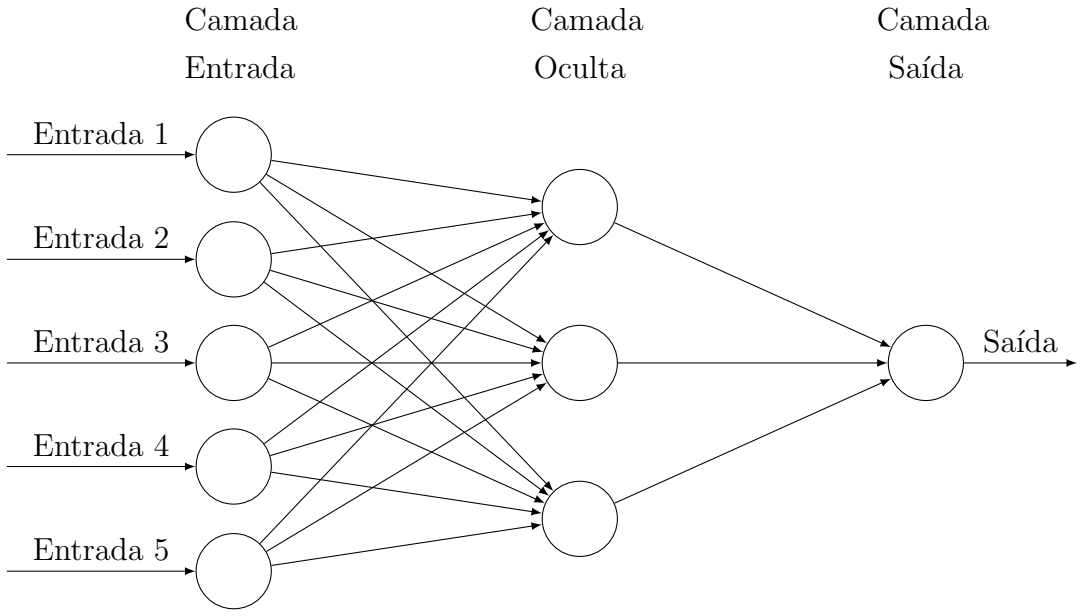
A Figura **??** mostra alguns tipos de funções de ativação que um neurônio pode ser atribuído. Note como a maioria dessas funções restringe o valor de x (nesse caso o valor calculado no neurônio), no caso da função Sigmoide e da Tangente hiperbólica (tanh) limitando o valor do neurônio em um intervalo, a ReLU, que é bastante utilizada, desconsidera os valores negativos e existe a função Linear que basicamente só vai repassar

a informação do neurônio para frente.

8.3.2 Arquitetura

Uma rede neural é estruturada em camadas formadas por um conjunto de neurônios. Conforme ilustrado na Figura **??**, temos as camadas de entrada, as camadas ocultas e a camada de saída. A camada de entrada é o ponto de partida da rede neural, pois é onde as informações das variáveis entram. Logo em seguida encontram-se as camadas ocultas, que são as principais responsáveis por criar redes mais complexas, pois o número de camadas e o número de neurônio dentro dessas camadas podem ser moldados ou adicionados dependendo do objetivo empregado pela rede, conforme ilustrado na Figura **??**. E por fim existe a camada de saída que contém o(s) valor(es) predito(s) pela rede.

Figura 3: Rede Neural com uma camada oculta



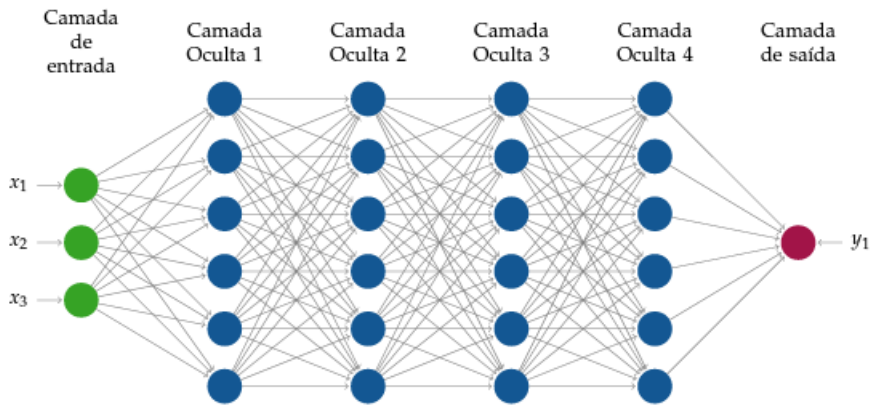


Figura 4: Arquitetura padrão de um rede neural *feedforward* **?**

Note que as Figuras **??** e **??** evidenciam um potencial muito grande de crescimento da rede e naturalmente esse aumento pode acabar gerando um custo computacional elevado quando a rede estiver em treinamento.

8.3.3 *Forward propagation*

O processo *Forward propagation* (ou propagação direta) é o responsável por transmitir as informações, desde a camada de entrada, passando pelas camadas ocultas, até chegar na camada de saída. O *Forward propagation* utiliza da generalização a Equação **??** para cada neurônio presente nas camadas internas da rede neural. Por isso temos que, para cada j-ésimo neurônio, da camada l:

$$z_j^{(l)} = b_j^{(l)} + \sum_{i=1}^{d_{(l-1)}} w_{ij} a_i^{(l-1)}$$

onde:

- w_{ij} é o peso associado à conexão entre o neurônio i na camada $l-1$ e o neurônio i na camada l;
- $a_{(i)}^{(l-1)}$ é a saída do neurônio i na camada anterior $(l-1)$;
- $b_j^{(l)}$ é o viés (bias) associado ao neurônio j na camada l.

Logo em seguida é aplicada uma função de ativação g em $z_i^{(l)}$ que vai ser a responsável por gerar o resultado final $a_i^{(l)}$, do o i-ésimo neurônio na l-ésima camada.

$$a_i^{(l)} = g(z_i^{(l)})$$

Esse processo vai ser realizado camada a camada, sequencialmente. Logo, supondo que uma rede neural tenha l camadas ocultas e, cada camada contendo d_l neurônios, considerando também w_{ij} como o peso presente no i-ésimo neurônio com a j-ésima saída na camada seguinte $(l+1)$, onde $l = 0, ..., H$. Temos que o resultado final da propagação é igual a:

$$f(\mathbf{x}) = \mathbf{a}^{H+1} = g(b_j^{(H+1)} + \sum_{i=1}^{d_H} w_{ij} a_i^H) \tag{8.3.2}$$

Note que a previsão da rede vem diretamente do resultado obtido da última camada oculta, e esse depende da camada que o antecede e assim sucessivamente até chegar na camada de entrada.

8.3.4 Função de perda

Para se obter informações sobre o desempenho do modelo, é escolhida uma função de perda. Uma função bastante utilizada é a do erro do quadrático médio:

$$EQM(f) = \frac{1}{n} \sum_{k=1}^{n} (f(\mathbf{x}_k) - y_k)^2$$

Essa função é uma indicadora do quão longe, em média, os valores preditos estão distantes dos valores reais. Note que o resultado da função f depende exclusivamente dos parâmetros da rede (viés e pesos), por isso, se essa função de perda tende a 0, significa que os parâmetros dessa rede alcançaram um ponto mínimo global. Entretanto, devido a complexidade desse modelo, acabam-se escolhidos pontos locais mínimos, que, dependendo do contexto, acabam satisfazendo o objetivo. A Figura **??** ilustra esse comportamento:

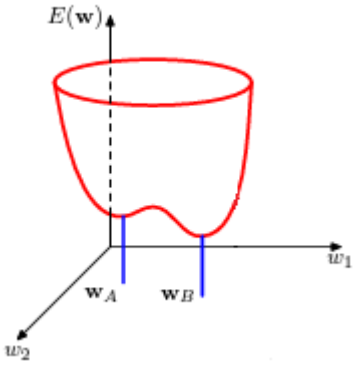


Figura 5: Comportamentos dos pesos em relação à função de perda. O ponto w_A representa um ponto local mínimo e w_B representa um ponto global mínimo. **?**

8.3.5 *Backpropagation*

Como a função de perda está relacionada com os parâmetros (θ) da rede, para se minimizar a função de perda R(θ), é necessário encontrar os valores de θ que resolvam esse problema de otimização. Para fazer isso, é necessário calcular o gradiente de R(θ) em relação à θ **?**,

$$\nabla R(\theta) = \frac{\partial R(\theta)}{\partial \theta} \tag{8.3.3}$$

A rede neural, durante todo o treinamento, aplica esse processo do cálculo do gradiente de R(θ) em relação à θ. Esse é um processo iterativo, com o objetivo de mudar o valor de θ afim de conseguir minimizar a função de perda. Com isso a Equação **??** pode ser descrita nesse processo iterativo como:

$$\nabla R(\theta^m) = \left.\frac{\partial R(\theta)}{\partial \theta}\right|_{\theta=\theta^m},$$

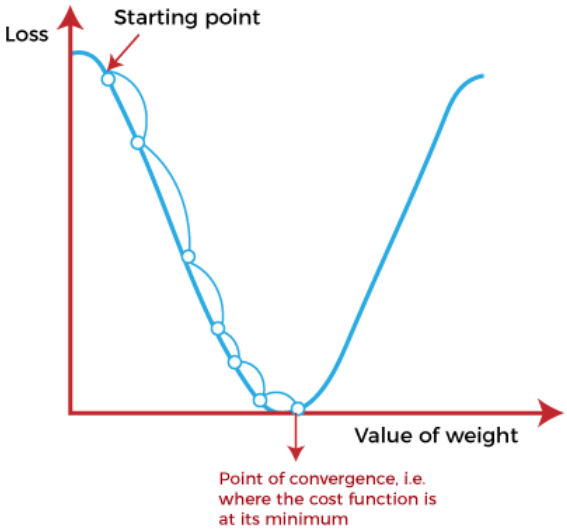
onde $\theta = \theta_m$ significa que o cálculo do gradiente está sendo realizado na iteração m.

E, para conseguir atualizar esse θ, conforme é calculado o gradiente durante as iterações, é utilizada a técnica de gradiente descendente, que pode ser descrita como:

$$\theta^{m+1} \leftarrow \theta^m - \lambda \frac{\partial R(\theta^m)}{\partial \theta^m}$$

sendo λ o parâmetro que vai definir a magnitude de influência da derivada $\frac{\partial R(\theta^m)}{\partial \theta^m}$ em θ^m.

Figura 6: Representação do método do gradiente descendente para a estimação de um parâmetro.



Fonte: https://www.javatpoint.com/gradient-descent-in-machine-learning

A Figura **??** demonstra o processo do gradiente descendente. Os parâmetros são iniciados com algum valor e, conforme ocorre os processos iterativos de aprendizado, o parâmetro converge para um mínimo da função de perda. Note que a distância entre cada ponto é definida pelo λ ou taxa de aprendizado.

Todo esse processo é realizado em cada parâmetro que existe na rede neural. Assim como as informações das variáveis são passadas camada a camada, saindo da camada de entrada, passando pelas camadas ocultas e chegando na camada de saída, visto anteriormente como *Forward propagation*, a informação do resultado da rede na função de perda é passada de forma contrária. O gradiente de cada parâmetro é calculado primeiro nas camadas mais próximas da saída, e essa informação é repassada para trás, chegando até os parâmetros próximos aos da camada de entrada. Esse processo é chamado de *Backpropagation* **?**.

8.4 SHAP

A estrutura de uma rede neural, por mais que proporcione bons resultados, mostra uma deficiência na parte interpretativa. Conhecida por ser uma "caixa-preta"pelo fato de sua estrutura ser muito complexa, existe a necessidade de se entender as predições feitas. Para isso, existem técnicas que abordam o tema de interpretação de modelos de redes neurais e dentro delas existe a técnica SHAP, que através dela é possível entender como as variáveis de entrada influenciam as previsões do modelo, fornecendo *insights* sobre sua lógica e permitindo uma explicação clara e confiável. Isso contribui para a

transparência, confiabilidade e aceitação dos modelos, além de auxiliar na detecção de viéses e discriminação.

8.4.1 Valores de Shapley

Os valores de Shapley foram desenvolvidos por Lloyd Shapley **?** no contexto da teoria de jogos, e essa técnica ganhou força na área de inteligência artificial pela sua capacidade de conseguir interpretar modelos preditivos tidos como "caixa-preta". No método criado por Shapley, existiam uma quantidade de jogadores que exerciam juntos determinada atividade, e o intuito era observar o ganho que um jogador (ou um conjunto de jogadores), obtinha ao ser adicionado para realizar a mesma tarefa, sem a presença do restante do grupo.

Podemos definir $\mathbf{F}$ como o conjunto de jogadores (ou as variáveis explicativas) presentes na atividade, logo $\mathbf{F} = \{1, 2, ..., \mathbf{M}\}$, onde $\mathbf{M}$ é o número de variáveis. Definindo $\mathbf{S}$ como uma coligação do conjunto $\mathbf{F}$ ($\mathbf{S} \subseteq \mathbf{F}$), temos, por exemplo, as seguintes possibilidades de $\mathbf{S}$, quando $\mathbf{M}$ é igual a 3:

$$\{\{\emptyset\}, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}$$

Podemos definir também ν como uma função que vai mapear um conjunto de valores e retornar um número real. Com isso, o retorno de $\nu(\mathbf{S})$ é um número real que pode ser definido como o "trabalho da coligação $\mathbf{S}$ ". Esse valor é equivalente ao total ganho que os jogadores podem obter caso trabalhem juntos em uma determinada coligação.

Para calcular o ganho ao adicionar uma variável i ou a importância do jogador em específico, pode-se calcular o ganho quando é adicionada aquela variável na coligação menos a coligação sem a adição daquela variável, ficando da seguinte maneira:

$$\nu(\mathbf{S} \cup \{i\}) - \nu(\mathbf{S})$$

No exemplo acima, caso queiramos calcular o efeito da coligação $\{3\}$, poderíamos fazer:

$$\textit{Contribuição de } \{3\} = \nu(\{1, 2, 3\}) - \nu(\{1, 2\})$$

Mas suponha que as variáveis (ou jogadores) $\{2\}$ e $\{3\}$ sejam extremamente semelhantes. Quando é calculado o ganho após inserir $\{2\}$ na coligação $\{1,2\}$, é possível notar um aumento substancial, mas quando é adicionado $\{3\}$ na coligação $\{1,2,3\}$, o ganho

obtido é muito pouco. Como as variáveis exercem um papel parecido, o ganho maior ficou sujeito à variável que foi adicionada primeiro na coligação, não necessariamente porque uma é mais importante que a outra. Por isso, para calcular o real ganho da variável {i}, é necessário testar todas as permutações de **F** (conjunto de jogadores) e obter a contribuição de {i} em cada uma delas, para então fazer a média dessas contribuições. Por exemplo, definindo **F** = {1,2,3,4}, suponha que estamos interessados em calcular a contribuição de {3}, logo, podemos obter a seguinte permutação de **F**:

$$[3, 1, 2, 4]$$

Calculando a contribuição de {3}, temos:

$$\nu(\{3\}) - \nu(\emptyset)$$

Outra permutação poderia ser :

$$[2, 4, 3, 1]$$

Calculando a contribuição de {3}, nessa permutação temos:

$$\nu(\textit{coligação de } [2, 4, 3]) - \textit{coligação de } [2, 4])$$

Uma observação deve ser feita: a função ν considera a coligação como argumento, não a permutação. A coligação é um conjunto, com isso a ordem dos elementos não importa, mas a permutação é uma coleção ordenada de elementos. Na permutação do tipo [3,1,2,4], 3 é a primeira variável adicionada e 4 é a última. Por isso, para cada permutação a ordem dos elementos pode mudar a contribuição do total ganho, contudo o total ganho da permutação somente depende dos elementos, não da ordem. Logo:

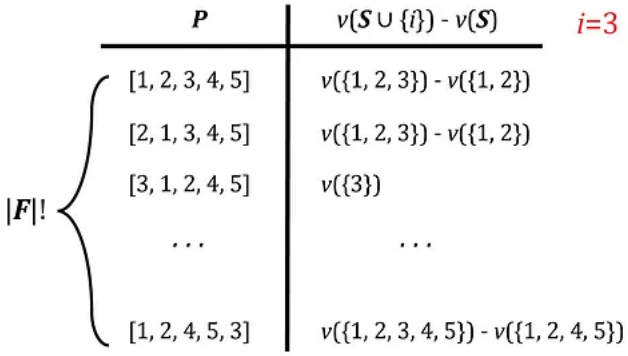
$$\nu(\textit{coligação de } [3, 1, 2, 4]) = \nu(\{1, 4, 2, 3\})$$

Sendo assim, para cada permutação **P**, é preciso primeiro calcular o ganho da coligação das variáveis que foram adicionadas antes de {i}, e esse conjunto pode ser chamado de coligação **S**. Feito isso, agora é preciso calcular o ganho das coligações que são formadas ao adicionar {i} em **S**, e podemos chamar isso de $\mathbf{S} \cup \{i\}$. Com isso, a contribuição da variável {i}, denotada por ϕ_i, é:

$$\phi_i = \frac{1}{|\mathbf{F}|!} \sum_{\mathbf{P}} [\nu(\mathbf{S} \cup \{i\}) - \nu(\mathbf{S})] \tag{8.4.1}$$

O número total de permutações de **F** é |**F**|!. Logo, podemos dividir a soma das contribuições por |**F**|! para encontrar o valor esperado de contribuição de {i}. A Figura **??** mostra como é feito esse calculo para um determinado jogador $\{i\}$.

Figura 7: Ganho do jogador 3 em relação à todas as permutações de jogadores.



$$\phi_i = \frac{1}{|F|!}\sum_{P}(v(S \cup \{i\}) - v(S))$$

Fonte: https://towardsdatascience.com/introduction-to-shap-values-and-their-application-in-machine-learning-8003718e6827

É possível perceber que algumas permutações possuem a mesma contribuição, desde que suas coligações $\mathbf{S} \cup \{i\}$ e **S** sejam as mesmas. Com isso, para reduzir o processo do cálculo de contribuição de cada permutação, pode-se identificar quantas vezes a permutação gerada vai resultar em uma contribuição que seja igual a outra.

Para fazer isso, é necessário descobrir quantas permutações podem ser formadas de cada coligação. Podemos definir $\mathbf{F} - \{i\}$ como o conjunto de todas as variáveis excluindo a variável {i}, e **S** como uma das coligações de $\mathbf{F} - \{i\}$ ($\mathbf{S} \subseteq \mathbf{F}$ - {i}).

Logo, para cada coligação **S** temos |**S**|! possíveis permutações, que corresponde às possibilidades de variáveis e suas respectivas ordens antes de adicionar a variável {i}.

Tendo os conjuntos $\mathbf{S} \cup \{i\}$ e **S** definidos, resta agora achar as possíveis permutações das variáveis restantes. E para saber o valor restante é preciso calcular o tamanho do conjunto gerado por: **F** - ($\mathbf{S} \cup \{i\} + 1$) que basicamente é o que resta das variáveis para completar o conjunto **F**.

A Figura **??** mostra o que acontece quando se escolhe o jogador i, nesse caso $i = 3$. Note que, na linha das coligações, é definido as possíveis coligações de **S**, que seria as permutações dos jogadores 1 e 2, temos a coligação de um único elemento na coluna $\{i\}$, que sempre vai ser o próprio elemento, em seguida a coligação dos jogadores restantes. Na linha das permutações é definida todas as possíveis permutações para **S**, para $\{i\}$ e

para $\mathbf{F} - \mathbf{S} - \{i\}$. E na última linha é representado o tamanho do conjunto formado pela permutação/coligação descrita anteriormente.

Figura 8: Relação entre permutações e coalizões.

Coalitions	***S***	+	$\{i\}$	+	***F*-*S*-$\{i\}$**	= ***F***
	$\{1, 2\}$	+	$\{3\}$	+	$\{4, 5\}$	= $\{1, 2, 3, 4, 5\}$
Permutations	$[1, 2]$ $[2, 1]$	+	$[3]$	+	$[4, 5]$ $[5, 4]$	= $[1, 2, 3, 4, 5]$ = $[1, 2, 3, 5, 4]$
					$[4, 5]$ $[5, 4]$	= $[2, 1, 3, 4, 5]$ = $[2, 1, 3, 5, 4]$
Number of Permutations	$\lvert S\rvert!$	+	1	+	$(\lvert F\rvert\text{-}\lvert S\rvert\text{-}1)!$	= $\lvert S\rvert!(\lvert F\rvert\text{-}\lvert S\rvert\text{-}1)!$

Fonte: https://towardsdatascience.com/introduction-to-shap-values-and-their-application-in-machine-learning-8003718e6827

Com isso, podemos reescrever a Equação **??** da seguinte maneira:

$$\phi_i = \sum_{\mathbf{S} \subseteq \mathbf{F} - \{i\}} \frac{|\mathbf{S}|!(|\mathbf{F}| - |\mathbf{S}| - 1)!}{|\mathbf{F}|!} [\nu(\mathbf{S} \cup \{i\}) - \nu(\mathbf{S})],$$

onde ϕ_i é o valor de shapley para a variável {i}.

8.4.2 Shapley Additive Explanations

Fazendo a relação do valor de Shapley para o SHAP (Shapley Additive Explanations), temos que a função característica ν é equivalente à função $f(x)$ responsável por fazer as predições. E os valores de SHAP são calculados a partir das observações que entram no modelo. Com isso, a fórmula do valor de SHAP, para cada conjunto de observação e variável especificada, se dá por:

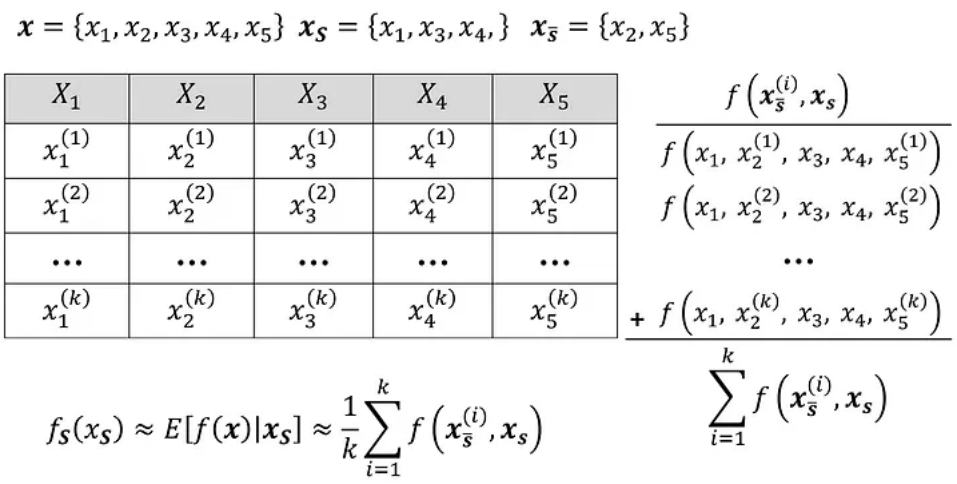
$$\phi_i(f, \mathbf{x}) = \sum_{\mathbf{S} \subseteq \mathbf{F} - \{i\}} \frac{|\mathbf{S}|!(|\mathbf{F}| - |\mathbf{S}| - 1)!}{|\mathbf{F}|!} [f_{\mathbf{S} \cup \{i\}}(\mathbf{x}_{\mathbf{S} \cup \{i\}}) - f_{\mathbf{S}}(\mathbf{x}_{\mathbf{S}})]$$

Perceba que $f_{\mathbf{S}}(\mathbf{x}_S)$ representa o resultado do modelo com somente as variáveis que estão na coligação $\mathbf{S}$, algo que na realidade não é permitido na maioria dos modelos.

Por isso, uma aproximação desse resultado é a seguinte:

$$f_S(\mathbf{x}_S)) \approx E[f(\mathbf{x}|\mathbf{x}_S)] \approx \frac{1}{k}\sum_{i=1}^{k} f(\mathbf{x}_{\bar{S}}^{(i)}, \mathbf{x}_S) \tag{8.4.2}$$

Figura 9: Cálculo de $f_{\mathbf{S}}$, sendo S o conjunto de variáveis X_1, X_3, X_4, dentre as observações de um conjunto de dados.



Fonte: https://towardsdatascience.com/introduction-to-shap-values-and-their-application-in-machine-learning-8003718e6827

A Figura **??** representa, para cada observação do conjunto de dados, a Equação **??**. Neste caso, as variáveis X_1, X_3 e X_4 representam o conjunto **S**, e escolhendo um valor x_1, x_3 e x_4 dessas variáveis, respectivamente, calcula-se f para cada observação do conjunto de dados, travando x_1, x_3 e x_4 na função e utilizando o valor das variáveis complementares, que neste caso são os valores de X_2 e X_5, em suas respectivas observações. Feito isso, é calculada média desses valores que corresponde justamente com o resultado da função $f_{\mathbf{S}}(x_{\mathbf{S}})$.

9 Metodologia

10 Resultados

Análise descritiva

Modelagem da rede neural

Arquitetura

Variações da arquitetura

Melhor modelo de redes neurais

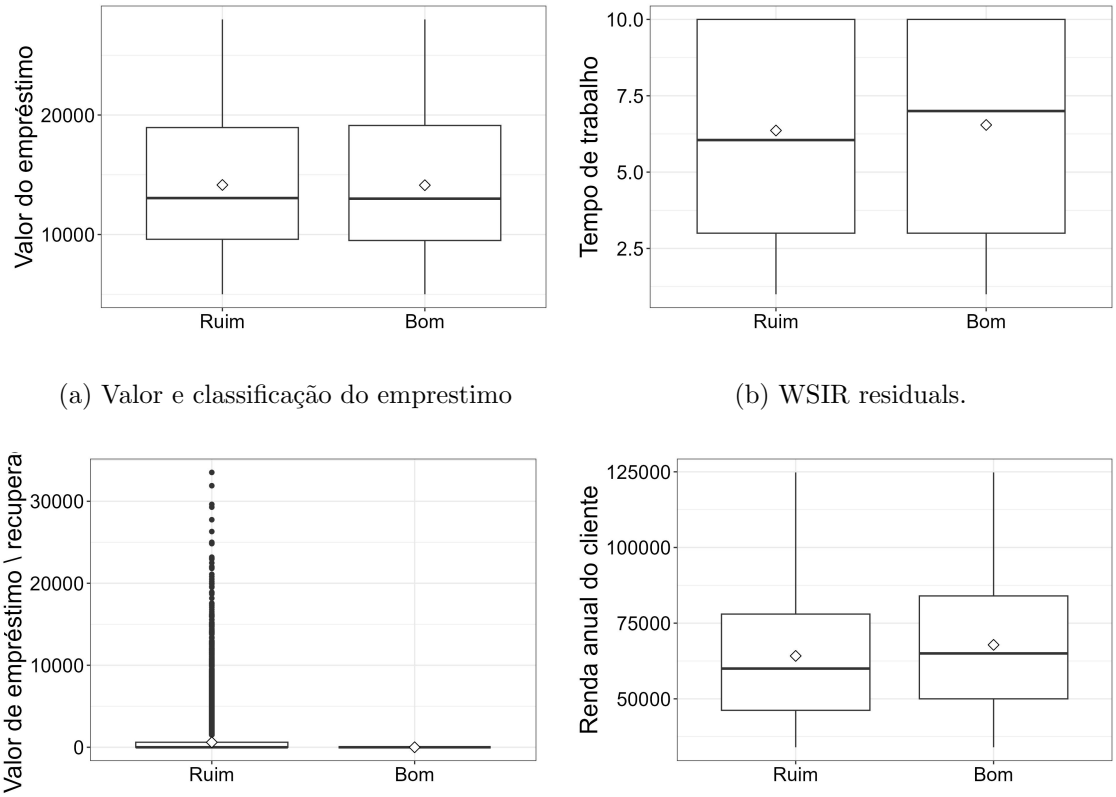
Modelagem da regressão logística

Interpretação de rede neural

Benchmark entre redes neurais e regressão logística

	correlação
emp_length_int	-0.02
annual_inc	-0.03
annual_inc.1	-0.03
loan_amount	0.00
interest_rate	0.18
dti	0.01
total_pymnt	-0.04
total_rec_prncp	-0.10
recoveries	0.39
duracao_emprestimo_dias	0.01

	contingência
home_ownership	0.04
income_category	0.04
application_type	0.01
purpose	0.06
interest_payments	0.14
grade	0.15
installment	0.36
region	0.01



(a) Valor e classificação do emprestimo

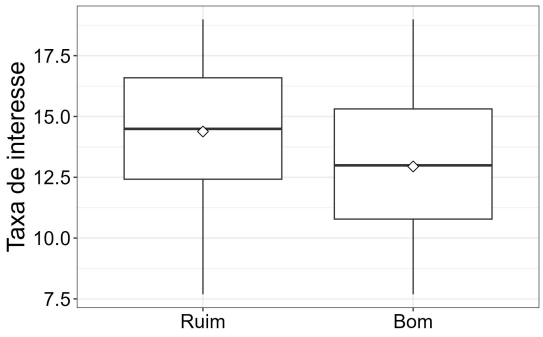
(b) WSIR residuals.

(c) WSIR predictions in the covariate space.

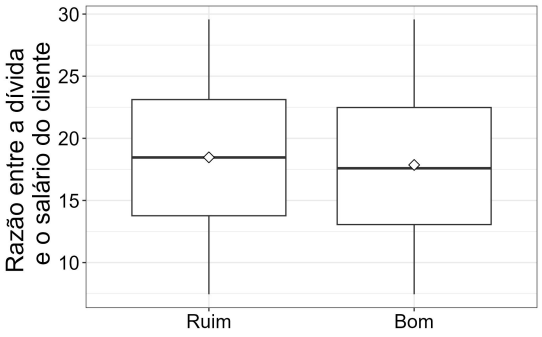
(d) WSIR residuals.

Figura 10: aloalo

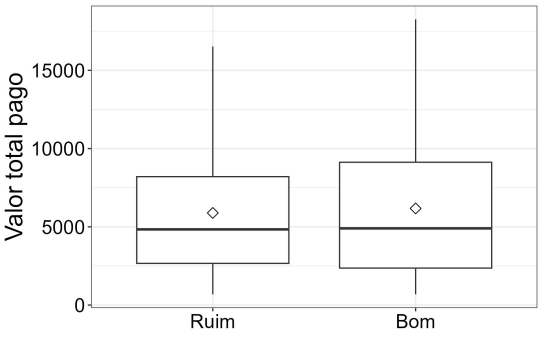
11 Conclusão



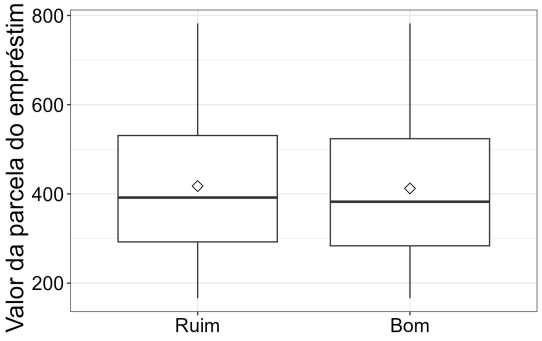
(a) Total de tipos de empréstimo.



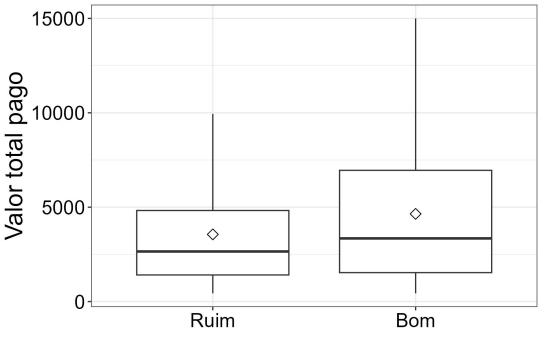
(b) WSIR residuals.



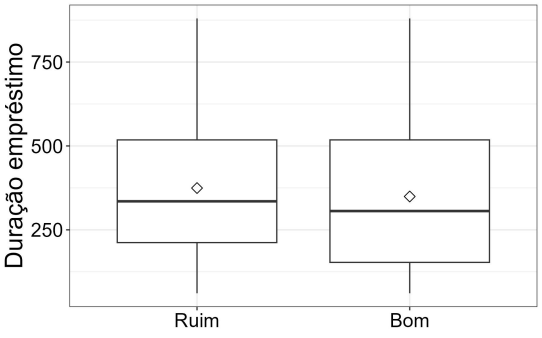
(c) WSIR predictions in the covariate space.



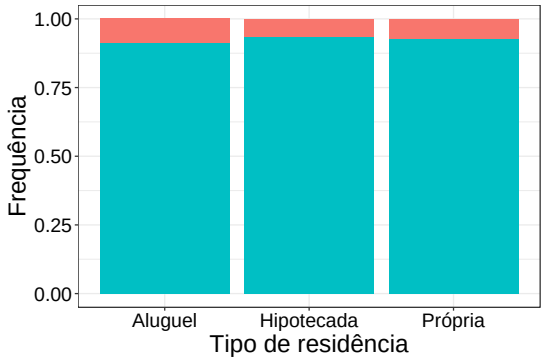
(d) WSIR residuals.



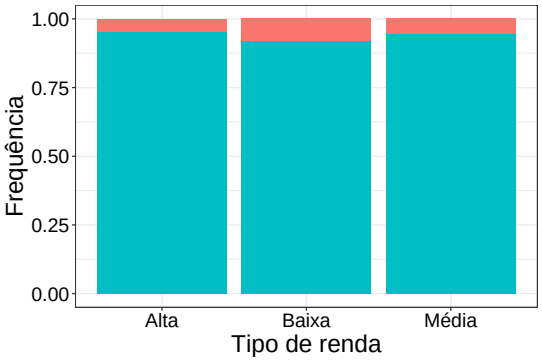
(a) Total de tipos de empréstimo.



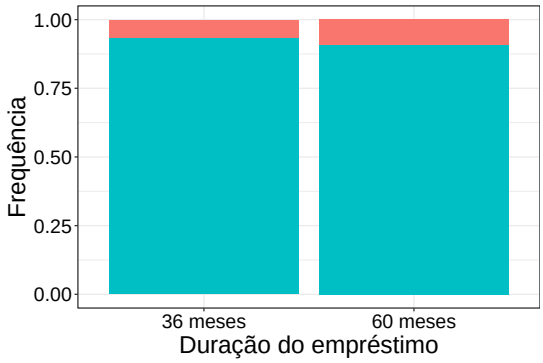
(b) WSIR residuals.



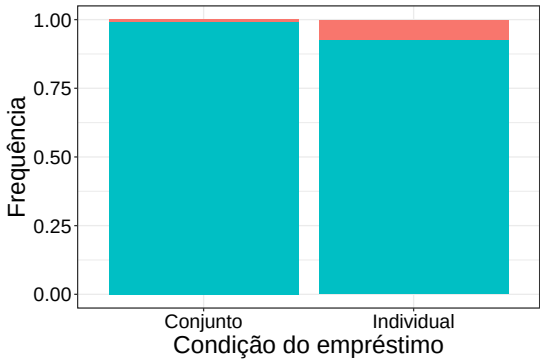
(a) Total de tipos de empréstimo.



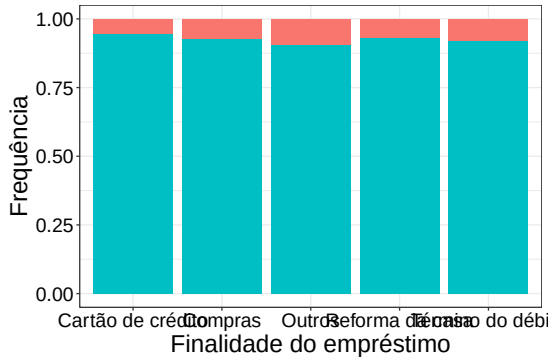
(b) WSIR residuals.



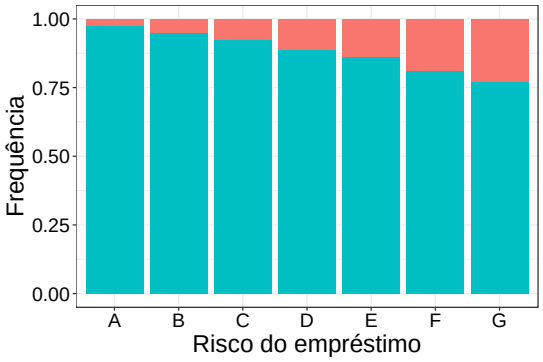
(c) WSIR predictions in the covariate space.



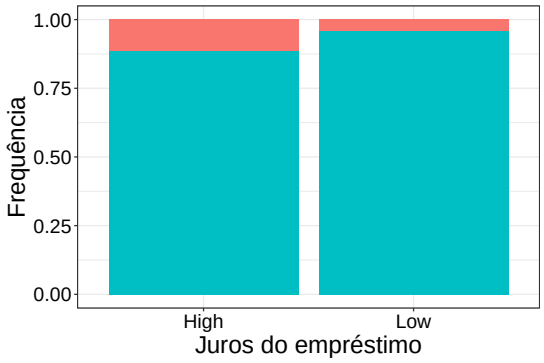
(d) WSIR residuals.



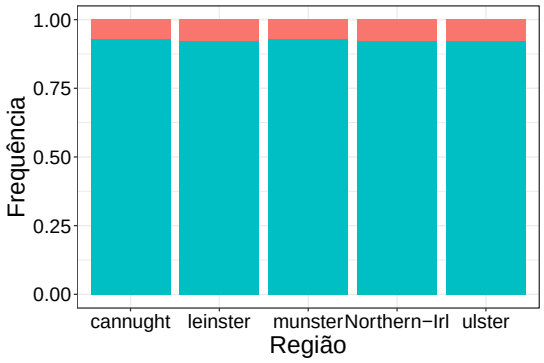
(a) Total de tipos de empréstimo.



(b) WSIR residuals.



(c) WSIR predictions in the covariate space.



(d) WSIR residuals.

12 Anexo